Diseño de un sistema completo de navegación, posicionamiento, captura y representación de mapas, y de los subsistemas de respaldo en el mundo virtual de la E. I. I. I.

Escuela de Ingenierías Industrial, Informática y Aeronáutica

Universidad de León

Tutor D. Fernando Jorge Fraile Fernández

Autor Hermes Alejandro Suárez Ferreras

Este documento cuyo título es «Diseño de un sistema completo de navegación, posicionamiento, captura y representación de mapas, y de los subsistemas de respaldo en el mundo virtual de la E. I. I. I.» constituye el Trabajo Fin de Carrera de Hermes Alejandro Suárez Ferreras, alumno de la Escuela de Ingenierías Industrial, Informática y Aeronáutica de León, con el objetivo de obtener el título de Ingeniero Técnico Industrial en la especialidad de Electrónica, Regulación y Automatismos.

La tutoría de este trabajo ha sido llevada a cabo por D. Fernando Jorge Fraile Fernández, profesor del Departamento de Tecnología Minera, Topográfica y de Estructuras de la Universidad de León.

V.°B.°Oficina Técnica: Tutor:

Fdo.: D. Manuel Castejón Limas Fdo.: D. Fernando Jorge Fraile Fernández

Autor:

Fdo.: Hermes Alejandro Suárez Ferreras

A mis padres:
A ti mamá, a ti papá;
os lo debo todo.
Con todo mi corazón, gracias.

A mi tutor:

Por brindarme la ocasión de participar en un proyecto apasionante, donde técnica, creatividad, ingenio e intuición se presentan como habilidades imprescindibles; por darme la oportunidad de formar parte de un equipo de ingenieros y artistas con un solo límite, la imaginación.

Resumen

En este proyecto se crean todas las herramientas que el usuario de un mundo virtual tridimensional necesita para nunca perderse en él.

En forma de sistemas, cada una de estas se diseña para realizar una función muy concreta, desde capturar los mapas del terreno o posicionar al usuario en ellos, hasta guiarlo cuando quiera alcanzar un destino o transportarlo cuando no quiera recorrer el camino. Junto a ellos se crean también sistemas de respaldo como uno de información u otro de traducción que, junto con la confección e integración en el conjunto de una cuidada y detallada interfaz, permiten al usuario interactuar con el sistema cómoda e intuitivamente, comunicándose con él en su propio idioma.

Se ha hecho especial hincapié en que cada sistema sea fácilmente configurable y ampliable, pues, creado en el seno del Grupo de Trabajo en Mundos Virtuales de la E. I. I. I., flexibilidad y adaptabilidad han de ser cualidades indispensables de este proyecto.

Finalmente, todo el código que sustenta a cada sistema se ha documentado profusamente, garantizando, además de la eficiencia y la optimización, su comprensión.

Abstract

This project creates all the tools that any 3D virtual world user will need to never get lost in it.

Each of these tools is a system designed to perform a very specific task, that goes from capturing maps of the terrain or finding the user's location in them, to providing directions when he wants to get to some place or immediately transporting him when he doesn't want to go all the way there.

Translation and information systems are also created among others, there to support the rest, and along with a detailed and meticulously made interface, allow the user to interact with the system easy and intuitively, and in his own language.

Special emphasis has been placed in making each system easy to configure, customize and expand, for adaptability and flexibility are must-have characteristics of this project, as a part of the «E. I. I. I.» Virtual World Work Team.

Finally, all the programming that consolidate each system has been profusely illustrated with comments, to guarantee, besides efficiency and optimization, that they can be easily understood.

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

Todo en la vida puede considerarse un viaje, desde algo tan habitual como las rutas que recorremos a diario para desplazarnos de un lugar a otro, hasta algo tan abstracto como las decisiones que conducen nuestros día a día, ellas también tienen un punto de partida, un rumbo y un desenlace; cualquier concepto en el que podamos pensar, por abstracto que sea, tiene un origen y tiene un destino, incluso la vida misma es en esencia un largo y emocionante viaje.

A partir de estas premisas podemos intuir la gran importancia que tienen todos aquellos elementos capaces de ayudarnos en las aventuras en las que nos embarcamos, de guiarnos, de apuntarnos el camino, de indicarnos dónde estamos, pues sin ellos, por la naturaleza misma de los viajes, corremos el riesgo de extraviarnos.

Este proyecto, consciente de la necesidad en cualquier ámbito de estos elementos indicadores, aporta todos los sistemas de ayuda que una persona podría necesitar para nunca perder su camino al enfrentarse a algo tan abstracto como un nuevo mundo virtual, uno imaginado, nacido de la mente humana y generado por ordenador, un mundo ideal, pero al que sin embargo la evolución de nuestras vidas y nuestra tecnología nos conduce de forma muy real.

Necesitaremos un sistema de captura de mapas para hacernos con cada remoto rincón y detalle de este nuevo mundo, uno de posicionamiento para predecir nuestro entorno y saber de antemano adónde nos llevan nuestros pasos, otro de ubicación presto siempre a responder con acierto a la pregunta *¿dónde estoy?*, uno de transporte para llevarnos allá donde deseemos cuando lo queramos sin demora, un sistema de navegación guiada para cuando no nos importe disfrutar del camino con la garantía de no perdernos

al recorrerlo, o uno de traducción, porque de nada sirve ninguna indicación si no nos la dan en un idioma que comprendamos. Todos son ejemplos de lo que este proyecto tiene para aportar a un apasionante área como la creación de nuevos mundos virtuales, y todos ellos se describirán con detalle en los capítulos que siguen a esta introducción.

Este es el punto de partida... le deseo un buen viaje.

Capítulo 2

Unity y primeros pasos

Unity es el nombre con el que se presenta la piedra angular de este proyecto, la plataforma de desarrollo y ecosistema que aporta cada una de las piezas que permiten levantar todo un mundo generado por ordenador desde cero.

Figura 2.1: Logotipo de Unity

Compatible con numerosas aplicaciones de modelado tridimensional, otorga total libertad al desarrollador para crear las piezas que conformarán cualquier mundo virtual que haya imaginado; y compatible con tres lenguajes de programación diferentes, le otorga a éste las herramientas con las que imbuir finalmente a ese nuevo mundo, hasta ahora inmóvil y estático, de vida.

Este proyecto, sin olvidar el primer grupo de herramientas, se concentra especialmente en el segundo. Así, a través de los lenguajes de programación JavaScript y C# se construyen los engranajes de un completo conjunto de instrumentos de navegación que aportan a un mundo virtual existente toda una nueva dimensión, una en el que el usuario deja de ser mero espectador para convertirse en dueño y señor de sus pasos, y en la que su experiencia de uso se enriquece por completo, permitiéndole comunicarse y controlar el sistema, haciéndole sentirse parte de este nuevo mundo con el que ahora puede interactuar mediante cuidadas interfaces. Gracias a toda la programación implicada no sólo un mundo virtual puede cobrar vida, se logra también que el usuario deje de sentirse un visitante de paso para sentirse en él un ciudadano de hecho.

2.1. Instalación de Unity

Existen dos versiones de Unity, una gratuita a disposición del público en http://unity3d.com/es/unity/download, y otra denominada Unity PRO que, junto con complementos que añaden funcionalidades adicionales al programa base, es de pago. Las diferencias entre ambas versiones agrupan mejoras diversas en terrenos como la optimización del rendimiento, efectos avanzados de gráficos, audio y vídeo, o gestión mejorada de las animaciones e iluminación.

La instalación del programa se realiza de la forma acostumbrada. Una vez bajado el mismo y ejecutado el instalador, este nos guiará a través de los pasos que conducirán finalmente a la creación de una copia operativa del entorno Unity en el ordenador huésped. Destacable entre estos es el paso en el que se nos da la opción de instalar el entorno de desarrollo integrado MonoDevelop.

2.2. MonoDevelop

Este entorno de desarrollo integrado, también conocidos por sus siglas en inglés IDE, es la opción que ofrece Unity de serie para la gestión de toda la programación de un proyecto (Figura 2.2). La integración entre ambos programas permite que las modificaciones realizadas en el código escrito en el IDE se reflejen automáticamente en Unity.

Este programa informático, como algo común a innumerables otros, está en constante evolución. Esto que implica que en una versión determinada puedan existir fallos que se resolverán en futuras. En la versión utilizada para este proyecto, la versión 4.0.1, los caracteres acentuados no pueden utilizarse de forma predeterminada, algo que, en un lenguaje como el español, supone un problema. Este contratiempo puede, sin embargo, resolverse fácilmente, y reflejaré aquí la sencilla metodología resolutiva como referencia.

La solución a este obstáculo consiste en modificar el valor de la preferencia *Input Methods* para cada fichero editado, cambiando su estado predeterminado *System* a *Simple*. Esto se logra pulsando el botón derecho del ratón en cualquier punto del fichero que se esté editando, pulsando a continuación en el menú que se despliega sobre *Input Methods* y eligiendo finalmente *Simple* de entre el listado que se presenta, tal y como se indica en la Figura 2.3. El proceso citado deberá repetirse tras cada uso de MonoDevelop debido a que el valor de la preferencia que hemos modificado se reinicia cada vez.

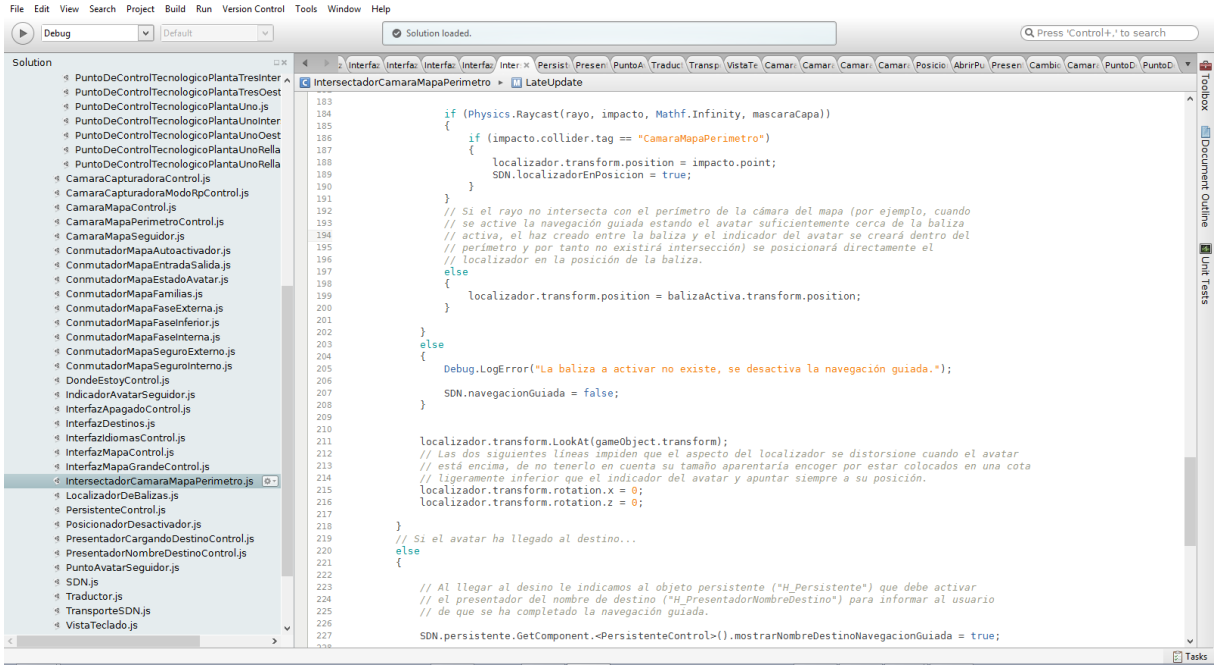


Figura 2.2: MonoDevelop

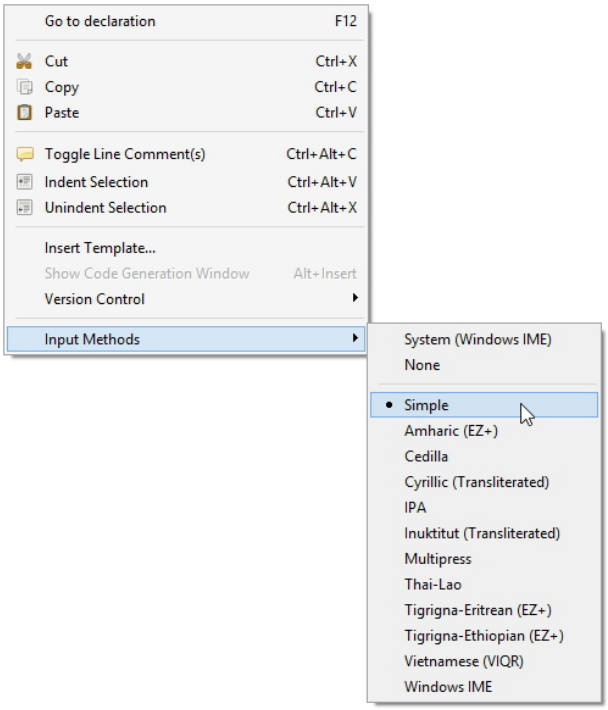


Figura 2.3: Valores de *Input Methods*

2.3. Importar un proyecto

En un proyecto como este que toma un mundo virtual existente como punto de partida el primero de los pasos consiste en importarlo.

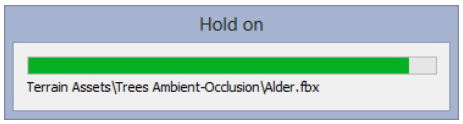


Figura 2.4: Unity importando un proyecto

Esta operación debería ser, aunque generalmente larga, automática y trivial, siendo Unity el que de forma invisible al desarrollador importara cada uno de los elementos constituyentes de este primer eslabón bastando indicarle la ubicación de los mismos; sin embargo pueden surgir problemas, y para evitar que futuros desarrolladores deban lidiar con la búsqueda de las soluciones, se dejará aquí reflejada la explicación y el procedimiento que resolvió finalmente el problema que en el presente caso detenía por completo este proceso a veces no tan trivial.

Para explicar el origen de este contratiempo hay que considerar de antemano algunos aspectos. Este proyecto es uno de varios que lo han precedido, desarrollados cada uno en plataformas y programas diferentes y armoniosamente unidos a su cabo en el corazón de Unity. Esta naturaleza multiplataforma y multiprograma del proyecto puede originar incompatibilidades entre formatos: aquellos con los que una determinada plataforma puede trabajar perfectamente puede que en otra originen conflictos. Tampoco podemos olvidar que Unity es un avanzado *software* que está en continuo desarrollo, versión tras versión se pule su funcionamiento y se corrigen errores conocidos con atención a su prioridad, por lo que en una determinada versión y cumpliéndose un contexto concreto quizás debamos lidiar con fallos aún sin resolver.

El contexto en este caso particular ha sido la versión 4.3.1 de Unity, el sistema operativo Windows 8 y, el desencadenante del problema, importar el proyecto contenedor del mundo virtual base. Combinados estos elementos Unity fallaba continuamente durante el proceso de importación, dejándolo siempre sin completar.

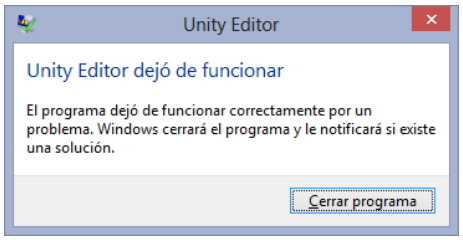


Figura 2.5: Error de importación

La solución a este inconveniente en este contexto concreto consiste en lanzar el programa utilizando una instrucción personalizada[1], `-force-d3d11`, que le indica explí-

[1]Se puede encontrar un listado completo de todas las instrucciones con las que es posible personalizar

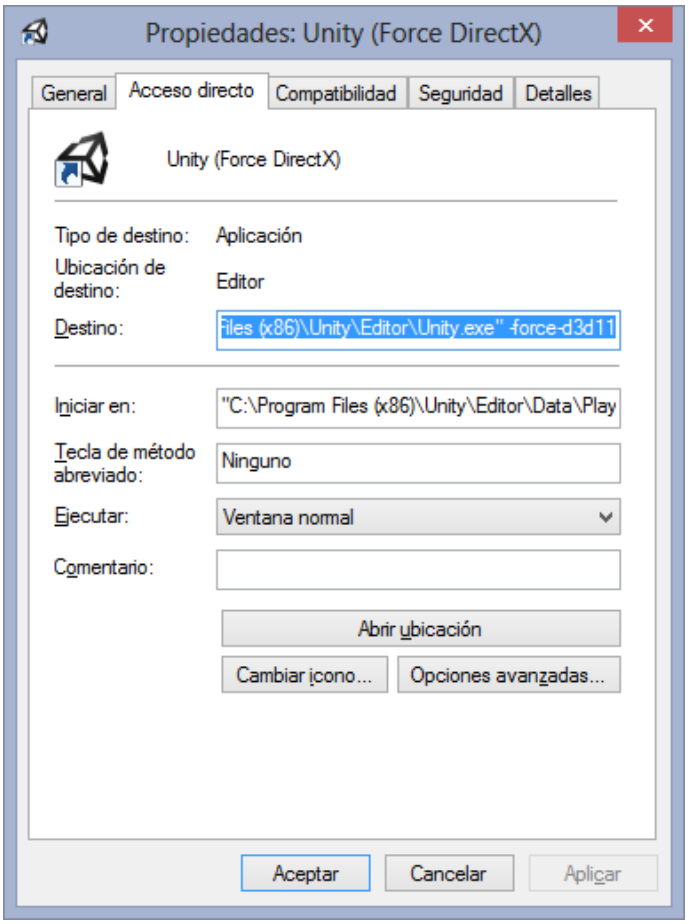


Figura 2.6: Acceso directo personalizado con la instrucción `-force-d3d11`

citamente el deseo de que el Editor de Unity utilice Direct3D 11 para el renderizado (generación de imágenes por ordenador a partir de modelos matemáticos de superficies tridimensionales) de los elementos.

En Windows puede editarse el acceso directo que permite lanzar el programa para incluir esta instrucción siempre que se inicie Unity. Para ello se añadirá, en las propiedades del acceso directo, al final del valor que ya incluye el campo «Destino». Se puede apreciar en la Figura 2.6 el citado campo ya modificado de un acceso directo, copia del original creado por el programa durante su instalación y al que, por diferenciarlo del primero, se le ha dado el nombre de "Unity (Force DirectX)".

Una vía alternativa de indicar la instrucción durante el inicio del programa consistiría en llamarlo desde una consola, especificando `-force-d3d11` como argumento. Para una instalación estándar habría que escribir
`"C:\Program Files (x86)\Unity\Editor\Unity.exe" -force-d3d11`.

el lanzamiento del programa en http://docs.unity3d.com/Manual/CommandLineArguments.html.

Capítulo 3

Definiciones

A lo largo de este tomo, y debido a la naturaleza de los temas tratados, se utilizarán términos quizás poco cotidianos, alguno recogido directamente del inglés, y otros que, si bien existentes en castellano, adquieren un significado especial en esta materia.

Con intención de solventar cualquier duda que pudiera surgir en cuanto a la terminología utilizada se dedica este capítulo a plantear y definir estos términos.

Unity Entorno de desarrollo multiplataforma para la creación de aplicaciones, programas y juegos tridimensionales.

Editor Conjunto de ventanas en Unity que permiten, organizan y facilitan el desarrollo de una aplicación. (En la Figura 3.1 puede verse el Editor de Unity y algunas de las definiciones siguientes apuntadas visualmente).

Aplicación, programa o juego Resultado final operativo e independiente de Unity obtenido tras la etapa de desarrollo.

Proyecto Denominación de la aplicación durante la etapa de desarrollo.

Inspector Ventana en Unity que agrupa las características y componentes asociados a cualquier elemento u objeto del proyecto, permite alterar sus características cómodamente.

Interfaz o GUI[1] Elementos que permiten la comunicación e interacción entre el usuario o desarrollador y el programa en cualquiera de sus etapas.

[1] El término GUI procede del inglés *Graphical User Interface*, interfaz gráfica de usuario.

Ventana de proyecto o ventana *Project* Ventana en Unity donde se muestra ordenadamente todos los elementos componentes de un proyecto.

Ventana de jerarquía, ventana *Hierarchy*, o jerarquía Ventana en Unity en la que se presentan los elementos relacionados con la escena actual, los que existen ya al principio de esta y todos aquellos que se crean durante su ejecución.

Escena Cada una de las partes en las que puede dividirse un proyecto a discreción del desarrollador. Representan las etapas de la aplicación.

Objeto Denominación de los elementos mostrados en la ventana de jerarquía, compuestos a su vez de componentes y recogidos estos en el Inspector.

Componente Cada uno de los elementos compositivos de un objeto, representación de las propiedades y características de los mismos.

Prefab Objetos preconfigurados que pueden reutilizarse cómodamente en cada escena mediante la clonación. El elemento original se recoge en la ventana de proyecto, el elemento clonado aparecerá en la ventana de jerarquía.

Clonación o *Instantiation* Procedimiento por el que se crea un clon de un prefab, convirtiéndolo en un objeto más de una escena.

Guiones o scripts Ficheros de texto con instrucciones en un lenguaje de programación determinado que, asociados por lo general a un prefab o a un objeto, permiten modificar o controlar su comportamiento.

Textura Imágenes contenidas en la ventana de proyecto. Sus funciones en Unity son variadas, una principal es la servir de aspecto exterior, de funda o «piel», a muchos prefab y objetos.

Importación Unity trabaja con ficheros corrientes, pero antes de poder utilizarlos debe importarlos. A través de este proceso automático es como una imagen se convierte en una textura.

Ventana de juego o ventana *Game* Ventana en Unity en la que puede comprobarse el funcionamiento de una escena tal y como quedaría en la aplicación final.

Ventana de escena o ventana *Scene* Ventana en Unity donde se construye cada escena, en ella se colocan y ajustan los elementos que las componen.

Consola Ventana en Unity que sirve de registro de mensajes, avisos y errores que el proyecto genere durante su desarrollo.

Guión, *script* o programa informático[2] Fichero de texto con instrucciones dadas en un lenguaje de programación determinando[3], que, en particular, determinará y modificará el comportamiento del objeto con el que se asocie en forma de componente, y, en general, gestionarán el comportamiento y funcionalidades globales del proyecto. Los guiones creados y disponibles se recogen en la ventana de proyecto.

Frame Cada uno de los instantes en los que se divide la ejecución de la aplicación o proyecto. Representan la unidad mínima de tiempo en la que todo sucede. Si imaginamos la aplicación o proyecto como una sucesión de imágenes, a modo de película, cada una de esas imágenes ocurriría en un *frame*. Unity utiliza esta medida de tiempo para realizar todos los cálculos que permitirán dibujar en pantalla la imagen que corresponda a continuación de la actual. Es una medida variable y dependiente del *frame rate*[4].

Frame rate Cantidad de *frames* por segundo a la que se ejecuta la aplicación. Equipos diferentes, más o menos potentes, ejecutarán la aplicación con mayor o menor eficiencia. El *frame rate* es una medida de esa eficiencia.

Cámaras Objetos que colocados en una escena determinan lo que el usuario verá de esta. Una cámara configurada para seguir los movimientos de un avatar permitirá crear la impresión de que el usuario observa el mundo virtual a través de los ojos de este.

Avatar Representación virtual del usuario en un mundo tridimensional y, en general, representación del mismo en cualquier entorno creado por ordenador.

Mundo virtual Entorno artificial generado por ordenador, pudiendo ser una representación del mundo real o uno totalmente imaginado.

Plataforma de compilación Es el sistema elegido como encargado de ejecutar la aplicación. El paso previo a la compilación final de la aplicación consiste en seleccionar el sistema donde se ejecutará: Windows, Mac OS X, Linux, iOS, Android, etcétera.

Compilación Proceso final por el que el proyecto, hasta ahora dependiente de Unity, se transforma en una aplicación autónoma.

[2] El término «programa», según el contexto, puede describir tanto a una aplicación como a un guión.
[3] Unity admite, como lenguajes de programación posibles, JavaScript, C# y Boo.
[4] Existe también otra medida de tiempo, constante e independiente del *frame rate*, que Unity utiliza para realizar los cálculos físicos con objeto de garantizar resultados similares en equipos de diferentes características.

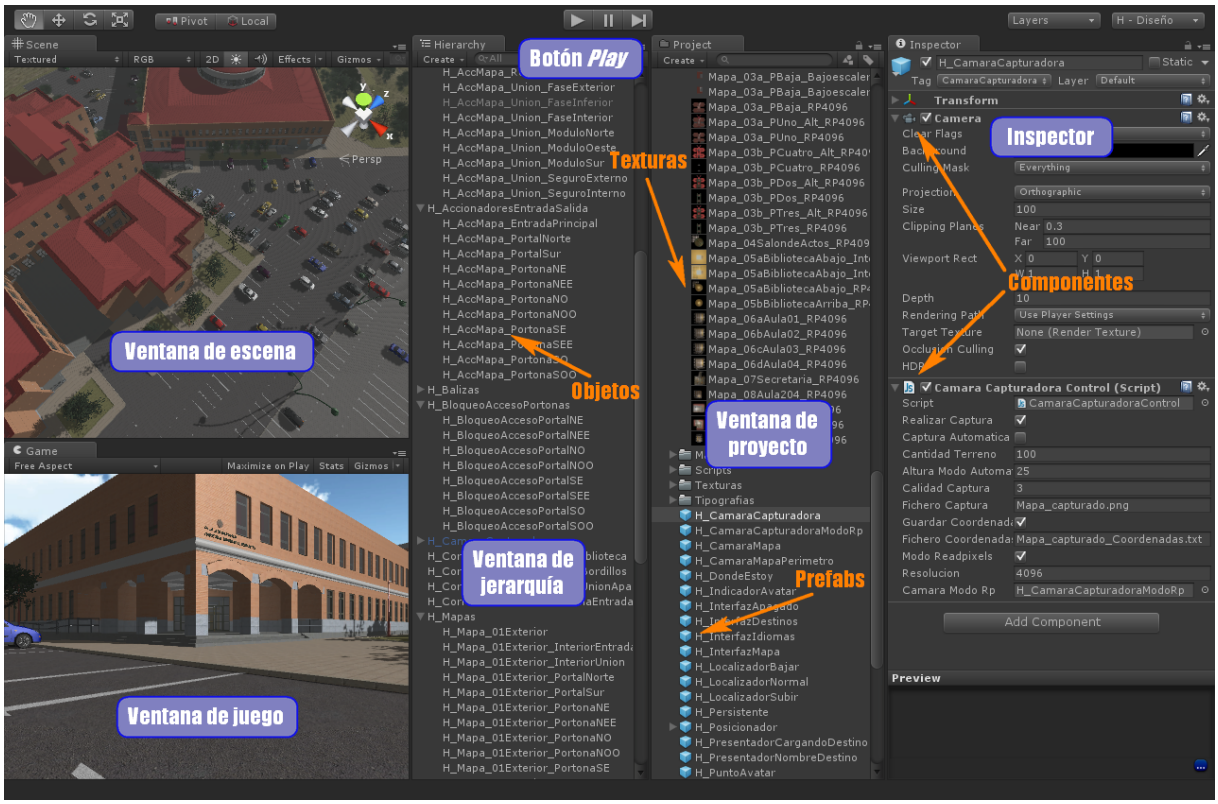


Figura 3.1: Descripción del Editor de Unity

Botón *Play* Botón de la interfaz de Unity que activa la ejecución del proyecto en fase de desarrollo. Permite al desarrollador comprobar el funcionamiento del mismo sin necesidad de compilarlo.

Proyecto legado Proyecto anterior al aquí tratado. Proporciona el mundo base virtual al que analizará e interpretará el completo sistema de navegación aquí desarrollado.

Capítulo 4

Estilo de código

Este proyecto se cimenta en la escritura de *scripts*[1], la programación de guiones o ficheros de texto que incluyen las instrucciones que una vez interpretadas por el compilador de Unity realizarán todo tipo tareas, desde dibujar un simple botón en pantalla, hasta controlar la posición, movimiento y física de cada elemento componente de un mundo virtual tridimensional. La programación y escritura de código otorga control absoluto al desarrollador de Unity sobre cada aspecto y detalle del mundo virtual que se está creando.

Semejante poder debe ajustarse a unas líneas organizativas para optimizar los resultados y garantizar que estos se obtengan siempre fácilmente, a ser posible de forma totalmente transparente, posibilitando que cualquiera, aún sin conocimientos previos de programación, pueda sacar el máximo rendimiento del código escrito, como aquel que enciende un ordenador y lo utiliza para infinidad de tareas sin necesidad de saber cómo funciona éste internamente.

Este proyecto se ha planteado como herramienta de futuro, los instrumentos que proporciona los usarán futuros desarrolladores en sus propios proyectos, por lo que los aspectos mencionados en el párrafo anterior se tornan especialmente importantes. Por esta razón se ha puesto especial cuidado en la claridad de la programación, incluyendo en cada guión un encabezamiento descriptivo y anotaciones explicativas de las funciones que realiza el código y los métodos que utiliza para llevarlas a cabo como puede observarse en la Figura 4.1. No sólo quedan resueltos así los cómos y los porqués, también se garantiza que las posibles ampliaciones ulteriores se efectúen fácil y cómodamente.

[1]Recogidos en el Anexo F.

CamaraCapturadoraControl ▸ Awake

```
1  #pragma strict
2
3  /*#############################################################
4
5  CamaraCapturadoraControl.js
6  ---------------------------
7
8  Guión encargado de gestionar la cámara capturadora de mapas.
9
10 Define y controla las variables que el usuario puede personaliz
11 el Inspector del prefab "H_CamaraCapturadora". Gestiona la crea
12 de los ficheros generados tras la captura. Controla la presenci
13 la escena durante la captura y reajusta los componentes de la c
14 controladamente el relevo. Gestiona también el uso del modo ava
15 (ReadPixels) cuando éste se ha activado mediante el Inspector y
16 de compilación presente para ajustarse a las posibles restricci
17
18 #############################################################
19
20
21 import System;
22 import System.IO;
23
24 var realizarCaptura : boolean;
25 var capturaAutomatica : boolean;
26
27 // Definimos en el Inspector la altura desde la que la cámara r
28 // alta que el objeto más alto de la escena a cartografiar, sin
29 // ignorar techos al capturar interiores sin necesidad de tener
30
31 var cantidadTerreno : int;
32 var alturaModoAutomatico : float;
```

Figura 4.1: Código documentado (fragmento)

En la misma línea, toda la programación realizada en JavaScript se ha escrito acorde a la directiva `#pragma strict`. Esta directiva obliga al desarrollador a escribir código optimizado y eficiente, forzándolo a realizar las decisiones que en otros escenarios sería el compilador el que las tomara por él. Se traduce esto en un control y seguimiento manual de la asignación del tipo de cada variable utilizada, liberando al compilador de esta tarea y logrando así la mayor optimización posible en lo que respecta a código JavaScript. El ajuste a esta directiva, además de aumentar el rendimiento, implica también la compatibilidad del código con plataformas móviles Android o iOS, incrementando el alcance de todo el proyecto.

Referido también a la optimización del rendimiento del código, se han utilizado referencias expresas a componentes, variables de control y *triggers*, siempre con la finalidad de ejecutar un código sólo cuando esto sea necesario hacerlo, evitando que por comodidad se ejecute continua e innecesariamente en segundo plano y mejorando el rendimiento. Las referencias expresas a componentes implican el almacenamiento previo de algún objeto o componente de este en una variable expresa para que Unity no tenga que buscarlo entre la jerarquía completa de objetos cada vez que sea necesario acceder a él; las variables de control, generalmente de tipo verdadero o falso, permiten gestionar la ejecución de un código sólo cuando la variable de control se haya activado a partir de un suceso previo; y finalmente, los *triggers* o disparadores, ejecutan el código que llevan asociado sólo cuando algo entra, sale o permanece en el entorno que controlan.

También, y siempre con miras a la optimización, el código tiene en cuenta la plataforma de compilación elegida como destinataria de la aplicación, y se ajusta automáticamente a las particularidades que ésta imponga. Para conseguir esto se utilizan

directivas[2] que permiten definir qué código se compilará exclusivamente para qué plataforma de compilación.

Por último, en aras de la claridad y facilidad de comprensión, se ha cuidado con esmero el formato, limpieza y estructura de todos los guiones escritos. Este aspecto, en apariencia únicamente estético, es, no obstante, sumamente importante. El correcto sangrado de cada bloque de código, la coherencia férrea en la notación de variables o funciones, el esfuerzo, a pesar de la naturaleza siempre críptica de los lenguajes de programación, de utilizar nombres autodescriptivos para estos elementos y, finalmente, el especial cuidado de separar aquellas variables útiles para la gestión interna de los guiones[3] de aquellas otras cuyos valores puedan o deban ser personalizados por el usuario, permite que el código de este proyecto pueda ser comprendido, utilizado, ampliado, revisado y compartido sin esfuerzo.

En conclusión, el estilo de código utilizado en este proyecto puede definirse con tres adjetivos:

- Claro
- Optimizado
- Documentado

[2]Se incluye un listado completo de las directivas disponibles para este fin en la siguiente dirección http://docs.unity3d.com/Manual/PlatformDependentCompilation.html

[3]Esto se logra asignando la directiva `@HideInInspetor` a aquellas variables que deban ser accesibles para el código del resto de los guiones pero que, en cambio, no sean útiles para el usuario o el desarrollador. Asignar esta directiva a una variable la oculta en el Inspector del objeto al que está asociado el guión en el que aparece, dejando a la vista tan sólo aquellas que el usuario deba manipular, y logrando así una limpieza y claridad que, sin esta gestión individualizada de cada variable, no se podría conseguir.

Capítulo 5

Ciclos de ejecución

Algo tan cotidiano como una cola de clientes en un establecimiento de reparto permitirá ilustrar la importancia del orden en la ejecución de una serie de procesos. La cola evita conflictos, un cliente sabe que su turno llega sólo cuando antes se han resuelto las necesidades de los que lo preceden, de lo contrario surgen problemas, pueden confundirse los pedidos, recibir los clientes un producto equivocado o estos directamente decidir marcharse con las manos vacías y frustrados. Si no se define y respeta un orden cualquier proceso queda a merced del caos.

Unity resuelve este importante aspecto estableciendo un conjunto de ciclos de ejecución claramente ordenados, a los que el desarrollador puede y debe atender para evitar cualquier conflicto en el código del proyecto, asegurándose de que todo lo que éste necesite quede siempre resuelto en un ciclo anterior y que su ejecución se desarrolle el momento adecuado.

En la práctica estos ciclos de ejecución toman forma de funciones, contenidas dentro de un guión que irá asociado a un objeto o a un prefab, y dentro de las que el desarrollador agrupará el código que desee que se evalúe en cada una.

Este proyecto ha prestado una especial atención a la forma y momento en que se despliega el código que en él se plantea. No sólo con vistas a la ejecución óptima y libre de conflictos de las instrucciones presentes en cada uno de sus guiones, también para garantizar que todo guión escrito posteriormente a este proyecto se resuelva sin contratiempos y, así como se ha puesto un especial cuidado en el presente y en el futuro, también se ha prestado atención al pasado, revisando los guiones ya presentes e integrándolos en un sistema libre de cualquier conflicto de ejecución, modificándolos y mejorándolos con este

fin cuando fuera necesario.

Las funciones que determinan estos ciclos de ejecución, agrupadas según el momento en el que se activan, se describen a continuación como forma de referencia. Asimismo, por la naturaleza profundamente técnica de esta información, al final de este capítulo se incluye también un diagrama que busca mostrar, de forma sencilla, una relación ordenada de todos los ciclos de ejecución disponibles y las vinculaciones que presentan.

5.1. Funciones activadas al cargar una escena

Awake Ejecutada nada más cargar una escena y sólo si el objeto asociado está activo, de no estarlo esta función se ejecutaría al momento de activarse, pero siempre una sola vez para cada objeto y escena.

OnEnable Ejecutada al activarse el objeto asociado (esta función puede repetirse si el objeto en algún momento se desactivara y posteriormente reactivara).

5.2. Funciones activadas justo antes del ciclo Update

Start Ejecutada justo antes del primer *frame* del ciclo de ejecución principal Update.

5.3. Funciones activadas entre *frames*

OnApplicationPause Ejecutada al finalizar el *frame* en el que se detecta la pausa.

5.4. Funciones activadas durante el ciclo Update

El ciclo de ejecución Update es donde el programa pasará la mayor parte de su tiempo. Todo aquello que deba ajustarse, comprobarse o modificarse de forma regular sucede aquí: cálculo de movimientos, captura de datos de entrada generados por el usuario o gestión de temporizadores son ejemplos de tareas que suceden en este ciclo.

Update Ejecutada una vez por cada frame. El tiempo que transcurre entre *frames* puede no ser constante pues depende del *frame rate* de la aplicación.

FixedUpdate Ejecutada una vez cada un tiempo fijo e independiente del *frame rate*.

LateUpdate Ejecutada una vez por cada frame, justo después de que todos los cálculos realizados en Update se hayan completado.

5.5. Funciones activadas durante el ciclo de renderizado

OnPreCull Ejecutada justo antes de que se determine cuáles son los objetos visibles a la cámara.

OnBecameVisible/OnBecameInvisible Ejecutada cuando un objeto se vuelve visible o invisible a la cámara.

OnWillRenderObject Ejecutada una sola vez para cada cámara que ve el objeto asociado.

OnPreRender Ejecutada justo antes de que la cámara comience a renderizar la escena.

OnRenderObject Ejecutada justo después de que la cámara termine de renderizar la escena.

OnRenderImage Ejecutada después de que la cámara termine de renderizar la escena con objeto de poder tratar la imagen obtenida (sólo en la versión PRO).

OnGUI Ejecutada varias veces por *frame* como respuesta a las peticiones de tipo GUI (interfaz gráfica de usuario). Los menús de una aplicación, los botones que el usuario puede pulsar para, por ejemplo, pausar, reiniciar o salir de una aplicación, son el propósito de este ciclo.

OnDrawGizmos Ejecutada como función de ayuda durante el desarrollo de la aplicación, permite dibujar elementos gráficos que no aparecerán en la versión final de la misma.

5.6. Corrutinas

Estas funciones detienen su propia ejecución hasta que la condición *yield* indicada termina.

yield Detiene su ejecución hasta el siguiente *frame*, justo al terminar todos los ciclos Update.

yield WaitForSeconds Detiene su ejecución durante el tiempo especificado, y continuará en el *frame* correspondiente justo después de que todos los ciclos Update hayan terminado.

yield WaitForFixedUpdate Detiene su ejecución hasta que el ciclo *FixedUpdate* de todos los guiones se haya completado.

yield WWW Detiene su ejecución mientras no se haya completado una descarga web.

yield StartCoroutine Permite encadenar corrutinas y detiene su ejecución hasta que la anterior se haya completado.

5.7. Funciones activadas cuando el objeto se desactiva o destruye

OnDisable Ejecutada cuando se desactiva el guión o el propio objeto asociado.

OnDestroy Ejecutada cuando un objeto se destruye, sea expresamente o porque la escena se haya cerrado.

OnApplicationQuit Ejecutada justo antes de salir de la aplicación (durante el desarrollo de la aplicación se ejecuta cuando se detiene el modo de reproducción en el Editor).

5.8. Diagrama de ejecución de un guión

El diagrama de la Figura 5.1 muestra el orden y repetición de las funciones citadas en los puntos anteriores.

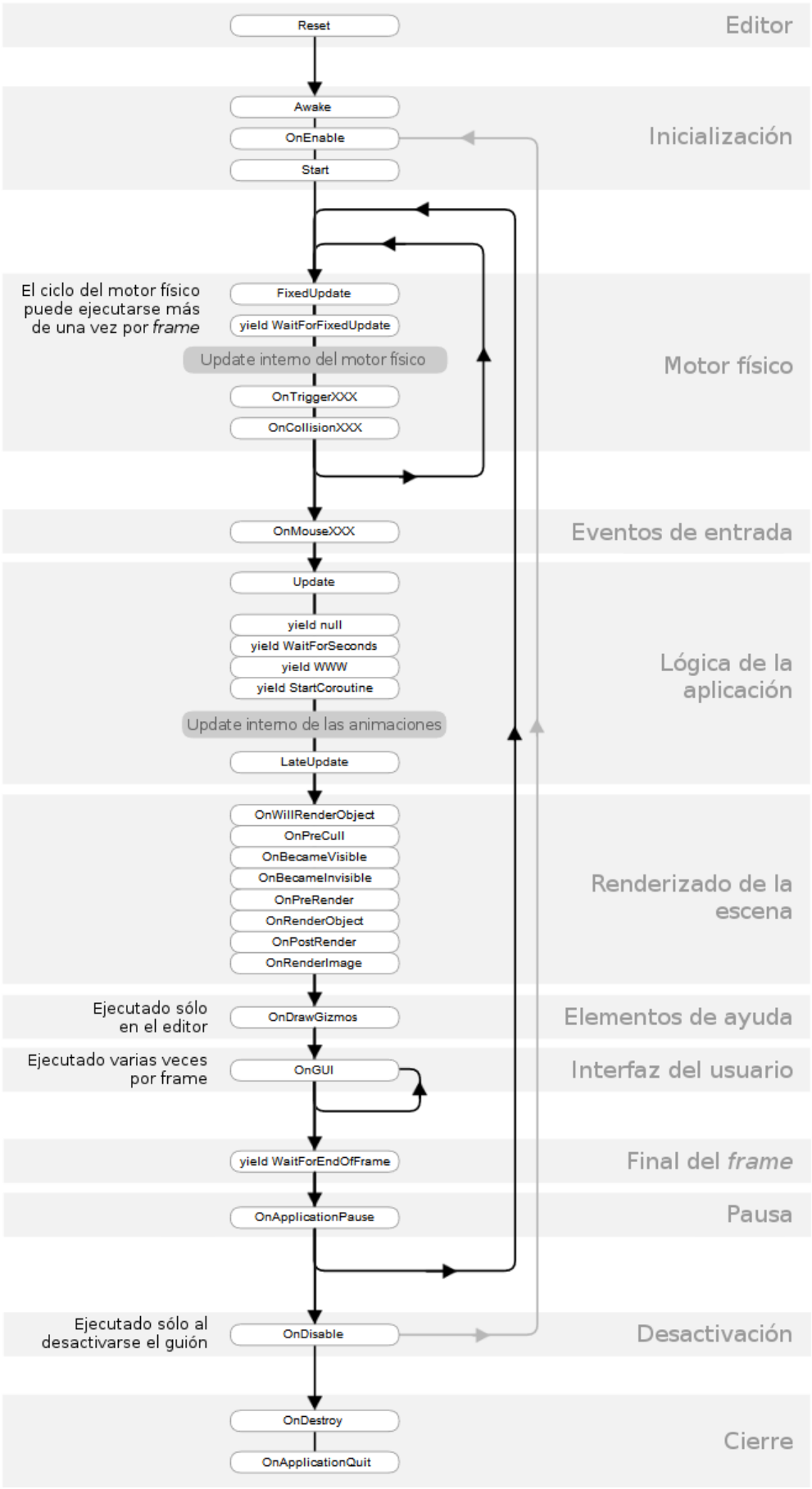


Figura 5.1: Ciclos de ejecución de un guión

Capítulo 6

Presentación del Proyecto

En los capítulos siguientes se explicará y describirá el funcionamiento por separado de cada uno de los sistemas que, funcionando juntos, integran y configuran un completo y preciso sistema de navegación multifunción para un entorno virtual tridimensional, objeto último de este Proyecto.

Las características de cada uno de los bloques componentes se explicarán con detalle en cada capítulo correspondiente. Aquí se expondrá un listado resumido, a modo de visión global introductoria, de las propiedades generales del sistema funcionando como uno solo.

Figura 6.1: Logotipo creado como imagen de presentación general

6.1. Características generales del sistema como conjunto

- Posibilidad de desactivarlo por completo con sólo desactivar una casilla en el Inspector (Figura 6.2).
- Código sólido y totalmente probado.
- Elementos gráficos cuidados minuciosamente:

- Interfaz.
  - Completamente autoajustable a los cambios de resolución en tiempo real.
  - Visualmente atractiva y eficiente.
  - Diferentes versiones para cada botón para mejorar la experiencia interactiva.
    - Normal
    - Activo
    - Pulsado
    - Elegido
- Mapas.
  - Editados cuidadosamente para mejorar la experiencia inmersiva.

- Personalizable
  - Total y sencillamente personalizable sin necesidad de conocimientos de programación.
  - Elementos personalizables concentrados en un mismo lugar para facilidad del desarrollador.
  - Análisis automático de los valores personalizables introducidos por el desarrollador y notificación de los errores detectados.

- Pensado como herramienta de futuro.

- Profusamente documentado.
  - Todos los guiones utilizados cuentan con una descripción introductoria y numerosos comentarios descriptivos de las funciones y porqués del código.
  - Instrucciones de uso de los principales sistemas incluidas como referencia y consulta rápida en la propia ventana del proyecto.

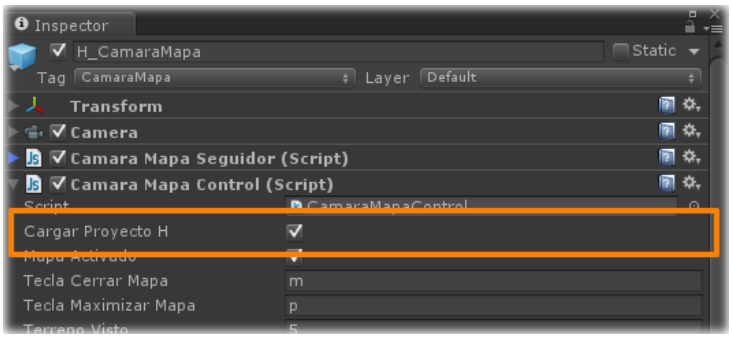


Figura 6.2: Desactivación completa del sistema con sólo desactivar una casilla

- Mejora del proyecto legado[1]

  - Mejora de aspectos clave para garantizar la estabilidad del conjunto en cualquier ampliación futura.

  - Mejora de aspectos diversos para mejorar la experiencia del usuario.

6.2. Elementos centrales

Hay tres elementos que, si bien no constituyen un sistema por sí mismos, por su importancia para el resto se presentarán de forma individual.

6.2.1. «CamaraMapaControl.js»[2][3]

El guión CamaraMapaControl.js es el núcleo del proyecto, es el encargado de clonar y preparar, nada más cargar la escena, cada uno de los sistemas que lo componen: la génesis.

Asociado al prefab H_CamaraMapa, presenta a través de su Inspector todos los elementos personalizables principales que el desarrollador puede cambiar según su gusto para modificar el comportamiento final de todo el Sistema de Navegación. Actúa también por tanto como elemento centralizador[4].

[1] En el Capítulo 18 se incluye un listado con las mejoras y modificaciones realizadas, así como la naturaleza y descripción de estas.
[3] La terminación «js» indica que está escrito en JavaScript.
[4] Estos elementos personalizables se describen cada uno en las secciones correspondientes a los sistemas que personalizan.

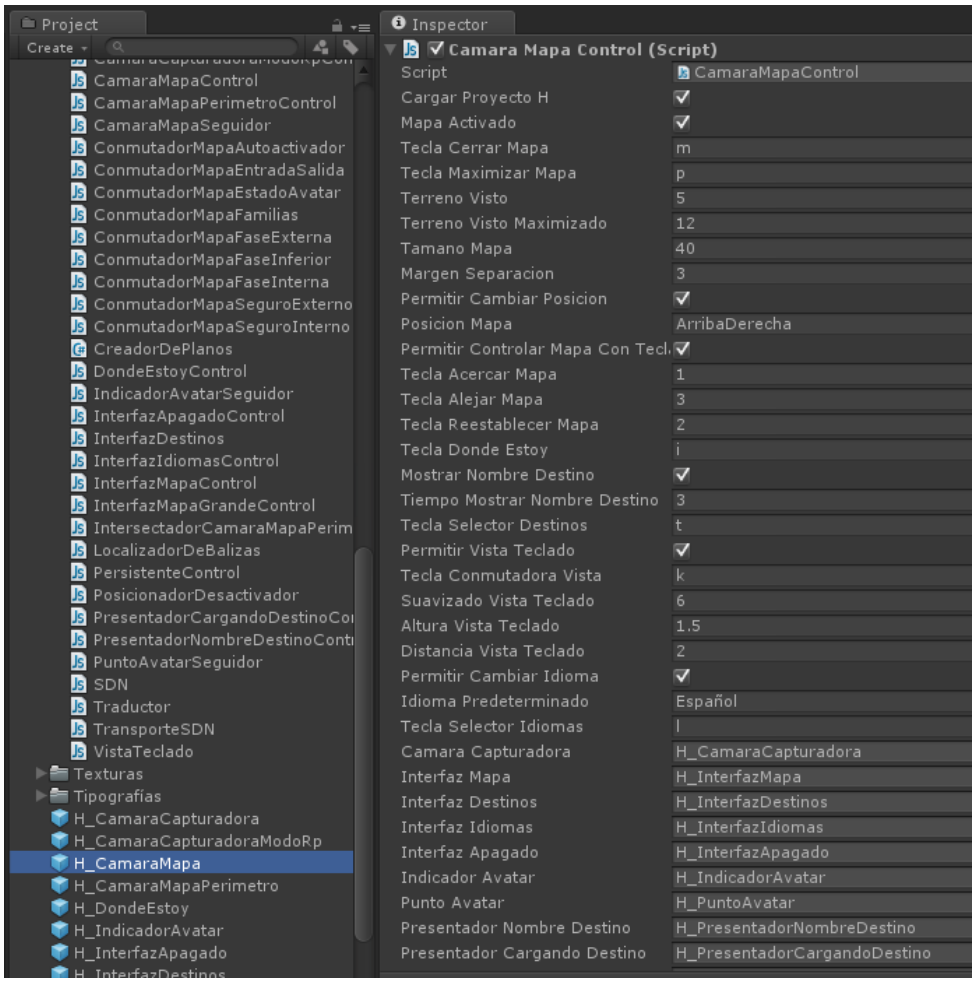


Figura 6.3: Inspector de H_CamaraMapa

Además de presentar estos elementos personalizables, contiene también el código necesario para la comprobación automática de la integridad de sus valores, notificando al desarrollador cuando detecte error en alguno de ellos, y asignando un valor predeterminado válido si no se corrige el error para garantizar en cualquier caso la correcta ejecución de la aplicación.

Como elemento creador, la tarea de clonar y preparar al resto de sistemas la realiza atendiendo con especial atención a los ciclos de ejecución, asegurando que todos los elementos que necesitan para su funcionamiento están disponibles antes de su puesta en marcha definitiva: variables, componentes, configuraciones, personalizaciones, interrelaciones, todas estarán perfectamente definidas y dispuestas antes de que el primero de los *frames* de la aplicación haya cargado.

El prefab H_CamaraMapa, como portador de este guión, ha de estar cargado en la escena para que Unity evalúe las instrucciones de CamaraMapaControl.js. Esta carga previa se realiza mediante un segundo guión, SustContr.js, asociado al objeto CambioCon-

trolador. Este objeto se encuentra inicialmente ya presente en cada una de las escenas del proyecto y forma parte de los elementos del proyecto legado. Originalmente encargado de clonar en la escena el avatar que el usuario hubiera elegido, se ha mejorado extensamente[5] para acomodarse correctamente a los ciclos de ejecución y servir de punto de partida del presente proyecto. Puede por tanto considerarse como el vínculo entre ambos.

El guión CamaraMapaControl.js es también el encargado de gestionar explícitamente el comportamiento de la cámara del Sistema de Posicionamiento, aspecto que se describirá en la sección correspondiente al mismo.

Una vez cargada la escena, este guión puede activarse y desactivarse según lo requiera el Sistema de Posicionamiento, por este motivo, una vez ha realizado sus funciones de preparación y carga del resto de sistemas, delega su función de gestor central del proyecto, en el guión PersistenteControl.js, que permanecerá siempre activo en la escena y cuyo funcionamiento se describirá a continuación.

6.2.2. «PersistenteControl.js»

El guión PersistenteControl.js actúa como observador permanente, como agente continuamente atento a los eventos generados por el resto de sistemas, y como intermediario entre ellos.

Asociado al prefab H_Persistente, aparece por primera vez en escena por virtud del guión ya introducido CamaraMapaControl.js y permanece en esta siempre activo, a la espera de que cualquier sistema que lo necesite le asigne una tarea que requiera de su cualidad de permanente. Si CamaraMapaControl.js se encarga de los preparativos iniciales, PersistenteControl.js recoge el testigo y continúa el trabajo más allá de la fase de inicialización.

Es el encargado de gestionar los eventos generados con el teclado, vigilando la pulsación de una tecla, asignándole una función y trasladando el resultado al sistema oportuno.

Actúa asimismo como base de información constante y centralizada dentro de una misma escena y, en especial, actúa como coordinador de los presentadores de texto (Capítulo 12), la herramienta que permite a todo el conjunto de sistemas presentarle en pantalla mensajes informativos al usuario. Es el guión PersistenteControl.js el que gestiona cuándo

[5] Relación de cambios recogidos en el Capítulo 18.

y cómo activarlos, aplicando todas las medidas de control necesarias para garantizar la perfecta armonía y precisión en el sistema.

6.2.3. «SDN.js»

Sin este guión, al cargar una nueva escena los datos recogidos en la anterior se perderían. La función del SDN.js es la de servir como base de datos global y persistente, para todos los sistemas y para todas las escenas.

El guión SDN.js es el encargado de que el estado de los elementos que así lo requieran pueda perdurar entre escenas. Así, si el usuario ha decidido modificar la ubicación del mapa integrado del Sistema de Posicionamiento (Capítulo 9), al cargar una nueva escena este aparecerá en el mismo lugar en el que estaba en la anterior; lo mismo ocurre si el usuario ha decidido modificar la cantidad de terreno que en él se muestra, o el modo de orientación, o el estilo de vista de la cámara, o el idioma; cualquier cambio que el usuario realice en el sistema, sin el SDN.js, se perdería al cargar una nueva escena.

Además de conservar entre escenas las modificaciones efectuadas por el usuario, realiza también la función equivalente para los sistemas internos de todo el sistema de navegación como conjunto, y proporciona asimismo un grupo de funciones globales, entendida una función como un bloque reutilizable de instrucciones agrupadas para la consecución de una tarea concreta, que son utilizadas por todo el sistema.

Se puede entender a este guión como una base de datos, y esta misma naturaleza hace innecesario tener que asociarlo a un objeto o a un prefab para que su código se ejecute. Es autónomo.

6.3. Organización de los recursos dentro de Unity

Buscando claridad en la organización, se han ubicado los prefab, guiones, texturas y demás elementos que estructuran el presente proyecto dentro de una misma carpeta, «Proyecto H», en la ventana *Project* según se indica en la Figura 6.4

Aquellos otros elementos que se hayan modificado o añadido fuera de esta estructura quedaran explicados en el apartado en el que se les haga mención.

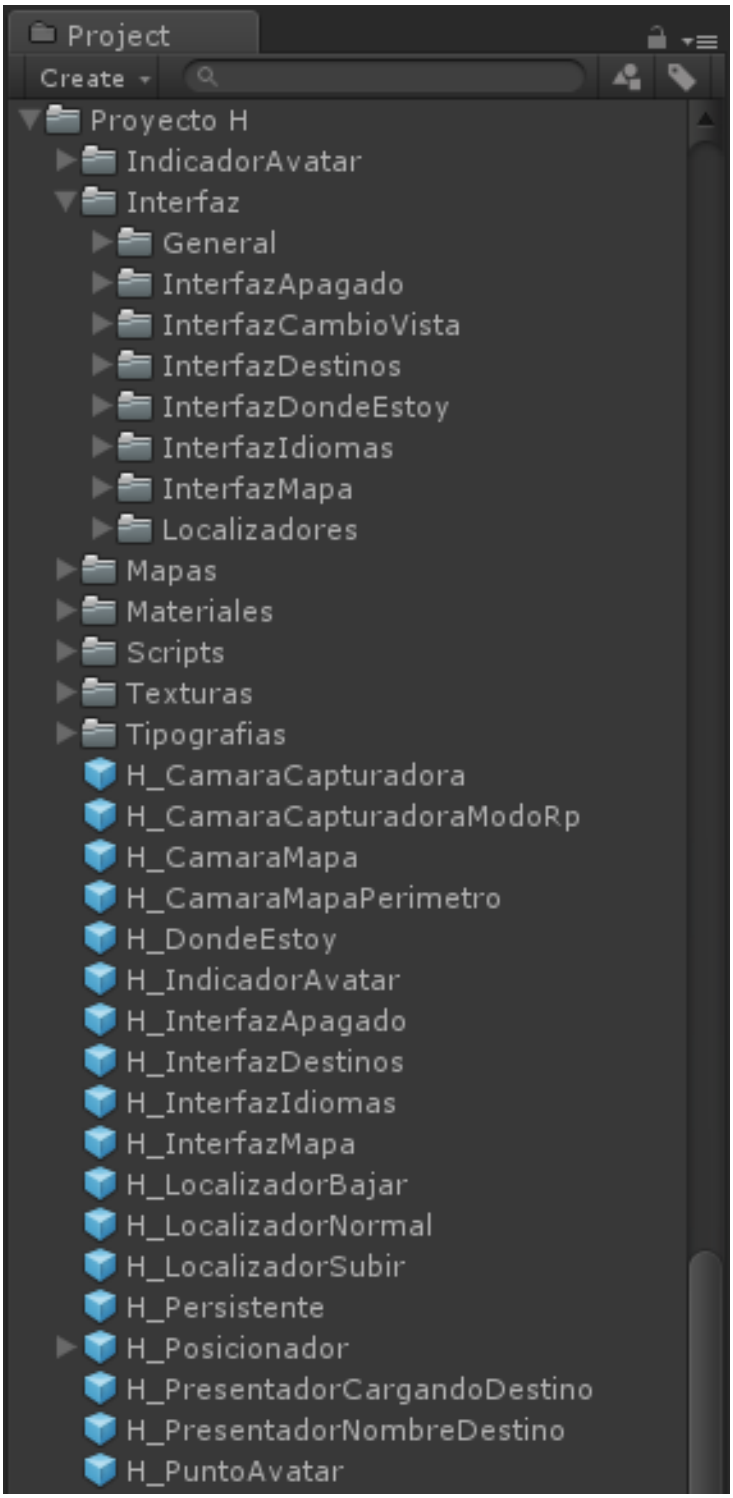


Figura 6.4: Organización del Proyecto

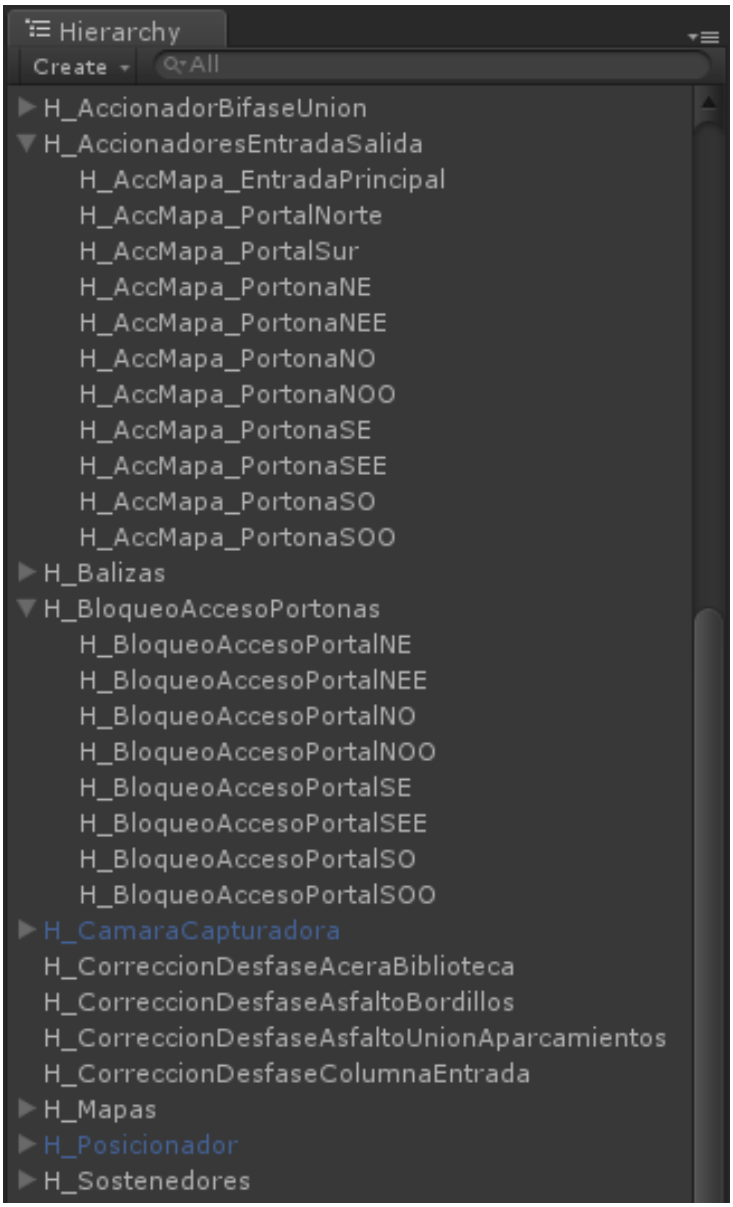


Figura 6.5: Organización de los objetos en las escenas

Los «objetos», las piezas que se añaden a cada escena y que quedan recogidos en la ventana *Hierarchy* correspondiente a cada una, son fácilmente reconocibles por contener todos en su nombre el prefijo «H_», como puede apreciarse en la Figura 6.5.

Capítulo 7

Sistema de Captura de Mapas

El primero de los pasos en la creación de un sistema de navegación, ya sea en el mundo real o en un mundo virtual, consiste en la elaboración de los mapas del terreno.

En el mundo que nos rodea, el físico, se encargan de esta tarea aviones, helicópteros o satélites, y todos ellos tienen un elemento clave en común, la cámara capturadora, el mismo elemento clave alrededor del que orbita el Sistema de Captura de Mapas de un mundo virtual que ahora nos ocupa.

7.1. Componentes

Guiones

- CamaraMapaControl.js
- CamaraCapturadoraControl.js
- CamaraCapturadoraModoRpControl.js

Prefabs

- H_CamaraMapa
- H_CamaraCapturadora

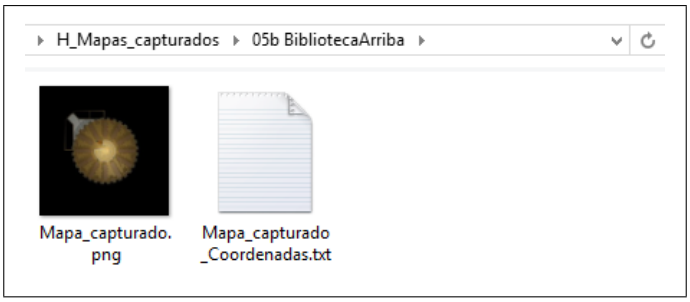


Figura 7.1: Ficheros obtenidos tras una captura

- H_CamaraCapturadoraModoRp

Recursos

- Texturas representativas de los mapas capturados, organizadas en la ventana de proyecto dentro de la carpeta «Proyecto H/Mapas».

7.2. Funcionamiento

La función de este sistema es la de «cartografiar» el terreno del mundo virtual, procesar la imagen capturada y almacenarla posteriormente, de forma organizada, poniéndola a disposición del desarrollador junto con un fichero de coordenadas con información sobre las características de la captura realizada (Figura 7.1).

El elemento principal, al que el desarrollador acudirá para personalizar y ajustar el sistema, es el prefab H_CamaraCapturadora. En su Inspector se encuentran cómodamente centralizadas todas las opciones que permiten modificar y acomodar el comportamiento del Sistema de Captura de Mapas a las necesidades particulares de cada captura.

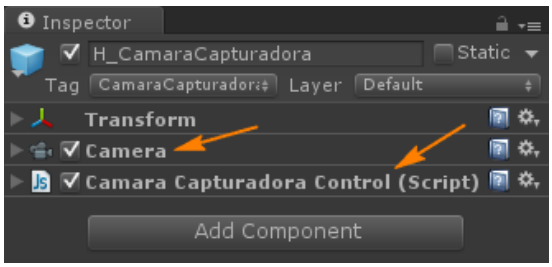


Figura 7.2: Componentes del prefab H_CamaraCapturadora

El prefab H_CamaraCapturadora (Figura 7.2) está constituido por el guión CamaraCapturadoraModoRpControl.js y por la propia cámara capturadora, cuyo comportamiento se encargan de gestionar el primero. Este prefab, cuando el desarrollador active el sistema, se clonará automáticamente en la

escena mediante el guión principal de este CamaraMapaControl.js.

El prefab H_CamaraCapturadoraModoRp y el guión que administra su cámara, CamaraCapturadoraModoRpControl.js, se usa de forma transparente para realizar capturas con el método avanzado RP, que se tratará más adelante.

Se ha cuidado que su uso sea muy cómodo, sencillo y personalizable, pues es la herramienta de partida que usarán futuros desarrolladores para integrar el sistema completo de navegación que este proyecto elabora en el suyo propio.

7.3. Modos de uso

Ofrece dos modos de uso principales, automático y manual, más un tercero, de tanteo, para la realización de pruebas previas a la captura.

En el **Anexo A** se incluye una guía, planteada como material de referencia rápida para el desarrollador, con los pasos para utilizar el Sistema de Captura y crear e integrar rápidamente los mapas que su propio proyecto necesite. Asimismo, el **Anexo B** ofrece una guía totalmente detallada de cada uno de estos pasos. Si en el presente capítulo el énfasis se hace en el aspecto teórico, en los citados anexos la exposición es eminentemente práctica, siendo, en definitiva, complementarios.

Modo automático En este modo el desarrollador indica la altura desde la que desea que la cámara capturadora realice la toma y la cantidad de terreno a capturar. El valor indicado de altura permite tomar imágenes no sólo de exteriores, sino también de interiores, cuando este es lo suficientemente pequeño. Ya sólo queda indicar expresamente que se desea utilizar el modo automático, activando para ello la casilla apropiada (Figura 7.3) en el Inspector de H_CamaraCapturadora, y poner en marcha el proyecto pulsando sobre el botón *Play*. El Sistema de Captura realizará todas las gestiones, controles y comprobaciones, y el desarrollador obtendrá finalmente una imagen representativa del terreno de la escena actual, un mapa, cuyo punto central será la posi-

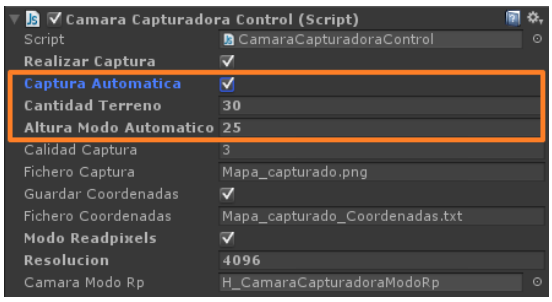


Figura 7.3: Preparativos de una captura automática

ción predeterminada en la que el avatar aparece nada más cargar ésta[1].

Modo manual Al utilizar este modo el desarrollador tiene completa libertad para ubicar la cámara en la escena (Figura 7.4). Esto, además de permitirle refinar al máximo la imagen final obtenida, le abre las puertas a las capturas de detalle (Capítulo 10), capturas de áreas específicas y acotadas inaccesibles de otra forma. El área bajo un tejado que protege la entrada de un edificio, o el espacio que unas escaleras dejan debajo de sí mismas (Figura 7.5), son ejemplos de emplazamientos que el modo manual puede capturar.

Modo de tanteo En este modo el desarrollador ubica la cámara en la escena, desactiva la casilla «Realizar Captura» (Figura 7.6) y, al pulsar el botón *Play*, podrá ver en la ventana de escena, en tiempo real, los límites que determinan la captura. Esto le permitirá ajustar con total precisión las coordenadas de la cámara en escenarios muy específicos, como por ejemplo aquel que requiriera ajustar la altura de la cámara a la altura del avatar en un punto concreto de la escena (Figura 7.7). Conocidas las coordenadas, ya sólo restaría trasladarlas al modo manual para obtener una captura de muy alta precisión.

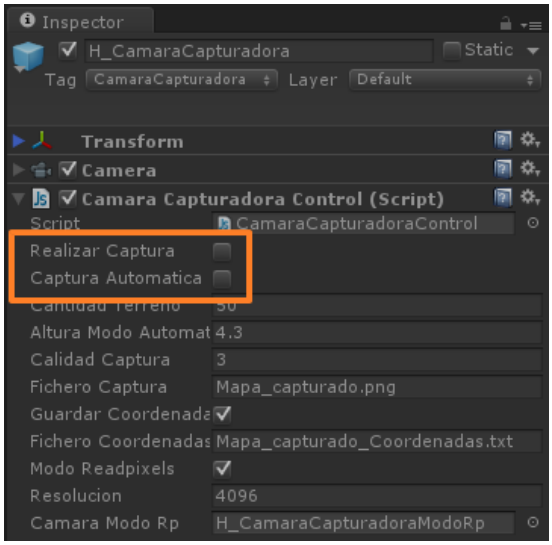
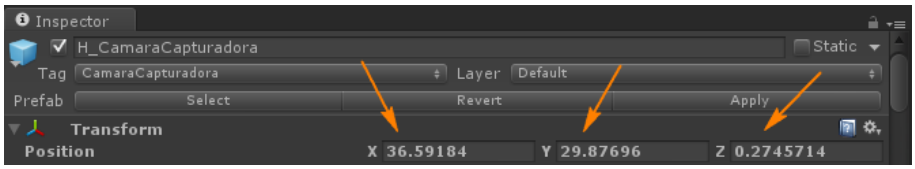


Figura 7.6: Ajustes para activar el modo de tanteo

Finalmente, el mapa obtenido, la imagen generada, se integrará en el proyecto utilizando un plano como lienzo. La posición y tamaño de este plano se añaden al fichero de coordenadas personalizado que acompaña a la imagen nada más realizar la captura, de esta forma el desarrollador no tiene más que introducir los valores indicados y tendrá ya un mapa totalmente operativo y que el Sistema de Posicionamiento, sistema íntimamente relacionado con este, podrá interpretar correctamente.

[1]Posición determinada por la del objeto ControlProj, presente en todas las escenas.

(a) La cámara capturadora, el cuadro blanco, puede moverse en la ventana de escena



(b) Su posición puede refinarse mediante el Inspector

(c) La ventana de juego muestra lo que la cámara ve en realidad, la imagen que se capturará

Figura 7.4: Preparando una captura

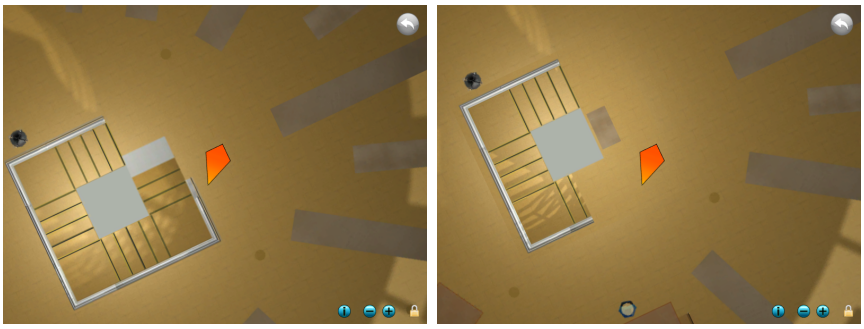


Figura 7.5: Al avanzar el avatar y esconderse las escaleras, descubre una estantería que permanecia oculta

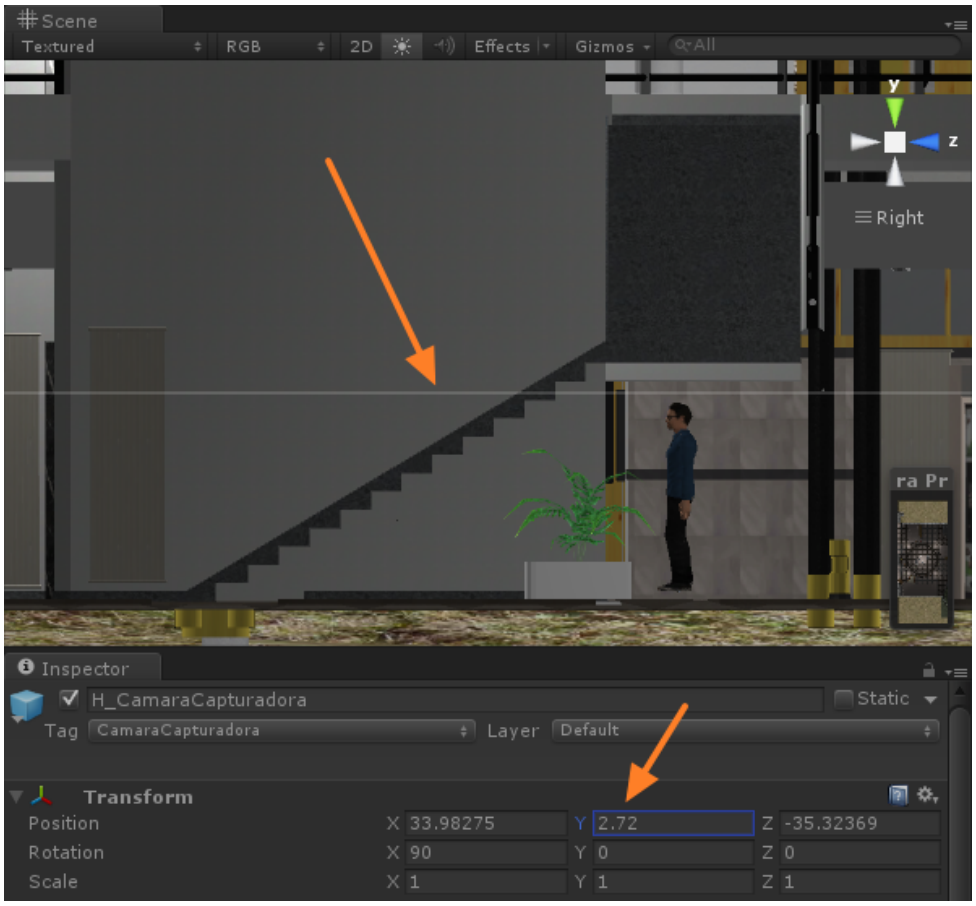


Figura 7.7: Mediante el modo de tanteo se puede calcular con precisión la altura de la captura (línea blanca) con respecto a la del avatar

7.4. Precisión, concordancia y rendimiento

Esta metodología de captura, donde los mapas se obtienen directamente a partir del mundo virtual mediante una cámara apropiadamente configurada para ello, convierten a este Sistema de Captura en uno totalmente preciso y fiable. Métodos alternativos de obtención de mapas, como utilizar la aplicación de modelado con la que se creó el área o edificio a capturar como fuente (Figura 7.8), pueden suponer una vía rápida para lograr este fin, pero nunca se logrará un resultado tan fidedigno, ni una concordancia tan exacta entre las coordenadas del mundo virtual y las del mapa, como el que este sistema consigue; además, si en algún momento posterior se decidiera añadir algún elemento nuevo a la escena, ningún otro método alternativo ofrece la facilidad que ofrece el presente para actualizar y reflejar esos cambios en los mapas[2].

El rendimiento del presente Sistema de Captura de Mapas también supera a sistemas alternativos. Uno al que se recurre con mucha frecuencia es la utilización de una segunda cámara, colocada encima del avatar, que renderice continuamente y en tiempo real, durante toda la ejecución de la aplicación, el terreno por el que transita el avatar. Este método, si bien muy sencillo de implementar dada la prácticamente nula necesidad de gestión que requiere, es muy poco eficiente, fuerza a renderizar por duplicado cada elemento de un mundo virtual que, dada su naturaleza tridimensional, pone ya de por sí una carga intensa en el sistema huésped. Ni qué decir que este método no ofrece tampoco ninguna de las posibilidades que el Sistema de Captura aquí tratado sí ofrece: personalización, mapas de detalles, captura de interiores o ajuste completo del resultado final según las necesidades del desarrollador.

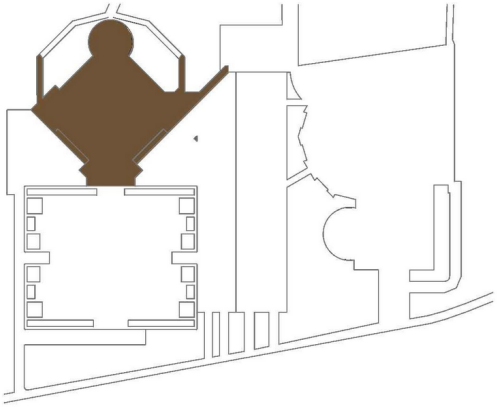


Figura 7.8: Creación del terreno en AutoCAD

El Sistema de Captura de Mapas diseñado en este proyecto garantiza precisión y rendimiento, algo que exige un elevado grado de gestión interna, una tarea llevada a cabo automáticamente por todo el código escrito en los dos guiones que lo administran; de esta forma, todo el aspecto técnico de la captura se vuelve transparente al desarrollador, quien

[2]Bastaría con realizar una nueva captura del área utilizando los mismos valores de configuración que se usó en la captura original, valores convenientemente recogidos en el fichero de coordenadas de esta primera captura.

ya sólo tiene que decidir una cosa, qué es lo que desea capturar, y dejar que esta avanzada herramienta realice el trabajo de precisión que ningún otro método le ofrece.

7.5. Métodos de captura

El sistema incluye dos métodos de captura de mapas: el método estándar ACS (*Application Capture Screenshot*), y el método avanzado RP (*ReadPixels*).

El primero de ellos ofrece resultados adecuados de forma inmediata, y resulta apropiado cuando la calidad de la captura no sea prioritaria o cuando no se disponga de una versión PRO de Unity para poder utilizar el segundo. El método RP garantiza resultados impecables a costa de requerir más tiempo de procesamiento durante la realización de la captura, y de utilizar funciones sólo disponibles en Unity PRO.

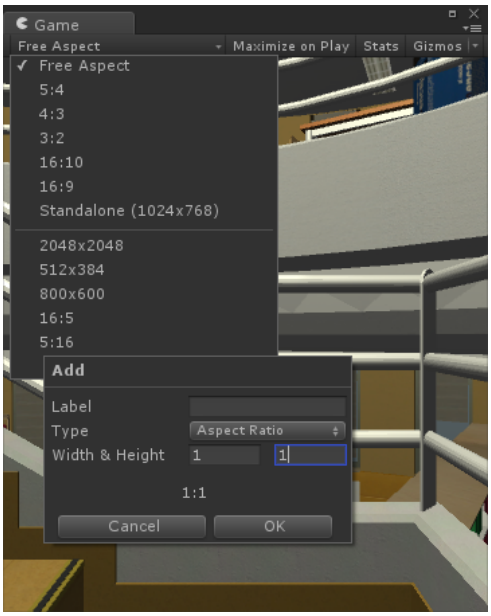
En ambos métodos es recomendable utilizar una resolución con una relación 1:1 en la ventana de juego (Figura 7.9). El Sistema de Captura genera mapas cuadrados, por lo que utilizar una resolución diferente durante la captura añadirá contenido adicional al mapa resultante que habrá que recortar como paso añadido. En caso de utilizar el método avanzado RP, esta recomendación se torna en requisito, y el propio sistema se encarga de avisarlo al desarrollador a través de la consola cuando detecte una resolución incompatible.

La calidad final de la captura puede personalizarse en los dos métodos a través del Inspector de H_CamaraMapa. A mayor calidad, más ocupará la imagen resultante. En ciertos escenarios, a discreción del desarrollador, puede sacrificarse la calidad en favor de un tamaño menor, y en otros se necesita toda la calidad que el sistema pueda entregar. Ambos casos y todos lo intermedios quedan resueltos al ofrecer la opción de elegir la calidad última del resultado modificando el campo «Calidad Captura» para ajustar el método ACS, y el campo «Resolucion»[3] para ajustar el método RP.

El método ACS garantiza buenos resultados por debajo de un valor de calidad cuatro, más allá de este pueden producirse elementos espurios en la captura debido a la naturaleza técnica de este método (Figura 7.10). El método RP no tiene limitación, salvo, acaso, la propia potencia del ordenador huésped, pudiendo requerir tiempos mayores para completar el proceso de captura. Allí donde el método estándar no puede llegar, llega el método avanzado.

[3]Las tildes de la variables utilizadas en el Inspector se han omitido voluntariamente por consistencia entre plataformas.

(a) Listado de resoluciones disponibles



(b) Se pueden añadir nuevas resoluciones pulsando sobre el botón «más»

(c) Ahora la resolución 1:1 está disponible

Figura 7.9: Diferentes resoluciones en la ventana de juego

(a) Detalle de una captura realizada con el método ACS con calidad 4

(b) Detalle de una captura realizada con el método RP en resolución media, 2048

Figura 7.10: Comparación del método ACS con el método RP en el valor límite

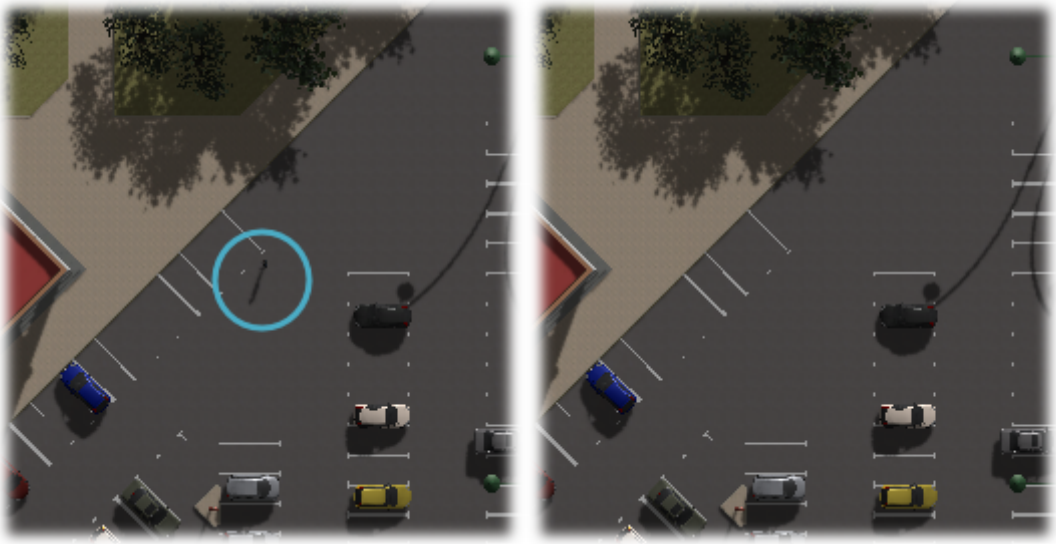


Figura 7.11: Desactivación automática del avatar al capturar el terreno (de no cuidar este aspecto tanto su posición como su sombra aparecería en el mapa final)

En cualquiera de los dos métodos, el Sistema de Captura cuida de todos los aspectos internos: comprobar que los valores personalizables introducidos están dentro de rango, desactivar el avatar en el momento de la captura para no recogerlo en el mapa final (Figura 7.11), gestionar los componentes de la cámara principal de la aplicación para evitar conflictos con la nueva cámara capturadora, o desactivar silenciosamente el modo automático tras su utilización para evitar sobreescrituras no deseadas en caso de olvido son algunos ejemplos de esta gestión interna.

7.6. Configuración y personalización de la captura

A continuación se describirá la función de cada uno de los campos personalizables en el Inspector del prefab H_CamaraCapturadora (Figura 7.12), que el desarrollador puede ajustar para obtener una captura siempre acorde a sus necesidades.

Realizar Captura Control maestro. Activado permite que la captura automática funcione y en el modo manual determina si al cargar la escena se desea realizar una captura. Si durante el modo manual permanece desactivado, permite utilizar el modo de tanteo o de prueba, pudiendo observar y ajustar los límites de la captura en tiempo real antes de llevar a cabo la captura.

Captura Automática Control que determina si se ha de realizar una captura automática con centro ubicado en la posición en la que aparece el avatar en la escena.

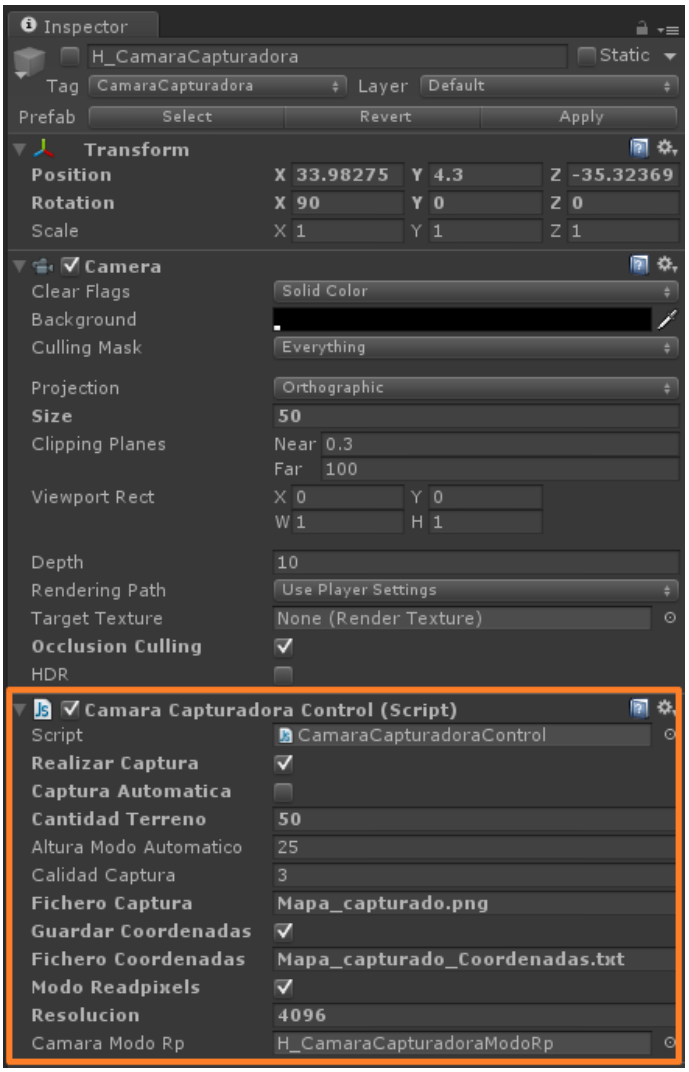


Figura 7.12: Personalización del Sistema de Captura de Mapas

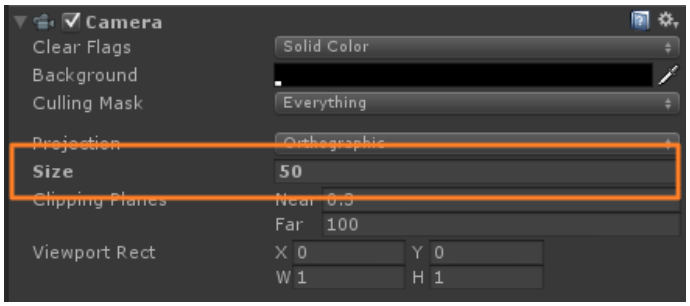


Figura 7.13: Valor *Size* del componente *Camera*

Cantidad Terreno Tamaño en metros de la mitad del lado del cuadrado de terreno que se capturará. Si se desea observar en tiempo real el efecto de esta variable durante la gestión en modo manual de la cámara en la ventana de juego, se podrá modificar el valor *Size* (Figura 7.13) dentro del componente *Camera* en el Inspector, no obstante, será siempre el valor «Cantidad de terreno» el que determinará el tamaño final del terreno capturado. El hecho de ser el valor correspondiente a la mitad del lado y no a la longitud total es para guardar uniformidad con el valor *Size* del componente *Camera* que, por definición, es la mitad.

Altura Modo Automático Altura desde la que se realizará la captura en el modo automático. Modificar este valor permite salvar obstáculos o realizar capturas de interiores (en el modo manual la altura la ajusta el usuario con los controles de movimiento en la ventana de escena.

Calidad Captura Valor entero positivo que determinará la calidad de la captura realizada en el modo de captura estándar ACS (el modo RP o «Readpixels» usará un valor diferente de calidad). Nótese que valores de calidad mayores a 3 pueden introducir elementos espurios por limitaciones de este método.

Fichero Captura Nombre del fichero en el que se guardará la captura realizada.

Guardar Coordenadas Control que determinará si se desea guardar un fichero de texto con las coordenadas en las que se deberá crear el plano que contendrá el terreno capturado.

Fichero Coordenadas Nombre del fichero en el que se guardarán las coordenadas donde crear el plano contenedor del terreno capturado.

Modo Readpixels Control que determinará si se desea usar el modo avanzado de captura o RP. El resultado de la imagen capturada es siempre mejor que el logrado con el método ACS, como contrapartida se necesita una versión PRO del programa Unity

y los tiempos de captura son mayores.

Resolucion Valor que determinará la calidad de la imagen capturada con el método ReadPixels. Valores útiles: 1024, 2048, 4096 (otros valores son posibles pero menos prácticos). Nótese que cuanto mayor sea el terreno a capturar mayor deberá ser la resolución si la calidad de la captura es importante.

Camara Modo RP Variable donde se indicará el prefab encargado de gestionar el método de captura avanzado RP, por defecto H_CamaraCapturadoraModoRP.

7.7. Mapas mejorados con programas de edición fotográfica

Los mapas que se han creado en el seno de este proyecto han sido editados minuciosamente mediante programas de edición fotográfica para optimizar la experiencia inmersiva, y corregir también cualquier imperfección que, presente en el mundo virtual, se hubiera trasladado por consecuencia a su mapa representativo en el momento de capturarlo.

7.7.1. Composición de capturas

Se ha recurrido frecuentemente a la técnica de composición de capturas. Realizando varias tomas de una misma área a diferentes alturas, se puede, mediante un programa de edición fotográfica, recoger de una los elementos que no aparecen en otra o, por ejemplo, quedarse con aquellos que en alguna aparecen mejor iluminados, para después fusionar todos los elementos seleccionados y obtener finalmente un mapa perfecto que combina lo mejor de cada toma.

7.7.2. Aislamiento de áreas

Se han utilizado también varias técnicas de aislamiento de áreas.

Mediante la selección minuciosa del contorno delimitante de un área concreta Figura 7.14), se logra separarla de su entorno, aumentando así la experiencia inmersiva del usuario. Esta técnica se ha utilizado en mapas de áreas que por sí mismas suponen un todo y no constituyen una parte de otra área mayor, como pueden ser las distintas plantas de un edificio.

Otra técnica de aislamiento, utilizada en esta ocasión para aquellas áreas que suponen una sección de otra más grande, consiste en añadir una capa difuminadora alrededor del perímetro de unión que las vincula con el área a la que pertenecen. Esta técnica se ha usado, por ejemplo, en aulas o habitaciones, ubicadas siempre en una determinada planta de un edificio que las contiene.

Por último, todas estas técnicas se han utilizado de forma combinada para garantizar siempre el resultado visual óptimo.

En la Figura 7.15 se puede apreciar esquemáticamente un ejemplo sencillo de una secuencia de trabajo en la que se aplican las técnicas mencionadas. En la fila de imágenes superior se observa cómo a partir de dos tomas realizadas a distintas alturas se obtiene una tercera, en la que, a los elementos de la primera, se le añaden los muros de la segunda, omitiendo los puntos de luz que esta última capturó también. En la fila de imágenes inferior se observa el refinamiento secuencial de los muros (coloreados en este ejemplo para ayuda a la visualización), hasta obtener exclusivamente el perímetro de la estancia, delimitando asimismo los dinteles de las puertas. Finalmente, en la última de las imágenes, la técnica de aislamiento difumina el contorno exterior del área, logrando que, cuando el Sistema de Posicionamiento interprete el mapa, este se mimetice con el fondo y el resultado visual sea óptimo.

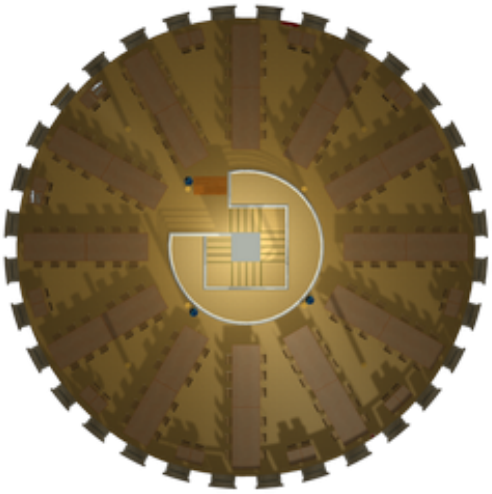


Figura 7.14: Edición minuciosa del perímetro de un área

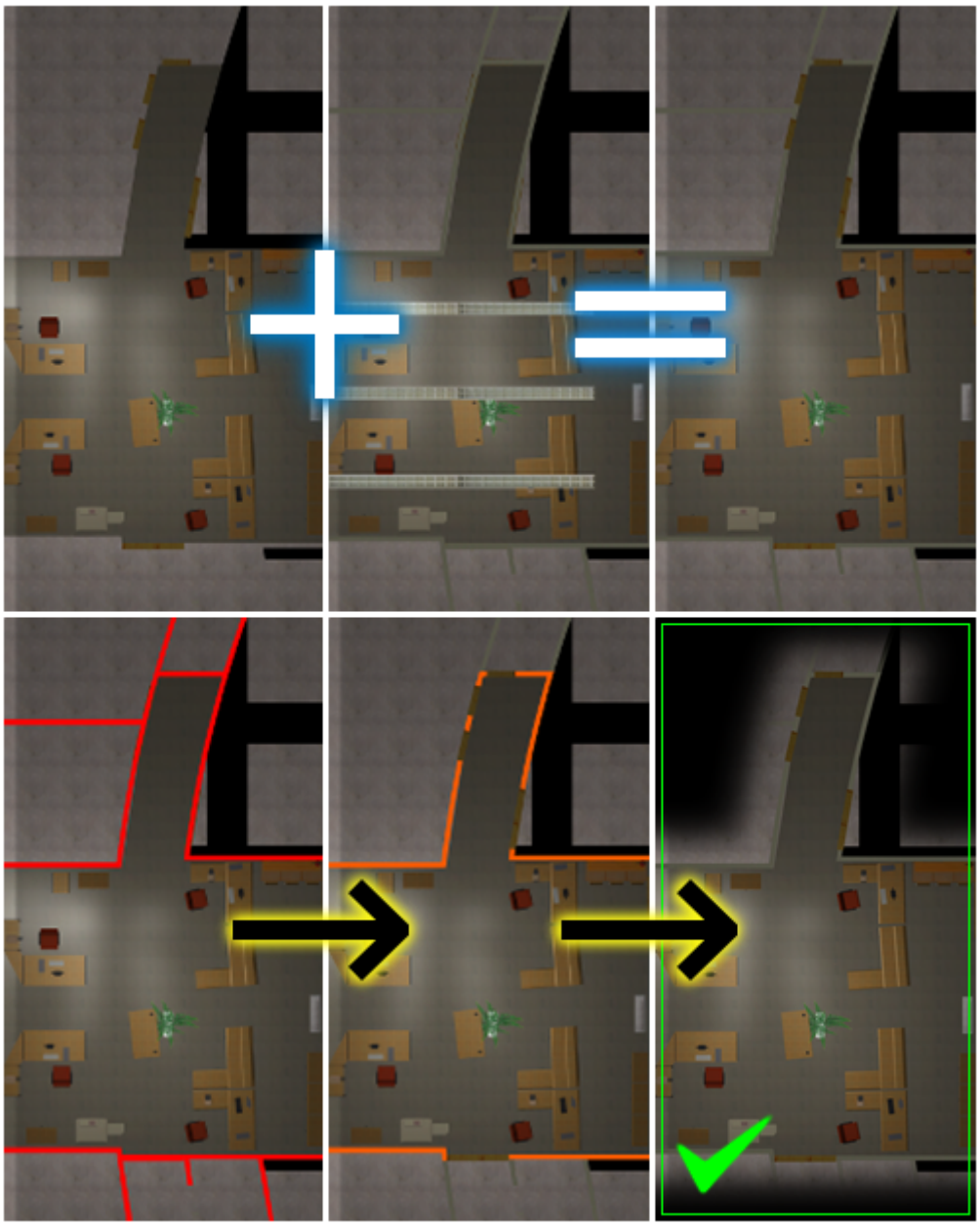


Figura 7.15: Secuencia de trabajo en la edición de mapas

Capítulo 8

Subsistema de Creación de Planos de Alto Rendimiento

En la Sección 7.3 se adelantó cómo el paso final para integrar en cualquier proyecto un mapa creado por el Sistema de Captura consistía en usar un plano como soporte. Este plano puede perfectamente crearse con las herramientas básicas que Unity ofrece para este fin, y ajustar posteriormente sus dimensiones a las indicadas en el fichero de coordenadas que acompaña a cada mapa cartografiado. Este tipo básico de planos, no obstante, ofrece un grado de rendimiento que en este contexto puede mejorarse, y este proyecto ofrece la herramienta para lograrlo.

Los planos que Unity utiliza para la creación de entornos tridimensionales están a su vez compuestos de una cantidad concreta de triángulos más pequeños. El número de estos triángulos determina aspectos como el realismo de la iluminación cuando un punto de luz incide sobre uno de estos planos. Esta cantidad de triángulos, en los planos básicos, equivale a doscientos, una cifra innecesariamente elevada cuando su función va a ser la de servir de soporte para un mapa (Figura 8.1).

Este proyecto, buscando la optimización del rendimiento, incluye una herramienta de creación de planos personalizados, el guión CreadorDePlanos.cs[1], más avanzada que la que propone Unity de serie y diseñada de forma que permite la creación de planos compuestos por tan sólo dos triángulos, el mínimo posible.

Los planos así creados suponen una mejoría en el rendimiento de la aplicación final, reduciendo la complejidad y el número de elementos que se han de evaluar, procesar y renderizar durante su ejecución.

[1]Código base por Michael Garforth[1] (http://wiki.unity3d.com/index.php/User:Mike) bajo licencia *Creative Commons Attribution-ShareAlike* (http://creativecommons.org/licenses/by-sa/3.0); modificado, mejorado y adaptado para este proyecto por el que escribe y autor del mismo.

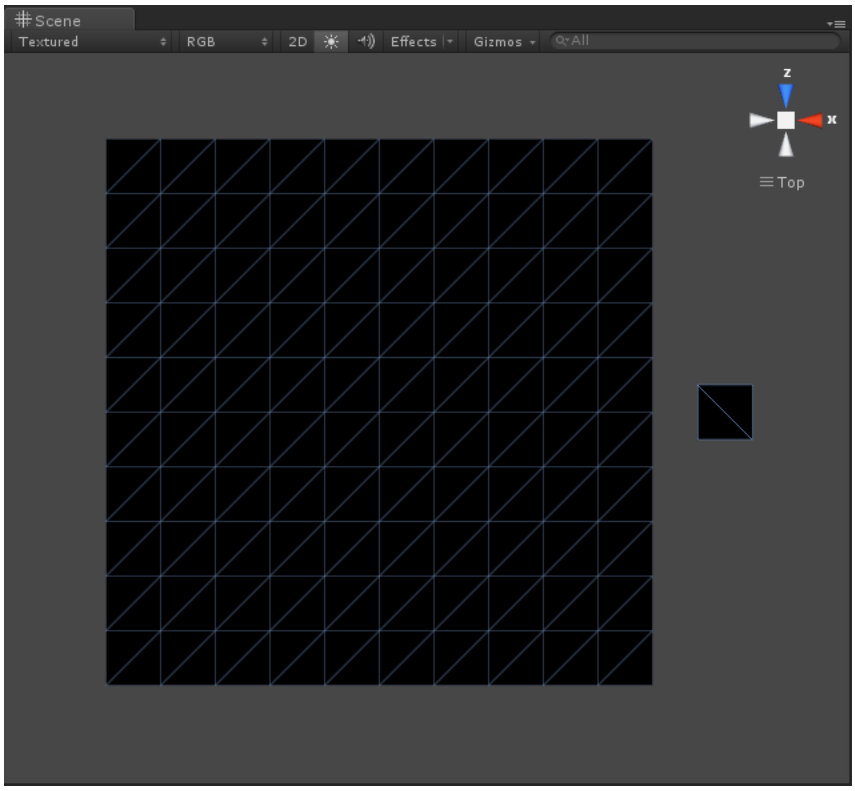


Figura 8.1: Comparación de un plano normal con uno de alto rendimiento

8.1. Componentes

Guiones

- CreadorDePlanos.cs[2]

8.2. Herramienta de uso general

La herramienta presente no está limitada a la creación de planos contenedores de aquellos mapas generados por el Sistema de Captura, es una herramienta de uso general, flexible y personalizable, pudiendo utilizarse sus planos para cualquier tarea en la que deba obtenerse el máximo rendimiento posible[3].

[2] La terminación «cs» indica que el guión está escrito en el lenguaje de programación C#.

[3] Las versiones más recientes de Unity incluyen un objeto de tipo *Quad*, un tipo especial de plano conformado también por dos triángulos. Similares a los que aquí se tratan pueden servir de alternativa, prescindiendo no obstante del grado de flexibilidad y personalización que estos últimos sí aportan.

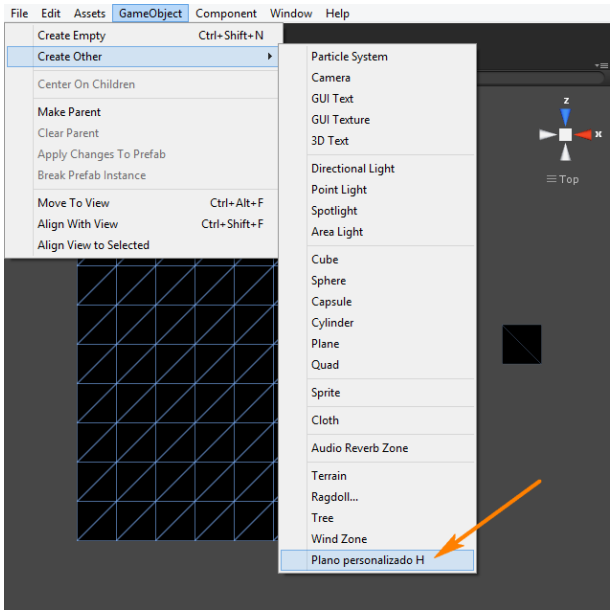


Figura 8.2: Ruta hacia los planos personalizados

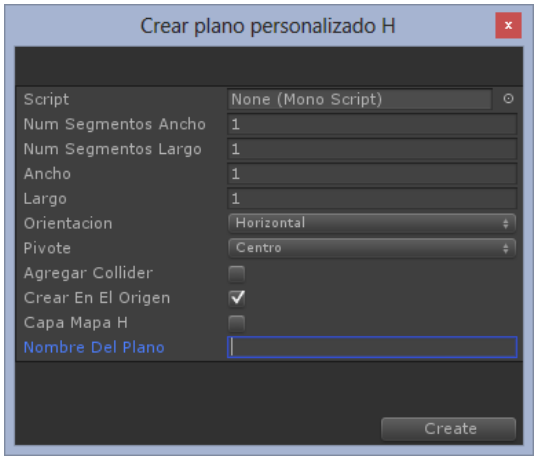


Figura 8.3: Menú de creación de planos de alto rendimiento (planos personalizados H)

8.3. Planos personalizados

El uso de esta herramienta se ha configurado para que el desarrollador disponga fácilmente de ella en *GameObject* ▷ *Create Other* ▷ *Plano personalizado H* (Figura 8.2). Una vez ejecutada, le mostrará a este un menú con las opciones de personalización para la creación de los planos disponibles, como puede apreciarse en la Figura 8.3.

Las opciones disponibles son:

Script Permite indicar el nombre de un guión que se añadirá automáticamente a los componentes del plano inmediatamente después de su creación.

Num Segmentos Ancho Indicador de la cantidad de elementos que compondrán el plano a lo ancho. Un valor de «1» logra la mayor optimización.

Num Segmentos Alto Indicador de la cantidad de elementos que compondrán el plano a lo alto. Un valor de «1» logra, una vez más, la mayor optimización.

Ancho Longitud del plano a lo ancho en metros.

Alto Longitud del plano a lo alto en metros.

Orientación Valor que determinará si el plano se creará horizontal o verticalmente respecto del plano X-Z.

Pivote Valor que determinará la posición del pivote que servirá como centro para ubicar el plano posteriormente en la escena.

Agregar Collider Opción que permitirá agregar automáticamente al plano un componente de tipo *Collider*[4] inmediatamente tras su creación.

Crear En El Origen Al activar esta casilla el plano se creará inicialmente en las coordenadas (0, 0, 0).

Etiqueta Mapa H Al activar esta casilla se le añadirá al plano que se creará la etiqueta «Mapa» (utilizada por el Sistema de Posicionamiento).

Nombre Del Plano Nombre que se le dará al plano una vez creado. Aparecerá con este nombre en la ventana de jerarquía, también en la ventana de proyecto, en la carpeta «Planos Personalizados H», con un sufijo añadido automáticamente que representará las características física del plano: número de segmentos, anchura y altura, orientación y posición del pivote.

En los **Anexos A y B** se indican los valores que han de adoptar estas opciones para utilizar esta herramienta como extensión del Sistema de Captura de Mapas.

[4]Un *Collider* es un tipo de componente que, en el entorno tridimensional, permite detectar colisiones entre objetos que disponen de él.

Capítulo 9

Sistema de Posicionamiento

El Sistema de Posicionamiento es el encargado de interpretar los mapas generados por el Sistema de Captura, de mostrarlos convenientemente en pantalla, de ubicar en ellos la posición exacta actual del avatar del usuario, y de generar la interfaz que le permitirá a éste interactuar cómodamente con el sistema, pudiendo personalizar y ajustar la visualización de los mapas completamente a su gusto o necesidades.

Nada más cargar una escena, en cuanto el usuario a accedido a este mundo virtual, el sistema de posicionamiento le presenta, en una de las esquinas de la pantalla, un pequeño mapa configurable con un indicador de su posición actual. A medida que el usuario desplaza a su avatar por el mundo virtual, el contenido del mapa cambia para reflejar, continuamente y de manera exacta, su posición.

El usuario sabe, con sólo mirar el mapa, dónde se encuentra en todo momento. Aunque fuera la primera vez que éste accede a una determinada localización, la vista aérea que le ofrece el mapa del lugar le permite conocer de antemano todas las características de su entorno, distancias, perímetros, salidas y entradas, toda la información que necesita para orientarse con precisión y sin ningún miedo en un nuevo entorno que ya no tiene misterio alguno para él (Figura 9.1).

Figura 9.1: El mapa le permite al usuario saber qué hay detrás de la puerta antes de cruzarla

9.1. Componentes

Guiones

- CamaraMapaControl.js
- CamaraMapaSeguidor.js
- InterfazMapaControl.js
- InterfazMapaGrandeControl.js
- IndicadorAvatarSeguidor.js
- PersistenteControl.js
- SDN.js

Prefabs

- H_CamaraMapa
- H_InterfazMapa
- H_IndicadorAvatar
- H_Persistente

Objetos

- Planos de alto rendimiento, contenedores de las texturas de los mapas cartografiados, agrupados en las escenas bajo el prefijo común «H_Mapa_» y pertenecientes a la capa o *Layer* «Mapa».

Recursos

- Textura del prefab H_IndicadorAvatar en la ventana de proyecto dentro de la carpeta «Proyecto H/IndicadorAvatar».
- Texturas constitutivas de la interfaz, organizadas en la ventana de proyecto dentro de la carpeta «Proyecto H/Interfaz/InterfazMapa» y «Proyecto H/Interfaz/General».

9.2. Funcionamiento

El guión CamaraMapaControl.js es el punto de partida de este sistema, el encargado de activar cada uno de los elementos que lo componen en cuanto se carga una nueva escena, y de ofrecerle al desarrollador, a través del Inspector del prefab H_CamaraMapa, todas las numerosas opciones de personalización que este sistema le da opción a modificar.

El prefab H_CamaraMapa, además del citado guión, contiene una cámara que se moverá automáticamente según lo haga el avatar del usuario. Una vez cargado en la escena, replicará la posición del avatar donde sea que este se encuentre. De esta forma se puede crear una correspondencia entre las coordenadas de la localización del avatar y el punto al que corresponderían en el mapa. La cámara está configurada para que, de todos

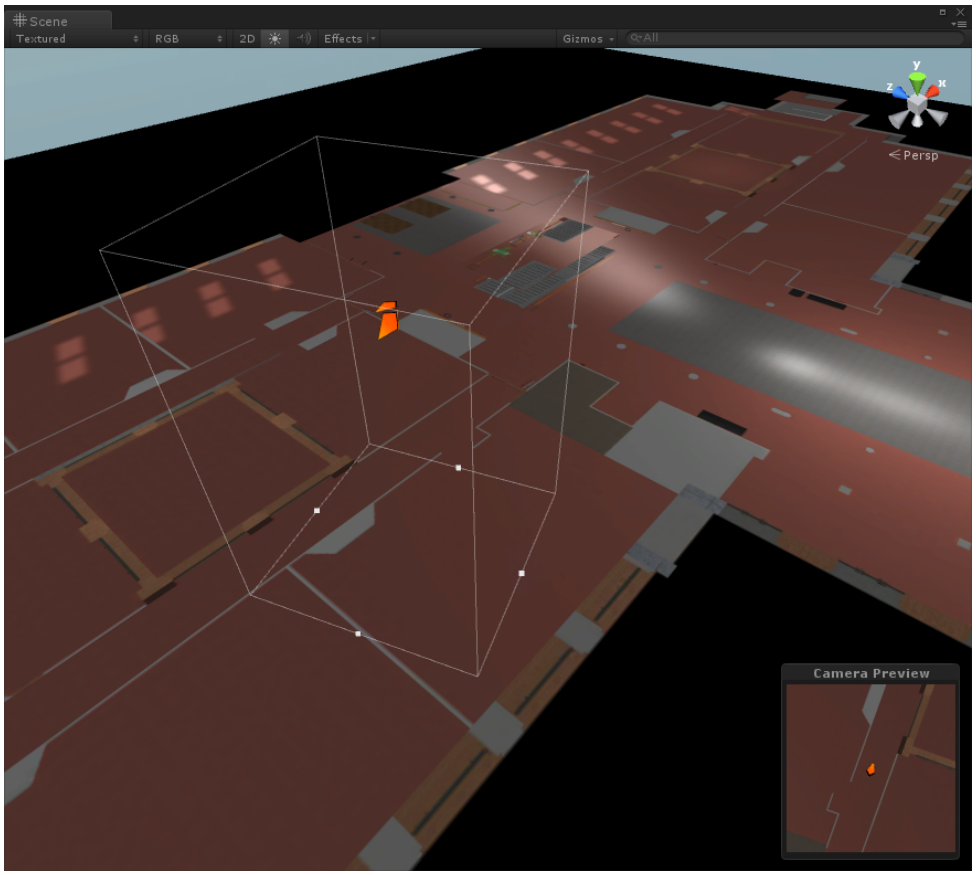


Figura 9.2: La cámara del Sistema de Posicionamiento, en blanco, observa un área del mapa

los elementos del mundo virtual, tan solo observe aquellos marcados previamente con la etiqueta «Mapa» (Figura 9.2), que definirá, con su nombre indica, a todas las texturas cuya función sea la de servir, precisamente, de mapas. Configurada la cámara de esta manera, el siguiente paso es colocar la textura que servirá de mapa de tal forma que, al moverse la cámara, el trozo de mapa que observe se corresponda con el área en el que se encuentra el avatar. La posición exacta donde colocar la textura queda indicada en el fichero de coordenadas que el Sistema de Captura genera junto con la imagen capturada: las coordenadas X y Z quedan determinadas por la posición de la cámara capturadora en el momento de cartografiar el terreno, y la cota, o coordenada Y, estará predefinida en -100, guardando una relación ordenada con las cotas del resto de los elementos que constituyen el Sistema de Posicionamiento[1] (Figura 9.3).

Configurados los mapas y alineada la cámara con la posición del avatar, sólo queda ya colocar y configurar un último elemento para vincular totalmente la posición del avatar en el mundo virtual con su correspondiente en los mapas: el prefab H_IndicadorAvatar, que se posiciona automáticamente, mediante las instrucciones de su guión controlador IndicadorAvatarSeguidor.js, entre la cámara y los mapas y alineado con el avatar (Figura

[1]También con los elementos del Sistema de Navegación Guiada como se verá en el Capítulo 14.

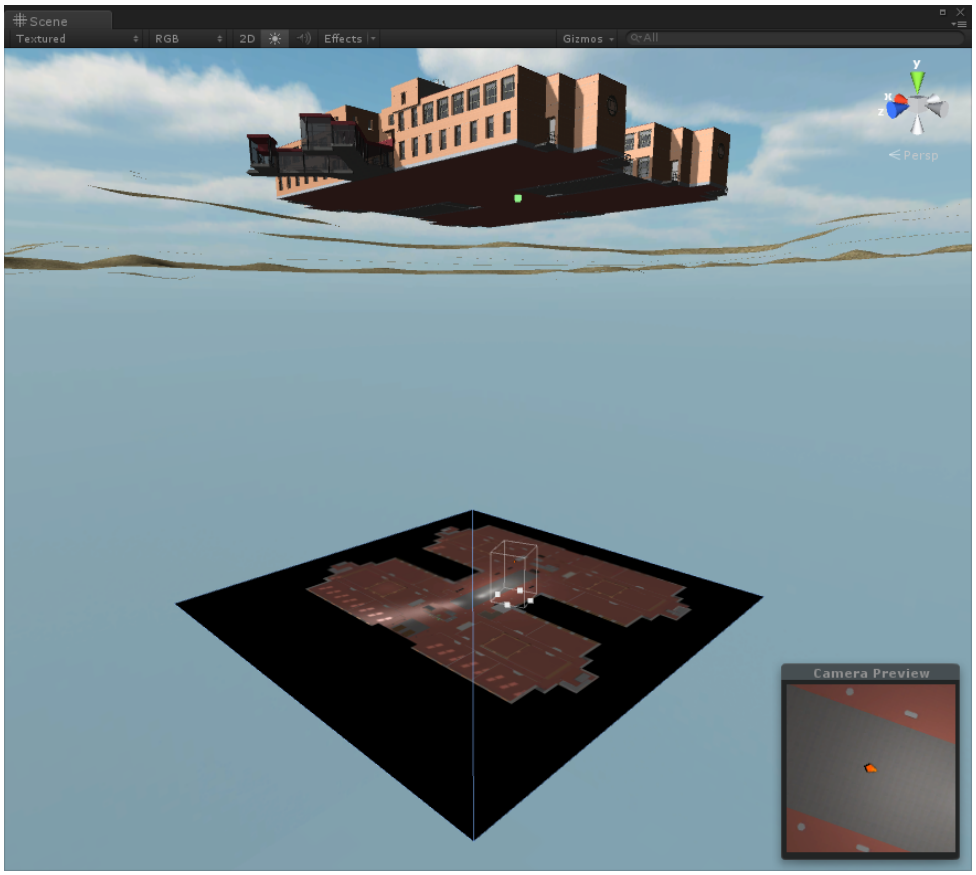


Figura 9.3: Una vez posicionado el mapa, es una proyección del mundo virtual

9.4). Posee la etiqueta «Mapa», como las texturas cartografiadas, lo que significa que la cámara también lo verá, logrando así en la práctica proyectar visualmente la ubicación del avatar sobre los mapas (Figura 9.6a).

Ya sólo resta presentarle al usuario el resultado de la combinación precisa de todos estos elementos. El prefab H_CamaraMapa se encargará de esto, reservando un área configurable de la pantalla para mostrar en ella el trozo de mapa del terreno en el que ahora se encuentra su avatar (Figura 9.6b); y sobre el mapa, el indicador visual de la posición del avatar, el prefab H_IndicadorAvatar o, más concretamente, su textura[2], y que puede verse con detalle en la Figura 9.5.

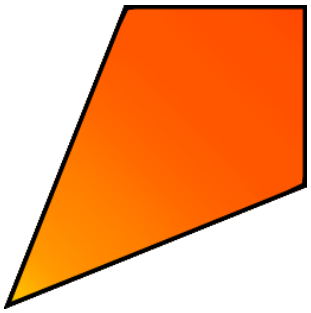


Figura 9.5: Indicador del avatar

Finalmente el prefab H_InterfazMapa se encarga de embellecer todo el conjunto, dibujando marcos, iconos

[2]Este prefab es, originalmente, un cubo, pero con una textura apropiada se le puede dar la apariencia de una punta de flecha que sirva para indicar visualmente hacia dónde está mirando el avatar, es decir, su orientación.

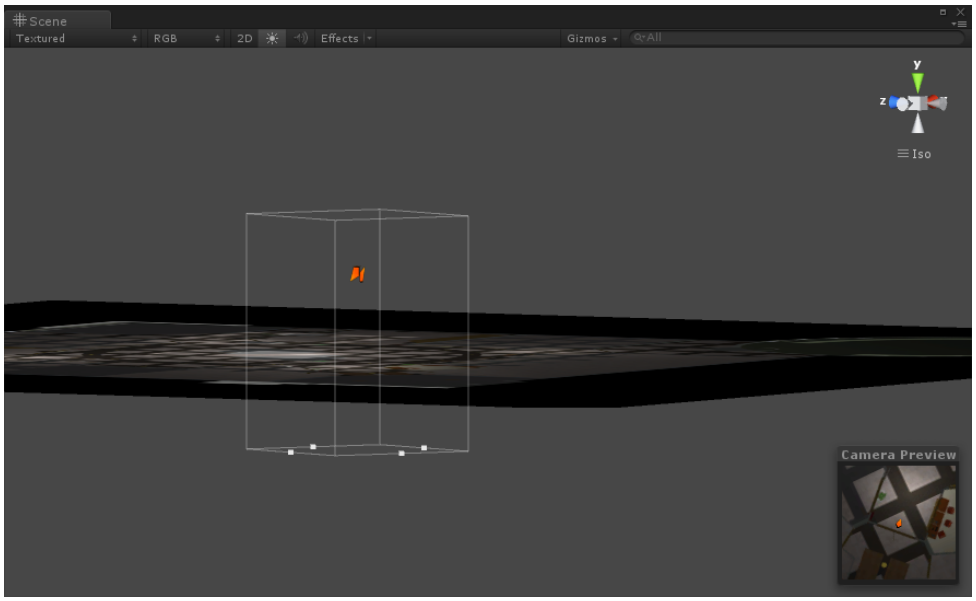


Figura 9.4: El indicador del avatar se posiciona encima de los mapas, la cámara (cubo blanco) observa a ambos desde arriba

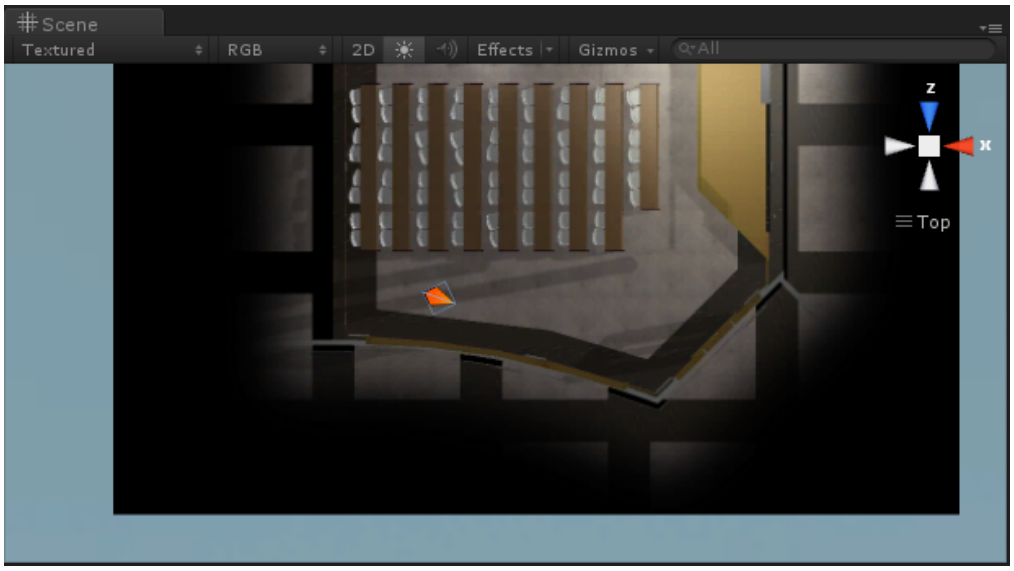
informativos y botones que permitirán al usuario interactuar con el sistema tal y como puede apreciarse, ampliado, en la Figura 9.7.

9.3. Posicionamiento y orientación de la cámara

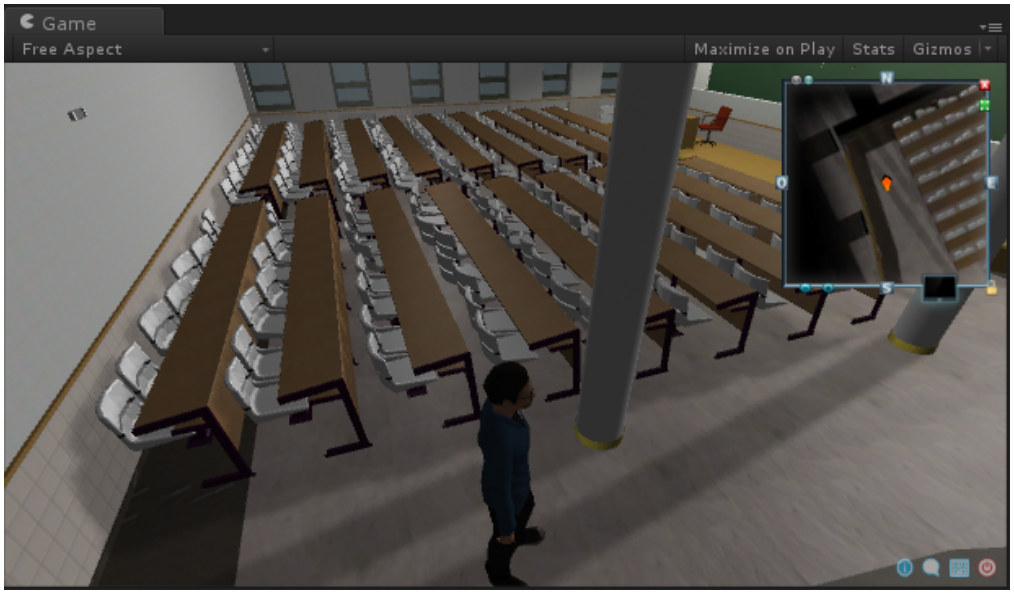
Como se ha mencionado ya, la cámara observadora del mapa replica la posición del avatar. Esto se consigue mediante el código del guión CamaraMapaSeguidor.js, asociado también al prefab H_CamaraMapa, y con instrucciones para moverlo y orientarlo en tiempo real según se mueva el avatar.

9.3.1. Movimiento suavizado:

Cada uno de los movimientos que ordena este guión se procesan internamente para que, cuando la cámara se mueva de un punto inicial a uno destino, este desplazamiento discreto se realice de forma suavizada y fluida. Sin estos cálculos previos el resultado sería un movimiento a trompicones, en términos matemáticos la cámara se desplazaría usando números enteros, pasaría directamente del 1 al 2 produciendo un salto visual. El guión CamaraMapaSeguidor.js introduce los cálculos que permiten que se mueva utilizando números racionales, ahora la cámara se mueve del 1 al 2 pasando por todos los decimales intermedios, garantizando así un movimiento fluido y visualmente muy agradable.



(a) Internamente, la posición del avatar se proyecta sobre el mapa mediante el prefab H_IndicadorAvatar



(b) Externamente, el prefab H_CamaraMapa combina todos los elementos y, con la ayuda de H_InterfazMapa, los presenta vistosamente en pantalla

Figura 9.6: Relación entre los elementos del Sistema de Posicionamiento

Figura 9.7: Marco, botones y elementos informativos alrededor el mapa

```
1 transform.eulerAngles.y = Mathf.LerpAngle(rotacionActualY, rotacionDeseadaY, Time.
      deltaTime * suavizadoDeMovimiento);
```

Cuadro 9.1: Corazón del movimiento suavizado

La clave para el movimiento suavizado es la función `Math.LerpAngle`[3], capaz de interpolar entre dos valores determinados dados y que, gestionada correctamente, abre las puertas de este vistoso y agradable efecto.

9.3.2. Seguimiento del avatar

En Unity, obtener la posición actual de cualquier objeto en la escena es una tarea sencilla. Todos ellos tienen un componente *Transform* que determina en tiempo real las coordenadas de su ubicación. Sabiendo esto, un objeto que deba seguir a otro ha de acudir, en primer lugar, al componente *Transform* del mismo. Así, para que la cámara observadora del mapa pueda replicar la posición del avatar, puede utilizarse el *Transform* de éste para asignar la posición de la primera.

[3]Existe otra función similar, `Math.Lerp`, con una importante diferencia, esta no interpola correctamente valores que disten más de 360°entre sí.

```
1 // Se recoge la posición de un objeto.
2 objetivo = SDN.avatar.transform;
3 // Se asigna su posición a otro.
4 transform.position = objetivo.position;
```

Cuadro 9.2: Código esquemático de asignación de posición

9.3.3. Orientación

El Sistema de Posicionamiento ofrece dos modos de orientar el mapa, una fijada al norte geográfico y otra determinada por la propia orientación del avatar.

9.3.3.1. Orientación fijada al norte

En este modo es el norte geográfico el que determina la orientación del mapa, que permanecerá fija, apuntando siempre al norte, mientras que el indicador representativo del avatar en el mapa rota para ajustarse continuamente a la orientación de este. En este modo el usuario conoce en todo momento hacía qué punto cardinal avanza, haciéndose una idea precisa de la disposición espacial de su entorno.

Este tipo de orientación es posible gracias al código del guión CamaraMapaSeguidor.js. Para ello corrige inicialmente el desfase existente entre la orientación de los ejes de coordenadas de una escena de referencia con respecto al norte, y a partir de aquí, corrige el desfase del resto de escenas. La escena tomada como referencia en este caso es la escena «01 Exterior», correspondiente al terreno que rodea a la Escuela de Ingenierías y el edificio tecnológico. Este desfase inicial se ha calculado manualmente (Figura 9.9), comparando mapas reales por satélite del terreno con los generados por el Sistema de Captura a partir del mundo virtual, y su valor es de -109,8°. Esta cifra, que puede introducirse y modificarse cómodamente en la configuración del sistema (Figura 9.8) mediante el Inspector de H_CamaraMapa, será la utilizada como referencia para compensar la orientación de todas las escenas creadas, tanto presentes como futuras.

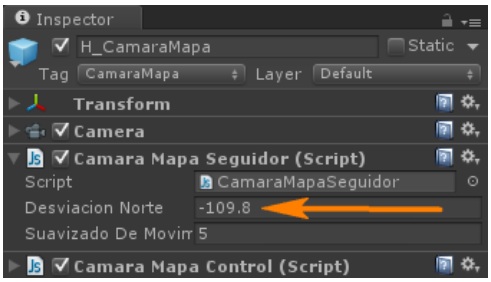


Figura 9.8: Configuración del norte

Las escenas creadas con anterioridad a este proyecto se construyeron sobre los ejes de coordenadas sin guardar una coherencia global. Así, si bien la escena exterior tiene un desfase con respecto al norte geográfico de -109.8°, el resto de escenas, correspondientes a aulas, laboratorios, habitaciones, etcétera, se posicionaron sobre los ejes X y Z de forma aleatoria, por lo que no guardan la relación natural esperada entre sí. Si el Sistema de Posicionamiento no tuviera esto en cuenta, el resultado sería que, al entrar el avatar en un aula cuya entrada aparece ubicada al norte en el mapa, cuando se cargara la nueva escena correspondiente a esta aula, la orientación con la que entró el avatar en ella cambiaría al estar ya dentro de ella. Esto confundiría al usuario y anularía la precisión del sistema. Para

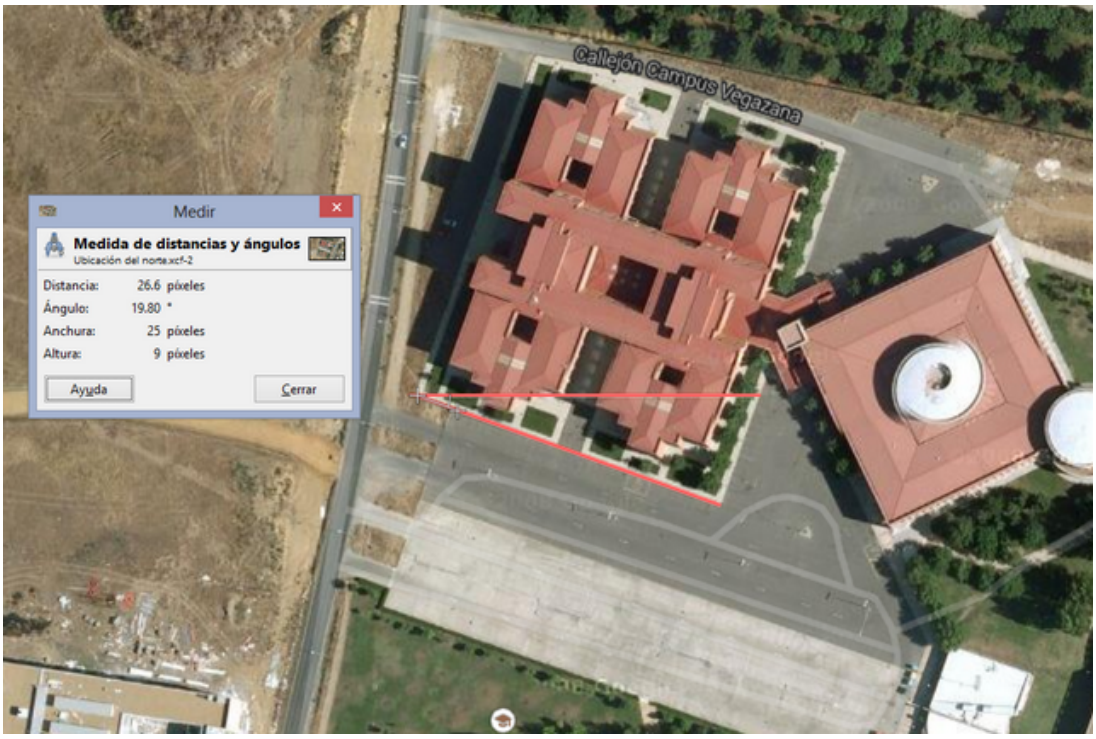


Figura 9.9: Cálculo del norte geográfico

evitar esta situación sin modificar la integridad original de las escenas creadas, el Sistema de Posicionamiento incluye en su código una sencilla base de datos, que incluye todos los desfases de cada una de las escenas anteriores a este proyecto con respecto a la escena de referencia «01 Exterior». De esta forma, si el avatar avanza ahora en dirección norte y cambia de escena, aparecerá en la nueva, como cabría esperar, orientado también al norte. Así, todas y cada una de las escenas, a pesar de haberse creado de forma individualizada sin atención a las demás, guardan, a través del Sistema de Posicionamiento, una coherencia virtual exacta entre sí.

9.3.3.2. Orientación fijada por la del avatar o libre

En este modo la orientación del indicador del avatar en el mapa permanece fija, siendo el propio mapa el que rota cuando rota el avatar.

Un modo muy efectista visualmente, permite al usuario olvidarse de los puntos cardinales y concentrarse siempre en la dirección de avance, punto hacia el que el mapa rotará continuamente según el usuario haga rotar a su avatar.

Los movimientos del mapa también se han suavizado de forma que, cuando el avatar gira, el mapa lo sigue con un movimiento amortiguado muy agradable a la vista.

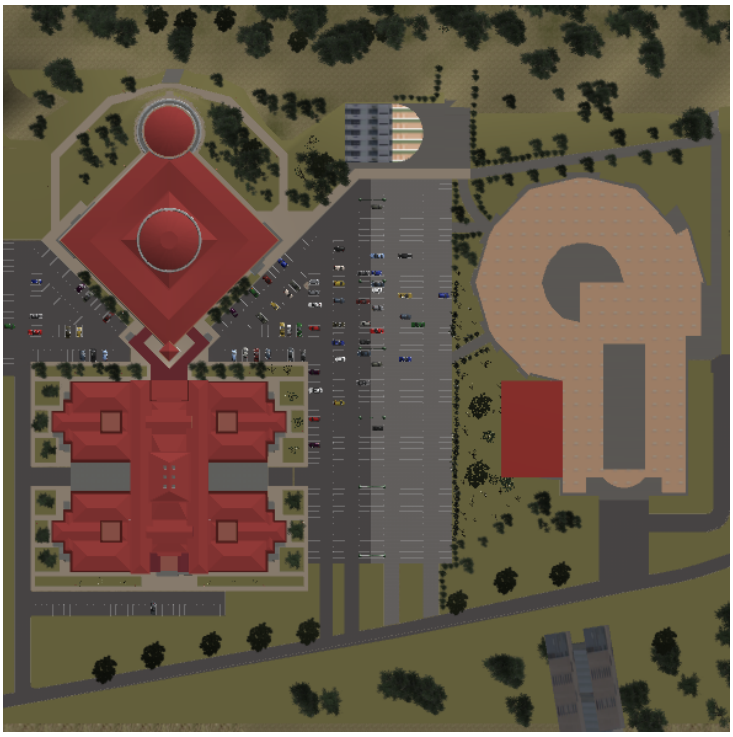


Figura 9.10: Orientación original de la escena de referencia, hay que girarla -109,8°para orientarla al norte geográfico

Puede apreciarse la diferencia entre ambos modos de orientación en la Figura 9.11, la posición del avatar no ha cambiado, el mapa sin embargo se ajusta automáticamente al norte geográfico en la imagen de la izquierda, y a la dirección de avance del avatar en la imagen de la derecha. Nótese que la interfaz cambia para reflejar el modo de orientación activado, aspecto que se tratará en la Sección 9.5.

9.4. Modos de visualización

El Sistema de Posicionamiento ofrece dos modos diferentes de visualización de los mapas, mapa normal o minimapa y mapa a pantalla completa, más un tercero correspondiente al mapa cerrado o minimizado.

9.4.1. Mapa normal o minimapa

Este es el modo estándar y el que ya se ha podido apreciar en las figuras anteriores: el mapa aparece en la esquina superior derecha o izquierda, según se haya configurado, y muestra el mapa del terreno inmediato al avatar. Tanto su posición, como se acaba de adelantar, como su tamaño pueden personalizarse tal y como se verá más adelante.

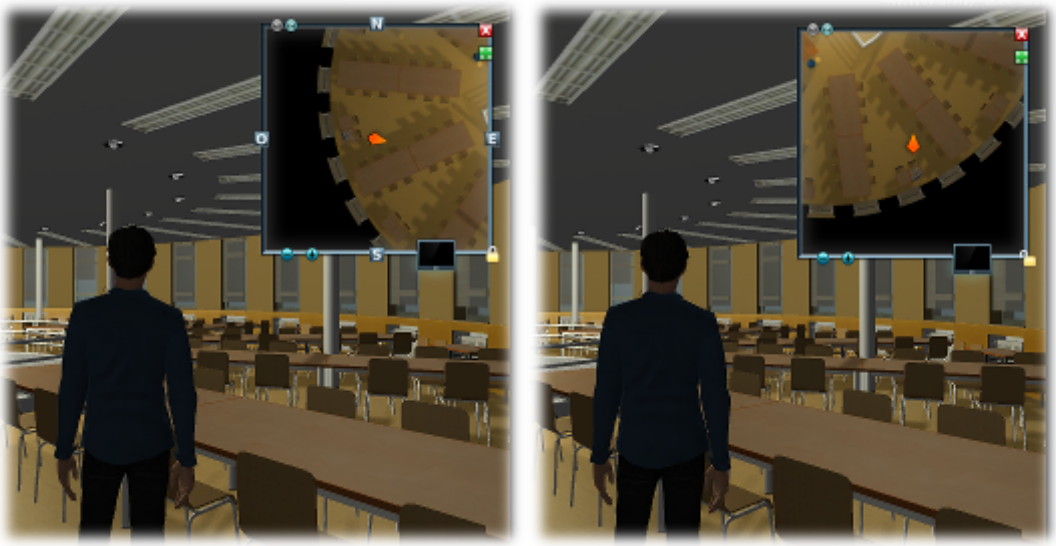


Figura 9.11: Orientación fijada al norte y orientación libre

9.4.2. Mapa a pantalla completa

En este modo el mapa ocupa la totalidad de la pantalla, la cantidad de terreno mostrado aumenta considerablemente y la sensación de inmersión es completa. El usuario puede desplazarse ahora por su entorno a vista de pájaro, tal y como puede apreciarse en la Figura 9.12.

Todas las consideraciones aplicables al mapa normal siguen siendo válidas en el mapa a pantalla completa, modos de orientación, suavizado de movimientos, etcétera. Se comporta como el primero con la salvedad de que toda la atención se centra ahora en el mapa y la posición del avatar, tan sólo un número voluntariamente reducido de botones puede distraer al usuario de lo realmente importante en este modo: su posición y su entrono.

9.4.3. Mapa cerrado o minimizado

Por último, el usuario puede, naturalmente, optar por cerrar el mapa para centrarse enteramente en el entorno tridimensional que lo envuelve. Este modo puede incluso configurarse para que sea el predeterminado al ejecutar la aplicación final, de forma que el mapa aparezca cerrado inicialmente hasta que el usuario decida abrirlo (Figura 9.13).

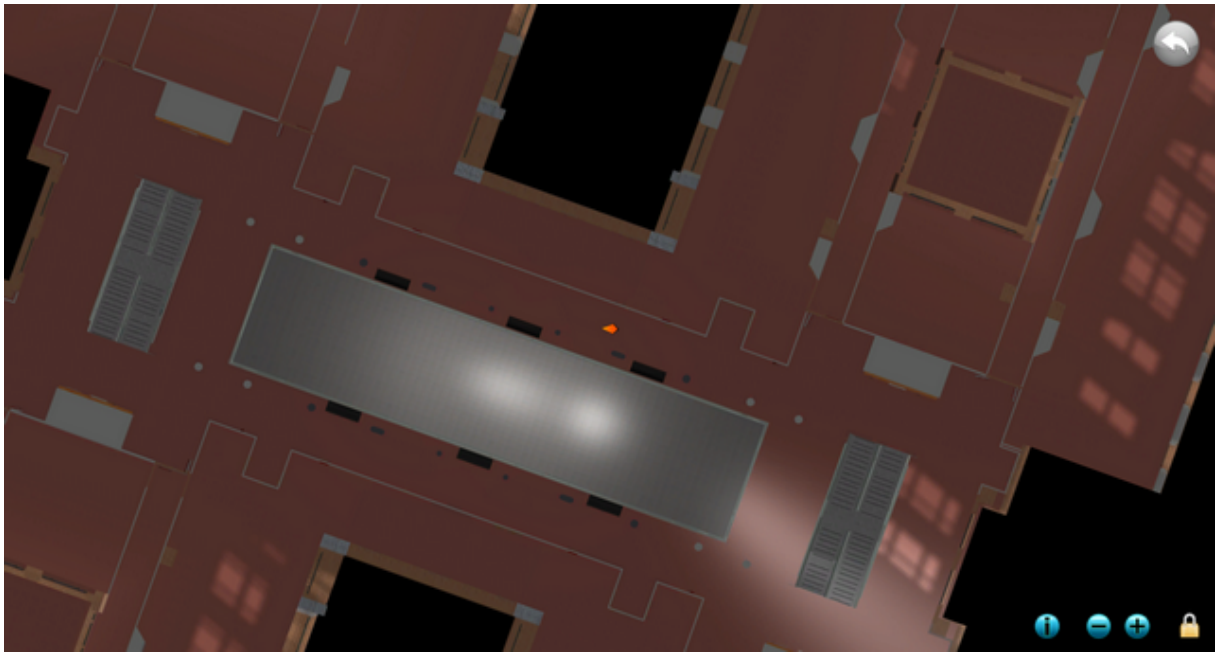


Figura 9.12: Mapa a pantalla completa

Figura 9.13: Mapa minimizado

Figura 9.14: Edición minuciosa de un elemento de la interfaz del mapa

9.5. Interfaz

La interfaz, como elemento de interacción entre el usuario y el Sistema de Posicionamiento se ha cuidado y estudiado minuciosamente.

Cada elemento gráfico se ha editado cuidadosamente (Figura 9.14) para garantizar que su aspecto sea óptimo, no sólo visual, también pragmáticamente.

El usuario puede interactuar cómodamente con el sistema utilizado los botones que para ello se facilitan: activar el modo a pantalla completa, cerrar el mapa, aumentar la cantidad de terreno mostrado, reducirla, cambiar la orientación, cambiar la posición del mapa, todos tan sólo a un golpe de clic de ratón.

Adicionalmente, los botones cambian de aspecto para aumentar la calidad de la interacción y para aportar una sensación de respuesta (Figuras 9.15 y 9.16). Así, en el momento que el usuario pulsa sobre alguno de ellos, su apariencia cambia para reflejar que ha sido pulsado, también cambia cuando el usuario pasa el ratón por encima indicándole precisamente que puede pulsar sobre el botón.

No sólo puede el usuario interactuar con el sistema mediante el uso del ratón,

Figura 9.15: Botón normal, activo y pulsado

Figura 9.16: Botón normal, activo, pulsado y elegido

también se le proporciona la posibilidad, siempre más rápida y para algunos más cómoda, de utilizar el teclado. Las funciones de cada botón y de cada tecla se describen a continuación.

9.5.1. Botones

Se describen aquí sólo las funciones de aquellos botones relacionados con el Sistema de Posicionamiento, el resto de botones de la interfaz se describirán en las secciones correspondientes a sus sistemas.

 Activar el modo de mapa a pantalla completa.

 En modo de mapa a pantalla completa, regresar al modo normal.

 Cerrar o minimizar el mapa.

 Abrir el mapa tras haberlo cerrado.

 Cambiar al modo de orientación fijada al norte (sólo presente durante el modo de orientación libre).

 Cambiar al modo de orientación determinada por la del avatar (sólo presente durante el modo de orientación fijada al norte)

 Aumentar la cantidad de terreno mostrado en el mapa.

 Reducir la cantidad de terreno mostrado en el mapa.

 Cambiar la posición del mapa a la esquina superior izquierda.

 Cambiar la posición del mapa a la esquina superior derecha.

 Presentes sólo en el modo de orientación fijada al norte, indican la posición de los puntos cardinales (son sólo informativos).

9.5.2. Teclas

Se describen a continuación las teclas asignadas con funciones relaciones con el Sistema de Posicionamiento, válidas tanto en el modo de mapa normal como en el modo a pantalla completa. Cada una de estas teclas es configurable, el valor que se indica es el valor predeterminado.

Aumentar la cantidad de terreno mostrado en el mapa Tecla predeterminada: **1**.

Reducir la cantidad de terreno mostrado en el mapa Tecla predeterminada: **3**.

Restablecer la cantidad de terreno mostrado en el mapa Tecla predeterminada: **2**.

Conmutar entre el modo a pantalla completa y cualquier otro modo Tecla predeterminada: **P**.

Conmutar entre el modo minimizado y el modo actual Tecla predeterminada: **M**.

9.6. Configuración y personalización

Es Sistema de Posicionamiento ofrece un alto grado de personalización. Puede modificarse desde el aspecto inicial del mapa hasta el tamaño de cada uno de los elementos de su interfaz, pasando por la modificación de las teclas predeterminadas para su manejo.

Se ha centralizado el lugar donde realizar todas estas modificaciones para comodidad del desarrollador. Así, aquellas relacionadas con el comportamiento del sistema

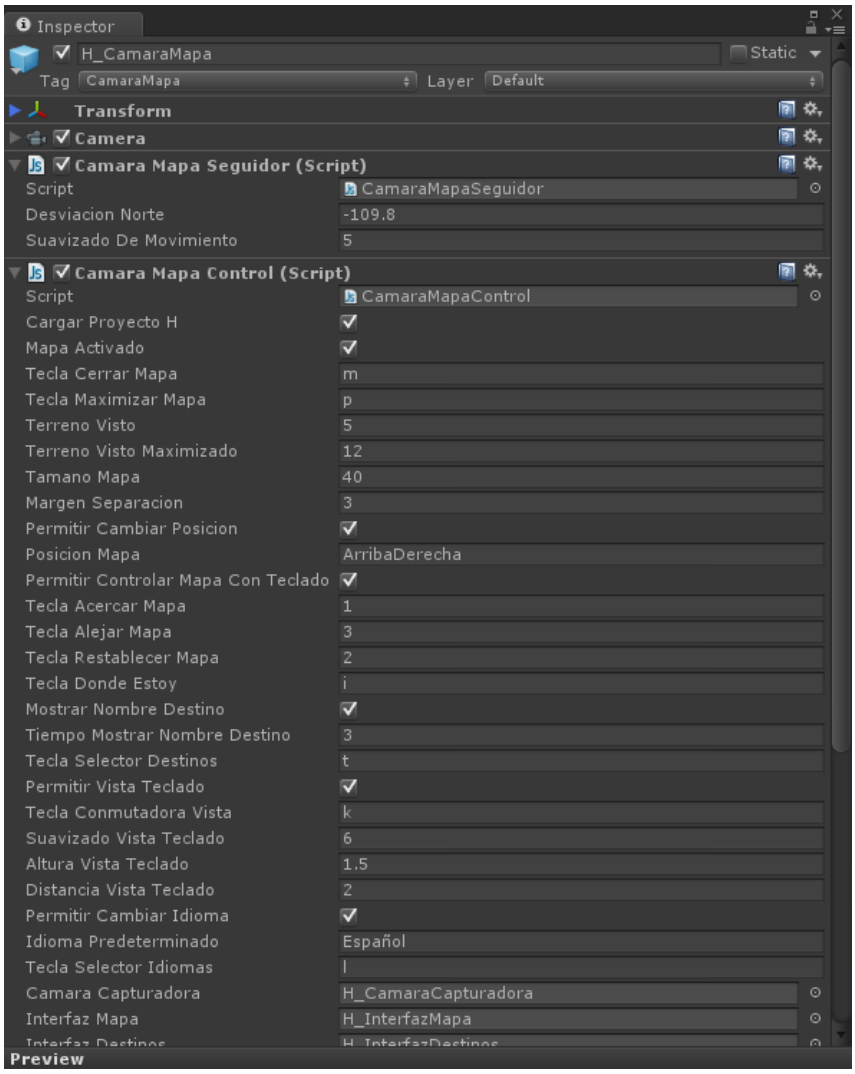


Figura 9.17: Personalización del Sistema de Posicionamiento (Inspector del prefab H_CamaraMapa)

se realizarán en el Inspector del prefab H_CamaraMapa (Figura 9.17), y todas aquellas relacionas con la modificación de su aspecto se realizarán en el Inspector del prefab H_InterfazMapa (Figuras 9.18 y 9.19).

Los valores configurables relacionados con el Sistema de Posicionamiento son los siguientes (el resto de valores no mencionados aquí se tratan en los capítulos de los sistemas a los que pertenecen).

9.6.1. Configuración del comportamiento, Inspector de «H_CamaraMapa»

Nótese que las tildes en los nombres de cada campo se omiten voluntariamente por consistencia en el código de los guiones.

9.6.1.1. Componente Camara Mapa Seguidor

Desviacion Norte Permite redefinir la desviación de la escena de referencia «01 Exterior» con respecto al norte geográfico.

Suavizado De Movimiento Campo donde indicar la cantidad de amortiguamiento de los movimientos del mapa en modo de orientación libre.

9.6.1.2. Componente Camara Mapa Control

Mapa Activado Determina si el mapa aparecerá inicialmente abierto o cerrado.

Tecla Cerrar Mapa Define la tecla asignada a la función de cerrar el mapa.

Tecla Maximizar Mapa Define la tecla asignada a la función de activar el modo de mapa a pantalla completa.

Terreno Visto Cantidad predeterminada de terreno visto inicialmente en el modo de mapa normal.

Terreno Visto Maximizado Cantidad predeterminada de terreno visto inicialmente en el modo de mapa maximizado.

Tamano Mapa Determina el espacio que ocupará el minimapa en pantalla.

Margen Separacion Indica el espacio de separación entre el borde de la pantalla y el borde del minimapa.

Permitir Cambiar Posicion Activado permite que el usuario pueda cambiar la posición del mapa entre las dos esquinas superiores de la pantalla (los botones para hacerlo desaparecen automáticamente de la interfaz si esta casilla se desactiva).

Posicion Mapa Posición inicial del mapa, «ArribaDerecha» o «ArribaIzquierda».

Permitir Controlar Mapa Con Teclado Activado permite que el usuario pueda cambiar la cantidad de terreno mostrado en el mapa normal y el maximizado utilizando el teclado.

Tecla Acercar Mapa Define la tecla asignada a la función de reducir la cantidad de terreno visto en el mapa.

Tecla Alejar Mapa Define la tecla asignada a la función de aumentar la cantidad de terreno visto en el mapa.

Tecla Restablecer Mapa Define la tecla asignada a la función de restablecer la cantidad de terreno visto en el mapa a su valor predeterminado (funciona independientemente para el mapa normal y para el mapa maximizado, utilizando el valor «Terreno Visto» para restablecer el primero, y «Terreno Visto Maximizado» para el segundo).

Interfaz Mapa Permite indicar el prefab encargado de gestionar la interfaz del Sistema de Posicionamiento, H_InterfazMapa.

9.6.2. Configuración del aspecto, Inspector de «H_InterfazMapa»

9.6.2.1. Aspecto del minimapa, componente «Interfaz Mapa Control»

Norte, Este, Sur, Oeste Estos cuatro campos definen los elementos gráficos estáticos que el sistema utilizará para representar los puntos cardinales.

Borde Mapa Define el elemento gráfico que servirá de marco del minimapa.

Porcentaje Puntos Cardinales Valor porcentual maestro que determina el tamaño de los puntos cardinales respecto del lado del minimapa. Este valor servirá de referencia para cada uno de los factores siguientes.

Factor Mas Menos Factor que, a partir del porcentaje de los puntos cardinales, indica el tamaño de los botones para aumentar y reducir la cantidad de terreno mostrado en el mapa.

Factor Candado Factor que, a partir del porcentaje de los puntos cardinales, indica el tamaño de los botones que permiten alternar entre el modo de orientación fijada al norte y el modo libre.

Factor Cerrar Mapa Factor que, a partir del porcentaje de los puntos cardinales, indica el tamaño del botón para minimizar el mapa.

Porcentaje Boton Abrir Mapa Valor porcentual respecto de la altura de la pantalla que determina el tamaño del botón para abrir el mapa cuando este está cerrado.

Porcentaje Margen Boton Abrir Mapa Valor porcentual respecto de la altura de la pan-

Figura 9.18: Configuración del aspecto del minimapa

talla que determina la distancia de separación entre el botón de abrir mapa y el borde de la pantalla.

Estilos Cada uno de ellos permite modificar los elementos gráficos de los botones a los que hace referencia su nombre. Los diferentes estados de los botones, normal, activo, pulsado o elegido, se modifican aquí.

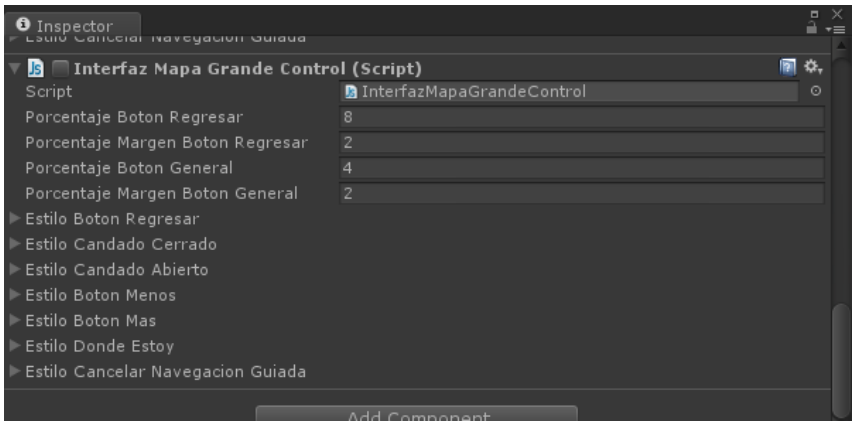


Figura 9.19: Configuración del aspecto del mapa maximizado

9.6.2.2. Aspecto del mapa maximizado, componente «Interfaz Mapa Grande Control»

Porcentaje Boton Regresar Valor porcentual respecto de la altura de la pantalla que determina el tamaño del botón para regresar desde el modo maximizado al modo normal o minimapa.

Porcentaje Margen Boton Regresar Valor porcentual respecto de la altura de la pantalla que determina la distancia de separación entre el citado botón y el borde de la pantalla.

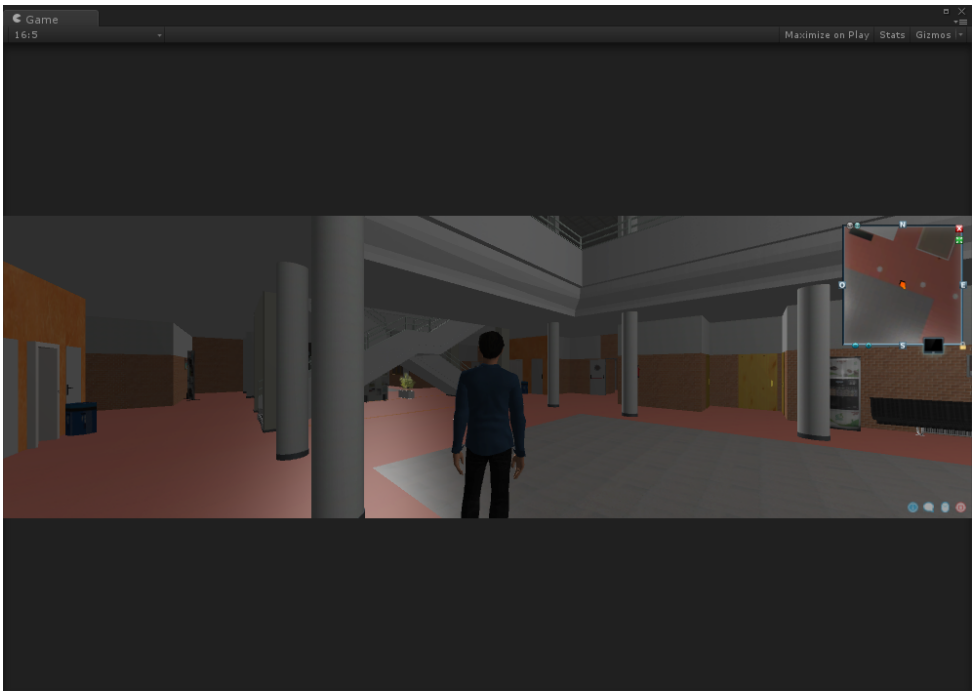
Estilos Permiten modificar los elementos gráficos de los botones a los que hace referencia su nombre. Los diferentes estados de los botones, normal, activo, pulsado o elegido, se modifican aquí.

9.7. Autoajustable a los cambios de resolución

En un mundo donde la única constante es el cambio, saber adaptarse a estos es una virtud.

En la búsqueda de este virtuosismo todos los elementos del Sistema de Posicionamiento se ajustan automáticamente a los cambios de resolución de pantalla. Esta cualidad permite ejecutar la aplicación final en infinitud de dispositivos, con cualquier tamaño de pantalla, manteniendo siempre las mismas condiciones óptimas visuales con las que se planteó inicialmente su diseño (Figura 9.20).

(a) Resolución 1:1



(b) Resolución 16:5

Figura 9.20: Adaptación automática de los cambios de resolución

```
1  function Update ()
2  {
3    if (Screen.width != anchuraPantalla || Screen.height != alturaPantalla)
4    {
5      alturaPantalla = Screen.height;
6      anchuraPantalla = Screen.width;
7
8      tamanoBotonRegresar = porcentajeBotonRegresar / 100 * alturaPantalla;
9      margenBotonRegresar = porcentajeMargenBotonRegresar / 100 * alturaPantalla;
10   }
11 }
```

Cuadro 9.3: Sencillo ejemplo de código para la detección y ajuste a los cambios de resolución

9.7.1. Adaptación en tiempo real

El tamaño del mapa, cada uno de los botones, toda la interfaz que integra este sistema modifican su tamaño inteligentemente para ajustarse a una nueva resolución, y no sólo en una primera instancia, lo hace en tiempo real, de tal forma que si el usuario o el sistema huésped que esté ejecutando la aplicación modifican, por ejemplo, la ventana en la que esta esté funcionando, la interfaz se adaptará al cambio inmediatamente.

9.7.2. Técnica

Para lograr esta adaptación automática a los cambios de resolución, el sistema analiza continuamente las dimensiones de la pantalla, tomando como referencia el lado menor, para realizar todos los cálculos que conducen al reajuste de tamaño de los elementos, sea cual sea el modo en el que esté funcionando el Sistema de Navegación, en cuanto detecta que se ha producido un cambio en la relación entre el ancho y el largo de la pantalla.

En el Cuadro 9.3 se recoge un sencillo código, a modo de ejemplo simplificado, para la detección y ajuste automático de un botón a los cambios de resolución.

9.8. Persistencia entre escenas

«Lo que funciona correctamente lo hace inadvertidamente».

Una característica importante que distingue al Sistema de Posicionamiento, presente en segundo plano, inadvertida, para garantizarle al usuario una experiencia de uso totalmente natural, es la persistencia entre escenas.

Esta cualidad distintiva logra que todo el estado actual del Sistema de Posicionamiento se replique de escena en escena, conservando cada aspecto que el usuario haya modificado, a pesar de que cuando una nueva escena se carga, todo lo anterior se destruye. De esta forma el usuario experimenta los cambios de escena sin saltos, los percibe como transiciones naturales de un lugar a otro y en las que nada ha cambiado, inconsciente de todo el entorno virtual en el que se encontraba anteriormente ya no existe.

9.8.1. Técnica

Para lograr la persistencia entre escenas, se recurre a la creación y minuciosa gestión de variables globales o estáticas[2]. La naturaleza de estas las hace permanentes desde que la aplicación se abre hasta que el usuario decide cerrarla.

La gestión de este tipo de variables es importante pues, convenientes en apariencia, sin una cuidada administración este método podría generar resultados inesperados cuando se utiliza imprudentemente. Se podría imaginar un escenario en el que una variable estática tramitara los puntos de vida de hipotéticos adversarios a los que el avatar debe enfrentarse escena tras escena. Cuando los puntos de vida de un adversario llegan a cero este ha sido derrotado y desaparece, sin embargo, por descuidar las particularidades de este tipo especial de variables, desaparece, no uno, sino todos los adversarios, hayan sido derrotados o no, de todas las escenas. Este ejemplo muestra por qué debe cuidarse minuciosamente la gestión de las variables globales, en él se ha cometido el fallo de asignar por comodidad una de estas variables a una condición común compartida por todos los oponentes, el concepto de puntos de vida, y cuando la variable global que los gestiona detecta que su valor ha llegado a cero se lo transmite a cada adversario pues, por su naturaleza, funciona globalmente.

El Sistema de Posicionamiento pone un especial cuidado en este sentido, garantizando un funcionamiento probadamente sólido.

Se recurre a la creación de una nueva clase estática, denominada SDN a partir de la contracción de las palabras «sistema de navegación», para albergar las variables y funciones globales de las que el Sistema de Posicionamiento, y en general, todos los sistemas diseñados, hará uso para posibilitar la persistencia de su estado entre escenas.

Una clase es, en esencia, una forma de organizar funciones y variables (Cuadro 9.4). Metafóricamente, si las variables fueran cajas contenedoras, y las funciones la maquinaria que manufactura sus contenidos, una clase sería la fábrica en la que estas cajas y maquinarias están dispuestas. Este es precisamente el propósito del guión SDN.js (intro-

```
1  static class SDN extends MonoBehaviour
2  {
3    // ======================
4    // VARIABLES PERSISTENTES
5    // ======================
6
7    // Elementos generales.
8    // --------------------
9
10   var camaraPrincipal : GameObject;
11   var avatar : GameObject;
12   var indicadorAvatar : GameObject;
```

Cuadro 9.4: Comienzo de la clase SDN, alberga variables y funciones estáticas

ducido en la Sección 6.2.3), el de servir de fábrica estructuradora y albergar las variables y las funciones estáticas de todo el conjunto de sistemas.

9.8.2. Puesta en práctica

El resultado de la aplicación de esta técnica permite:

- Conservar el modo actual del mapa, ya sea normal, maximizado o cerrado, pudiendo cambiar de escena en cualquier modo y que este perdure en la nueva, sin saltos y perfectamente configurado para la escena actual.
- Conservar el modo actual de orientación, fijada al norte o libre, conservando en todo momento la relación coherente entre la ubicación de todas las áreas.
- Mantener la cantidad de terreno visto de forma independiente para el mapa normal y el mapa maximizado. Si el usuario ha modificado la cantidad de terreno que muestra el mapa, esta personalización se conservará entre escenas, tanto para el minimapa como para el mapa a pantalla completa.
- Adaptar la interfaz automáticamente al estado del sistema entre escenas. De esta forma, los iconos y botones de la interfaz reflejan en todo momento correctamente las decisiones que el usuario haya tomado, desde el momento en que una escena se carga hasta el momento en que se destruye.

Capítulo 10

Subsistema de Conmutación de Mapas (mapas de detalle)

Este proyecto, buscando siempre ir un paso más allá, no se limita a presentar solamente un mapa para cada área, expande esta funcionalidad integrando la infraestructura necesaria para ofrecer varios niveles de mapas por terreno.

Esto abre las puertas a efectos tan vistosos como tejados que se ocultan automáticamente cuando el avatar pasa por debajo de ellos revelándole lo que antes permanecía oculto, o incluso a escaleras que desaparecen cuando el avatar atraviesa el hueco bajo estas, mostrando mobiliario u objetos que de otra forma pasarían desapercibidos.

10.1. Componentes

Guiones

- CamaraMapaControl.js
- ConmutadorMapaAutoactivador.js
- ConmutadorMapaEntradaSalida.js
- ConmutadorMapaEstadoAvatar.js
- ConmutadorMapaFamilias.js

- ConmutadorMapaFaseExterna.js
- ConmutadorMapaFaseInterna.js
- ConmutadorMapaSeguroExterno.js
- ConmutadorMapaSeguroInterno.js
- PuntoAvatarSeguidor.js

Prefabs

- H_PuntoAvatar

Objetos

- Elementos accionadores de tipo *Is Trigger* agrupados en las escenas bajo el prefijo común «H_AccMapa_».
- Planos de alto rendimiento, contenedores de las texturas de los mapas de detalle, agrupados en las escenas bajo el prefijo común «H_Mapa_» y pertenecientes a la capa o *Layer* «MapaInterior».

10.2. Técnica

La base de la conmutación de mapas son las funciones de tipo `OnTrigger`, que permiten activar bloques determinados de código cuando el avatar entra, abandona o permanece en el área que controlan.

Sabiendo que una cámara puede, en Unity, configurarse para acotar los objetos que puede ver mediante la distribución de los mismos en capas determinadas, se puede entonces asignar una capa o *Layer* a los mapas generales de terreno, en este caso una con el nombre «Mapa», y otra capa específica a los mapas de detalle o de interiores, con el nombre «MapaInterior» (Figura 10.1), para, así, poder configurar la cámara que utiliza el Sistema de Posicionamiento e indicarle qué debe mostrarle al usuario y cuándo hacerlo.

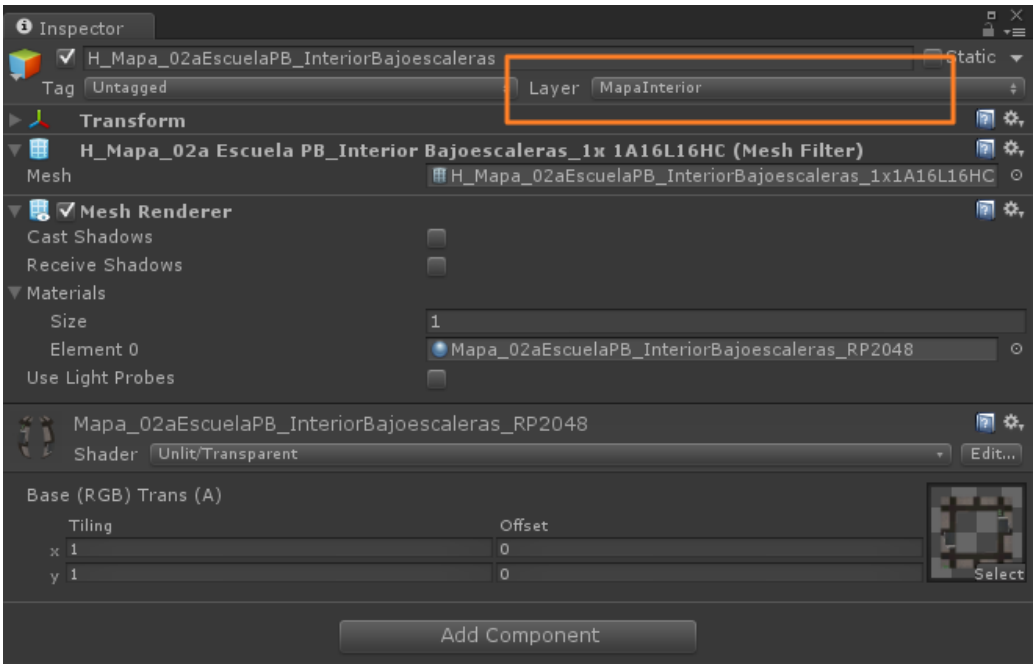


Figura 10.1: Capa «MapaInterior»

Combinando estos dos puntos, se puede colocar en la escena un elemento invisible que, asociado con un guión con las instrucciones apropiadas, configure la cámara del Sistema de Posicionamiento para mostrar un mapa de detalle cuando el avatar lo atraviese.

Para que el elemento invisible dispare las funciones de tipo `OnTrigger` que contendrá el guión y, en consecuencia, configurar la cámara del mapa en respuesta a las acciones del avatar, tendrá que activarse la casilla *Is Trigger* en su Inspector, bajo el componente *Box Collider* (Figura 10.2).

Finalmente, será el álgebra de Boole el que gestione, mediante las instrucciones del guión configurador, las capas que la cámara del mapa observará. Puede observarse, en el Cuadro 10.1, el ejemplo más sencillo de una función `OnTrigger` junto con una aplicación documentada del álgebra de Boole (fragmento extraído directamente del guión ConmutadorMapaEntradaSalida.js).

Este tipo de mapas de detalle se gestionarán de la misma forma que los mapas generales. Todo lo válido para los segundos, su forma de obtención mediante el Sistema de Captura o su integración final en la escena correspondiente (Anexos A y B), es aplicable también para los mapas de detalle. Sólo dos aspectos difieren entre ellos, la capa a la que pertenecen y la coordenada Y en la que se coloca el plano contenedor, siendo esta -100 para los planos generales y -99 para los planos de detalle.

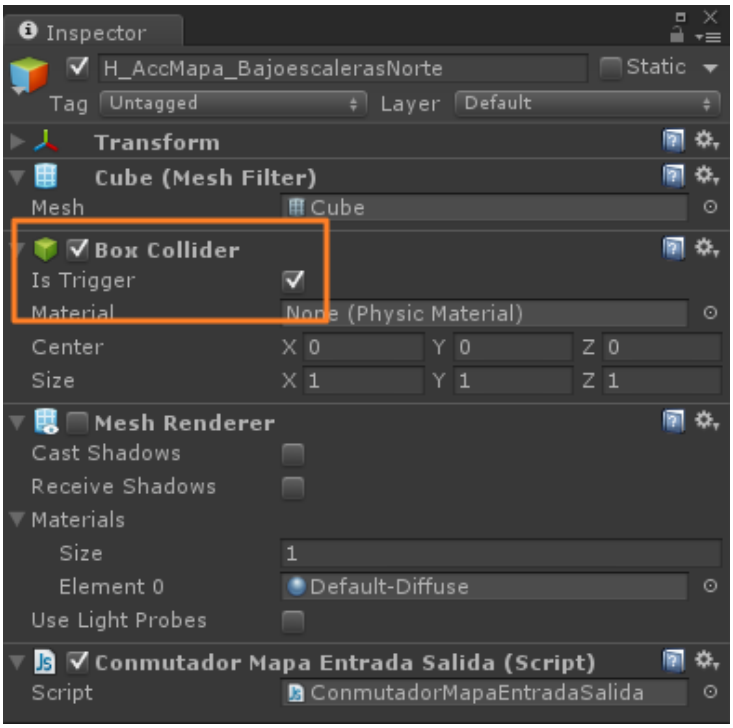


Figura 10.2: Opción *Is Trigger* activada

```
1  function OnTriggerEnter (colisionador : Collider)
2  {
3    if (colisionador.tag == "PlayerPunto")
4    {
5      // Colocamos un uno en los bits de las capas "Mapa" y "MapaInterno"
6      // Mapa (p. e. capa 8): 0000 1000 0000
7      // MapaInterior (p. e. capa 9): 0001 0000 0000
8
9      // Nota: usamos el operador OR (|) para sumar las dos máscaras de bits:
10     // 0000 1000 0000   OR   0001 0000 0000   =   0001 1000 0000
11
12     // Nota: la función LayerMask.NameToLayer("...") permite utilizar los nombres de las
           capas en lugar de sus números, de esta forma se consigue que, en caso de cambiar
           de posición las capas, el código siga siendo eficaz.
13
14     camaraMapa.camera.cullingMask = 1 << LayerMask.NameToLayer("Mapa") | 1 << LayerMask.
           NameToLayer("MapaInterior");
15   }
16 }
```

Cuadro 10.1: Función de tipo `OnTrigger` y álgebra de Boole

```
1  function Start ()
2  {
3    objetivo = SDN.avatar.transform;
4
5    transform.position = objetivo.position;
6  }
7
8
9  function LateUpdate ()
10 {
11   // Vinculamos la posición con la del avatar.
12
13   transform.position = objetivo.position;
14
15   // El centro de coordenadas del avatar se encuentra en sus pies, por lo que elevamos la
         posición del concentrador de masa (este) hasta una altura nominal por encima del
        suelo.
16
17   transform.position.y += 0.5;
18 }
```

Cuadro 10.2: Fragmento del código que vincula el prefab H_PuntoAvatar con el avatar

10.3. Prefab «H_PuntoAvatar»

Para poder calcular con total precisión las colisiones, o instantes en el que el cuerpo del avatar incide con otro de su entorno, y detectar sin error el momento en el que el avatar atraviesa las superficies invisibles disparadoras de las funciones `OnTrigger` se ha creado el prefab H_PuntoAvatar.

Este prefab está compuesto por una esfera transparente de dimensiones conocidas, un guión controlador PuntoAvatarSeguidor.js y un componente *RigidBody* que permitirá gestionar los movimientos de la esfera mediante simulaciones físicas realistas a partir del motor físico de Unity, comportándose como lo haría cualquier objeto sujeto a las leyes físicas del mundo real.

En este caso concreto, la finalidad última el componente *RigidBody* es la de garantizar la exactitud en las colisiones del avatar con los elementos disparadores de su entorno, por lo que características como la respuesta a la gravedad o las fuerzas de arrastre, intrínsecas a todo cuerpo *RigidBody*, son innecesarias y se prescinde de ellas.

El sencillo código que contiene el guión controlador de la esfera la vincula, nada más cargar una escena, con el avatar, haciendo que se mueva según lo haga este (Cuadro 10.2).

A efectos prácticos, lo que este procedimiento consigue, es concentrar y localizar la masa del avatar en una figura geométrica elemental y totalmente definida: una esfera (Figura 10.3).

Finalmente, asignado una etiqueta única al prefab, «PlayerPunto», y configurando las funciones `OnTrigger` para que sólo respondan cuando un elemento con esta etiqueta las atraviesa, conseguimos que los disparadores se activen exclusivamente cuando la esfera entre en contacto con ellos. De esta forma, además de concentrar virtualmente la masa del avatar en una esfera de radio conocido, también se logra que su entorno responda ante ella como si esta fuera el propio avatar, permitiendo alcanzar así la garantía de precisión perseguida con esta técnica.

Figura 10.3: Esfera concentradora de masa

Cabe notar también, como efecto adicional muy destacable, que si bien se ha creado el prefab H_PuntoAvatar como herramienta aseguradora del rigor en las colisiones como parte del completo sistema de navegación que este proyecto diseña, su utilidad se extiende a cualquier proyecto anterior[1] y a todos los venideros. Bastará configurar los *Trigger* que se utilicen para que respondan sólo a la etiqueta «PlayerPunto», para poder disfrutar del mismo grado de precisión en la detección de colisiones del que disfruta este sistema de navegación.

10.4. Tipos de conmutadores

Estos elementos invisibles entorno a los que ha girado el desarrollo del apartado anterior, con los que el avatar puede entrar en contacto y activar así las instrucciones que estén programados para realizar ante tal circunstancia, son los elementos que estudiará con detalle el presente apartado: los conmutadores, desde el punto de vista de los mapas de detalle en particular, pero atendiendo a todo su potencial en general.

Se han diseñado tres tipos diferentes de conmutadores. Cada con con una solución optimizada a unas necesidades particulares de conmutación que, en conjunto, resuelven las exigencias de cualquier escenario, sea cual sea la naturaleza de estas.

[1] Con ánimo, no sólo de ampliar, sino también de mejorar y enriquecer el proyecto global se han modificado ya todos los guiones anteriores a este proyecto particular tal y como se recoge en la Sección 18.10.

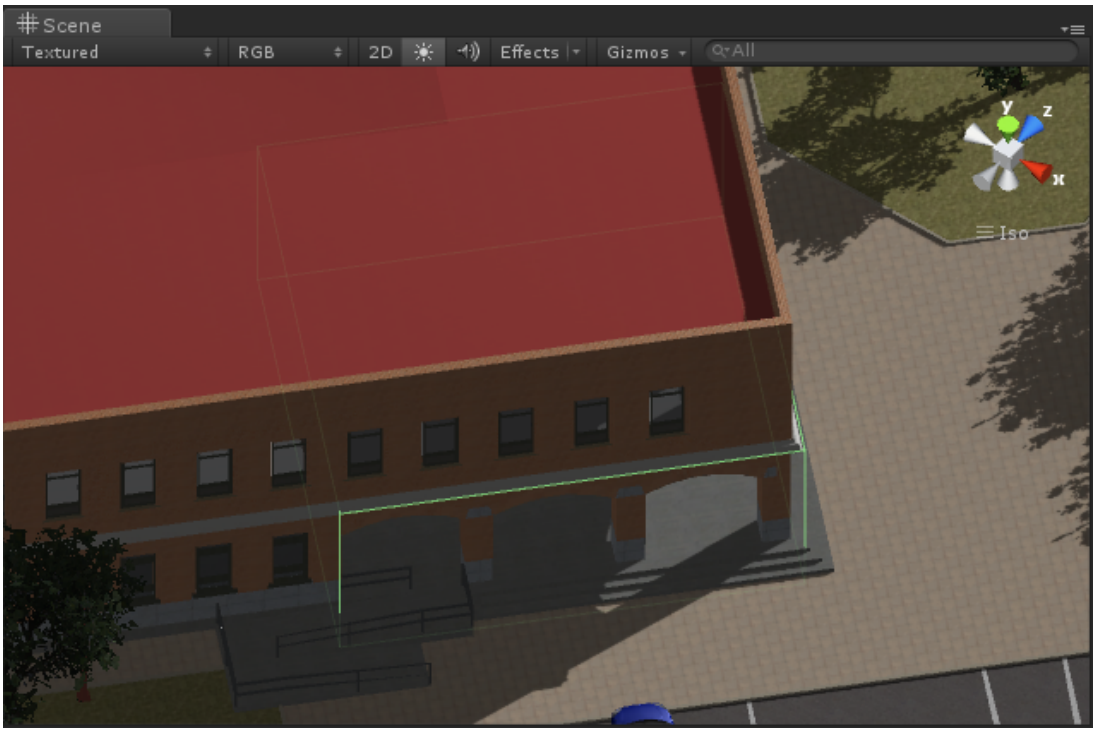


Figura 10.4: Conmutador de tipo entrada/salida

10.4.1. Conmutador de tipo entrada/salida

El conmutador más sencillo de los tres, el de tipo entrada/salida, permite activar los mapas internos cuando el avatar entra en un área concreta, y desactivarlos cuando este la abandona. Es útil para mostrar mapas de detalle como bajoescaleras o pórticos de entrada bajo tejado y, en general cualquier área con un perímetro básico.

Su puesta en marcha pasa por integrar en la escena un cubo invisible con la opción *Is Trigger* activada, que abarque el área a controlar (Figura 10.4), y añadirle el guión ConmutadorMapaEntradaSalida.js. Cuando el avatar entre en el cubo el guión se encargará de configurar automáticamente la cámara del mapa, haciéndole visible también, además del mapa general, el mapa asignado a la capa «MapaInterior» que se corresponderá con la vista de detalle de esta área controlada (Figura).

10.4.2. Conmutador de familias de mapas

Este tipo de conmutador abarca áreas extensas y permite activar y desactivar conjuntos completos de mapas en una misma escena. Es útil en aquellas que engloben más de un nivel de altura, como pueden ser los diferentes pisos de un edificio.

Para integrar este conmutador en una escena que albergue dos niveles diferentes

Figura 10.5: Conmutador de tipo entrada/salida en acción

de altura, una planta baja y un primer piso, por ejemplo, bastará con colocar un cubo invisible que comprenda la totalidad del área de una de las plantas, activar en su Inspector la opción *Is Trigger* y añadirle el guión ConmutadorMapaFamilias. Tras esto sólo resta asignarle a los campos personalizables del guión los mapas correspondientes a cada una de las plantas, con la posibilidad de asignar también los mapas correspondientes de detalle. Ahora, cuando el avatar suba o baje a uno de los dos pisos, los mapas cambiarán en tiempo real para ofrecerle al usuario los adecuados a la planta en la que se encuentra.

Los mapas a gestionar pueden agruparse en carpetas, lo que permite que, utilizando el nombre de la misma a la hora de configurar el guión, este active o desactive la totalidad de los mapas que ella incluya, algo especialmente útil para gestionar los mapas de detalle, pues si el mapa general será único, los de detalle pueden ser muchos más de uno.

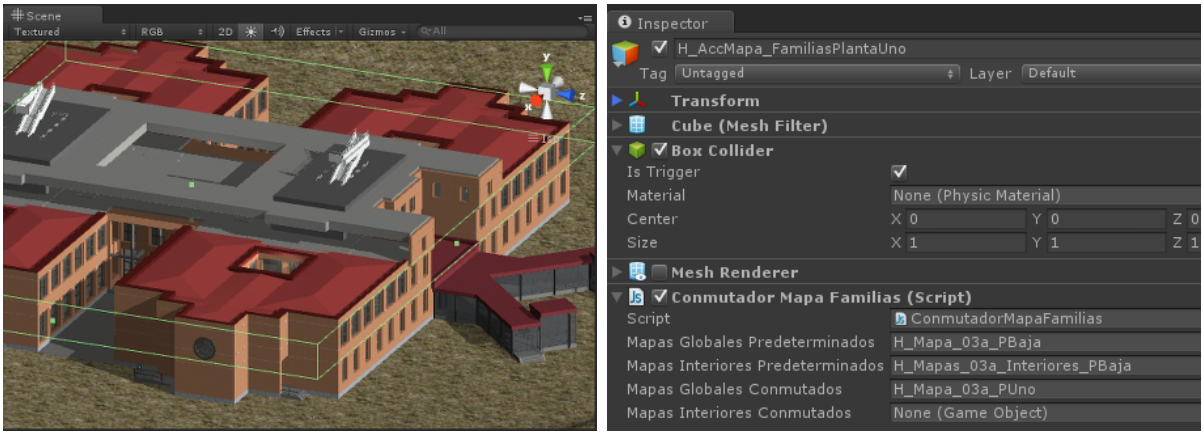
Como puede apreciarse en la Figura 10.6a, el cubo invisible abarca la totalidad de la primera planta del edificio. Según esta disposición, la asignación de los mapas en el guión, según se puede observar en la Figura 10.6b, es la siguiente:

Mapas Globales Predeterminados El mapa general correspondiente a la planta no controlada por el cubo.

Mapas Interiores Predeterminados Los mapas de detalle correspondientes a la planta no controlada por el cubo (si son más de uno puede asignarse una carpeta que los contenga, también puede dejarse en blanco).

Mapas Globales Conmutados El mapa general correspondiente a la planta en la que se encuentra el cubo.

Mapas Interiores Conmutados Los mapas de detalle correspondientes a la planta en la



(a) Colocación del conmutador (b) Configuración del conmutador

Figura 10.6: Conmutador de familias de mapas

que se encuentra el cubo (si son más de uno puede asignarse una carpeta que los contenga, también puede dejarse en blanco).

10.4.2.1. Sobrecarga del conmutador (control de más de dos niveles)

Si en una misma escena abarcara más de dos niveles, si, como ocurre en la escena «02b EdifTecnologicoP2» del proyecto legado, son tres los pisos a controlar, este tipo de conmutador puede sobrecargarse para realizar la conmutación entre cada una de las áreas.

Para ello se colocarán esta vez dos cubos, configurados de la forma normal, y abarcando cada uno los dos pisos superiores. En la Figura 10.7 se aprecian las plantas dos, tres y cuatro de un edificio a controlar.

La configuración del guión conmutador se hará de la siguiente forma, correspondiente a la Figura 10.8.

Cubo intermedio

Mapas Globales Predeterminados El mapa general correspondiente a la planta baja, no controlada por ningún cubo.

Mapas Interiores Predeterminados Los mapas de detalle correspondientes a la planta baja.

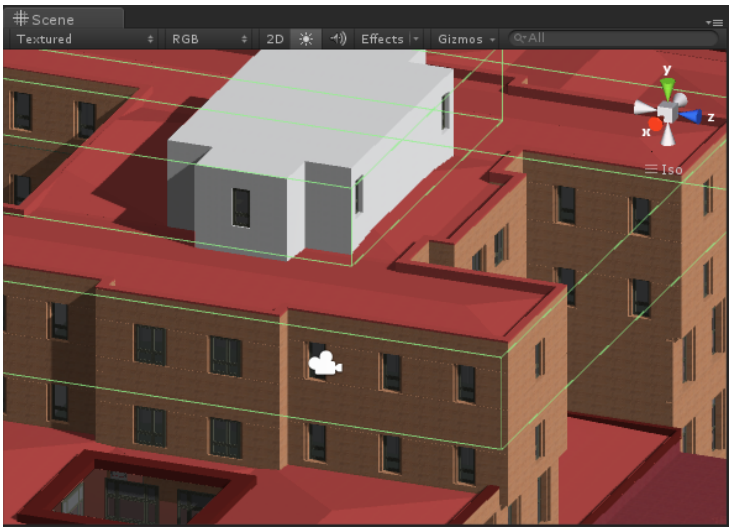


Figura 10.7: Conmutador de familias de mapas sobrecargado para controlar tres niveles

Mapas Globales Conmutados El mapa general correspondiente a la planta en la que se encuentra el cubo intermedio.

Mapas Interiores Conmutados Los mapas de detalle correspondientes a la planta en la que se encuentra el cubo intermedio.

Cubo superior

Mapas Globales Predeterminados El mapa general correspondiente a la planta baja, no controlada por ningún cubo.

Mapas Interiores Predeterminados Los mapas de detalle correspondientes a la planta baja.

Mapas Globales Conmutados El mapa general correspondiente a la planta en la que se encuentra el cubo superior.

Mapas Interiores Conmutados Los mapas de detalle correspondientes a la planta en la que se encuentra el cubo superior.

10.4.3. Conmutador bifase

Este tipo especial de conmutador permite activar un mapa concreto cuando el avatar atraviesa un portal en un sentido y desactivarlo cuando lo vuelve a atravesar en sentido contrario. Demuestra todo su potencial a la hora de controlar áreas cerradas

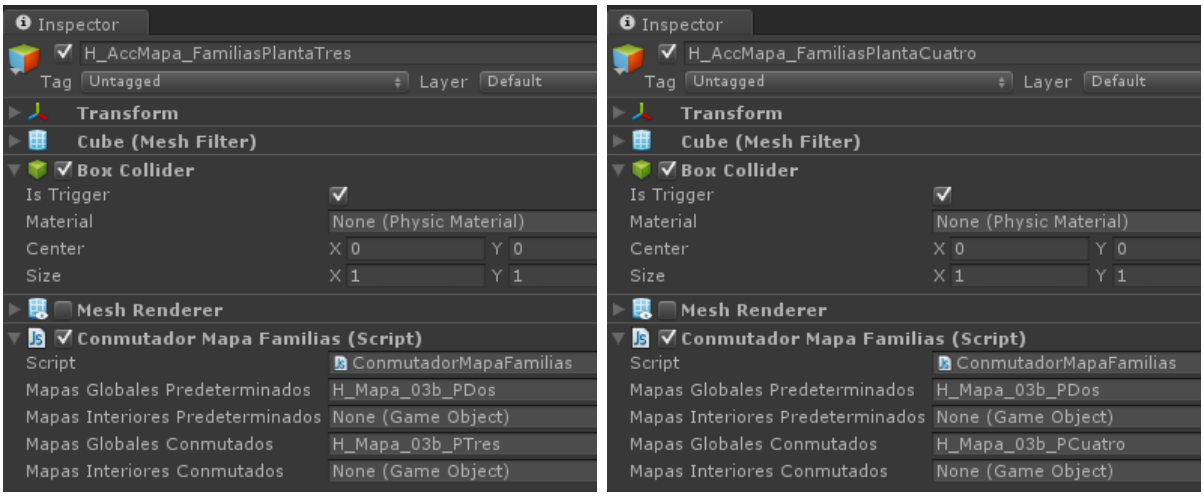


Figura 10.8: Configuración sobrecargada del conmutador de familias de mapas

complejas, siendo un ejemplo real de esta labor el pasaje entre la escuela y el edificio tecnológico. En especial este conmutador, el más avanzado de los tres, no sólo es útil en el contexto de la activación y desactivación de mapas, sirve como conmutador de precisión de ámbito general pues solventa automáticamente escenarios adversos posibles: un bajo FPS[2] de ejecución del programa, resolución desordenada de las instrucciones de control cuando se generan en un mismo *frame*, el avatar dándose media vuelta en el punto clave de conmutación, éste abandonando lateralmente la zona de conmutación, etcétera.

10.4.3.1. Componentes

El sistema de conmutación bifase está compuesto por varios elementos:

- Una fase interna.
- Una fase externa.
- Un módulo de registro.
- Módulos autoactivadores.
- Módulos de seguro.
- El elemento de precisión H_PuntoAvatar.

[2] *Frames* por segundo.

(a) Dimensiones de las fases (b) Distancia entre fases

Figura 10.9: Disposición de las fases interna y externa con el prefab H_PuntoAvatar como referencia

10.4.3.2. Disposición

Las fases interna y externa son los elementos centrales del conmutador. Están representadas por sendos planos verticales invisibles controlados por los guiones ConmutadorMapaFaseInterna.js y ConmutadorMapaFaseExterna.js respectivamente.

La ubicación en la escena de estas dos fases ha de realizarse con atención a algunos aspectos. La fase interna se colocará en el punto donde se desea realizar la conmutación, por ejemplo una hipotética línea de meta o, como algo más concreto y fácil de visualizar, el vano de una puerta; y sus dimensiones se ajustarán a las del área atravesable (en el caso del ejemplo de la puerta, abarcará desde el suelo hasta el dintel y, a lo ancho, las dimensiones entre los marcos). La fase externa se colocará paralela a la interna, separada una distancia menor que el diámetro del elemento de precisión H_PuntoAvatar (Sección 10.3). Sus dimensiones serán las mismas que la fase interna a lo alto, y, a lo ancho, serán como mínimo el ancho de la fase interna más dos veces el diámetro del elemento H_PuntoAvatar.

En la Figura 10.9a puede comprobarse cómo se han dispuesto las fases para controlar el acceso a un recinto cerrado de perímetro complejo, el pasaje de unión entre la Escuela de Ingenierías y el edificio tecnológico, y en la Figura 10.9b se puede observar la distancia entre las mismas para este mismo punto).

El módulo de registro sirve como base de datos con información sobre el estado del avatar, su posición en términos de dentro o fuera de la zona controlada por el conmutador bifase, estado compartido y actualizado por todos los elementos del sistema. Está consti-

tuido por un objeto vacío[3] controlado por el guión ConmutadorMapaEstadoAvatar.js.

Los módulos autoactivadores son cubos invisibles controlados por el guión ConmutadorMapaAutoactivador.js y constituyen por sí mismos un subtipo básico de conmutador. Si el área a controlar dispone de puntos de acceso a través de los que el avatar puede entrar en la misma sólo a través de otra escena, la colocación de uno de estos módulos en el lugar donde aparece el avatar en el área tras cargar la escena simplifica el control de la conmutación. Su función es la de activar, en este particular, los mapas de detalle correspondientes al interior del área controlada y, en general, activar cualquiera que sea la función a conmutar cuando el avatar entra en la citada área (siguiendo el ejemplo anterior de una línea de meta, marcar un tanto). Lo hacen en cuanto detectan la presencia en su entorno del avatar y, aparte de actualizar también el estado del módulo de registro, no realizan ningún otro control especial. Sin estos módulos autoactivadores habría que colocar pares de fases interna y externa en cada acceso al interior del área, sin distinguir si el acceso se produce desde la misma escena o desde una diferente, algo perfectamente posible pero innecesario gracias a este tipo de módulos, permitiendo reducir la necesidad global de cálculo.

En la Figura 10.10 se señalan tres módulos autoactivadores que gestionan el acceso al pasaje cuando este se hace desde las entradas junto a las que están colocados. Estos accesos particulares se realizan siempre desde una escena diferente en la que se encuentra el conmutador bifase, correspondientes en este caso a los dos edificios que aparecen en la figura, la Escuela de Ingenierias y el edificio tecnológico, por lo que el uso de este tipo de módulos en estos puntos simplifican el control de esa área cerrada que representa el pasaje de unión. El resto de elementos no señalados se corresponden con la fase interna y externa del conmutador, ubicadas en la entrada principal al recinto, y dos elementos correspondientes a un nuevo tipo de módulo que se describirá en el párrafo que sigue.

Finalmente, los módulos de seguro son dos por cada par de fases interna y externa. Constituidos por sendos cubos controlados por los guiones ConmutadorMapaSeguroInterno.js y ConmutadorMapaSeguroExterno.js respectivamente, se colocarán en el entorno inmediato de la fase que les dé nombre. Así, el seguro interior se colocará junto a la fase interior, dentro del área controlada, a una distancia algo mayor que el diámetro del elemento H_PuntoAvatar desde la posición de la fase externa; y el seguro externo a una algo mayor que la longitud del diámetro de dicho elemento desde la fase interna. La anchura mínima de los seguros será la de la fase correspondiente más dos veces el diámetro de

[3]Un objeto vacío es aquel cuya única propiedad es su posición. Se usan en Unity como objetos esqueleto, como punto de partida o lienzo en blanco para numerosas aplicaciones, pudiendo personalizarlos el desarrollador con aquellos componentes que necesite (en este caso el citado guión), y prescindiendo de todos los demás.

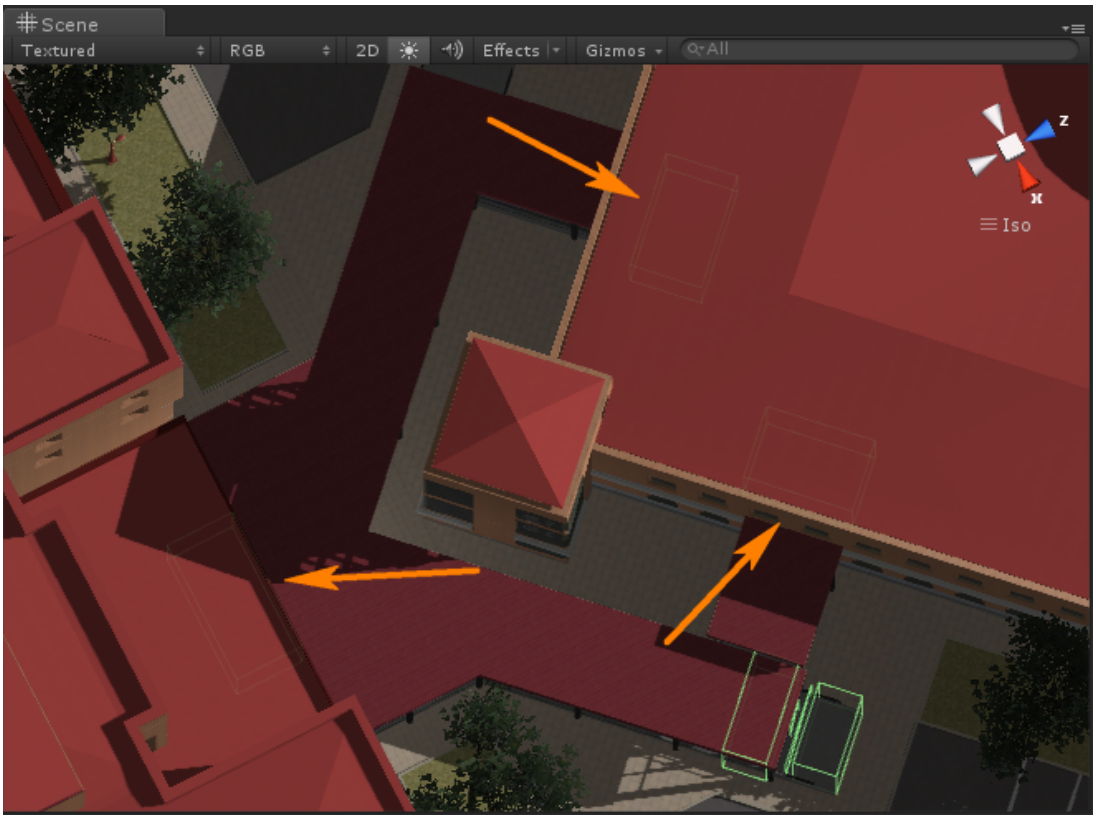


Figura 10.10: Módulos autoactivadores

H_PuntoAvatar (en la Figura 10.11 pueden verse este tipo de módulos señalados, junto a cada una de las fases del conmutador).

Los módulos de seguro garantizan la efectividad del conmutador bifase al cien por cien, resolviendo los escenarios, remotos pero posibles, en los que las funciones `OnTrigger`, núcleo de cualquier sistema conmutador, no detecten el paso del avatar, o que dichas funciones `OnTrigger` se activen en un mismo *frame* y se resuelvan en un orden aleatorio en el siguiente.

El primero de los dos escenarios planteados en el párrafo anterior, relacionado con objetos que se mueven muy rápidamente, puede solventarse activando la opción «*Collision Detection/Continuous*» en el componente *RigidBody* del elemento de precisión H_PuntoAvatar o, alternativamente, aumentando los *frames* por segundo de ejecución de la aplicación mediante el «*Time Manager*»[4]. No obstante, el primer caso no es siempre eficaz y aumentar los *frames* por segundo aumenta el consumo de recursos. Por tanto, los módulos de seguro resuelven este escenario sin detrimento del rendimiento.

El segundo de los escenarios está relacionado con los ciclos de ejecución (Capítulo

[4]Herramienta interna de Unity que permite alterar las unidades básicas de tiempo del motor físico. Su comportamiento se detalla en http://docs.unity3d.com/Manual/class-TimeManager.html.

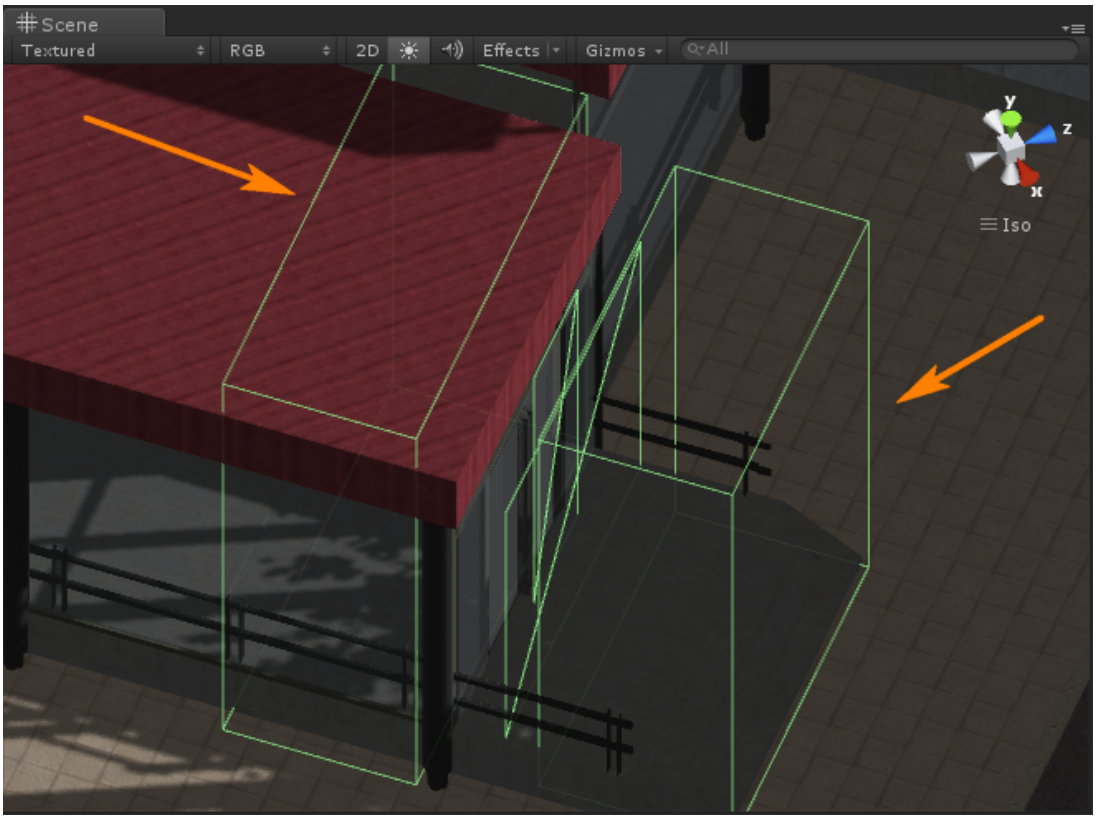


Figura 10.11: Módulos de seguro

5). Si han de resolverse más de dos funciones `OnTrigger` en un mismo *frame*, se sabe cuándo se resolverán, pero no en qué orden. Aunque reproducible en un porcentaje mínimo de los casos, de no considerar también este escenario, podría suceder que los mapas de detalle quedaran activados al salir del área controlada o viceversa. Los módulos de seguro, por tanto, garantizan la fiabilidad total del sistema.

10.4.3.3. Concepto base

Esta disposición particular de cada uno de los los elementos asegura que la conmutación se realizará únicamente cuando el avatar haya atravesado la fase interna en un sentido, o la fase externa en el otro, y resuelve íntegramente todos los problemas que cualquier otro sistema alternativo presentaría.

El ingenio del conmutador bifase se asienta en el siguiente concepto: sólo se activarán los mapas de detalle cuando la fase interna detecte que el avatar ya no está en contacto con ella, pero tampoco en contacto con la fase exterior. Si el avatar abandonara la fase interna pero aún siguiera en contacto con la fase externa, significaría entonces que avanza hacia exteriores, momento en el que la fase interna le daría el relevo del control a la externa, la que comenzaría un nuevo ciclo en el que se aplicaría este mismo razonamiento

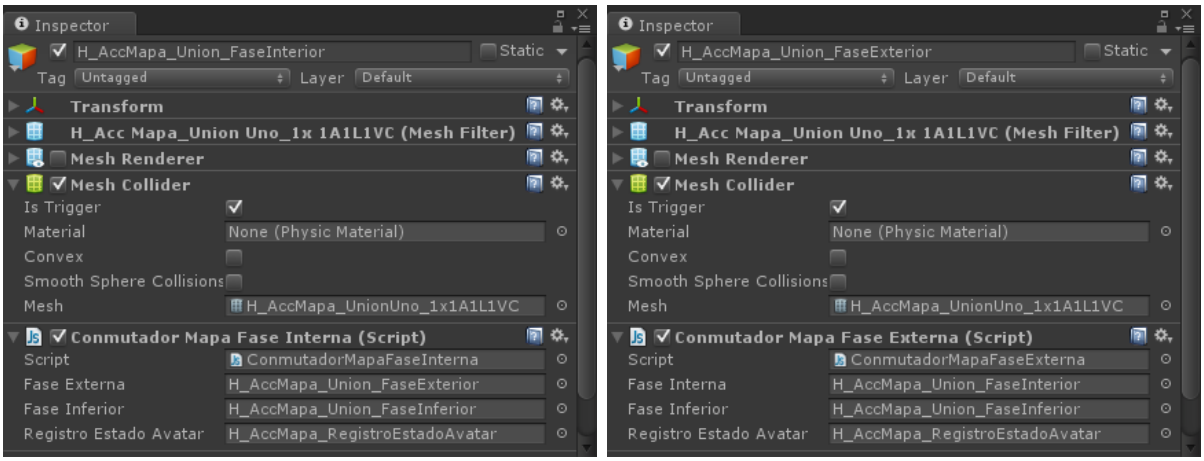


Figura 10.12: Inspector de las fases interna y externa

pero desde el punto de vista de exteriores.

Técnicamente las dos fases están vinculadas la una con la otra, de no estarlo no sería posible llevar a cabo el concepto planteado. Esto se consigue asignando la fase interna en el Inspector de la externa y la externa en el de la interna, tal y como puede apreciarse en la Figura 10.12.

La razón de que la distancia que separa ambas fases deba ser menor que el diámetro del elemento H_PuntoAvatar, la esfera concentradora y representativa del avatar, es precisamente para garantizar que esta siempre estará en contacto con una fase, sea interior, exterior o las dos, salvo que haya atravesado por completo el conmutador, en cuyo caso podrá accionarse el código de conmutación con seguridad plena de que el avatar estará dentro o fuera, sin limbos, sin estados no definidos, con precisión total.

En la disposición de elementos se ha mencionado que la anchura de la fase exterior debe ser mayor que la de la inferior. La razón de ser de esta particularidad es la de resolver aquel escenario en el que el avatar abandonara lateralmente y a la vez ambas fases. Sin un código respaldando al sistema de esta eventualidad este reaccionaría ambiguamente ante ella. Sin embargo el conmutador bifase está protegido: cuando el avatar abandone lateralmente las fases (algo sólo posible en el lado exterior por la particular disposición de los elementos), en el momento en el que deje de estar en contacto con la línea de conmutación, representada por la fase interna, seguirá estando en contacto con la fase externa, que activará entonces los controles correspondientes a exteriores anulando cualquier traza de ambigüedad.

10.4.3.4. Comparación con métodos alternativos

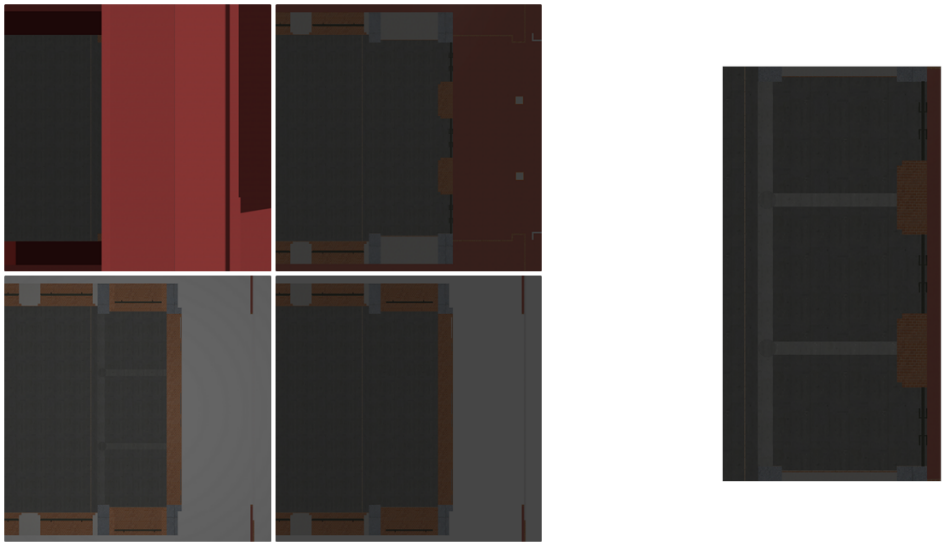
El primer sistema alternativo posible en que se puede pensar consiste en crear un simple cubo en la entrada del área a controlar, asociarle un guión que actué cuando el avatar entre y salga y que cuide de conocer la posición del avatar, sea interior o exterior. Esta sencilla idea tiene un problema no obstante, la interacción con el cubo se realizará con la cara horizontal de entrada y con la cara horizontal de salida, lo que obliga a que el avatar atraviese siempre completamente el cubo para que este sistema sea útil, pues si una vez dentro del mismo, el avatar se diera media vuelta y saliera de éste por la cara por la que a entrado en él, se produciría un limbo en el sistema que anularía completamente su fiabilidad. Adicionalmente, las caras laterales introducirían nuevos estados de incertidumbre que lo harían aún menos preciso.

Hacer el cubo tan plano como se quiera no resuelve este problema pues internamente sigue estando compuesto por varias caras y el resultado será de nuevo errático y no fiable en el cien por cien de los casos.

Sustituir el cubo por un único plano debería resolver el problema idealmente pero, al contrario de lo por lógica cabría esperar, en la práctica no ocurre así. Este sistema no dispone de ninguno de los elementos de control de los que sí dispone el conmutador bifase, y los múltiples escenarios en los que el avatar no se limita a atravesar el plano sin más, sino que se da media vuelta, avanza lateralmente, lo cruza más o menos rápido o se interpone un bajo FPS de ejecución del programa, se resuelven sin garantía alguna de acierto.

Un tercer método alternativo consistiría en replicar todo el perímetro interior del área a controlar mediante un programa de modelado tridimensional, conservando su forma y dimensión, colocarlo posteriormente en Unity dentro de esta área y configurar este nuevo espacio como si de un conmutador de tipo entrada/salida se tratara. Es un método en principio válido, sin embargo, para poder ponerlo en práctica deben conocerse con exactitud cada una de las dimensiones que determinan el área a controlar y, la forma de esta, en cuanto se aleja de una figura geométrica sencilla, puede complicarlo sobremanera haciéndolo del todo inviable. Este último método, por tanto, sería viable si en el momento en el que se modeló el área se tomó también la precaución de crear un modelo cerrado de su interior, en cualquier otro caso, el enorme esfuerzo que implicaría ponerlo en práctica lo descarta como posibilidad.

En resumen, donde ningún otro método puede garantizar su fiabilidad, el conmutador bifase sí lo hace, y donde otros métodos que sí lo harían requieren de esfuerzos impensables, el conmutador bifase se despliega sin apenas esfuerzo ofreciendo además una



(a) Capturas fuente (b) Composición resultante

Figura 10.13: Composición de un mapa de detalle

flexibilidad inaccesible a los demás.

10.5. Mapas de detalle mejorados con programas de edición fotográfica

Los mapas de detalle comparten la misma esencia que los mapas generales, los crea el mismo Sistema de Captura y el proceso de integración es idéntico salvando unas pocas diferencias (Anexo C.1), por esta razón, si los mapas generales que se incluyen en este proyecto han sido editados meticulosamente para lograr el mejor de los resultados finales, también lo han sido los mapas de detalle, con más cuidado aun si cabe por tratarse de mapas de dimensiones contenidas y por tener que fusionarse estos sin fisura alguna con los mapas generales que los engloban, creado una transición visual de unos a otros impecable.

Además de la técnica de composición de capas ya tratada en la edición de los mapas generales (Sección 7.7.1), merecen en este caso especial mención otras dos técnicas especialmente útiles en este tipo particular de mapas: la gestión del canal *alpha* y la corrección de iluminación.

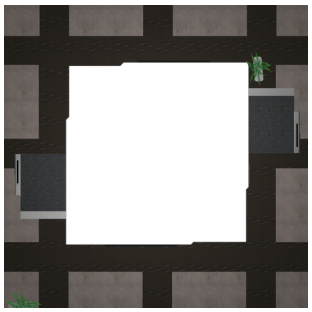


Figura 10.14: Mapa de detalle con una transparencia interior

10.5.1. Gestión del canal *alpha*

Tras esta denominación rimbombante se esconden las transparencias (Figura 10.14).

Trabajando en formato PNG[5], un formato de compresión gráfica basado en un algoritmo sin pérdidas de calidad que añade un canal *alpha* a las imágenes, se pueden introducir áreas transparentes en los mapas de detalle (Figura 10.15) que, una vez superpuestos sobre los mapas generales tras el proceso de integración, producen la sensación visual de que sólo ciertos elementos concretos han cambiado. Se logra «vestir» a los mapas generales, como si se les añadieran o quitaran prendas, produciendo un efecto visual absolutamente natural.

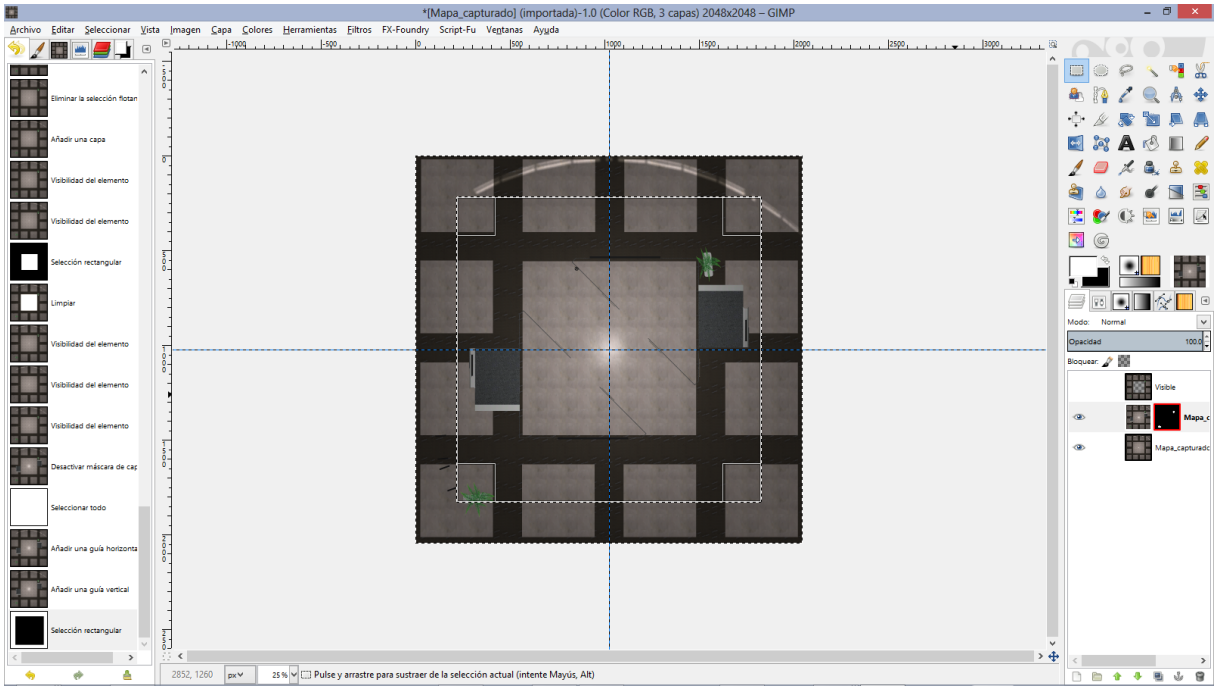
10.5.2. Corrección de iluminación

Cuando se captura un mapa de detalle, posicionando por lo general la cámara cartografiadora a una altura que permita salvar un tejado o cualquier tipo de cubierta que oculte lo que hay debajo, se produce un cambio de iluminación que, si lo que se persigue es el resultado óptimo, hay que corregir manualmente con un programa de edición fotográfica.

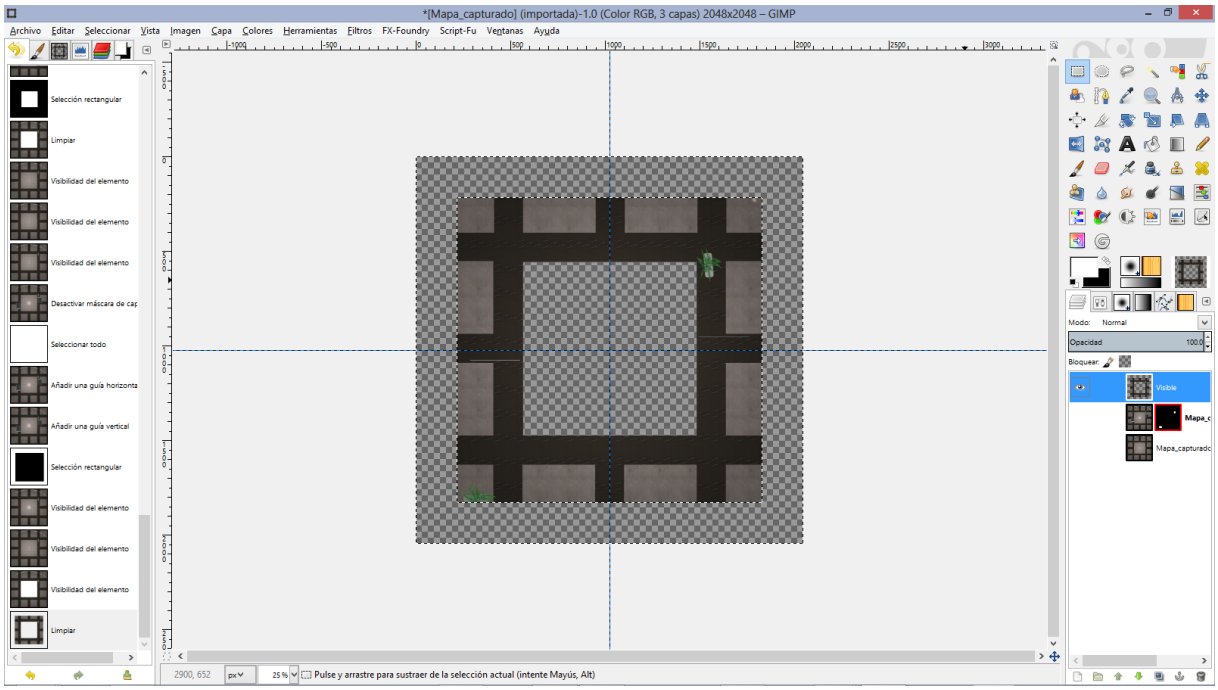
El proceso requiere de un alto grado de meticulosidad y perfeccionismo pues exige encontrar el tono concreto de color, brillo y contraste para cada zona de la imagen a partir de una de referencia determinada por la captura general.

En la Figura 10.16a se puede observar cómo la imagen de referencia de la izquierda tiene una iluminación diferente a la del mapa de detalle aún sin editar de la derecha que, en la Figura 10.16b ya ha sido corregida para mimetizarse totalmente con la de referencia.

[5] *Portable Network Graphics*

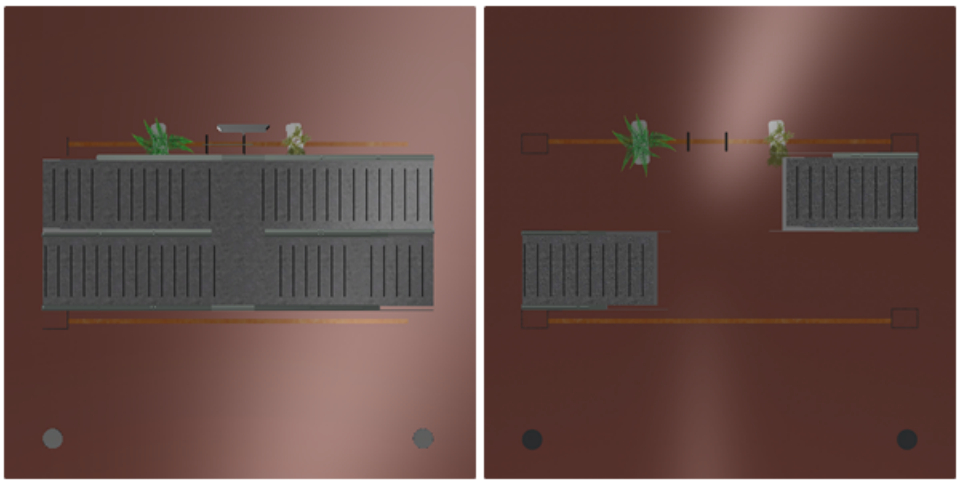


(a) Antes



(b) Después

Figura 10.15: Proceso de creación de un área transparente



(a) Iluminaciones diferentes

(b) Iluminación corregida

Figura 10.16: Proceso de corrección de iluminación de un mapa de detalle

10.6. Lugares de aplicación

La conmutación de mapas y los mapas de detalle se han utilizado en las siguientes ubicaciones organizadas por escenas.

Escena «01 Exterior»

- En las ocho puertas de acceso secundarias al edificio tecnológico (conmutadores de tipo entrada/salida).
- En las dos portonas de acceso norte y sur al edificio tecnológico (conmutadores de tipo entrada/salida).
- En el pasaje de unión entre la Escuela de Ingenierías y el edificio tecnológico (conmutador bifase).
- En la entrada principal a la Escuela de Ingenierías (conmutador de tipo entrada/salida).

Escena «02a Escuela PB»

- En los huecos bajo las dos escaleras de acceso a la planta primera (conmutadores de tipo entrada/salida).

Escena «03a EdifTecnologicoP1»

- En los huecos bajo los dos bloques principales de escaleras de acceso a la planta primera (conmutadores de tipo entrada/salida).
- En los huecos bajo las cuatro escaleras secundarias de acceso a la planta primera (conmutadores de tipo entrada/salida).
- En la planta primera, para controlar esta y la planta baja (conmutador de familias de mapas).

Escena «03b EdifTecnologicoP2»

- En las plantas tres y cuatro, para controlar estas y la planta dos (conmutador de familias de mapas sobrecargado).

Escena «05a BibliotecaArriba»

- En el hueco bajo la escalera de acceso al piso superior (conmutador de tipo entrada/salida).

Capítulo 11

Sistema de Localización

Este sistema se encarga de informar al usuario cuando este lo solicita, con un mensaje por pantalla temporizado y en el idioma actual, del lugar en el que se encuentra, tal y como puede observarse en la Figura 11.1. Es, asimismo, el encargado de la primordial función de determinar y proporcionar internamente la ubicación del avatar al resto de sistemas cuyo funcionamiento depende de esta información, como es el caso del Sistema de Navegación Guiada (Capítulo 14).

11.1. Componentes

Guiones

- CamaraMapaControl.js
- DondeEstoyControl.js
- PersistenteControl.js
- InterfazMapaControl.js
- InterfazMapaGrandeControl.js
- Traductor.js
- PresentadorNombreDestinoControl.js

Figura 11.1: ¿Dónde estoy?

- SDN.js
- Guiones agrupados bajo el prefijo común «PuntoDeControl» controladores de los puntos de control.

Prefabs

- H_CamaraMapa
- H_DondeEstoy
- H_IntefazMapa
- H_Persistente
- H_PuntoAvatar
- H_PresentadorNombreDestino
- H_AccMapa_RegistroEstadoAvatar[1]

[1]Módulo de registro del conmutador bifase del pasaje de unión entre la Escuela de Ingenierías y el edificio tecnológico.

Objetos

- Puntos de control agrupados en las escenas bajo el prefijo común H_PuntoDeControl.

Recursos

- Texturas constitutivas de la interfaz, organizadas en la ventana de proyecto dentro de las carpetas «Proyecto H/Interfaz/InterfazDondeEstoy» y «Proyecto H/Interfaz/General»

11.2. Funcionamiento

Tras los preparativos iniciales llevados a cabo por el guión CamaraMapaControl.js y el prefab H_CamaraMapa, aparece en la escena el prefab nuclear de este sistema, H_DondeEstoy. Este, bajo la gestión del guión DondeEstoyControl.js, es el encargado de determinar la ubicación del avatar y transformarla en términos comprensibles como «Secretaría», «Planta dos de la biblioteca» o «Ala este de la planta baja del edificio tecnológico».

Una vez el sistema ha precisado el área en el que se encuentra el avatar[2], la hace accesible al resto asignando esa información a la variable estática o global `SDN.dondeEstoy`, cuyo contenido se encarga de actualizar transparentemente cuando el avatar activa un punto de control o cambia de escena, marcando así la transición a una nueva área.

La determinación del área en la que el avatar se halla se realiza de forma optimizada, minimizando la utilización de recursos de la aplicación final. Esto se consigue precisamente actualizando la ubicación del avatar sólo cuando este dispara un punto de control o cambia de escena. La microgestión de estos dos activadores, y en particular de los puntos de control, diferencia a este sistema de cualquier otro alternativo que, prescindiendo de este tipo microgestión, interpretara la ubicación del avatar de forma continuada, ganando en comodidad lo que pierde en optimización.

La asignación del valor de la variable `SDN.dondeEstoy` o, lo que es lo mismo, la traducción interna de la posición del avatar en términos matemáticos a términos cotidianos y comprensibles, se realiza, ya sea directamente cuando un punto de control detecta el paso del avatar o mediante una base de datos recogida en el guión DondeEstoyControl.js

[2]El Anexo D recoge un listado de las áreas gestionadas hasta el momento por el Sistema de Localización.

```
1 case "Nombre interno de la nueva escena":
2   SDN.dondeEstoy = "Nombre comprensible de la nueva escena";
3   break;
```

Figura 11.2: Ampliación de la base de datos de escenas del Sistema de Localización

que, cuando una nueva escena se carga, traduce el nombre máquina de la misma a uno inteligible por el usuario.

Esta base de datos encargada de interpretar los nombres internos de las escenas, puede ampliarse fácilmente cuando una nueva escena se modele y se añada al proyecto global. Para ello bastará con añadir un nuevo y sencillo bloque de código de tipo `case` en el guión citado a continuación de los ya recogidos, tal y como se ejemplifica en el Cuadro 11.2.

11.3. Puntos de control

Esta sencillez de ampliación del Sistema de Localización sigue siéndolo cuando se trata de los puntos de control.

Un punto de control no es más que un plano o un cubo activador que, gestionado mediante un guión con el código adecuado, actualiza la variable global albergadora de la ubicación del avatar cuando este entra o sale de él.

11.3.1. Tipos de puntos de control

11.3.1.1. De entrada

El más básico de ellos se activa y actualiza la variable de ubicación sólo cuando detecta que el avatar ha entrado él.

Es útil cuando se desee dividir completamente una escena en diferentes subáreas de tal forma que, cubriendo en conjunto la totalidad del terreno se pueden definir nombres particulares a divisiones concretas del área general. Se menciona este uso como principal, pero su utilización es generalista, otra función apropiada es también la de gestionar zonas de paso intermedias y, por ejemplo, aportar un grado adicional de control entre dos plantas de un edificio.

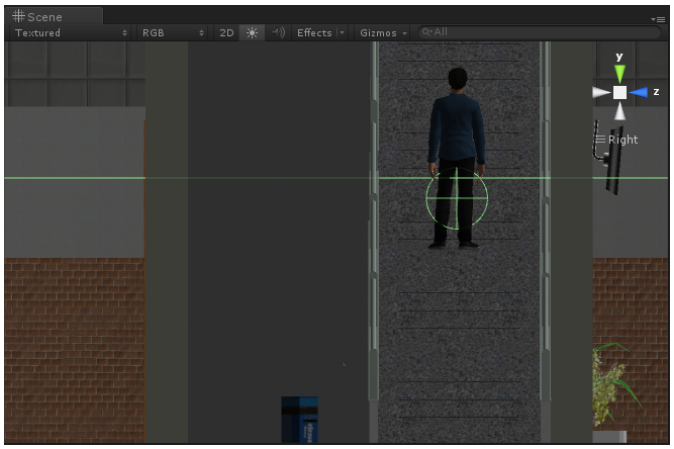


Figura 11.3: Un plano horizontal actúa como punto de control en el punto medio de unas escaleras

Este tipo de puntos de control se ha utilizado en este proyecto para la división y gestión de todas las plantas del edificio tecnológico en alas este y oeste. También se ha utilizado ingeniosamente para el control de la compleja red de escaleras del mismo edifico, posibilitando que el Sistema de Navegación Guiada, como se verá más adelante, conduzca exitosamente al usuario a través de esta intrincada maraña de accesos entre plantas.

En la Figura 11.3 se puede observar cómo un plano horizontal, colocado en la altura media de unas escaleras, sirve de punto de control para detectar cuándo el avatar ha alcanzado ese lugar. En el momento en el que la esfera entre en contacto con el plano, este indicará al Sistema de Localización que la ubicación del avatar es la que este punto de control gestiona.

11.3.1.2. De entrada y salida

Se activan cuando el avatar entra en ellos y también cuando sale.

Extremademente útiles cuando el área que se desea controlar está contenida en otra mayor. A diferencia del tipo anterior, que cubren la totalidad de un terreno, los puntos de control de entrada y salida se colocan dentro un terreno para controlar un área limitada en él, como puede ser una habitación o un despacho en una planta de un edificio. Al entrar el avatar en esta área el punto de control asigna a la variable de ubicación el valor correspondiente al interior de la misma, y cuando el avatar abandona el recinto controlado, reasigna el valor correspondiente al área general exterior.

Este tipo de puntos de control reflejan con sencillez la idea de ejecutar código sólo cuando sea necesario que se anticipó en el Capítulo 4, «Estilo de código», y de la que se ha

hablado aquí al mencionar la microgesión que los puntos de control permitían. Sin estos y la infraestructura que los rodea para gestionar los eventos que producen, la ubicación del avatar debería calcularse de forma continuada, desperdiciando recursos inútilmente.

11.3.1.3. Intersticial

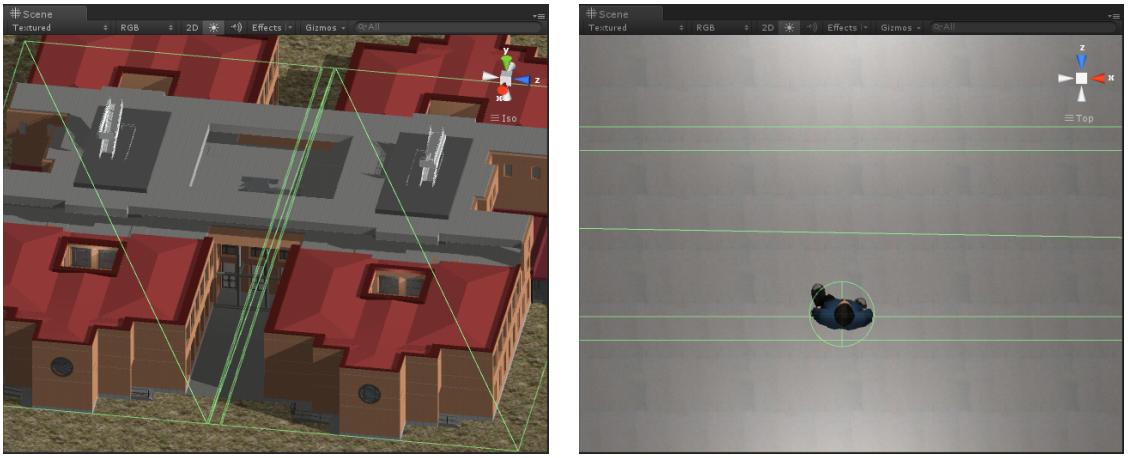
Existen para resolver una limitación en el despliegue de los puntos de control de entrada: las líneas de unión entre ellos.

Al describir el primer tipo de puntos de control se mencionó, en el primer ejemplo de utilización, que debían abarcar la totalidad del terreno cubriendo todas las subáreas de división. Esto es así para asegurar que, cuando el avatar abandone una subárea, el punto de control de la aledaña se active asignando inmediatamente a la variable de ubicación el nombre de la subárea a la que se accede. En este planteamiento hay, sin embargo, un problema: si el avatar, justo en la línea de unión de ambas áreas, se da media vuelta, el punto de control del área que ahora deja atrás se habría activado ya al detectar su fugaz entrada, pero el del área que nunca ha llegado a abandonar, al no haber detectado una nueva entrada por no haber llegado el avatar a salir del todo, no reasigna el nombre del área en el que éste se encuentra en realidad.

Se podría pensar que, separando sencillamente los puntos de control, sean cubos o planos, una distancia mayor al diámetro del elemento H_PuntoAvatar que representa a la avatar, se resolvería este rebuscado obstáculo, pero sólo en parte. El escenario aún más remoto en el que el avatar se transportara por algún medio directamente a esta zona de separación, lo dejaría, a efectos del sistema, en tierra de nadie, y su ubicación sólo dejaría de ser ambigua cuando el avatar se moviera dentro de alguno de los puntos de control adyacentes.

En un sistema que se enorgullece de su precisión esto no puede permitirse, aun si se trata de una circunstancia totalmente remota. Los puntos de control intersticiales se presentan, por tanto, para garantizar esa precisión total.

Desplegados en esa tierra de nadie virtual (Figura 11.4), su función es la de asignar a la variable global de ubicación el nombre de una de las dos áreas adyacentes, comportándose a efectos prácticos como una extensión de esta. Sólo si la variable global no ha sido asignada antes, lo que se traduce en que el avatar se ha transportado a la escena directamente dentro de esta zona particular, se activará este tipo de puntos de control realizando ellos la asignación. Finalmente la precisión total se garantiza con el seguro adicional que incluyen en su código, por el que se resuelve el hipotético caso en



(a) Punto de control intersticial entre dos de tipo entrada (b) Líneas de unión entre tres puntos de control

Figura 11.4: Planta de un edificio gestionada por puntos de control

el que el avatar se transportara, no dentro del punto de control intersticial, sino sobre la línea de intersección de este con el adyacente de tipo entrada, en cuyo caso retardaría automáticamente las instrucciones de su propio código, otorgando preferencia al otro punto de control y permitiendo que se adelante en la asignación.

En resumen, la utilización conjunta de los puntos de control de entrada y los intersticiales, cuando el despliegue particular de los primeros así lo requiera, es garantía de precisión.

11.3.1.4. Singulares

Un último tipo de puntos de control engloba a aquellos encargados de resolver escenarios muy específicos.

Un ejemplo de este tipo es el controlado por el guión PuntoDeControlTecnologicoPlantaBajaOeste.js, encargado de gestionar el ala oeste de la planta baja del edificio tecnológico, y que incluye un bloque de código con la función de simular que el ala oeste de esta planta es una extensión del ala este cuando el Sistema de Navegación Guiada está activado. De no incluir este código, cuando el destino de navegación de este último sistema fuera precisamente esta planta baja, podrían producirse ambigüedades, pues la división en alas este y oeste es tan sólo virtual.

Por tanto estos puntos de control singulares pueden considerarse como una herramienta personalizada al servicio de la fiabilidad total del conjunto.

11.4. Interfaz

El usuario puede solicitarle al sistema información sobre su localización de dos maneras, pulsando con el ratón sobre el botón de información siempre visible, o utilizando el teclado pulsando la tecla configurable asignada a esta función.

Los guiones InterfazMapaControl.js e InterfazMapaGrandeControl.js controladores del prefab H_IntefazMapa, y el guión PersistenteControl.js controlador del prefab H_Persistente, son los encargados de gestionar estas solicitudes; las generadas por el ratón los primeros, y las generadas por el teclado el último.

A continuación se agrupan esquemáticamente estas dos formas de interacción.

11.4.1. Botones

 Solicitar ubicación en modo de mapa normal.

 Solicitar ubicación en modo de mapa a pantalla completa.

11.4.2. Teclado

Solicitar ubicación en cualquier modo de mapa Tecla predeterminada: **I**.

11.5. Configuración y personalización

La tecla con la que el usuario puede llamar al sistema es configurable y puede personalizarse a través del Inspector del prefab centralizador H_CamaraMapa tal y como se puede observar en la Figura 11.5a. El tiempo en segundos durante el que se muestra el mensaje en pantalla también lo es y puede configurarse en este mismo prefab (Figura 11.5b).

(a) Personalización de la tecla asignada (b) Personalización del tiempo

Figura 11.5: Personalización del Sistema de Localización

11.6. Integración con el Sistema de Traducción

Este Sistema de Localización se integra con el de traducción (Capítulo 16) permitiendo que, cuando el usuario solicita información sobre su ubicación, reciba ésta en el idioma que en ese momento esté activo.

La integración se realiza a partir del valor de la variable de ubicación, el Sistema de Traducción la recoge y la compara con la información de su propia base de datos. Si encuentra una coincidencia devuelve la traducción apropiada.

Puesto que este Proyecto se ha planteado de base como herramienta multilingüe se ha previsto de antemano esta integración y, para simplificar la notación, a la hora de asignar el valor de la variable `SDN.dondeEstoy` mediante los puntos de control o las denominaciones internas de las escenas, en lugar de utilizar nombres ya definitivos para ser mostrados directamente al usuario se han utilizado nombres intermedios, más fáciles de manejar, que serán los que cotejará el Sistema de Traducción. No obstante, esto es sólo un aspecto de notación y no impide de ninguna manera utilizar nombres finales, o cadenas de texto de cualquier tipo, para asignar el valor de la variable global de ubicación. Si el Sistema de Traducción no encuentra una traducción apropiada en su base de datos para el valor asignado a `SDN.dondeEstoy`, devolverá sencillamente el valor original. Así, en caso de que se añadan nuevas áreas en el futuro al proyecto global y no se actualice de manera acorde la base de datos de traducciones, la funcionalidad de todo el sistema nunca se verá mermada y seguirá siendo óptima.

11.7. Integración con el Subsistema de Información por Pantalla

La infraestructura mediante la que el usuario recibe el mensaje con la información sobre su ubicación actual la proporciona el Subsistema de Información por Pantalla. Éste

recoge el nombre de la ubicación ya interpretado por el Sistema de Traducción y lo muestra en pantalla cuando el usuario solicite su localización. Es este sistema también el encargado de temporizar el mensaje y hacerlo desaparecer automáticamente al cabo del tiempo configurado.

Este subsistema se describirá en el Capítulo siguiente.

Capítulo 12

Subsistema de Información por Pantalla (presentadores de texto)

El encargado de gestionar los mensajes con los que el sistema informa al usuario de lo que sucede en el mundo virtual es el Subsistema de Información por Pantalla, y lo hace cuidando al detalle no sólo los aspectos más puramente técnicos del proceso, sino también la faceta visual, consciente de que si a través de él va a comunicarse el sistema con el usuario, esta comunicación ha de resultar, ante todo, atractiva.

12.1. Componentes

Guiones

- CaparaMapaControl.js
- PersistenteControl.js
- PresentadorNombreDestinoControl.js
- PresentadorCargandoDestinoControl.js
- SDN.js

Prefabs

- H_CamaraMapa
- H_PresentadorCargandoDestino
- H_PresentadorNombreDestino
- H_Persistente

Recursos

- Texturas de fondo organizadas en la ventana de proyecto dentro de la carpeta «Proyecto H/Interfaz/General/Fondos».
- Tipografías organizadas en la ventana de proyecto dentro de la carpeta «Proyecto H/Tipografias».

12.2. Funcionamiento

Una vez el prefab H_CamaraMapa ha cargado en la escena todos los elementos constituyentes del sistema, este estará listo para dar respuesta a cualquier componente que requiera de una comunicación con el usuario y, en particular, para servir de voz a los sistemas de Localización, Transporte, Navegación Guiada y de Posicionamiento.

Cuando alguno de estos sistemas necesita informar al usuario de algún estado, ya sea, por ejemplo, notificarle sencillamente que debe esperar cuando una nueva escena está cargando, se lo hace saber al prefab H_Persistente, el que, a partir de los parámetros recibidos en función del sistema que ha generado el aviso, carga en escena el presentador de texto adecuado, encargado de transmitirle al usuario el mensaje deseado.

12.2.1. Presentadores de texto

Los presentadores de texto son los prefab H_PresentadorCargandoDestino y H_PresentadorNombreDestino, controlados respectivamente por los guiones Presentador-

(a) Cambio de escena con el mapa maximizado (b) Un nuevo destino se está cargando

Figura 12.1: Prefab H_PresentadorCargandoDestino en acción

CargandoDestinoControl.js y PresentadorNombreDestinoControl.js, y son los encargados de interpretar y configurar visualmente el mensaje a mostrar.

El primero de estos presentadores, H_PresentadorCargandoDestino, el más sencillo de ambos, gestiona dos mensajes genéricos, *«Cargando...»* y *«Cargando destino...*, utilizados para informar al usuario de que se está cargando una escena cuando el mapa se encuentra en modo a pantalla completa el primero (Figura 12.1a), y el segundo, cuando el usuario activa el Sistema de Transporte (Capítulo 13), para informarle de que se está cargando el destino hasta que la nueva escena haya cargado por completo (Figura 12.1b). Cumplida alguna de estas dos circunstancias, el presentador se activará por instrucción del prefab H_Persistente y se destruirá una vez la nueva escena haya cargado.

El presentador H_PresentadorNombreDestino comparte con el anterior la tarea de mostrar mensajes informativos, pero contiene una carga técnica mayor y su vinculación con los sistemas que dependen de él es también más profunda.

Es el encargado de mostrar en pantalla el nombre del área en el que se encuentra el avatar, como ya se ha visto, cuando el usuario activa el Sistema de Localización; de mostrar el destino al que el avatar acaba de transportarse utilizando el Sistema de Transporte; o de anunciar, cuando el Sistema de Navegación Guiada (Capítulo 14) está activo, que ésta se ha completado; y lo hace, distintivamente, temporizando y gestionando el tiempo que el mensaje permanece en pantalla.

A diferencia de su homónimo más simple cuya activación implica siempre la destrucción de una escena y, por tanto, automáticamente la suya propia cuando la siguiente se carga; este presentador se activa en el caso opuesto, en el instante de cargar una nueva escena o durante la ejecución de ésta. Esto implica tener que controlar cuándo debe

desaparecer el mensaje y por ello debe disponer de un código de control del tiempo.

Este temporizador interno, configurable cómodamente como se verá más adelante, tiene además en cuenta aquellos escenarios durante los que el mensaje debe dejar de mostrarse temporalmente, como cuando el usuario activa un menú, por ejemplo el menú de selección de idiomas del Sistema de Traducción (Capítulo 16), y retoma cuando esta eventualidad finaliza (cuando el usuario ha salido del menú) en el mismo punto en que la cuenta atrás se detuvo; de tal forma que si el tiempo total configurado durante el que se debe mostrar el mensaje es de tres segundos, y el menú de ejemplo se ha activado en el segundo uno y ha permanecido abierto durante seis, cuando se cierre el menú, el mensaje seguirá mostrándose durante otros dos segundos, hasta completar el total de los tres tal y como se configuró inicialmente.

Aspectos detallistas como el descrito en el párrafo anterior, si bien pueden parecer insignificantes o accesorios a primera vista cuando se los compara, por ejemplo, con funciones primordiales como las que el Sistema de Captura de Mapas ofrece para capturar la totalidad de un terreno, son, sin embargo, los que el usuario nota primero en falta cuando no se pone el cuidado de incluirlos. Sabiendo esto, para lograr que el usuario jamás tenga la sensación de que falta algo, aun cuando no sepa con certeza el qué, este Proyecto ha puesto tanta atención a los detalles como la ha puesto a las funciones primordiales.

Resumen de los tipos de mensajes que gestionan

Se recogen a continuación, a modo de resumen, los principales tipos de mensajes que gestionan los presentadores de texto.

- Mensajes de cambio de escena en el modo de mapa a pantalla completa.
- Mensajes de carga de destino del Sistema de Transporte.
- Mensajes informativos sobre la ubicación de destino del Sistema de Transporte.
- Mensajes del Sistema de Localización sobre la posición actual.
- Mensajes de navegación completada del Sistema de Navegación Guiada.

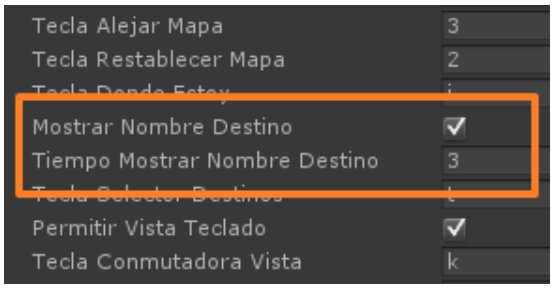


Figura 12.2: Configuración de los presentadores de texto

12.3. Configuración y personalización

Las opciones de configuración de este sistema se encuentran, como las de los demás, recogidas cómodamente en el Inspector del prefab H_CamaraMapa como puede observarse en la Figura 12.2. Aquellas otras de personalización, se recogen en los prefab de cada uno de los presentadores.

12.3.1. Configuración del comportamiento

Mostrar Nombre Destino Determina si se ha de mostrar automáticamente en pantalla el nombre del destino al que el avatar acaba de transportarse mediante el Sistema de Transporte (su desactivación no afecta a otros sistemas).

Tiempo Mostrar Nombre Destino Permite configurar el tiempo en segundos durante el que los mensajes permanecerán en pantalla (no afecta a los mensajes genéricos).

12.3.2. Configuración del aspecto

Como ya se ha mencionado antes, se ha puesto un especial énfasis en cuidar el aspecto visual de los mensajes que genera en pantalla este sistema. Se han probado numerosas de configuraciones, tipografías, colores y combinaciones para garantizar un aspecto final impecable. Todo estos aspecto son, no obstante, perfectamente configurables y, si el desarrollador desea modificarlos puede hacerlo a través del Inspector de ambos presentadores. Los dos comparten las mismas opciones de personalización como puede comprobarse en la Figura 12.3, y son:

Porcentaje Ancho Espacio porcentual que podrá ocupar el mensaje a lo ancho, y anchura total de la imagen de fondo.

(a) Inspector de H_PresentadorNombreDestino (b) Inspector de H_PresentadorCargandoDestino

Figura 12.3: Personalización de aspecto de los presentadores de texto

Porcentaje Alto Espacio porcentual que podrá ocupar el mensaje a lo alto, y altura total de la imagen de fondo.

Margen Inferior Factor de la altura que determinará la distancia de separación del texto respecto del borde inferior de la pantalla (son valores posibles aquellos entre 1 y 2, ambos incluidos)

Textura Fondo Textura que se dibujará tras el fondo (se incluye una textura transparente en la carpeta «Proyecto H/General/Fondos» para aplicar en caso de que no se desee un fondo).

Estilos Configuran el aspecto del texto, tipografía, forma, grosor, color, márgenes, relleno, alineación, etcétera (Figura 12.4).

12.4. Autoajustable a los cambios de resolución

De la misma forma que se destacó la capacidad de toda la interfaz del Sistema de Posicionamiento de ajustarse automáticamente a los cambios de resolución y adaptarse a diferentes tamaños de pantalla, se puede hacer lo mismo para la interfaz de los presentadores de texto, pues los dos incluyen en sus guiones controladores el código que hace esto posible.

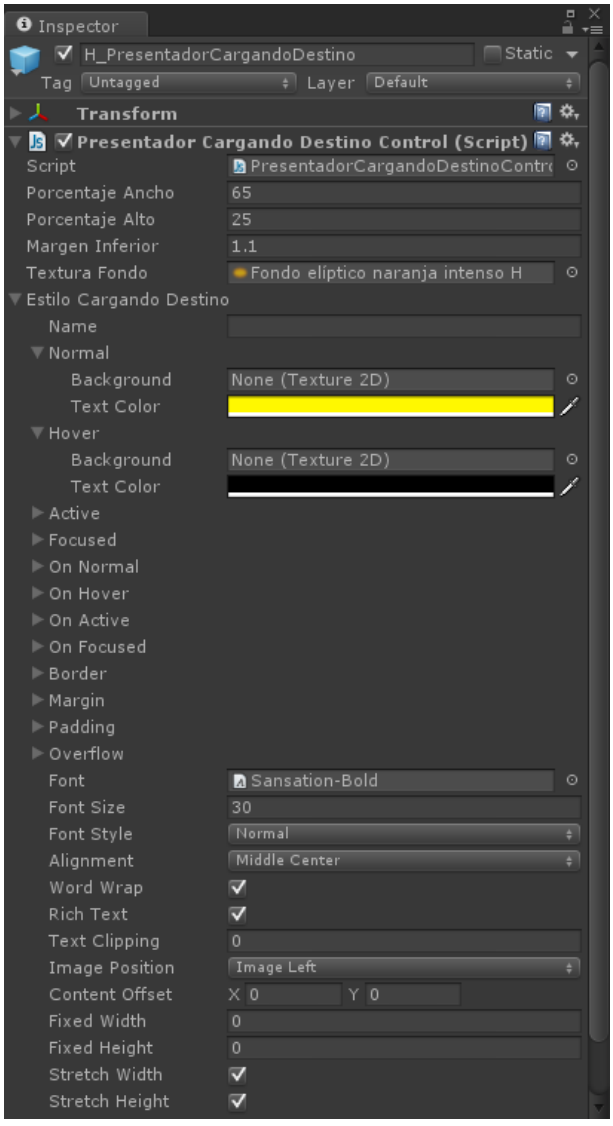


Figura 12.4: Configuración del texto mediante «estilos»

Capítulo 13

Sistema de Transporte

En el mundo que nos rodea, el mundo real, cualquier desplazamiento que se realice exige invertir en él un preciado tiempo para completarlo, para ir desde el lugar de origen hasta el punto final de destino; esto es un hecho, y tiene un inconveniente, tan natural como evidente, que el tiempo es oro.

En el mundo virtual, sin embargo, los hechos inamovibles pueden dejar de serlo, las leyes físicas pueden romperse con unas cuantas líneas de código, y esta es precisamente la función del Sistema de Transporte, permitir lo impensable en el mundo real, que con tan sólo pulsar un botón el usuario pueda desplazarse inmediatamente de un lugar a cualquier otro.

Este flexible sistema es capaz de gestionar todo tipo de reubicaciones. Eficaz y eficiente, permite añadir muy fácilmente nuevos puntos de destino a su base de datos, o incluso definirlos al vuelo a partir de las coordenadas del lugar destino. Si tiempo es oro, el Sistema de Transporte existe para que nada se pierda de ese tesoro.

13.1. Componentes

Guiones

- CamaraMapaControl.js
- TransporteSDN.js

- InterfazDestinos.js
- PersistenteControl.js
- SDN.js
- SustContr.js (guión legado, modificado y mejorado)
- PosicionadorDesactivador.js

Prefabs

- H_CamaraMapa
- H_InterfazDestinos
- H_Persistente
- H_Posicionador y H_CamaraPosicionador

Recursos

- Texturas de la interfaz organizadas en la ventana de proyecto dentro de las carpetas «Proyecto H/Interfaz/General», «Proyecto H/Interfaz/InterfazDestinos» y «Proyecto H/Interfaz/InterfazMapa»,

13.2. Funcionamiento

El corazón de este sistema se encuentra en el guión TransporteSDN.js. Es el que proporciona los recursos que posibilitan el transporte, las funciones que lo activan y la base de datos de destinos.

El intermediario, el prefab que creará la interfaz con la que el usuario se comunicará con el Sistema de Transporte, es H_InterfazDestinos, cuya clonación, nada más cargar la escena, corre a cargo del prefab central H_CamaraMapa.

Cuando el usuario activa el menú de destinos, H_InterfazDestinos le presenta un listado con aquellos configurados en la base de datos del sistema. En el momento que se selecciona uno de ellos, este prefab transmite esa información al guión TransporteSDN.js, que pone inmediatamente en marcha la gestión de la reubicación del avatar, administrando adecuadamente la destrucción y creación de escenas y el posicionamiento posterior del avatar en las coordenadas exactas asociadas al punto que se desea alcanzar.

Es un sistema no sólo eficaz, sino también eficiente. Tiene en cuenta la ubicación actual y la compara con la de destino para cargar una nueva escena sólo cuando sea necesario, así, cuando se detecte que el lugar de destino se encuentra en el área actual del avatar, el transporte consistirá en la relocalización controlada del avatar, modificando sus coordenadas sin afectar a ninguno de los elementos de la escena actual, optimizando de esta modo el uso de los recursos.

Cuando el transporte se realiza entre escenas, no de forma local, y es necesario destruir una y recrear otra, la clonación última del avatar, es decir, su recreación en la nueva escena en las coordenadas predefinidas, se realiza, como último paso, a través del guión legado SustContr.js, garantizando así la absoluta armonía entre el proyecto legado y este (ampliado en la Sección 13.11 del presente capítulo).

El guión SDN.js aporta aquellas variables y funciones globales que el sistema utiliza para, entre otras, pausar y reanudar la aplicación al activar el usuario el menú de destinos o albergar información interna útil sobre el lugar de destino. En particular, el código de pausado de la aplicación garantiza que, cuando el usuario active el menú de destinos y se desplace por él con el ratón, no moverá al mismo tiempo la cámara del avatar como efecto colateral, algo que desconcertaría al usuario. De esta forma, gestionando el tiempo de ejecución de la aplicación y reduciéndolo a cero para, efectivamente, pausarla, se consigue que este menú, y cualquiera de los otros sistemas como ya se verá, parezca funcionar en una franja de tiempo completamente independiente.

Finalmente, mientras se ejecuta y una vez se ha completado el transporte, los presentadores de texto, en combinación con el Sistema de Traducción, se encargan de mantener informado al usuario, avisándole de que hay una escena cargándose o informándole de que se ha llegado a destino, anunciándole por pantalla en el idioma elegido el nombre de la nueva área en la que se encuentra.

13.3. Utilización

La programación interna del sistema se ha configurado para que sea extremadamente sencillo de utilizar. Así, cualquier desarrollador futuro que desee integrar en su propio proyecto las funciones que el Sistema de Transporte ofrece podrá hacerlo sin encontrar dificultad alguna, sólo tendrá que recurrir a cualquiera de las dos instrucciones siguientes para activar el sistema.

```
1 TransporteSDN.Destino("Lugar")
2 TransporteSDN.Destino(X, Y, Z, rotacionY, "Nombre de la escena", "Etiqueta a mostrar").
```

La primera coteja el destino con los incluidos en la base de datos interna y recoge de ahí las coordenadas de transporte.

La segunda permite indicar manualmente las coordenadas X, Y y Z del punto de destino, la rotación Y del avatar con la que aparecerá tras transportarse, el nombre interno de la escena a la que transportarse y el texto informativo que se mostrará en pantalla al llegar al destino[1]. Este método de uso es completamente independiente de la base de datos, rápido y cómodo sin necesidad de gestión alguna adicional.

13.4. Ampliación de la base de datos de destinos

Como se ha visto, la primera de las dos instrucciones citadas en el apartado anterior coteja el nombre de destino con la base de datos interna del sistema. Esta base de datos es, sencillamente, una manera organizada de almacenar las características de ubicación de los lugares de destino, como sus coordenadas o escena en la que se encuentran, y poder referirse a ellos posteriormente sin necesidad de tener que definir de nuevo cada vez cada uno de estos valores de ubicación.

La base de datos, albergada en el guión TransporteSDN.js, incluye una plantilla que el desarrollador podrá utilizar para añadir nuevos destinos. Un ejemplo tipo de destino tiene la siguiente forma:

```
1 case "EscuelaPlantaBaja":
2
3   nombreDestino = "Escuela de ingenierías\nPlanta baja";
4
5   posicionX = 18.95518;
6   posicionY = 0.5336218;
7   posicionZ = -46.98695;
8
```

[1]Gestionado por el presentador de nombres de destino H_PresentadorNombreDestino.

```
9   rotacionY = 45;
10
11  escena = "02a EscuelaPB";
12
13  break;
```

El primero de los valores «EscuelaPlantaBaja», se corresponde con el nombre personalizado que definirá a este destino, el que se usará dentro de la instrucción abreviada para llamar al Sistema de Transporte.

El siguiente valor «Escuela de ingenierías\nPlanta baja», es el mensaje que los presentadores de texto mostrarán por pantalla al completarse el transporte. Puede asignarse cualquier texto, incluso uno vacío si no se desea que para este destino en particular se activen los presentadores. El valor aquí indicado lo utilizará también el Sistema de Traducción para cotejarlo con su propia base de datos, devolviendo una cadena de texto traducida si encuentra coincidencia. Asimismo, cabe destacar que pueden utilizarse aquí caracteres de formato, como el incluido en el ejemplo, «`\n`», que añade una nueva línea en ese punto, logrando así que el texto ocupe dos renglones cuando se muestre por pantalla.

Los siguientes valores especifican la posición en la que debe aparecerá el avatar y su rotación u orientación, es decir, el lugar hacía el que mira.

Por último, en el valor «escena» se especificará el nombre interno de la escena en la que se encuentre este punto de destino.

13.5. Prefab de ayuda al cálculo de las coordenadas de destino

El sistema incluye un prefab independiente, H_Posicionador creado para asistir visualmente al desarrollador que desea añadir un punto de transporte en el cálculo de las coordenadas (Figura 13.1).

Este elemento está constituido por una cápsula con dimensiones similares a las del avatar, por una cámara colocada de tal forma que replica la visión que tendría el usuario al manejar el verdadero avatar, y por un guión controlador PosicionadorDesactivador.js.

Como se puede ya intuir, la función de este objeto es la de replicar al avatar en la ventana de escena y de ofrecerle al desarrollador, mediante su cámara y a través de la ventana de juego, la vista que tendría el usuario durante la ejecución de la aplicación de encontrarse en el punto en el que está el prefab. De esta manera, el desarrollador puede mover y rotar libremente el posicionador por la escena, comprobar que lo que vería el

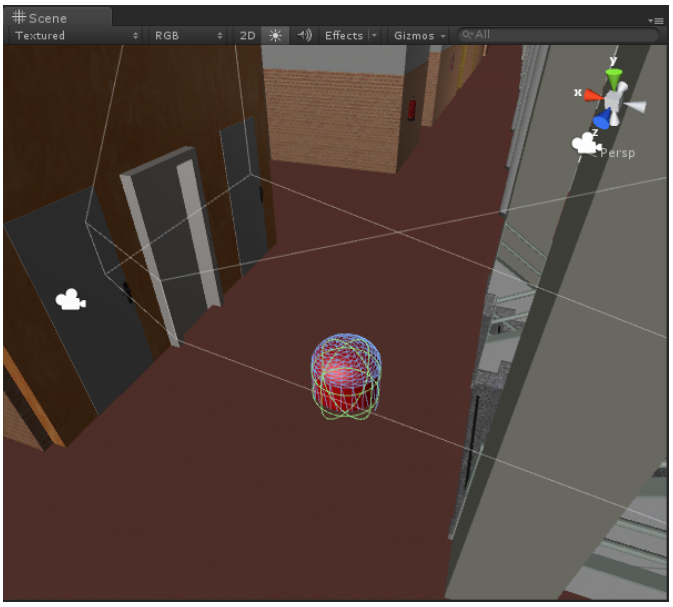


Figura 13.1: Elemento posicionador de ayuda al cálculo de coordenadas

usuario donde lo ha colocado es lo que querría que viera de transportarse a ese punto, y anotar las coordenadas para poder introducirlas luego cómodamente en el Sistema de Transporte.

El método alternativo de calcular las coordenadas sin este prefab específico, sería ejecutar la escena, avanzar manualmente con el avatar hasta donde sea que se pretende crear un punto de destino, y comprobar ahí sus coordenadas en el componente *Transform* de su Inspector. Parar luego la escena, introducir las coordenadas en el sistema, ejecutar otra vez la escena y comprobar que esas son las coordenadas adecuadas. Mientras que es un método válido, utilizar el prefab que se ha creado para esta tarea puede llegar a suponer un ahorro considerable de tiempo.

La función del guión controlador es la de asegurar que cuando la escena se ejecuta, este prefab permanece siempre desactivado. Al poseer una cámara este posicionador, configurada además para otorgarle prioridad sobre las demás[2], si no se recuerda desactivarlo después de utilizarlo, al ejecutar la aplicación el usuario vería sólo lo que fuera que viera este prefab desde su posición, ignorando por completo la cámara del avatar. Para evitar esta situación, el guión PosicionadorDesactivador.js desactiva el prefab automáticamente, aun si se dejó activado por olvido: como si nunca hubiera estado ahí.

[2]La prioridad de las cámaras (la propiedad *Depth* en su Inspector) es una característica que define cuál tiene preferencia para dibujar en la pantalla, pues cámaras puede haber muchas en una escena, pero pantallas, por lo general, sólo una

Figura 13.2: Botón de transporte normal, activo y pulsado

13.6. Interfaz

La interfaz de este sistema está constituida, por un lado, por el menú de destinos donde el usuario elige el lugar al que desplazarse y, por otro, por los botones y teclas asignadas que activan dicho menú. Todas estas formas de interacción se agrupan a continuación:

13.6.1. Botones

 Activar el menú de destinos desde el mapa en modo normal.

El aspecto final de este botón, como el de todos los incluidos en el Proyecto, se ha elaborado minuciosamente, y así puede apreciarse en sus diferentes estados en la Figura 13.2 y su proceso de creación en la Figura 13.3.

 Con el menú de selección de destino activo, permite salir de él sin realizar ningún transporte.

13.6.2. Teclado

Activar el menú de destinos desde cualquier modo de mapa Tecla predeterminada: **T**.

13.6.3. Menú de destinos

El aspecto visual de este menú se define mediante el guión InterfazDestinos.js. Es el resultado de una minuciosa distribución de los elementos a partir de numerosas instrucciones anidadas de tipo `GUILayout`, y de configurar cada detalle estético de las

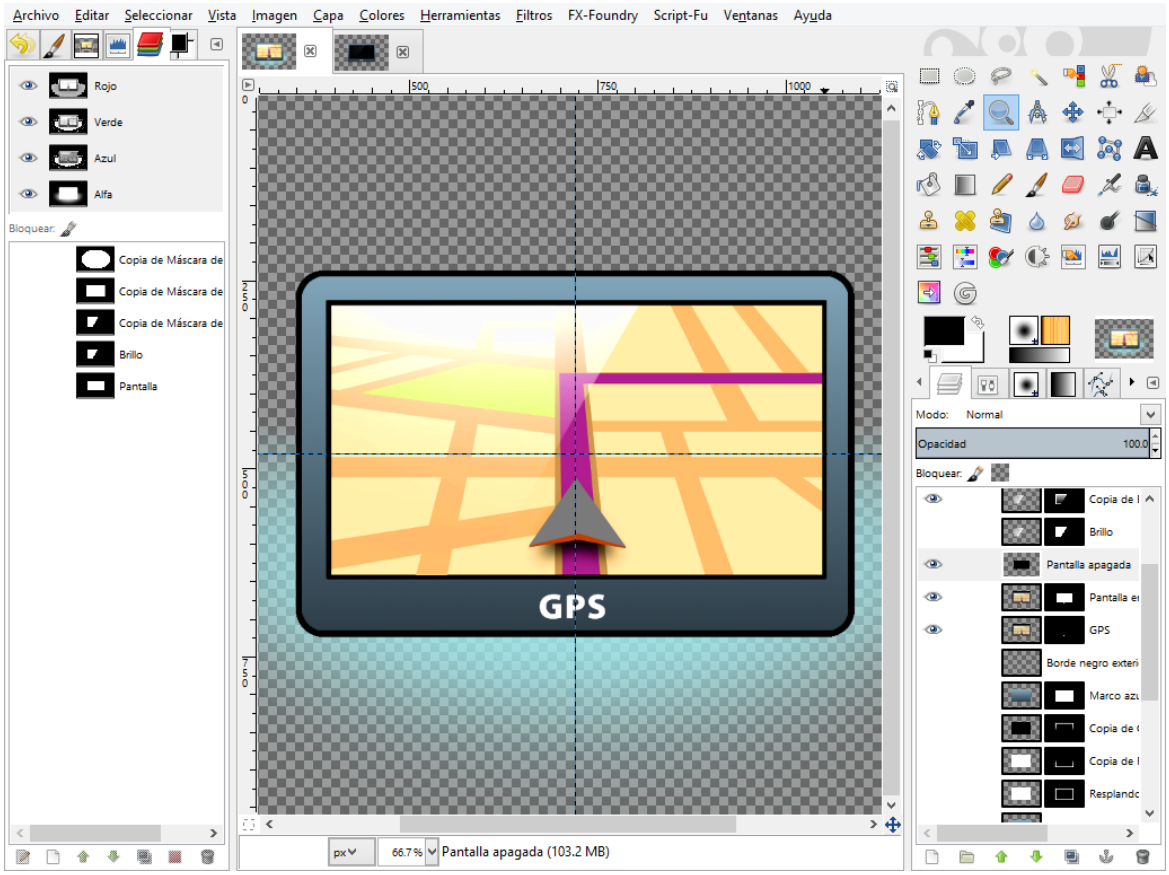


Figura 13.3: Proceso de creación del icono activador del menú de destinos

texturas de los fondos, botones, de cada uno de los iconos, desde el concepto base hasta el resultado final; colores, tamaños, gradientes, marcos, todo se ha cuidado a nivel de píxel.

Se han introducido también detalles visuales adicionales complementarios como el efecto del difuminado[3] de los elementos de la escena, redondeado así el efecto visual de todo el conjunto.

En la Figura 13.4 se puede ver el menú en acción (las texturas y botones que se observan y no se han descrito en el apartado anterior correspondiente son parte del Sistema de Navegación Guiada y se explicarán en el Capítulo 14). En la Figura 13.5 una etapa del proceso de creación de las texturas.

[3]Sólo disponible en la versión PRO de Unity, no obstante, en caso de ejecutarse la aplicación en una versión normal, el Proyecto incluye el código de control necesario para desactivar estas funciones avanzadas y avisar al desarrollador de ello.

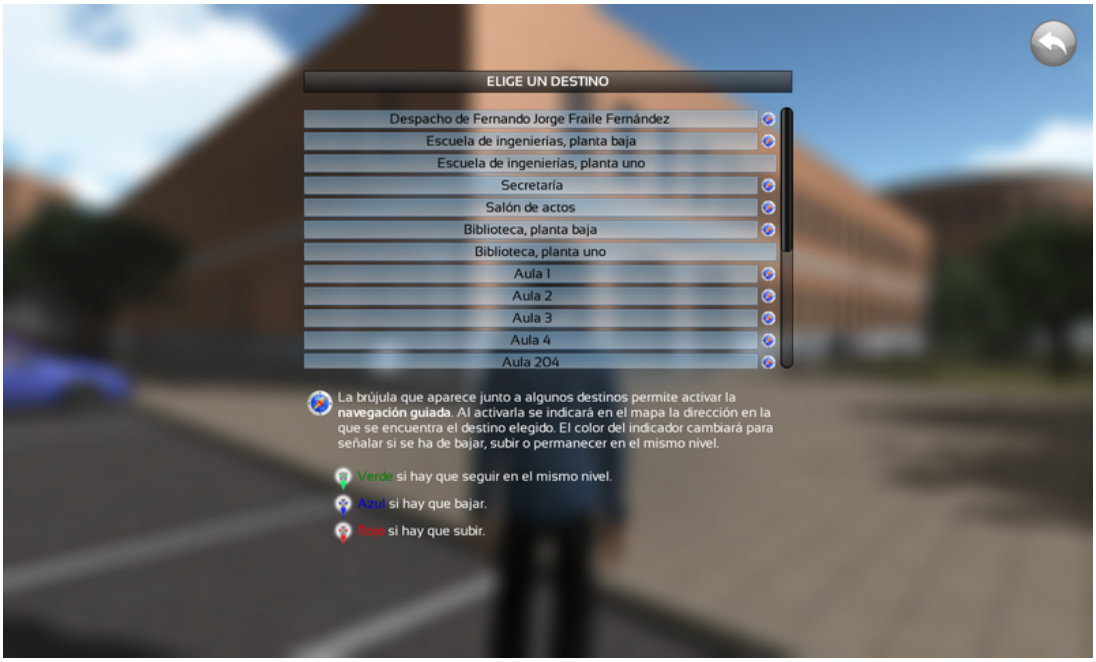


Figura 13.4: Menú de destinos

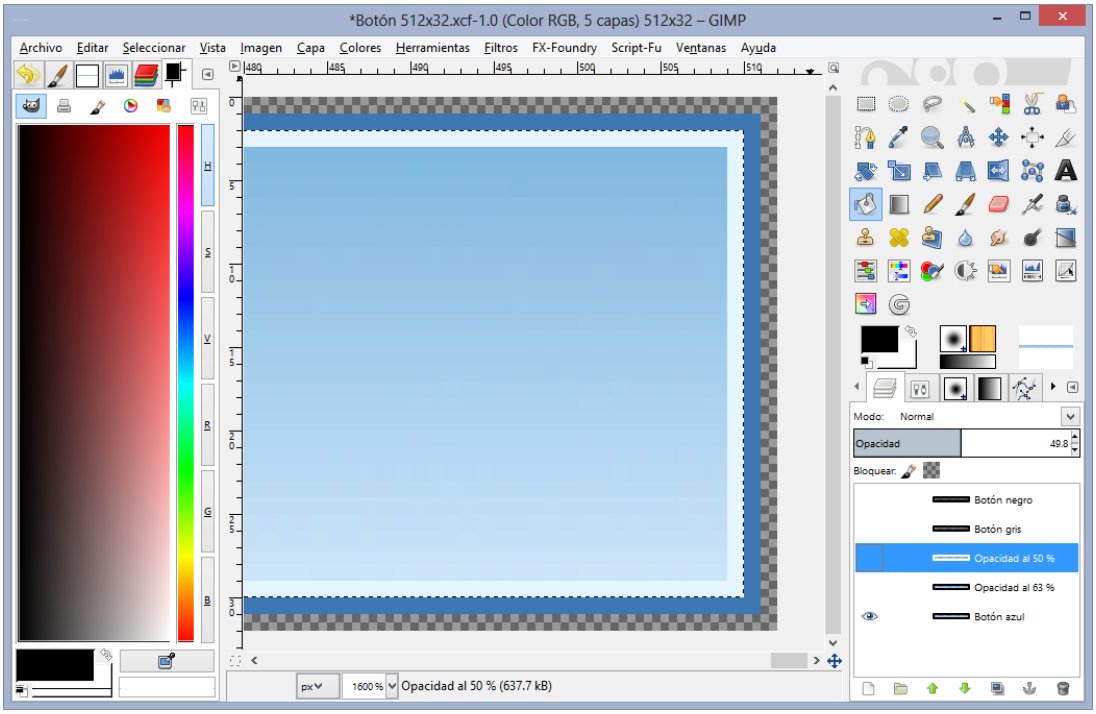


Figura 13.5: Creación de una textura para el menú de destinos

13.7. Autoajustable a los cambios de resolución

Todos los elementos del menú de destinos y el botón general de activación del sistema se adaptan inteligentemente a los cambios de resolución modificando en tiempo real sus dimensiones.

El código incluido en los guiones InterfazDestinos.js e InterfazMapaControl.js hace esto posible, vigilando continuamente los cambios de resolución que se produzcan, de forma similar a lo planteado en la Sección 9.7.

13.8. Configuración y personalización

13.8.1. Configuración del comportamiento

La configuración técnica de este sistema se corresponde con la de presentadores de texto, pudiendo configurar tanto si se desea que se muestren los mensajes informativos de que el transporte se ha realizado, como el tiempo que permanecerán estos en pantalla. Es válido aquí, por tanto, todo lo que se dijo en la la Sección 12.3.1 de configuración correspondiente a los presentadores.

Además de las funciones relacionadas con los presentadores también puede configurarse al gusto la tecla activadora del menú de destinos.

En la Figura 13.6, se pueden observar estos tres valores personalizables, todos recogidos en el Inspector del prefab H_CamaraMapa, a saber:

Mostrar Nombre Destino Determina si se ha de mostrar automáticamente en pantalla el nombre del destino al que el avatar acaba de transportarse mediante el Sistema de Transporte.

Tiempo Mostrar Nombre Destino Permite configurar el tiempo en segundos durante el que los mensajes permanecerán en pantalla.

Tecla Selector Destinos Define la tecla asignada a la función de abrir el menú de destinos.

Figura 13.6: Configuración del Sistema de Transporte

13.8.2. Configuración del aspecto

El prefab H_InterfazDestinos (Figura 13.7) contiene todos los valores personalizables que determinan el aspecto final del menú de destinos, desde su tamaño y posición hasta el aspecto de sus elementos gráficos.

Los valores correspondientes a este sistema son:

Porcentaje Ancho Determina porcentualmente el tamaño a lo ancho del menú.

Porcentaje Alto Determina porcentualmente el tamaño a lo alto del menú.

Porcentaje Boton Regresar Valor porcentual respecto de la altura de la pantalla que determina el tamaño del botón para abandonar el menú sin elegir un destino.

Porcentaje Margen Boton Regresar Valor porcentual respecto de la altura de la pantalla que determina la distancia de separación entre el citado botón y el borde de la pantalla.

Textura Fondo Define una textura que se colocará bajo los elementos del menú.

Estilos Permiten modificar los elementos gráficos de los botones a los que hace referencia su nombre. Los diferentes estados de los botones, normal, activo, pulsado o elegido, se modifican aquí.

Asimismo, si se desea cambiar el tamaño o aspecto del botón activador del menú de destinos, puede hacerse en el prefab H_InterfazMapa, modificando los siguientes campos

Factor Navegador Factor que, a partir del porcentaje de los puntos cardinales de la interfaz del minimapa, indica el tamaño del botón para abrir el menú de destinos.

Estilo Navegador Permite modificar el aspecto final del citado botón, en particular, sus

diferentes estados «normal», «activo» y «pulsado».

13.9. Integración con el Sistema Traductor

El Sistema de Transporte se integra con el de Traducción haciendo posible que los mensajes informativos, aquellos con el nombre del lugar de destino que el usuario recibe al completar un transporte, y aquellos genéricos de tipo «Cargando destino...» que se producen cuando este se realiza entre diferentes escenas, los recibe en el idioma que se haya elegido.

Si de las dos instrucciones posibles para llamar al sistema se utiliza la abreviada, la integración se realiza a partir de los campos `nombreDestino` incluidos para cada destino en la base de datos de estos; si por el contrario se utiliza la que permite asignar los campos manualmente, la integración se realiza entonces a partir del valor asignado al último de esos campos.

El Sistema de Traducción cotejará esos campos con las traducciones de su base de datos y, si encuentra coincidencia, devolverá la traducción asignada, en caso contrario, devolverá el campo original.

13.10. Integración con el Subsistema de Información por Pantalla

Como se ha mencionado ya, este subsistema es el que se encarga de mostrar en pantalla, tras realizar un transporte, el nombre del área de destino, generado por este Sistema de Transporte y traducido por el de Traducción; y, cuando el transporte se realiza entre escenas, de mantenerle informado del proceso de carga, filtrado también por el traductor.

13.11. Integración con el proyecto legado

El proyecto legado incluye un sistema básico de transporte. Este se activa únicamente cuando el avatar acciona un objeto de tipo *Is Trigger* colocado, por ejemplo, en el vano de una puerta que conduce a un aula que está modelada en otra escena. De esta forma se logra crear una sensación de continuidad entre escenas a partir de sus uniones naturales, sean puertas, escaleras, o cualquier otra similar.

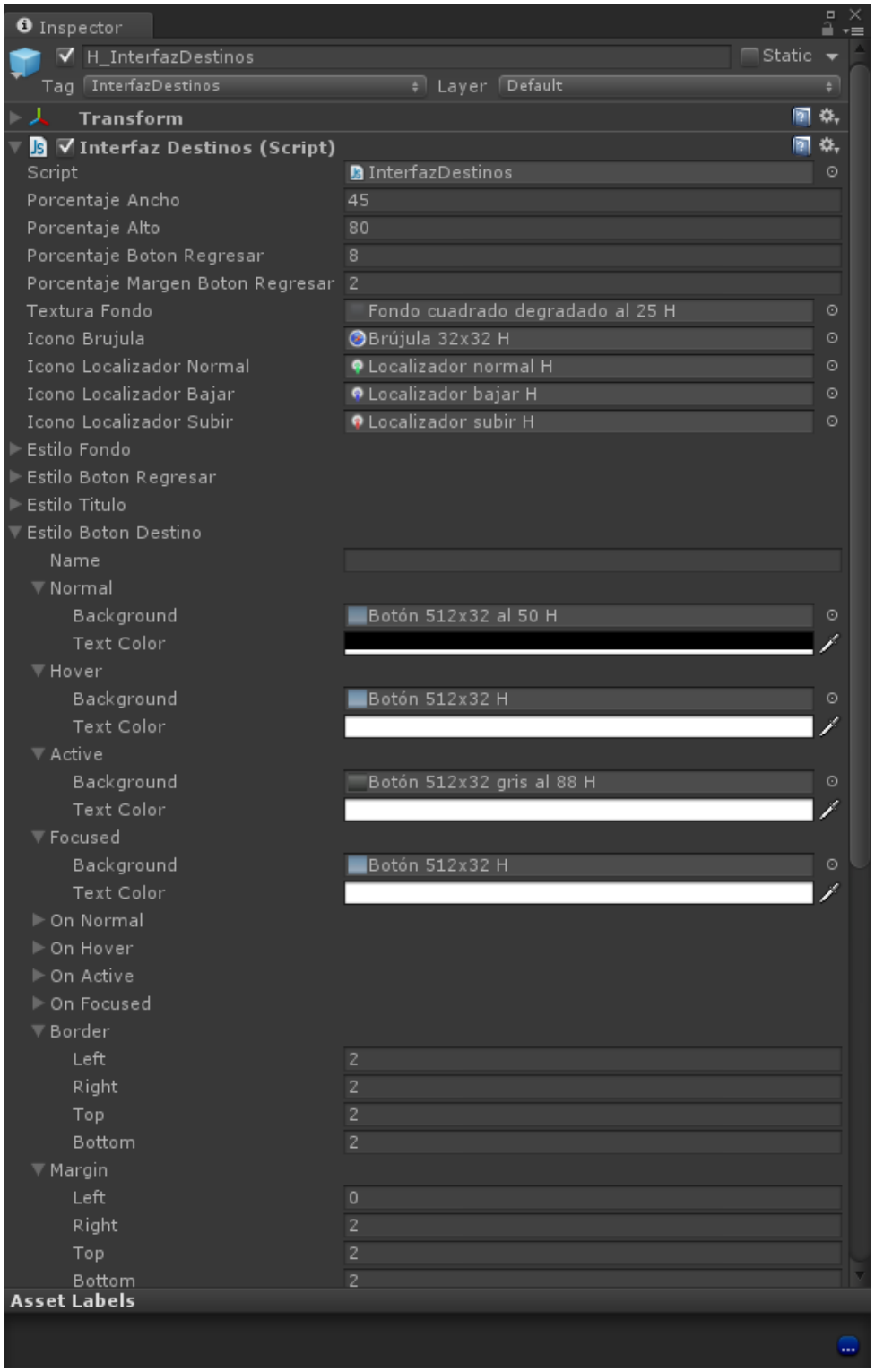


Figura 13.7: Personalización de menú de destinos

El nuevo sistema de transporte que presenta este Proyecto es, planteado de base como herramienta multifunción, más avanzado y flexible y puede sustituir efectivamente al anterior, simplificando de hecho sobremanera el tipo natural de cambios de escena que gestiona. Sin embargo, para no alterar la esencia del proyecto legado, en nuevo sistema se ha integrado de tal forma que ambos convivan armoniosamente, pudiendo, en el futuro, fácilmente sustituir el anterior por el nuevo si así se desea.

Esta integración se realiza en el guión legado SustContr.js, modificado y mejorado por este Proyecto para ajustarse correctamente a los ciclos de ejecución, tal y como se verá en el Capítulo 18.

13.12. Integración con los sistemas de Vista Alternativa, Traducción y Apagado

Como se verá en la Sección 15.6 más detalladamente, la relación entre estos cuatro sistemas es estética. Surge de la posibilidad que tiene el desarrollador de desactivar individualmente tanto el Sistema de Vista Alternativa como el de Traducción.

Cuando alguno de estos sistemas se anula, su icono, ubicado junto a los de los demás en una misma fila en la esquina inferior derecha de la pantalla, desaparece.

Para evitar que quede un vacío en mitad de la fila de iconos, se ha incluido un código en el guión InterfazMapaControl.js que elimina el hueco desplazando hacia la derecha los iconos de los sistemas que sí estén activos.

Capítulo 14

Sistema de Navegación Guiada

El sistema desarrollado en el capítulo anterior se caracteriza por proporcionar una forma de transporte inmediata, pero, ¿y si el usuario desea llegar a un destino recorriendo él el camino? El Sistema de Navegación nace para ofrecerle esta alternativa, y lo hace con las mismas garantías que el otro sistema le ofrecía: el usuario sabe con certeza que, usando cualquiera de los dos, siempre llegará a su destino.

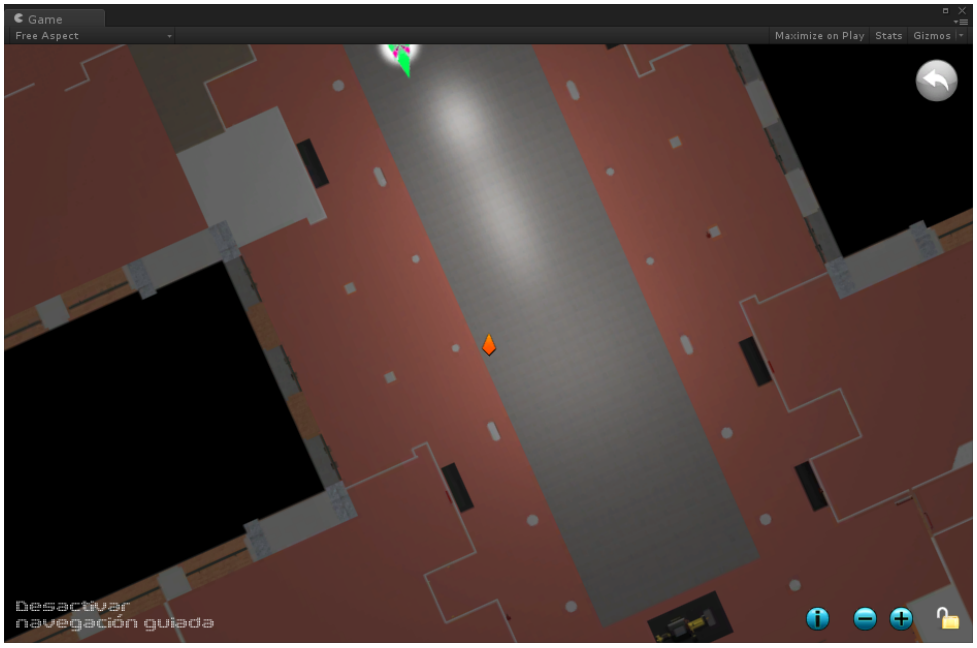
Una vez activado este sistema, en el borde del mapa aparecerá una referencia continua de posición, un localizador (Figura 14.1), indicador de la dirección hacia la que el usuario debe avanzar para alcanzar el destino elegido. No importa dónde se encuentre, lo lejos que esté, o los saltos de escena que haga, esa referencia ajustará continuamente su emplazamiento en el mapa en función de la posición del avatar con respecto al destino que el usuario a elegido.

Uno automático e inmediato, otro manual y guiado, la decisión de qué sistema usar es ahora del usuario, pero sea cual sea el que elija, tendrá siempre la certeza de que nunca se perderá.

14.1. Componentes

Guiones

- CamaraMapaControl.js



(a) En mapa a pantalla completa

(b) En mapa en modo normal

Figura 14.1: Durante la navegación guiada aparece un localizador en el mapa indicando el camino

- IntersectadorCamaraMapaPerimetro.js
- CamaraMapaPerimetroControl.js
- LocalizadorDeBalizas.js
- InterfazDestinos.js
- PersistenteControl.js
- PresentadorNombreDestinoControl.js
- SDN.js

Prefabs

- H_CamaraMapaPerimetro
- H_IndicadorAvatar
- H_LocalizadorBajar
- H_LocalizadorNormal
- H_LocalizadorSubir
- H_InterfazDestinos
- H_InterfazMapa
- H_Persistente
- H_PresentadorNombreDestino

Objetos

- Balizas agrupadas en las escenas bajo el prefijo común «H_Baliza» (en el Anexo E se recoge una relación de las todas las balizas utilizadas).

Recursos

- Texturas de los localizadores organizadas en la ventana de proyecto dentro de la carpeta «Proyecto H/Interfaz/Localizadores».
- Texturas del menú de destinos, específicas de este sistema, organizadas en la ventana de proyecto dentro de la carpeta «Proyecto H/Interfaz/InterfazDestinos».

Adicionales

- Todos aquellos elementos que conforman el Sistema de Localización (Sección 11.1).

14.2. Funcionamiento

Quizás el más complejo, pero dejará de serlo una vez se haya explicado su funcionamiento.

En primer lugar, como cabe ya esperar, el prefab H_CamaraMapa prepara el sistema clonando en la escena cada uno de sus componentes. Destacables son los prefab de los localizadores y el prefab H_CamaraMapaPerimetro. Preparado está también el guión independiente LocalizadorDeBalizas.js, así como la interfaz del menú de destinos, H_InterfazDestinos, compartida con el Sistema de Transporte, que activará todo el proceso de navegación guiada cuando el usuario así lo decida para cualquiera de los destinos disponibles.

El menú de destinos, junto a los botones que activan los transportes instantáneos del capítulo anterior, muestra una pequeña brújula (naturalmente configurable). Esta brújula es el punto de partida. Cuando el usuario la pulsa, todo el Sistema de Navegación Guiada se pone en funcionamiento.

Cuando esto sucede el guión IntersectadorCamaraMapaPerimetro.js, pieza clave de todo el Sistema de Navegación Guiada, presente en el prefab H_IndicadorAvatar y hasta ahora siempre inactivo, se pone en marcha. Su función es la de generar un rayo invisible que atravesará la escena hacia una baliza concreta. Cabe recordar que este último prefab es el encargado de proyectar la posición del avatar sobre los mapas según se vio en el Capítulo 9, «Sistema de Posicionamiento», y que, alineado siempre con éste, irá donde el avatar vaya.

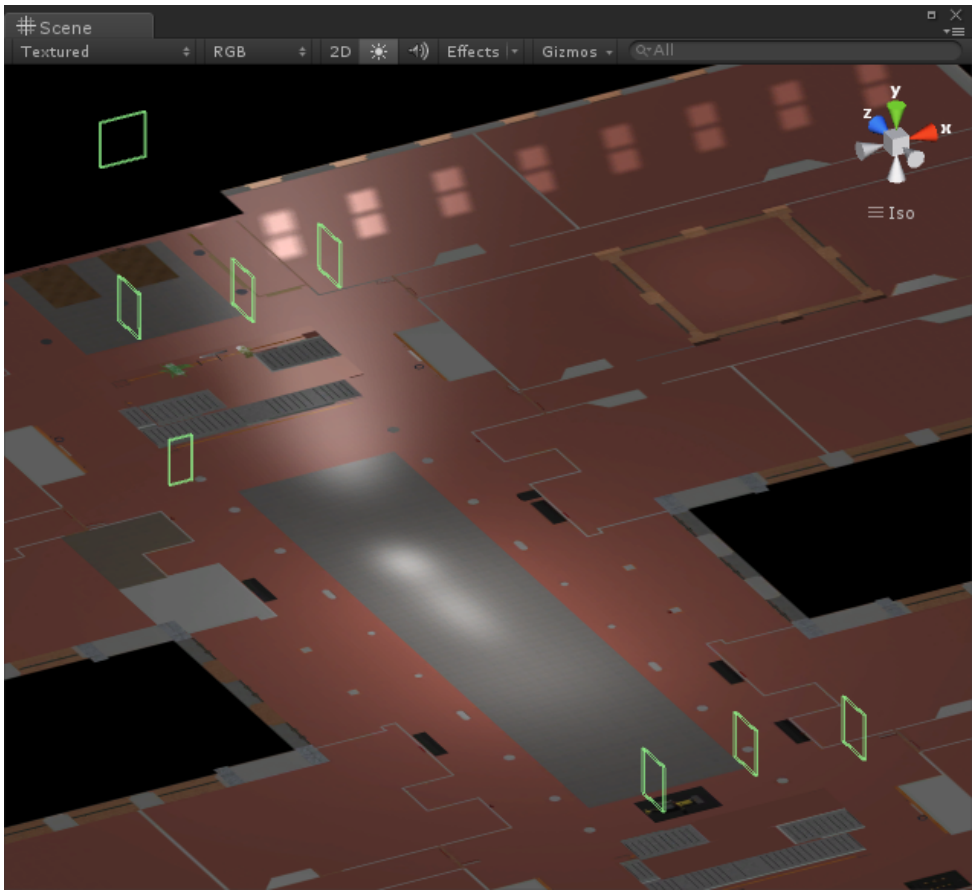


Figura 14.2: Balizas ubicadas sobre el mapa, en las entradas, salidas y escaleras

Las balizas representan las etapas que separan al avatar de su destino. Son elementos invisibles colocados en las posiciones que se corresponden con las entradas y las salidas de las escenas, y en todas aquellas zonas de paso intermedias que permitan guiar con total precisión al avatar, saltando de baliza en baliza, hasta que alcanza el lugar de destino (Figura 14.2).

Si bien Las coordenadas X y Z de las balizas son, como se ha dicho, las de las zonas de paso pertinentes, su coordenada Y se corresponde con la del prefab H_IndicadorAvatar, ubicado siempre dentro del campo de visión de la camara del mapa, de forma que el rayo generado entre ambos elementos es siempre horizontal y perpendicular a las paredes ficticias que delimitan las dimensiones de la citada cámara.

Si se combinan estas premisas, un rayo generado desde la posición del avatar hacia el siguiente punto que debe alcanzar para llegar al destino, se tiene ya la dirección de avance (Figura 14.3).

El siguiente paso será colocar, en algún punto de ese haz generado, un localizador, un elemento con una textura determinada (Figura 14.5), configurado para ser visible a

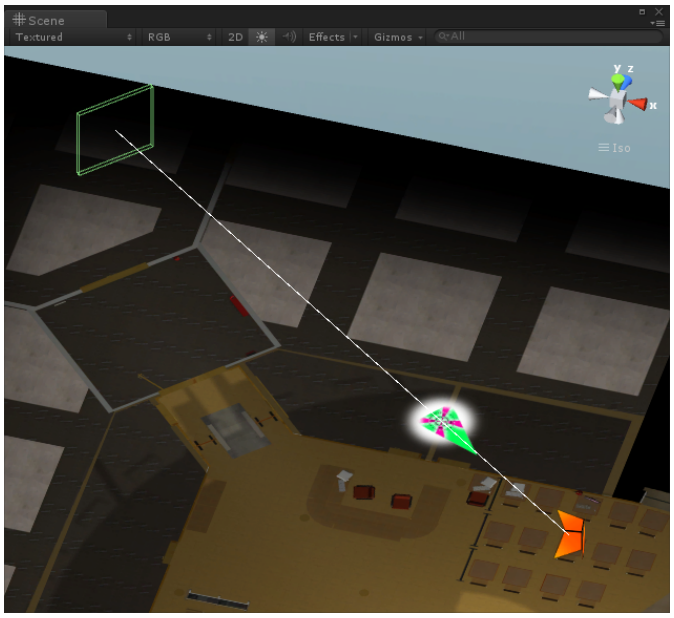


Figura 14.3: El haz indica la dirección de avance

la cámara del mapa[1] y diseñado para representar visualmente en el mismo esa dirección de avance. Entra aquí en juego el prefab H_CamaraMapaPerimetro. Este crea un cubo transparente, al mismo nivel que las balizas, cuyas dimensiones se corresponden siempre con las del área de visión de la cámara del mapa, la cámara del Sistema de Posicionamiento y siempre alineada con H_IndicadorAvatar. Las dimensiones de esta cámara definen, en última instancia, la cantidad de terreno que se muestra en el minimapa y en el mapa maximizado, y, como se ha visto anteriormente, este valor puede ser modificado libremente por el usuario, lo que significa que el perímetro de esta cámara es variable y que el cubo ha de adaptarse a él dinámicamente (Figura 14.4).

El código que posibilita la adaptación del cubo perimetral en tiempo real a las dimensiones de la cámara del mapa es el incluido en el Cuadro 14.1, y se adapta tanto a las dimensiones del mapa en modo normal cuadrado, como las del mapa en modo maximizado, cuyas dimensiones dependen del tamaño total de la pantalla.

Si con estos elementos se lograra calcular de alguna forma el punto exacto en el que el rayo generado entre una baliza y la posición del avatar intersecta al contorno del cubo perimetral (Figura 14.6), se habría determinado entonces el lugar en el que posicionar el localizador, un lugar que, por todo lo dicho, estaría dentro del campo de visión de la cámara del mapa y estaría ubicado justo en el borde del trozo de mapa que esta observa, y puesto que el usuario ve todo lo que esa cámara ve, también vería a este localizador indicándole en los límites del mapa hacia dónde avanzar.

[1]La cámara del mapa sólo ve aquellos objetos pertenecientes a la capa o *Layer* «Mapa».

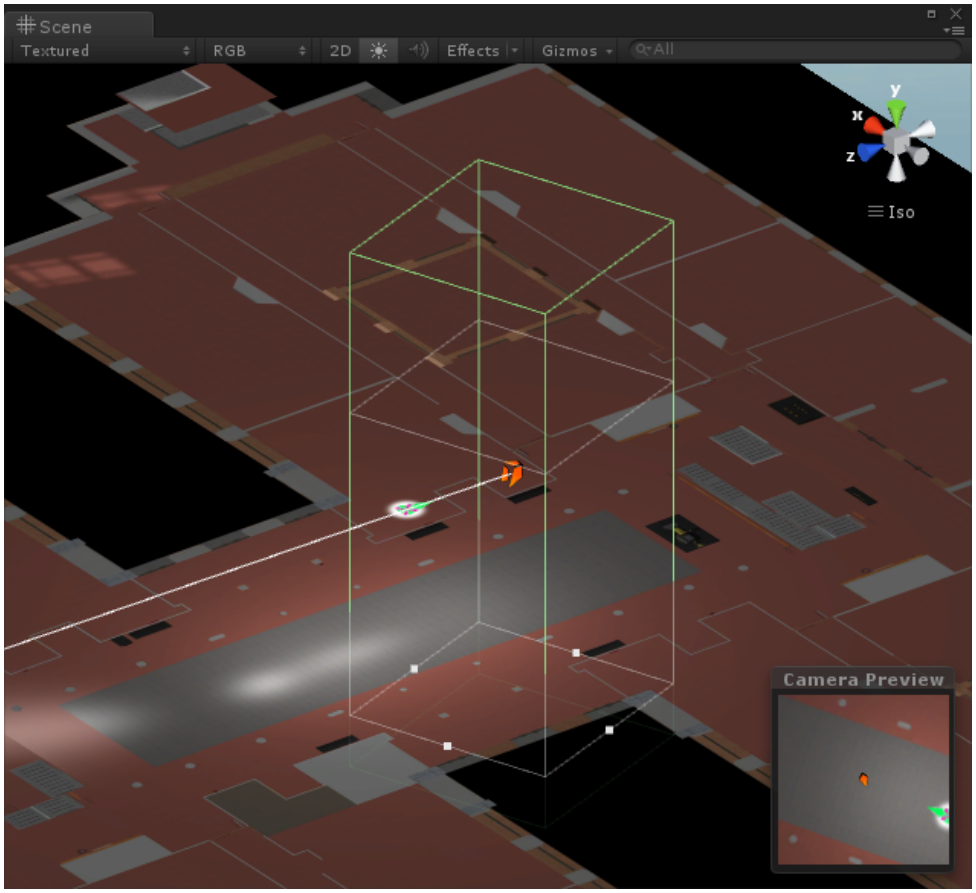


Figura 14.4: El cubo perimetral, en verde, se adapta al perímetro de la cámara del mapa, en blanco

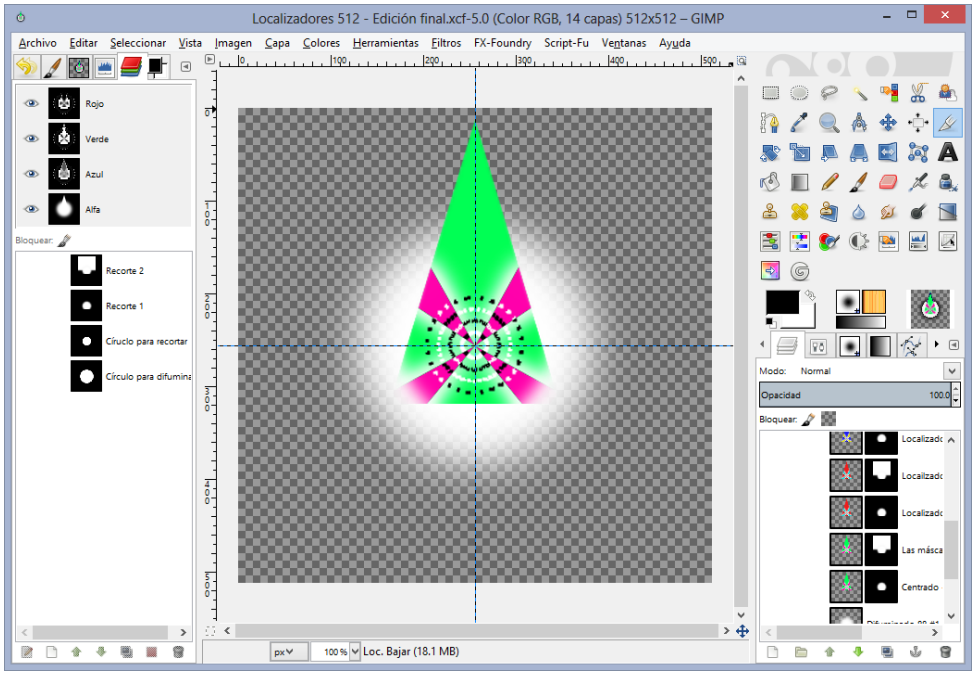


Figura 14.5: Diseño de los localizadores

```
function Update ()
{
  if (SDN.navegacionGuiada == true)
  {
    // El valor "Size" de la cámara representa la mitad de su dimensión real.

    ancho = camaraMapa.camera.orthographicSize * 2;

    // Cuando el minimapa pasa a pantalla completa pierde su proporción cuadrada
    // y la relación de sus lados depende entonces de su relación de aspecto.

    largo = ancho * camaraMapa.camera.aspect;

    transform.position = camaraMapa.transform.position;
    transform.rotation = camaraMapa.transform.rotation;
    transform.position.y = posicionAltura;

    // Redimensionamos el perímetro según cambie el perímetro del área
    // de visión de la cámara del mapa.

    transform.localScale = Vector3(largo, ancho, alto);
  }
}
```

Cuadro 14.1: Código que redimensiona el cubo perimetral en tiempo real según cambia la cámara del mapa

Pues bien, el cálculo de ese punto de intersección es posible y de ello se encarga el mismo guión que genera el haz: IntersectadorCamaraMapaPerimetro.js. El Cuadro 14.2 recoge el código que posibilita esto.

Se menciona explícitamente este fragmento de código, con los comentarios originales explicativos, por incluir solución a dos problemas importantes. Uno de ellos es que el haz se debe crear de fuera hacia adentro del cubo y no al revés. Esto ocurre porque en las formas básicas que Unity pone a disposición del desarrollador sólo sus caras externas están diseñadas para responder a este tipo de eventos. Una forma alternativa de resolver este contratiempo es la de modelar, en un programa externo, un cubo cuyas caras internas sean también activas; sin embargo, la solución planteada por este Proyecto, es mucho más límpia, flexible y elegante. Otro importante problema que este código resuelve limpia e ingeniosamente, es aquel caso en el que los dos elementos al extremo del rayo, la baliza y H_IndicadorAvatar, estén tán próximos entre sí que el haz se genere dentro del cubo perimetral, no existiendo por tanto intersección alguna. La solución planteada para este caso es, como se aprecia en el código, ubicar el localizador directamente sobre la baliza, donde permanecerá hasta que el avatar se aleje lo suficiente como para que se produzca de nuevo intersección, o accione la baliza pasando sobre ella y active así la siguiente en la ruta. Ninguna otra solución alternativa más compleja ofrece la eficacia que esta, con su sencillez, sí aporta.

En el párrafo anterior se ha avanzado un nuevo aspecto, gestionado esta vez, además de por el guión central del Sistema de Navegación Guiada, IntersectadorCamara-

Figura 14.6: La intersección del haz en el cubo perimetral define la posición del localizador

```
// Dirección y sentido del rayo. El sentido será desde la baliza hasta el localizador
// y no al revés, porque un rayo no detecta colisiones con el interior de un objeto.
// Para solucionar esto lanzaremos el rayo desde el exterior hacia el interior.

direccion = transform.position - balizaActiva.transform.position;

// Definición del rayo, partiendo de la baliza con sentido hacia el localizador.

rayo = Ray(balizaActiva.transform.position, direccion);

Debug.DrawRay(balizaActiva.transform.position, direccion);


if (Physics.Raycast(rayo, impacto, Mathf.Infinity, mascaraCapa))
{
  if (impacto.collider.tag == "CamaraMapaPerimetro")
  {
    localizador.transform.position = impacto.point;
    SDN.localizadorEnPosicion = true;
  }
}
// Si el rayo no intersecta con el perímetro de la cámara del mapa (por ejemplo, cuando
// se active la navegación guiada estando el avatar suficientemente cerca de la baliza
// activa el haz creado entre la baliza y el indicador del avatar se creará dentro del
// perímetro y por tanto no existirá intersección) se posicionará directamente el
// localizador en la posición de la baliza.
else
{
  localizador.transform.position = balizaActiva.transform.position;
}
```

Cuadro 14.2: Código que detecta el punto de intersección del haz con el cubo perimetral

MapaPerimetro.js, por otro guión principal, LocalizadorDeBalizas.js, y es la activación de la siguiente baliza en la ruta a recorrer por el avatar hasta llegar su destino.

Este segundo importante guión aporta al sistema la función global `Localizar()`, que agrupa el código que permite, en función de la ubicación actual del avatar (asignada globalmente por el Sistema de Localización como se vio en el Capítulo 11 en la variable `SDN.dondeEstoy`) y de la ubicación de destino (definida por el propio Sistema de Navegación Guiada cuando el usuario lo activa mediante el menú de destinos en la variable `SDN.dondeDeboEstar`) determinar la baliza que corresponde al siguiente tramo del recorrido guiado. Esta función asigna, en una variable estática almacenada en el guión SDN.js la referencia a esa baliza, y será posteriormente el guión central IntersectadorCamaraMapaPerimetro.js el que la active, generando ahora el haz entre ella y el indicador del avatar.

El guión LocalizadorDeBalizas.js es, en esencia, una base de datos autocotejadora, en la que se han introducido las relaciones que resuelven de manera óptima la siguiente pregunta: «Si estoy aquí, ¿cuál de todas las salidas me permite llegar más rápido allí?». En el Cuadro 14.3 puede verse un fragmento de este guión que da respuesta a ese interrogante cuando la ubicación actual es la planta uno de la Escuela de Ingenierías.

Esta pregunta se resuelve cada vez que el avatar, estando el Sistema de Navegación Guiada en marcha, cambia de escena o activa un punto de control, es decir, cuando el Sistema de Localización detecta la nueva ubicación del avatar y asigna un nuevo valor a la variable global `SDN.dondeEstoy`. Cuando esto sucede, se da también instrucción al guión IntersectadorCamaraMapaPerimetro.js de reanalizar el estado del sistema y activar la nueva baliza, respuesta a la pregunta formulada.

Todo este proceso se repite transparentemente hasta que el usuario llega al destino o este cancela la navegación guiada. Cuando lo primero ocurre, las dos variables `SDN.dondeEstoy` y `SDN.dondeDeboEstar` coinciden, momento en el que el sistema activa los presentadores de texto para notificárselo al usuario y tras ello, se apaga.

El desarrollo hasta aquí planteado ha descrito los puntos clave del proceso por el que el Sistema de Navegación Guiada calcula la ruta que el avatar debe seguir. Si sólo ellos conformaran el sistema, este entonces cumpliría su tarea bien, sólo bien, y esta puntuación no es aceptable cuando este Proyecto, desde el primer instante, ha buscado ir siempre un paso más alla, garantizando no sólo el funcionamiento de las partes esenciales, sino también cada pequeño pero siempre importantísimo detalle, pues es en los detalles cuando algo deja de ser sólo bueno para sobresalir frente al resto.

```
case "EscuelaPlantaUno":

  switch (SDN.dondeDeboEstar)
  {
    case "AulaUno":
    case "AulaDos":
    case "AulaTres":
    case "AulaCuatro":
      SDN.balizaActiva = "H_BalizaEscuelaEscaleraNorteRellano";
      SDN.localizadorTipo = "bajar";
      break;

    case "TecnologicoDespachoFernandoJFF":
    case "Secretaria":
    case "SalonDeActos":
    case "AulaHacheTres":
    case "TecnologicoBanoChicasPlantaBaja":
    case "TecnologicoPlantaBaja":
    case "EscuelaPlantaBaja":
      SDN.balizaActiva = "H_BalizaEscuelaEscaleraSurRellano";
      SDN.localizadorTipo = "bajar";
      break;

    case "BibliotecaPlantaBaja":
      SDN.balizaActiva = "H_BalizaBiblioteca";
      SDN.localizadorTipo = "normal";
      break;

    case "AulaDoscientoscuatro":
      SDN.balizaActiva = "H_BalizaAulaDoscientoscuatro";
      SDN.localizadorTipo = "normal";
      break;

    case "AulaDoscientosdieciseis":
      SDN.balizaActiva = "H_BalizaAulaAulaDoscientosdieciseis";
      SDN.localizadorTipo = "normal";
      break;


    // default:
  }

  break;
```

Cuadro 14.3: Fragmento de código del guión LocalizadorDeBalizas.js

Figura 14.7: Localizadores

La atención a los detalles se demuestra, por ejemplo, es la utilización de tres tipos de localizadores diferentes (Figura 14.7), uno para cada uno de los tres escenarios posibles: que el avatar deba subir, bajar o permanecer en el nivel en el que se encuentra al final del tramo actual de su ruta hacia el destino. Sin esta distinción adicional, el usuario no sabría con claridad qué hacer ante las escaleras dobles del edificio tecnológico, vería un localizador indicándole una dirección, pero no sabría si debe subir o bajar, tendría que probar. No así sin embargo con tres localizadores diferentes que definen siempre claramente hacia dónde hay que ir, no sólo a lo largo, también a lo alto.

Otro ejemplo del cuidado a los detalles es la propuesta de una u otra salida cuando varias son posibles en función de la cercanía del avatar a cada una. La alternativa sencilla a este escenario consitiría es asignar una unica salida para todos los destinos posibles a los que se pudiera acceder desde un mismo lugar. De recurrir al camino fácil, se estaría obligando a usuario, por ejemplo, a dar un rodeo para salir de un recinto por la puerta principal cuando está al lado de una puerta secundaría que le permitiría llegar mucho antes al destino elegido. En este Proyecto se ha rechazado esa vía fácil; de esta forma, cuando el avatar abandona el edificio tecnológico y el siguiente tramo es la Escuela de Ingenierías, no se le sugiere que entre por la entrada principal, ubicada al otro extremo de su posición como sería lo más sencillo, se le indica la entrada por el pasaje de unión, correspondiente al acceso lógico y natural en el mundo real.

Un ejemplo más, que demuestra que se ha buscado activamente la superación frente al conformismo, se demuestra en la gestión de la navegación guiada en el edificio tecnológico. Cada una de sus plantas se ha dividido en alas este y oeste, y en cada una de sus escaleras dobles se han añadido puntos de control adicionales intermedios para que, en este último caso, el usuario sepa siempre qué tramo ha de seguir sin tener que recurrir a la adivinación cuando llega sus rellanos entre plantas, el punto crítico donde los tramos de subida y bajada paralelos se combinan, como puede observarse en la Figura 14.8.

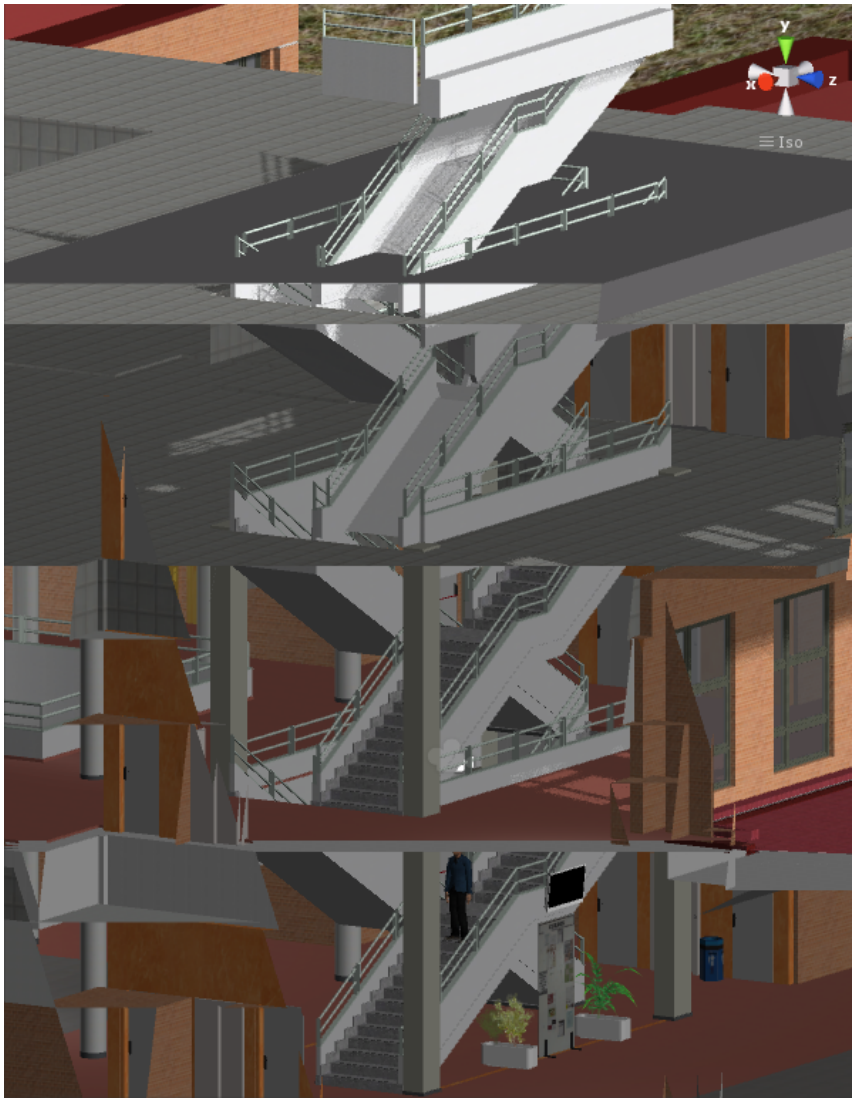
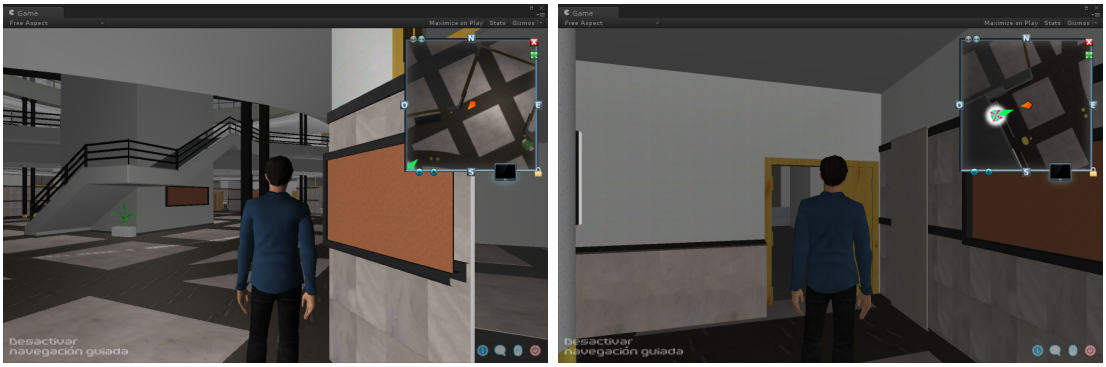


Figura 14.8: Maraña de escaleras



(a) El avatar está aún lejos del destino (b) El avatar ya casi ha llegado

Figura 14.9: Efecto de la distancia en la forma de los localizadores

La división en alas este y oeste posibilita, es este caso, indicarle a usuario cuál de los dos bloques de escaleras dobles es el que le permite llegar antes a su destino en función de su posición actual, actualizándose instantáneamente aun si el avatar decidiera alternar, por lo que fuera, entre un ala y otra.

Detalle es, también, que la particular forma con la que se han creado los localizadores está pensada de partida para que el extremo estrecho apunte continuamente al avatar mientras este se desplaza, y el extremo ancho permanezca oculto tras más allá del campo de visión del mapa, dando la sensación de que es una punta de flecha la que indica la dirección a seguir cuando el avatar aún está lejos, para convertirse en una suerte de diana al estar el avatar ya próximo a su destino (Figura 14.9).

Por último, el más pequeño de los detalles también se ha atendido, aquel por el que, al estar ubicados el indicador del avatar (el prefab H_IndicadorAvatar) y el localizador sobre el mismo haz horizontal y, por tanto, a la misma altura, uno de los dos dos elementos aparecería dibujado en el mapa que el usuario ve debajo del otro. Por lógica, el indicador no debería ocultarse nunca, y por eso se ha modificado internamente su cota para estar siempre unas milésimas por encima de cualquier localizador, suficiente para que resolver el más pequeño de los detalles que, como se dijo ya antes, por pequeño no deja de ser importante.

14.3. Persistente entre escenas

Una importante cualidad del Sistema de Navegación Guiada, sin la que lo dicho sólo tendría validez limitada al área actual y no más allá, es la persistencia entre escenas.

Utilizando el guión global SDN.js como respaldo permanente para almacenar la

(a) Mapa normal (b) Mapa maximizado

Figura 14.10: Ajuste automático de los localizadores al modo de mapa

información relativa a su estado interno que deba perdurar, y con la infraestructura adecuada proporcionada por el guión PersistenteControl.js, se logra que al saltar el avatar de una escena a otra, destruyendo por definición en el proceso todos los datos existentes en la escena anterior, el Sistema de Navegación Guiada se inice automáticamente en la nueva escena, sin necesidad de interacción por parte del usuario, y desde el primer instante con los valores correctos y actualizados de ubicación de balizas y localizadores para la escena recién cargada.

14.4. Preciso en todos los modos de mapa

Otra característica notable de este sistema, anticipada durante el desarrollo de su funcionamiento, es la validez de su preción en todos los modos de mapa, tanto en el modo normal como en el modo a pantalla completa, ni siquiera la minimización de este afecta a la continuidad de su exactitud.

El elemento clave de esta cualidad es el guión creador del cubo perimetral, CamaraMapaPerimetroControl.js, que ajusta inmediata y automáticamene sus dimensiones cuando el modo del mapa cambia (Figura 14.10), logrando de esta forma que los localizadores se reajusten de manera acorde a las dimensiones de este.

14.5. Interfaz

El usuario puede interactuar con el Sistema de Navegación Guiada para realizar dos funciones, activarlo y desactivarlo.

La primera de ellas se efectúa desde el menú de destinos, explicado ya en el Capítulo 13, «Sistema de Transporte», y ejemplificado en la Figura 13.4. Pulsando sobre la pequeña brújula al lado del nombre de los destinos.

La segunda función se ejecuta pulsando sobre el texto que aparece en la esquina inferior izquierda de la pantalla, en cualquiera de los modos del mapa, cerrado, normal o maximizado, siempre que la navegación guiada está activada.

En resumen:

 Activar el Sistema de Navegación Guiada hacia el destino junto al que aparece este botón.

Desactivar el Sistema de Navegación Guiada (sólo visible cuando está activo).

14.6. Configuración y personalización

Según lo expuesto, la personalización de este sistema es por tanto equivalente a la personalización del menú de destinos del Sistema de Transporte, incluyendo en este caso además, tal y como se muestra en la Figura 14.11, los iconos y el texto con las instrucciones para el uso del presente sistema que este menú muestra informativamente en su sección inferior, y el botón junto a los nombres de destino que activa la navegación guiada hacia ellos.

También puede personalizarse la apariencia y dimensiones del texto de desactivación, gestionado por el prefab H_InterfazMapa, modificando los estilos denominados «Estilo Cancelar Navegacion Guiada» tanto en el componente «Interfaz Mapa Control» como en el componente «Interfaz Mapa Grande Control» (Figura 14.12).

14.7. Integración con el Sistema de Localización

Como se ha visto al describir el funcionamiento del sistema protagonista de este capítulo, su vinculación con el Sistema de Localización es muy alta, pues es este el encargado de gestionar la variable global `SDN.dondeEstoy`, pivote del guión LocalizadorDeBalizas.js y por tanto indispensable para el Sistema de Navegación.

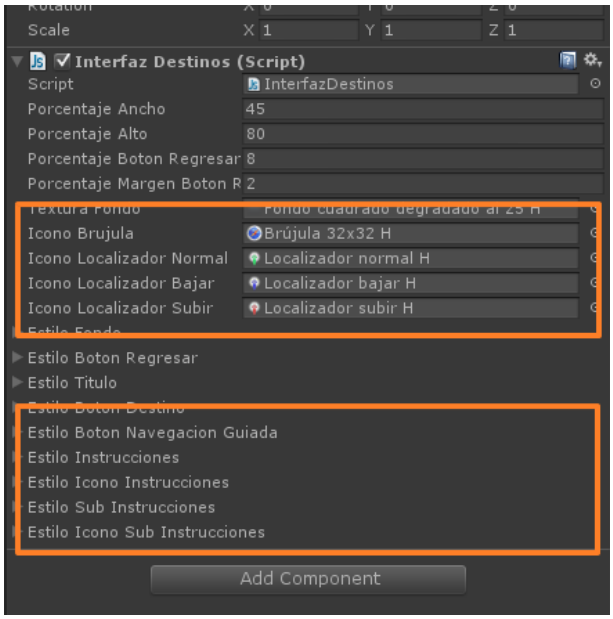


Figura 14.11: Personalización del menú de destinos para el Sistema de Navegación Guiada

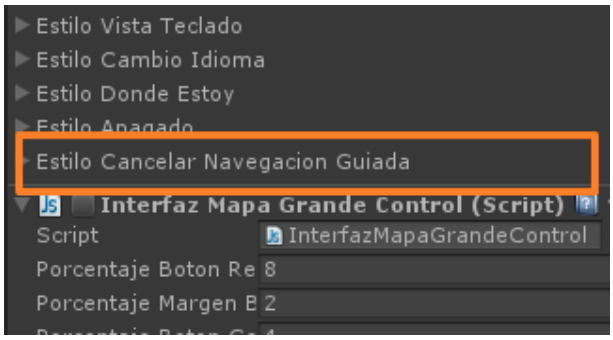


Figura 14.12: Personalización del botón de desactivación de la navegación guiada

Figura 14.13: Navegación guiada completada

14.8. Integración con el Subsistema de Información por Pantalla

El nexo entre estos dos sistemas queda sellado en el instante en que el usuario completa una navegación guiada, momento en el que esto se le notifica al prefab H_Persistente para que active los presentadores de texto con el objeto de poder informar al usuario de que acaba de llegar a su destino. Puede verse el mensaje utilizado para este fin en la Figura 14.13.

14.9. Integración con el Sistema de Traducción

Este sistema es el encargado de traducir todos aquellos mensajes con los que el de navegación se comunica con el usuario. A saber, el mensaje de navegación completada al llegar al destino; el texto del botón de desactivación de la navegación guiada; y todo el menú de destinos, en particular las instrucciones de uso del sistema que este menú incluye debajo del listado de ubicaciones.

14.10. Integración con el Sistema de Transporte

El nexo entre estos dos sistemas es pequeño, pero implica la resolución de un escenario que de no tenerse en cuenta podría producir resultados visualmente exóticos, razón por la que merece la pena documentarlo.

Este escenario es aquel en el que, estando activada la navegación guiada hacia un determinado destino, el usuario activara el Sistema de Transporte en cualquier otro punto para transportarse inmediatamente a ese destino en particular.

De no prever esta eventualidad, los presentadores de texto activarían al mismo tiempo, tanto el mensaje de navegación guiada completada por haber llegado ya al destino, como el mensaje con el nombre de la ubicación que el Sistema de Transporte genera por la misma razón.

Este, no obstante, es ahora tan solo un escenario teórico, pues se ha previsto con antelación y se le ha aplicado solución. Así, cuando estas circunstancias se cumplan, el Sistema de Tranporte mostrará el nombre del lugar de destino y, una vez cumplido su tiempo, el Sistema de Navegación Guiada mostrará entonces su mensaje.

Juntos, pero nunca revueltos.

Capítulo 15

Sistema de Vista Alternativa

La forma que el usuario tiene de cambiar en el proyecto legado el punto de vista desde el que observa el mundo virtual que rodea a su avatar es mediante el ratón. El movimiento del avatar y los de la cámara que representa este punto de vista son independientes, el uno se maneja con el teclado y el otro con el ratón. Este método, si bien muy práctico para apreciar el entorno cuando el avatar está parado, tiene, no obstante, dos inconvenientes: la necesidad de reajustar continua y manualmente el punto de vista cuando el avatar se está moviendo, y que al desplazar el ratón para manipular los diferentes botones en pantalla la cámara se mueve forzosamente.

En la Figura 15.1 puede observarse cómo el avatar está avanzando pero, por no haber reajustado el punto de vista manualmente no se puede ver hacia dónde.

Para resolver ese inconveniente nace este sistema, ofreciendo al usuario un modo

Figura 15.1: El avatar avanza pero si saber hacia dónde

Figura 15.2: El modo alternativo de vista muestra siempre lo que hay delante

alternativo de vista que, a diferencia del legado, se orienta continua y automáticamente hacia donde el avatar esté mirando, sin necesidad de usar el raton, facilitando y simplificando enormemente su manejo (Figura 15.2).

15.1. Componentes

Guiones

- CamaraMapaControl.js
- VistaTeclado.js
- PersistenteControl.js
- SDN.js

Prefabs

- H_CamaraMapaPerimetro
- H_InterfazMapa
- H_Persistente

Objetos

- La cámara del avatar, asociada a él en la escena bajo el nombre «*Camera*».

Recursos

- Texturas de los botones de activación organizadas en la ventana de proyecto dentro de la carpeta «Proyecto H/Interfaz/InterfazCambioVista».

15.2. Funcionamiento

Al cargar una escena, como parte de sus rutinas preparativas, el guión CamaraMapaControl.js carga en ella los prefab H_InterfazMapa y H_Persistente. Estos dos elementos son los que preparan la interfaz con la que el usuario podrá activar o desactivar el modo de vista alternativo y, por tanto, el punto de partida de este sistema. Así, el primero de ellos controla el botón que el usuario puede pulsar para conmutar el modo de vista actual, y el segundo permanece atento al teclado, detectando la pulsación de la tecla asignada a esa misma función.

Cuando se detecta cualquiera de estos dos eventos estando activo el modo de vista legado u orbital, se generan, simplificadamente, las siguientes instrucciones:

- Se desactiva el guión controlador original de la cámara del avatar, MouseOrbit.js.
- Se le añade el guión controlador que este sistema proporciona, VistaTeclado.js, si no se le ha añadido anteriormente ya.
- El nuevo guión toma el control de la cámara y la posiciona automáticamente detrás del avatar.

Cuando el modo de vista activo sea el alternativo, entonces las instrucciones generadas son:

- Se desactiva el guión controlador alternativo de la cámara, VistaTeclado.js.
- Se activa el guión controlador original, MouseOrbit.js.

La tarea del guión VistaTeclado.js, como pieza central de este sistema, es la de posicionar la cámara en el lugar adecuado tras el avatar y la de controlar sus movimientos cuando este se mueve. Entran aquí en juego dos sistemas de referencia de coordenadas, el local del avatar para posionar la cámara adecuadamente detrás de él, y el global del mundo virtual para mover la cámara tras su posicionamiento inicial; ambos gestionados con precisión en el corazón de este guión, e integrados en su siguiente función: suavizar los movimientos.

15.2.1. Suavizado del movimiento de la cámara

Esta es una tarea importante pues, visualmente, determina que las imágenes que se suceden en la pantalla del usuario cuando mueve al avatar lo hagan a trompicones o de forma agradable y completamente fluida.

En el Cuadro 15.3 se puede observar parte del código encargado se suavizar el movimiento de la cámara. La idea clave consiste en relacionar la posición actual de la cámara con la posición que tendrá un instante después. La primera la determina la propia cámara, la seguna la determina el avatar al moverse pues es detrás de este donde siempre debe estar. Conocidos estos valores, de la relación entre ellos se encargará la función `Mathf.LerpAngle()`, interpolándolos y calculando las posiciones intermedias que otorgan el movimiento amortiguado a la cámara. El resto es diferenciar y gestionar adecuadamente por separado posiciones y rotaciones, pues la cámara, además de estar ubicada en un lugar concreto, también tiene en él una orientación determinada, por lo que grados y coordenadas habrán de tenerse en cuenta.

15.3. Persistente entre escenas

De no hacer al sistema persistente entre escenas, cuando el avatar saliera de una para entrar en otra y se destruyeran todos los elementos de la primera, al cargar la segunda el modo de vista volvería a ser el legado, como modo predeterminado, aun cuando el usuario hubiera activado anteriormente el modo alternativo.

Para lograr esta persistencia el sistema mantiene siempre actualizada una variable estática en el guión SDN.js, en la que asigna el modo actual de vista cuando el usuario cambia entre ellas. Así, una vez se haya cargado una nueva escena, se puede consultar el estado de esa variable y determinar a partir de ella si se ha de habilitar automáticamente, por ser el que el usuario tenía activado antes, el modo de vista alternativa.

```
1  // Definimos las rotaciones destino y actual ("quaternion"), así como la rotación
       específica
2  // sobre el eje Y ("float"), para suavizar no sólo la rotación, sino también la posición
3  // de la cámara.
4
5  rotacionActualTotal = SDN.camaraPrincipal.transform.rotation;
6  rotacionDestinoTotal = SDN.avatar.transform.transform.rotation;
7
8  anguloActualEjeY = SDN.camaraPrincipal.transform.eulerAngles.y;
9  anguloDestinoEjeY = SDN.avatar.transform.eulerAngles.y;
10
11 // Suavizamos la rotación específica sobre el eje Y.
12
13 anguloSuavizadoEjeY = Mathf.LerpAngle(anguloActualEjeY, anguloDestinoEjeY, Time.deltaTime
       * suavizado);
14
15 // Transformamos el ángulo ("float") en rotación ("quaternion").
16
17 rotacionSuavizadaEjeY = Quaternion.Euler(0, anguloSuavizadoEjeY, 0);
18
19 // Suavizamos la posición de la cámara teniendo en cuenta su rotación.
20
21 transform.position = SDN.avatar.transform.position;
22 transform.position -= rotacionSuavizadaEjeY * Vector3.forward * distancia;
23 transform.position.y = SDN.avatar.transform.position.y + altura;
24
25 // Suavizamos la rotación de la cámara.
26
27 transform.rotation = Quaternion.Lerp(rotacionActualTotal, rotacionDestinoTotal, Time.
       deltaTime * suavizado);
```

Figura 15.3: Fragmento de código encargado de amortiguar la cámara

15.4. Interfaz

Como ya se adelantó, el usuario puede activar el Sistema de Vista Alternativa de dos formas, mediante el ratón y mediante el teclado. Agrupadas estas formas quedarian:

15.4.1. Botones

Estos botones no aparecen en el modo de pantalla completa, ya que siendo la vista siempre cenital en él, no tienen ahí razón de ser.

 Cambiar al modo de vista alternativa (sólo visible en el modo de vista legado)

 Cambiar al modo de vista legado (sólo visible en el modo de vista alternativa)

Figura 15.4: Configuración del Sistema de Vista Alternativa

15.4.2. Teclado

Conmutar entre los dos modos de vista Tecla predeterminada: **K**.

15.5. Configuración y personalización

Pueden configurarse tanto la posición relativa de la cámara respecto del avatar en la que se coloca detrás de él, como la cantidad de suavizado que tendrán sus movimientos; y puede personalizarse el valor de la tecla asignada para conmutar entre los dos modos de vista con el teclado. Todos estos valores modificables se encuentran cómodamente centralizados en el prefab central H_CamaraMapa tal y como puede verse en la Figura 15.4.

En es misma figura puede observarse una casilla adicional, «Permitir Vista Teclado», que permite al desarrollador, desactivándola, impedir que el usuario pueda cambiar de modo de vista anulando individualmente este sistema.

En resumen, las opciones configurables son:

Permitir Vista Teclado Desactiva el Sistema de Vista Alternativa.

Tecla Conmutadora Vista Designa la tecla que tendrá la función de conmutar entre los modos orbital y alternativo.

Suavizado Vista Teclado Determina la cantidad de amortiguamiento que tendrá el movimiento de la cámara al seguir los del avatar.

Altura Vista Teclado Determina la altura a la que se posicionará la cámara detrás del avatar.

Distancia Vista Teclado Determina la distancia a la que se ubicará la cámara detrás del

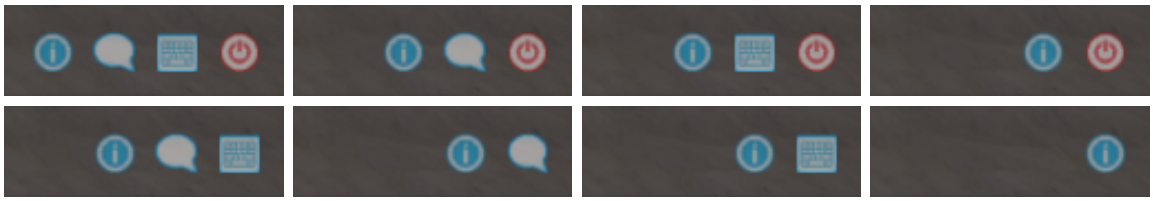


Figura 15.5: Los botones permanecen siempre organizados

avatar.

15.6. Integración con los sistemas de Localización, Traducción y Apagado

La relación entre estos cuatro sistemas es estética. Se fundamenta en la posibilidad que tiene el desarrollador de desactivar individualmente, como se ha visto ya en la sección anterior para este sistema en concreto, cualquiera de ellos con excepción del de Localización.

Cuando alguno de estos sistemas se desactiva, su icono, que en principio está ubicado junto a los de los demás, en una misma fila en la esquina inferior derecha de la pantalla, desaparece.

Para evitar que quede un vacío en mitad de la fila de iconos, se ha incluido un código en el guión InterfazMapaControl.js que elimina el hueco desplazando hacia la derecha los iconos de los sistemas que sí estén activos. En la Figura 15.5 pueden comprobarse las posibilidades que pueden darse.

Capítulo 16

Sistema de Traducción

De nada sirve un mensaje si no puede ser comprendido, para que todo el mundo lo entienda antes debe de ser traducido.

El mundo es grande y la imparable globalización difumina las barreras fronterizas, personas de todas las partes del globo están ahora más cerca que nunca, aun así, puede que las siga separando una última barrera: la lingüística.

Este Proyecto es consciente de la importancia que tienen los idiomas en el mundo actual, por esta razón se ha diseñado el Sistema de Traducción, encargado de traducir los textos y mensajes con los que el resto de sistemas se comunican con el usuario (Figuras 16.1, 16.2 y 16.1).

Actualmente son tres los idiomas instalados, inglés americano, alemán y español, y pueden ser fácilmente ampliados en el futuro.

16.1. Componentes

16.1.1. Guiones

- CamaraMapaControl.js
- Traductor.js
- InterfazIdiomasControl.js

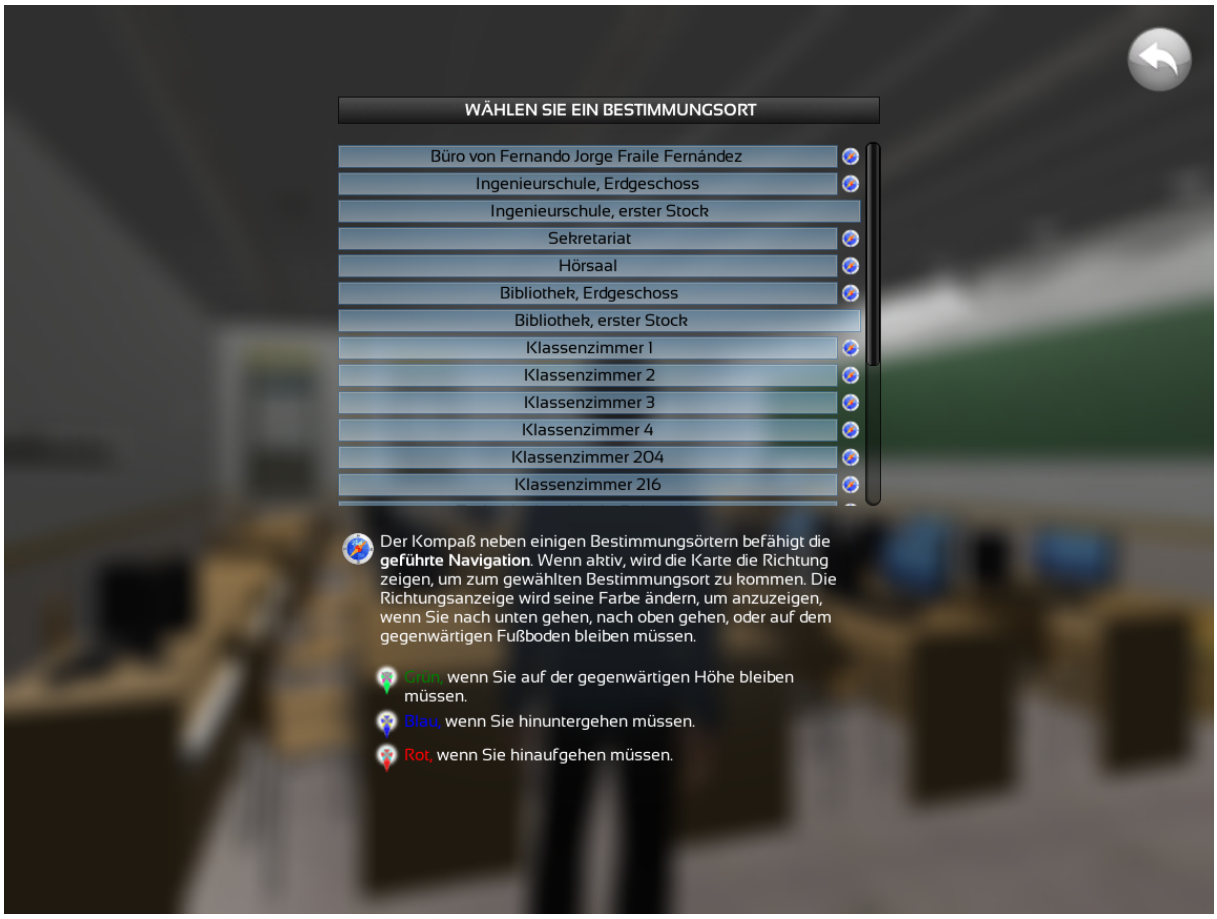


Figura 16.1: Menú de destinos en alemán

Figura 16.2: Ubicación en inglés

- InterfazMapaControl.js
- SDN.js

16.1.2. Prefabs

- H_CamaraMapa
- H_InterfazIdiomas
- H_InterfazMapa
- H_Persistente

16.1.3. Recursos

- Texturas de la interfaz organizadas en la ventana de proyecto dentro de las carpetas «Proyecto H/Interfaz/General» y «Proyecto H/Interfaz/InterfazIdiomas».

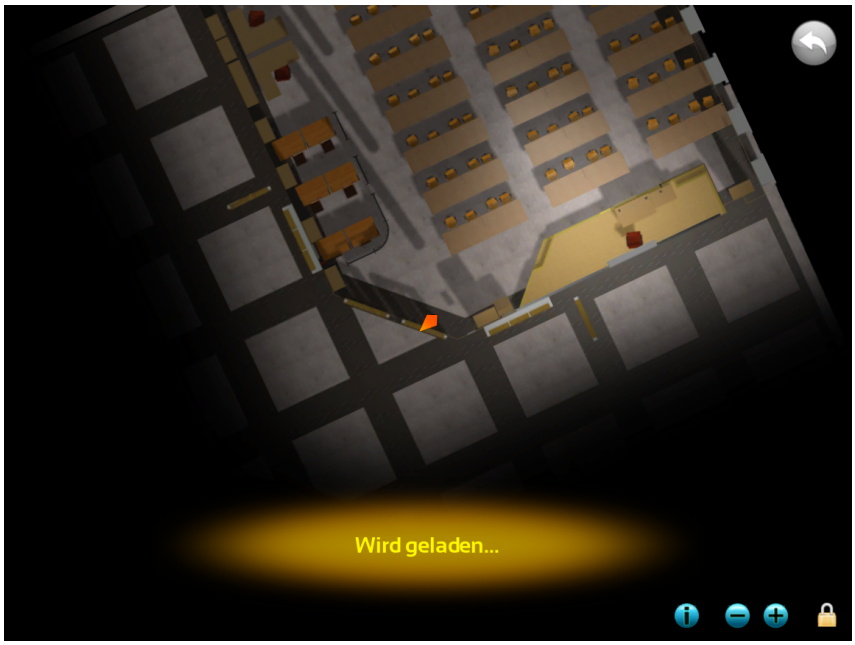


Figura 16.3: Mensaje informativo «Cargando...» en alemán

16.2. Funcionamiento

Es sencillo, se basa en la idea de recibir un texto, compararlo con una base de datos y, si se ha encontrado coincidencia, devolver el texto traducido.

Este proceso se realiza mediante una sencilla función, `Traductor.Traducir()`, que este sistema pone a disposición de todos los demás.

Cualquier texto puede ser filtrado por esa función, incluso si no se ha previsto una traducción para él, pues la función ha sido programada para devolver el texto original cuando no halle una coincidencia con su base de datos interna.

La detección del idioma al que realizar la traducción se logra mediante la lectura de una variable global almacenada en el guión SDN.js. Esta queda definida al cargar la aplicación, a partir del lenguaje predeterminado fácilmente configurable, y en el momento en el que el usuario decide cambiar de idioma durante la ejecución de la misma, utilizando para ello el menú que la interfaz de este sistema, H_InterfazIdiomas, cargada en la escena por el guión CamaraMapaControl.js, pone a su disposición.

El guión SDN.js aporta también las funciones globales que el sistema utiliza para pausar y reanudar la aplicación al activar el usuario el menú de idiomas.

En el lado del usuario, este puede interactuar con el sistema mediante el menú

```
case "TEXTO A TRADUCIR":
  switch (idioma)
  {
    case "ingles":
      textoTraducido = "TEXTO TRADUCIDO AL INGLÉS";
      break;
    case "aleman":
      textoTraducido = "TEXTO TRADUCIDO AL ALEMÁN";
      break;
    default:
      textoTraducido = "TEXTO TRADUCIDO AL ESPAÑOL";
      break;
  }

  break;
```

Cuadro 16.1: Plantilla para añadir nuevas traducciones

de idiomas, gestionado por el prefab H_InterfazIdiomas y clonado en la escena como parte de los preparativos iniciales llevados a cabo por el prefab central H_CamaraMapa.

16.3. Ampliación de la base de datos y uso

Pueden añadirse nuevas traducciones a la base de datos con gran sencillez. La propia base de datos incluida en el guión Traductor.js, proporciona una plantilla (Cuadro 16.1) y las instrucciones necesarias para este fin.

En ella se sustituirán los textos escritos en mayúsculas y la propia base de datos determinará apropiadamente qué idioma utilizar.

El texto a traducir no tiene por qué ser necesariamente comprensible, puede ser de hecho cualquier cadena de caracteres, pues se usará sólo para determinar la coincidencia y devolver el texto asociado correspondiente al idioma en uso. Esto permite, por ejemplo, traducir textos largos muy cómodamente, tal y como puede ser un párrafo entero de instrucciones, sin tener que escribir en todos los guiones que soliciten la traducción el texto entero original cada vez.

En el Cuadro 16.2 se puede ver, ya en la práctica, un ejemplo de lo dicho en el párrafo anterior. Recogido directamente del guión que crea el menú de destinos, este fragmento de código presenta en pantalla las instrucciones que este menú incluye. Maqueta su aspecto mediante dos instrucciones de tipo *GUI*, y las traduce al idioma en uso mediante la función `Traductor.Traducir()`. Puede comprobarse cómo se utiliza una palabra corta y descriptiva como texto de entrada. El Sistema de Traducción cotejará esa sencilla palabra con su base de datos y devolverá, ahora sí, todo el texto íntegro, en cualquiera de los idiomas activados.

```
1 GUILayout.Label(Traductor.Traducir("InstruccionesSelectorDestino"), estiloInstrucciones);
```

Cuadro 16.2: Uso práctico del traductor

16.4. Interfaz

El usuario puede activar en cualquier momento el menú selector de idiomas, pulsando sobre el botón que aparece en la parte inferior derecha de la pantalla para este fin. También puede activarlo pulsando en el teclado en cualquier momento la tecla configurada en el sistema para ello.

16.4.1. Botones

 Activa el menú selector de idiomas (no disponible en el mapa maximizado).

 Con el menú de idiomas activo, permite cancelarlo y regresar a la aplicación.

16.4.2. Teclado

Activar el menú selector de idiomas desde cualquier modo Tecla predeterminada: **L**.

16.4.3. Menú de idiomas

El aspecto visual de este menú se define mediante el guión InterfazIdiomasControl.js. El diseño (Figura 16.4), la apariencia y distribución de sus elementos se ha cuidado meticulosamente.

Las banderas[1] que sirven para cambiar de idioma poseen cuatro estados (Figura 16.5):

Activo Cuando el usuario pasa el ratón por encima.

[1] Se han modificado a partir de los iconos del paquete *«European Union Flags»* de Dooffy Design[3] http://www.dooffy.com, con licencia *Freeware*, tomándolos como base para crear los diferentes estados de las banderas.

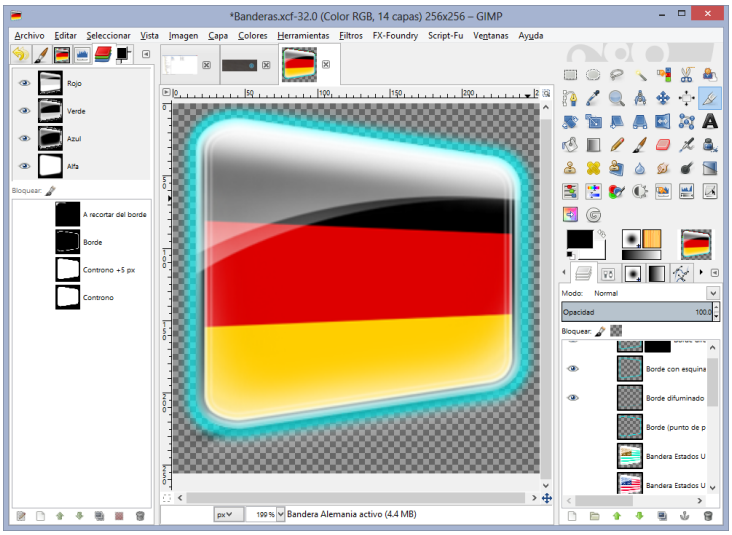


Figura 16.4: Diseño de las banderas

Figura 16.5: Estados de las banderas

Pulsado Cuando el usuario pulsa sobre el botón.

Elegido Cuando se corresponde con el idioma en uso actual.

Normal Cuando no se dan los casos anteriores.

Como toque final, se dibuja una textura con gradiente debajo de las banderas y se difumina la escena, Figura 16.6.

16.5. Autoajustable a los cambios de resolución

Tanto los elementos del menú de idiomas como el botón general de activación modifican automáticamente sus dimensiones para adaptarse inteligentemente a los cambios de resolución.

Esto es posible gracias al código incluido en los guiones InterfazIdiomasControl.js e InterfazMapaControl.js que, de forma similar a lo planteado en la Sección 9.7, vigilan continuamente los cambios de resolución que se produzcan.

Figura 16.6: Menú de idiomas

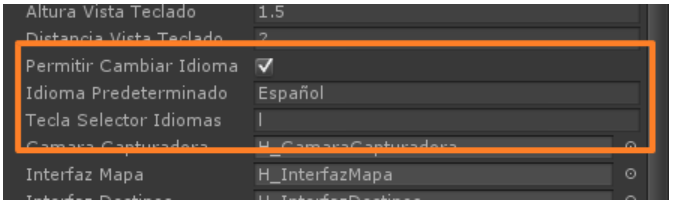


Figura 16.7: Configuración del Sistema de Traducción

16.6. Configuración y personalización

Desde el Inspector del prefab H_CamaraMapa pueden configurarse varios aspectos relacionados con el comportamiento del sistema como se observa en la Figura 16.7. La función de cada campo es la siguiente:

Permitir Cambiar Idioma Desactivada anula la posibilidad de que el usuario cambie el idioma, utilizando como idioma para la aplicación aquel definido por el desarrollador como predeterminado.

Idioma Predeterminado Indica el idioma predeterminado en el que cargará la aplicación inicialmente, puede ser inglés, alemán o español (con o sin acentos, con o sin mayúscula inicial y con o sin eñe)

Tecla Selector Idiomas Define la tecla con la que poder activar el menú de idiomas usando el teclado.

También puede personalizarse el aspecto de los elementos del menú de idiomas en el Inspector de H_InterfazIdiomas. (Figura 16.6)

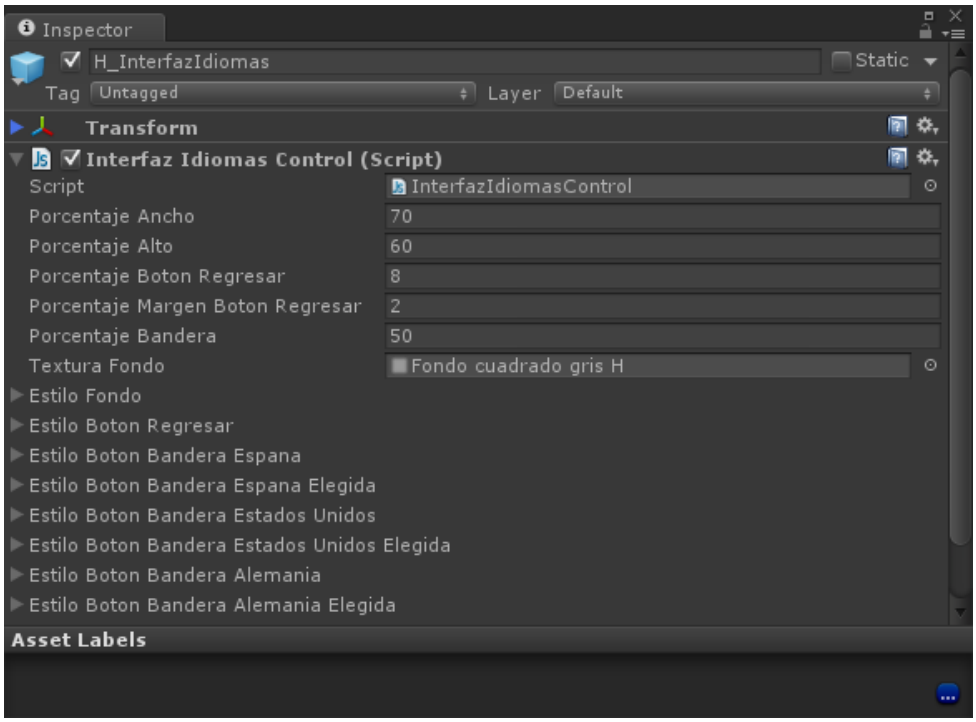


Figura 16.8: Personalización del menú de idiomas

Porcentaje Ancho Determina porcentualmente el tamaño a lo ancho del menú.

Porcentaje Alto Determina porcentualmente el tamaño a lo alto del menú.

Porcentaje Boton Regresar Valor porcentual respecto de la altura de la pantalla que determina el tamaño del botón para abandonar el menú sin elegir un destino.

Porcentaje Margen Boton Regresar Valor porcentual respecto de la altura de la pantalla que determina la distancia de separación entre el citado botón y el borde de la pantalla.

Porcentaje Bandera Valor porcentual respecto de la altura de la pantalla que determina las dimensiones de las banderas.

Textura Fondo Define una textura que se colocará bajo los elementos del menú.

Estilos Permiten modificar los elementos gráficos de los botones a los que hace referencia su nombre. Los diferentes estados de las banderas, normal, activo, pulsado o elegido, se modifican aquí.

16.7. Integración con los sistemas de Localización, Vista Alternativa y Apagado

Tal y como se vio en la Sección 15.6, la relación entre estos cuatro sistemas es estética, dependiente de la la posibilidad que tiene el desarrollador de desactivar individualmente algunos de ellos. Cuando esto ocurre los iconos de activación de estos sistemas desaparecen, dejando un hueco en el lugar en el que antes estaba ubicando junto a los demás.

Para evitar que quede un vacío en mitad de la fila de iconos, el guión InterfazMapaControl.js incluye un código de control que elimina el hueco desplazando hacia la derecha los iconos de los sistemas que sí estén activos.

Capítulo 17

Subsistema de Apagado

Todo lo que tiene un principio, tiene un final.

El Subsistema de Apagado encarna este concepto filosófico. Todo este Proyecto ha buscado en su esencia hacerle al usuario su estancia en el mundo virtual lo más agradable posible; ha puesto a su disposición un numeroso conjunto de herramientas para que siempre pudiera encontrar su camino. Ahora bien, por muy cómoda que se le procure la estancia, el usuario en algún momento abandonará el mundo virtual, y ahí estará el Subsistema de Apagado, para ponérselo, incluso al final, fácil.

17.1. Componentes

17.1.1. Guiones

- CamaraMapaControl.js
- InterfazApagadoControl.js
- InterfazMapaControl.js
- SDN.js

17.1.2. Prefabs

- H_CamaraMapa
- H_InterfazApagado
- H_InterfazMapa

17.1.3. Recursos

- Texturas de la interfaz organizadas en la ventana de proyecto dentro de las carpetas «Proyecto H/Interfaz/General» y «Proyecto H/Interfaz/InterfazApagado».

17.2. Funcionamiento

La función última del Subsistema de Apagado es la de mostrarle al usuario el «gran botón rojo[4]» (Figura 17.1), ser el último elemento de la aplicación con el que el usuario interactúe. Hasta llegar a ese punto, el proceso que tiene lugar es el que sigue.

En primer lugar, el guión central, CamaraMapaControl.js, clona en la escena, nada más cargar esta, al prefab encargado de presentar el menú de apagado, H_InterfazApagado, que permanecerá inactivo hasta que el usuario dé instrucción de activarlo, como se verá en la sección «Interfaz».

Figura 17.1: «Gran botón rojo»

Una vez activo, el menú de apagado le muestra el gran botón rojo, que tras pulsarlo, cerrará definitivamente la aplicación.

De manera secundaria, el guión SDN.js proporciona las funciones de pausado, reinicio o desenfoque del fondo que

el menú de apagado utilizará. También participa el guión InterfazMapaControl.js, encargándose de dibujar el botón que activará el menú de apagado.

17.3. Restricción a las plataformas *Standalone*

Para encabezar esta sección, se debe recordar que una aplicación desarrollada en Unity puede compilarse, como paso final, para numerosas plataformas diferentes, ordenadores, móviles, consolas, etcétera.

Un gran botón rojo de apagado cobra sentido en un tipo particular de plataformas, las *Standalone*, forma con la Unity designa a Windows, Mac OS X y Linux. En otras, como podría ser un navegador web, donde la forma natural de salir de la aplicación es cerrando la ventana en la que se ejecuta o directamente el propio navegador, no tiene tanto sentido.

Por esta razón, el código del guión InterfazMapaControl.js incluye un bloque específico de instrucciones para determinar la plataforma de compilación destino y, en caso de ser esta diferente a las *Standalone*, excluir de la aplicación final el código que presenta en pantalla el botón de activación del menú de apagado, anulando en la práctica a este sistema.

17.4. Interfaz

El usuario puede interactuar con el Subsistema de Apagado utilizando los siguientes botones:

 Activar el menú de apagado (no disponible en el mapa maximizado).

 Con el menú de apagado activo, permite cancelarlo y regresar a la aplicación.

17.5. Menú de apagado

Este menú tiene una sola función, apagar la aplicación, y por ello su aspecto visual debe transmitir también esta idea: un botón rojo y un texto informativo (Figura 17.2).

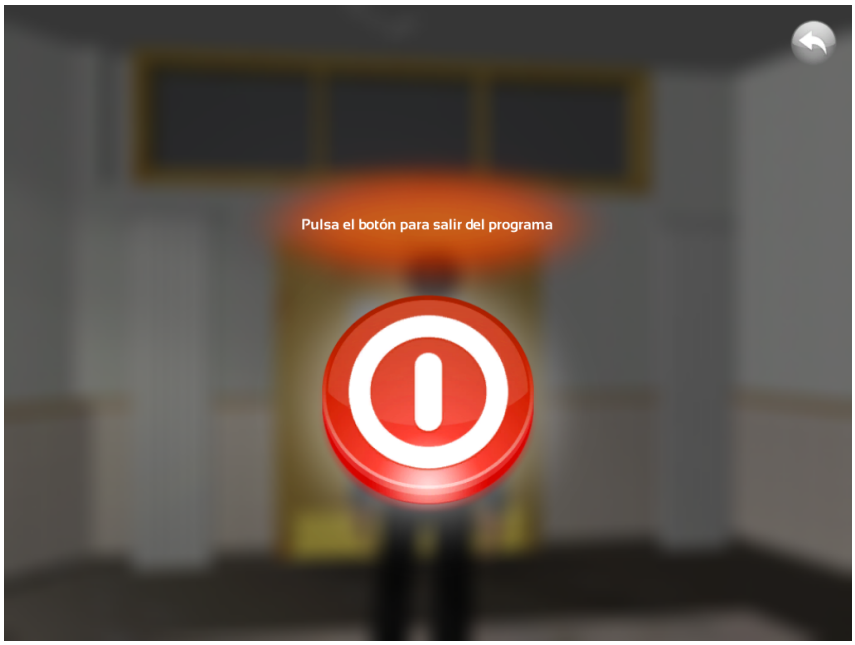


Figura 17.2: Menú de apagado

Se le da también al usuario la opción de regresar a la aplicación, mostrándole además un botón para cancelar el menú.

Capítulo 18

Relación de cambios y mejoras en el proyecto legado

A continuación se recogen y describen las modificaciones realizadas al proyecto legado. Manteniendo siempre su estructura y esencia, cada uno de los cambios se ha realizado para garantizar la estabilidad del conjunto en cualquier ampliación futura y mejorar la experiencia del usuario durante su paseo virtual.

18.1. Mejora del sistema de ubicación del avatar al cargar una escena

Problema

El sistema anterior efectuaba sus fases en ciclos de ejecución incompatibles con una programación más compleja, en consecuencia, elementos indispensables como la referencia al avatar o a la cámara principal del juego no estaban disponibles para el resto de guiones ni aquellos cálculos que dependen de esas referencias.

Descripción y solución

El guión del proyecto legado, SustContr.js[1], es el encargado de clonar y posicionar al avatar y a la cámara principal en cada nueva escena, y siendo estos los dos elementos

[1]Localizado en la carpeta «SelAv Scripts».

principales de cualquier aplicación lo convierte en la pieza angular de ese y de todo proyecto posterior.

Esta clonación, no obstante, la ejecutaba en el ciclo `Start()`, un ciclo demasiado tardío para este fin, lo que se traducía en que aquellos otros guiones, sí ajustados a los ciclos de ejecución, que necesitaran tener una referencia del avatar o de la cámara principal para funcionar, cuando trataban de localizarlos, se encontraran con que no había ni avatar ni cámara cargados aún en la escena. Metafórica, y posiblemente cómicamente también, se puede explicar como sigue: no se puede lanzar una pelota si no se tiene una pelota. Si un guión está programado para hacer este lanzamiento, antes de poder hacerlo va a extender el brazo y a buscar en la caja donde se ponen las pelotas una que poder lanzar, pero si el guión que está programado para poner ahí las pelotas justo antes de que empiece el partido se demora un solo instante, cuando el primero extienda el brazo no va a encontrar nada y, por consecuencia, va a tratar de lanzar algo que no existe.

Los cálculos que realizan los guiones del presente proyecto tienen una gran dependencia de estos dos elementos principales, cámara y avatar, y, si bien se trató en primera instancia de resolver este problema sin modificar el proyecto legado, programando con corrutinas a estos guiones para que volvieran a buscar en la caja si no hubieran encontrado la pelota a tiempo, incluso si el partido hubiera ya comenzado, finalmente se hizo necesaria una reescritura completa del guión SustContr.js.

Se podría pensar que bastaría con cambiar las instrucciones que clonaban al avatar y a la cámara de un ciclo a otro, con adelantarlas, moverlas desde la función `Start()` a la función `Awake()`, pero en la práctica hizo falta mucho más.

Cuando el guión SustContr.js clona a estos dos elementos, los posiciona en función de las coordenadas que le transmite un tercer jugador que ha estado observando el partido desde el banquillo hasta este momento. Continuando con la metáfora, la función de este tercer participante sería la de lanzar una pelota que ya tiene a un jugador que espera encontrar antes de empezar el siguiente partido. Si se adelantan las instrucciones de clonación de SustContr.js y sólo se hace eso, el efecto secundario es que el tercer jugador y él se van a descoordinar y la pelota no va a llegar a su destino.

La solución definitiva, pues, fue no sólo reorganizar las instrucciones de SustContr.js, también la reescritura de ese tercer jugador, y también de todos los demás como él, uno por cada puerta de entrada y salida que existe en el mundo virtual, que permanecen en el banquillo hasta que se les necesita.

La reescritura consistió en centralizar todas las coordenadas correspondientes a

todas las entradas y salidas, posición y rotación, en el propio guión SustContr.js, modificado consecuentemente, haciéndolo independiente del resto de «jugadores».

Adicionalmente, durante el proceso de centralización, además de dejar documentada la ubicación a la que se corresponden, se recalcularon manualmente todas las citadas coordenadas no limitándose a copiar las ya existentes, optimizando su precisión y, corrigiendo así de paso, el problema que se describe en la sección siguiente.

Cabe destacar que el Sistema de Transporte del presente proyecto, para reubicar al avatar entre escenas cuando el usuario activa un destino, añade código propio al guión SustContr.js y poder así realizar la clonación final en la posición final. El código nuevo incluido en este guión determina también, por tanto, si el cambio de escena se produce por haberse activado el Sistema de Transporte o porque el avatar haya cruzado una puerta corriente, y ejecuta consecuentemente un juego de instrucciones u otro.

En el Cuadro 18.1 se recoge el texto introductorio explicativo incorporado al guión `SustContr.js`. Este tipo de cabecera, añadida en todos los guiones que el presente proyecto ha creado o modificado, es un complemento de las abundantes anotaciones aclaratorias incluidas en el el código, por lo que se recomienda leer el propio guión (Sección F.54) si se desea más información a este respecto.

18.2. El avatar se clona por encima del suelo al cargar una escena

Problema

Al cargar una nueva escena y aparecer el avatar en ella, lo hace muy por encima del suelo, cayendo hasta él inmediatamente después (Figura 18.1).

Descripción y solución

Tal y como se adelantó en el apartado anterior, este problema es consecuencia de la inexactitud de las coordenadas que determinaban la posición inicial del avatar en una escena cuando este llegaba a ella desde un acceso natural de la anterior, una puerta o unas escaleras.

Para solucionarlo, además de recalcular con precisión todas estas coordenadas como se describió, se han reubicado también en cada una de las escenas todos los objetos

```
/*#################################################################

SustContr.js
------------

Guión altamente modificado del proyecto legado para el Proyecto H.
Encargado originalmente de sustuir el controlador estándar por el
controlador Mixano y reubicarlo en la escena, se le han añadido
ahora las funciones de transporte SDN del proyecto H al tiempo
que conserva armónicamente el sistema de transporte antiguo, mejorado
ahora éste para que todas sus variables se encuentren centralizadas
en este mismo guión, y reestructurado para que sus instrucciones
se ejecuten organizadamente en los ciclos de ejecución apropiados.

Las coordenadas de las que dependía el sistema de transporte antiguo
se han recalculado con precisión para evitar que el avatar aparezca por
encima del suelo como sucecía anteriormente, y se han añadido comentarios
a cada entrada para informar de la ubicación con la que se corresponden.

Asigna ahora también globalmente las variables correspondientes al avatar,
a la cámara principal, y a la cámara del mapa, de forma que estén
   accesibles
al resto de elementos del proyecto.

Se incluye al final la versión original del guión como referencia.

#################################################################*/
```

Cuadro 18.1: Cabecera explicativa añadida al guión `SustContr.js`

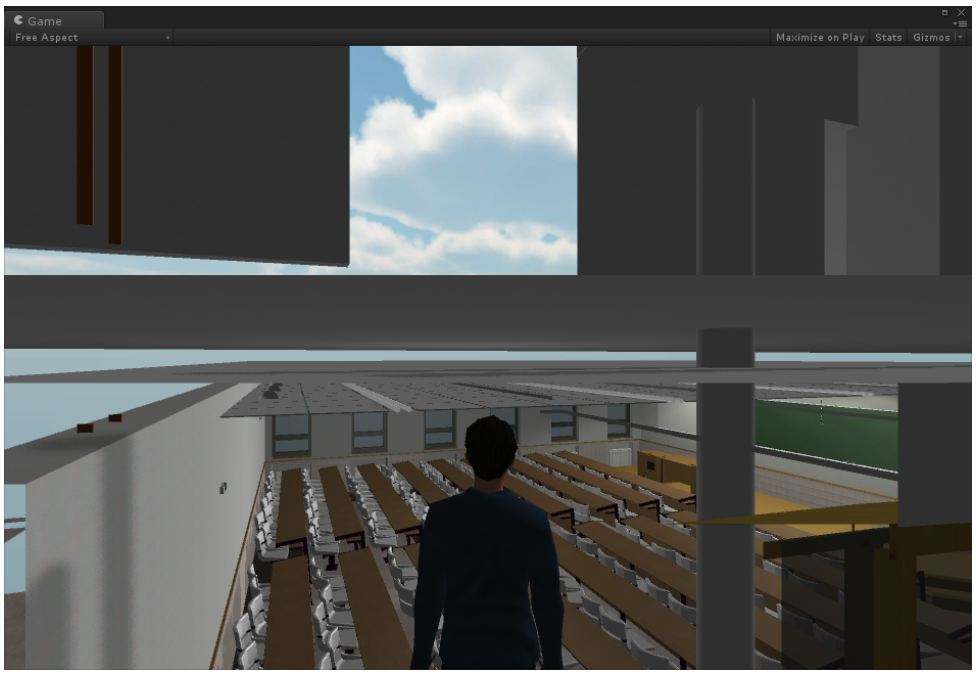


Figura 18.1: El avatar aparece en la escena muy por encima del suelo

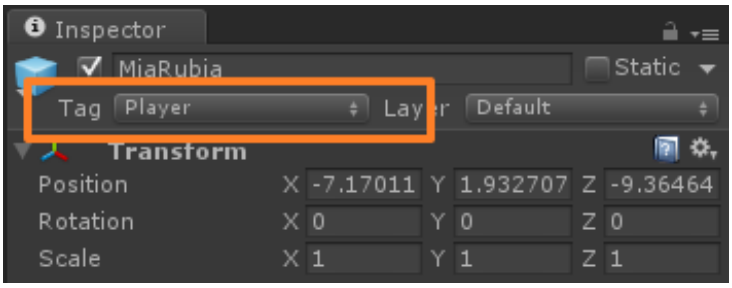


Figura 18.2: Etiqueta *Player* en el Inspector de «MiaRubia»

ControlProj, encargados de determinar la posición inicial del avatar cuando este no accede a ellas desde otra anterior. Esto ocurre, por ejemplo, en la escena correspondiente a los aparcamientos de la Escuela de Ingenierías, donde el avatar aparece directamente al cargar la aplicación.

18.3. Corrección de aspectos relacionados con el avatar «MiaRubia»

- Se ha modificado el valor del campo *Center* en su prefab para evitar que los pies del avatar aparecieran siempre por debajo del suelo (Figura 18.3).
- Se ha modificado el valor *Clipping Planes* de la cámara asociada al avatar para evitar que las paredes se interpusieran entre el avatar y la visión del usuario (Figura 18.4).
- Se le ha añadido la etiqueta «Player», sin la que cualquier código que necesitara interaccionar con el avatar no podría identificarlo (Figura 18.2).

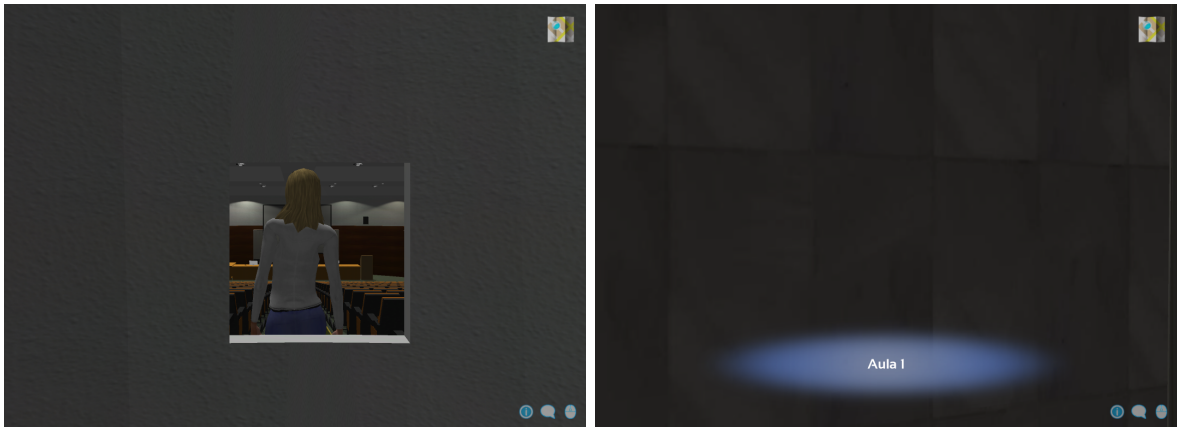
18.4. Solucionada la incorporeidad de numerosos elementos del mundo virtual

Problema

El avatar atravesaba elementos de su entorno que por su naturaleza deberían ser sólidos. Figuras 18.7.

(a) Antes (b) Después

Figura 18.3: Corregida la altura de «MiaRubia»



(a) Antes

(b) Después

Figura 18.4: Corregida la posición de la cámara de «MiaRubia»

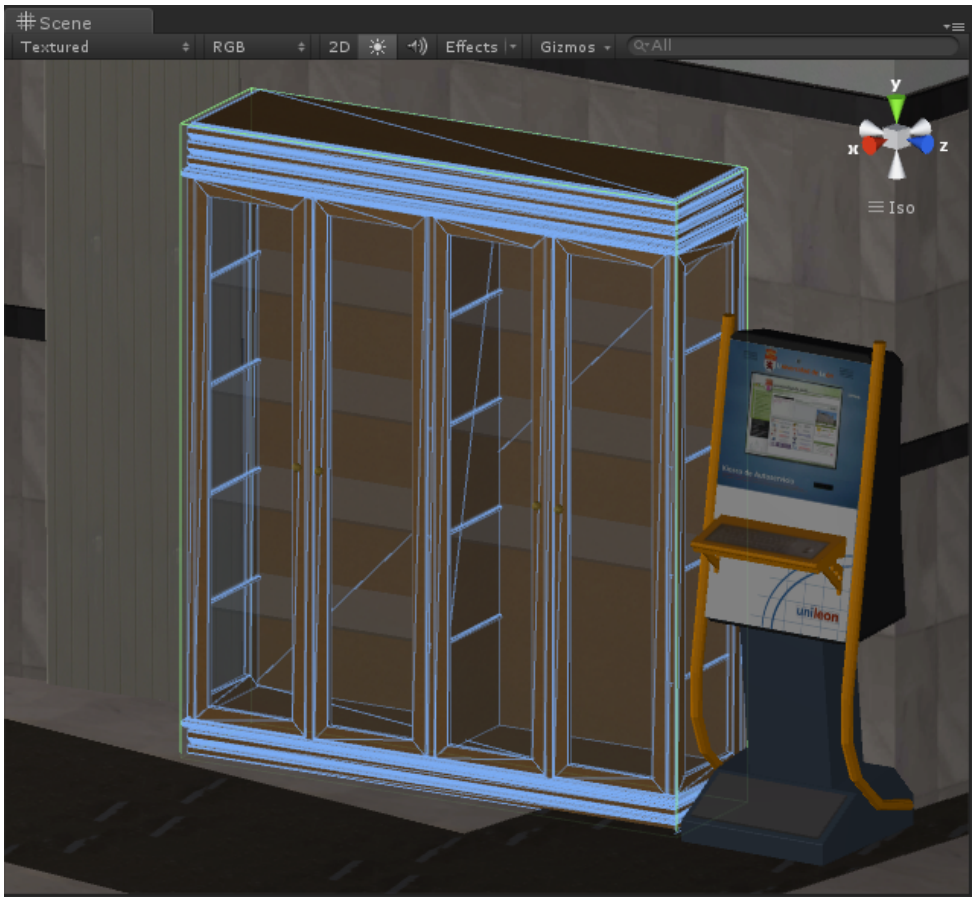


Figura 18.5: Sencillo *Box Collider* (contorno verde) alrededor de un mueble antes «incorpóreo»

Solución

Añadir un componente *Box Collider* (Figura 18.5) o *Mesh Collider* (Figura 18.6) a cada uno de estos elementos. El tipo de *Collider* añadido queda determinado por las características del elemento, otorgando preferencia al *Box Collider* cuando fuera posible para optimizar el rendimiento.

Listado de objetos por escena

Se conserva por consistencia la grafía original al citarlos.

- Escena «02a EscuelaPB»:
  - «MesaPeq/Layer:Mesa2» x7
  - «Manguera/Layer:Metal» x3

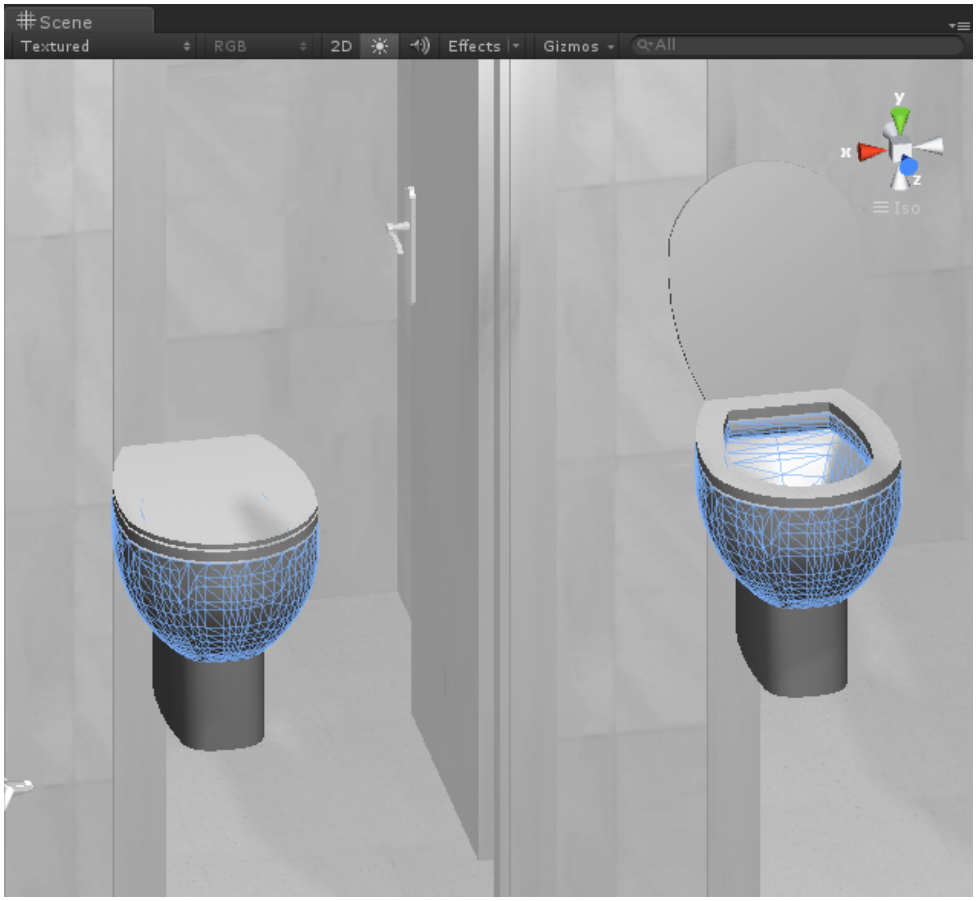


Figura 18.6: El *Mesh Collider* traza el contorno exacto alrededor de un objeto

- «Escuela - Vitrina/Layer:Madera»
- «Escuela - AccesoriosPB/Layer:KioskoNegro»
- «Escuela - PapeleraMaq»
- «PapeleraRec/Layer:Base»
- «Revistero/Layer:Cuerpo» x2
- «Extintor/Layer:Cuerpo» x3

- Escena «02b EscuelaP1":
  - «PapelerasP1/Papelera/Layer:Oro» x15
  - «Extintor2/Layer:Cuerpo» x4
  - «Manguera/Layer:Metal» x3
  - «Escuela - Maquinas/Layer:MaderaM2»
  - «Escuela - PapeleraMaq»
  - «Escuela - AccesoriosP1/Layer:Macetas» x15 (añadido *Mesh Collider*)
- Escena «05a BibliotecaAbajo»:
  - «Manguera/Layer:Metal» x2
  - «Biblioteca - Acesorios/Layer:BuzonR»
  - «PapeleraRc/Layer:Base» x3
  - «Biombo/Layer:BiomboMetal» x3
  - «Extintor/Layer:Cuerpo» x2
- Escena «05b BibliotecaArriba»:
  - «Biblioteca - MesaArriba»

- «Extintor/Layer:Cuerpo»
- «Manguera/Layer:Metal»
- «PapeleraRec/Layer:Base» x3

- Escena «07 Secretaria":
  - «Layer:Mob01 CopyPrintFax»
  - «Layer:Mob01 neg_arm1»
  - «Extintor/Layer:Cuerpo»
  - «Escena "11 Aula 216»
  - «Plotter 3ds/HP650C_sup»
- Escena «08 Aula 204»:
  - «Percheros/Perchero/Layer:PercheroMetalNegro»
  - «Escena "09 Aula H3»
  - «PizarraH3/Layer:Patas»
  - «Plotter/HP650C_sup»
  - «SillaProf/Layer:Cuero»
  - «SillaDibujo/Layer:SillaDibMadera» x2
- Escena «10 BanoET»:
  - «Bano/Layer:Caldera»
  - «SecadorBanoET/Layer:base»
  - «Bano/Layer:UrinariosPorc» (añadido *Mesh Collider*)
  - «Bano/Layer:WCArriba» (añadido *Mesh Collider*)

- Escena «03a EdifTecnologicoP1»:
  - «PapeleraRec/Layer:Base» x4
  - «EdTecn - Maqina2/Layer:negro»
  - «EdTecn - Maqina3/Layer:negro»
  - «ExtinyMang/Extintor/Layer:Cuerpo» x11
  - «31 EdTecn - EstrPByP1/Layer:InterseccionEsq»
  - «31 EdTecn - EstrPByP1/Layer:InterseccionEsq 1» (añadido *Mesh Collider*)

18.5. Evitado que el avatar pueda subirse encima de la planta de Secretaría

Se ha añadido para ello, alrededor del objeto «Secretaria/Layer:Mob01 Plastico_blanco» (planta decorativa de la escena «07 Secretaria»), un cubo invisible, H_AislantePlanta, que impide que el avatar pueda subirse encima. Puede verse en la Figura 18.8.

18.6. Solucionado el problema de formato de codificación de los vídeos

Problema

La codificación original de los vídeos incorporados en el proyecto legado los hace incompatibles con plataformas no Apple.

Descripción y solución

Estos vídeos, cuatro en total, se usan para simular que, tanto los monitores incluidos en diferentes escenas del mundo virtual como la pantalla de proyección del salón de actos, muestran secuencias de imágenes como lo haría un televisor real.

Figura 18.7: Objetos incorpóreos

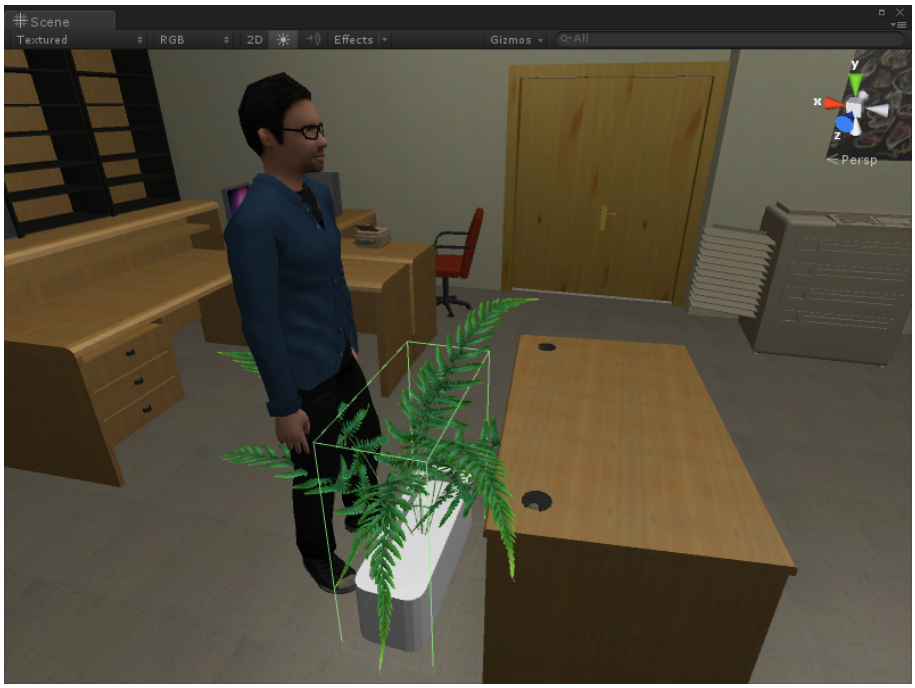


Figura 18.8: Aislante de la planta de Secretaría

La codificación XVID original de los vídeos, no obstante, hace que esto sea imposible en plataformas diferentes de Apple sin instalar antes, en la plataforma que se esté usando, ese o un códec compatible (Figura 18.10). Sin embargo, las alternativas para hacer esto o son de pago o son experimentales.

La solución definitiva consiste en recodificar los vídeos y convertirlos a un formato universal compatible con cualquier plataforma. La elección ha sido: formato MOV y método de compresión H264 (Figura 18.9).

Cabe destacar también que, en plataformas Windows, Unity importa los vídeos utilizando la aplicación externa QuickTime[2], que será necesario instalar de no estarlo ya. Una vez hecho Unity ya podrá importar vídeos como si de cualquier otro tipo de fichero se tratara.

Los vídeos originales se han conservado, pero a todo efecto práctico han sido sustituidos por los nuevos. La ubicación de los vídeos, tanto los antiguos como los recodificados, es «Proyecto/Videos».

Vídeos recodificados

- Presentación2004.avi → Presentación2004_MOV-H264.mov

[2]Puede bajarse gratuitamente desde la siguiente dirección: http://www.apple.com/es/quicktime/download/

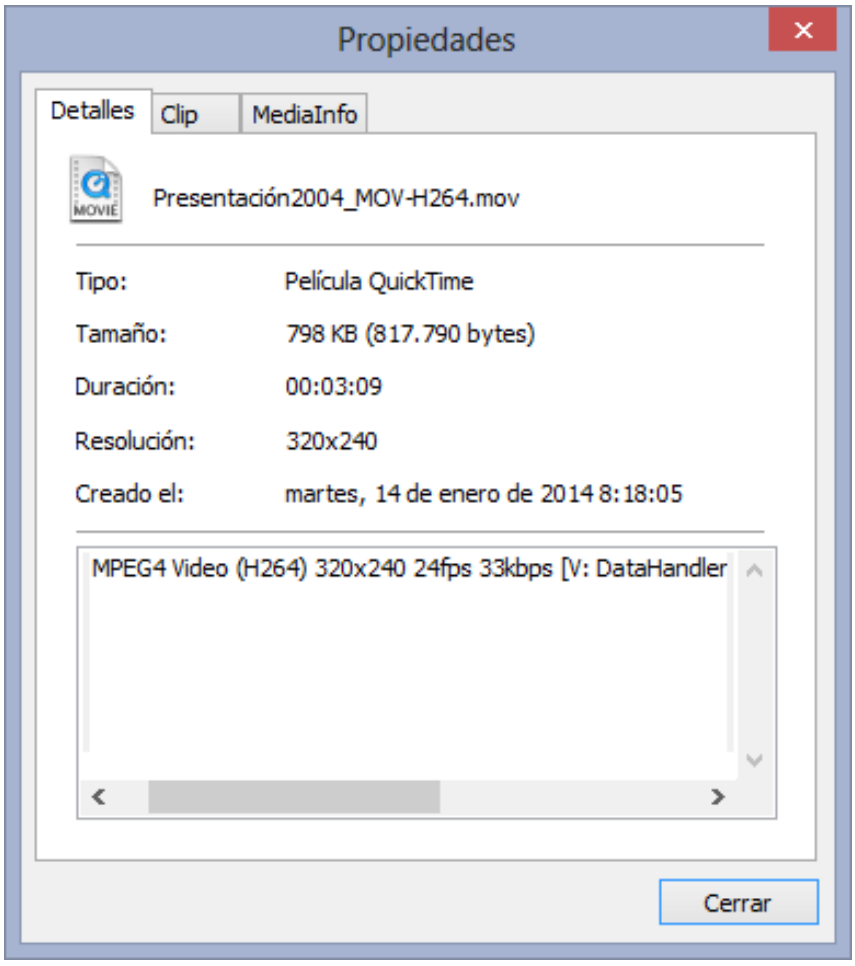


Figura 18.9: Propiedades de un nuevo vídeo recodificado

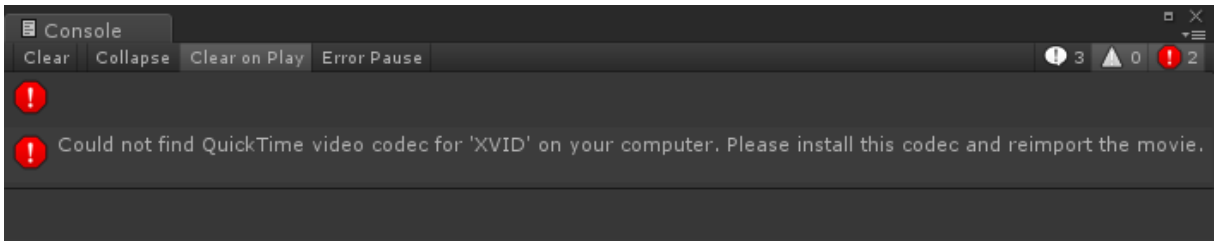


Figura 18.10: Codificación XVID incompatible

- Ruta-1.avi → Ruta-1_MOV-H264.mov
- Ruta-2.avi → Ruta-2_MOV-H264.mov
- Ruta-3.avi → Ruta-3_MOV-H264.mov

18.7. Ajustada la posición de los vídeos en sus respectivas pantallas

Problema

Los vídeos se mostraban desplazados, dejando ver bandas negras no uniformes en el espacio no cubierto. En la Figura 18.11 puede verse un ejemplo.

Solución

Asignar a las pantallas tres nuevos materiales creados expresamente con este fin, ubicados en la carpeta «Proyecto H/Materiales», y ajustar minuciosamente los valores *Tiling* y *Offset* (para las coordenadas X e Y).

Materiales creados

- Pantalla_SalonDeActos
- PantallaDos_03aEdifTecnologicoP1
- PantallaUno_02aEscuelaPB

(a) Antes

(b) Después

Figura 18.11: Antes y después de ajustar la posición de los vídeos

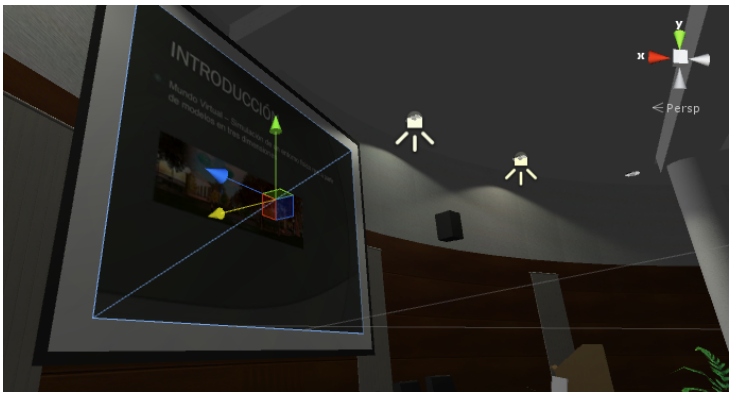


Figura 18.12: Borde en la pantalla de proyección del salón de actos

18.8. Corregidas las discordancias de color en la pantalla del salón de actos

Problema

Se aprecia un borde alrededor del vídeo asociado con esta pantalla, que, siendo el lienzo de un proyector, le hace parecer por contra un monitor (Figura 18.12).

Descripción y solución

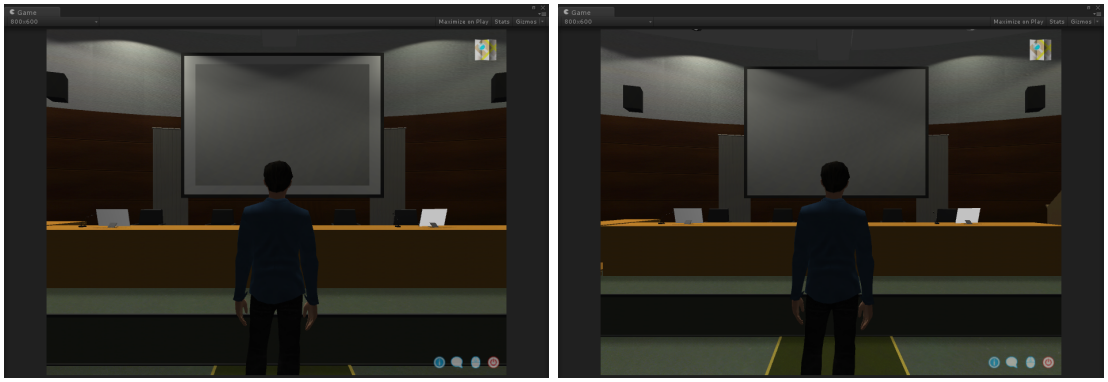
Las dimensiones del objeto original que sirve de lienzo cuando el vídeo no se está reproduciendo son diferentes a las de este. Así, cuando el vídeo se activa, siendo el lienzo más grande, aparecen las franjas de diferente color.

Para solucionarlo se ha añadido un nuevo objeto H_Pantalla en la escena «04 SalondeActos» que corrige la diferencia de tamaño (Figura 18.13). El objeto anterior, Pantalla, se ha conservado, pero se mantiene desactivado.

18.9. Mejorado el guión encargado de reproductor los vídeos

Problema

El código del guión «StartVideo 1.js», ubicado en la carpeta «Proyecto/Videos» y encargado de gestionar la presentación de los distintos vídeos habilitados en el proyecto, no cumple con la directiva `#pragma strict`, particularmente meticulosa en este marco.



(a) Antes (b) Después

Figura 18.13: Antes y después de incluir el liezo corrector

```
/*#################################################################

StartVideo 1.js
---------------

Guión del proyecto legado modificado y mejorado para el Proyecto H.

Guión encargado de gestionar la presentación de los distintos vídeos habilitados en
el proyecto.

Se ha modificado para adaptarse a la directiva "#pragma strict" y las recomendaciones
indicadas en "http://docs.unity3d.com/Documentation/Components/class-MovieTexture.html":

  "If you have #pragma strict enabled in your code a MovieTexture object
  should be declared somewhere and the object should be initialized with
  renderer.material.mainTexture. Then isPlaying, Play() and Stop() should
  be called for this MovieTexture object."

Se incluye al final la versión original del guión como referencia.

#################################################################*/
```

Cuadro 18.2: Documentación incluida tras modificar el guión «StartVideo 1.js»

Solución

Se ha modificado y mejorado el guión para ajustarse a dicha directiva y se han explicado esos cambios en el mismo (Cuadro 18.2).

18.10. Mejorados los guiones encargados de la apertura y cierre de la puertas

Problema

Los eventos *OnTrigger* de los guiones originales no acotaban su funcionamiento al avatar, pudiendo activarse más de una vez cuando no corresponde.

Descripción y solución

Si la respuesta de un objeto con la casilla *Is Trigger* activada no se acota manualmente, este activará el código que lleve asociado siempre que cualquier objeto entre en su área de acción.

Este comportamiento puede o puede no ser deseado. En este caso, el de las puertas que se abren o cierran según el avatar se acerca o aleja, no es deseado. Si el avatar estuviera compuesto por dos objetos en lugar de uno, aquella puerta a la que se acercara se activaría dos veces en respuesta a los dos cuerpos que acaban de interactuar con ella. Originalmente el avatar estaba compuesto por un solo objeto, pero en el presente proyecto se le ha añadido el objeto invisible, H_PuntoAvatar, para el cálculo preciso de colisiones (Sección 10.3). El resultado anterior a las modificaciones de los guiones, es que tanto las animaciones de apertura y cierre de las puertas, como los sonidos correspondientes, se reproducían dos veces como respuesta a ambos objetos.

Para solucionarlo se ha añadido a los bloques `OnTriggerEnter()` y `OnTriggerExit()` de todos y cada uno de estos guiones una comparación por etiqueta o *tag* (Cuadro 18.3) que permite acotar su funcionamiento exclusivamente a la interacción del objeto etiquetado con el área activadora, en este caso el avatar, identificado siempre con la etiqueta distintiva «Player». Se resuelven así todos aquellos escenarios en los que pudiera activarse la puerta más de una vez para una misma sola acción de entrada o salida, respondiendo tan solo al avatar.

Se ha tratado de manera especial el guión «Proyecto/03 EdifTecn/Obs/AbrirPuertaD277.js», resolviendo la particularidad de los avatares utilizados[3] por la que al transportarse directamente dentro del área activadora no desencadenan la acción hasta que el mismo no se mueve dentro de ella.

[3]Carecen de componente *RigidBody*.

```
1 function OnTriggerEnter (other : Collider)
2 {
3   if (other.tag == "Player")
4   {
5     targetValue = AngleY;
6     currentValue = 0;
7     AudioSource.PlayClipAtPoint(puertaAbrir, transform.position);
8   }
9 }
```

Cuadro 18.3: Comparación por etiqueta

18.11. Reajustados elementos desfasados en la escena «01 Exterior»

Problema

Posiblemente debido a la utilización de diferentes versiones de Unity para el desarrollo del proyecto global, algunos elementos de la escena «01 Exterior», correspondiente a los aparcamientos de la Escuela de Ingenierías, aparecían desfasados.

Solución

Se ha reajustado la posición, y dimensión allí donde hiciera falta, de los elementos implicados: asfalto, aceras, muros, pilares, líneas de aparcamiento, césped, bordillos, vehículos y terreno maestro.

Nótese que se han añadido, para corregir los desfases pertinaces, nuevos elementos agrupados en la escena bajo el prefijo «H_CorreccionDesfase», y nuevas texturas ubicadas en la carpeta «Proyecto H/Texturas/CorreccionesEscenaExterior». Estos elementos, mediante el ajuste preciso de sus texturas, se mimetizan perfectamente en el entorno. Puede verse un ejemplo en la Figura 18.19.

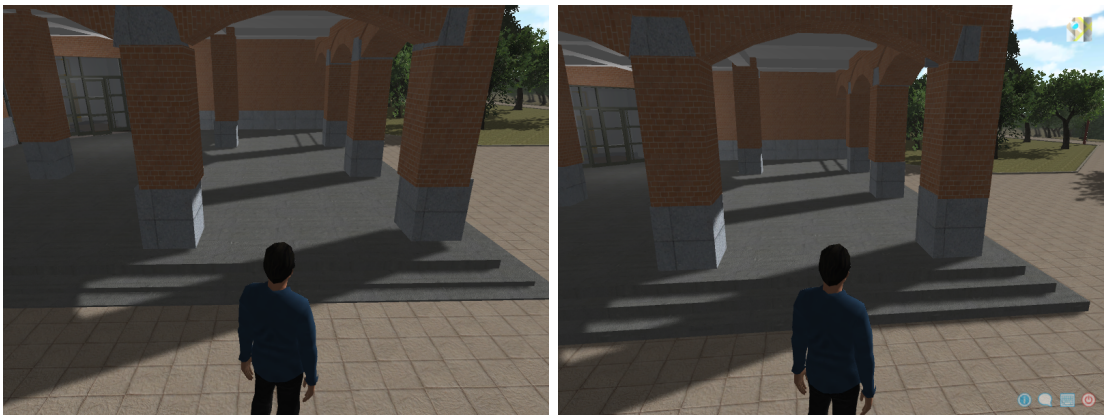
En las Figuras 18.14, 18.15, 18.16, 18.17 y 18.18, se pueden apreciar algunos ejemplos de este problema, antes y después de ser corregido.

(a) Antes (b) Después

Figura 18.14: Vehículos y asfalto desfasados

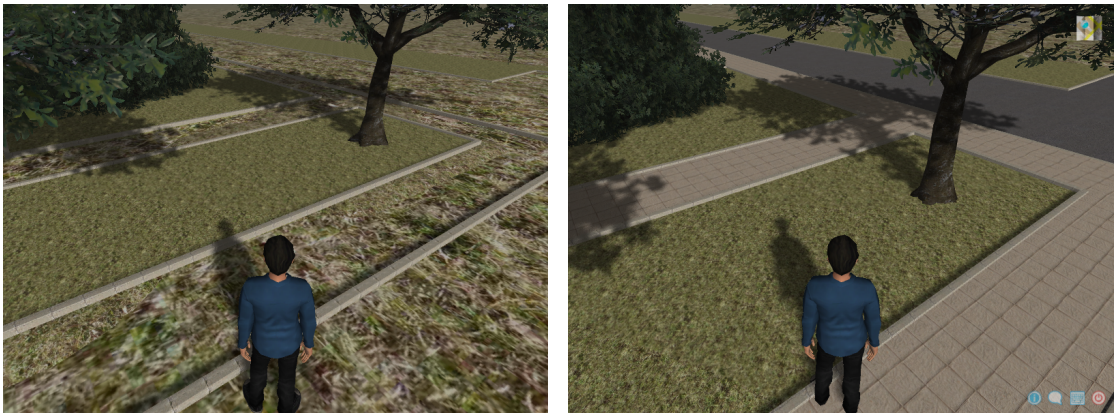
(a) Antes (b) Después

Figura 18.15: Acera y rampa desfasados



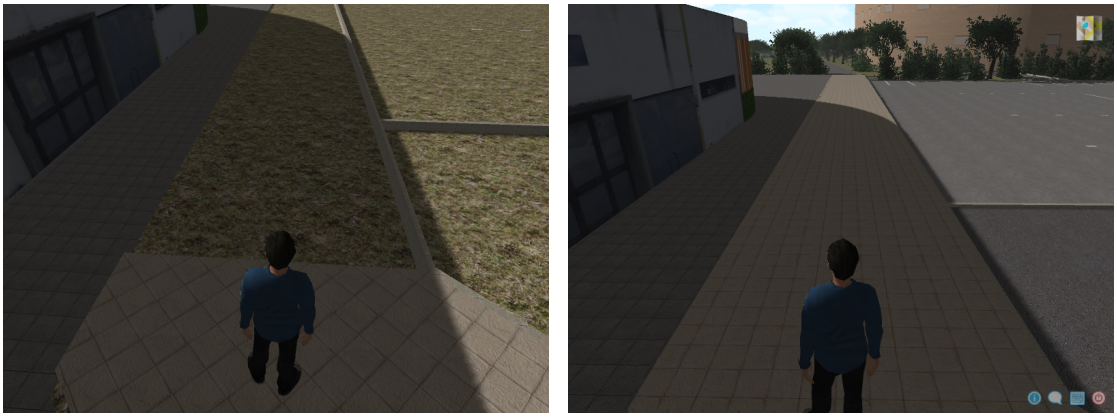
(a) Antes (b) Después

Figura 18.16: Columnas y acera desfasados



(a) Antes (b) Después

Figura 18.17: Aceras y asfalto desfasados



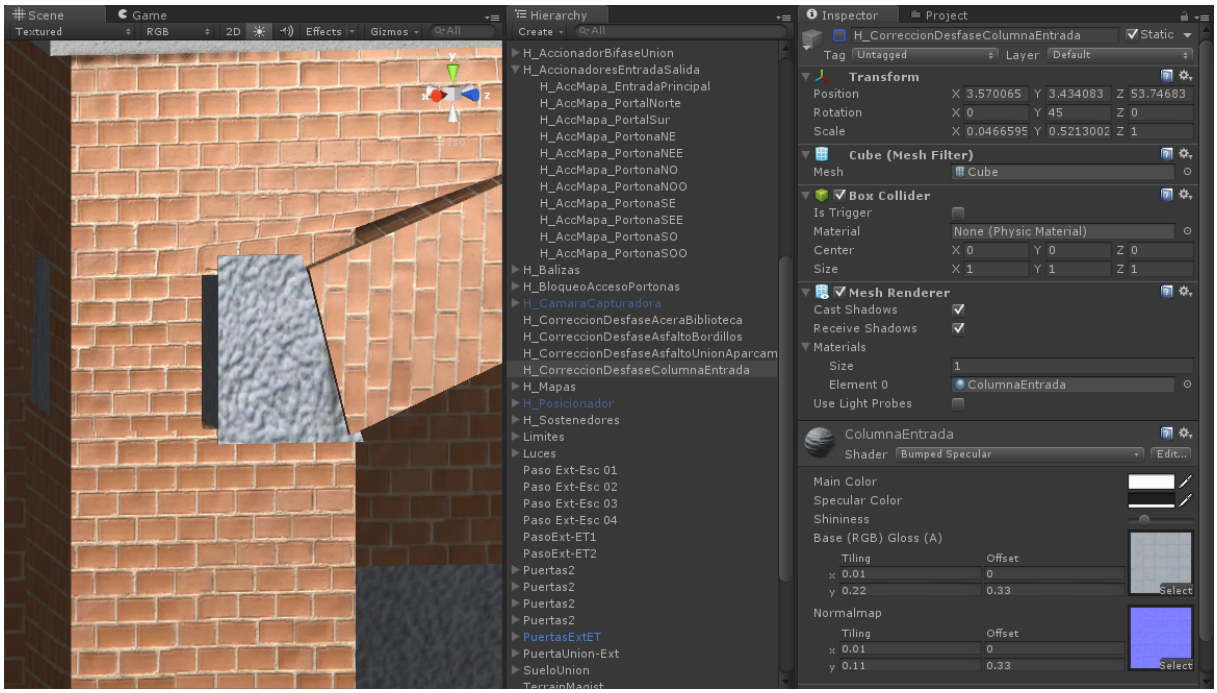
(a) Antes (b) Después

Figura 18.18: Aceras desfasadas

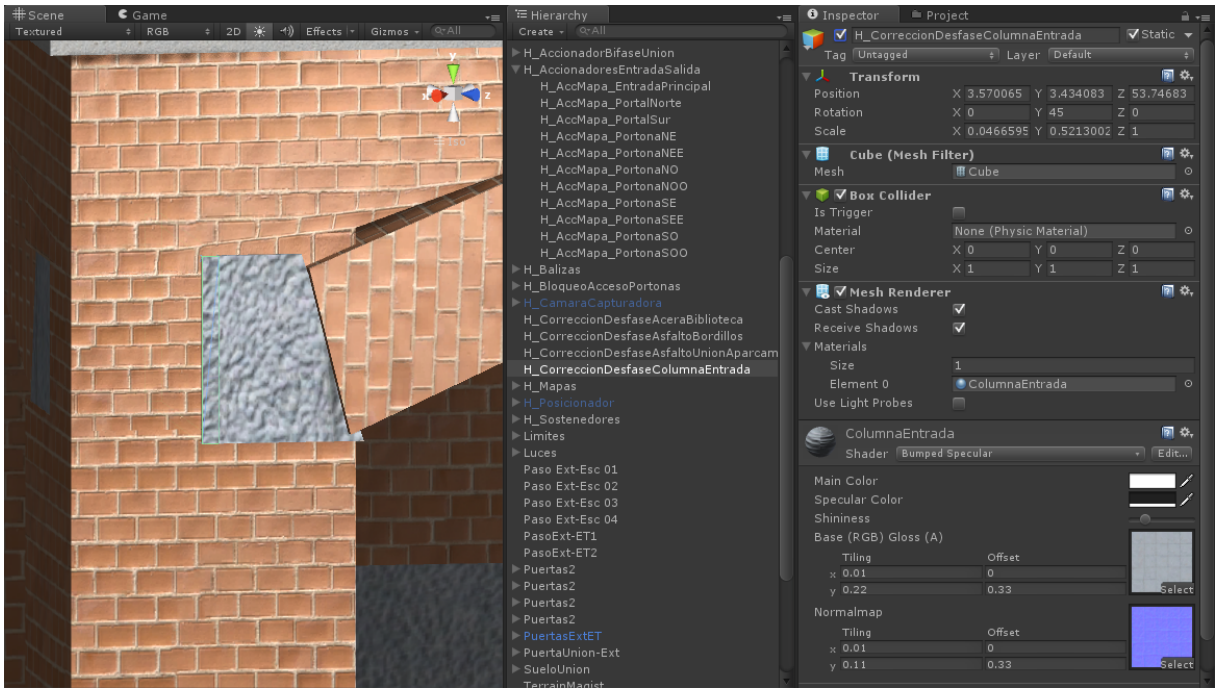
18.12. Solucionada la desaparición de elementos en las compilaciones finales

Problema

Concluida la parte técnica del presente proyecto se realizaron compilaciones finales para diversas plataformas, *Standalone* y *Web Player*. Se comprobó que en el resultado



(a) Desactivado



(b) Activado

Figura 18.19: Efecto de un objeto corrector añadido en una columna

final, elementos de la escena exterior («01 Exterior») presentes durante el desarrollo, desaparecían, cambiaban su posición o perdían su textura en esta fase final (Figuras 18.20)

Descripción y solución

Los elementos implicados son los mismos que los que originalmente habían aparecido ya desfasados (Sección 18.11), en particular los firmes del aparcamiento sur y las aceras este de la Escuela de Ingenierías.

Una solución aparente, pero en la práctica inviable, era reaplicar las mismas correcciones que solventaron los desfases por duplicado, es decir, si en la sección citada se modificó la coordenada X de un elemento desde -4 hasta -8, para corregir este nuevo problema habría que llevarla hasta -12. Esto significaba trabajar a ciegas pues el resultado de las modificaciones sólo se vería en la fase final, tras la compilación, impidiendo comprobar así la buena marcha de las nuevas correcciones; y, por otro lado, esta solución teórica exigiría modificar los elementos del mundo virtual de tal forma que su posición, durante la etapa de desarrollo, sería incorrecta. Por tanto, esta solución era intolerable.

La solución final eficaz consta de dos fases, una de ellas necesaria para todas las plataformas de compilación, la otra, necesaria adicionalmente para la plataforma *Web Player*.

Para las plataformas *Standalone* ha sido necesario activar la característica *Static* (Figura 18.21) en los elementos implicados en el problema. Esta propiedad, activada manualmente en objetos del mundo virtual que no vayan a moverse, permite considerarlos como una sola unidad al procesarlos para dibujarlos en pantalla[4]. No sólo esto aumenta el rendimiento de la aplicación final, sino que en este caso, también resuelve este particular problema en estas plataformas *Standalone*.

Esta solución, no obstante, no era total para la plataforma *Web Player*. En este caso fue también necesario activar explícitamente el procesamiento *Batch Static* al configurar los parámetros de compilación para esta plataforma[5] (Figura 18.22).

Esto permitía solucionar el aspecto visual, los objetos implicados ahora sí se veían en la compilación final para esta plataforma, sin embargo, eran incorpóreos, el avatar atravesaba estas aceras y asfaltos problemáticos y caía a las capas inferiores del terreno. Fue necesario por esta razón añadir elementos invisibles de soporte bajo estos firmes,

[4]Más información en http://docs.unity3d.com/Manual/DrawCallBatching.html.
[5]Sólo disponible en la versión Unity PRO.

(a) El firme ha desaparecido

(b) La textura del suelo ha cambiado

(c) La posición de las aceras es diferente

Figura 18.20: Problemas encontrados y corregidos en las compilaciones finales

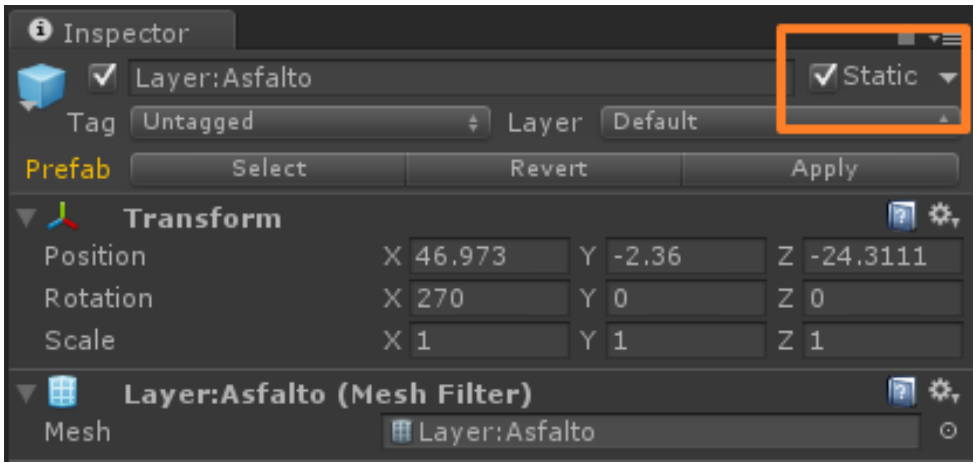


Figura 18.21: Propiedad *Static* activada

cuidando minuciosamente sus dimensiones, rotación y posición.

Nótese que, además de a estos objetos, se ha activado la propiedad *Static* también en aquellos otros que sin ser problemáticos, pudieran conformarse a la finalidad de uso de esta propiedad: muros, terreno, decorados, etcétera, con objeto de ganar en rendimiento. En particular, todos los elementos incluidos dentro de «Terreno» en la ventana de jerarquía de esta escena.

18.13. Solucionados los saltos descontrolados del avatar ante obstáculos

Problema

Al encontrarse con un desnivel el avatar saltaba incontroladamente (Figura 18.23) pudiendo incluso abandonar los límites construidos del escenario (Figura 18.24).

Descripción y solución

Este problema está relacionado con la ausencia de componente *RigidBody* del avatar, y que define su comportamiento mediante leyes físicas simuladas, entre ellas, la gravedad.

La solución menos intrusiva por la que ha optado para el tipo de avatar utilizado en la aplicación, ha sido modificar los guiones internos que controlan en particular su

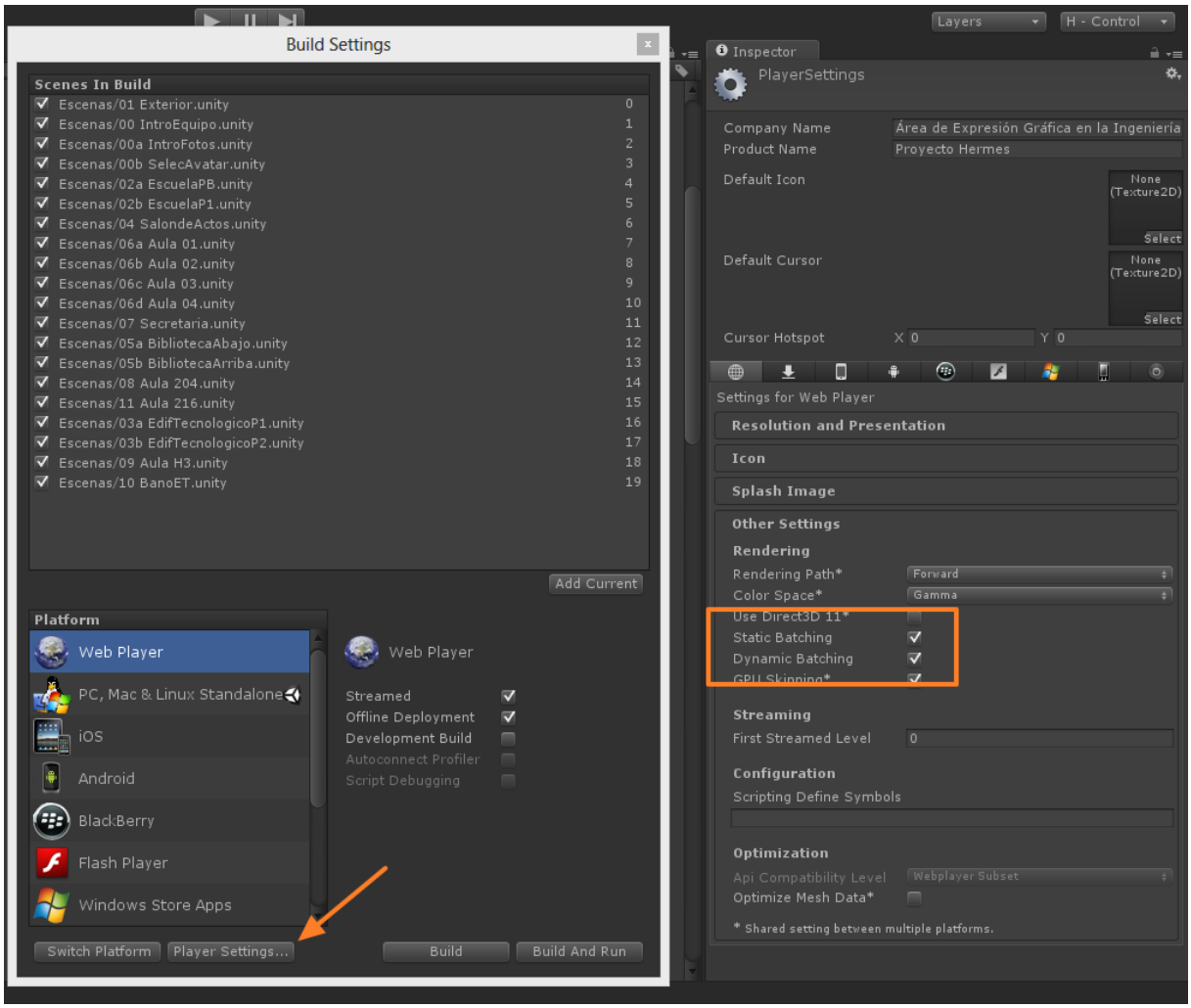


Figura 18.22: *Batch Static* activado para la plataforma *Web Player*

Figura 18.23: El avatar salta incontroladamente al encontrar un desnivel

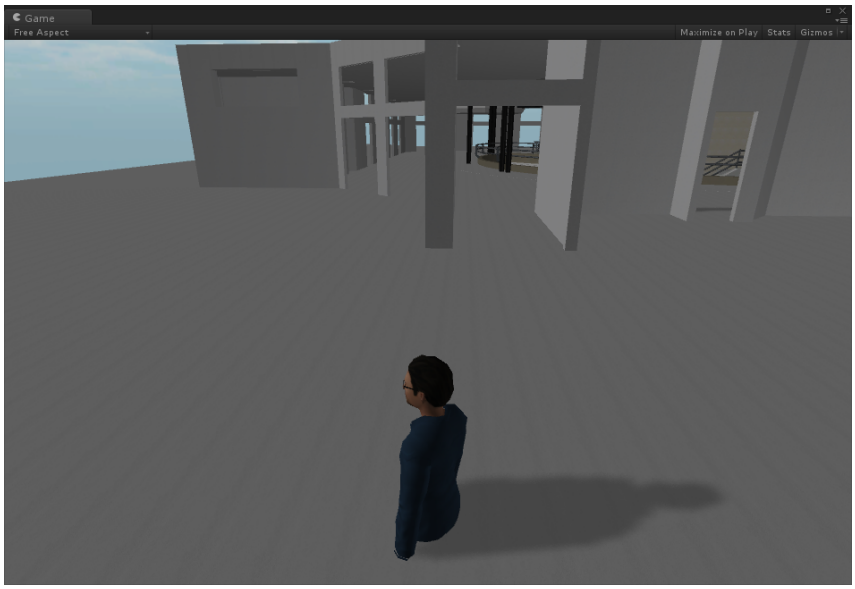


Figura 18.24: El avatar se encuentra ahora fuera de los límites de la escena

comportamiento, tanto para los avatares masculinos como para los femeninos. Los guiones modificados son:

- «Controladores/LocomotionPackage/Scripts/MixamoMaleLocomotionControlScripts.cs»
- «Controladores/FemaleLocomotionPackage/Scripts/MixamoFemaleLocomotionControlScripts.cs»

En ellos se ha ajustado el valor de la variable `gravity`, definida anteriormente con un valor de `9.81f`, a uno superior.

Valor original:

```
1 private float gravity = 9.81f;
```

Valor modificado:

```
1 private float gravity = 72.81f;
```

18.14. Solucionados los cambios de escena no deseados

Problema

El avatar, al cargar ciertas escenas, vuelve automáticamente a la escena anterior.

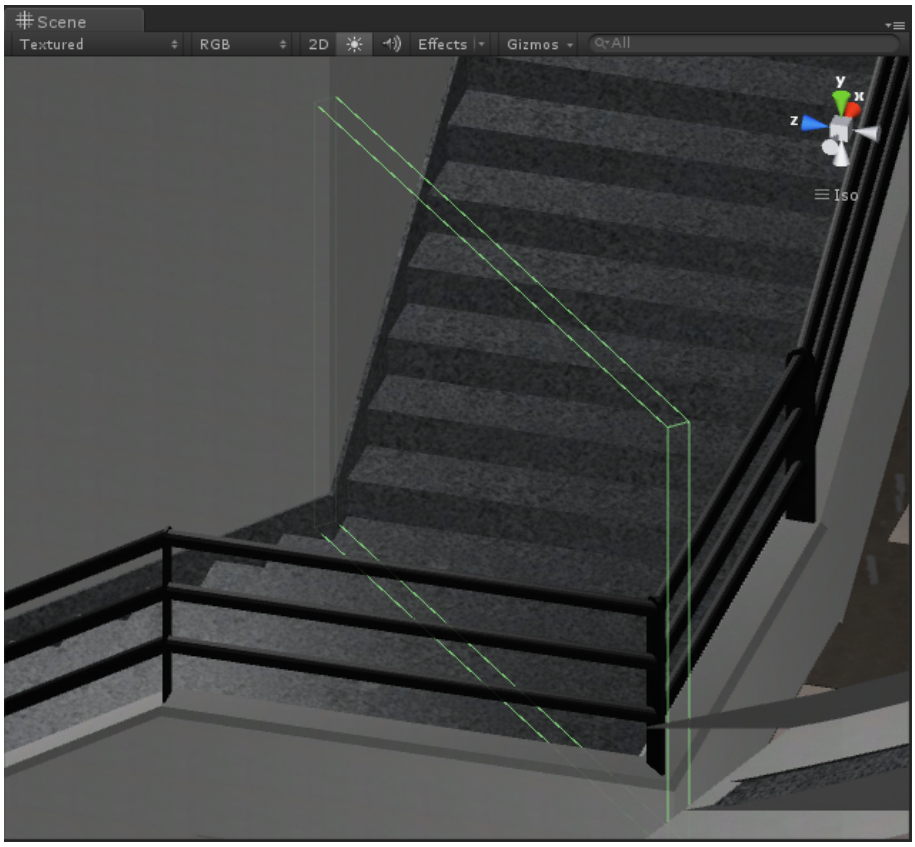


Figura 18.25: «Cubo» activador ubicado en las escaleras de unión entre dos escenas

Descripción y solución

Este problema surge como consecuencia del reajuste con mayor precisión de las coordenadas que definen el punto de aparición del avatar en las escenas, cuando cambia entre ellas por vías normales, puertas o escaleras (Sección 18.1).

Depurar estas coordenadas ha hecho necesario afinar también la posición de algunos de los «cubos» activadores de los cambios de escena que, ubicados en las puertas o escaleras de acceso entre ellas, se encargan en el proyecto legado de activar dicho cambio (Figura 18.25).

Sin este reajuste posterior, al cambiar el avatar de escena atravesando estos cubos, inmediatamente después de cargar la escena siguiente regresaba a la anterior como consecuencia de «caer», en la nueva escena, en el área de acción del cubo, activando así la carga automática e involuntaria de la escena previa.

La solución consiste en redimensionar y desplazar dichos cubos, impidiendo que su área de acción pueda entrar en conflicto con la posición inicial del avatar en las escenas implicadas.

Elementos cambiados:

- Escena correspondiente a la planta baja de la Escuela de Ingenierías, «02a EscuelaPB»:
  - «10 Paso Esc-A1»
  - «10 Paso Esc-A4»
  - «03 Paso Esc-Union»
  - «04 Paso Esc-Union»
- Escena correspondiente al primer piso de la Escuela de Ingenierías, «02b EscuelaP1»:
  - «14 Paso EscP1-216»
- Escena correspondiente a la planta baja de la biblioteca, «05a BibliotecaAbajo»:
  - «Paso Bibl-Esc»
- Escena que engloba la planta baja y primera del edificio tecnológico «03a EdifTecnologicoP1»:
  - «19 Paso ET-H3»

18.15. Reajustada la posición de los cubos de transporte del edificio tecnológico

Problema

El tránsito por los rellanos de las escaleras principales resultaba complicado con la disposición original de estos elementos, siempre implicaba un cambio de escena.

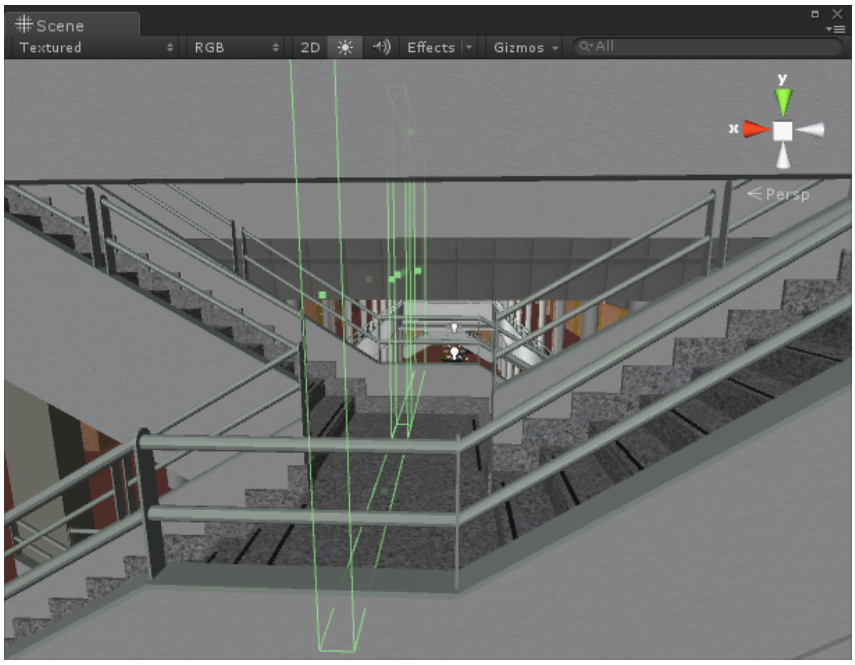


Figura 18.26: Disposición original de los cubos

Descripción y solución

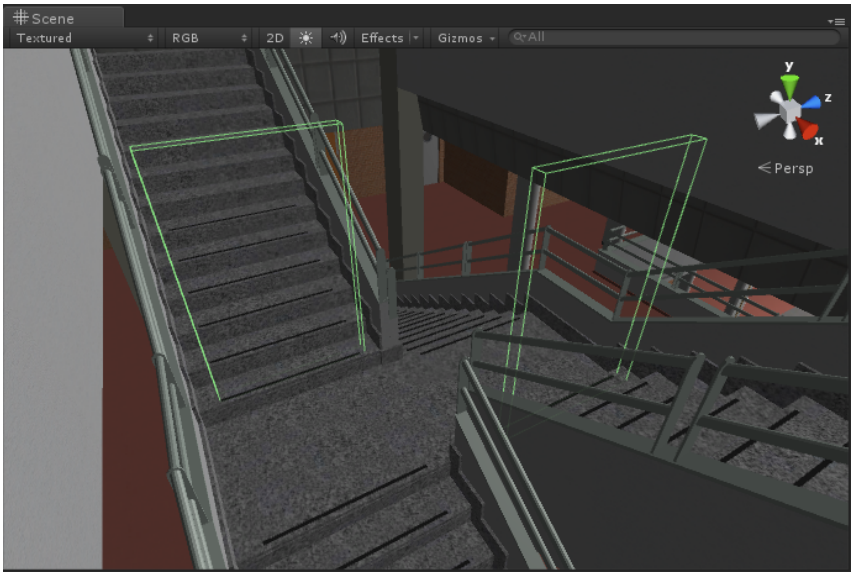
Estando los cubos originalmente ubicados en mitad de los rellanos de las escaleras (Figura 18.26), el usuario que, sin dejar de bajar o de subir quisiera cambiar de tramo, no podría. Al intentarlo entraría en contacto con uno de los cubos y activaría un cambio de escena involuntario, dejándolo al otro lado de las escaleras y en otra escena como resultado.

Para solucionarlo se han alejado los ocho cubos implicados de los rellanos, ubicándolos con precisión al principio de cada tramo de escalera correspondiente (Figura 18.27). De esta forma los rellanos quedan libres para poder ser transitados libremente, activando el cambio de escena sólo al iniciar el ascenso o descenso correspondiente.

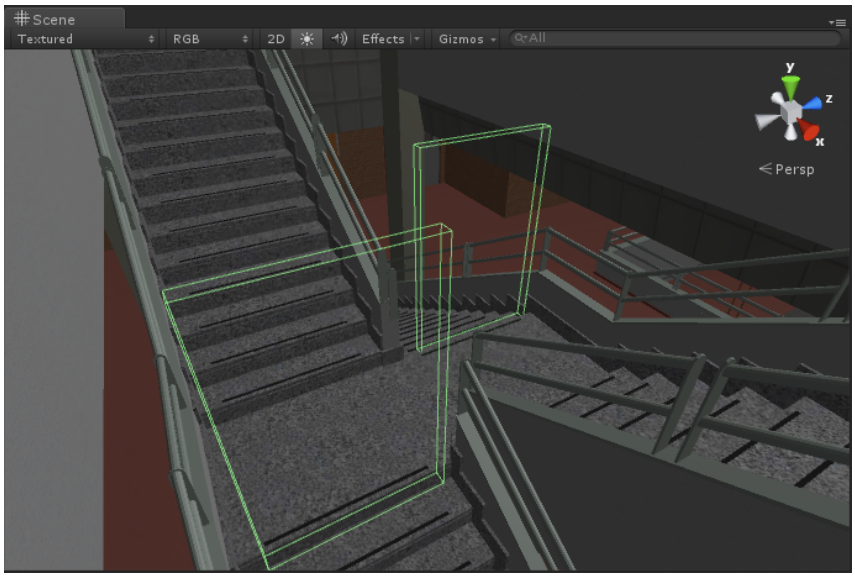
18.16. Corregido el guión «ScriptsPuertas/CambioEscenaBibl19.js»

Problema

El guión asociado al objeto que gestiona el cambio desde la escena «05b BibliotecaArriba» a «05a BibliotecaAbajo» contiene código erróneo que trata de cargar una escena inexistente («04a BibliotecaAbajo»). El cambio de escena es posible por la superposición



(a) Escena «03a EdifTecnologicoP1»



(b) Escena «03b EdifTecnologicoP2»

Figura 18.27: Disposición mejorada de los cubos

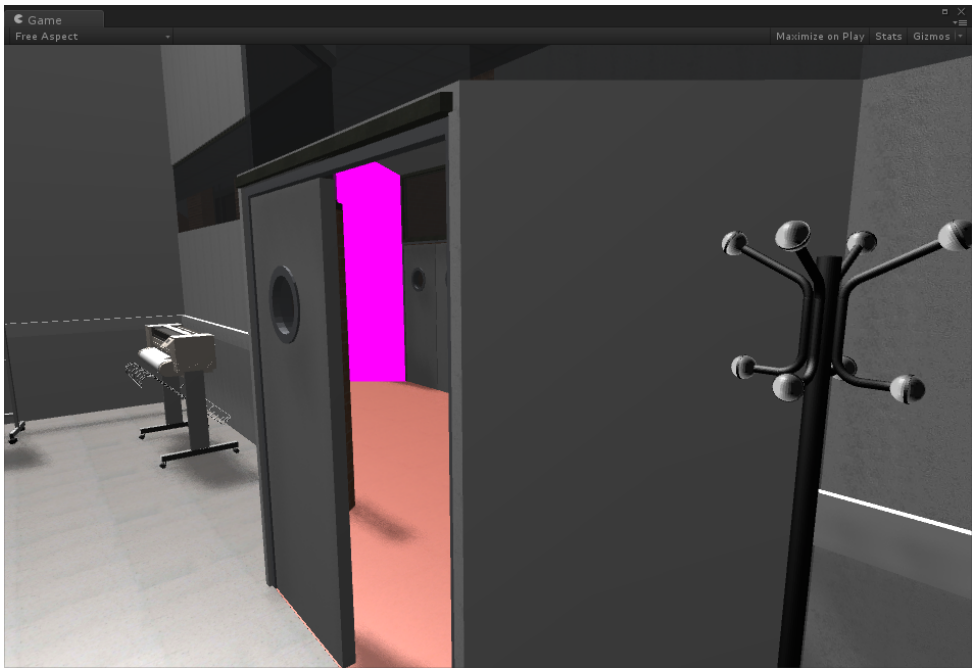


Figura 18.28: Una pared sin textura resalta poderosamente tras la puerta

con otro objeto con código incompleto que sí carga la escena adecuada.

Solución

Se ha corregido el código del guión del primer objeto y se ha desactivado el segundo.

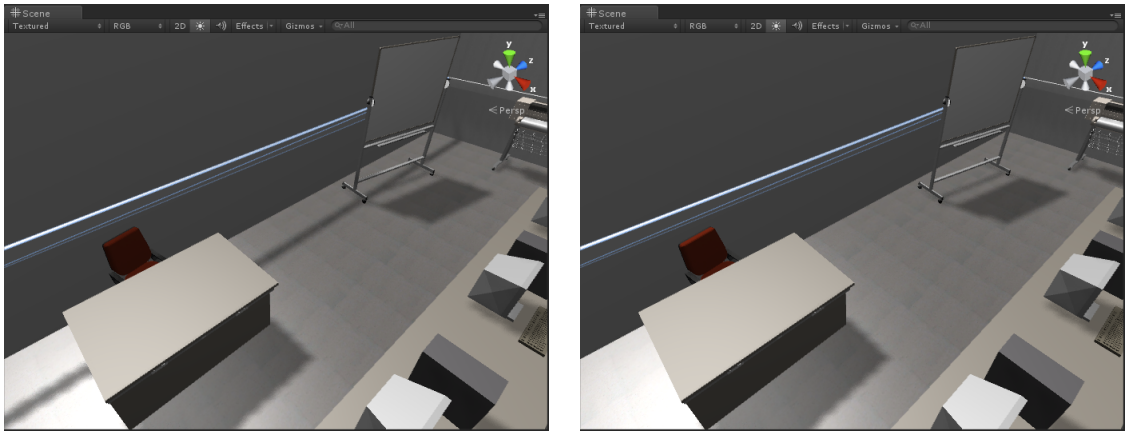
18.17. Corregida la falta de textura de una pared en el aula H3

Problema

El elemento «H3/Layer:LadrilloIntP1» de la escena «09 Aula H3» resaltaba poderosamente sobre el resto con el color rosa chillón predefinido para superficies sin textura (Figura 18.28).

Solución

Se ha añadido la textura «TexturasEscuela/LadrilloEscuela» en modo *Bumped Specular* a ese objeto para que el elemento se mimetice con el entorno.



(a) Con *Cast Shadows* activado hay una sombra irreal

(b) Sin *Cast Shadow*s activado se corrige

Figura 18.29: Correción de una sombra mediante *Cast Shadows*

18.18. Corregida la sombra antinatural del aula H3

Problema

La cenefa de la pared en la escena «09 Aula H3» producía un sombra irreal sobre el suelo.

Descripción y solución

Ese objeto, con la propiedad *Cast Shadows* activada que determina si ha de producir sombra, y por las características particulares de las paredes de esta aula, arrojaba una sombra antinatural sobre el suelo[6] (Figura 18.29).

Para solucionarlo se ha desactivado, para ese objeto, la citada propiedad (Figura 18.30).

[6]Resultado de considerar estas paredes elementos virtualmente transparentes.

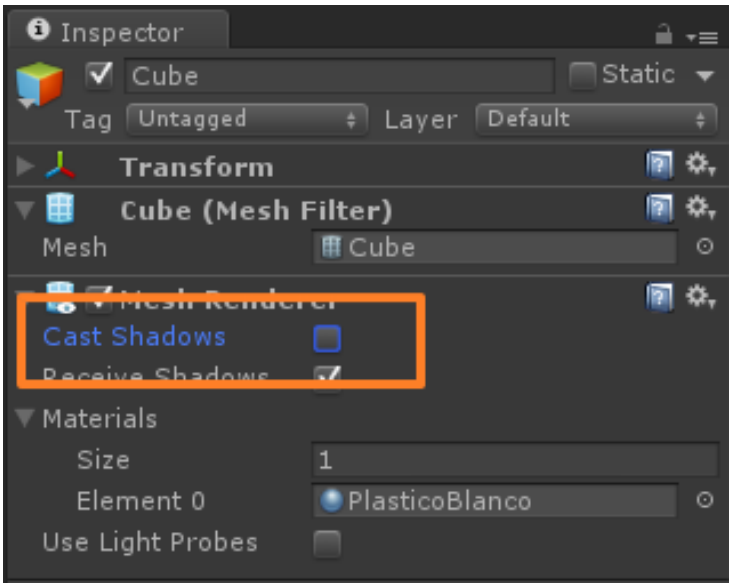


Figura 18.30: Desactivación de *Cast Shadows* en el Inspector

18.19. Ajustada la intensidad de luz en el aula H3

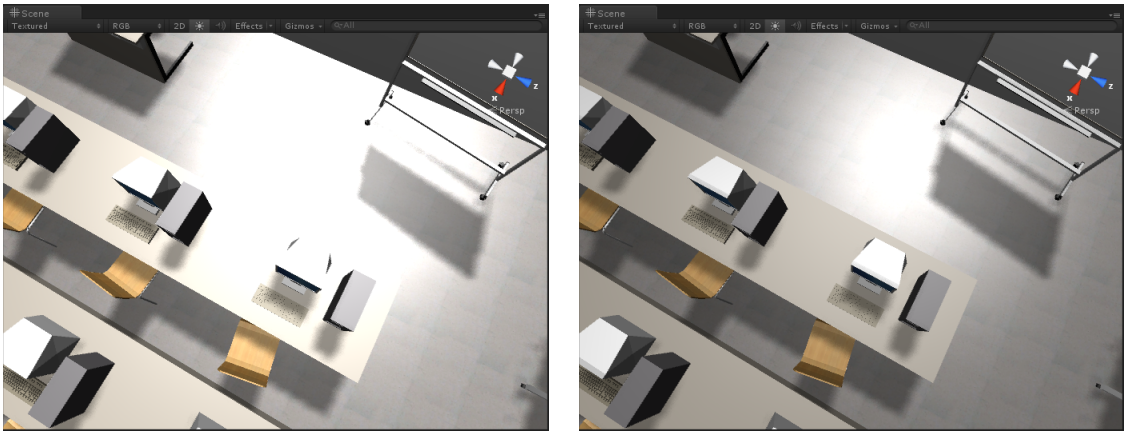
Problema

La luz es demasiado intensa y produce brillos excesivos en las superficies blancas del aula H3.

Descripción y solución

Consecuencia de una luz muy intensa, no se podían apreciar los detalles de las superficies de la escena «09 Aula H3».

La intensidad de un punto de luz queda determinada por su propiedad «Intensity» en el Inspector, por ello, para solucionar este inconveniente, se ha reducido ese valor para el objeto «Luces/Directional light», de 0.5 a 0.32 (Figura 18.31).



(a) Valor *Intensity* demasiado alto (b) Valor Intensity corregido

Figura 18.31: Correción de la intensidad de la luz en el aula H3

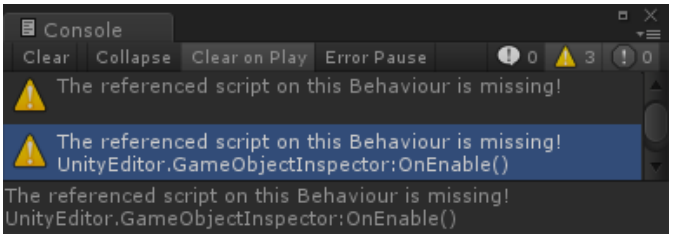


Figura 18.32: Advertencia en la consola de referencia rota

18.20. Corregida la ausencia del componente del objeto «PasoExt-ET2»

Problema

Un mensaje de advertencia aparecia en la consola cada vez que se cargaba la escena «01 Exterior» en la que se encuentra ese objeto (Figura 18.32).

Descripción y solución

El mensaje es inócuo, y se genera cuando se borra de la ventana de proyecto un elemento al que un objeto hacía antes referencia mediante su Inspector. Al borrar el elemento si eliminar también la referencia, Unity avisa de que hay una relación rota (Figura 18.33)

La solución consiste en eliminar la referencia en desuso.

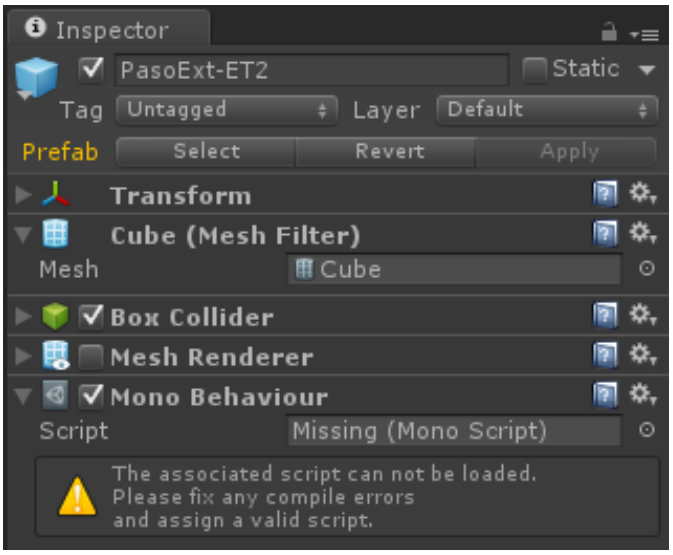


Figura 18.33: Inspector con una relación en desuso

18.21. Bloqueados los accesos a puntos a los que el avatar no puede entrar

Problema

El avatar podía acceder a lugares no modelados a través de espacios en las intersecciones de ciertos muros y paredes incorpóreas.

Descripción y solución

Se han añadido elementos bloqueadores de paso en aquellos puntos a los que el avatar no debería tener acceso:

Puntos corregidos:

- Cada uno de los ocho accesos auxiliares al edificio tecnológico.

  (El avatar podía acceder, en la escena «01 Exterior», al interior del edificio tecnológico sin modelar, mediante fisuras en las intersecciones de los muros).

- Puerta derecha de las dos secundarias de acceso a la planta baja de la escuela.

(La puerta derecha está cerrada por diseño, aun así se podía atravesar; se ha bloqueado el acceso por ella en ambos sentidos).

- Acceso al interior de todos los ascensores del edificio tecnológico.

  (El avatar podía acceder al hueco de los ascensores y, en ciertos casos, quedar atrapado dentro; los accesos se ha bloqueado hasta que no se modele el interior).

- Acceso al recinto ocupado por los servicios de impresión.

  (Se podía acceder a esta área sin modelar a través del marco de las puertas cerradas de entrada a este espacio).

- Acceso al recinto ocupado por el laboratorio de aeronáutica.

  (Se podía acceder a él atravesando las puertas cerradas de acceso).

Elementos utilizados:

Para realizar los bloqueos se han añadido planos invisibles verticales de alto rendimiento, ubicados en cada una de las entradas a los recintos citados. Si bien se han elegido estos objetos por cuestión de optimización de recursos, puede utilizarse para este fin cualquier otro elemento correctamente configurado. Estos planos están agrupados en las escenas correspondientes bajo el sufijo «H_BloqueoAcceso».

Nótese que, para impedir el paso correctamente mediante esta técnica, el eje Z positivo del plano ha de coincidir con la dirección de avance del avatar (Figura 18.34). En particular, para bloquear el acceso a través de la puerta derecha de la planta baja de la escuela, se han utilizado dos planos, uno en cada sentido de avance pues el avatar podía estar a un lado u otro de la puerta.

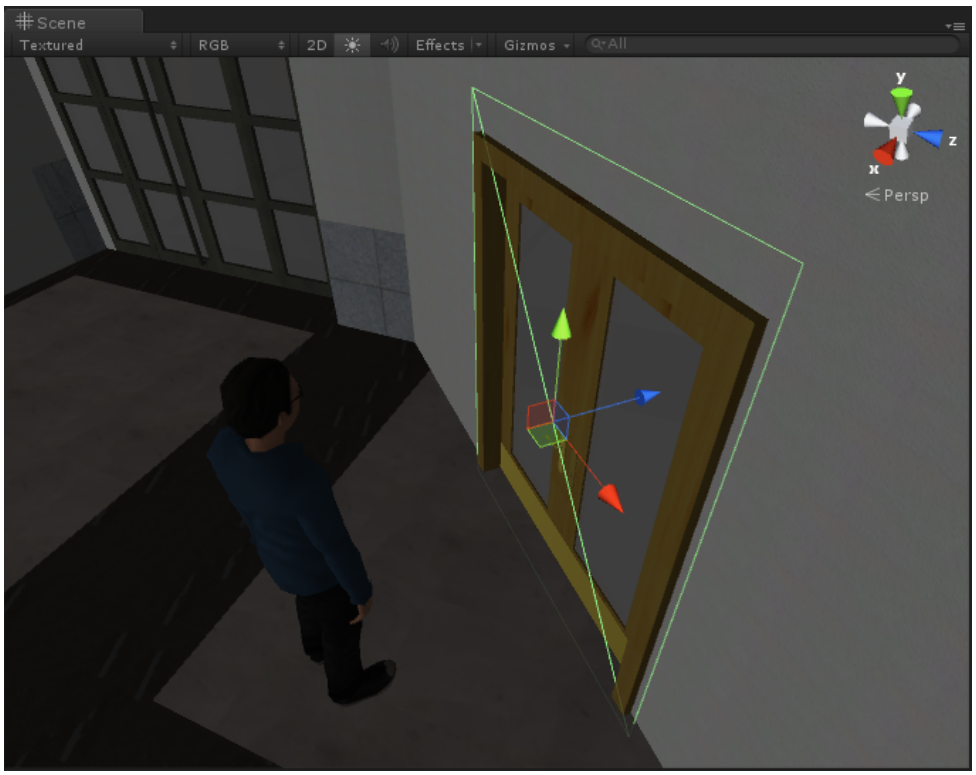


Figura 18.34: Plano invisible configurado para bloquear el acceso en un sentido

Capítulo 19

Despedida

Querido lector, llegado este punto, ese viaje al que poéticamente se hacía mención en la introducción, llega ahora a su fin.

Espero que haya disfrutado de la lectura y haber sabido, en forma grata, transmitirle el interés, encanto y potencial de los mundos virtuales tridimensionales.

Le doy sinceramente las gracias por el tiempo que ha dedicado a compartir las experiencias, conocimientos y descubrimientos realizados en este campo a través de este proyecto. Es usted, el lector, el que le otorga valor a estas líneas.

Me despido invitándole a curiosear los apéndices, si deseara profundizar más en este proyecto ahí se encuentran los guiones que lo sustentan, incluyen gran cantidad de información sobre los cómos y los porqués de cada paso dado en ellos.

Finalmente, no quisiera decirle adiós sin desearle, ahora que el paseo por este mundo virtual termina, que en este otro, aquí en el real... sea usted muy feliz.

Bibliografía

[1] "CreatePlane - unify community wiki." [En línea]. Disponible en: http://wiki.unity3d.com/index.php?title=CreatePlane

[2] "How to pass data between scenes – static variables | christian h." [En línea]. Disponible en: http://blog.christianhenschel.com/2013/05/16/how-to-pass-data-between-scenes-static-variables/

[3] "IKONY KE STAŽENÍ: vlajky evropské unie EU (free icons download) | dooffy design - world for everyone (adobe photoshop, tutorials, icons, freebies, fun, dooffy photos, vectors and more...)." [En línea]. Disponible en: http://www.dooffy.com/cs/ikony-ke-stazeni-vlajky-evropske-unie-eu-free-icons-download.html

[4] "Shutdown iconos, iconos gratis en crysigns, (buscador de íconos)." [En línea]. Disponible en: http://findicons.com/icon/77937/shutdown

[5] "Unity performance quick tips: Draw calls, triangles, + more | team think labs." [En línea]. Disponible en: http://blog.teamthinklabs.com/index.php/2013/06/25/unity-performance-quick-tips-draw-calls-triangles-more/

[6] W. Goldstone, *Unity Game Development Essentials*. Birmingham, UK: Packt Publishing Ltd., Oct. 2009.

[7] "3 ways to capture a screenshot in Unity3D | ralph barbagallo's self indulgent blog." [En línea]. Disponible en: http://ralphbarbagallo.com/2012/04/09/3-ways-to-capture-a-screenshot-in-unity3d/

[8] "The best place for answers about unity - unity answers." [En línea]. Disponible en: http://answers.unity3d.com/index.html

[9] "compass iconos, iconos gratis en WP woothemes ultimate, (buscador de íconos)." [En línea]. Disponible en: http://findicons.com/icon/69562/compass?id=69672

[10] "DocWiki | main / class-Texture2D." [En línea]. Disponible en: http://docwikiunity3d.ruskyhosting.ru/indexf8b8.html?n=Main.class-Texture2D

[11] "GPS navigation (PSD) | PSDGraphics." [En línea]. Disponible en: http://www.psdgraphics.com/psd-icons/gps-navigation-psd/

[12] "Gratis mapas icono :: gratuito para uso comercial :: disponible en png formatos :: diseñado por bharathp." [En línea]. Disponible en: http://www.fancyicons.com/gratis-iconos/157/aplicacion-general-icon-set/gratis-mapas-icon-png/

[13] "Sansation | dafont.com." [En línea]. Disponible en: http://www.dafont.com/es/sansation.font

[14] "Stack overflow." [En línea]. Disponible en: http://stackoverflow.com/

[15] "Strider van - YouTube." [En línea]. Disponible en: http://www.youtube.com/user/Stridervan/featured

[16] "Super mono - 193 iconos gratuitos, buscador de íconos." [En línea]. Disponible en: http://findicons.com/pack/2332/super_mono

[17] "Unity community." [En línea]. Disponible en: http://forum.unity3d.com/

[18] "Unity - manual: Unity manual." [En línea]. Disponible en: http://docs.unity3d.com/Manual/

[19] "Unity - learn - modules." [En línea]. Disponible en: https://unity3d.com/learn/tutorials/modules

[20] "Unity - scripting API:." [En línea]. Disponible en: http://docs.unity3d.com/ScriptReference/

[21] "UnitySpain." [En línea]. Disponible en: http://unityspain.com/

[22] "V5 prophit | dafont.com." [En línea]. Disponible en: http://www.dafont.com/es/v5prophit.font

[23] "Wikipedia, the free encyclopedia." [En línea]. Disponible en: http://en.wikipedia.org/wiki/Main_Page

[24] M. Smith y C. Queiroz, *Unity 4.x Cookbook*. Birmingham, UK: Packt Publishing Ltd., Jun. 2013.

[25] V. Gerasimov y D. Kraczla, *Unity 3.x Scripting*. Birmingham, UK: Packt Publishing Ltd., Jun. 2012.

[26] "Lenguaje visual c# (c#)." [En línea]. Disponible en: http://msdn.microsoft.com/es-es/library/aa287558(v=vs.71).aspx

[27] "Using movie texture for video playback in unity | unity cookie." [En línea]. Disponible en: http://cgcookie.com/unity/2013/08/11/using-movie-texture-for-video-playback-in-unity/

Apéndice A

Guía rápida de uso del Sistema de Captura de Mapas

A continuación se resumen los pasos necesarios para realizar una captura con el Sistema de Captura e integrar el mapa resultante en cualquier proyecto.

1. Asegurarse de que el objeto CambioControlador en la escena tiene en su guión Sust-Contr.js la variable «Camara Mapa» asignada con H_CamaraMapa, si no, arrastrar el prefab H_CamaraMapa desde la carpeta «Proyecto H» en la ventana *Project* a la variable «Camara Mapa» mencionada para asignarlo.

2. Copiar H_CamaraCapturadora en la escena si no está ya presente (de la ventana *Project* a la ventana *Hierarchy*).

    a) Ajustar su posición (se recomienda trabajar en modo isométrico) y configurarla según las necesidades de la captura.

3. Cambiar la resolución de la ventana *Game* a formato cuadrado 1:1.

4. Ejecutar la escena para realizar la captura y esperar hasta que aparezca el mensaje en la consola «Captura realizada».

5. Localizar la captura en el árbol de directorios del sistema operativo y arrastrarla a la carpeta «Proyecto H/Mapas» de la ventana *Project* en Unity (o a cualquier otra carpeta creada para tal efecto) .

    a) Cambiarle el nombre.

    b) Cambiar el tamaño de la textura (*Max Size*) a 2048 o 4096 según el nivel de

detalles presentes en la la escena.

6. Crear el plano contenedor mediante la herramienta de creación de planos de alto rendimiento (menú *GameObject ▷ Create Other ▷ Plano personalizado H*).

   a) Configurar su tamaño según el valor indicado en el fichero de coordenadas.

   b) Añadir la etiqueta «Mapa H».

   c) Darle un nombre.

7. Creado el plano, desactivar de su componente *Mesh renderer* las casillas *Cast Shadow* y *Receive Shadow*.

8. Arrastrar la textura del mapa debajo del componente *Mesh renderer* del plano.

   a) Cambiar el *Shader* de *Diffuse* a *Unlit/Texture*.

9. Posicionar el plano según las coordenadas X, Y y Z indicadas en el fichero de coordenadas.

10. Desactivar (o borrar) el objeto H_CamaraCapturardora copiado a la escena en el paso 2 (ventana *Hierarchy*). (No borrar el prefab original ubicado en la ventana *Project*).

11. Devolver la resolución de la ventana *Game* a su estado anterior.

Apéndice B

Guía detallada de uso del Sistema de Captura de Mapas

A continuación se describen detalladamente cada uno de los pasos necesarios para realizar una captura con el Sistema de Captura e integrar el mapa resultante en cualquier proyecto.

B.1. Ajustes en el guión «SustContr.js»

En el Inspector del objeto CambioControlador, presente en cada escena, se añadirá en la variable «Camara Mapa» de su componente SustContr.js, el prefab H_CamaraMapa[1] en caso de que no estuviera ya añadido. Para ello bastará pulsar con el ratón sobre el objeto CambioControlador y, a continuación, arrastrar el prefab citado sobre el campo de asignación de la variable «Camara Mapa» que aparece en el Inspector. Hecho esto, el texto *None (Game Object)* que aparecía en el campo de la variable será sustituido por H_CamaraMapa (Figura B.1).

Este paso es necesario pues, H_CamaraMapa es elemento central del presente proyecto, el constructor, el que se encarga de preparar inicialmente el resto de sistemas nada más cargar una escena.

[1]El prefab H_CamaraMapa está incluido en la ventana *Project* dentro de la carpeta «Proyecto H».

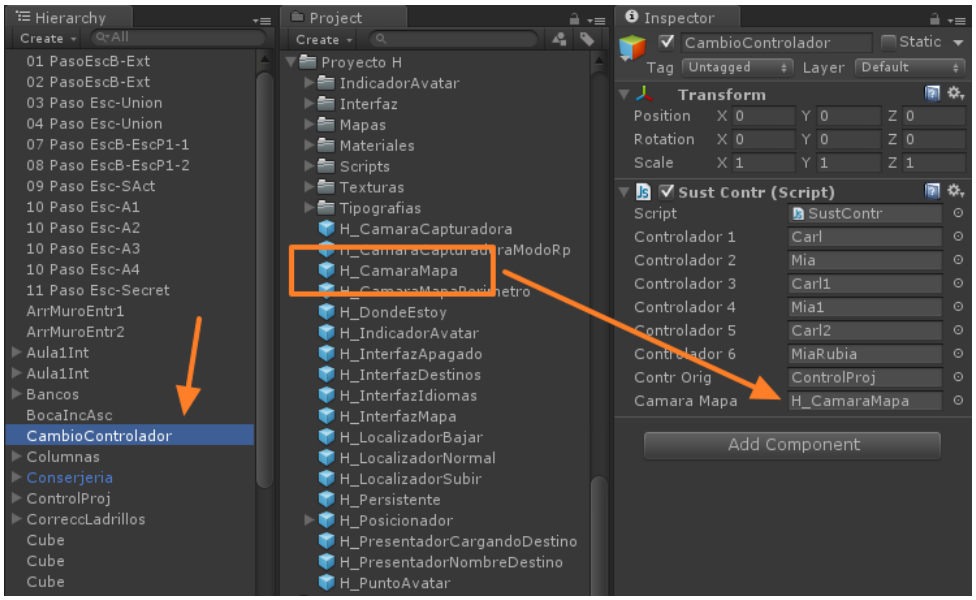


Figura B.1: Ajustes previos a la captura

B.2. Realización de la captura

El elemento central para la realización de la captura es el prefab H_CamaraCapturadora. Este puede capturar el plano tanto de forma automática como manual.

Método automático Para utilizar el método automático basta con activar la casilla «Captura Automatica» en el Inspector de H_CamaraCapturadora. De este modo, nada más ejecutar la escena pulsando el botón *Play*, se realizará una captura con los valores de configuración que se hayan elegido, cantidad de terreno, altura desde la que se hará la toma y la calidad de esta (Figura B.2). La captura tendrá como centro la posición en la que el avatar aparece de forma predeterminada en la escena. Una vez se haya detenido la ejecución de la escena, pulsando sobre el botón *Stop*, la casilla «Captura Automatica» se desactivará automáticamente para prevenir sobreescrituras accidentales, así, si se desea realizar una nueva captura habrá que volver a activarla.

Método manual En el método manual el desarrollador tiene la libertad de colocar el centro de captura, las coordenadas de posición[2] X, Y y Z de la cámara, donde desee, y puede asimismo, mientras ajusta la posición de la cámara, previsualizar el resultado final en la ventana *Game*. Para ello se deberá copiar el prefab H_CamaraCapturadora en

[2]Los valores de rotación, configurados de manera predeterminada para que la cámara observe el terreno a vista de pájaro, se mantendrán constantes.

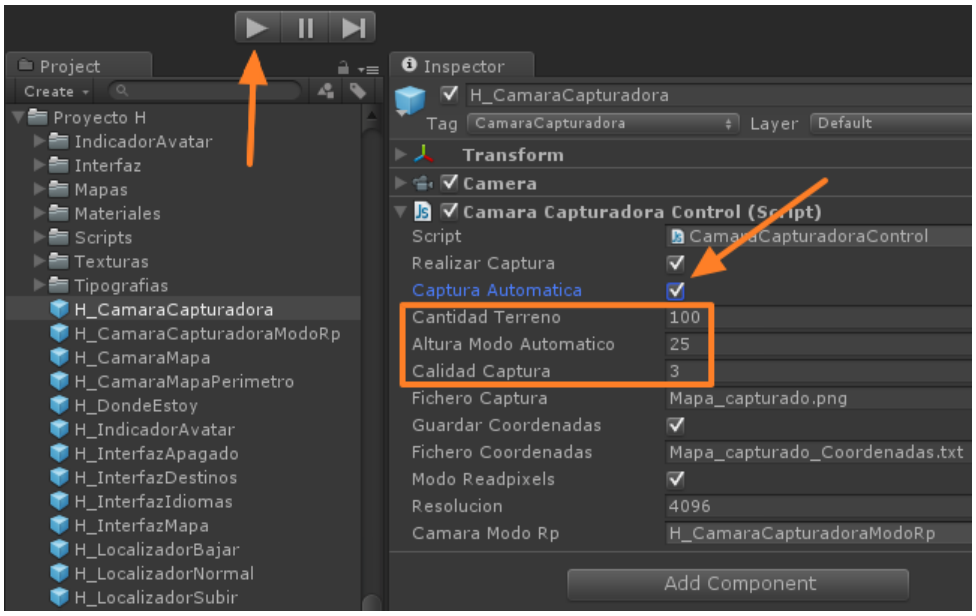


Figura B.2: Captura en modo automático

la jerarquía de la escena, desactivar la casilla «Captura Automática[3]» en su Inspector y comprobar que la casilla «Realizar Captura» está activada (Figura B.3). Mientras esta permanezca activada, siempre que se ejecute la escena se realizará una nueva captura[4]. Si se desea evitar esto bastará con desactivar la citada casilla, o borrar o inhabilitar el prefab en la ventana de jerarquía (para inhabilitarlo habrá que desactivar la casilla que aparece en la primera línea de su Inspector, a la izquierda de su nombre).

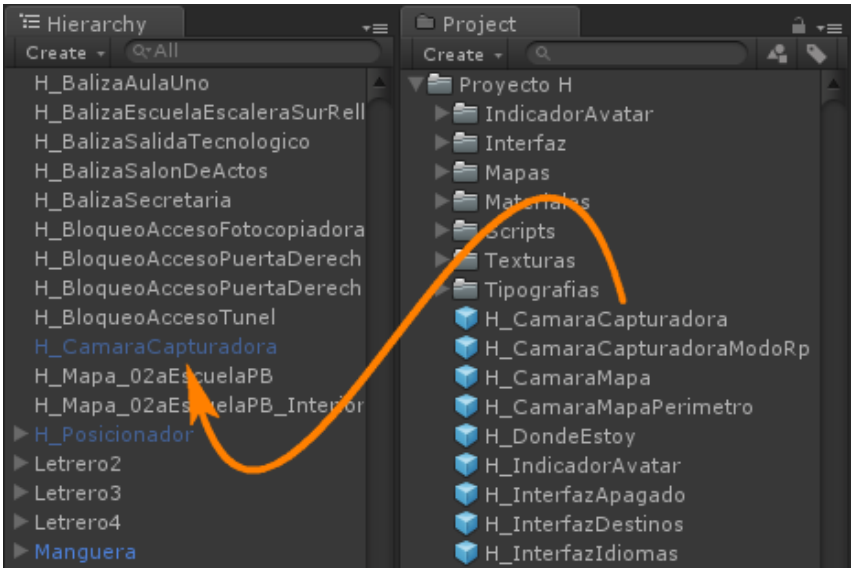
En ambos casos se informará al usuario de que la captura se ha realizado con el mensaje «Captura realizada» a través de la consola, y, en caso de haber detectado el sistema algún error de uso, se informará también.

B.3. Configuración de la captura

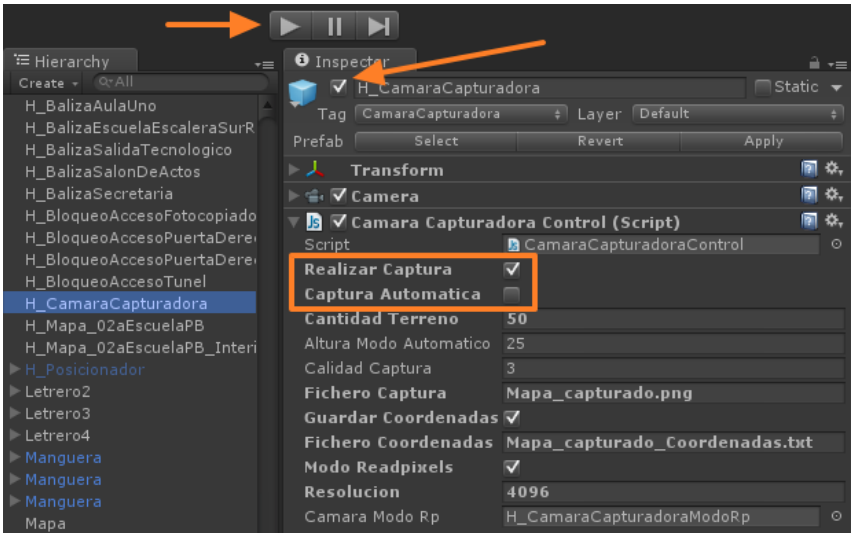
Hay varios aspectos configurables que el usuario puede modificar para que la captura se ajuste al máximo a sus necesidades (Figura B.4). Todos ellos contenidos en el apartado *Camara Capturadora Control (Script)* en el Inspector del prefab H_CamaraCapturadora. Los cambios se realizarán directamente sobre el prefab original si la captura es automática, y sobre el objeto copiado en la ventana de jerarquía si la captura es manual.

[3]Si la casilla «Captura Automática» estuviera activada durante el modo manual, el valor asignado al campo «Altura Modo Automático», en el Inspector, sería el que la cámara cartografiadora utilizaría como coordenada Z en lugar del elegido manualmente por el usuario.

[4]La desactivación automática no es posible pues el sistema de desarrollo Unity no permite hacer permanente ningún cambio que se haya realizado en la escena durante la ejecución de ésta, pensado con el objetivo de permitir al desarrollador la realización de pruebas y evitar la aparición de errores.



(a) Copia de H_CamaraCapturadora desde la ventana de proyecto a la de jeraquía



(b) Preparativos de una captura manual

Figura B.3: Preparativos de una captura manual

Realizar Captura Control maestro. Activado permite que la captura automática funcione y, en el modo manual, determina si al cargar la escena se desea realizar una captura. Desactivado, y con el prefab H_CamaraCapturadora habilitado en la ventana de jerarquía, permite utilizar el modo de tanteo al pulsar el botón *Play*, posibilitando de esta forma observar, en tiempo real en la ventana de escena, los límites que tendría la captura para las coordenadas de posición actuales, pudiendo ajustarlos con precisión antes de realizar una captura.

Captura Automática Control que determina si se ha de realizar una captura automática con centro ubicado en la posición predeterminada en la que aparece el avatar en la escena.

Cantidad terreno Tamaño en metros de la mitad del lado del cuadrado de terreno que se capturará. (Si se desea observar en tiempo real el efecto de esta variable durante la gestión en modo manual de la cámara, se podrá modificar el valor *Size* dentro del apartado *Camera* en el Inspector, no obstante, será siempre el valor «Cantidad de terreno» el que determinará el tamaño final del terreno capturado). El hecho de ser el valor correspondiente a la mitad del lado y no a la longitud total es para guardar uniformidad con el valor *Size* del componente *Camera* que, por definición, es la mitad.

Altura Modo Automático Altura desde la que se realizará la captura en el modo automático. Modificar este valor permite salvar obstáculos o realizar capturas de interiores (en el modo manual la altura la ajusta el desarrollador con los controles de movimiento en de la ventana *Scene*).

Calidad Captura Valor entero positivo que determinará la calidad de la captura realizada con el modo de captura estándar ACS (*Application Capture System*). Nótese que valores de calidad mayores a 3 pueden introducir elementos espurios con respecto a la realidad capturada por limitaciones técnicas de este método.

Fichero Captura Nombre del fichero en el que se guardará la captura realizada.

Guardar Coordenadas Control que determinará si se desea guardar un fichero de texto con las coordenadas en las que se deberá crear el plano que contendrá el terreno capturada.

Fichero Coordenadas Nombre del fichero en el que se guardarán las coordenadas donde crear el plano contenedor del terreno capturado.

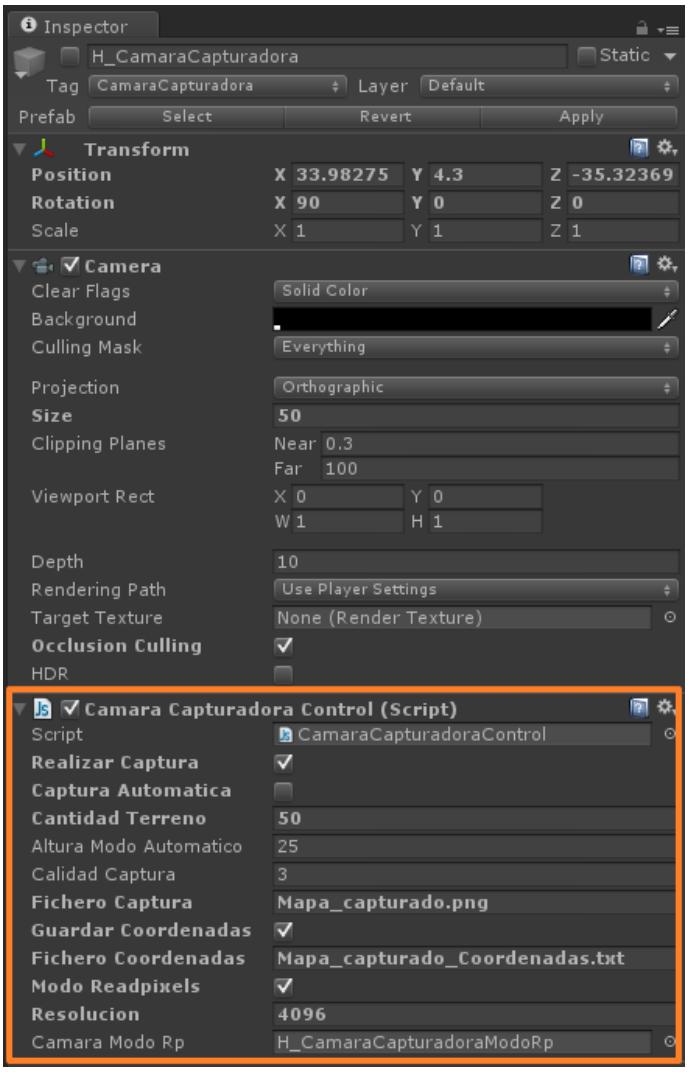


Figura B.4: Personalización del Sistema de Captura

Modo Readpixels Control que determinará si se desea usar el modo avanzado de captura o RP. El resultado de la imagen capturada es siempre mejor que el logrado con el método ACS, como contrapartida es necesaria una versión PRO del programa Unity y los tiempos de captura son mayores.

Resolucion Valor que determinará la calidad de la imagen capturada con el método Readpixels. Valores útiles: 1024, 2048, 4096 (otros valores son posibles pero menos prácticos). Nótese que cuanto mayor sea el terreno a capturar mayor deberá ser la resolución si la calidad de la captura es importante.

Camara Modo RP Variable donde se indicará el prefab encargado de gestionar el método de captura avanzado RP, por defecto H_CamaraCapturadoraModoRP.

B.4. Relación de aspecto de la ventana *Game* al realizar la captura

La imagen del terreno capturado será siempre cuadrada, así, si bien la relación de aspecto de la ventana *Game* puede ser cualquiera si se usa el método de captura ACS, es recomendable que sea cuadrada, y en el caso de usar el método RP es obligatorio.

En el método ACS, si la relación de aspecto de esta ventana no es cuadrada, se obtendrá como resultado de la captura una imagen que contendrá, a la izquierda, un cuadrado con el terreno a capturar deseado, y a la derecha, a mayores, el contenido sobrante de la ventana *Game*. Haciéndolo así será por tanto necesario recortar posteriormente la imagen eliminando el excedente y dejando sólo el terreno cuadrado que se deseaba capturar. Si previamente se cambia la resolución de la ventana *Game* a 1:1 (altura igual a anchura), esta paso extra se hace innecesario.

En el método RP, la relación de aspecto cuadrada es obligatoria, y en caso de que el usuario no la haya cambiado se le avisará con un mensaje de error informándole e invitándole a cambiarla.

Nótese que la relación de aspecto cuadrada no está incluida por defecto entre las que ofrece Unity y hay que añadirla a la lista. Para ello, en la ventana *Game* y arriba a la izquierda, justo debajo de la palabra *Game*, pulsar y desplegar el menú de resoluciones disponibles (aparecerán las palabras *Free Aspect* donde habrá que pulsar para obtener este menú si no se ha cambiado el comportamiento antes). Al final de este listado aparece un símbolo «más» que, pulsando sobre él, nos permitirá añadir resoluciones personalizadas. En el menú de creación de nueva resolución escribiremos como nombre (*Label*) «1:1»; elegiremos como tipo de relación (*Type*), *Aspect Ratio*; y en los campos de anchura y altura (*Width & Height*), indicaremos «1» y «1» respectivamente. Hecho esto quedará registrada la nueva resolución y estará disponible para elegirla cuando se desee.

B.5. Ubicación de los ficheros generados tras la captura

La imagen capturada y el fichero de coordenadas generado (Figura B.5) se almacenarán en una carpeta con el mismo nombre de la escena actual dentro de la carpeta «H_Mapas_capturados», ubicada ésta en el árbol de directorios del sistema operativo, en el mismo lugar donde se encuentre la carpeta «Assets» del proyecto (Figura B.6).

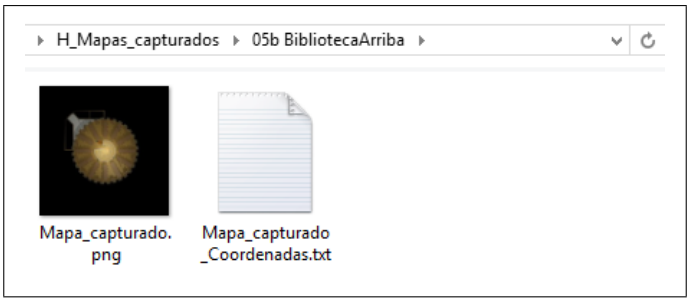


Figura B.5: Elementos obtenidos tras la captura

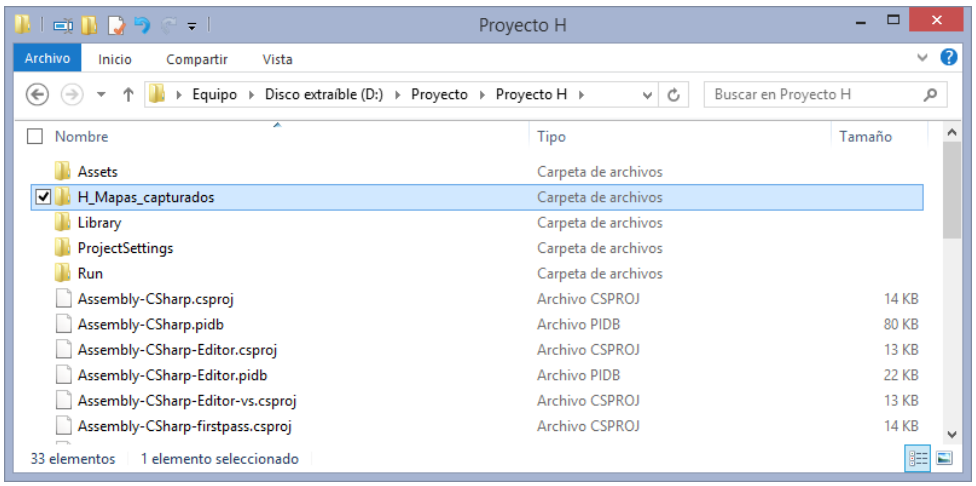


Figura B.6: Ubicación de los ficheros generados

Nótese que la escritura fuera de la carpeta «Assets» tiene asociadas unas consideraciones de seguridad, por esa razón esta tarea sólo es posible cuando la plataforma de compilacion destino del proyecto es diferente a *WebPlayer*. Para comprobar el objetivo de compilación o cambiarlo para la realización de las capturas hay que acudir al menú *File* ▷ *Build Settings...* y elegir una plataforma de compilación diferente a *WebPlayer*. Tras ello, pulsar sobre el botón *Switch Platform* y cerrar la ventana.

B.6. Creación del plano contenedor de la imagen capturada

Tras la captura del terreno deseado se creará en la escena el plano contenedor de la imagen. Para ello se usará la herramienta de creación de planos de alto rendimiento o «planos personalizados H» (tratados con detalle en el Capítulo 8), accesible desde el menú GameObject ▷ Create Other ▷ Plano personalizado H.

Se abrirá una ventana cuyos campos rellenaremos con los indicados en el fichero de coordenadas creado junto a la imagen capturada (Figura B.7).

Script Permite indicar el nombre de un guión que se añadirá automáticamente a los componentes del plano. Lo dejaremos en blanco.

Num Segmentos Ancho Indicador de la cantidad de elementos que compondrán el plano a lo ancho. Indicaremos un valor de «1» para lograr la mayor optimización.

Num Segmentos Alto Indicador de la cantidad de elementos que compondrán el plano a lo alto. Indicaremos un valor de «1» para lograr, una vez más, la mayor optimización.

Ancho Longitud del plano a lo ancho en metros. Indicaremos aquí el valor «Longitud de cada lado» incluido en el fichero de coordenadas.

Alto Longitud del plano a lo alto en metros. Indicaremos también aquí el mismo valor «Longitud de cada lado» incluido en el fichero de coordenadas.

Orientación Valor que determinará si el plano se creará horizontal o verticalmente respecto del plano X-Z. Elegiremos «Horizontal».

Pivote Valor que determinará la posición del pivote que servirá como centro para ubicar el plano posteriormente en la escena. Elegiremos «Centro».

Agregar Collider Opción que permitirá agregar automáticamente al plano un componente de tipo *Collider* inmediatamente tras su creación. Dejaremos la casilla desactivada.

Crear En El Origen Al activar esta casilla el plano se creará inicialmente en las coordenadas (0, 0, 0). La dejaremos activada si bien luego modificaremos la posición del plano.

Etiqueta Mapa H Al activar esta casilla se le añadirá al plano la etiqueta «Mapa». Necesaria en este caso para el correcto desempeño del Sistema de Navegación y, por tanto, la activamos.

Nombre Del Plano Nombre que se le dará al plano una vez creado. Con el fin de conservar la uniformidad le asignaremos un nombre con la siguiente estructura H_Mapa_<nombre de la escena>, por ejemplo H_Mapa_04SalonDeActos. Nótese que en la ventana *Project* al nombre del plano se le agregará, en forma de sufijo, una cadena de caracteres que representarán un resumen de las características físicas de éste, a saber, número de segmentos, anchura y altura, orientación y posición del pivote.

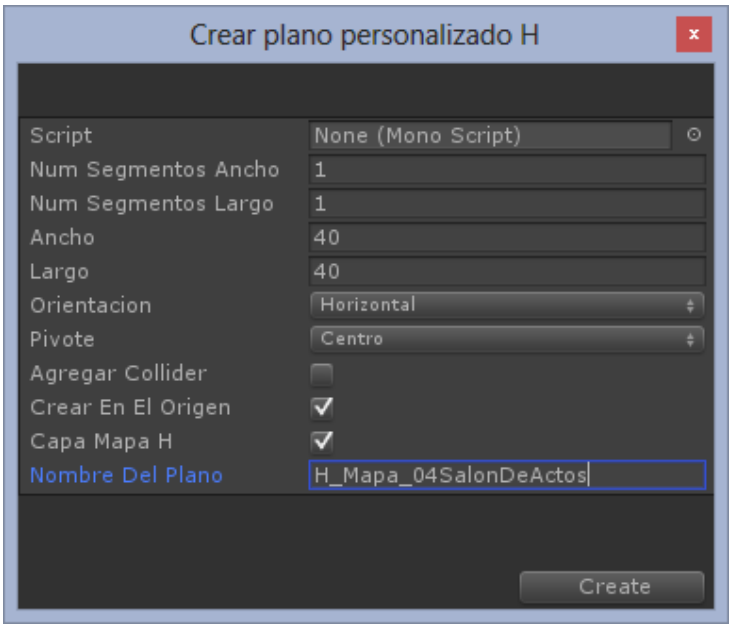


Figura B.7: Ejemplo de creación de un plano personalizado para el Sistema de Captura

```
Coordenadas y dimensiones del plano al que aplicar la textura
-------------------------------------------------------------
Coordenada X:
61.94769
Coordenada Y:
-100
Coordenada Z:
-61.95943
Longitud de cada lado:
40
-------------------------------------------------------------
```

Cuadro B.1: Contenido de ejemplo de un fichero de coordenadas

B.7. Posicionamiento del plano contenedor de la imagen

Una vez creado el plano contenedor de la imagen, este estará ubicado en la escena de forma predeterminada en la posición (0, 0, 0), coordenadas que será necesario sustituir por aquellas incluidas en el fichero de texto que el Sistema de Captura creó junto con el mapa del terreno.

Para modificar la posición del plano contenedor, se modificarán en su Inspector, en el componente *Transform,* los campos X, Y y Z del campo *Position* según la información indicada en el fichero de coordenadas (Cuadro B.1).

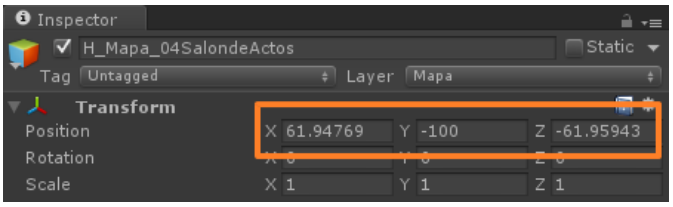


Figura B.8: Modificación del componente *Transform* del plano

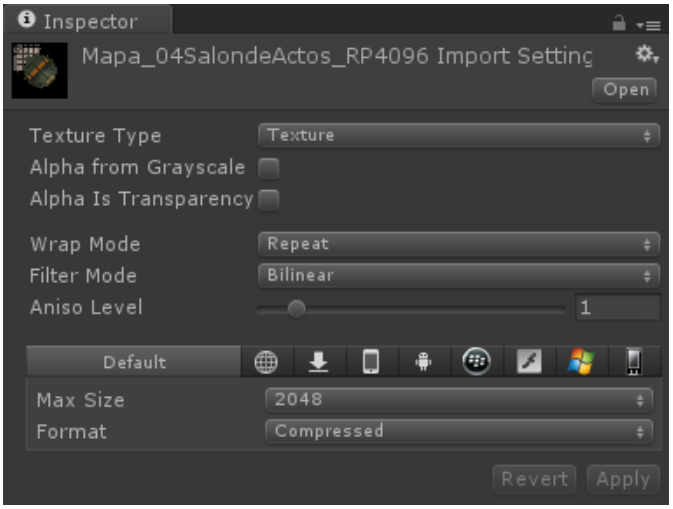


Figura B.9: Configuración de una textura para el Sistema de Captura

B.8. Importación de la imagen capturada dentro del proyecto

Para importar la imagen dentro del proyecto bastará con arrastrarla, desde la carpeta donde esté ubicada en el sistema operativo, a la ventana *Project* en Unity. Este proceso la convertirá en una «textura», denominación con la que en Unity se conoce a las imágenes. Como recomendación, y por mantener una estructura ordenada, la arrastraremos a una carpeta creada expresamente para contener los mapas de las escenas cartografiadas[5]. Tras ello le cambiaremos el nombre por uno de la siguiente forma Mapa_<nombre de la escena>_<método de captura><calidad o resolución de la captura>, por ejemplo, Mapa_04SalonDeActos_RP2048 o Mapa_06cAula03_ACS2.

Convertida la imagen en textura podremos configurar sus características modificando los valores que nos ofrece su Inspector (Figura B.9). Modificar estos valores permitirá lograr una textura final de mayor o menor calidad a costa de aumentar su tamaño en megas dentro del proyecto. Valores a considerar[6]:

[5]El presente proyecto utiliza para este fin la carpeta «Proyecto H/Mapas», pero cualquier otra servirá igualmente, pudiendo personalizar así la agrupación de los mapas según se vayan añadiendo nuevas escenas en el futuro.

[6]Se puede encontrar una explicación detallada de todos los valores personalizables de una textura en http://docs.unity3d.com/Documentation/Components/class-Texture2D.html.

Texture Si cambiamos este valor de *Texture* a *Advanced* podremos entonces activar las casillas relativas a *Generate Mip Maps*, que determinarán si Unity debe crear versiones consecutivas más pequeñas de la textura, algo que busca mejorar el rendimiento a costa de aumentar el consumo en memoria. En este contexto podemos dejar este campo sin modificar, *Texture*.

Max Size Indicaremos el valor que mejor se ajuste a las necesidades de la textura, como recomendación se indicará un valor de 1024, 2048 o 4098 acorde al tamaño del terreno capturado. Un mayor valor proporciona mayor calidad pero también requiere mayor espacio. Nótese que, por lo general, las diferencias visuales entre elegir 2048 y 4096 son apenas notables pero el tamaño aumenta en un factor de cuatro[7]. En la Figura B.10 se puede observar la diferencia de elegir un valor *Max Size* de 1024, a la izquierda, y uno de 4096, a la derecha.

Format La cantidad de elementos mostrados en esta lista dependerá del valor elegido en el campo *Texture* mencionado anteriormente. Influye en la calidad de la imagen final así como en características como su transparencía, pero también influye en el tamaño final de la misma en megas. Puede dejarse en principio sin modificar, *Compressed*.

Pulsaremos finalmente en el botón *Apply* para guardar los cambios realizados (de no hacerlo el sistema nos informará de que hay cambios pendientes de asignar y nos dará la opción de guardarlos o ignorarlos).

B.9. Asignación de la imagen al plano contenedor

Para asignar la textura al plano personalizado H creado en la Sección B.6, se seleccionará primero el plano dentro de la ventana *Hierarchy*, a continuación se localizará la textura recién importada en la ventana *Project* y finalmente se arrastrará sobre el componente *Mesh Renderer* del plano en el Inspector o, también, debajo del botón *Add Component* (Figura B.11). Al soltar se asociará la textura al plano como nuevo componente.

La configuración final pasa por desactivar las casillas relacionadas con la proyec-

[7]Si se desea compilar el proyecto para diferentes plataformas el tamaño de las texturas puede ser un factor importante

Figura B.10: Diferencia entre un valor *Max Size* de 1024 (izqda.) y de 4096 (dcha.)

ción de sombras, *Cast Shadow* y *Receive Shadow*, en el componente *Mesh Renderer* del plano por no ser necesarias en este contexto, y por modificar el campo *Shader*, sustituyendo su valor predeterminado *Diffuse* por *Unlit/Texture,* para ajustar adecuadamente así la iluminación de la textura.

B.10. Ajustes finales

Seguidos estos pasos, el Sistema De Posicionamiento mostrará y gestionará ahora el mapa de la escena durante su ejecución. Como ajustes finales se puede eliminar el objeto H_CamaraCapturadora dentro de la ventana *Hierarchy* si se optó por copiarlo para utilizar el modo manual (el prefab original dentro de la ventana *Project* se conservará siempre), y reajustar la resolución de la ventana *Game* si se modificó para utilizar una cuadrada.

Si hiciera falta repetir el proceso o algún elemento de los introducidos en la estructura del proyecto no fuera a utilizarse, sean las texturas o los planos creados para contenerlas, en aras de la claridad sería recomendable eliminarlos. En este caso, terminado todo el proceso, los elementos a eliminar serían cuatro, repartidos entre la ventana de proyecto y la ventana de jerarquía. En la ventana *Project* se encontrarían tres elementos, a saber: la textura, incluida en la carpeta creada expresamente para contener los mapas; la malla del plano contenedor, dentro de la carpeta «Planos personalizados H»; y por último el elemento *Material*, creado automáticamente al asignarle la textura al plano e incluido en la subcarpeta «Material», dentro de la carpeta contenedora de la textura. Por otro lado, en la ventana *Hierarchy* se encontraría el propio objeto representativo del plano

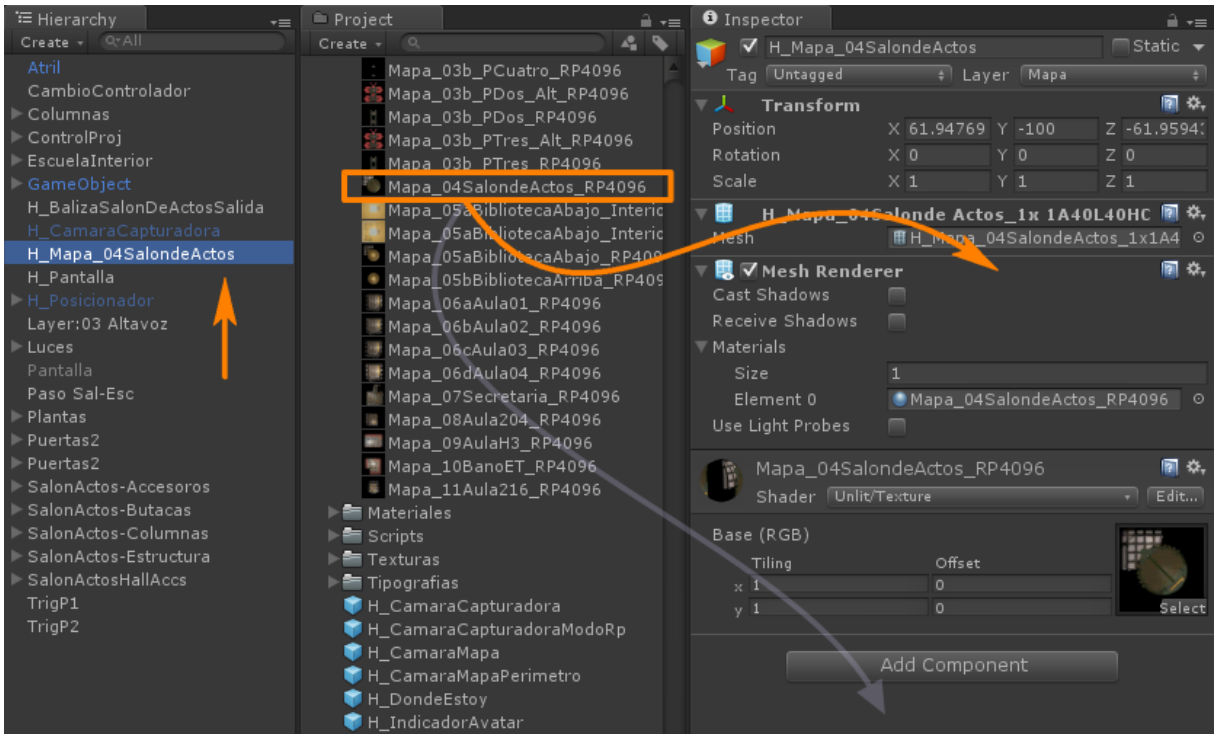


Figura B.11: Asignación de la textura al plano

contenedor. Todos ellos pertenecientes a la escena en la que estemos trabajando y, por tanto, identificables por tener el nombre de esta escena en el suyo propio además de la palabra «Mapa» si se han seguido los consejos de denominación sugeridos con anterioridad, tal y como puede apreciarse en la Figura B.12, en la que los cuatro objetos vinculados aparecen seleccionados .

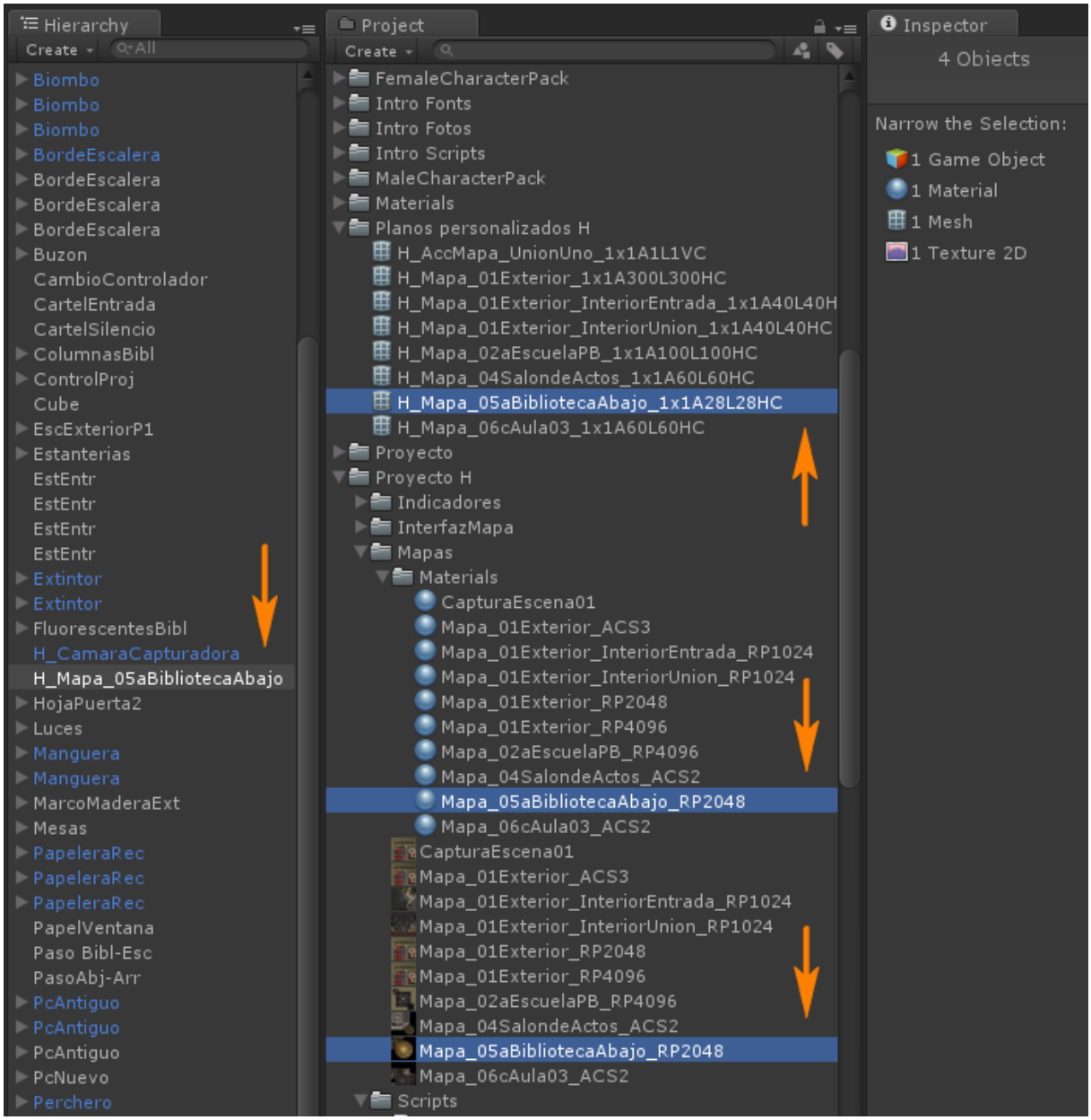


Figura B.12: Elementos creados durante la integración de un mapa en una escena

Apéndice C

Diferencias en el uso del Sistema de Captura para mapas de detalle

Se describe en este anexo las diferencias de uso del Sistema de Captura, con respecto a los pasos recogidos en los anexos anteriores para integrar un mapa general en cualquier proyecto, cuando el mapa a integrar es uno de detalle.

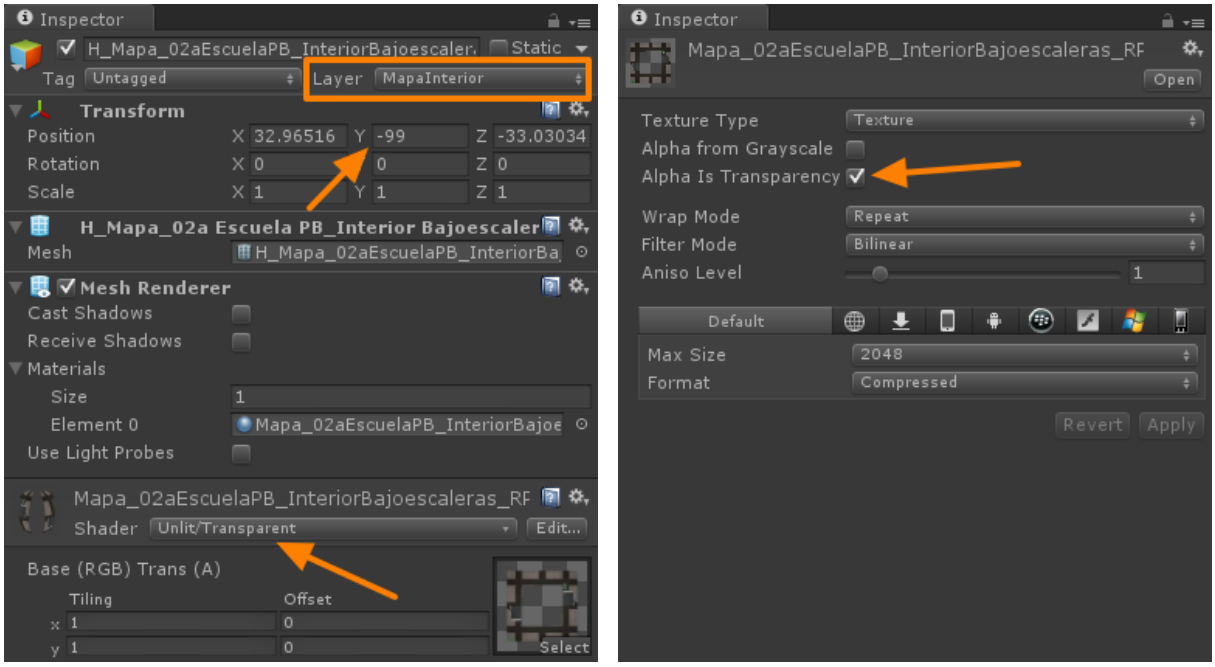
C.0.1. Cambio en la capa a la que pertenece el plano contenedor

La primera de las diferencias es la capa a la que pertenece el plano contenedor del mapa; para un mapa general es, simplemente, «Mapa», para uno de detalle es **«MapaInterior»**.

Este cambio se realizará a través del Inspector del plano, mediante el menú desplegable *Layer*.

C.0.2. Cambio en la localización del plano contenedor

La segunda diferencia es la coordenada Y del plano contenedor; para un mapa general es siempre -100, para un mapa de detalle es **-99**.



(a) Cambios en el Inspector del mapa (b) Cambios en el Inspector de la textura

Figura C.1: Diferencias en la configuración de un mapa de detalle

C.0.3. Cambios en el «Shader» de la textura del mapa

Por último, si previamente se ha modificado el mapa de detalle con un programa de edición fotográfica para añadirle zonas transparentes, habrá que modificar el *Shader* de la textura una vez importado para que refleje estas zonas adecuadamente.

Para ello, en el Inspector del plano contenedor, en su componente *Mesh Renderer* se modificará el campo *Shader*, sustituyendo su valor predeterminado *Diffuse*, o el valor *Unlit/Texture* utilizado para los mapas generales, por el valor ***Unlit/Transparent***.

También, es este caso, habrá que activar la casilla ***Alpha Is Transparency*** en el Inspector de la textura para que las áreas transparentes aparezcan como tales.

Apéndice D

Relación de ubicaciones vigentes

Se recoge a continuación un listado con las ubicaciones vigentes utilizadas por el Sistema de Localización.

- Exterior
- ExteriorUnion
- TecnologicoDespachoFernandoJFF
- EscuelaPlantaBaja
- EscuelaPlantaUno
- TecnologicoPlantaBaja
- TecnologicoPlantaBajaOeste
- TecnologicoPlantaUno
- TecnologicoPlantaUnoOeste
- TecnologicoPlantaUnoRellanoEscaleras
- TecnologicoPlantaUnoRellanoEscalerasOeste
- TecnologicoPlantaDos

- TecnologicoPlantaDosOeste
- TecnologicoPlantaDosRellanoEscaleras
- TecnologicoPlantaDosRellanoEscalerasOeste
- TecnologicoPlantaTres
- TecnologicoPlantaTresOeste
- TecnologicoPlantaCuatroEste
- TecnologicoPlantaCuatroOeste
- TecnologicoDespachoFernandoJFF
- TecnologicoBanoChicasPlantaBaja
- SalonDeActos
- BibliotecaPlantaBaja
- BibliotecaPlantaUno
- AulaUno
- AulaDos
- AulaTres
- AulaCuatro
- Secretaria
- AulaDoscientoscuatro
- AulaHacheTres
- TecnologicoBanoChicasPlantaBaja
- AulaDoscientosdieciseis

Apéndice E

Relación de balizas vigentes

Se recoge a continuación un listado, organizado por escenas, con las balizas vigentes utilizadas por el Sistema de Navegación Guiada.

Áreas con varios accesos

Escena exterior, «01 Exterior»

- H_BalizaEscuelaEntrada
- H_BalizaEscuelaEntradaUnion
- H_BalizaTecnologicoEntrada

Escuela, planta baja, «02a EscuelaPB»

- H_BalizaAulaUno
- H_BalizaAulaDos
- H_BalizaAulaTres
- H_BalizaAulaCuatro
- H_BalizaSecretaria

- H_BalizaSalidaTecnologico
- H_BalizaEscuelaEscaleraSurRellano
- H_BalizaSalonDeActos

Escuela, planta uno, «02b EscuelaP1»

- H_BalizaEscuelaEscaleraNorteRellano
- H_BalizaEscuelaEscaleraSurRellano (la más cercana a la entrada)
- H_BalizaBiblioteca
- H_BalizaAulaDoscientoscuatro
- H_BalizaAulaAulaDoscientosdieciseis

Edificio tecnológico, plantas baja y primera, «03a EdifTecnologicoP1»

- H_BalizaTecnologicoEscaleraEsteAbajo
- H_BalizaTecnologicoEscaleraEsteArriba
- H_BalizaTecnologicoEscaleraEsteRellano
- H_BalizaTecnologicoEscaleraOesteAbajo
- H_BalizaTecnologicoEscaleraOesteArriba
- H_BalizaTecnologicoEscaleraOesteRellano
- H_BalizaTecnologicoSalida
- H_BalizaAulaHacheTres (piso uno)
- H_BalizaTecnologicoBanoChicasPlantaBaja (planta baja)

Edificio tecnológico, plantas dos, tres y cuatro, «03b EdifTecnologicoP2»

- H_BalizaTecnologicoEscaleraEsteAbajo
- H_BalizaTecnologicoEscaleraEsteArriba
- H_BalizaTecnologicoEscaleraEsteRellano
- H_BalizaTecnologicoEscaleraOesteAbajo
- H_BalizaTecnologicoEscaleraOesteArriba
- H_BalizaTecnologicoEscaleraOesteRellano
- H_BalizaTecnologicoDespachoFernandoJFFSalida (piso dos)

Áreas con una sola salida

Biblioteca, planta baja, «05a BibliotecaAbajo»

- H_BalizaBibliotecaSalida

Biblioteca, planta uno, «05b BibliotecaArriba»

- H_BalizaBibliotecaEscaleras

Baño de chicas del edificio tecnológico, planta baja, «10 BanoET»

- H_BalizaTecnologicoBanoChicasPlantaBajaSalida

Despacho de D. Fernando Jorge Fraile Fernández, «TecnologicoDespachoFernandoJFF» (punto de control)

- H_BalizaTecnologicoDespachoFernandoJFFSalida

Salón de actos, «04 SalondeActos»

- H_BalizaSalonDeActosSalida

Secretaría, «07 Secretaria»

- H_BalizaSecretariaSalida

Aula 1, «06a Aula 01»

- H_BalizaAulaUnoSalida

Aula 2, «06b Aula 02»

- H_BalizaAulaDosSalida

Aula 3, «06c Aula 03»

- H_BalizaAulaTresSalida

Aula 4, «06d Aula 04»

- H_BalizaAulaCuatroSalida

Aula 204, «08 Aula 204»

- H_BalizaAulaDoscientoscuatroSalida

Aula 216, «11 Aula 216»

- H_BalizaAulaDoscientosdieciseisSalida

Aula H3, «09 Aula H3»

- H_BalizaAulaHacheTresSalida

Apéndice F

Relación de Guiones o *Scripts* utilizados en el Proyecto

F.1. CamaraCapturadoraControl.js

```
#pragma strict

/*#################################################################

CamaraCapturadoraControl.js
---------------------------

Guión encargado de gestionar la cámara capturadora de mapas.

Define y controla las variables que el usuario puede personalizar mediante
el Inspector del prefab "H_CamaraCapturadora". Gestiona la creación y ubicación
de los ficheros generados tras la captura. Controla la presencia del avatar en
la escena durante la captura y reajusta los componentes de la cámara original tomando
controladamente el relevo. Gestiona también el uso del modo avanzado de captura RP
(ReadPixels) cuando éste se ha activado mediante el Inspector y evalua la plataforma
de compilación presente para ajustarse a las posibles restricciones de esta.

#################################################################*/


import System;
import System.IO;

var realizarCaptura : boolean;
var capturaAutomatica : boolean;

// Definimos en el Inspector la altura desde la que la cámara realizará la captura. En teoría debería ser más
// alta que el objeto más alto de la escena a cartografiar, sin embargo, una altura menor puede servir para
// ignorar techos al capturar interiores sin necesidad de tener que desactivarlos previamente.

var cantidadTerreno : int;
```

```
32 var alturaModoAutomatico : float;
33 var calidadCaptura : int;
34 var ficheroCaptura : String;
35 var GuardarCoordenadas : boolean;
36 var ficheroCoordenadas : String;
37
38 @HideInInspector var directorio : String;
39
40 private var interfazMapa : GameObject;
41
42 private var controlador : GameObject;
43
44 // Variables útiles para el modo de captura alternativo ReadPixels:
45
46 var ModoReadpixels : boolean;
47
48 // Variable a indicar a través del Inspector por la que se especifica la resolución de la
49 // imagen capturada en cantidad de píxeles por cantidad de píxeles (resolución x
       resolución).
50
51 var resolucion : int;
52
53 var camaraModoRp : GameObject;
54 private var camaraModoRpClon : GameObject;
55
56
57 function Start ()
58 {
59   controlador = SDN.avatar;
60
61   if (capturaAutomatica == true)
62   {
63     if (alturaModoAutomatico < -1000 || alturaModoAutomatico > 1000)
64     {
65       Debug.Log("El valor de la variable \"Altura Modo Automatico\" está fuera de rango.
           Se establecerá el valor predeterminado \"50\".");
66       transform.position.y = 50;
67     }
68     else
69     {
70       transform.position.y = alturaModoAutomatico;
71     }
72   }
73   else
74   {
75     // Si la captura es manual, los valores de posición se especifican mediante la
         ubicación manual de la cámara en la escena.
76
77     alturaModoAutomatico = transform.position.y;
78   }
79
80   if (cantidadTerreno <= 0)
81   {
82     Debug.Log("El valor de la variable \"Cantidad Terreno\" está fuera de rango. Se
         establecerá el valor predeterminado \"50\".");
83     camera.orthographicSize = 100;
84   }
85   else
86   {
```

```
87      camera.orthographicSize = cantidadTerreno;
88    }
89
90    // Desactivamos la interfaz de la cámara del mapa (GUI) para no capturar a mayores
          también sus elementos.
91
92    interfazMapa = SDN.interfazMapa;
93
94    if (interfazMapa != null)
95    {
96      interfazMapa.SetActive(false);
97    }
98
99    // La cámara del mapa no haria falta desactivarla por tener un valor "depth" inferior a
          la cámara
100   // capturadora, pero la desactivamos para que, en capturas no cuadradas con el modo ACS
          , no aparezca
101   // el mapa bajo el terreno capturado.
102
103   SDN.camaraMapa.SetActive(false);
104
105   // El siguiente código es dependiente de la plataforma objetivo de ejecución.
106
107   // Las funciones necesarias para gestionar la creación de fichero están, por motivos
108   // de seguridad, restringidas en un entorno 'Web Player', por esta razón limitamos
109   // su ejecución a las plataformas de compilación diferentes a 'Web Player'.
110
111   // Para cambiar la plataforma de compilación y poder usar las funciones de escritura en
          ficheros de
112   // texto para guardar las coordenadas hay que acudir al menú en Unity 'File' -> 'Build
          Settings...'
113
114   #if UNITY_WEBPLAYER
115     Debug.Log("No es posible crear ficheros locales si la plataforma objetivo de
          compilación es 'Web Player'. Para activar esta función, por favor, cambia la
          plataforma de compilación en el menú 'File' -> 'Build Settings...'");
116   #else
117     if (realizarCaptura == true)
118     {
119       // Comprobaciones iniciales sobre las variables públicas.
120       // En caso de no haberse definido éstas se utilizan los valores predefinidos.
121
122       if (calidadCaptura <= 0) { calidadCaptura = 2; }
123
124       if (!ficheroCaptura)
125       {
126         Debug.Log("No se ha definido un nombre para el fichero con la imagen capturada,
              se usará \"Mapa_capturado.png\".");
127         ficheroCaptura = "Mapa_capturado.png";
128       }
129
130       if (!ficheroCoordenadas)
131       {
132         Debug.Log("No se ha definido un nombre para el fichero con las coordenadas de la
              imagen capturada, se usará \"Mapa_capturado_Coordenadas.txt\".");
133         ficheroCoordenadas = "Mapa_capturado_Coordenadas.txt";
134       }
135
136       //----------------if (capturaAutomatica == null) { capturaAutomatica = false; }
```

```
137
138       // Comprobamos que existe el directorio donde guardar los ficheros, si no se crea.
139
140       // Nota: En caso de querer crear directorios o ficheros que aparezcan en la
141       // ventana "Project" dentro de Unity se usará "Application.dataPath" (de esta forma
142       // se crearán dentro del directorio Assets en la jerarquia de ficheros del proyecto
              ).
143
144       // (También se puede optar por utilizar: Application.dataPath + "/../Fichero.png"
145       // y se obtendría el mismo resultado).
146
147       directorio = "H_Mapas_capturados/" + Application.loadedLevelName;
148
149       if (!Directory.Exists(directorio))
150       {
151         Directory.CreateDirectory(directorio);
152       }
153
154       // Nos aseguramos de que la cámara observa el terreno.
155
156       // Modificar este valor, por ejemplo de 90 a 93, puede servir para capturar paredes
              o superficies verticales
157       // que no tienen volumen. Así, inclinando levemente la cámara se les aporta grosor
              artificialmente.
158
159       transform.eulerAngles.x = 90;
160
161       // Definimos un cuadro de visón de la cámara cuadrado.
162
163       camera.rect = Rect(camera.rect.x, camera.rect.y, camera.rect.width / camera.aspect,
              camera.rect.height);
164
165       // Desactivamos el controlador del avatar de la escena para que la cámara no lo
              recoja en la captura.
166
167       controlador = GameObject.FindWithTag("Player");
168       controlador.SetActive(false);
169
170       // Al desactivar el controlador desactivamos también el componente "Audio Listener"
              de su cámara, por lo que
171       // nos aseguramos de que la cámara capturadora disponga de uno (siempre debe haber
              uno activo).
172
173       if (!gameObject.GetComponent(AudioListener))
174       {
175         gameObject.AddComponent(AudioListener);
176       }
177
178       // Realizamos la captura:
179
180       // Se guardará en el directorio raíz del proyecto (en el árbol de ficheros del
              sistema
181       // operativo, no dentro del programa Unity).
182
183       if (ModoReadpixels == true)
184       {
185         if (capturaAutomatica == true)
186         {
```

```
187           camaraModoRpClon = Instantiate(camaraModoRp, controlador.transform.position,
                  camaraModoRp.transform.rotation);
188         }
189         else
190         {
191           // Si la captura no es automática clonamos la camara capturadora en modo RP en
                  el mismo lugar donde se
192           // haya ubicado manualmente la cámara capturadora en la escena.
193
194           camaraModoRpClon = Instantiate(camaraModoRp, transform.position, camaraModoRp.
                  transform.rotation);
195         }
196       }
197       else
198       {
199         Application.CaptureScreenshot(directorio + "/" + ficheroCaptura, calidadCaptura);
200         Debug.Log("Captura realizada.");
201       }
202
203       // Si así se ha indicado guardamos las coordenadas en un fichero de texto para
              facilitar su
204       // utilización.
205
206       if (GuardarCoordenadas == true)
207       {
208         if (File.Exists(ficheroCoordenadas))
209         {
210           Debug.Log("El fichero " + ficheroCoordenadas + " ya existe. Se sobreescribirá."
                  );
211         }
212
213         var coordenadaX : float = transform.position.x;
214         var coordenadaY : float  = transform.position.y;
215         var coordenadaZ : float = transform.position.z;
216         var dimensiones : int = camera.orthographicSize;
217
218         var fd = File.CreateText(directorio + "/" + ficheroCoordenadas);
219
220         fd.WriteLine("Coordenadas y dimensiones del plano al que aplicar la textura");
221         fd.WriteLine("-----------------------------------------------------------------");
222         fd.WriteLine("");
223         fd.WriteLine("Coordenada X:");
224         fd.WriteLine(camera.transform.position.x);
225         fd.WriteLine("");
226         // Indicamos para la coordenada Y un valor de -100 para colocar el plano
                suficientemente
227         // por debajo del resto de los elementos de la escena.
228         fd.WriteLine("Coordenada Y:");
229         fd.WriteLine("-100");
230         fd.WriteLine("");
231         fd.WriteLine("Coordenada Z:");
232         fd.WriteLine(camera.transform.position.z);
233         fd.WriteLine("");
234         fd.WriteLine("Longitud de cada lado:");
235         // Camera.orthographicSize es la mitad de la altura, por tanto la multiplicamos
                por dos para
236         // conocer la dimensión real del lado.
237         fd.WriteLine(camera.orthographicSize * 2);
238         fd.WriteLine("");
```

```
239         fd.WriteLine("-----------------------------------------------------------------");
240         fd.WriteLine("");
241         fd.WriteLine("Para crear el plano ir a:");
242         fd.WriteLine("GameObject -> Create Other -> Plano Personalizado H");
243         fd.WriteLine("");
244         fd.WriteLine("");
245         fd.WriteLine("=================================================================");
246         fd.WriteLine("");
247         fd.WriteLine("");
248         fd.WriteLine("Datos de referencia de la captura");
249         fd.WriteLine("---------------------------------");
250         fd.WriteLine("");
251         fd.WriteLine("Coordenada Y de referencia desde la que se hizo la captura:");
252         fd.WriteLine(camera.transform.position.y);
253         fd.WriteLine("");
254         fd.WriteLine("Valor \"Size\" de referencia con el que se hizo la captura:");
255         fd.WriteLine(camera.orthographicSize);
256 
257         fd.Close();
258       }
259 
260       // Desactivamos la captura automatática tras la primera captura del terreno. De
              esta forma
261       // se obliga a indicar explícitamente (de forma manual en el prefab clonado de la
              ventana
262       // "Hierarchy") el deseo de realizar una nueva captura durante la siguiente vez que
               ejecute
263       // el programa, y se evitan sobreescrituras no deseadas.
264 
265           if (capturaAutomatica == true)
266           {
267             capturaAutomatica = false;
268           }
269           else
270           {
271               if (realizarCaptura == true)
272             {
273               realizarCaptura = false;
274             }
275           }
276     }
277   #endif
278 }
```

F.2. CamaraCapturadoraModoRP.js

```
1  #pragma strict
2 
3  /*###################################################################
4 
5  CamaraCapturadoraModoRP.js
6  --------------------------
7 
8  Método alternativo de captura de imágenes que hace uso de las "render textures" (
       limitadas a la versión
9  Pro de Unity).
10 
```

```
11 El modo normal de capturas mediante "Application.CaptureScreenshot" tiene limitaciones
       que éste método
12 no sufre (por ejemplo, en las capturas realizadas con el método normal indicando una
       calidad superior a 3
13 pueden aparecer incongruencias con la realidad).
14
15 ################################################################*/
16
17
18 import System.IO;
19
20 @HideInInspector var resolucion : int;
21
22 @HideInInspector var directorio : String;
23 @HideInInspector var ficheroCaptura : String;
24
25 private var resolucionAlto : int;
26 private var resolucionAncho : int;
27
28 private var texturaCapturada : Texture2D;
29 private var texturaRender : RenderTexture;
30
31 private var controlador : GameObject;
32 private var camaraCapturadora : GameObject;
33
34 private var tamanoCamara : int;
35
36
37 function Start ()
38 {
39   // Para poder utilizar las "render textures" hace falta una versión Pro de Unity, en
         caso de no disponer de esta
40   // licencia se avisa al usuario y se detiene este procedimiento de captura.
41
42   if ( !Application.HasProLicense() )
43   {
44     Debug.LogError("Este método de captura sólo puede utilizarse en una versión de Unity
         con licencia <b>Pro</b>.");
45     Destroy(gameObject);
46   }
47   else
48   {
49     if ( Screen.height / Screen.width != 1 )
50     {
51       // Nota:
52
53       // Por una limitación de Unity aún no corregida (versión 4.3.1f1), al utilizar una
           "render texture" con un "depth buffer"
54       // activo junto con un cuadro de visión de cámara cuyas dimensiones H/W sean
           inferiores a 1 se produce un error de tipo
55       // "Dimensions of color surface does not match dimensions of depth surface".
56
57       // Al realizar la captura la hacemos con una cámara configurada con un cuadro de
           visión cuadrado por lo que se cumplen las
58       // condiciones para que aparezca el error. Esto sin embargo no impide que la
           captura se realice correctamente pero la
59       // aparición de este error arbitrario en la consola puede puede confundir al
           usuario.
60
```

```
61      // Para evitar esta situación, hasta que una versión posterior de Unity corrija
            este aspecto, obligamos al usuario a cambiar
62      // la relación de aspecto de la ventana "Game" en el editor de Unity a 1:1. De esta
            forma, aun utilizado una cámara de las
63      // características citadas el error no se reproduce.
64
65      // (La relación de aspecto 1:1 no está incluida entre las presentes de forma
            predeterminada, por lo que el usuario deberá
66      // añadirla manualmente).
67
68      Debug.LogError("La resolución de la <b>ventana \"Game\"</b> ha de tener una <b>
            relación de aspecto 1:1</b> para usar este método. Por favor, asigna una
            resolución de este tipo antes de realizar la captura.");
69      Destroy(gameObject);
70    }
71    else
72    {
73      #if UNITY_WEBPLAYER
74        Debug.Log("No es posible crear ficheros locales si la plataforma objetivo de
              compilación es 'Web Player'. Para activar esta función, por favor, cambia la
              plataforma de compilación en el menú 'File' -> 'Build Settings...'");
75      #else
76        // Tomamos el valor de las variables necesarias desde la cámara capturadora
              principal.
77
78        camaraCapturadora = GameObject.FindWithTag("CamaraCapturadora");
79
80        resolucion = camaraCapturadora.GetComponent.<CamaraCapturadoraControl>().
              resolucion;
81        directorio = camaraCapturadora.GetComponent.<CamaraCapturadoraControl>().
              directorio;
82        ficheroCaptura = camaraCapturadora.GetComponent.<CamaraCapturadoraControl>().
              ficheroCaptura;
83
84        // Las dimensiones del terreno a capturar las definimos también en la cámara
              capturadora principal
85        // (variable "Size" del componente "Camera" en el Inspector).
86
87        tamanoCamara = camaraCapturadora.camera.orthographicSize;
88        camera.orthographicSize = tamanoCamara;
89
90
91        if ( resolucion <= 0 || resolucion == null)
92        {
93          Debug.Log("La resolución indicada está fuera de rango, se utilizará el valor
                predeterminado \"1024\".");
94          resolucion = 1024;
95        }
96
97        resolucionAlto = resolucion;
98        resolucionAncho = resolucion;
99
100       // Convertimos el cuadro de visión de la cámara en cuadrado y especificamos la
              altura y rotación de ésta.
101
102       camera.rect = Rect(camera.rect.x, camera.rect.y, camera.rect.width / camera.
              aspect, camera.rect.height);
103       transform.position.y = camaraCapturadora.transform.position.y;
104       transform.eulerAngles.x = 90;
```

```
105
106         // Procedimiento de captura mediante "render texture":
107
108         // 1.- Creación de la "render texture" (textura creada y actualizada durante la
                ejecución del programa).
109         // 2.- Creamos una "Texture2D" de tamaño determinado por la variable resolución
110         // 2.- Le asignamos la "render texture" a la cámara.
111         // 3.- Esperamos a que todo el renderizado se haya completado.
112         // 3.- Definimos la "render texture" como la textura activa (en la que se realiza
                 el renderizado).
113         // 5.- Guardamos la "render texture" (el contenido de la pantalla) en la "
                Texture2D".
114         // 6.- Guardamos el contenido de la textura en la GPU.
115         // 7.- Codificamos la textura en formato PNG y la guardamos como un array de
                bytes
116         // 8.- Guardamos el array de bytes en un fichero (si el fichero existe se
                sobreescribirá).
117
118         texturaRender = new RenderTexture(resolucionAncho, resolucionAlto, 24);
119
120         // (Otro formato para la textura diferente a RGB24 hace que la superficie
                correspondiete al terreno no se capture).
121
122         texturaCapturada = new Texture2D(resolucionAncho, resolucionAlto, TextureFormat.
                RGB24, false);
123
124         camera.targetTexture = texturaRender;
125
126         yield WaitForEndOfFrame();
127
128         // Si el usuario no ha cambiado la relación de aspecto a 1:1 y se permite la
                ejecución del código, la siguiente
129         // línea de código impide que el error "Dimensions of color surface does not
                match dimensions of depth surface"
130         // no se reproduzca indefinidamente.
131
132         // camera.rect = Rect(camera.rect.x, camera.rect.y, resolucionAncho,
                resolucionAlto);
133
134         RenderTexture.active = texturaRender;
135
136         // (Si definiéramos aquí un "Rect" cuadradado en lugar de hacerlo al definir el
                cuadro de visión de la cámara
137         // obtendríamos una textura cuadrada, pero la textura, en lugar de contener un
                recorte cuadrado de la pantalla,
138         // contendría la totalidad de ésta deformada hasta ajustarse a las proporciones
                cuadradas).
139
140         texturaCapturada.ReadPixels(new Rect(0, 0, resolucionAncho, resolucionAlto), 0,
                0, true);
141         texturaCapturada.Apply();
142
143         var bytes = texturaCapturada.EncodeToPNG();
144         Destroy(texturaCapturada);
145
146         File.WriteAllBytes(directorio + "/" + ficheroCaptura, bytes);
147
148         Debug.Log("Captura realizada.");
149     #endif
```

```
150     }
151   }
152 }
```

F.3. CamaraMapaControl.js

```
1  #pragma strict
2
3  /*#####################################################################
4
5  CamaraMapaControl.js
6  --------------------
7
8  Núcleo del proyecto H.
9
10 Guión encargado, entre otras cosas, de fijar los aspectos de la cámara del mapa
11 relacionados con su comportamiento o posición, de clonar los distintos elementos
12 constituyentes del Proyecto H nada más cargar una escena y gestionar su funcionamiento,
13 o de centralizar la gestión de las variables que usarán el resto de elementos y
       asignarlas
14 tras comprobar que sus valores son adecuados.
15
16 #####################################################################*/
17
18
19 // Variable que permite activar o desactivar por completo el Proyecto H.
20
21 var cargarProyectoH : boolean;
22
23 // Variable que determina desde el Inspector si se inicia la escena con el mapa abierto o
24 // minimizado
25
26 var mapaActivado : boolean;
27
28 // Variable que asigna la tecla que minimizará y reabrira el minimapa.
29
30 var teclaCerrarMapa : String;
31
32 var teclaMaximizarMapa : String;
33
34 // Variable que establecerá la cantidad de terreno que muestra el mapa de forma inicial.
35
36 var terrenoVisto : float;
37
38 // Variable que establecerá la cantidad de terreno que muestra el mapa cuando está
       maximizado.
39
40 var terrenoVistoMaximizado : float;
41
42 // Las siguientes variables públicas nos permiten modificar en el Inspector tanto el
       tamaño del
43 // minimapa, como su posición en la pantalla y su separación con respecto a los bordes de
        esta.
44
45 // Para el tamaño y la separación respecto a los bordes se introducirán valores desde 0 a
        100
46 // que indicarán el porcentaje de pantalla que asumirán ambas propiedades (valores fuera
```

```
47 // de ese rango activarán los valores predeterminados).
48
49 // Para la posición en la pantalla hay cuatro posibles valores:
50 //    ArribaIzquierda: coloca el minimapa en la esquina superior izquierda
51 //    ArribaDerecha: coloca el minimapa en la esquina superior derecha
52 //    (Los dos valores siguientes se han anulando/comentado en el código para asegurar
53 //    que los espacios inferiores quedan libren para otros elementos de la interfaz).
54 //    AbajoDerecha: coloca el minimapa en la esquina inferior derecha
55 //    AbajoIzquierda: coloca el minimapa en la esquina inferior izquierda
56 //
57 // Cualquier otro valor activará la posición predeterminada: arriba a la derecha.
58
59 // La medida de pantalla que servirá como referencia para los valores porcentuales es la
       altura.
60
61 var tamanoMapa : float;
62 var margenSeparacion : float;
63
64 // Variable que, si se activa desde el Inspector, le permite al usuario cambiar la
       ubicación
65 // del mapa durante la ejecución del programa.
66
67 var permitirCambiarPosicion : boolean;
68
69 // Variable que determina desde el Inspector la ubicación inicial del mapa (
       ArribaIzquierda,
70 // ArribaDerecha, AbajoDerecha, AbajoIzquierda).
71
72 var posicionMapa : String;
73
74 @HideInInspector var posicionCambiada : boolean;
75
76 // Las siguientes variables determinan si se podrá manejar el zoom del mapa con el
       teclado
77 // y las teclas asignadas a cada control.
78
79 var permitirControlarMapaConTeclado : boolean;
80
81 var teclaAcercarMapa : String;
82 var teclaAlejarMapa : String;
83 var teclaRestablecerMapa : String;
84
85 // Tecla a la que se asignará la función de información "donde estoy".
86
87 var teclaDondeEstoy : String;
88
89 // Variable que, activada en el Inspector, indica el deseo de que al transportarse el
       avatar
90 // mediante el sistema de transporte del SDN se muestre en pantalla el nombre del destino
       .
91
92 //var mostrarNombreDestinoSDN : boolean;
93 var mostrarNombreDestino : boolean;
94
95 // Tiempo durante el que se mostrará el nombre del destino en pantalla al que se haya
96 // transportado el avatar.
97
98 var tiempoMostrarNombreDestino : float;
99
```

```
100 var teclaSelectorDestinos : String;
101
102 // Variable que permitirá que el usuario alterne entre el control de la vista de la
    escena
103 // controlado por el ratón o controlado por el teclado (vista orbital o no).
104
105 var permitirVistaTeclado : boolean;
106
107 var teclaConmutadoraVista : String;
108
109 // Variables para el control de la vista por teclado. Las recoge el guión "VistaTeclado.
    js"
110 // al activar el usuario este modo de vista.
111
112 var suavizadoVistaTeclado : float;
113 var alturaVistaTeclado : float;
114 var distanciaVistaTeclado : float;
115
116 // Variable que permitirá que el usuario cambie de idioma.
117
118 var permitirCambiarIdioma : boolean;
119
120 // Variable que definirá el idioma predeterminado.
121
122 var idiomaPredeterminado : String;
123
124 var teclaSelectorIdiomas : String;
125
126 // Cámara capturadora. Especificada desde el Inspector para habilitar la función de
    captura automática
127 // (por defecto H_CamaraCapturadora).
128
129 var camaraCapturadora : GameObject;
130
131 // Objeto que proporcionará la interfaz del mapa.
132
133 var interfazMapa : GameObject;
134
135 // Objeto que proporcionará la interfaz del selector de destinos del SDN.
136
137 var interfazDestinos : GameObject;
138
139 // Objeto que proporcionará la interfaz del selector de idiomas.
140
141 var interfazIdiomas : GameObject;
142
143 // Objeto que proporcionará la interfaz del selector de apagado.
144
145 var interfazApagado : GameObject;
146
147 // Objeto que proporcionará el indicador del avatar en el mapa.
148
149 var indicadorAvatar : GameObject;
150
151 // Objeto que representará al avatar en términos de colisión de tipo "trigger"
152 // (este objeto será dimensionalmente un punto).
153
154 var puntoAvatar : GameObject;
155
```

```
156 // Objeto encargado de mostrar temporalmente en pantalla el nombre del destino al
157 // que se haya transportado el avatar mediante el sistema de transporte del SDN.
158
159 var presentadorNombreDestino : GameObject;
160
161 // Objeto encargado de indicarle al usuario que la escena se está cargando.
162
163 var presentadorCargandoDestino : GameObject;
164
165 // Objecto que permanecerá cargado en escena para proporcionar aquel código del
166 // SDN que deba estar siempre accesible.
167
168 var objetoPersistente : GameObject;
169
170 // Objeto que servirá de referencia en los bordes del mapa durante la navegación
171 // guiada.
172
173 var localizador : GameObject;
174
175 // Objeto que determinará la ubicación del avatar para el sistema de localización.
176
177 var dondeEstoy : GameObject;
178
179 // Objeto que creará la superficie a interceptar por el rayo del sistema de navegación
180 // guiada para conocer la posición donde colocar los localizadores.
181
182 var camaraMapaPerimetro : GameObject;
183
184 // Localizadores para el sistema de navegación guiada. Se clonarán, se guardarán sus
185 // referencias para informar al objeto persistenta y tras ello se desactivarán.
186
187 var localizadorNormal : GameObject;
188 @HideInInspector var localizadorNormalClon : GameObject;
189 var localizadorSubir : GameObject;
190 @HideInInspector var localizadorSubirClon : GameObject;
191 var localizadorBajar : GameObject;
192 @HideInInspector var localizadorBajarClon : GameObject;
193
194 // Variables privadas para la gestión interna del guión.
195
196 private var indicadorAvatarClon : GameObject;
197 private var puntoAvatarClon : GameObject;
198
199 private var anchuraPantalla : float;
200 private var alturaPantalla : float;
201 private var relacionDeAspecto : float;
202
203 private var margenX : float;
204 private var margenY : float;
205 private var margenXNormalizado : float;
206 private var margenYNormalizado : float;
207
208 @HideInInspector var alturaCamara : float;
209 @HideInInspector var anchuraCamara : float;
210 private var alturaCamaraNormalizada : float;
211 private var anchuraCamaraNormalizada : float;
212
213 private var posicionVigente : Rect;
214
```

```
215 private var camaraCapturadoraClon : GameObject;
216 private var interfazMapaClon : GameObject;
217 private var interfazDestinosClon : GameObject;
218 private var centro : GameObject;
219
220 @HideInInspector var pulsadoConmutadorMaximizar : boolean;
221
222
223 function Awake ()
224 {
225
226   // Si no se desea cargar el Proyecto H basta con indicarlo así en el Inspector y se
        cargará
227   // exclusivamente el proyecto legado, sólo permanecerán las mejoras realizadas en los
        elementos
228   // ya existentes con anterioridad.
229
230   if (cargarProyectoH == false)
231   {
232     Destroy(gameObject);
233   }
234   else
235   {
236     // ------------------
237     // OBJETO PERSISTENTE
238     // ------------------
239
240     // Clonamos en la escena el objecto que permanecerá siempre activo para gestionar el
          código
241     // que deba estar en todo momento disponible y configuramos sus variables según las
          elegidas
242     // en el Inspector.
243
244     if (objetoPersistente == null)
245     {
246       Debug.LogError("Se debe especificar el <i>prefab</i> \"Objeto Persistente\" para el
            Sistema de Navegación.");
247     }
248     else
249     {
250       SDN.persistente = Instantiate(objetoPersistente);
251     }
252
253     if (presentadorNombreDestino == null)
254     {
255       Debug.LogError("No se ha especificado el <i>prefab</i> \"Presentador Nombre
            Destinos\" para el Sistema de Navegación.");
256     }
257     else
258     {
259       SDN.persistente.GetComponent.<PersistenteControl>().presentadorNombreDestino =
            presentadorNombreDestino;
260     }
261
262     if (presentadorCargandoDestino == null)
263     {
264       Debug.LogError("No se ha especificado el <i>prefab</i> \"Presentador Escena
            Cargando\" para el Sistema de Navegación.");
265     }
```

```
266     else
267     {
268       SDN.persistente.GetComponent.<PersistenteControl>().presentadorCargandoDestino =
              presentadorCargandoDestino;
269     }
270
271     if (teclaCerrarMapa == "")
272     {
273       Debug.Log("No se ha especificado la tecla para minimizar y reabrir el mapa, se
              usará el valor predeterminado \"m\".");
274
275       teclaCerrarMapa = "m";
276     }
277
278     if (teclaMaximizarMapa == "")
279     {
280       Debug.Log("No se ha especificado la tecla para maximizar el mapa, se usará el valor
               predeterminado \"p\".");
281
282       teclaMaximizarMapa = "p";
283     }
284
285     if (teclaConmutadoraVista == "")
286     {
287       Debug.Log("No se ha especificado la tecla para cambiar entre los diferentes modos
              de vista, se usará el valor predeterminado \"k\".");
288
289       teclaConmutadoraVista = "k";
290     }
291
292     if (teclaConmutadoraVista == "")
293     {
294       Debug.Log("No se ha especificado la tecla para activar la información \"donde estoy
              \", se usará el valor predeterminado \"i\".");
295
296       teclaConmutadoraVista = "i";
297     }
298
299     if (teclaSelectorDestinos == "")
300     {
301       Debug.Log("No se ha especificado la tecla para activar el \"selector de destinos\",
               se usará el valor predeterminado \"t\".");
302
303       teclaSelectorDestinos = "t";
304     }
305
306     if (teclaSelectorIdiomas == "")
307     {
308       Debug.Log("No se ha especificado la tecla para activar el \"selector de idiomas\",
              se usará el valor predeterminado \"l\".");
309
310       teclaSelectorIdiomas = "l";
311     }
312
313     if (suavizadoVistaTeclado < 1 || suavizadoVistaTeclado > 1000)
314     {
315       Debug.Log("El valor de suavizado está fuera de rango. Se usará el valor
              predeterminado \"6\".");
316       suavizadoVistaTeclado = 6;
```

```
317     }
318
319     if (alturaVistaTeclado < 0 || alturaVistaTeclado > 2.4)
320     {
321       Debug.Log("El valor de suavizado está fuera de rango. Se usará el valor
              predeterminado \"1.5\".");
322       alturaVistaTeclado = 1.5;
323     }
324
325     if (distanciaVistaTeclado < 0 || distanciaVistaTeclado > 5)
326     {
327       Debug.Log("El valor de suavizado está fuera de rango. Se usará el valor
              predeterminado \"2\".");
328       distanciaVistaTeclado = 2;
329     }
330
331     if (teclaAcercarMapa == "")
332     {
333       Debug.Log("No se ha especificado la tecla para la función de \"acercar la vista del
              mapa\", se usará el valor predeterminado \"1\".");
334
335       teclaAcercarMapa = "1";
336     }
337
338     if (teclaAlejarMapa == "")
339     {
340       Debug.Log("No se ha especificado la tecla para la función de \"acercar la vista del
              mapa\", se usará el valor predeterminado \"3\".");
341
342       teclaAlejarMapa = "3";
343     }
344
345     if (teclaRestablecerMapa == "")
346     {
347       Debug.Log("No se ha especificado la tecla para la función de \"restablecer la vista
              del mapa\", se usará el valor predeterminado \"2\".");
348
349       teclaRestablecerMapa = "2";
350     }
351
352     SDN.persistente.GetComponent.<PersistenteControl>().teclaDondeEstoy = teclaDondeEstoy
          ;
353
354     SDN.persistente.GetComponent.<PersistenteControl>().teclaConmutadoraVista =
          teclaConmutadoraVista;
355
356     SDN.persistente.GetComponent.<PersistenteControl>().teclaCerrarMapa = teclaCerrarMapa
          ;
357
358     SDN.persistente.GetComponent.<PersistenteControl>().teclaMaximizarMapa =
          teclaMaximizarMapa;
359
360     SDN.persistente.GetComponent.<PersistenteControl>().permitirVistaTeclado =
          permitirVistaTeclado;
361
362     SDN.persistente.GetComponent.<PersistenteControl>().mostrarNombreDestino =
          mostrarNombreDestino;
363
```

```
364     SDN.persistente.GetComponent.<PersistenteControl>().teclaSelectorDestinos =
            teclaSelectorDestinos;
365
366     SDN.persistente.GetComponent.<PersistenteControl>().teclaSelectorIdiomas =
            teclaSelectorIdiomas;
367
368     SDN.persistente.GetComponent.<PersistenteControl>().permitirControlarMapaConTeclado =
             permitirControlarMapaConTeclado;
369
370     SDN.persistente.GetComponent.<PersistenteControl>().teclaAcercarMapa =
            teclaAcercarMapa;
371
372     SDN.persistente.GetComponent.<PersistenteControl>().teclaAlejarMapa = teclaAlejarMapa
            ;
373
374     SDN.persistente.GetComponent.<PersistenteControl>().teclaRestablecerMapa =
            teclaRestablecerMapa;
375
376
377     // ---------------------
378     // INDICADOR DE POSICIÓN
379     // ---------------------
380
381     // Cargamos en la escena un clon del indicador de posición del controlador.
382
383     indicadorAvatarClon = Instantiate(indicadorAvatar);
384
385     // Creamos una referencia global del indicador.
386
387     SDN.indicadorAvatar = indicadorAvatarClon;
388
389
390     // ----------------------------------------------------
391     // CONCENTRADOR DE MASA (gestión precisa de colisiones)
392     // ----------------------------------------------------
393
394     // Cargamos en la escena un clon del objeto "H_PuntoAvatar".
395
396     // Este objeto tiene las dimensiones de una esfera, y lo usaremos para un cálculo
            preciso de las intersecciones del
397     // avatar controlador con superficies que tengan el valor "Is Trigger" activado.
398
399     puntoAvatarClon = Instantiate(puntoAvatar);
400
401
402     // -----------------
403     // INTERFAZ DEL MAPA
404     // -----------------
405
406     // Clonamos la interfaz del mapa.
407
408     interfazMapaClon = Instantiate(interfazMapa);
409
410     // Guardamos una referencia a la interfaz del minimapa para futuros accesos locales.
411
412     interfazMapa = interfazMapaClon;
413
414     // Guardamos una referencia global de la interfas del minimapa.
415
```

```
416     SDN.interfazMapa = interfazMapaClon;
417
418     // Si se ha activado la casilla "Mapa Activado" en el Inspector se informa al guión
419     // "InterfazMapaControl.js" de que ha de minimizar el mapa y cambiar la interfaz para
420     // reflejar este estado.
421
422     // La mencionada casilla definirá el aspecto del mapa sólo para la primera escena,
            cuando
423     // el juego acaba de cargar (estado en el que la variable "SDN.mapaCerrado" aún tiene
            valor
424     // vacío, ""), en las escenas posteriores el estado inicial del mapa, abierto o
            cerrado, quedará
425     // determinado por el estado del mapa en la escena anterior.
426
427     if (SDN.mapaAbiertoCerrado == "")
428     {
429       if (mapaActivado == false)
430       {
431         SDN.mapaAbiertoCerrado = "Cerrado";
432       }
433       else
434       {
435         SDN.mapaAbiertoCerrado = "Abierto";
436       }
437     }
438
439     // Guardamos una referencia global de la cantidad inicial de mapa que mostrara la
            camara del mapa, tanto
440     // en modo normal como en modo reducido, a partir de los valores indicados en el
            Inspector
441
442     // Nota: hacemos esto solamente cuando no se haya establecido un valor para estas
            variables globales con
443     // anterioridad, es decir, sólo al cargar la primera escena; a partir de ahí las
            cantidades de terreno a
444     // mostrar las gestionarán los interfaces (normal y maximizado).
445
446     if (terrenoVisto <= 0)
447     {
448       Debug.Log("El valor introducido para la cantidad de terreno mostrado en el mapa
              está fuera de rango, se fijará el valor predeterminado \"5\".");
449       terrenoVisto = 5;
450     }
451
452     if (SDN.terrenoVisto == 0)
453     {
454       SDN.terrenoVisto = terrenoVisto;
455     }
456
457     // Comprobamos que el valor indicado para la cantidad de terreno visto con el mapa
            maximizado es correcto.
458
459     if (terrenoVisto <= 0)
460     {
461       Debug.Log("El valor introducido para la cantidad de terreno mostrado en el mapa
              maximizado está fuera de rango, se fijará el valor predeterminado \"12\".");
462       terrenoVistoMaximizado = 12;
463     }
464
```

```
465     if (SDN.terrenoVistoMaximizado == 0)
466     {
467       SDN.terrenoVistoMaximizado = terrenoVistoMaximizado;
468     }
469
470
471     // ---------------------------------------
472     // GESTIÓN DEL ESTILO DE CONTROL DE VISTA
473     // ---------------------------------------
474
475     // El siguiente bloque desactiva la vista orbital (controlada con el ratón) de la
            cámara principal nada más
476     // cargarse la escena cuando se haya especificado globalmente que se desea la vista
            controlada con el teclado
477     // (asi se mantiene el mismo estilo de vista entre escenas).
478
479     if (SDN.vistaOrbital == false)
480     {
481       // Activamos la siguiente variable para que, en el modo de vista teclado, la cámara
              se posicione
482       // inmediatamente tras el avatar sin movimiento suavizado siempre que se cargue una
              nueva escena,
483       // haciendo más natural al usuario el posicionamiento inicial de la cámara.
484
485       SDN.vistaOrbitalNoAlPrincipioEscena = true;
486
487       SDN.DesactivarVistaOrbital();
488     }
489
490
491     // ----------------------------------------------------
492     // SELECTOR DE DESTINOS (interfaz y nombre de destino)
493     // ----------------------------------------------------
494
495     // Clonamos desactivada la interfaz del selector de destinos del SDN y la
            referenciamos para poder localizarla
496     // posteriormente (nota: las llamadas de tipo "find" no encuentran objetos
            desactivados).
497
498     SDN.interfazDestinos = Instantiate(interfazDestinos);
499     SDN.interfazDestinos.SetActive(false);
500
501
502     // ------------------
503     // SELECTOR DE IDIOMA
504     // ------------------
505
506     // Clonamos desactivada la interfaz del selector de idomas y la referenciamos para
            poder localizarla
507     // posteriormente (nota: las llamadas de tipo "find" no encuentran objetos
            desactivados).
508
509     SDN.interfazIdiomas = Instantiate(interfazIdiomas);
510     SDN.interfazIdiomas.SetActive(false);
511
512
513     // --------------------
514     // SELECTOR DE APAGADO
515     // --------------------
```

```
516
517     // Clonamos desactivada la interfaz del selector de apagdo y la referenciamos también
            para poder localizarla
518     // posteriormente.
519
520     SDN.interfazApagado = Instantiate(interfazApagado);
521     SDN.interfazApagado.SetActive(false);
522
523
524     // ----------------------------------
525     // SISTEMA DE AYUDA A LA LOCALIZACIÓN
526     // ----------------------------------
527
528     // Clonamos en la escena el objecto encargado de establecer la ubicación del avatar
            según la escena
529     // o la subárea en la que se encuentre.
530
531     Instantiate(dondeEstoy);
532
533     // Clonamos en la escena el objecto encargado de crear la superficie a interceptar
            por el rayo que
534     // determinará la posición de los localizadores.
535
536     Instantiate(camaraMapaPerimetro);
537
538
539     // ----------------------------
540     // SISTEMA DE NAVEGACIÓN GUIADA
541     // ----------------------------
542
543     // El guión maestro (este) se encargará de clonar en la escena los localizadores que
            usará el sistema
544     // de navegación guiada para indicar en el mapa la dirección que debe seguir el
            avatar. Guardará las
545     // referencias y se las transmitirá al guión persistente, luego desactivará los
            localizadores. (El
546     // intersectador "IntersectadorCamaraMapaPerimetro.js" será el encargado de gestionar
            su activación
547     // y desactivación posterior).
548
549     localizadorNormalClon = Instantiate(localizadorNormal);
550     localizadorSubirClon = Instantiate(localizadorSubir);
551     localizadorBajarClon = Instantiate(localizadorBajar);
552
553     SDN.persistente.GetComponent.<PersistenteControl>().localizadorNormalClon =
            localizadorNormalClon;
554     SDN.persistente.GetComponent.<PersistenteControl>().localizadorSubirClon =
            localizadorSubirClon;
555     SDN.persistente.GetComponent.<PersistenteControl>().localizadorBajarClon =
            localizadorBajarClon;
556
557     localizadorNormalClon.SetActive(false);
558     localizadorSubirClon.SetActive(false);
559     localizadorBajarClon.SetActive(false);
560   }
561
562 }
563
564
```

```
565 function OnEnable ()
566 {
567   // Gestión del tamaño, posición y cantidad de terreno visto del mapa cuando este se
        reactiva
568   // en una misma escena después de haberlo desactivado anteriormente (bloque de código
        ejecutado
569   // cada vez que se activa el mapa).
570
571   // Nota: el ciclo "Start()" será el encargado de ajustar estos parámetros en los
        cambios de
572   // escena (código ejecutado una sóla vez al cargar la escena).
573
574   if (SDN.mapaGrande == true)
575   {
576     camera.rect = Rect(0, 0, 1, 1);
577
578     camera.orthographicSize = SDN.terrenoVistoMaximizado;
579
580     interfazMapaClon.GetComponent.<InterfazMapaControl>().enabled = false;
581     interfazMapaClon.GetComponent.<InterfazMapaGrandeControl>().enabled = true;
582   }
583   else
584   {
585     camera.rect = posicionVigente;
586
587     camera.orthographicSize = SDN.terrenoVisto;
588
589     interfazMapaClon.GetComponent.<InterfazMapaControl>().enabled = true;
590     interfazMapaClon.GetComponent.<InterfazMapaGrandeControl>().enabled = false;
591   }
592 }
593
594
595 function Start ()
596 {
597   // Referenciamos el avatar para accesos posteriores.
598
599   centro = SDN.avatar;
600
601
602   // ---------------------------------------------------------------------
603   // CÁMARA DEL MAPA Y MINIMAPA (comportamiento, posición y terreno visto)
604   // ---------------------------------------------------------------------
605
606   // Definimos inicialmente la variable que determinará los cambios entre mapa normal y
        mapa maximizado
607   // a valor desactivado.
608
609   pulsadoConmutadorMaximizar = false;
610
611   // Los siguientes bloques "if" comprueban los datos introducidos en las variables
        públicas en el Inspector.
612   // Si los valores están fuera de rango se utilizan los valores predeterminados y se
        avisa al usuario de ello.
613
614   if (tamanoMapa < 0 || tamanoMapa > 100)
615   {
616     Debug.Log("Tamaño del cuadro de cámara fuera de rango, establecido el valor por
          defecto.");
```

```
617     alturaCamara = 40;
618     anchuraCamara = 40;
619   }
620   else
621   {
622     alturaCamara = tamanoMapa;
623     anchuraCamara = tamanoMapa;
624   }
625
626   if (margenSeparacion < 0 || margenSeparacion > 100)
627   {
628       Debug.Log("Margen de separación fuera de rango, establecido el valor por defecto.")
            ;
629     margenY = 3;
630   }
631   else
632   {
633     margenY = margenSeparacion;
634   }
635
636   alturaPantalla = Screen.height;
637   anchuraPantalla = Screen.width;
638
639   // El cálculo de la relación de aspecto de la pantallá (anchura entre altura) nos
        permitirá operar para
640   // compensar los valores dispares que Unity asigna a las dimensiones del cuadro de
        visión de la cámara
641   // del minimapa.
642
643   relacionDeAspecto = anchuraPantalla / alturaPantalla;
644
645   // Normalizamos los valores porcentuales introducidos desde el Inspector para
        transformarlos en cifras
646   // comprendidas entre 0 y 1.
647
648   alturaCamaraNormalizada = alturaCamara / 100;
649   anchuraCamaraNormalizada = alturaCamaraNormalizada / relacionDeAspecto;
650
651   margenYNormalizado = margenY / 100;
652   margenXNormalizado = margenYNormalizado / relacionDeAspecto;
653
654   // Establecemos el tamaño y posición del cuadro de visión de la camara del minimapa.
655
656   // Si se ha definido ya la posición con anterioridad se utilizará ese valor.
657
658   if (SDN.posicionMapa != "")
659   {
660     posicionMapa = SDN.posicionMapa;
661   }
662
663   switch (posicionMapa)
664   {
665     case "ArribaIzquierda":
666       // Arriba a la izquierda
667       camera.rect = Rect(margenXNormalizado, 1 - alturaCamaraNormalizada -
            margenYNormalizado, anchuraCamaraNormalizada, alturaCamaraNormalizada);
668       SDN.posicionMapa = posicionMapa;
669       break;
670
```

```
671     // Las posiciones "abajo a la izquierda" y "abajo a la derecha" se incluyen como
            posibles pero se desactivan por ocupar otros
672     // elementos de la interfaz estos espacios.
673
674     /*
675     case "AbajoDerecha":
676       // Abajo a la derecha
677       camera.rect = Rect(1 - anchuraCamaraNormalizada - margenXNormalizado,
              margenYNormalizado, anchuraCamaraNormalizada, alturaCamaraNormalizada);
678       SDN.posicionMapa = posicionMapa;
679       break;
680
681     case "AbajoIzquierda":
682       // Abajo a la izquierda
683       camera.rect = Rect(margenXNormalizado, margenYNormalizado, anchuraCamaraNormalizada
              , alturaCamaraNormalizada);
684       SDN.posicionMapa = posicionMapa;
685       break;
686     */
687
688     default:
689       // Arriba a la derecha
690       camera.rect = Rect(1 - anchuraCamaraNormalizada - margenXNormalizado, 1 -
              alturaCamaraNormalizada - margenYNormalizado, anchuraCamaraNormalizada,
              alturaCamaraNormalizada);
691       SDN.posicionMapa = posicionMapa;
692   }
693
694   posicionCambiada = false;
695
696   // Almacenamos las coordenadas del cuadro de visión en una variable para referencias
          posteriores.
697
698   posicionVigente = camera.rect;
699
700   // CANTIDAD DE TERRENO MOSTRADO Y GESTIÓN DEL MAPA ENTRE ESCENAS
701
702   // Si el mapa está abierto al cargar la escena definimos tanto el tamaño del mapa como
          la cantidad de
703   // terreno visto.
704   if (SDN.mapaAbiertoCerrado == "Abierto")
705   {
706     if (SDN.mapaGrande == true)
707     {
708       camera.rect = Rect(0, 0, 1, 1);
709
710       camera.orthographicSize = SDN.terrenoVistoMaximizado;
711
712       interfazMapaClon.GetComponent.<InterfazMapaControl>().enabled = false;
713       interfazMapaClon.GetComponent.<InterfazMapaGrandeControl>().enabled = true;
714     }
715     else
716     {
717       camera.rect = posicionVigente;
718
719       camera.orthographicSize = SDN.terrenoVisto;
720
721       interfazMapaClon.GetComponent.<InterfazMapaControl>().enabled = true;
722       interfazMapaClon.GetComponent.<InterfazMapaGrandeControl>().enabled = false;
```

```
723       }
724     }
725     // Si el mapa no debe estar abierto definimos la cantidad de terreno visto y lo
            cerramos.
726     else
727     {
728       if (SDN.mapaGrande == true)
729       {
730         camera.orthographicSize = SDN.terrenoVistoMaximizado;
731
732         // Al cambiar entre escenas con el mapa maximizado pero cerrado hay que indicar al
                cargar
733         // la nueva escena que el interfaz del mapa a cargar es el correspondiente al mapa
                normal
734         // (el que incluye el botón de abrir el mapa).
735         interfazMapaClon.GetComponent.<InterfazMapaControl>().enabled = true;
736         interfazMapaClon.GetComponent.<InterfazMapaGrandeControl>().enabled = false;
737       }
738       else
739       {
740         camera.orthographicSize = SDN.terrenoVisto;
741       }
742
743       gameObject.SetActive(false);
744     }
745
746
747     // --------------------
748     // CAPTURAS AUTOMÁTICAS
749     // --------------------
750
751     // El siguiente código gestionará las capturas automática del terreno de la escena (
            para la creación del plano que
752     // mostrará la cámara del mapa).
753
754     // Se puede activar esta función indicándolo así en la casilla correspondiente del
            prefab "H_CamaraCapturadora".
755     // Una vez hecho se creará en la ventana "Hierarchy" de la escena un clon de la cámara
            capturadora que tomará
756     // una imagen del terreno según las carácterísticas que se le hayan indicado.
757
758     if ( camaraCapturadora.GetComponent.<CamaraCapturadoraControl>().capturaAutomatica ==
            true )
759     {
760       camaraCapturadoraClon = Instantiate(camaraCapturadora, centro.transform.position,
              camaraCapturadora.transform.rotation);
761
762       // Nos aseguramos de que tras realizar la captura la función de captura automática
              queda desactivada; de esta forma
763       // se obliga al usuario a indicar explícitamente el deseo de activar la cámara
              capturadora cada vez, evitando así
764       // la posibilidad de sobreescribir accidentalmente alguna captura anterior.
765
766       camaraCapturadora.GetComponent.<CamaraCapturadoraControl>().capturaAutomatica = false
              ;
767
768     }
769
770
```

```
771   // ---------------------------------
772   // GESTIÓN DEL IDIOMA PREDETERMINADO
773   // ---------------------------------
774
775   // Definimos el idioma según el elegido como predeterminado sólo si el idioma no se ha
          elegido ya.
776   if (SDN.idioma == "" || SDN.idioma == null)
777   {
778     switch (idiomaPredeterminado)
779     {
780       case "Alemán":
781       case "alemán":
782       case "Aleman":
783       case "aleman":
784         SDN.idioma = "aleman";
785         break;
786
787       case "Inglés":
788       case "inglés":
789       case "Ingles":
790       case "ingles":
791         SDN.idioma = "ingles";
792         break;
793
794       case "Español":
795       case "español":
796       case "Espanol":
797       case "espanol":
798         SDN.idioma = "espanol";
799         break;
800
801       default:
802           Debug.Log("El idioma indicado no es válido, se establecerá el español como
                predeterminado.");
803         SDN.idioma = "espanol";
804     }
805   }
806 }
807
808
809 // Cuando se cierre el mapa, guardamos una referencia de la cantidad de terreno visto.
810
811 function OnDisable ()
812 {
813   if (SDN.mapaGrande == false)
814   {
815     SDN.terrenoVisto = camera.orthographicSize;
816   }
817   else
818   {
819     SDN.terrenoVistoMaximizado = camera.orthographicSize;
820   }
821 }
822
823
824 function Update ()
825 {
826
827   // POSICIÓN DEL MAPA
```

```
828
829   // Si las proporciones de la pantalla han cambiado, volvemos a calcular la posición y
          dimensión del cuadro de la cámara
830   // con respecto a estas nuevas proporciones, asegurando así la continua coherencia
          entre las dimensiones sea cual sea
831   // el tamaño de la pantalla.
832
833   // Estos calculos son para la reubicacion del minimapa por lo que, si estamos en modo a
          pantalla completa, no los
834   // realizaremos hasta salir de él pues, en caso de no tener en cuenta esto,
          tranformaríamos el mapa a pantalla completa
835   // en minimapa sin deseo consciente del usuario.
836
837   if (SDN.mapaGrande == false)
838   {
839     if (Screen.width != anchuraPantalla || Screen.height != alturaPantalla ||
            posicionCambiada == true)
840     {
841       alturaPantalla = Screen.height;
842       anchuraPantalla = Screen.width;
843
844       // El cálculo de la relación de aspecto de la pantallá (anchura entre altura) nos
              permitirá operar para
845       // compensar los valores dispares que Unity asigna a las dimensiones del cuadro de
              visión de la cámara
846       // del minimapa.
847
848       relacionDeAspecto = anchuraPantalla / alturaPantalla;
849
850       // Normalizamos los valores porcentuales introducidos desde el Inspector para
              transformarlos en cifras
851       // comprendidas entre 0 y 1.
852
853       alturaCamaraNormalizada = alturaCamara / 100;
854       anchuraCamaraNormalizada = alturaCamaraNormalizada / relacionDeAspecto;
855
856       margenYNormalizado = margenY / 100;
857       margenXNormalizado = margenYNormalizado / relacionDeAspecto;
858
859       // Establecemos el tamaño y posición final del cuadro de visión de la camara del
              minimapa.
860
861       switch (posicionMapa)
862       {
863         case "ArribaIzquierda":
864           // Arriba a la izquierda
865           camera.rect = Rect(margenXNormalizado, 1 - alturaCamaraNormalizada -
                  margenYNormalizado, anchuraCamaraNormalizada, alturaCamaraNormalizada);
866           break;
867
868         case "AbajoDerecha":
869           // Abajo a la derecha
870           camera.rect = Rect(1 - anchuraCamaraNormalizada - margenXNormalizado,
                  margenYNormalizado, anchuraCamaraNormalizada, alturaCamaraNormalizada);
871           break;
872
873         case "AbajoIzquierda":
874           // Abajo a la izquierda
```

```
875             camera.rect = Rect(margenXNormalizado, margenYNormalizado,
                    anchuraCamaraNormalizada, alturaCamaraNormalizada);
876             break;
877
878           default:
879             // Arriba a la derecha
880             camera.rect = Rect(1 - anchuraCamaraNormalizada - margenXNormalizado, 1 -
                    alturaCamaraNormalizada - margenYNormalizado, anchuraCamaraNormalizada,
                    alturaCamaraNormalizada);
881         }
882
883         // Almacenamos las coordenadas del cuadro de visión en una variable para
                referencias posteriores.
884
885         posicionVigente = camera.rect;
886
887         // Avisamos al guión "InterfazMapaControl.js" de que la resolución de la pantalla
                ha cambiado para
888         // que pueda recolocar los elementos de la interfaz consecuentemente. También a los
                 presentadores de
889         // nombre de destino y de destino cargando.
890
891         interfazMapaClon.GetComponent.<InterfazMapaControl>().actualizarInterfaz = true;
892
893         posicionCambiada = false;
894       }
895     }
896
897
898     // CANTIDAD DE TERRENO MOSTRADO
899
900     // Si el usuario ha pulsado sobre los indicadores más o menos de la interfaz del mapa
            se
901     // cambia la cantidad de terreno visto por la cámara.
902
903     if (SDN.mapaGrande == false)
904     {
905       if (camera.orthographicSize != SDN.terrenoVisto)
906       {
907         camera.orthographicSize = SDN.terrenoVisto;
908       }
909     }
910     else
911     {
912       if (camera.orthographicSize != SDN.terrenoVistoMaximizado)
913       {
914         camera.orthographicSize = SDN.terrenoVistoMaximizado;
915       }
916     }
917
918
919     if (pulsadoConmutadorMaximizar == true)
920     {
921       posicionCambiada = true;
922
923       if (SDN.mapaGrande == false)
924       {
925         camera.rect = Rect(0, 0, 1, 1);
926
```

```
927         SDN.mapaGrande = true;
928
929         // Al cambiar a mapa a pantalla completa guardamos el valor actual de cantidad
930         // de terreno mostrado en el minimapa y establecemos un valor de tamaño de
931         // terreno para el mapa grande definido por "terrenoVistoMaximizado", independiente
                del
932         // minimapa.
933
934         SDN.terrenoVisto = camera.orthographicSize;
935         camera.orthographicSize = SDN.terrenoVistoMaximizado;
936
937         interfazMapaClon.GetComponent.<InterfazMapaControl>().enabled = false;
938         interfazMapaClon.GetComponent.<InterfazMapaGrandeControl>().enabled = true;
939
940
941         pulsadoConmutadorMaximizar = false;
942
943         SDN.mapaGrande = true;
944       }
945       else
946       {
947         camera.rect = posicionVigente;
948
949         SDN.mapaGrande = false;
950
951         // Al cambiar a mapa normal desde pantalla completa guardamos el valor actual de
              terreno
952         // mostrado a pantalla completa y establecemos como valor de terreno visto para el
              modo
953         // normal aquel contenido en "terrenoVisto".
954
955         SDN.terrenoVistoMaximizado = camera.orthographicSize;
956         camera.orthographicSize = SDN.terrenoVisto;
957
958         interfazMapaClon.GetComponent.<InterfazMapaControl>().enabled = true;
959         interfazMapaClon.GetComponent.<InterfazMapaGrandeControl>().enabled = false;
960
961         pulsadoConmutadorMaximizar = false;
962
963         SDN.mapaGrande = false;
964       }
965     }
966 }
```

F.4. CamaraMapaPerimetroControl.js

```
1  #pragma strict
2
3  /*#################################################################
4
5  CamaraMapaPerimetroControl.js
6  -----------------------------
7
8  Guión encargado de gestionar el cubo perimetral que rodeará el área de
9  visión de la cámara del mapa. Este cubo tendrá la función, en el sistema
10 de navegación guiada, de producir la superficie a intersectar por el haz
11 creado entre el seguidor del avatar y las balizas de destino, generando
```

```
12 el punto donde ubicar los localizadores (indicadores de dirección a seguir).
13
14 ##################################################################*/
15
16
17 private var camaraMapa : GameObject;
18 private var posicionAltura : float;
19 private var largo : float;
20 private var ancho : float;
21 private var alto : float;
22
23
24 function Start ()
25 {
26   camaraMapa = SDN.camaraMapa;
27
28   // Cota Y a la que se encontrará el centro del cubo.
29
30   posicionAltura = -95;
31
32   // Altura fija que tendrá el cubo.
33
34   alto = 25;
35
36   transform.localScale = Vector3(0, alto, 0);
37 }
38
39
40 function Update ()
41 {
42   if (SDN.navegacionGuiada == true)
43   {
44     // El valor "Size" de la cámara representa la mitad de su dimensión real.
45
46     ancho = camaraMapa.camera.orthographicSize * 2;
47
48     // Cuando el minimapa pasa a pantalla completa pierde su proporción cuadrada
49     // y la relación de sus lados depende entonces de su relación de aspecto.
50
51     largo = ancho * camaraMapa.camera.aspect;
52
53     transform.position = camaraMapa.transform.position;
54     transform.rotation = camaraMapa.transform.rotation;
55     transform.position.y = posicionAltura;
56
57     // Redimensionamos el perímetro según cambie el perímetro del área
58     // de visión de la cámara del mapa.
59
60     transform.localScale = Vector3(largo, ancho, alto);
61   }
62 }
```

F.5. CamaraMapaSeguidor.js

```
1 #pragma strict
2
3
```

```
4  /*##################################################################
5
6  CamaraMapaSeguidor.js
7  ---------------------
8
9  Guión encargado de fijar los aspectos de la cámara observadora del mapa
10 relacionados con su posición y orientación (orientación fijada al norte
11 geográfico o determinada por la del avatar).
12
13 ##################################################################*/
14
15
16 // Variable pública que permite definir manualmente la desviación de la posición del
       norte geográfico
17 // respecto de la orientación de la escena (por defecto: -109.80).
18
19 var desviacionNorte : float;
20
21 // Cuanto más se aproxime a 1 este valor más suave será el movimiento del mapa en modo
       libre.
22
23 var suavizadoDeMovimiento : float;
24
25 private var desfaseNorteEscena : float;
26
27 @HideInInspector var objetivo : Transform;
28
29 private var altura : int;
30
31 private var rotacionVigente : Quaternion;
32 private var rotacionVigenteY : float;
33
34
35 // Variables para el cálculo del movimiento suavizado del mapa en modo libre.
36
37 private var rotacionActualY : float;
38 private var rotacionDeseadaY : float;
39
40 @HideInInspector var enPosicion : boolean;
41
42
43 function Start ()
44 {
45   // Las escenas creadas anteriormente a la integración del sistema de navegación están
         dispuestas sobre los ejes X y Z
46   // sin prestar atención a una coherencia global. Con la integración del sistema de
         navegación es necesario que cada
47   // escena esté orientada acorde a la escena "01 Exterior" de referencia, para así poder
          ubicar el norte geográfico con
48   // total exactitud.
49
50   // Para resolver esta situación, y las que puedan surgir en el futuro al desarrollar
         nuevas escenas sin el cuidado de
51   // orientarlas congruentemente, se indican a continuación el valor de los defases en
         grados de aquellas escenas que lo
52   // requieran respecto de la escena de referencia "01 Exterior".
53
54   // Nota: La escena de referencia "01 Exterior", eligiendo la vista en planta ("Top",
         eje X hacia la derecha, eje Z hacia
```

```
55  // arriba y eje Y perpendicular a ambos hacia dentro y oculto) en la ventana "Scene" de
        Unity, deja la biblioteca ubicada
56  // en el punto más alejado del eje Z.
57
58  // En el futuro, si una escena se hubiera añadido al proyecto sin vigilar su
        orientación, se observaría su desfase
59  // respecto de la escena exterior y se añadiría un nuevo bloque "case".
60
61  switch (Application.loadedLevelName)
62  {
63    case "02a EscuelaPB":
64      desfaseNorteEscena = 135;
65      break;
66
67    case "02b EscuelaP1":
68      desfaseNorteEscena = 135;
69      break;
70
71    case "04 SalondeActos":
72      desfaseNorteEscena = 135;
73      break;
74
75    case "06a Aula 01":
76      desfaseNorteEscena = 45;
77      break;
78
79    case "06b Aula 02":
80      desfaseNorteEscena = 45;
81      break;
82
83    case "06c Aula 03":
84      desfaseNorteEscena = 135;
85      break;
86
87    case "06d Aula 04":
88      desfaseNorteEscena = 135;
89      break;
90
91    case "07 Secretaria":
92      desfaseNorteEscena = 135;
93      break;
94
95    case "05a BibliotecaAbajo":
96      desfaseNorteEscena = 135;
97      break;
98
99    case "05b BibliotecaArriba":
100     desfaseNorteEscena = 135;
101     break;
102
103   case "08 Aula 204":
104     desfaseNorteEscena = 135;
105     break;
106
107   case "11 Aula 216":
108     desfaseNorteEscena = 135;
109     break;
110
111   case "03a EdifTecnologicoP1":
```

```
112         desfaseNorteEscena = 0;
113         break;
114
115       case "03b EdifTecnologicoP2":
116         desfaseNorteEscena = 0;
117         break;
118
119       case "09 Aula H3":
120         desfaseNorteEscena = 0;
121         break;
122
123       case "10 BanoET":
124         desfaseNorteEscena = 0;
125         break;
126
127       // Añadir a continuación los desfases de aquellas nuevas escenas que así lo requieran
              .
128
129       default:
130         desfaseNorteEscena = 0;
131     }
132
133
134     if (desviacionNorte < -360 || desviacionNorte > 360)
135     {
136       Debug.Log("El valor de compensación del norte introducido está fuera de rango, se
              establecerá el valor predeterminado \"-109.80\".");
137         desviacionNorte = -109.80;
138     }
139
140     if (suavizadoDeMovimiento < 1 || suavizadoDeMovimiento > 1000)
141     {
142       Debug.Log("El valor de suavizado de movimiento introducido está fuera de rango, se
              establecerá el valor predeterminado \"5\".");
143         suavizadoDeMovimiento = 5;
144     }
145
146     transform.eulerAngles.x = 90;
147
148     // Establecemos la orientación del norte aplicando la desviación según la escena en la
          que nos encontremos y la
149     // desviación global.
150
151     transform.eulerAngles.y = desviacionNorte + desfaseNorteEscena;
152
153     altura = -90;
154
155     objetivo = SDN.avatar.transform;
156
157     rotacionVigenteY = transform.rotation.eulerAngles.y;
158
159     // Hacemos que inicialmente el mapa aparezca en posición instantáneamente al cargar
160     // la escena, sin movimiento suavizado (como sí pasaría al cambiar manualmente de
161     // vista libre a fijada al norte).
162
163     enPosicion = true;
164
165     // Hacemos que la orientación del mapa esté inicialmente fijada al norte.
166     //
```

```
167   // bloqueoMapa = true;
168   //
169   // Actualización: la orientación la asigna inicialmente el guión "CamaraMapaControl.js" a partir
170   // de la variable global "SDN.bloqueoMapa".
171 }
172
173
174 function LateUpdate ()
175 {
176   // Orientación libre (determinada por la orientación del avatar).
177   if (SDN.bloqueoMapa == false)
178   {
179     // Código SIN suavizado:
180
181     //transform.eulerAngles.y = objetivo.transform.eulerAngles.y;
182
183
184     // Código CON suavizado:
185
186     rotacionActualY = transform.rotation.eulerAngles.y;
187     rotacionDeseadaY = objetivo.transform.eulerAngles.y;
188
189     // Si se usa la función "Mathf.Lerp" en lugar de "Mathf.LerpAngle", cuando el mapa gire más allá
190     // de 360 grados se producirá un efecto visual no deseado, "Mathf.LerpAngle" tiene precisamente en
191     // cuenta este tipo de giros y por ello es la función adecuada en este caso.
192
193     transform.eulerAngles.y = Mathf.LerpAngle(rotacionActualY, rotacionDeseadaY, Time.deltaTime * suavizadoDeMovimiento);
194
195     enPosicion = false;
196
197   }
198   // Orientación fijada al norte.
199   else
200   {
201     // Código SIN suavizado:
202
203     //transform.rotation = rotacionVigente;
204
205
206     // Código CON suavizado:
207
208     // Si el usuario ha cambiado la orientación del mapa a orientación libre, al regresar a orientación
209     // fija, el movimiento que realiza hasta colocarse en posición lo suavizamos. Una vez en posición
210     // se asignamos directamente la rotación correspondiente al norte sin necesidad de suavizar..
211     if (enPosicion == false)
212     {
213       rotacionActualY = transform.rotation.eulerAngles.y;
214       rotacionDeseadaY = rotacionVigenteY;
215
216       transform.eulerAngles.y = Mathf.LerpAngle(rotacionActualY, rotacionDeseadaY, Time.deltaTime * suavizadoDeMovimiento);
217
```

```
218       enPosicion = true;
219     }
220     else
221     {
222       transform.rotation = Quaternion.Euler(90, rotacionVigenteY, 0);
223     }
224   }
225
226   // Las coordenadas X y Z de la cámara se corresponden con las del avatar,
227   // la coordenada Y está fija.
228
229   transform.position = objetivo.position;
230   transform.position.y = altura;
231 }
```

F.6. ConmutadorMapaAutoactivador.js

```
1  #pragma strict
2
3  /*#####################################################################
4
5  ConmutadorMapaAutoactivador.js
6  ------------------------------
7
8  Guión encargado de conmutar automáticamente entre los mapas interiores y los exteriores
9  cuando el avatar activa alguno de los módulos del sistema conmutador de mapas de dos
       fases.
10
11 #####################################################################*/
12
13
14 private var camaraMapa : GameObject;
15
16 // Variable a la que se le asignará desde el Inspector el módulo registrador del estado
17 // del avatar, el objeto "H_AccMapa_RegistroEstadoAvatar" en la escena.
18
19 var registroEstadoAvatar : GameObject;
20
21
22 function Start ()
23 {
24   camaraMapa = SDN.camaraMapa;
25 }
26
27
28 function OnTriggerEnter (colisionador : Collider)
29 {
30   if (colisionador.tag == "PlayerPunto")
31   {
32     registroEstadoAvatar.GetComponent.<ConmutadorMapaEstadoAvatar>().avatarFuera = false;
33
34     camaraMapa.camera.cullingMask = 1 << LayerMask.NameToLayer("Mapa") | 1 << LayerMask.
           NameToLayer("MapaInterior");
35   }
36 }
```

F.7. ConmutadorMapaEntradaSalida.js

```
#pragma strict


/*#################################################################

ConmutadorMapaEntradaSalida.js
------------------------------

Guión que activa y desactiva la visualización de los mapas correspondientes a
áreas interiores en la camara del mapa.

A diferencia del sistema de fases en el que el avatar conmuta los mapas al atravesar
un plano vertical, este guión se encarga individualmente de la conmutación cuando el
avatar entra y sale de un área.

La mencionada área estará representada por un figura geométrica transparente con
volumen con la característica "Is Trigger" activada.

#################################################################*/


private var camaraMapa : GameObject;


function Start ()
{
  camaraMapa = SDN.camaraMapa;
}


// Para establecer la "Culling Mask" de la cámara del mapa usamos álgebra de Boole y operadores
// binarios.

// Unity almacena las capas en un entero de 32 bits, representando cada bit a una capa con sus dos
// estados: 1, activado; 0, desactivado.

// Para indicarle a la cámara del mapa que deseamos que observe una capa determinada hemos de poner
// un 1 en el bit correspondiente a esa capa y un cero en el resto usando una máscara de bits
// mediante "camera.cullingMask".

function OnTriggerEnter (colisionador : Collider)
{
  if (colisionador.tag == "PlayerPunto")
  {
    // Colocamos un uno en los bits de las capas "Mapa" y "MapaInterno"
    // Mapa (p. e. capa 8): 0000 1000 0000
    // MapaInterior (p. e. capa 9): 0001 0000 0000

    // Nota: usamos el operador OR (|) para sumar las dos máscaras de bits:
    // 0000 1000 0000   OR   0001 0000 0000   =   0001 1000 0000

```

```
52      // Nota: la función LayerMask.NameToLayer("...") permite utilizar los nombres de las
            capas
53      // en lugar de sus números, de esta forma se consigue que, en caso de cambiar de
            posición las
54      // capas, el código siga siendo eficaz.
55
56      camaraMapa.camera.cullingMask = 1 << LayerMask.NameToLayer("Mapa") | 1 << LayerMask.
            NameToLayer("MapaInterior");
57    }
58  }
59
60
61  function OnTriggerExit (colisionador : Collider)
62  {
63    if (colisionador.tag == "PlayerPunto")
64    {
65      camaraMapa.camera.cullingMask = 1 << LayerMask.NameToLayer("Mapa");
66    }
67  }
```

F.8. ConmutadorMapaEstadoAvatar.js

```
1  #pragma strict
2
3  /*#####################################################################
4
5  ConmutadorMapaEstadoAvatar.js
6  -----------------------------
7
8  Guión encargado de almacenar el estado del avatar en el sistema de
9  conmutación de mapas bifase.
10
11 Acceden al él las fases, los modulos de autoactivación y los módulos
12 de seguro.
13
14 #####################################################################*/
15
16
17 var avatarFuera : boolean;
18
19
20 function Awake ()
21 {
22   avatarFuera = true;
23 }
```

F.9. ConmutadorMapaFamilias.js

```
1  #pragma strict
2
3  /*#####################################################################
4
5  ConmutadorMapaFamilias.js
6  -------------------------
```

```
7
8  Guión encargado de conmutar el estado entre activo e inactivo de dos familias de mapas
9  dentro de una misma escena.
10
11 Cada familia de mapas podrá contener tanto el mapa general como los submapas interiores,
12 y el guión activará (o desactivará) una de las familias al tiempo que desactiva (o activa
       )
13 la otra al entrar el avatar  en el elemento activador al que el guión esté asociado.
14
15 Las familias de mapas se agruparán dentro de un objeto vacío contenedor: uno para el mapa
16 general de la familia uno, otro para el mapa general de la familia dos, otro para los
       mapas
17 interiores de la familia uno, y otro para los mapas interiores de la familia dos.
18
19 Cada uno de estos objetos agrupadores se asignará en el Inspector a las variables
       correspondientes.
20
21 ##################################################################*/
22
23
24 // Mapas activados mientras el avatar no haya entrado dentro del área activadora.
25
26 var mapasGlobalesPredeterminados : GameObject;
27 var mapasInterioresPredeterminados : GameObject;
28
29 // Mapas activados cuando el avatar entre dentro del área activadora.
30
31 var mapasGlobalesConmutados : GameObject;
32 var mapasInterioresConmutados : GameObject;
33
34
35 function Start ()
36 {
37   if (mapasGlobalesPredeterminados != null)
38   {
39     mapasGlobalesPredeterminados.SetActive(true);
40   }
41   if (mapasInterioresPredeterminados != null)
42   {
43     mapasInterioresPredeterminados.SetActive(true);
44   }
45   if (mapasGlobalesConmutados != null)
46   {
47     mapasGlobalesConmutados.SetActive(false);
48   }
49   if (mapasInterioresConmutados != null)
50   {
51     mapasInterioresConmutados.SetActive(false);
52   }
53 }
54
55
56 function OnTriggerEnter (colisionador : Collider)
57 {
58   if (colisionador.tag == "PlayerPunto")
59   {
60     if (mapasGlobalesPredeterminados != null)
61     {
62       mapasGlobalesPredeterminados.SetActive(false);
```

```
63      }
64      if (mapasInterioresPredeterminados != null)
65      {
66        mapasInterioresPredeterminados.SetActive(false);
67      }
68      if (mapasGlobalesConmutados != null)
69      {
70        mapasGlobalesConmutados.SetActive(true);
71      }
72      if (mapasInterioresConmutados != null)
73      {
74        mapasInterioresConmutados.SetActive(true);
75      }
76    }
77  }
78
79
80  function OnTriggerExit (colisionador : Collider)
81  {
82    if (colisionador.tag == "PlayerPunto")
83    {
84      if (mapasGlobalesPredeterminados != null)
85      {
86        mapasGlobalesPredeterminados.SetActive(true);
87      }
88      if (mapasInteriosPredeterminados != null)
89      {
90        mapasInteriosPredeterminados.SetActive(true);
91      }
92      if (mapasGlobalesConmutados != null)
93      {
94        mapasGlobalesConmutados.SetActive(false);
95      }
96      if (mapasInteriosConmutados != null)
97      {
98        mapasInteriosConmutados.SetActive(false);
99      }
100   }
101 }
```

F.10. ConmutadorMapaFaseExterna.js

```
1  #pragma strict
2
3  /*#####################################################################
4
5  ConmutadorMapaFaseExterna.js
6  ----------------------------
7
8  Guión correspondiente a la fase externa de las tres (externa, interna e inferior) que
       permiten
9  detectar el paso del avatar a través de ellas inequívocamente y desencadenar una acción
       en
10 consecuencia, en este caso el cambio del comportamiento de la cámara del minimapa para
       mostrar
11 mapas interiores o exteriores.
12
```

```
13 Nota: La distancia que separará los dos planos verticales que representarán las fases
       externa e interna
14 de este sistema de detección deberá ser inferior al diámetro de la esfera representativa
       del avatar e
15 identificada como "H_AvatarPunto" en la jerarquía del proyecto.
16
17 ##################################################################*/
18
19
20 @HideInInspector var avatarFuera : boolean;
21 var faseInterna : GameObject;
22 var faseInferior : GameObject;
23
24 var registroEstadoAvatar : GameObject;
25
26 private var avatarEnFaseInterna : boolean;
27 @HideInInspector var avatarEnFaseExterna : boolean;
28 private var avatarEnFaseInferior : boolean;
29
30 private var camaraMapa : GameObject;
31
32
33 function Start ()
34 {
35   camaraMapa = SDN.camaraMapa;
36
37   avatarFuera = registroEstadoAvatar.GetComponent.<ConmutadorMapaEstadoAvatar>().
         avatarFuera;
38 }
39
40
41 function OnTriggerEnter (colisionador : Collider)
42 {
43   if (colisionador.CompareTag("PlayerPunto"))
44   {
45     avatarEnFaseExterna = true;
46     // Debug.Log("El avatar ha entrado en la fase EXTERNA");
47   }
48 }
49
50
51 function OnTriggerExit (colisionador : Collider)
52 {
53   if (colisionador.CompareTag("PlayerPunto"))
54   {
55     avatarEnFaseExterna = false;
56     // Debug.Log("El avatar ha salido de la fase EXTERNA");
57
58     avatarFuera = registroEstadoAvatar.GetComponent.<ConmutadorMapaEstadoAvatar>().
           avatarFuera;
59
60     // Si el avatar sale a exteriores es la fase exterior la que hace las comprobaciones.
61     if (avatarFuera == false)
62     {
63       avatarEnFaseInterna = faseInterna.GetComponent.<ConmutadorMapaFaseInterna>().
             avatarEnFaseInterna;
64
65       /*
```

```
66        // Nota: Anulada la fase inferior, sustituida por la fase exterior modificada para
              que abarque el espacio
67        // abarcado anteriormente por la fase inferior.
68
69        // Si el avatar sigue en contacto con la fase inferior significa que no está
              saliendo hacia el interior
70        // ni hacia el exterior, sino hacia un lado; en ese caso no se modifica el
              comportamiento de la cámara.
71
72        // avatarEnFaseInferior = faseInferior.GetComponent.<ConmutadorMapaFaseInferior>().
              avatarEnFaseInferior;
73        */
74
75        // Si el avatar abandona la fase externa y la esfera representativa del mismo no
              está en contacto con
76        // la fase interna significa que ha salido hacia el exterior.
77        if (avatarEnFaseInterna == false)
78        {
79          camaraMapa.camera.cullingMask = 1 << LayerMask.NameToLayer("Mapa");
80
81          registroEstadoAvatar.GetComponent.<ConmutadorMapaEstadoAvatar>().avatarFuera =
                true;
82        }
83        // ... si no, significa que el avatar ha abandonado la fase externa hacia el
              interior y por tanto no
84        // se desactivan los mapas de interiores.
85      }
86    }
87  }
```

F.11. ConmutadorMapaFaseInterna.js

```
1  #pragma strict
2
3  /*#####################################################################
4
5  ConmutadorMapaFaseInterna.js
6  ----------------------------
7
8  Guión correspondiente a la fase interna de las tres (externa, interna e inferior) que
       permiten
9  detectar el paso del avatar a través de ellas inequívocamente y desencadenar una acción
       en
10 consecuencia, en este caso el cambio del comportamiento de la cámara del minimapa para
       mostrar
11 mapas interiores o exteriores.
12
13 Nota: La distancia que separará los dos planos verticales que representarán las fases
       externa e interna
14 de este sistema de detección deberá ser inferior al diámetro de la esfera representativa
       del avatar e
15 identificada como "H_AvatarPunto" en la jerarquía del proyecto.
16
17 #####################################################################*/
18
19
20 @HideInInspector var avatarFuera : boolean;
```

```
21 var faseExterna : GameObject;
22 var faseInferior : GameObject;
23
24 var registroEstadoAvatar : GameObject;
25
26 @HideInInspector var avatarEnFaseInterna : boolean;
27 private var avatarEnFaseExterna : boolean;
28 private var avatarEnFaseInferior : boolean;
29
30 private var camaraMapa : GameObject;
31
32
33 function Start ()
34 {
35   camaraMapa = SDN.camaraMapa;
36
37   avatarFuera = registroEstadoAvatar.GetComponent.<ConmutadorMapaEstadoAvatar>().
         avatarFuera;
38 }
39
40
41 function OnTriggerEnter (colisionador : Collider)
42 {
43   if (colisionador.CompareTag("PlayerPunto"))
44   {
45     avatarEnFaseInterna = true;
46     // Debug.Log("El avatar ha entrado en la fase INTERNA");
47   }
48 }
49
50
51 function OnTriggerExit (colisionador : Collider)
52 {
53   if (colisionador.CompareTag("PlayerPunto"))
54   {
55     avatarEnFaseInterna = false;
56     // Debug.Log("El avatar ha salido de fase INTERNA");
57
58     avatarFuera = registroEstadoAvatar.GetComponent.<ConmutadorMapaEstadoAvatar>().
           avatarFuera;
59
60     // Si el avatar entra en interiores es la fase interior la que hace las
           comprobaciones.
61     if (avatarFuera == true)
62     {
63       avatarEnFaseExterna = faseExterna.GetComponent.<ConmutadorMapaFaseExterna>().
             avatarEnFaseExterna;
64
65       /*
66       // Nota: Anulada la fase inferior, sustituida por la fase exterior modificada para
             que abarque el espacio
67       // abarcado anteriormente por la fase inferior.
68
69       // Si el avatar sigue en contacto con la fase inferior significa que no está
             saliendo hacia el interior
70       // ni hacia el exterior, sino hacia un lado; en ese caso no se modifica el
             comportamiento de la cámara.
71
```

```
72        // avatarEnFaseInferior = faseInferior.GetComponent.<ConmutadorMapaFaseInferior>().
              avatarEnFaseInferior;
73        */
74
75        // Si el avatar abandona la fase interna y la esfera representativa del mismo no
              está en contacto con
76        // la fase externa significa que ha entrado en interiores.
77        if (avatarEnFaseExterna == false)
78        {
79          camaraMapa.camera.cullingMask = 1 << LayerMask.NameToLayer("Mapa") | 1 <<
                LayerMask.NameToLayer("MapaInterior");
80
81          registroEstadoAvatar.GetComponent.<ConmutadorMapaEstadoAvatar>().avatarFuera =
                false;
82        }
83        // ... si no, significa que el avatar ha abandonado la fase interna hacia el
              exterior.
84      }
85    }
86  }
```

F.12. ConmutadorMapaSeguroExterno.js

```
1  #pragma strict
2
3  /*#####################################################################
4
5  ConmutadorMapaSeguroExterno.js
6  ------------------------------
7
8  Guión gestor del módulo de seguridad del sistema de conmutación bifase correspondiente
9  a la fase externa.
10
11 Este módulo garantiza la integridad y precisión del sistema al salir
12 hacia exteriores aun cuando los "frames" por segundo de ejecución del programa sean
13 elevados o cuando el programa ejecute las instrucciones dadas por las fases exterior
14 e interior en un mismo frame de forma aleatoria.
15
16 #####################################################################*/
17
18
19 private var camaraMapa : GameObject;
20 var registroEstadoAvatar : GameObject;
21
22
23 function Start ()
24 {
25   camaraMapa = SDN.camaraMapa;
26 }
27
28
29 function OnTriggerStay (colisionador : Collider)
30 {
31   if (colisionador.tag == "PlayerPunto")
32   {
33     if (registroEstadoAvatar.GetComponent.<ConmutadorMapaEstadoAvatar>().avatarFuera !=
          true)
```

```
34      {
35        registroEstadoAvatar.GetComponent.<ConmutadorMapaEstadoAvatar>().avatarFuera = true
              ;
36        camaraMapa.camera.cullingMask = 1 << LayerMask.NameToLayer("Mapa");
37      }
38    }
39  }
```

F.13. ConmutadorMapaSeguroInterno.js

```
1  #pragma strict
2
3  /*#####################################################################
4
5  ConmutadorMapaSeguroInterno.js
6  ------------------------------
7
8  Guión gestor del módulo de seguridad del sistema de conmutación bifase correspondiente
9  a la fase interna.
10
11 Este módulo garantiza la integridad y precisión del sistema al salir
12 hacia interiores aun cuando los "frames" por segundo de ejecución del programa sean
13 elevados o cuando el programa ejecute las instrucciones dadas por las fases exterior
14 e interior en un mismo frame de forma aleatoria.
15
16 #####################################################################*/
17
18
19 private var camaraMapa : GameObject;
20 var registroEstadoAvatar : GameObject;
21
22
23 function Start ()
24 {
25   camaraMapa = SDN.camaraMapa;
26 }
27
28
29 function OnTriggerStay (colisionador : Collider)
30 {
31   if (colisionador.tag == "PlayerPunto")
32   {
33     if (registroEstadoAvatar.GetComponent.<ConmutadorMapaEstadoAvatar>().avatarFuera !=
          false)
34     {
35       registroEstadoAvatar.GetComponent.<ConmutadorMapaEstadoAvatar>().avatarFuera =
            false;
36       camaraMapa.camera.cullingMask = 1 << LayerMask.NameToLayer("Mapa") | 1 << LayerMask
            .NameToLayer("MapaInterior");
37     }
38   }
39 }
```

F.14. DondeEstoyControl.js

```
#pragma strict

/*#####################################################################

DondeEstoyControl.js
--------------------

Guión encargado de determinar la ubicación del avatar y de asignar este valor
a la variable global "SDN.dondeEstoy".

Es un elemento principal del sistema de navegación guiada y del sistema informativo
de localización.

#####################################################################*/


private var registroEstadoAvatarExterior : GameObject;
private var controlCambioUnion : boolean = false;


function Awake ()
{

  switch (Application.loadedLevelName)
  {

    // Nota: aquellas áreas considerables individuales que no conformen una escena por sí mismas
    // sino que pertenezcan en su lugar a una más grande estarán controladas por objetos accionadores
    // que las abarquen mediante las funciones "OnTriggerEnter()" y "OnTriggerExit()". Ejemplos:
    // Todas las plantas del edificio tecnológico (alas este y oeste de las plantas baja, uno, dos, tres y
    // cuatro, rellanos de las escaleras del tecnológico o despacho de Fernando Jorge Fraile Fernández).
    // La función "OnTriggerEnter()" establecerá la subárea al entrar, y la función "OnTriggerExit()"
    // establecerá como área aquella más grande a la que pertenece la subárea al salir.

    // Nota: las ubicaciones gestionadas por puntos de control están controladas por guiones ubicados en
    // la carpeta "Proyecto H/Scripts/PuntosDeControl".

    case "01 Exterior":
      SDN.dondeEstoy = "Exterior";
      break;

    case "02a EscuelaPB":
      SDN.dondeEstoy = "EscuelaPlantaBaja";
      break;

    case "02b EscuelaP1":
      SDN.dondeEstoy = "EscuelaPlantaUno";
      break;

    // Ojo: el "de" del nombre original de la escena va en minúsculas.
    case "04 SalondeActos":
      SDN.dondeEstoy = "SalonDeActos";
```

```
53        break;
54
55      case "05a BibliotecaAbajo":
56        SDN.dondeEstoy = "BibliotecaPlantaBaja";
57        break;
58
59      case "05b BibliotecaArriba":
60        SDN.dondeEstoy = "BibliotecaPlantaUno";
61        break;
62
63      case "06a Aula 01":
64        SDN.dondeEstoy = "AulaUno";
65        break;
66
67      case "06b Aula 02":
68        SDN.dondeEstoy = "AulaDos";
69        break;
70
71      case "06c Aula 03":
72        SDN.dondeEstoy = "AulaTres";
73        break;
74
75      case "06d Aula 04":
76        SDN.dondeEstoy = "AulaCuatro";
77        break;
78
79      case "07 Secretaria":
80        SDN.dondeEstoy = "Secretaria";
81        break;
82
83      case "08 Aula 204":
84        SDN.dondeEstoy = "AulaDoscientoscuatro";
85        break;
86
87      case "09 Aula H3":
88        SDN.dondeEstoy = "AulaHacheTres";
89        break;
90
91      case "10 BanoET":
92        SDN.dondeEstoy = "TecnologicoBanoChicasPlantaBaja";
93        break;
94
95      case "11 Aula 216":
96        SDN.dondeEstoy = "AulaDoscientosdieciseis";
97        break;
98
99      default:
100       SDN.dondeEstoy = "";
101       Debug.Log("Nombre de escena no recogido. La base de datos no está actualizada o se
              está usando un punto de control manual.");
102
103   }
104
105 }
106
107
108 // El siguiente código (bloques "Start()" y "LateUpdate()") permite determinar si el
      avatar, estando en la escena
```

```
109 // correspondiente al exterior de los edificios (aparcamientos), está dentro del módulo
        que une la escuela de
110 // ingenierías con el edificio tecnológico. De esta forma la navegación guiada podrá
        indicar la puerta de entrada
111 // a la escuela más cercana al edificio tecnológico cuando el avatar desea acceder desde
        la unión.
112
113 function Start ()
114 {
115   if (Application.loadedLevelName == "01 Exterior")
116   {
117     // Localizamos el módulo registrador de estado del sistema de conmutación bifase y
          determinamos el
118     // estado inicial del avatar (en interiores o exteriores) a partir de él.
119
120     registroEstadoAvatarExterior = GameObject.Find("H_AccMapa_RegistroEstadoAvatar");
121     controlCambioUnion = registroEstadoAvatarExterior.GetComponent.<
          ConmutadorMapaEstadoAvatar>().avatarFuera;
122   }
123 }
124
125
126 function LateUpdate ()
127 {
128   // Activamos el punto de control de la unión entre la escuela y el edificio tecnológico
        sólo cuando el avatar entre o salga de ella.
129
130   if (Application.loadedLevelName == "01 Exterior" && controlCambioUnion !=
        registroEstadoAvatarExterior.GetComponent.<ConmutadorMapaEstadoAvatar>().
        avatarFuera)
131   {
132     if (registroEstadoAvatarExterior.GetComponent.<ConmutadorMapaEstadoAvatar>().
          avatarFuera == true)
133     {
134       SDN.dondeEstoy = "Exterior";
135     }
136     else
137     {
138       SDN.dondeEstoy = "ExteriorUnion";
139     }
140
141     LocalizadorDeBalizas.Localizar();
142
143     controlCambioUnion = registroEstadoAvatarExterior.GetComponent.<
          ConmutadorMapaEstadoAvatar>().avatarFuera;
144
145     // Avisamos al intersectador de que se ha activado un punto de control.
146
147     SDN.puntoDeControlNavegacionGuiada = true;
148   }
149 }
```

F.15. IndicadorAvatarSeguidor.js

```
1 #pragma strict
2
3 /*#################################################################
```

```
4
5  IndicadorAvatarSeguidor.js
6  --------------------------
7
8  Guión encargado de gestionar la posición y rotación del elemento representativo
9  del avatar en el mapa, "H_IndicadorAvatar".
10
11 ##################################################################*/
12
13
14 private var objetivo : Transform;
15
16 private var altura : int;
17
18
19 function Start ()
20 {
21   // Nota: Plano a -100, indicador a -94.999 y cámara a -90. (Los localizadores usados
22   // en la navegación guiada están colocados a -95, así el indicador quedará siempre
23   // por encima).
24
25   altura = -94.999;
26
27   objetivo = SDN.avatar.transform;
28
29   // Se establece el indicador de posición como "hijo" del controlador para emparejar la
         rotación de ambos.
30
31   transform.parent = objetivo.transform;
32
33   // Coordinamos la rotacion inicial de controlador e indicador.
34
35   transform.rotation = objetivo.rotation;
36
37   // Rotamos el cubo que sirve de indicador 45 grados. De esta forma respecto de
38   // los ejes de coordenadas parecerá un rombo, y podremos así aprovechar
39   // sus ángulos para indicar visualmente el sentido de marcha del avatar.
40
41   transform.rotation *= Quaternion.Euler(0, -45, 0);
42 }
43
44 function LateUpdate ()
45 {
46   // Seguimos al objetivo desde una cota predeterminada.
47
48   // Nota: Utilizamos el ciclo "LateUpdate" para asegurarnos de que la cámara
49   // sigue fielmente al avatar cualesquiera movimientos que haya realizado éste
50   // (los movimientos del avatar se resuelven en el ciclo previo "Update").
51
52   transform.position = objetivo.position;
53   transform.position.y = altura;
54 }
```

F.16. InterfazApagadoControl.js

```
1 #pragma strict
2
```

```
3  /*#####################################################################
4
5  InterfazApagadoControl.js
6  -------------------------
7
8  Guión encargado de controlar los aspectos visuales y de posición de los
9  elementos del menú de apagado (activo en las plataformas "Standalone",
10 Windows, Mac OS X y Linux.
11
12 #####################################################################*/
13
14
15 // Valores configurables indicadores de la anchura y la altura del menú
16 // selector de destino.
17
18 var porcentajeBotonRegresar : float;
19 var porcentajeMargenBotonRegresar : float;
20 var texturaFondo : Texture;
21
22 var porcentajeBotonApagado : float;
23 var porcentajeAnchoTexto : float;
24 var porcentajeAltoTexto : float;
25
26 var estiloBotonRegresar : GUIStyle;
27
28 var estiloInformacionApagado : GUIStyle;
29 var estiloBotonApagado : GUIStyle;
30
31 private var anchoTexto : float;
32 private var altoTexto : float;
33 private var alturaPantalla : float;
34 private var anchuraPantalla : float;
35 private var tamanoBotonRegresar : float;
36 private var margenBotonRegresar : float;
37 private var tamanoBotonApagado : float;
38
39
40 function Start()
41 {
42
43   if (porcentajeAnchoTexto <= 0 || porcentajeAnchoTexto > 100)
44   {
45     Debug.Log("La anchura indicada para el <i>texto informativo</i> está fuera de rango,
         se utilizará el valor predeterminado \"45\".");
46     porcentajeAnchoTexto = 45;
47   }
48
49   if (porcentajeAltoTexto <= 0 || porcentajeAltoTexto > 100)
50   {
51     Debug.Log("La altura indicada para el fondo del texto informativo del botón apagado
         está fuera de rango, se utilizará el valor predeterminado \"30\".");
52     porcentajeAltoTexto = 30;
53   }
54
55   if (porcentajeBotonRegresar <= 0 || porcentajeBotonRegresar > 100)
56   {
57     Debug.Log("El tamaño del botón <i>regresar</i> indicado está fuera de rango, se
         utilizará el valor predeterminado \"8\".");
58     porcentajeBotonRegresar = 8;
```

```
59    }
60
61    if (porcentajeMargenBotonRegresar <= 0 || porcentajeMargenBotonRegresar > 100)
62    {
63      Debug.Log("El porcentaje de margen indicado entre la pantalla y el botón <i>regresar
          </i> está fuera de rango, se utilizará el valor predeterminado \"2\".");
64      porcentajeMargenBotonRegresar = 2;
65    }
66
67    if (porcentajeBotonApagado <= 0 || porcentajeBotonApagado > 100)
68    {
69      Debug.Log("El tamaño del botón de <i>apagado</i> indicado está fuera de rango, se
          utilizará el valor predeterminado \"35\".");
70      porcentajeBotonRegresar = 35;
71    }
72
73
74    anchuraPantalla = Screen.width;
75    alturaPantalla = Screen.height;
76
77    anchoTexto = porcentajeAnchoTexto / 100 * anchuraPantalla;
78    altoTexto = porcentajeAltoTexto / 100 * alturaPantalla;
79
80    /*
81    // El siguiente pseudo-código permite tomar como referencia para el tamaño de la
82    // interfaz el lado de la pantalla que sea menor, en lugar de la altura de la pantalla
83    // (útil si se va a ejecutar el programa en plataformas con pantallas verticales).
84
85    if (anchuraPantalla <= alturaPantalla)
86    {
87      factorTamanoIcono = anchuraPantalla;
88    }
89    else
90    {
91      factorTamanoIcono = alturaPantalla;
92    }
93
94    // Calculado "factorTamanoIcono", solamente cabría sustituir el factor indicador del
95    // tamaño de los iconos (actualmente "alturaPantalla"), para obtener el tamaño
          dependiente
96    // del lado menor.
97
98    // Por ejemplo: tamanoBotonApagado = porcentajeBotonApagado / 100 * factorTamanoIcono;
99    */
100
101   tamanoBotonRegresar = porcentajeBotonRegresar / 100 * alturaPantalla;
102   margenBotonRegresar = porcentajeMargenBotonRegresar / 100 * alturaPantalla;
103   tamanoBotonApagado = porcentajeBotonApagado / 100 * alturaPantalla;
104
105 }
106
107
108 function Update ()
109 {
110
111   if (Screen.width != anchuraPantalla || Screen.height != alturaPantalla)
112   {
113
114     alturaPantalla = Screen.height;
```

```
115     anchuraPantalla = Screen.width;
116
117     anchoTexto = porcentajeAnchoTexto / 100 * anchuraPantalla;
118     altoTexto = porcentajeAltoTexto / 100 * alturaPantalla;
119
120     /*
121     // Pseudo-código que permite tomar como referencia para el tamaño de la
122     // interfaz el lado de la pantalla que sea menor (explicado en el ciclo
123     // "Start()".
124
125     if (anchuraPantalla <= alturaPantalla)
126     {
127       factorTamanoIcono = anchuraPantalla;
128     }
129     else
130     {
131       factorTamanoIcono = alturaPantalla;
132     }
133     */
134
135     tamanoBotonRegresar = porcentajeBotonRegresar / 100 * alturaPantalla;
136     margenBotonRegresar = porcentajeMargenBotonRegresar / 100 * alturaPantalla;
137     tamanoBotonApagado = porcentajeBotonApagado / 100 * alturaPantalla;
138   }
139
140 }
141
142
143 function OnGUI ()
144 {
145
146   // Textura por debajo del botón de apagado.
147
148   GUI.DrawTexture(Rect(anchuraPantalla / 2 - tamanoBotonApagado * 2 / 2,
149             alturaPantalla / 2 - tamanoBotonApagado * 2 / 2 + alturaPantalla * 0.13,
150             tamanoBotonApagado * 2, tamanoBotonApagado * 2), texturaFondo);
151
152   // Botón regresar.
153
154   if (GUI.Button(Rect(Screen.width - tamanoBotonRegresar - margenBotonRegresar,
        margenBotonRegresar, tamanoBotonRegresar, tamanoBotonRegresar), "",
        estiloBotonRegresar))
155   {
156     SDN.DesactivarSelectorApagado();
157   }
158
159   // Texto informativo
160
161   GUI.Label(Rect(anchuraPantalla / 2 - anchoTexto / 2,
162           alturaPantalla / 2 - tamanoBotonApagado / 2 - alturaPantalla * 0.13,
163           anchoTexto, altoTexto),
164           Traductor.Traducir("PreguntaApagado"), estiloInformacionApagado);
165
166   // Botón de apagado.
167
168   if (GUI.Button(Rect(anchuraPantalla / 2 - tamanoBotonApagado / 2,
169             alturaPantalla / 2 - tamanoBotonApagado / 2 + alturaPantalla * 0.13,
170             tamanoBotonApagado, tamanoBotonApagado),
171             "", estiloBotonApagado))
```

```
172   {
173     Application.Quit();
174   }
175
176 }
```

F.17. InterfazDestinos.js

```
1  #pragma strict
2
3  /*#####################################################################
4
5  InterfazDestinos.js
6  -------------------
7
8  Guión encargado de controlar los aspectos visuales y de posición, y el
9  funcionamiento de los elementos del menú de selección de destino.
10
11 Los elementos que incluye son los nombres de los destinos a los que se puede
12 acceder mediante el transporte SDN, y para aquellos que dispongan de ello,
13 la posibilidad de activar la navegación guiada para alcanzarlos; también
14 incluye las instrucciones para el usuario.
15
16 #####################################################################*/
17
18
19 // Valores configurables indicadores de la anchura y la altura del menú
20 // selector de destino.
21
22 var porcentajeAncho : float;
23 var porcentajeAlto : float;
24
25 var porcentajeBotonRegresar : float;
26 var porcentajeMargenBotonRegresar : float;
27 var texturaFondo : Texture;
28 var iconoBrujula : Texture;
29 var iconoLocalizadorNormal : Texture;
30 var iconoLocalizadorBajar : Texture;
31 var iconoLocalizadorSubir : Texture;
32 //var tipografiaTitulo : Font;
33 //private var tipografiaOriginal : Font;
34
35 private var ancho : float;
36 private var alto : float;
37 private var margenAncho : float;
38 private var margenAlto : float;
39 private var alturaPantalla : float;
40 private var anchuraPantalla : float;
41 private var tamanoBotonRegresar : float;
42 private var margenBotonRegresar : float;
43
44 private var posicionScroll : Vector2;
45 private var posicionScrollInstrucciones : Vector2;
46
47 var estiloFondo : GUIStyle;
48 var estiloBotonRegresar : GUIStyle;
49 var estiloTitulo : GUIStyle;
```

```
50 var estiloBotonDestino : GUIStyle;
51 var estiloBotonNavegacionGuiada : GUIStyle;
52 var estiloInstrucciones : GUIStyle;
53 var estiloIconoInstrucciones : GUIStyle;
54 var estiloSubInstrucciones : GUIStyle;
55 var estiloIconoSubInstrucciones : GUIStyle;
56
57
58 function Start()
59 {
60
61   if (porcentajeAncho <= 0 || porcentajeAncho > 100)
62   {
63     Debug.Log("La anchura indicada para el menú selector de destinos está fuera de rango,
          se utilizará el valor predeterminado \"45\".");
64     porcentajeAncho = 45;
65   }
66
67   if (porcentajeAlto <= 0 || porcentajeAlto > 100)
68   {
69     Debug.Log("La altura indicada para el menú selector de destinos está fuera de rango,
         se utilizará el valor predeterminado \"80\".");
70     porcentajeAlto = 80;
71   }
72
73   if (porcentajeBotonRegresar <= 0 || porcentajeBotonRegresar > 100)
74   {
75     Debug.Log("El tamaño del botón <i>regresar</i> indicado está fuera de rango, se
         utilizará el valor predeterminado \"8\".");
76     porcentajeBotonRegresar = 8;
77   }
78
79   if (porcentajeMargenBotonRegresar <= 0 || porcentajeMargenBotonRegresar > 100)
80   {
81     Debug.Log("El porcentaje de margen indicado entre la pantalla y el botón <i>regresar
         </i> está fuera de rango, se utilizará el valor predeterminado \"2\".");
82     porcentajeMargenBotonRegresar = 2;
83   }
84
85
86   anchuraPantalla = Screen.width;
87   alturaPantalla = Screen.height;
88
89   ancho = porcentajeAncho / 100 * anchuraPantalla;
90   alto = porcentajeAlto / 100 * alturaPantalla;
91
92   margenAncho = (anchuraPantalla - ancho) / 2;
93   margenAlto = (alturaPantalla - alto) / 2;
94
95   //estiloTitulo = GUIStyle(GUI.skin.box);
96   //estiloTitulo.normal.textColor = Color.black;
97   //estiloBotonDestino.hover.textColor = Color.yellow;
98
99   tamanoBotonRegresar = porcentajeBotonRegresar / 100 * alturaPantalla;
100  margenBotonRegresar = porcentajeMargenBotonRegresar / 100 * alturaPantalla;
101
102 }
103
104
```

```
105 function Update ()
106 {
107
108   if (Screen.width != anchuraPantalla || Screen.height != alturaPantalla)
109   {
110
111     alturaPantalla = Screen.height;
112     anchuraPantalla = Screen.width;
113
114     ancho = porcentajeAncho / 100 * anchuraPantalla;
115     alto = porcentajeAlto / 100 * alturaPantalla;
116
117     margenAncho = (anchuraPantalla - ancho) / 2;
118     margenAlto = (alturaPantalla - alto) / 2;
119
120     tamanoBotonRegresar = porcentajeBotonRegresar / 100 * alturaPantalla;
121     margenBotonRegresar = porcentajeMargenBotonRegresar / 100 * alturaPantalla;
122
123   }
124
125 }
126
127
128 function OnGUI ()
129 {
130
131   //estiloBotonDestino = GUIStyle(GUI.skin.button);
132   //estiloBotonDestino.normal.textColor = Color.black;
133   //estiloBotonDestino.hover.textColor = Color.yellow;
134
135
136   // Textura de fondo.
137
138   GUI.DrawTexture(Rect(margenAncho, margenAlto, ancho, alto), texturaFondo);
139
140
141   if (GUI.Button(Rect(Screen.width - tamanoBotonRegresar - margenBotonRegresar,
        margenBotonRegresar, tamanoBotonRegresar, tamanoBotonRegresar), "",
        estiloBotonRegresar))
142   {
143     SDN.DesactivarSelectorDestinos();
144   }
145
146
147   GUILayout.BeginArea(Rect(margenAncho, margenAlto, ancho, alto));
148
149
150
151     GUILayout.BeginVertical();
152
153
154
155       GUILayout.BeginHorizontal();
156
157         //GUI.skin.font = tipografiaTitulo;
158         GUILayout.Box(Traductor.Traducir("ELIGE UN DESTINO"), estiloTitulo);
159         //GUI.skin.font = tipografiaOriginal;
160
161       GUILayout.EndHorizontal();
```

```
162
163
164
165         GUILayout.BeginHorizontal();
166
167           GUILayout.BeginVertical();
168
169             GUILayout.Space(16);
170
171           GUILayout.EndVertical();
172
173         GUILayout.EndHorizontal();
174
175
176
177         posicionScroll = GUILayout.BeginScrollView(posicionScroll, GUILayout.Width(ancho),
                GUILayout.Height(alto / 2));
178
179
180
181           /*
182           // El siguiente destino se puede personalizar para acceder a cualquier punto de
                forma rápida
183           // cuando el desarrollador así lo requisiera.
184
185           GUILayout.BeginHorizontal();
186
187             if (GUILayout.Button (Traductor.Traducir("Transporte manual"),
                    estiloBotonDestino))
188             {
189               TransporteSDN.Destino(0, 0, 0, 0, "01 Exterior", "Transporte\nmanual");
190             }
191
192           GUILayout.EndHorizontal();
193           */
194
195
196
197           GUILayout.BeginHorizontal();
198
199             if (GUILayout.Button(Traductor.Traducir("BotonDespachoFernandoJFF"),
                    estiloBotonDestino))
200             {
201               TransporteSDN.Destino("TecnologicoDespachoFernandoJFF");
202             }
203
204             if (GUILayout.Button("", estiloBotonNavegacionGuiada, GUILayout.Height(18)))
205             {
206               SDN.ActivarNavegacionGuiada("TecnologicoDespachoFernandoJFF");
207
208               SDN.DesactivarSelectorDestinos();
209             }
210
211           GUILayout.EndHorizontal();
212
213
214
215           GUILayout.BeginHorizontal();
216
```

```
217             if (GUILayout.Button(Traductor.Traducir("BotonEscuelaPlantaBaja"),
                    estiloBotonDestino))
218             {
219               TransporteSDN.Destino("EscuelaPlantaBaja");
220             }
221
222             if (GUILayout.Button("", estiloBotonNavegacionGuiada))
223             {
224               SDN.ActivarNavegacionGuiada("EscuelaPlantaBaja");
225
226               SDN.DesactivarSelectorDestinos();
227             }
228
229           GUILayout.EndHorizontal();
230
231
232
233           GUILayout.BeginHorizontal();
234
235             if (GUILayout.Button(Traductor.Traducir("BotonEscuelaPlantaUno"),
                    estiloBotonDestino))
236             {
237               TransporteSDN.Destino("EscuelaPlantaUno");
238             }
239
240           GUILayout.EndHorizontal();
241
242
243
244           GUILayout.BeginHorizontal();
245
246             if (GUILayout.Button(Traductor.Traducir("BotonSecretaria"), estiloBotonDestino)
                    )
247             {
248               TransporteSDN.Destino("Secretaria");
249             }
250
251             if (GUILayout.Button("", estiloBotonNavegacionGuiada))
252             {
253               SDN.ActivarNavegacionGuiada("Secretaria");
254
255               SDN.DesactivarSelectorDestinos();
256             }
257
258           GUILayout.EndHorizontal();
259
260
261
262           GUILayout.BeginHorizontal();
263
264             if (GUILayout.Button(Traductor.Traducir("BotonSalonDeActos"),
                    estiloBotonDestino))
265             {
266               TransporteSDN.Destino("SalonDeActos");
267             }
268
269             if (GUILayout.Button("", estiloBotonNavegacionGuiada))
270             {
271               SDN.ActivarNavegacionGuiada("SalonDeActos");
```

```
272
273             SDN.DesactivarSelectorDestinos();
274           }
275
276         GUILayout.EndHorizontal();
277
278
279
280         GUILayout.BeginHorizontal();
281
282           if (GUILayout.Button(Traductor.Traducir("BotonBibliotecaPlantaBaja"),
                estiloBotonDestino))
283           {
284             TransporteSDN.Destino("BibliotecaPlantaBaja");
285           }
286
287           if (GUILayout.Button("", estiloBotonNavegacionGuiada))
288           {
289             SDN.ActivarNavegacionGuiada("BibliotecaPlantaBaja");
290
291             SDN.DesactivarSelectorDestinos();
292           }
293
294         GUILayout.EndHorizontal();
295
296
297
298         GUILayout.BeginHorizontal();
299
300           if (GUILayout.Button(Traductor.Traducir("BotonBibliotecaPlantaUno"),
                estiloBotonDestino))
301           {
302             TransporteSDN.Destino("BibliotecaPlantaUno");
303           }
304
305         GUILayout.EndHorizontal();
306
307
308
309         GUILayout.BeginHorizontal();
310
311           if (GUILayout.Button(Traductor.Traducir("BotonAulaUno"), estiloBotonDestino))
312           {
313             TransporteSDN.Destino("AulaUno");
314           }
315
316           if (GUILayout.Button("", estiloBotonNavegacionGuiada))
317           {
318             SDN.ActivarNavegacionGuiada("AulaUno");
319
320             SDN.DesactivarSelectorDestinos();
321           }
322
323         GUILayout.EndHorizontal();
324
325
326
327         GUILayout.BeginHorizontal();
328
```

```
329             if (GUILayout.Button(Traductor.Traducir("BotonAulaDos"), estiloBotonDestino))
330             {
331               TransporteSDN.Destino("AulaDos");
332             }
333 
334             if (GUILayout.Button("", estiloBotonNavegacionGuiada))
335             {
336               SDN.ActivarNavegacionGuiada("AulaDos");
337 
338               SDN.DesactivarSelectorDestinos();
339             }
340 
341         GUILayout.EndHorizontal();
342 
343 
344 
345         GUILayout.BeginHorizontal();
346 
347             if (GUILayout.Button(Traductor.Traducir("BotonAulaTres"), estiloBotonDestino))
348             {
349               TransporteSDN.Destino("AulaTres");
350             }
351 
352             if (GUILayout.Button("", estiloBotonNavegacionGuiada))
353             {
354               SDN.ActivarNavegacionGuiada("AulaTres");
355 
356               SDN.DesactivarSelectorDestinos();
357             }
358 
359         GUILayout.EndHorizontal();
360 
361 
362 
363         GUILayout.BeginHorizontal();
364 
365             if (GUILayout.Button(Traductor.Traducir("BotonAulaCuatro"), estiloBotonDestino)
                    )
366             {
367               TransporteSDN.Destino("AulaCuatro");
368             }
369 
370             if (GUILayout.Button("", estiloBotonNavegacionGuiada))
371             {
372               SDN.ActivarNavegacionGuiada("AulaCuatro");
373 
374               SDN.DesactivarSelectorDestinos();
375             }
376 
377         GUILayout.EndHorizontal();
378 
379 
380 
381         GUILayout.BeginHorizontal();
382 
383             if (GUILayout.Button(Traductor.Traducir("BotonAulaDoscientoscuatro"),
                    estiloBotonDestino))
384             {
385               TransporteSDN.Destino("AulaDoscientoscuatro");
```

```
386           }
387
388           if (GUILayout.Button("", estiloBotonNavegacionGuiada))
389           {
390             SDN.ActivarNavegacionGuiada("AulaDoscientoscuatro");
391
392             SDN.DesactivarSelectorDestinos();
393           }
394
395         GUILayout.EndHorizontal();
396
397
398
399         GUILayout.BeginHorizontal();
400
401           if (GUILayout.Button(Traductor.Traducir("BotonAulaDoscientosdieciseis"),
                estiloBotonDestino))
402           {
403             TransporteSDN.Destino("AulaDoscientosdieciseis");
404           }
405
406           if (GUILayout.Button("", estiloBotonNavegacionGuiada))
407           {
408             SDN.ActivarNavegacionGuiada("AulaDoscientosdieciseis");
409
410             SDN.DesactivarSelectorDestinos();
411           }
412
413         GUILayout.EndHorizontal();
414
415
416
417         GUILayout.BeginHorizontal();
418
419           if (GUILayout.Button(Traductor.Traducir("BotonTecnologicoPlantaBaja"),
                estiloBotonDestino))
420           {
421             TransporteSDN.Destino("TecnologicoPlantaBaja");
422           }
423
424           if (GUILayout.Button("", estiloBotonNavegacionGuiada))
425           {
426             SDN.ActivarNavegacionGuiada("TecnologicoPlantaBaja");
427
428             // El siguiente bloque "if" le indica al intersectador, cuando el destino
429             // de la navegación guiada es la planta baja del edificio tecnológico y
430             // esta se activa estando en el ala oeste de la mencionada planta, que ya
431             // hemos llegado.
432
433             // Nota: este código es necesario por la particular distribución del edificio
434             // tecnológico y sólo necesario para el ala oeste.
435             if (SDN.dondeEstoy == "TecnologicoPlantaBajaOeste")
436             {
437               SDN.navegacionGuiadaForzarLlegada = true;
438             }
439
440             SDN.DesactivarSelectorDestinos();
441           }
442
```

```
443         GUILayout.EndHorizontal();
444
445
446
447         GUILayout.BeginHorizontal();
448
449           if (GUILayout.Button(Traductor.Traducir("BotonTecnologicoPlantaUno"),
                estiloBotonDestino))
450           {
451             TransporteSDN.Destino("TecnologicoPlantaUno");
452           }
453
454         GUILayout.EndHorizontal();
455
456
457
458         GUILayout.BeginHorizontal();
459
460           if (GUILayout.Button(Traductor.Traducir("BotonTecnologicoPlantaDos"),
                estiloBotonDestino))
461           {
462             TransporteSDN.Destino("TecnologicoPlantaDos");
463           }
464
465         GUILayout.EndHorizontal();
466
467
468
469         GUILayout.BeginHorizontal();
470
471           if (GUILayout.Button(Traductor.Traducir("BotonTecnologicoPlantaTres"),
                estiloBotonDestino))
472           {
473             TransporteSDN.Destino("TecnologicoPlantaTres");
474           }
475
476         GUILayout.EndHorizontal();
477
478
479
480         GUILayout.BeginHorizontal();
481
482           if (GUILayout.Button(Traductor.Traducir("BotonTecnologicoPlantaCuatroEste"),
                estiloBotonDestino))
483           {
484             TransporteSDN.Destino("TecnologicoPlantaCuatroEste");
485           }
486
487         GUILayout.EndHorizontal();
488
489
490
491         GUILayout.BeginHorizontal();
492
493           if (GUILayout.Button(Traductor.Traducir("BotonTecnologicoPlantaCuatroOeste"),
                estiloBotonDestino))
494           {
495             TransporteSDN.Destino("TecnologicoPlantaCuatroOeste");
496           }
```

```
497
498         GUILayout.EndHorizontal();
499
500
501
502         GUILayout.BeginHorizontal();
503
504           if (GUILayout.Button(Traductor.Traducir("BotonTecnologicoPlantaBajaAseoChicas")
                , estiloBotonDestino))
505           {
506             TransporteSDN.Destino("AseoChicasTecnologicoPlantaBaja");
507           }
508
509           if (GUILayout.Button("", estiloBotonNavegacionGuiada))
510           {
511             SDN.ActivarNavegacionGuiada("TecnologicoBanoChicasPlantaBaja");
512
513             SDN.DesactivarSelectorDestinos();
514           }
515
516         GUILayout.EndHorizontal();
517
518
519
520         GUILayout.BeginHorizontal();
521
522           if (GUILayout.Button(Traductor.Traducir("BotonAulaHacheTres"),
                estiloBotonDestino))
523           {
524             TransporteSDN.Destino("AulaHacheTres");
525           }
526
527
528           if (GUILayout.Button("", estiloBotonNavegacionGuiada))
529           {
530             SDN.ActivarNavegacionGuiada("AulaHacheTres");
531
532             SDN.DesactivarSelectorDestinos();
533           }
534
535         GUILayout.EndHorizontal();
536
537
538
539         GUILayout.BeginHorizontal();
540
541           if (GUILayout.Button(Traductor.Traducir("BotonEscuelaExteriorEntrada"),
                estiloBotonDestino))
542           {
543             TransporteSDN.Destino("ExteriorEntrada");
544           }
545
546         GUILayout.EndHorizontal();
547
548
549
550       GUILayout.EndScrollView ();
551
552
```

```
553
554         // SEPARADOR
555
556         GUILayout.BeginHorizontal();
557
558           GUILayout.BeginVertical();
559
560             GUILayout.Space(25);
561
562           GUILayout.EndVertical();
563
564         GUILayout.EndHorizontal();
565
566
567
568         // BLOQUE DE INSTRUCCIONES
569
570         posicionScrollInstrucciones = GUILayout.BeginScrollView(posicionScrollInstrucciones
                , GUILayout.Width(ancho), GUILayout.Height(alto / 3));
571
572
573           GUILayout.BeginHorizontal();
574
575
576             GUILayout.Label(iconoBrujula, estiloIconoInstrucciones, GUILayout.Height(32));
577
578             GUILayout.Label(Traductor.Traducir("InstruccionesSelectorDestino"),
                    estiloInstrucciones);
579
580
581           GUILayout.EndHorizontal();
582
583
584
585           GUILayout.BeginHorizontal();
586
587             GUILayout.BeginVertical();
588
589               GUILayout.Space(15);
590
591             GUILayout.EndVertical();
592
593           GUILayout.EndHorizontal();
594
595
596
597           GUILayout.BeginHorizontal();
598
599
600             GUILayout.Label(iconoLocalizadorNormal, estiloIconoSubInstrucciones, GUILayout.
                    Height(32));
601
602             GUILayout.Label(GUIContent(Traductor.Traducir("InstruccionesLocalizadorNormal")
                    ), estiloSubInstrucciones);
603
604
605           GUILayout.EndHorizontal();
606
607
```

```
608
609         GUILayout.BeginHorizontal();
610
611
612           GUILayout.Label(iconoLocalizadorBajar, estiloIconoSubInstrucciones, GUILayout.
                  Height(32));
613
614           GUILayout.Label(GUIContent(Traductor.Traducir("InstruccionesLocalizadorBajar"))
                  , estiloSubInstrucciones);
615
616
617         GUILayout.EndHorizontal();
618
619
620
621         GUILayout.BeginHorizontal();
622
623
624           GUILayout.Label(iconoLocalizadorSubir, estiloIconoSubInstrucciones, GUILayout.
                  Height(32));
625
626           GUILayout.Label(GUIContent(Traductor.Traducir("InstruccionesLocalizadorSubir"))
                  , estiloSubInstrucciones);
627
628
629         GUILayout.EndHorizontal();
630
631
632
633       GUILayout.EndScrollView ();
634
635
636
637     GUILayout.EndVertical();
638
639
640
641   GUILayout.EndArea();
642
643 }
```

F.18. InterfazIdiomasControl.js

```
1  #pragma strict
2
3  /*#####################################################################
4
5  InterfazIdiomasControl.js
6  -------------------------
7
8  Guión encargado de gestionar la posición y aspecto de los elementos
9  del menú de selección de idioma.
10
11 #####################################################################*/
12
13
14 // Valores configurables indicadores de la anchura y la altura del menú
```

```
15 // selector de destino.
16
17 var porcentajeAncho : float;
18 var porcentajeAlto : float;
19
20 var porcentajeBotonRegresar : float;
21 var porcentajeMargenBotonRegresar : float;
22 var porcentajeBandera : float;
23 var texturaFondo : Texture;
24
25 var estiloFondo : GUIStyle;
26 var estiloBotonRegresar : GUIStyle;
27
28 var estiloBotonBanderaEspana : GUIStyle;
29 var estiloBotonBanderaEspanaElegida : GUIStyle;
30 var estiloBotonBanderaEstadosUnidos : GUIStyle;
31 var estiloBotonBanderaEstadosUnidosElegida : GUIStyle;
32 var estiloBotonBanderaAlemania : GUIStyle;
33 var estiloBotonBanderaAlemaniaElegida : GUIStyle;
34
35
36 private var ancho : float;
37 private var alto : float;
38 private var margenAncho : float;
39 private var margenAlto : float;
40 private var alturaPantalla : float;
41 private var anchuraPantalla : float;
42 private var tamanoBotonRegresar : float;
43 private var margenBotonRegresar : float;
44 private var tamanoBandera : float;
45
46 private var posicionScroll : Vector2;
47 private var posicionScrollInstrucciones : Vector2;
48
49 private var banderaEspanaElegida : boolean = false;
50 private var banderaEstadosUnidosElegida : boolean = false;
51 private var banderaAlemaniaElegida : boolean = false;
52
53
54 function Start()
55 {
56
57   if (porcentajeAncho <= 0 || porcentajeAncho > 100)
58   {
59     Debug.Log("La anchura indicada para el menú selector de destinos está fuera de rango,
          se utilizará el valor predeterminado \"40\".");
60     porcentajeAncho = 40;
61   }
62
63   if (porcentajeAlto <= 0 || porcentajeAlto > 100)
64   {
65     Debug.Log("La altura indicada para el menú selector de destinos está fuera de rango,
         se utilizará el valor predeterminado \"80\".");
66     porcentajeAlto = 80;
67   }
68
69   if (porcentajeBotonRegresar <= 0 || porcentajeBotonRegresar > 100)
70   {
```

```
71      Debug.Log("El tamaño del botón <i>regresar</i> indicado está fuera de rango, se
            utilizará el valor predeterminado \"8\".");
72      porcentajeBotonRegresar = 8;
73    }
74
75    if (porcentajeMargenBotonRegresar <= 0 || porcentajeMargenBotonRegresar > 100)
76    {
77      Debug.Log("El porcentaje de margen indicado entre la pantalla y el botón <i>regresar
            </i> está fuera de rango, se utilizará el valor predeterminado \"2\".");
78      porcentajeMargenBotonRegresar = 2;
79    }
80
81    if (porcentajeBandera <= 0 || porcentajeBandera > 100)
82    {
83      Debug.Log("El porcentaje indicado para el tamaño de las <i>banderas</i> está fuera de
             rango, se utilizará el valor predeterminado \"2\".");
84      porcentajeBandera = 2;
85    }
86
87
88    anchuraPantalla = Screen.width;
89    alturaPantalla = Screen.height;
90
91    ancho = porcentajeAncho / 100 * anchuraPantalla;
92    alto = porcentajeAlto / 100 * alturaPantalla;
93
94    margenAncho = (anchuraPantalla - ancho) / 2;
95    margenAlto = (alturaPantalla - alto) / 2;
96
97    tamanoBotonRegresar = porcentajeBotonRegresar / 100 * alturaPantalla;
98    margenBotonRegresar = porcentajeMargenBotonRegresar / 100 * alturaPantalla;
99
100   // Definimos el tamaño de las banderas en función del espacio disponible definido
101   // y no del total de la pantalla.
102
103   tamanoBandera = porcentajeBandera / 100 * alto;
104
105 }
106
107
108 function Update ()
109 {
110
111   if (Screen.width != anchuraPantalla || Screen.height != alturaPantalla)
112   {
113
114     alturaPantalla = Screen.height;
115     anchuraPantalla = Screen.width;
116
117     ancho = porcentajeAncho / 100 * anchuraPantalla;
118     alto = porcentajeAlto / 100 * alturaPantalla;
119
120     margenAncho = (anchuraPantalla - ancho) / 2;
121     margenAlto = (alturaPantalla - alto) / 2;
122
123     tamanoBotonRegresar = porcentajeBotonRegresar / 100 * alturaPantalla;
124     margenBotonRegresar = porcentajeMargenBotonRegresar / 100 * alturaPantalla;
125
126     // Definimos el tamaño de las banderas en función del espacio disponible definido
```

```
127     // y no del total de la pantalla.
128
129     tamanoBandera = porcentajeBandera / 100 * alto;
130
131   }
132
133 }
134
135
136 function OnGUI ()
137 {
138
139   // Textura de fondo.
140
141   GUI.DrawTexture(Rect(margenAncho, margenAlto, ancho, alto), texturaFondo);
142
143
144   if (GUI.Button(Rect(Screen.width - tamanoBotonRegresar - margenBotonRegresar,
          margenBotonRegresar, tamanoBotonRegresar, tamanoBotonRegresar), "",
          estiloBotonRegresar))
145   {
146     SDN.DesactivarSelectorIdiomas();
147   }
148
149
150   GUILayout.BeginArea(Rect(margenAncho, margenAlto, ancho, alto));
151
152
153     GUILayout.BeginHorizontal();
154
155
156       GUILayout.BeginVertical();
157
158         // Las banderas poseen cuatro estados:
159         //
160         // Activo: el que adquieren cuando el usuario pasa el ratón por encima.
161         // Pulsado: el que adquieren cuando el usuario pulsa sobre ellas.
162         // Elegido: el que poseen cuando se corresponden con el idioma actual.
163         // Normal: el que poseen si no están activadas, pulsadas o elegidas.
164
165         // Los estados Normal, Activo y Pulsado se definen partiendo del
166         // GUIStyle "estiloBotonBanderaEspana"; el estado elegido se
167         // define a través del GUIStyle "estiloBotonBanderaEspanaElegida".
168
169         if (SDN.idioma != "espanol")
170         {
171           if (GUILayout.Button("", estiloBotonBanderaEspana))
172           {
173             SDN.idioma = "espanol";
174             SDN.DesactivarSelectorIdiomas();
175           }
176         }
177         else
178         {
179           if (GUILayout.Button("", estiloBotonBanderaEspanaElegida))
180           {
181             SDN.idioma = "espanol";
182             SDN.DesactivarSelectorIdiomas();
183           }
```

```
184         }
185
186       GUILayout.EndVertical();
187
188
189       GUILayout.BeginVertical();
190
191         if (SDN.idioma != "ingles")
192         {
193           if (GUILayout.Button("", estiloBotonBanderaEstadosUnidos))
194           {
195             SDN.idioma = "ingles";
196             SDN.DesactivarSelectorIdiomas();
197           }
198         }
199         else
200         {
201           if (GUILayout.Button("", estiloBotonBanderaEstadosUnidosElegida))
202           {
203             SDN.idioma = "ingles";
204             SDN.DesactivarSelectorIdiomas();
205           }
206         }
207
208       GUILayout.EndVertical();
209
210
211       GUILayout.BeginVertical();
212
213         if (SDN.idioma != "aleman")
214         {
215           if (GUILayout.Button("", estiloBotonBanderaAlemania))
216           {
217             SDN.idioma = "aleman";
218             SDN.DesactivarSelectorIdiomas();
219           }
220         }
221         else
222         {
223           if (GUILayout.Button("", estiloBotonBanderaAlemaniaElegida))
224           {
225             SDN.idioma = "aleman";
226             SDN.DesactivarSelectorIdiomas();
227           }
228         }
229
230       GUILayout.EndVertical();
231
232
233     GUILayout.EndHorizontal();
234
235
236   GUILayout.EndArea();
237
238 }
```

F.19. InterfazMapaControl.js

```
#pragma strict

/*#####################################################################

InterfazMapaControl.js
----------------------

Guión encargado de la gestión de la interfaz del mapa cuando esté está cerrado
o es estado normal (minimapa).

Controla el aspecto, posición y funcionalidad de todos los elementos de la interfaz
del mapa: marco, puntos cardinales, botones de cambio de posición, botones de zoom,
navegador, botón de bloqueo de rotación, botones de cierre y maximización, botón de
reapertura del mapa, botones generales de selección de modo de vista, idioma, apagado
e información de ubicación; y botón para desactivar la navegación guiada cuando esta
esté activada.

Todos los elementos cambian su posición y tamaño en funcion de la altura de la pantalla
permitiendo así su reajuste a los cambios de resolución, también en tiempo real.

#####################################################################*/


var norte : Texture2D;
var este : Texture2D;
var sur : Texture2D;
var oeste : Texture2D;

var bordeMapa : Texture2D;

// La siguiente variable gestiona el tamaño de los iconos representativos de los puntos
// cardinales, se calcula como un porcentaje de la mitad del lado del mapa, y su valor
// sirve como base para el resto de los iconos del minimapa.
var porcentajePuntosCardinales : float;
var factorMasMenos : float;
var factorCandado : float;
var factorCerrarMapa : float;
var factorNavegador : float;

var porcentajeBotonAbrirMapa : float;
private var tamanoBotonAbrirMapa : float;
var porcentajeMargenBotonAbrirMapa : float;
private var margenBotonAbrirMapa : float;

var porcentajeBotonesGenerales : float;
// Para no hacer excesivamente largas las definiciones de posición de los botones
    generales
// usamos "TG" en lugar de "tamanoBotonesGenerales".
private var TG : float;
var porcentajeMargenBotonesGenerales : float;
private var margenBotonesGenerales : float;

private var anchuraCamara : float;
private var alturaCamara : float;

// La siguiente variable determina la ubicación del mapa en una de las cuatro esquinas.
```

```
56  // Trabajamos en principio para la esquina superior derecha.
57
58  private var posicionMapa : String;
59
60  // La siguiente variable pública permite determinar desde el inspector si se desea que el
        usuario
61  // pueda cambiar la posición del mapa durante la ejecución del programa.
62
63  private var permitirCambiarPosicion : boolean;
64
65  private var camaraMapa : GameObject;
66  private var dimensionesCamaraMapa : Rect;
67  private var bloqueoMapa : boolean;
68
69  @HideInInspector var actualizarInterfaz : boolean;
70
71  // Declaraciones de estilos (a modificar a través del Inspector)
72
73  var estiloBotonMas : GUIStyle;
74  var estiloBotonMenos : GUIStyle;
75  var estiloCandadoCerrado : GUIStyle;
76  var estiloCandadoAbierto : GUIStyle;
77  var estiloCerrarMapa : GUIStyle;
78  var estiloAbrirMapa : GUIStyle;
79  var estiloMaximizarMapa : GUIStyle;
80  var estiloMinimizarMapa : GUIStyle;
81  var estiloPosicion1 : GUIStyle;
82  var estiloPosicionActivada1 : GUIStyle;
83  var estiloPosicion2 : GUIStyle;
84  var estiloPosicionActivada2 : GUIStyle;
85  var estiloPosicion3 : GUIStyle;
86  var estiloPosicionActivada3 : GUIStyle;
87  var estiloPosicion4 : GUIStyle;
88  var estiloPosicionActivada4 : GUIStyle;
89  var estiloNavegador : GUIStyle;
90  var estiloVistaOrbital : GUIStyle;
91  var estiloVistaTeclado : GUIStyle;
92  var estiloCambioIdioma : GUIStyle;
93  var estiloDondeEstoy : GUIStyle;
94  var estiloApagado : GUIStyle;
95  var estiloCancelarNavegacionGuiada : GUIStyle;
96
97  // Declaración de variables internas para el cálculo de la posición de los elementos
98
99  private var TPC : float;
100 private var MITPC : float;
101 private var TM : float;
102 private var MITM : float;
103 private var TC : float;
104 private var MITC : float;
105 private var TCM : float;
106 private var MITCM : float;
107 private var TAM : float;
108 private var MITAM : float;
109 private var TN : float;
110 private var MITN : float;
111 private var mitadMitadAnchura : float;
112 private var mitadAnchura : float;
113 private var mitadAltura : float;
```

```
114
115 private var interfaz : GameObject;
116
117 // Variables para el control de la interfaz general
118
119 private var anchuraPantalla : float;
120 private var alturaPantalla : float;
121 private var relacionDeAspecto : float;
122
123
124 function Start ()
125 {
126   // Localizamos los "GameObject" clonados al cargar la escena tanto de la cámara del
        mapa como de su interfaz.
127
128   camaraMapa = SDN.camaraMapa;
129   interfaz = SDN.interfazMapa;
130
131
132   // ---------------------------------
133   // CONTROL DE LOS DATOS INTRODUCIDOS
134   // ---------------------------------
135
136   // Comprobación de que los valores introducidos en el Inspector son adecuados, y en
        caso contrario se
137   // asignan unos predeterminados.
138
139   if (porcentajePuntosCardinales <= 0 || porcentajePuntosCardinales > 100)
140   {
141     porcentajePuntosCardinales = 13.5;
142     Debug.Log("El porcentaje de tamaño de los puntos cardinales indicado está fuera de
          rango, se utilizará el valor predeterminado.");
143   }
144
145   // Factores de tamaño de los elementos de la interfaz.
146
147   // Toman como referencia el tamaño de los puntos cardinales.
148
149   if (factorCandado <= 0 || factorCandado > 100)
150   {
151     factorCandado = 1.1;
152     Debug.Log("El tamaño del icono \"Candado\" indicado está fuera de rango, se utilizará
           el valor predeterminado.");
153   }
154
155   if (factorMasMenos <= 0 || factorMasMenos > 100)
156   {
157     factorMasMenos = 0.8;
158     Debug.Log("l tamaño de los iconos \"Más y Menos\" indicados están fuera de rango, se
          utilizará el valor predeterminado.");
159
160   }
161
162   if (factorCerrarMapa <= 0 || factorCerrarMapa > 100)
163   {
164     factorCerrarMapa = 1;
165     Debug.Log("l tamaño del icono \"Cerrar mapa\" indicado está fuera de rango, se
          utilizará el valor predeterminado.");
166
```

```
167   }
168
169   if (factorNavegador <= 0 || factorNavegador > 100)
170   {
171     factorNavegador = 4;
172     Debug.Log("El tamaño del icono \"Navegador\" indicado está fuera de rango, se
          utilizará el valor predeterminado.");
173
174   }
175
176   if (porcentajeBotonAbrirMapa <= 0 || porcentajeBotonAbrirMapa > 100)
177   {
178     Debug.Log("El porcentaje indicado para el icono <i>abrir mapa</i> está fuera de rango
          , se utilizará el valor predeterminado \"8\".");
179     porcentajeBotonAbrirMapa = 8;
180   }
181
182   if (porcentajeMargenBotonAbrirMapa <= 0 || porcentajeMargenBotonAbrirMapa > 100)
183   {
184     Debug.Log("El porcentaje de margen indicado para el icono <i>abrir mapa</i> está
          fuera de rango, se utilizará el valor predeterminado \"2\".");
185     porcentajeMargenBotonAbrirMapa = 2;
186   }
187
188   if (porcentajeBotonesGenerales <= 0 || porcentajeBotonesGenerales > 100)
189   {
190     Debug.Log("El porcentaje indicado para los iconos \"Generales\" está fuera de rango,
          se utilizará el valor predeterminado \"3\".");
191     porcentajeBotonesGenerales = 3;
192   }
193
194   if (porcentajeMargenBotonesGenerales <= 0 || porcentajeMargenBotonesGenerales > 100)
195   {
196     Debug.Log("El porcentaje de margen indicado para los <i>botones generales</i> está
          fuera de rango, se utilizará el valor predeterminado \"2\".");
197     porcentajeMargenBotonesGenerales = 2;
198   }
199
200
201   // ------------------------------------------------
202   // GESTIóN DEL COMPORTAMIENTO Y ASPECTO DE LA INTERFAZ
203   // ------------------------------------------------
204
205   actualizarInterfaz = false;
206
207   // Se comprueba si se ha activado desde el Inspector la opción de permitir al usuario
          cambiar la
208   // posición del mapa. (En caso negativo los botones de cambio de posición desaparecerán
          durante la
209   // ejecución del programa).
210
211   permitirCambiarPosicion = camaraMapa.GetComponent.<CamaraMapaControl>().
          permitirCambiarPosicion;
212
213
214   // ------------------------------------------------
215   // POSICIÓN Y ASPECTO DE LOS ELEMENTOS DE LA INTERFAZ
216   // ------------------------------------------------
217
```

```
218  // Cálculos correspondientes para la ubicación de los elementos.
219
220  // La coordenada (0, 0) de los elementos "GUI" se corresponde con la esquina superior
         izquierda
221  // de la pantalla.
222
223  // Nota: esto difiere, por ejemplo, con la componente "rect" de una cámara, cuyo (0, 0)
         se corresponde
224  // con la esquina inferior izquierda.
225
226  // Los elementos "GUI" se posicionan en función de la longitud total en píxeles de los
         lados de la pantalla
227  // y no en valores de 0 a 1 como otros componentes.
228
229  alturaPantalla = Screen.height;
230  anchuraPantalla = Screen.width;
231
232  // Si el mapa está activado...
233  if (SDN.camaraMapa.activeInHierarchy == true)
234  {
235
236    dimensionesCamaraMapa = camaraMapa.GetComponent.<Camera>().pixelRect;
237    alturaCamara = dimensionesCamaraMapa.width;
238    anchuraCamara = dimensionesCamaraMapa.height;
239    posicionMapa = camaraMapa.GetComponent.<CamaraMapaControl>().posicionMapa;
240
241    mitadAnchura = dimensionesCamaraMapa.width / 2;
242    mitadAltura = dimensionesCamaraMapa.height / 2;
243
244    // PUNTOS CARDINALES:
245
246    // Su tamaño se calculará de forma dinámica como una relación entre la mitad del lado
           del cuadro de visión de la
247    // cámara del minimapa y un valor indicado manualmente en la variable "
           porcentajePuntosCardinales" que permitirá
248    // hacerlos más o menos grandes de forma sencilla.
249
250    // TPC = Tamaño Puntos Cardinales
251    // MTPC = Mitad Tamaño Puntos Cardinales
252
253    //TPC = mitadAnchura / porcentajePuntosCardinales;
254    TPC = mitadAnchura * porcentajePuntosCardinales / 100;
255
256    MTPC = TPC/ 2;
257
258    // El tamaño del resto de los indicadores del mapa se calcula como una relación entre
           un factor predefinido y el tamaño de los
259    // puntos cardinales.
260
261    // INDICADORES MÁS Y MENOS:
262
263    // TM = Tamaño Más/Menos
264    // MTM = Mitad Tamaño Candado
265
266    TM = TPC * factorMasMenos;
267    MTM = TM / 2;
268
269    // Usamos la variable "mitadMitadAnchura" como referencia indicadora de dónde colocar
           los botones más y menos más en la mitad
```

```
270     // izquiera del lado inferior del minimapa. Dividiendo por 2 los indicadores se
            ubicarían en la mitad, haciéndolo por un valor
271     // mayor los colocaría más a la izquierda, y por uno menor más a la derecha.
272
273     mitadMitadAnchura = mitadAnchura / 3;
274
275     // CANDADO DE BLOQUEO:
276
277     // TC = Tamaño Candado
278     // MTC = Mitad Tamaño Candado
279
280     TC = TPC * factorCandado;
281     MTC = TC / 2;
282
283     // CERRAR MAPA
284
285     // TCM = Tamaño Cerrar Mapa
286     // MTCM = Mitad Tamaño Cerrar Mapa
287
288     TCM = TPC * factorCerrarMapa;
289     MTCM = TC / 2;
290
291     // NAVEGADOR
292
293     // TN = Tamaño Navegador
294     // MTN = Mitad Tamaño Navegador
295
296     TN = TPC * factorNavegador;
297     MTN = TN / 2;
298
299   }
300
301   // ICONOS GENERALES (p. e. cambio de vista; su tamaño no es
302   // dependiente de los iconos del mapa).
303
304   tamanoBotonAbrirMapa = porcentajeBotonAbrirMapa / 100 * alturaPantalla;
305   margenBotonAbrirMapa = porcentajeMargenBotonAbrirMapa / 100 * alturaPantalla;
306
307   TG = porcentajeBotonesGenerales / 100 * alturaPantalla;
308   margenBotonesGenerales = porcentajeMargenBotonesGenerales / 100 * alturaPantalla;
309
310 }
311
312 function OnEnable ()
313 {
314   // Se consulta el estado de la orientación del mapa para saber si se han de dibujar o
          no los puntos
315   // cardinales y dibujar el icono representativo de ésta apropiado (candado abierto o
          cerrado).
316
317   bloqueoMapa = SDN.bloqueoMapa;
318   actualizarInterfaz = true;
319 }
320
321
322 function Update ()
323 {
324   if (Screen.width != anchuraPantalla || Screen.height != alturaPantalla ||
          actualizarInterfaz == true)
```

```
325  {
326      alturaPantalla = Screen.height;
327      anchuraPantalla = Screen.width;
328
329      // Si el mapa está activado...
330      if (SDN.camaraMapa.activeInHierarchy == true)
331      {
332
333          dimensionesCamaraMapa = camaraMapa.GetComponent.<Camera>().pixelRect;
334          alturaCamara = dimensionesCamaraMapa.width;
335          anchuraCamara = dimensionesCamaraMapa.height;
336          posicionMapa = camaraMapa.GetComponent.<CamaraMapaControl>().posicionMapa;
337
338          mitadAnchura = dimensionesCamaraMapa.width / 2;
339          mitadAltura = dimensionesCamaraMapa.height / 2;
340
341          // PUNTOS CARDINALES:
342
343          // Su tamaño se calculará de forma dinámica como una relación entre el lado del
                cuadro de visión de la cámara del minimapa y
344          // un valor indicado manualmente en la variable "porcentajePuntosCardinales" que
                permitirá hacerlos más o menos grandes de
345          // forma sencilla.
346
347          // TPC = Tamaño Puntos Cardinales
348          // MTPC = Mitad Tamaño Puntos Cardinales
349
350          //TPC = mitadAnchura / porcentajePuntosCardinales;
351          TPC = mitadAnchura * porcentajePuntosCardinales / 100;
352          MTPC = TPC/ 2;
353
354          // El tamaño del resto de los indicadores del mapa se calcula como una relación
                entre un factor predefinido y el tamaño de los
355          // puntos cardinales.
356
357          // INDICADORES MÁS Y MENOS:
358
359          // TM = Tamaño Más/Menos
360          // MTM = Mitad Tamaño Candado
361
362          TM = TPC * factorMasMenos;
363          MTM = TM / 2;
364
365          // Usamos la variable "mitadMitadAnchura" como referencia indicadora de dónde
                colocar los botones más y menos más en la mitad
366          // izquiera del lado inferior del minimapa. Dividiendo por 2 los indicadores se
                ubicarían en la mitad, haciéndolo por un valor
367          // mayor los colocaría más a la izquierda, y por uno menor más a la derecha.
368
369          mitadMitadAnchura = mitadAnchura / 3;
370
371          // CANDADO DE BLOQUEO:
372
373          // TC = Tamaño Candado
374          // MTC = Mitad Tamaño Candado
375
376          TC = TPC * factorCandado;
377          MTC = TC / 2;
378
```

```
379       // CERRAR MAPA
380
381       // TCM = Tamaño Cerrar Mapa
382       // MTCM = Mitad Tamaño Cerrar Mapa
383
384       TCM = TPC * factorCerrarMapa;
385       MTCM = TC / 2;
386
387       // NAVEGADOR
388
389       // TN = Tamaño Navegador
390       // MTN = Mitad Tamaño Navegador
391
392       TN = TPC * factorNavegador;
393       MTN = TN / 2;
394
395     }
396
397     // ICONOS GENERALES (p. e. cambio de vista; su tamaño no es
398     // dependiente de los iconos del mapa).
399
400     tamanoBotonAbrirMapa = porcentajeBotonAbrirMapa / 100 * alturaPantalla;
401     margenBotonAbrirMapa = porcentajeMargenBotonAbrirMapa / 100 * alturaPantalla;
402
403     TG = porcentajeBotonesGenerales / 100 * alturaPantalla;
404     margenBotonesGenerales = porcentajeMargenBotonesGenerales / 100 * alturaPantalla;
405   }
406
407 }
408
409
410 function OnGUI ()
411 {
412   if (SDN.mapaAbiertoCerrado == "Abierto")
413   {
414     // Dibujo del borde que servirá de marco para el cuadro de visión de la cámara del
          mapa.
415
416     GUI.DrawTexture(Rect(dimensionesCamaraMapa.x, alturaPantalla - dimensionesCamaraMapa.
          y - alturaCamara, dimensionesCamaraMapa.width, dimensionesCamaraMapa.height),
          bordeMapa);
417
418     // Dibujo de los puntos cardinales.
419
420     // Nota: si se optara por incluir los puntos cardinales como parte intrínseca del
          borde, si bien el código para representar un solo
421     // objeto sería mucho más sencillo que el que hace falta para representarlos todos
          individualmente, limitaría enormemente las posibilidades
422     // de personalización.
423
424     // Se dibujarán sólo cuando la orientación del mapa esté bloqueada.
425
426     if (bloqueoMapa == true)
427     {
428       GUI.DrawTexture(Rect(dimensionesCamaraMapa.x + mitadAnchura - MTPC, alturaPantalla
            - dimensionesCamaraMapa.y - alturaCamara - MTPC, TPC, TPC), norte);
429       GUI.DrawTexture(Rect(dimensionesCamaraMapa.x + dimensionesCamaraMapa.width - MTPC,
            alturaPantalla - dimensionesCamaraMapa.y - mitadAltura - MTPC, TPC, TPC), este)
            ;
```

```
430       GUI.DrawTexture(Rect(dimensionesCamaraMapa.x + mitadAnchura - MTPC, alturaPantalla
              - dimensionesCamaraMapa.y - MTPC, TPC, TPC), sur);
431       GUI.DrawTexture(Rect(dimensionesCamaraMapa.x - MTPC, alturaPantalla -
              dimensionesCamaraMapa.y - mitadAltura - MTPC, TPC, TPC), oeste);
432     }
433
434     // Dibujo del los indicadores MÁS Y MENOS.
435
436     if (GUI.Button(Rect(dimensionesCamaraMapa.x + mitadMitadAnchura - TM - MTM,
            alturaPantalla - dimensionesCamaraMapa.y - MTM, TM, TM), "", estiloBotonMenos))
437     {
438       SDN.terrenoVisto *= 1.5;
439
440     }
441     if (GUI.Button(Rect(dimensionesCamaraMapa.x + mitadMitadAnchura + MTM, alturaPantalla
            - dimensionesCamaraMapa.y - MTM, TM, TM), "", estiloBotonMas))
442     {
443       SDN.terrenoVisto /= 1.5;
444
445     }
446
447     // Dibujo del CANDADO DE BLOQUEO.
448
449     if (bloqueoMapa == true)
450     {
451       if (GUI.Button(Rect(dimensionesCamaraMapa.x + dimensionesCamaraMapa.width - MTC,
              alturaPantalla - dimensionesCamaraMapa.y - MTC, TC, TC), "",
              estiloCandadoCerrado))
452       {
453         bloqueoMapa = false;
454         // Asignamos también la variable global para que el guión "CamraMapaControl.js"
                pueda gestionar el estado
455         // de la orientación entre escenas.
456         SDN.bloqueoMapa = false;
457       }
458     }
459     else
460     {
461       if (GUI.Button(Rect(dimensionesCamaraMapa.x + dimensionesCamaraMapa.width - MTC,
              alturaPantalla - dimensionesCamaraMapa.y - MTC, TC, TC), "",
              estiloCandadoAbierto))
462       {
463         bloqueoMapa = true;
464         SDN.bloqueoMapa = true;
465       }
466
467     }
468
469     // Dibujo del icono CERRAR MAPA.
470
471     if (GUI.Button(Rect(dimensionesCamaraMapa.x + dimensionesCamaraMapa.width - TCM,
            alturaPantalla - dimensionesCamaraMapa.y - alturaCamara, TCM, TCM), "",
            estiloCerrarMapa))
472     {
473       camaraMapa.SetActive(false);
474
475       SDN.mapaAbiertoCerrado = "Cerrado";
476     }
477
```

```
478     // Dibujo del icono MAXIMIZAR MAPA (mismas dimensiones que Cerrar Mapa: TCM y MTCM).
479
480     if (GUI.Button(Rect(dimensionesCamaraMapa.x + dimensionesCamaraMapa.width - TCM,
            alturaPantalla - dimensionesCamaraMapa.y - alturaCamara + TCM + MTCM, TCM, TCM),
            "", estiloMaximizarMapa))
481     {
482       camaraMapa.GetComponent.<CamaraMapaControl>().pulsadoConmutadorMaximizar = true;
483     }
484
485     // El dibujo de los iconos que permiten cambiar la posición del mapa sólo se realiza
            si se ha activado
486     // la opción "Cambiar Posición" en el objeto "CamaraMapa" desde el Inspector.
487
488     // Tienen las mismas dimensiones que los iconos de cerrar y maximizar mapa (TCM y
            MTCM).
489
490     if (permitirCambiarPosicion == true)
491     {
492       if(camaraMapa.GetComponent.<CamaraMapaControl>().posicionMapa == "ArribaIzquierda")
493       {
494         GUI.Label(Rect(dimensionesCamaraMapa.x + MTCM, alturaPantalla -
                dimensionesCamaraMapa.y - alturaCamara - TCM + MTCM, TCM, TCM), "",
                estiloPosicionActivada1);
495       }
496       else
497       {
498         if (GUI.Button(Rect(dimensionesCamaraMapa.x + MTCM, alturaPantalla -
                dimensionesCamaraMapa.y - alturaCamara - TCM + MTCM, TCM, TCM), "",
                estiloPosicion1))
499         {
500           camaraMapa.GetComponent.<CamaraMapaControl>().posicionMapa = "ArribaIzquierda";
501           camaraMapa.GetComponent.<CamaraMapaControl>().posicionCambiada = true;
502           SDN.posicionMapa = "ArribaIzquierda";
503         }
504       }
505
506       if(camaraMapa.GetComponent.<CamaraMapaControl>().posicionMapa == "ArribaDerecha")
507       {
508         GUI.Label(Rect(dimensionesCamaraMapa.x + MTCM + TCM, alturaPantalla -
                dimensionesCamaraMapa.y - alturaCamara - TCM + MTCM, TCM, TCM), "",
                estiloPosicionActivada2);
509       }
510       else
511       {
512         if (GUI.Button(Rect(dimensionesCamaraMapa.x + MTCM + TCM, alturaPantalla -
                dimensionesCamaraMapa.y - alturaCamara - TCM + MTCM, TCM, TCM), "",
                estiloPosicion2))
513         {
514           camaraMapa.GetComponent.<CamaraMapaControl>().posicionMapa = "ArribaDerecha";
515           camaraMapa.GetComponent.<CamaraMapaControl>().posicionCambiada = true;
516           SDN.posicionMapa = "ArribaDerecha";
517         }
518       }
519
520       /*
521       // Nota: los siguientes bloques de código se corresponden con las posiciones
              inferiorer a la derecha y a la izquierda.
522       // Están desactivados por ocupar este espacio otros elementos de la interfaz.
523
```

```
524       if(camaraMapa.GetComponent.<CamaraMapaControl>().posicionMapa == "AbajoIzquierda")
525       {
526         GUI.Label(Rect(dimensionesCamaraMapa.x + MTCM + TCM + TCM, alturaPantalla -
              dimensionesCamaraMapa.y - alturaCamara - TCM + MTCM, TCM, TCM), "",
              estiloPosicionActivada3);
527       }
528       else
529       {
530         if (GUI.Button(Rect(dimensionesCamaraMapa.x + MTCM + TCM + TCM, alturaPantalla -
              dimensionesCamaraMapa.y - alturaCamara - TCM + MTCM, TCM, TCM), "",
              estiloPosicion3))
531         {
532           camaraMapa.GetComponent.<CamaraMapaControl>().posicionMapa = "AbajoIzquierda";
533           camaraMapa.GetComponent.<CamaraMapaControl>().posicionCambiada = true;
534           SDN.posicionMapa = "AbajoIzquierda";
535         }
536       }
537
538       if(camaraMapa.GetComponent.<CamaraMapaControl>().posicionMapa == "AbajoDerecha")
539       {
540         GUI.Button(Rect(dimensionesCamaraMapa.x + MTCM + TCM + TCM + TCM, alturaPantalla
              - dimensionesCamaraMapa.y - alturaCamara - TCM + MTCM, TCM, TCM), "",
              estiloPosicionActivada4);
541       }
542       else
543       {
544         if (GUI.Button(Rect(dimensionesCamaraMapa.x + MTCM + TCM + TCM + TCM,
              alturaPantalla - dimensionesCamaraMapa.y - alturaCamara - TCM + MTCM, TCM,
              TCM), "", estiloPosicion4))
545         {
546           camaraMapa.GetComponent.<CamaraMapaControl>().posicionMapa = "AbajoDerecha";
547           camaraMapa.GetComponent.<CamaraMapaControl>().posicionCambiada = true;
548           SDN.posicionMapa = "AbajoDerecha";
549         }
550       }
551       */
552
553     }
554
555     // Dibujo del icono NAVEGADOR.
556
557     if (GUI.Button(Rect(dimensionesCamaraMapa.x + dimensionesCamaraMapa.width -
          mitadMitadAnchura - TN / 1.13, alturaPantalla - dimensionesCamaraMapa.y - MTN, TN
           * 1.2, TN), "", estiloNavegador))
558     {
559       SDN.ActivarSelectorDestinos();
560     }
561   }
562   else
563   {
564     // Dibujo del icono ABRIR MAPA.
565
566     if (GUI.Button(Rect(anchuraPantalla - tamanoBotonAbrirMapa - margenBotonAbrirMapa,
          margenBotonAbrirMapa, tamanoBotonAbrirMapa, tamanoBotonAbrirMapa), "",
          estiloAbrirMapa))
567     {
568       SDN.mapaAbiertoCerrado = "Abierto";
569
570       camaraMapa.SetActive(true);
```

```
571
572         if (SDN.mapaGrande == true)
573         {
574           GetComponent.<InterfazMapaControl>().enabled = false;
575           GetComponent.<InterfazMapaGrandeControl>().enabled = true;
576         }
577
578       }
579     }
580
581
582
583
584     #if UNITY_STANDALONE
585
586       // Dibujo del icono de apagado (sólo en plataformas "standalone": Windows, MacOS y
              Linux).
587
588       if (GUI.Button(Rect(anchuraPantalla - TG - margenBotonesGenerales, alturaPantalla -
              TG - margenBotonesGenerales, TG, TG), "", estiloApagado))
589       {
590         SDN.ActivarSelectorApagado();
591       }
592
593       // Dibujo del icono conmutador entre VISIÓN ORBITAL y de teclado.
594
595       if (camaraMapa.GetComponent.<CamaraMapaControl>().permitirVistaTeclado)
596       {
597         if (SDN.vistaOrbital == true)
598         {
599           if (GUI.Button(Rect(anchuraPantalla - TG - margenBotonesGenerales - TG * 1.5,
                  alturaPantalla - TG - margenBotonesGenerales, TG, TG), "", estiloVistaTeclado
                  ))
600           {
601             SDN.DesactivarVistaOrbital();
602           }
603         }
604         else
605         {
606           if (GUI.Button(Rect(anchuraPantalla - TG - margenBotonesGenerales - TG * 1.5,
                  alturaPantalla - TG - margenBotonesGenerales, TG, TG), "", estiloVistaOrbital
                  ))
607           {
608             SDN.ActivarVistaOrbital();
609           }
610
611         }
612       }
613
614
615       // Dibujo del icono para cambiar de idioma.
616
617       if (camaraMapa.GetComponent.<CamaraMapaControl>().permitirCambiarIdioma)
618       {
619         if (camaraMapa.GetComponent.<CamaraMapaControl>().permitirVistaTeclado)
620         {
621           if (GUI.Button(Rect(anchuraPantalla - TG - margenBotonesGenerales - TG * 3.044,
                  alturaPantalla - TG - margenBotonesGenerales, TG, TG), "", estiloCambioIdioma
                  ))
```

```
622         {
623           SDN.ActivarSelectorIdiomas();
624         }
625       }
626       // Si el icono de cambio de vista no está activo dibujamos este a la izquierda del
              todo.
627       else
628       {
629         if (GUI.Button(Rect(anchuraPantalla - TG - margenBotonesGenerales - TG * 1.544,
              alturaPantalla - TG - margenBotonesGenerales, TG, TG), "", estiloCambioIdioma
              ))
630         {
631           SDN.ActivarSelectorIdiomas();
632         }
633       }
634     }
635
636
637     // Dibujo del icono para mostrar la información de ubicación actual.
638
639     // Si es el único icono activo (se ha desactivado el de cambio de vista y el de
          selección de idiomas) lo dibujamos a la izquierda
640     // del todo.
641     if (!camaraMapa.GetComponent.<CamaraMapaControl>().permitirVistaTeclado && !
          camaraMapa.GetComponent.<CamaraMapaControl>().permitirCambiarIdioma)
642     {
643       if (GUI.Button(Rect(anchuraPantalla - TG - margenBotonesGenerales - TG * 1.5,
            alturaPantalla - TG - margenBotonesGenerales, TG, TG), "", estiloDondeEstoy))
644       {
645         SDN.persistente.GetComponent.<PersistenteControl>().mostrarDondeEstoy = true;
646       }
647     }
648     else
649     {
650       // Si sólo uno de los iconos está desactivado dibujamos este al lado del que está
            activo.
651       if (!camaraMapa.GetComponent.<CamaraMapaControl>().permitirVistaTeclado || !
            camaraMapa.GetComponent.<CamaraMapaControl>().permitirCambiarIdioma)
652       {
653         if (GUI.Button(Rect(anchuraPantalla - TG - margenBotonesGenerales - TG * 3,
              alturaPantalla - TG - margenBotonesGenerales, TG, TG), "", estiloDondeEstoy))
654         {
655           SDN.persistente.GetComponent.<PersistenteControl>().mostrarDondeEstoy = true;
656         }
657       }
658       else
659       {
660         // Si los tres iconos están activados dibujamos este junto a los otros dos.
661         if (GUI.Button(Rect(anchuraPantalla - TG - margenBotonesGenerales - TG * 4.5,
              alturaPantalla - TG - margenBotonesGenerales, TG, TG), "", estiloDondeEstoy))
662         {
663           SDN.persistente.GetComponent.<PersistenteControl>().mostrarDondeEstoy = true;
664         }
665       }
666     }
667
668   #else
669
670     // Dibujo del icono conmutador entre VISIÓN ORBITAL y de teclado.
```

```
671
672     if (camaraMapa.GetComponent.<CamaraMapaControl>().permitirVistaTeclado)
673     {
674       if (SDN.vistaOrbital == true)
675       {
676         if (GUI.Button(Rect(anchuraPantalla - TG - margenBotonesGenerales, alturaPantalla
                - TG - margenBotonesGenerales, TG, TG), "", estiloVistaTeclado))
677         {
678           SDN.DesactivarVistaOrbital();
679           //SDN.MostrarTexto("Control orbital desactivado");
680         }
681       }
682       else
683       {
684         if (GUI.Button(Rect(anchuraPantalla - TG - margenBotonesGenerales, alturaPantalla
                - TG - margenBotonesGenerales, TG, TG), "", estiloVistaOrbital))
685         {
686           SDN.ActivarVistaOrbital();
687           //SDN.MostrarTexto("Control orbital activado");
688         }
689
690       }
691     }
692
693
694     // Dibujo del icono para cambiar de idioma.
695
696     if (camaraMapa.GetComponent.<CamaraMapaControl>().permitirCambiarIdioma)
697     {
698       if (camaraMapa.GetComponent.<CamaraMapaControl>().permitirVistaTeclado)
699       {
700         if (GUI.Button(Rect(anchuraPantalla - TG - margenBotonesGenerales - TG * 1.544,
               alturaPantalla - TG - margenBotonesGenerales, TG, TG), "", estiloCambioIdioma
               ))
701         {
702           SDN.ActivarSelectorIdiomas();
703         }
704       }
705       // Si el icono de cambio de vista no está activo dibujamos este a la izquierda del
             todo.
706       else
707       {
708         if (GUI.Button(Rect(anchuraPantalla - TG - margenBotonesGenerales, alturaPantalla
                - TG - margenBotonesGenerales, TG, TG), "", estiloCambioIdioma))
709         {
710           SDN.ActivarSelectorIdiomas();
711         }
712       }
713     }
714
715
716     // Dibujo del icono para mostrar la información de ubicación actual.
717
718     // Si es el único icono activo (se ha desactivado el de cambio de vista y el de
          selección de idiomas) lo dibujamos a la izquierda
719     // del todo.
720     if (!camaraMapa.GetComponent.<CamaraMapaControl>().permitirVistaTeclado && !
          camaraMapa.GetComponent.<CamaraMapaControl>().permitirCambiarIdioma)
721     {
```

```
722         if (GUI.Button(Rect(anchuraPantalla - TG - margenBotonesGenerales , alturaPantalla
                - TG - margenBotonesGenerales, TG, TG), "", estiloDondeEstoy))
723         {
724           SDN.persistente.GetComponent.<PersistenteControl>().mostrarDondeEstoy = true;
725         }
726       }
727       else
728       {
729         // Si sólo uno de los iconos está desactivado dibujamos este al lado del que está
                activo.
730         if (!camaraMapa.GetComponent.<CamaraMapaControl>().permitirVistaTeclado || !
                camaraMapa.GetComponent.<CamaraMapaControl>().permitirCambiarIdioma)
731         {
732           if (GUI.Button(Rect(anchuraPantalla - TG - margenBotonesGenerales - TG * 1.5 ,
                    alturaPantalla - TG - margenBotonesGenerales, TG, TG), "", estiloDondeEstoy))
733           {
734             SDN.persistente.GetComponent.<PersistenteControl>().mostrarDondeEstoy = true;
735           }
736         }
737         else
738         {
739           // Si los tres iconos están activados dibujamos este junto a los otros dos.
740           if (GUI.Button(Rect(anchuraPantalla - TG - margenBotonesGenerales - TG * 3 ,
                    alturaPantalla - TG - margenBotonesGenerales, TG, TG), "", estiloDondeEstoy))
741           {
742             SDN.persistente.GetComponent.<PersistenteControl>().mostrarDondeEstoy = true;
743           }
744         }
745       }
746
747     #endif
748
749
750     // Dibujo del mensaje de navegacion guiada activada y cancelacion de navegacion guiada.
751
752     if (SDN.navegacionGuiada == true)
753     {
754       if (GUI.Button(Rect(margenBotonesGenerales, alturaPantalla * 0.95 -
              margenBotonesGenerales, anchuraPantalla * 0.22, alturaPantalla * 0.05), Traductor
              .Traducir("DesactivarNavegacionGuiada"), estiloCancelarNavegacionGuiada))
755       {
756         SDN.DesactivarNavegacionGuiada();
757       }
758     }
759 }
```

F.20. InterfazMapaGrandeControl.js

```
1 #pragma strict
2
3 /*#################################################################
4
5 InterfazMapaGrandeControl.js
6 ----------------------------
7
8 Guión encargado de la gestión de los elementos de la interfaz del mapa cuando
9 este está en modo maximizado.
```

```
10
11  Controla el aspecto, posición y funcionalidad de los botones de zoom, bloqueo
12  de orientación, botón de regreso a modo normal, botón de desactivación de la
13  navegación guiada cuando esta esté activada.
14
15  ##################################################################*/
16
17
18  var porcentajeBotonRegresar : float;
19  var porcentajeMargenBotonRegresar : float;
20  private var tamanoBotonRegresar : float;
21  private var margenBotonRegresar : float;
22
23  var porcentajeBotonGeneral : float;
24  var porcentajeMargenBotonGeneral : float;
25  private var tamanoBotonGeneral : float;
26  private var margenBotonGeneral : float;
27
28  var estiloBotonRegresar : GUIStyle;
29  var estiloCandadoCerrado : GUIStyle;
30  var estiloCandadoAbierto : GUIStyle;
31  var estiloBotonMenos : GUIStyle;
32  var estiloBotonMas : GUIStyle;
33  var estiloDondeEstoy : GUIStyle;
34
35  var estiloCancelarNavegacionGuiada : GUIStyle;
36
37  private var camaraMapa : GameObject;
38  private var bloqueoMapa : boolean;
39  private var alturaPantalla : float;
40  private var anchuraPantalla : float;
41
42
43  function Start()
44  {
45    camaraMapa = SDN.camaraMapa;
46
47    if (porcentajeBotonRegresar <= 0 || porcentajeBotonRegresar > 100)
48    {
49      Debug.Log("El tamaño del botón <i>regresar</i> indicado está fuera de rango, se
          utilizará el valor predeterminado \"8\".");
50      porcentajeBotonRegresar = 8;
51    }
52
53    if (porcentajeMargenBotonRegresar <= 0 || porcentajeMargenBotonRegresar > 100)
54    {
55      Debug.Log("El porcentaje de margen indicado entre la pantalla y el botón <i>regresar
          </i> está fuera de rango, se utilizará el valor predeterminado \"2\".");
56      porcentajeMargenBotonRegresar = 2;
57    }
58
59    if (porcentajeBotonGeneral <= 0 || porcentajeBotonGeneral > 100)
60    {
61      Debug.Log("El tamaño indicado para los <i>botones generales</i> está fuera de rango,
          se utilizará el valor predeterminado \"4\".");
62      porcentajeBotonGeneral = 4;
63    }
64
65    if (porcentajeMargenBotonGeneral <= 0 || porcentajeMargenBotonGeneral > 100)
```

```
66    {
67      Debug.Log("El porcentaje de margen indicado para los <i>botones generales</i> está
            fuera de rango, se utilizará el valor predeterminado \"2\".");
68      porcentajeMargenBotonGeneral = 2;
69    }
70
71    anchuraPantalla = Screen.width;
72    alturaPantalla = Screen.height;
73
74    tamanoBotonRegresar = porcentajeBotonRegresar / 100 * alturaPantalla;
75    margenBotonRegresar = porcentajeMargenBotonRegresar / 100 * alturaPantalla;
76
77    tamanoBotonGeneral = porcentajeBotonGeneral / 100 * alturaPantalla;
78    margenBotonGeneral = porcentajeMargenBotonGeneral / 100 * alturaPantalla;
79  }
80
81
82  function OnEnable ()
83  {
84    // Se consulta el estado de la orientación del mapa para dibujar el icono
          representativo de
85    // ésta apropiado (candado abierto o candado cerrado).
86
87    bloqueoMapa = SDN.bloqueoMapa;
88  }
89
90
91  function Update ()
92  {
93    if (Screen.width != anchuraPantalla || Screen.height != alturaPantalla)
94    {
95      alturaPantalla = Screen.height;
96      anchuraPantalla = Screen.width;
97
98      tamanoBotonRegresar = porcentajeBotonRegresar / 100 * alturaPantalla;
99      margenBotonRegresar = porcentajeMargenBotonRegresar / 100 * alturaPantalla;
100
101     tamanoBotonGeneral = porcentajeBotonGeneral / 100 * alturaPantalla;
102     margenBotonGeneral = porcentajeMargenBotonGeneral / 100 * alturaPantalla;
103   }
104 }
105
106
107 function OnGUI ()
108 {
109   if (GUI.Button(Rect(Screen.width - tamanoBotonRegresar - margenBotonRegresar,
          margenBotonRegresar, tamanoBotonRegresar, tamanoBotonRegresar), "",
          estiloBotonRegresar))
110   {
111     camaraMapa.GetComponent.<CamaraMapaControl>().pulsadoConmutadorMaximizar = true;
112   }
113
114   // Dibujo del CANDADO DE BLOQUEO.
115
116   if (bloqueoMapa == true)
117   {
118     if (GUI.Button(Rect(Screen.width - tamanoBotonGeneral - margenBotonGeneral, Screen.
          height - tamanoBotonGeneral - margenBotonGeneral, tamanoBotonGeneral,
          tamanoBotonGeneral), "", estiloCandadoCerrado))
```

```
119     {
120       bloqueoMapa = false;
121 
122       // Asignamos también el valor de la variable global para que el guión "
              CamaraMapaControl.js" pueda gestionar la
123       // orientación del mapa entre escenas.
124 
125       SDN.bloqueoMapa = false;
126     }
127   }
128   else
129   {
130     if (GUI.Button(Rect(Screen.width - tamanoBotonGeneral - margenBotonGeneral, Screen.
          height - tamanoBotonGeneral - margenBotonGeneral, tamanoBotonGeneral,
          tamanoBotonGeneral), "", estiloCandadoAbierto))
131     {
132       bloqueoMapa = true;
133       SDN.bloqueoMapa = true;
134     }
135 
136   }
137 
138 
139   if (GUI.Button(Rect(Screen.width - tamanoBotonGeneral - margenBotonGeneral -
        tamanoBotonGeneral * 5.5, Screen.height - tamanoBotonGeneral - margenBotonGeneral,
        tamanoBotonGeneral, tamanoBotonGeneral), "", estiloDondeEstoy))
140   {
141     SDN.persistente.GetComponent.<PersistenteControl>().mostrarDondeEstoy = true;
142   }
143 
144 
145   // Dibujo de los botones MÁS Y MENOS (acercar y alejar)
146 
147   // La cantidad de terreno visto en el mapa a pantalla completa es independiente de la
        mostrada en el minimapa.
148 
149   // Se parte de un valor "orthographicSize" de 11 (definido en el guión "
        CamaraMapaControl.js" y con estos
150   // controles lo modificamos.
151 
152   if (GUI.Button(Rect(Screen.width - tamanoBotonGeneral - margenBotonGeneral -
        tamanoBotonGeneral * 3.5, Screen.height - tamanoBotonGeneral - margenBotonGeneral,
        tamanoBotonGeneral, tamanoBotonGeneral), "", estiloBotonMenos))
153   {
154     SDN.terrenoVistoMaximizado *= 1.5;
155   }
156   if (GUI.Button(Rect(Screen.width - tamanoBotonGeneral - margenBotonGeneral -
        tamanoBotonGeneral * 2, Screen.height - tamanoBotonGeneral - margenBotonGeneral,
        tamanoBotonGeneral, tamanoBotonGeneral), "", estiloBotonMas))
157   {
158     SDN.terrenoVistoMaximizado /= 1.5;
159   }
160 
161 
162   // Dibujo del mensaje de navegacion guiada activada y cancelacion de navegacion guiada.
163 
164   if (SDN.navegacionGuiada == true)
165   {
166 
```

```
167     if (GUI.Button(Rect(margenBotonGeneral, Screen.height * 0.95 - margenBotonGeneral,
            Screen.width * 0.22, Screen.height * 0.05), Traductor.Traducir("
            DesactivarNavegacionGuiada"), estiloCancelarNavegacionGuiada))
168     {
169       SDN.DesactivarNavegacionGuiada();
170     }
171
172   }
173 }
```

F.21. IntersectadorCamaraMapaPerimetro.js

```
1  #pragma strict
2
3  /*#####################################################################
4
5  IntersectadorCamaraMapaPerimetro.js
6  -----------------------------------
7
8  Guión clave en el sistema de navegación guiada, encargado de generar y gestionar el haz
       que
9  intersectará con el área perimetral de la cámara del mapa entre la baliza activa y el
       objeto
10 seguidor del avatar y portador de este guión ("H_IndicadorAvatar"), de calcular la
       posición
11 de colocación de los localizadores, de activar el localizador apropiado dependiendo de si
        hay
12 que subir, bajar o permanecer en el mismo nivel y de activar la aparición de los mensajes
13 informativos que le indicarán al usuario el estado de la navegación guiada.
14
15 #####################################################################*/
16
17
18 private var balizaActiva : GameObject;
19 private var localizador : GameObject;
20 private var impacto : RaycastHit;
21 private var rayo : Ray;
22 private var direccion : Vector3;
23 private var mascaraCapa : int;
24
25 @HideInInspector var puntoDeControlNavegacionGuiada : boolean = false;
26
27
28 // Al activar el intersectador (este guión) cuando se active la de navegación guiada, o
       al cambiar
29 // de escena durante esta, el intersectador analizará en primer lugar el tipo de
       localizador que
30 // se ha de utilizar.
31
32 function OnEnable ()
33 {
34
35   if (SDN.navegacionGuiada == true)
36   {
37
38     // Buscamos la baliza activa desde la que el intersectador lanzará el rayo.
39
```

```
40      balizaActiva = GameObject.Find(SDN.balizaActiva);
41
42      // Activamos el localizador apropiado según el área en el que nos encontremos.
43
44      // Si habia un localizador activo antes activar el intersectador nos aseguramos de
            desactivarlo.
45      if (localizador != null && localizador.activeSelf == true)
46      {
47        localizador.SetActive(false);
48      }
49
50      // A continuación determinamos el tipo localizador necesario para el tramo actual de
            la navegación
51      // guiada, antes de llegar al primer punto de control.
52      switch (SDN.localizadorTipo)
53      {
54        case "normal":
55          localizador = SDN.persistente.GetComponent.<PersistenteControl>().
                localizadorNormalClon;
56          localizador.SetActive(true);
57
58          break;
59
60        case "subir":
61          localizador = SDN.persistente.GetComponent.<PersistenteControl>().
                localizadorSubirClon;
62          localizador.SetActive(true);
63
64          break;
65
66        case "bajar":
67          localizador = SDN.persistente.GetComponent.<PersistenteControl>().
                localizadorBajarClon;
68          localizador.SetActive(true);
69
70          break;
71
72        default:
73          Debug.Log("El tipo de localizador no está definido, se activa el predeterminado."
                );
74
75          localizador = SDN.persistente.GetComponent.<PersistenteControl>().
                localizadorNormalClon;
76          localizador.SetActive(true);
77      }
78
79      SDN.localizadorEnPosicion = false;
80
81      // Para lograr que el rayo intersecte exclusivamente con el área delimitada por el
82      // cubo perimetral creamos una máscara de capa que excluya todo los elementos salvo
83      // los incluidos en la capa especificada, en este caso el cubo perimetral.
84
85      mascaraCapa = 1 << LayerMask.NameToLayer("CamaraMapaPerimetro");
86
87    }
88
89  }
90
91
```

```
92  function LateUpdate ()
93  {
94
95    // Al atravesar el avatar por un punto de control (al cambiar de escena o entrar en un
          área activadora)
96    // el intersectador revisará el tipo de localizador que se ha de utilizar para el nuevo
          tramo de la
97    // navegación guiada. Llamar un nuevo destino guiado desde el selector de destinos
          también activará
98    // este proceso.
99
100   if (SDN.puntoDeControlNavegacionGuiada == true)
101   {
102
103     balizaActiva = GameObject.Find(SDN.balizaActiva);
104
105     // Si había un localizador activo antes del punto de control, nos aseguramos de
          desactivarlo.
106     if (localizador != null && localizador.activeSelf == true)
107     {
108       localizador.SetActive(false);
109     }
110
111     // A continuación determinamos el tipo localizador necesario más allá del punto de
          control
112     // y activamos el apropiado.
113     switch (SDN.localizadorTipo)
114     {
115       case "normal":
116         localizador = SDN.persistente.GetComponent.<PersistenteControl>().
              localizadorNormalClon;
117         localizador.SetActive(true);
118
119         break;
120
121       case "subir":
122         localizador = SDN.persistente.GetComponent.<PersistenteControl>().
              localizadorSubirClon;
123         localizador.SetActive(true);
124
125         break;
126
127       case "bajar":
128         localizador = SDN.persistente.GetComponent.<PersistenteControl>().
              localizadorBajarClon;
129         localizador.SetActive(true);
130
131         break;
132
133       default:
134         Debug.Log("El tipo de localizador no está definido, se activa el predeterminado."
              );
135
136         localizador = SDN.persistente.GetComponent.<PersistenteControl>().
              localizadorNormalClon;
137         localizador.SetActive(true);
138     }
139
140     SDN.puntoDeControlNavegacionGuiada = false;
```

```
141
142   }
143
144
145   // El objeto "H_IndicadorAvatar" se clona inicialmente con el intersectador desactivado
          tal como
146   // está configurado en su prefab, y sólo se activa hasta que se active el sistema de
          navegación
147   // guiada por petición del usuario; si por algún motivo se modificara la configuración
          del prefab
148   // y el objeto se clonara inicialmente con el intersectador activada el propio guión
          intersectador
149   // (este) se autodesactivaría como medida de control.
150
151   // (Se comprueba también la variable "SDN.dondeEstoy", para evitar devolver mensajes de
          "baliza no
152   // encontrada" mientras los puntos de control actualizan el valor de esta variable en
          aquellas
153   // áreas gestionadas por ellos y no por el nombre de la escena a la que pertenecen
          estas. En estas
154   // áreas el objeto persistente reactivará el intersectador hasta que el punto de
          control haya
155   // asignado la información de ubicación apropiada en "SDN.dondeEstoy").
156
157   if (SDN.navegacionGuiada == true && SDN.dondeEstoy != "")
158   {
159
160     // Si el avatar aún no ha llegado al destino y no se ha activado la llegada forzada
            ...
161
162     // (Nota: la llegada forzada la pueden activar destinos especiales como el ala oeste
            del
163     // edificio tecnológico cuando un área cómun como la planta baja de este edificio
            admite
164     // dos ubicaciones, ala oeste y ala este, y a efectos de la navegación una de ellas
            se
165     // comporta como si fuera la otra).
166     if ( SDN.dondeEstoy != SDN.dondeDeboEstar && SDN.navegacionGuiadaForzarLlegada ==
            false)
167     {
168       if (balizaActiva != null)
169       {
170
171         // Dirección y sentido del rayo. El sentido será desde la baliza hasta el
              localizador
172         // y no al revés, porque un rayo no detecta colisiones con el interior de un
              objeto.
173         // Para solucionar esto lanzaremos el rayo desde el exterior hacia el interior.
174
175         direccion = transform.position - balizaActiva.transform.position;
176
177         // Definición del rayo, partiendo de la baliza con sentido hacia el localizador.
178
179         rayo = Ray(balizaActiva.transform.position, direccion);
180
181         Debug.DrawRay(balizaActiva.transform.position, direccion);
182
183
184         if (Physics.Raycast(rayo, impacto, Mathf.Infinity, mascaraCapa))
```

```
185         {
186           if (impacto.collider.tag == "CamaraMapaPerimetro")
187           {
188             localizador.transform.position = impacto.point;
189             SDN.localizadorEnPosicion = true;
190           }
191         }
192         // Si el rayo no intersecta con el perímetro de la cámara del mapa (por ejemplo,
                cuando
193         // se active la navegación guiada estando el avatar suficientemente cerca de la
                baliza
194         // activa, el haz creado entre la baliza y el indicador del avatar se creará
                dentro del
195         // perímetro y por tanto no existirá intersección) se posicionará directamente el
196         // localizador en la posición de la baliza.
197         else
198         {
199           localizador.transform.position = balizaActiva.transform.position;
200         }
201
202       }
203       else
204       {
205         Debug.LogError("La baliza a activar no existe, se desactiva la navegación guiada.
                ");
206
207         SDN.navegacionGuiada = false;
208       }
209
210
211       localizador.transform.LookAt(gameObject.transform);
212       // Las dos siguientes líneas impiden que el aspecto del localizador se distorsione
              cuando el avatar
213       // está encima, de no tenerlo en cuenta su tamaño aparentaría encoger por estar
              colocados en una cota
214       // ligeramente inferior que el indicador del avatar y apuntar siempre a su posición
              .
215       localizador.transform.rotation.x = 0;
216       localizador.transform.rotation.z = 0;
217
218     }
219     // Si el avatar ha llegado al destino...
220     else
221     {
222
223       // Al llegar al desino le indicamos al objeto persistente ("H_Persistente") que
              debe activar
224       // el presentador del nombre de destino ("H_PresentadorNombreDestino") para
              informar al usuario
225       // de que se ha completado la navegación guiada.
226
227       SDN.persistente.GetComponent.<PersistenteControl>().
              mostrarNombreDestinoNavegacionGuiada = true;
228
229       // Hemos llegado al destino, desactivamos la navegación guiada.
230
231       SDN.navegacionGuiada = false;
232
```

```
233         // Reestablecemos también el sistema a sus valores iniciales: desactivamos las
                balizas, los
234         // localizadores y el propio intersectador (este guión).
235
236         // Nota: aun cuando al activar el intersectador se activan también automáticamente
                los
237         // localizadores, si en el mismo frame detectamos que estamos en el destino, cuando
                 el
238         // "renderizado" tenga lugar los localizadores ya estarán desactivados y no
                llegarán por
239         // tanto a dibujarse (importante cuando se active la navegación guiada estando ya
                en el
240         // destino).
241
242         SDN.balizaActiva = "";
243         // (Nota: en caso de asignar un valor vacío ("") al tipo de localizador como valor
244         // inicial, en lugar de "normal", y cuando la navegación guiada se activara estando
245         // ya en el destino y el localizador de balizas no asignara por tanto el tipo de
246         // localizador, el bucle "switch" selector del tipo de localizador de este guión lo
247         // detectaría nada más activarse el intersectador y le asignaría el valor "normal"
248         // automáticamente. Indicándolo sin embargo aquí manualmente evitamos que se
                muestre
249         // el mensaje informativo de asignación automática).
250         SDN.localizadorTipo = "normal";
251         SDN.navegacionGuiadaForzarLlegada = false;
252
253         localizador.SetActive(false);
254
255         gameObject.GetComponent.<IntersectadorCamaraMapaPerimetro>().enabled = false;
256
257       }
258
259     }
260     else
261     {
262       // Si la navegación guiada está desactivada devolvemos el sistema a sus valores
              iniciales,
263       // desactivamos los localizadores y desactivamos el intersectador (este guión).
264
265       SDN.balizaActiva == "";
266       SDN.localizadorTipo == "normal";
267       SDN.navegacionGuiadaForzarLlegada = false;
268
269       localizador.SetActive(false);
270
271       gameObject.GetComponent.<IntersectadorCamaraMapaPerimetro>().enabled = false;
272     }
273 }
```

F.22. LocalizadorDeBalizas.js

```
1 #pragma strict
2
3 /*#################################################################
4
5 LocalizadorDeBalizas.js
6 ----------------------
```

```
7
8  Guión encargado de determinar la baliza que se ha de activar como parte
9  del sistema de navegación guiada.
10
11 Como elemento clave del sistema de navegación guiada, cumple con el cometido
12 de encontrar la baliza desde la que se creará el haz que, al intersectar con
13 el área perimetral generado por el objeto "H_CamaraMapaPerimetro", permitirá
14 definir la posición de los localizadores indicadores de la dirección que ha
15 de seguir el avatar para alcanzar su destino.
16
17 Incluye la la función "Localizar()", como parte de la clase "LocalizadorDeBalizas",
18 y encargada de localizar la baliza correspondiente y asignarla a la variable
19 globlal "SDN.balizaActiva" en función de la ubicación y del destino del avatar
20 (variable globales "SDN.dondeEstoy" y "SDN.dondeDeboEstar".
21
22 ##################################################################*/
23
24
25 static class LocalizadorDeBalizas extends MonoBehaviour
26 {
27
28   function Localizar ()
29   {
30
31     switch (SDN.dondeEstoy)
32     {
33
34       case "Exterior":
35
36         switch (SDN.dondeDeboEstar)
37         {
38           case "AulaUno":
39           case "Secretaria":
40           case "BibliotecaPlantaBaja":
41           case "SalonDeActos":
42           case "AulaDoscientoscuatro":
43           case "AulaDoscientosdieciseis":
44           case "AulaDos":
45           case "AulaTres":
46           case "AulaCuatro":
47           case "EscuelaPlantaBaja":
48             SDN.balizaActiva = "H_BalizaEscuelaEntrada";
49             SDN.localizadorTipo = "normal";
50             break;
51
52           case "TecnologicoDespachoFernandoJFF":
53           case "AulaHacheTres":
54           case "TecnologicoBanoChicasPlantaBaja":
55           case "TecnologicoPlantaBaja":
56             SDN.balizaActiva = "H_BalizaTecnologicoEntrada";
57             SDN.localizadorTipo = "normal";
58             break;
59
60           // default:
61         }
62         break;
63
64
65       case "ExteriorUnion":
```

```
66
67          switch (SDN.dondeDeboEstar)
68          {
69            case "AulaUno":
70            case "Secretaria":
71            case "BibliotecaPlantaBaja":
72            case "SalonDeActos":
73            case "AulaDoscientoscuatro":
74            case "AulaDoscientosdieciseis":
75            case "AulaDos":
76            case "AulaTres":
77            case "AulaCuatro":
78            case "EscuelaPlantaBaja":
79              SDN.balizaActiva = "H_BalizaEscuelaEntradaUnion";
80              SDN.localizadorTipo = "normal";
81              break;
82
83            case "TecnologicoDespachoFernandoJFF":
84            case "AulaHacheTres":
85            case "TecnologicoBanoChicasPlantaBaja":
86            case "TecnologicoPlantaBaja":
87              SDN.balizaActiva = "H_BalizaTecnologicoEntrada";
88              SDN.localizadorTipo = "normal";
89              break;
90
91            // default:
92          }
93          break;
94
95
96        case "EscuelaPlantaBaja":
97
98          switch (SDN.dondeDeboEstar)
99          {
100           case "AulaUno":
101             SDN.balizaActiva = "H_BalizaAulaUno";
102             SDN.localizadorTipo = "normal";
103             break;
104
105           case "TecnologicoDespachoFernandoJFF":
106           case "AulaHacheTres":
107           case "TecnologicoBanoChicasPlantaBaja":
108           case "TecnologicoPlantaBaja":
109             SDN.balizaActiva = "H_BalizaSalidaTecnologico";
110             SDN.localizadorTipo = "normal";
111             break;
112
113           case "Secretaria":
114             SDN.balizaActiva = "H_BalizaSecretaria";
115             SDN.localizadorTipo = "normal";
116             break;
117
118           case "BibliotecaPlantaBaja":
119           case "AulaDoscientoscuatro":
120           case "AulaDoscientosdieciseis":
121             SDN.balizaActiva = "H_BalizaEscuelaEscaleraSurRellano";
122             SDN.localizadorTipo = "subir";
123             break;
124
```

```
125           case "SalonDeActos":
126             SDN.balizaActiva = "H_BalizaSalonDeActos";
127             SDN.localizadorTipo = "normal";
128             break;
129
130           case "AulaDos":
131             SDN.balizaActiva = "H_BalizaAulaDos";
132             SDN.localizadorTipo = "normal";
133             break;
134
135           case "AulaTres":
136             SDN.balizaActiva = "H_BalizaAulaTres";
137             SDN.localizadorTipo = "normal";
138             break;
139
140           case "AulaCuatro":
141             SDN.balizaActiva = "H_BalizaAulaCuatro";
142             SDN.localizadorTipo = "normal";
143             break;
144
145
146
147           // default:
148         }
149
150         break;
151
152
153       case "EscuelaPlantaUno":
154
155         switch (SDN.dondeDeboEstar)
156         {
157           case "AulaUno":
158           case "AulaDos":
159           case "AulaTres":
160           case "AulaCuatro":
161             SDN.balizaActiva = "H_BalizaEscuelaEscaleraNorteRellano";
162             SDN.localizadorTipo = "bajar";
163             break;
164
165           case "TecnologicoDespachoFernandoJFF":
166           case "Secretaria":
167           case "SalonDeActos":
168           case "AulaHacheTres":
169           case "TecnologicoBanoChicasPlantaBaja":
170           case "TecnologicoPlantaBaja":
171           case "EscuelaPlantaBaja":
172             SDN.balizaActiva = "H_BalizaEscuelaEscaleraSurRellano";
173             SDN.localizadorTipo = "bajar";
174             break;
175
176           case "BibliotecaPlantaBaja":
177             SDN.balizaActiva = "H_BalizaBiblioteca";
178             SDN.localizadorTipo = "normal";
179             break;
180
181           case "AulaDoscientoscuatro":
182             SDN.balizaActiva = "H_BalizaAulaDoscientoscuatro";
183             SDN.localizadorTipo = "normal";
```

```
184             break;
185
186           case "AulaDoscientosdieciseis":
187             SDN.balizaActiva = "H_BalizaAulaAulaDoscientosdieciseis";
188             SDN.localizadorTipo = "normal";
189             break;
190
191
192           // default:
193         }
194
195         break;
196
197
198       case "TecnologicoPlantaBaja":
199
200         switch (SDN.dondeDeboEstar)
201         {
202           case "AulaUno":
203           case "Secretaria":
204           case "BibliotecaPlantaBaja":
205           case "SalonDeActos":
206           case "AulaDoscientoscuatro":
207           case "AulaDoscientosdieciseis":
208           case "AulaDos":
209           case "AulaTres":
210           case "AulaCuatro":
211           case "EscuelaPlantaBaja":
212             SDN.balizaActiva = "H_BalizaTecnologicoSalida";
213             SDN.localizadorTipo = "normal";
214             break;
215
216           case "TecnologicoDespachoFernandoJFF":
217           case "AulaHacheTres":
218             SDN.balizaActiva = "H_BalizaTecnologicoEscaleraEsteRellano";
219             SDN.localizadorTipo = "subir";
220             break;
221
222           case "TecnologicoBanoChicasPlantaBaja":
223             SDN.balizaActiva = "H_BalizaTecnologicoBanoChicasPlantaBaja";
224             SDN.localizadorTipo = "normal";
225             break;
226
227           // default:
228         }
229
230         break;
231
232
233       case "TecnologicoPlantaBajaOeste":
234
235         switch (SDN.dondeDeboEstar)
236         {
237           case "AulaUno":
238           case "Secretaria":
239           case "BibliotecaPlantaBaja":
240           case "SalonDeActos":
241           case "AulaDoscientoscuatro":
242           case "AulaDoscientosdieciseis":
```

```
243           case "AulaDos":
244           case "AulaTres":
245           case "AulaCuatro":
246           case "EscuelaPlantaBaja":
247             SDN.balizaActiva = "H_BalizaTecnologicoSalida";
248             SDN.localizadorTipo = "normal";
249             break;
250
251           case "TecnologicoDespachoFernandoJFF":
252           case "AulaHacheTres":
253             SDN.balizaActiva = "H_BalizaTecnologicoEscaleraOesteRellano";
254             SDN.localizadorTipo = "subir";
255             break;
256
257           case "TecnologicoBanoChicasPlantaBaja":
258             SDN.balizaActiva = "H_BalizaTecnologicoBanoChicasPlantaBaja";
259             SDN.localizadorTipo = "normal";
260             break;
261
262           // default:
263         }
264
265         break;
266
267
268       case "TecnologicoPlantaBajaRellanoEscaleras":
269
270         switch (SDN.dondeDeboEstar)
271         {
272           case "AulaUno":
273           case "Secretaria":
274           case "BibliotecaPlantaBaja":
275           case "SalonDeActos":
276           case "AulaDoscientoscuatro":
277           case "AulaDoscientosdieciseis":
278           case "AulaDos":
279           case "AulaTres":
280           case "AulaCuatro":
281           case "TecnologicoBanoChicasPlantaBaja":
282           case "TecnologicoPlantaBaja":
283           case "EscuelaPlantaBaja":
284             SDN.balizaActiva = "H_BalizaTecnologicoEscaleraEsteAbajo";
285             SDN.localizadorTipo = "bajar";
286             break;
287
288           case "TecnologicoDespachoFernandoJFF":
289           case "AulaHacheTres":
290             SDN.balizaActiva = "H_BalizaTecnologicoEscaleraEsteArriba";
291             SDN.localizadorTipo = "subir";
292             break;
293
294           // default:
295         }
296
297         break;
298
299
300       case "TecnologicoPlantaBajaRellanoEscalerasOeste":
301
```

```
302         switch (SDN.dondeDeboEstar)
303         {
304           case "AulaUno":
305           case "Secretaria":
306           case "BibliotecaPlantaBaja":
307           case "SalonDeActos":
308           case "AulaDoscientoscuatro":
309           case "AulaDoscientosdieciseis":
310           case "AulaDos":
311           case "AulaTres":
312           case "AulaCuatro":
313           case "TecnologicoBanoChicasPlantaBaja":
314           case "TecnologicoPlantaBaja":
315           case "EscuelaPlantaBaja":
316             SDN.balizaActiva = "H_BalizaTecnologicoEscaleraOesteAbajo";
317             SDN.localizadorTipo = "bajar";
318             break;
319
320           case "TecnologicoDespachoFernandoJFF":
321           case "AulaHacheTres":
322             SDN.balizaActiva = "H_BalizaTecnologicoEscaleraOesteArriba";
323             SDN.localizadorTipo = "subir";
324             break;
325
326           // default:
327         }
328
329         break;
330
331
332       case "TecnologicoPlantaUno":
333
334         switch (SDN.dondeDeboEstar)
335         {
336           case "AulaUno":
337           case "Secretaria":
338           case "BibliotecaPlantaBaja":
339           case "SalonDeActos":
340           case "AulaDoscientoscuatro":
341           case "AulaDoscientosdieciseis":
342           case "AulaDos":
343           case "AulaTres":
344           case "AulaCuatro":
345           case "TecnologicoBanoChicasPlantaBaja":
346           case "TecnologicoPlantaBaja":
347           case "EscuelaPlantaBaja":
348             SDN.balizaActiva = "H_BalizaTecnologicoEscaleraEsteRellano";
349             SDN.localizadorTipo = "bajar";
350             break;
351
352           case "TecnologicoDespachoFernandoJFF":
353             SDN.balizaActiva = "H_BalizaTecnologicoEscaleraEsteRellano";
354             SDN.localizadorTipo = "subir";
355             break;
356
357           case "AulaHacheTres":
358             SDN.balizaActiva = "H_BalizaAulaHacheTres";
359             SDN.localizadorTipo = "normal";
360             break;
```

```
361
362
363           // default:
364         }
365
366         break;
367
368
369       case "TecnologicoPlantaUnoOeste":
370
371         switch (SDN.dondeDeboEstar)
372         {
373           case "AulaUno":
374           case "Secretaria":
375           case "BibliotecaPlantaBaja":
376           case "SalonDeActos":
377           case "AulaDoscientoscuatro":
378           case "AulaDoscientosdieciseis":
379           case "AulaHacheTres":
380           case "AulaDos":
381           case "AulaTres":
382           case "AulaCuatro":
383           case "TecnologicoBanoChicasPlantaBaja":
384           case "TecnologicoPlantaBaja":
385           case "EscuelaPlantaBaja":
386             SDN.balizaActiva = "H_BalizaTecnologicoEscaleraOesteRellano";
387             SDN.localizadorTipo = "bajar";
388             break;
389
390           case "TecnologicoDespachoFernandoJFF":
391             SDN.balizaActiva = "H_BalizaTecnologicoEscaleraOesteRellano";
392             SDN.localizadorTipo = "subir";
393             break;
394
395           // default:
396         }
397
398         break;
399
400
401       case "TecnologicoPlantaUnoRellanoEscaleras":
402
403         switch (SDN.dondeDeboEstar)
404         {
405           case "AulaUno":
406           case "Secretaria":
407           case "BibliotecaPlantaBaja":
408           case "SalonDeActos":
409           case "AulaDoscientoscuatro":
410           case "AulaDoscientosdieciseis":
411           case "AulaHacheTres":
412           case "AulaDos":
413           case "AulaTres":
414           case "AulaCuatro":
415           case "TecnologicoBanoChicasPlantaBaja":
416           case "TecnologicoPlantaBaja":
417           case "EscuelaPlantaBaja":
418             SDN.balizaActiva = "H_BalizaTecnologicoEscaleraEsteAbajo";
419             SDN.localizadorTipo = "bajar";
```

```
420             break;
421
422           case "TecnologicoDespachoFernandoJFF":
423             SDN.balizaActiva = "H_BalizaTecnologicoEscaleraEsteArriba";
424             SDN.localizadorTipo = "subir";
425             break;
426
427           // default:
428         }
429
430         break;
431
432
433       case "TecnologicoPlantaUnoRellanoEscalerasOeste":
434
435         switch (SDN.dondeDeboEstar)
436         {
437           case "AulaUno":
438           case "Secretaria":
439           case "BibliotecaPlantaBaja":
440           case "SalonDeActos":
441           case "AulaDoscientoscuatro":
442           case "AulaDoscientosdieciseis":
443           case "AulaHacheTres":
444           case "AulaDos":
445           case "AulaTres":
446           case "AulaCuatro":
447           case "TecnologicoBanoChicasPlantaBaja":
448           case "TecnologicoPlantaBaja":
449           case "EscuelaPlantaBaja":
450             SDN.balizaActiva = "H_BalizaTecnologicoEscaleraOesteAbajo";
451             SDN.localizadorTipo = "bajar";
452             break;
453
454           case "TecnologicoDespachoFernandoJFF":
455             SDN.balizaActiva = "H_BalizaTecnologicoEscaleraOesteArriba";
456             SDN.localizadorTipo = "subir";
457             break;
458
459           // default:
460         }
461
462         break;
463
464
465       case "TecnologicoPlantaDos":
466
467         switch (SDN.dondeDeboEstar)
468         {
469           case "AulaUno":
470           case "Secretaria":
471           case "BibliotecaPlantaBaja":
472           case "SalonDeActos":
473           case "AulaDoscientoscuatro":
474           case "AulaDoscientosdieciseis":
475           case "AulaHacheTres":
476           case "AulaDos":
477           case "AulaTres":
478           case "AulaCuatro":
```

```
479           case "TecnologicoBanoChicasPlantaBaja":
480           case "TecnologicoPlantaBaja":
481           case "EscuelaPlantaBaja":
482             SDN.balizaActiva = "H_BalizaTecnologicoEscaleraEsteRellano";
483             SDN.localizadorTipo = "bajar";
484             break;
485
486           case "TecnologicoDespachoFernandoJFF":
487             SDN.balizaActiva = "H_BalizaTecnologicoDespachoFernandoJFFSalida";
488             SDN.localizadorTipo = "normal";
489             break;
490
491           // default:
492         }
493
494         break;
495
496
497       case "TecnologicoPlantaDosOeste":
498
499         switch (SDN.dondeDeboEstar)
500         {
501           case "AulaUno":
502           case "Secretaria":
503           case "BibliotecaPlantaBaja":
504           case "SalonDeActos":
505           case "AulaDoscientoscuatro":
506           case "AulaDoscientosdieciseis":
507           case "AulaHacheTres":
508           case "AulaDos":
509           case "AulaTres":
510           case "AulaCuatro":
511           case "TecnologicoBanoChicasPlantaBaja":
512           case "TecnologicoPlantaBaja":
513           case "EscuelaPlantaBaja":
514             SDN.balizaActiva = "H_BalizaTecnologicoEscaleraOesteRellano";
515             SDN.localizadorTipo = "bajar";
516             break;
517
518           case "TecnologicoDespachoFernandoJFF":
519             SDN.balizaActiva = "H_BalizaTecnologicoDespachoFernandoJFFSalida";
520             SDN.localizadorTipo = "normal";
521             break;
522
523           // default:
524         }
525
526         break;
527
528
529       case "TecnologicoPlantaDosRellanoEscaleras":
530
531         switch (SDN.dondeDeboEstar)
532         {
533           case "AulaUno":
534           case "TecnologicoDespachoFernandoJFF":
535           case "Secretaria":
536           case "BibliotecaPlantaBaja":
537           case "SalonDeActos":
```

```
538           case "AulaDoscientoscuatro":
539           case "AulaDoscientosdieciseis":
540           case "AulaHacheTres":
541           case "AulaDos":
542           case "AulaTres":
543           case "AulaCuatro":
544           case "TecnologicoBanoChicasPlantaBaja":
545           case "TecnologicoPlantaBaja":
546           case "EscuelaPlantaBaja":
547             SDN.balizaActiva = "H_BalizaTecnologicoEscaleraEsteAbajo";
548             SDN.localizadorTipo = "bajar";
549             break;
550
551           // default:
552         }
553
554         break;
555
556
557       case "TecnologicoPlantaDosRellanoEscalerasOeste":
558
559         switch (SDN.dondeDeboEstar)
560         {
561           case "AulaUno":
562           case "TecnologicoDespachoFernandoJFF":
563           case "Secretaria":
564           case "BibliotecaPlantaBaja":
565           case "SalonDeActos":
566           case "AulaDoscientoscuatro":
567           case "AulaDoscientosdieciseis":
568           case "AulaHacheTres":
569           case "AulaDos":
570           case "AulaTres":
571           case "AulaCuatro":
572           case "TecnologicoBanoChicasPlantaBaja":
573           case "TecnologicoPlantaBaja":
574           case "EscuelaPlantaBaja":
575             SDN.balizaActiva = "H_BalizaTecnologicoEscaleraOesteAbajo";
576             SDN.localizadorTipo = "bajar";
577             break;
578
579           // default:
580         }
581
582         break;
583
584
585       case "TecnologicoPlantaTres":
586
587         switch (SDN.dondeDeboEstar)
588         {
589           case "AulaUno":
590           case "TecnologicoDespachoFernandoJFF":
591           case "Secretaria":
592           case "BibliotecaPlantaBaja":
593           case "SalonDeActos":
594           case "AulaDoscientoscuatro":
595           case "AulaDoscientosdieciseis":
596           case "AulaHacheTres":
```

```
597           case "AulaDos":
598           case "AulaTres":
599           case "AulaCuatro":
600           case "TecnologicoBanoChicasPlantaBaja":
601           case "TecnologicoPlantaBaja":
602           case "EscuelaPlantaBaja":
603             SDN.balizaActiva = "H_BalizaTecnologicoEscaleraEsteRellano";
604             SDN.localizadorTipo = "bajar";
605             break;
606
607           // default:
608         }
609
610         break;
611
612
613       case "TecnologicoPlantaTresOeste":
614
615         switch (SDN.dondeDeboEstar)
616         {
617           case "AulaUno":
618           case "TecnologicoDespachoFernandoJFF":
619           case "Secretaria":
620           case "BibliotecaPlantaBaja":
621           case "SalonDeActos":
622           case "AulaDoscientoscuatro":
623           case "AulaDoscientosdieciseis":
624           case "AulaHacheTres":
625           case "AulaDos":
626           case "AulaTres":
627           case "AulaCuatro":
628           case "TecnologicoBanoChicasPlantaBaja":
629           case "TecnologicoPlantaBaja":
630           case "EscuelaPlantaBaja":
631             SDN.balizaActiva = "H_BalizaTecnologicoEscaleraOesteRellano";
632             SDN.localizadorTipo = "bajar";
633             break;
634
635           // default:
636         }
637
638         break;
639
640
641       case "TecnologicoPlantaCuatroEste":
642
643         switch (SDN.dondeDeboEstar)
644         {
645           case "AulaUno":
646           case "TecnologicoDespachoFernandoJFF":
647           case "Secretaria":
648           case "BibliotecaPlantaBaja":
649           case "SalonDeActos":
650           case "AulaDoscientoscuatro":
651           case "AulaDoscientosdieciseis":
652           case "AulaHacheTres":
653           case "AulaDos":
654           case "AulaTres":
655           case "AulaCuatro":
```

```
656           case "TecnologicoBanoChicasPlantaBaja":
657           case "TecnologicoPlantaBaja":
658           case "EscuelaPlantaBaja":
659             SDN.balizaActiva = "H_BalizaTecnologicoEscaleraEsteAbajo";
660             SDN.localizadorTipo = "bajar";
661             break;
662
663           // default:
664         }
665
666         break;
667
668
669       case "TecnologicoPlantaCuatroOeste":
670
671         switch (SDN.dondeDeboEstar)
672         {
673           case "AulaUno":
674           case "TecnologicoDespachoFernandoJFF":
675           case "Secretaria":
676           case "BibliotecaPlantaBaja":
677           case "SalonDeActos":
678           case "AulaDoscientoscuatro":
679           case "AulaDoscientosdieciseis":
680           case "AulaHacheTres":
681           case "AulaDos":
682           case "AulaTres":
683           case "AulaCuatro":
684           case "TecnologicoBanoChicasPlantaBaja":
685           case "TecnologicoPlantaBaja":
686           case "EscuelaPlantaBaja":
687             SDN.balizaActiva = "H_BalizaTecnologicoEscaleraOesteAbajo";
688             SDN.localizadorTipo = "bajar";
689             break;
690
691           // default:
692         }
693
694         break;
695
696
697       // ------------------------------------------------
698       // ------ Ubicaciones con una única salida --------
699       // ------------------------------------------------
700       // No hace falta comprobar el lugar de destino pues
701       // siempre habrá que salir de estas ubicaciones por
702       // el mismo lugar.
703       // ------------------------------------------------
704
705
706       case "BibliotecaPlantaBaja":
707
708         SDN.balizaActiva = "H_BalizaBibliotecaSalida";
709         SDN.localizadorTipo = "normal";
710
711         break;
712
713
714       case "BibliotecaPlantaUno":
```

```
715
716         SDN.balizaActiva = "H_BalizaBibliotecaEscaleras";
717         SDN.localizadorTipo = "bajar";
718
719         break;
720
721
722       case "TecnologicoBanoChicasPlantaBaja":
723
724         SDN.balizaActiva = "H_BalizaTecnologicoBanoChicasPlantaBajaSalida";
725         SDN.localizadorTipo = "normal";
726
727         break;
728
729
730       case "TecnologicoDespachoFernandoJFF":
731         SDN.balizaActiva = "H_BalizaTecnologicoDespachoFernandoJFFSalida";
732         SDN.localizadorTipo = "normal";
733
734         break;
735
736
737       case "SalonDeActos":
738
739         SDN.balizaActiva = "H_BalizaSalonDeActosSalida";
740         SDN.localizadorTipo = "normal";
741
742         break;
743
744
745       case "Secretaria":
746
747         SDN.balizaActiva = "H_BalizaSecretariaSalida";
748         SDN.localizadorTipo = "normal";
749
750         break;
751
752
753       case "AulaUno":
754
755         SDN.balizaActiva = "H_BalizaAulaUnoSalida";
756         SDN.localizadorTipo = "normal";
757
758         break;
759
760
761       case "AulaDos":
762
763         SDN.balizaActiva = "H_BalizaAulaDosSalida";
764         SDN.localizadorTipo = "normal";
765
766         break;
767
768
769       case "AulaTres":
770
771         SDN.balizaActiva = "H_BalizaAulaTresSalida";
772         SDN.localizadorTipo = "normal";
773
```

```
774         break;
775
776
777       case "AulaCuatro":
778
779         SDN.balizaActiva = "H_BalizaAulaCuatroSalida";
780         SDN.localizadorTipo = "normal";
781
782         break;
783
784
785       case "AulaDoscientoscuatro":
786
787         SDN.balizaActiva = "H_BalizaAulaDoscientoscuatroSalida";
788         SDN.localizadorTipo = "normal";
789
790         break;
791
792
793       case "AulaDoscientosdieciseis":
794
795         SDN.balizaActiva = "H_BalizaAulaDoscientosdieciseisSalida";
796         SDN.localizadorTipo = "normal";
797
798         break;
799
800
801       case "AulaHacheTres":
802
803         SDN.balizaActiva = "H_BalizaAulaHacheTresSalida";
804         SDN.localizadorTipo = "normal";
805
806         break;
807
808
809       //default:
810
811       // (Nota: en caso que no se encontrara coincidencia con la información de ubicación
              actual
812       // el intersectador lo gestionaría sin necesidad de predefinir un caso "default").
813
814     }
815
816   }
817
818 }
```

F.23. PersistenteControl.js

```
1 #pragma strict
2
3
4 /*#################################################################
5
6 PersistenteControl.js
7 ----------------------
8
```

```
9  Guión encargado de gestionar los eventos generados por el teclado, de servir
10 como una base de información siempre activa durante la ejecución de una misma
11 escena, de intermediario permanente para aquellos otros guiones que lo necesiten,
12 y de gestionar los objetos presentadores de texto.
13
14 ##################################################################*/
15
16
17 // Control del estado del mapa, minimizado o maximizado
18 // ----------------------------------------------------
19
20 @HideInInspector var teclaCerrarMapa : String;
21 @HideInInspector var teclaMaximizarMapa : String;
22
23 private var camaraMapa : GameObject;
24 private var interfazMapa : GameObject;
25
26 // Vista de teclado.
27 // -----------------
28
29 @HideInInspector var teclaConmutadoraVista : String;
30 @HideInInspector var permitirVistaTeclado : boolean;
31
32 // Informacion "dónde estoy"
33 // -------------------------
34
35 @HideInInspector var teclaDondeEstoy : String;
36 // La siguiente variable la activa el interfaz del mapa, cuando el usuario
37 // pulse sobre el botón de solicitud de información de ubicación.
38 @HideInInspector var mostrarDondeEstoy : boolean;
39
40 // Teclas activadoras de los selectores de destinos e idiomas
41 // ----------------------------------------------------------
42
43 @HideInInspector var teclaSelectorDestinos : String;
44 @HideInInspector var teclaSelectorIdiomas : String;
45
46 // Transporte SDN.
47 // ---------------
48
49 @HideInInspector var presentadorNombreDestino : GameObject;
50 @HideInInspector var presentadorCargandoDestino : GameObject;
51
52 @HideInInspector var presentadorNombreDestinoClon : GameObject;
53 @HideInInspector var presentadorCargandoDestinoClon : GameObject;
54
55 @HideInInspector var mostrarNombreDestino : boolean;
56
57 // Navegación guiada.
58 // ------------------
59
60 // Las referencias a los localizadores las transmite el guión maestro "CamaraMapaControl.
       js"
61 // y las recoge posteriormete el guión intersectador "IntersectadorCamaraMapaPerimetro.js
       ".
62
63 @HideInInspector var localizadorNormalClon : GameObject;
64 @HideInInspector var localizadorSubirClon : GameObject;
65 @HideInInspector var localizadorBajarClon : GameObject;
```

```
66
67  // Cuando el intersectador detecte que se ha llegado al destino indicará que es el
        momento
68  // de informar al usuario.
69
70  @HideInInspector var mostrarNombreDestinoNavegacionGuiada : boolean = false;
71
72  // Control de la vista del mapa (acercar, alejar, restablecer).
73  // ------------------------------------------------------------
74
75  @HideInInspector var permitirControlarMapaConTeclado : boolean;
76
77  @HideInInspector var teclaAcercarMapa : String;
78  @HideInInspector var teclaRestablecerMapa : String;
79  @HideInInspector var teclaAlejarMapa : String;
80
81  // Variable de control para la conmutación entre los estados abierto y cerrado del mapa
82  // mediante el teclado.
83
84  private var estadoMapaControl : boolean;
85
86
87  function Start ()
88  {
89    camaraMapa = SDN.camaraMapa;
90    interfazMapa = SDN.interfazMapa;
91
92
93    // ----------------------------
94    // SISTEMA DE NAVEGACIÓN GUIADA
95    // ----------------------------
96
97    // Nota: los localizadores los clona y desactiva inicialmente el guión maestro ("
          CamaraMapaControl.js")
98    // y los gestiona el intersectador ("IntersectadorCamaraMapaPerimetro.js"), el objeto
          persistente
99    // (este) sólo almacena las referencias de acceso a ellos.
100
101   if (SDN.navegacionGuiada == true)
102   {
103     LocalizadorDeBalizas.Localizar();
104     SDN.puntoDeControlNavegacionGuiada = true;
105   }
106 }
107
108
109 function Update ()
110 {
111   // -------------------------------------
112   // GESTIÓN DE LOS PRESENTADORES DE TEXTO
113   // -------------------------------------
114
115
116   // Presentadores para el transporte SDN.
117   // -------------------------------------
118
119   // Si el desarrollador ha activado la opción de mostrar los nombres de destino en el
          Inspector del
```

```
120   // prefab "H_CamaraMapa" (variable "mostrarNombreDestino") y es el momento de mostrar
          un nombre de
121   //destino (variable "SDN.mostrarNombreDestino"), entonces...
122   if (SDN.mostrarNombreDestino == true && mostrarNombreDestino == true)
123   {
124     // Nos aseguramos de que no hay un presentador de nombres ya activo para evitar
          duplicados.
125     if (presentadorNombreDestinoClon != null)
126     {
127       Destroy(presentadorNombreDestinoClon);
128     }
129
130     // Evitamos mostrar el nombre del destino antes de haber cargado totalmente la escena
          destino
131     // comprobando "Application.isLoadingLevel".
132     if(Application.isLoadingLevel == false)
133     {
134       presentadorNombreDestinoClon = Instantiate(presentadorNombreDestino);
135
136       SDN.mostrarNombreDestino = false;
137     }
138     // En case de que la escena esté cargándose se lo indicamos al usuario.
139     else
140     {
141       // Si hemos elegido un nuevo destino mientras el nombre del anterior aún está en
            pantalla
142       // no queremos que el mensaje de "Cargando destino..." y el nombre del destino
            anterior
143       // se superpongan, para ello, al mostrar el mensajes de que el destino está
            cargando,
144       // destruimos el objeto encargado de mostrar el nombre de destino anterior.
145       if (presentadorNombreDestinoClon != null)
146       {
147         Destroy(presentadorNombreDestinoClon);
148       }
149
150       presentadorCargandoDestinoClon = Instantiate(presentadorCargandoDestino);
151     }
152   }
153
154
155   // Presentadores para la navegación guiada.
156   // ----------------------------------------
157
158   // Cuando el intersectador detecte que se ha alcanzado el destino deseado se informará
          al usuario mediante
159   // un mensaje por pantalla. Si se ha llegado al destino mediante el transporte SDN, el
          mensaje informativo
160   // del sistema de navegación guiada se mostrará después de que el presentador de
          nombres del transporte SDN
161   // haya terminado su labor.
162
163   if (mostrarNombreDestinoNavegacionGuiada == true && presentadorNombreDestinoClon ==
          null)
164   {
165
166     // Evitamos mostrar el nombre del destino antes de haber cargado totalmente la escena
          destino
167     // comprobando "Application.isLoadingLevel".
```

```
168
169     if(Application.isLoadingLevel == false)
170     {
171
172       // El presentador de nombres recoge el valor de la variable "TransporteSDN.
              nombreDestino" para
173       // saber qué mostrar por pantalla. Personalizamos este valor con el siguiente texto
              .
174       TransporteSDN.nombreDestino = Traductor.Traducir("Has llegado al destino");
175
176       presentadorNombreDestinoClon = Instantiate(presentadorNombreDestino);
177
178       mostrarNombreDestinoNavegacionGuiada = false;
179
180     }
181
182   }
183
184
185   // Presentador para informar al usuario de que se está cargando una escena cuando el
          mapa grande está
186   // activado.
187   //
          ----------------------------------------------------------------------------------------------------
188
189   if(Application.isLoadingLevel == true && SDN.mapaGrande == true)
190   {
191     // Nos aseguramos de que no hay un presentador de nombres ya activo para evitar
            duplicados.
192     if (presentadorNombreDestinoClon != null)
193     {
194       Destroy(presentadorNombreDestinoClon);
195     }
196
197     if (presentadorNombreDestinoClon != null)
198     {
199       Destroy(presentadorNombreDestinoClon);
200     }
201
202     presentadorCargandoDestinoClon = Instantiate(presentadorCargandoDestino);
203   }
204
205
206
207   // ------------------------------------------------
208   // GESTIÓN DE LOS EVENTOS GENERADOS CON EL TECLADO
209   // ------------------------------------------------
210
211   // Nota: en el código de todos los eventos se comprueba si "Time.timeScale != 0", de
          esta forma se evita
212   // que el código que llevan asociado se ejecute mientras algún interfaz de selección
          está activado (selector
213   // de destinos o selector de idiomas, por ejemplo); la excepción son los propios
          eventos de apertura y cierre
214   // de estas interfaces.
215
216   // Conmutación entre vista orbital y vista H.
217   // ----------------------------------------
```

```
218
219   if (Input.GetKeyDown(teclaConmutadoraVista) && permitirVistaTeclado == true && Time.
          timeScale != 0)
220   {
221     if (SDN.vistaOrbital == true)
222     {
223       SDN.DesactivarVistaOrbital();
224     }
225     else
226     {
227       SDN.ActivarVistaOrbital();
228     }
229   }
230
231   // Cierre (minimización) y reapertura del mapa
232   // -------------------------------------------
233
234   // Se comprueba que el juego no está pausado para impedir que se pueda activar el mapa
          cuando hay un menú
235   // activo en pantalla.
236
237   if (Input.GetKeyDown(teclaCerrarMapa) && Time.timeScale != 0)
238   {
239
240     // Le indicamos a la interfaz del mapa que la cámara del mapa ha cambiado y ha de
            ajustarse a los cambios.
241     SDN.camaraMapa.GetComponent.<CamaraMapaControl>().posicionCambiada = true;
242
243     if (SDN.mapaAbiertoCerrado == "Abierto")
244     {
245       // La siguiente variable global permite mantener el estado abierto/cerrado del mapa
246       // entre escenas. Su valor queda asignado tanto por la pulsación de la tecla
              conmutadora
247       // del mapa (este bloque de código), como por la pulsación de los botones cerrar y
              abrir
248       // en la interfaz del mapa (guión "InterfazMapaControl.js"); y la usa el guión
249       // "CamaraMapaControl.js" para determinar al cargar cada nueva escena si el mapa ha
              de
250       // mostrarse inicialmente abierto o cerrado.
251
252       SDN.mapaAbiertoCerrado = "Cerrado";
253
254       // Haciéndolo de la siguiente forma da el siguiente error:
255       // objeto.active = false;
256       // BCW0012: WARNING: 'UnityEngine.GameObject.active' is obsolete. GameObject.active
              is obsolete. Use GameObject.SetActive(), GameObject.activeSelf or GameObject.
              activeInHierarchy.
257       // Haciéndolo de la siguiente forma se soluciona:
258
259       camaraMapa.SetActive(false);
260
261       // Al entrar en el modo de mapa a pantalla completa se desactiva la interfaz del
              minimapa, por tanto,
262       // al pulsar la tecla conmutadora en este modo, volvemos a activar la interfaz del
              minimapa para que ésta
263       // gestione el icono de reactivación del mapa (los estados quedarán como si se
              hubiera pulsado la tecla
264       // mientras se mostraba el minimapa en vez del mapa grande).
265
```

```
266         if (SDN.mapaGrande == true)
267         {
268           interfazMapa.GetComponent.<InterfazMapaControl>().enabled = true;
269           interfazMapa.GetComponent.<InterfazMapaGrandeControl>().enabled = false;
270         }
271       }
272       else
273       {
274         SDN.mapaAbiertoCerrado = "Abierto";
275
276         camaraMapa.SetActive(true);
277
278         // Si reactivamos el mapa tras desactivarlo en modo a pantalla completa, al pulsar
              la tecla conmutadora
279         // reactivamos la interfaz del mapa grande y desactivamos la del mapa pequeño, para
               devolver así el
280         // estado en el que se encontraba en el modo de mapa grande.
281
282         if (SDN.mapaGrande == true)
283         {
284           interfazMapa.GetComponent.<InterfazMapaControl>().enabled = false;
285           interfazMapa.GetComponent.<InterfazMapaGrandeControl>().enabled = true;
286         }
287       }
288     }
289
290
291     // Maximización del mapa
292     // ---------------------
293
294     // Alterna entre la vista maximizada y la vista normal del mapa, y lo hace desde
          cualquier estado de éste;
295     // (estando el mapa cerrado también lo maximiza).
296
297     if (Input.GetKeyDown(teclaMaximizarMapa) && Time.timeScale != 0)
298     {
299       if (SDN.mapaAbiertoCerrado == "Abierto")
300       {
301         camaraMapa.GetComponent.<CamaraMapaControl>().pulsadoConmutadorMaximizar = true;
302       }
303       else
304       {
305         SDN.mapaGrande = true;
306         SDN.mapaAbiertoCerrado = "Abierto";
307
308         camaraMapa.SetActive(true);
309       }
310     }
311
312
313     // Apertura y cierre del selector de destinos.
314     // -------------------------------------------
315
316     // Nota: en este y el siguiente bloque no hace falta comprobar que no hay una escena
          cargándose
317     // pues el sistema cargará los selectores únicamente cuando la nueva escena haya
          cargado por
318     // completo.
319
```

```
320   if (Input.GetKeyDown(teclaSelectorDestinos))
321   {
322     if (SDN.interfazDestinos.activeInHierarchy == false)
323     {
324       SDN.DesactivarSelectorIdiomas();
325       SDN.DesactivarSelectorApagado();
326       SDN.ActivarSelectorDestinos();
327     }
328     else
329     {
330       SDN.DesactivarSelectorDestinos();
331     }
332   }
333
334   // Apertura y cierre del selector de idiomas.
335   // ------------------------------------------
336
337   if (Input.GetKeyDown(teclaSelectorIdiomas))
338   {
339     if (SDN.interfazIdiomas.activeInHierarchy == false)
340     {
341       SDN.DesactivarSelectorApagado();
342       SDN.DesactivarSelectorDestinos();
343       SDN.ActivarSelectorIdiomas();
344     }
345     else
346     {
347       SDN.DesactivarSelectorIdiomas();
348     }
349   }
350
351
352   // Acercar, alejar y restablecer la vista del mapa.
353   // ------------------------------------------------
354
355   if (Input.GetKeyDown(teclaAcercarMapa) && Time.timeScale != 0 &&
          permitirControlarMapaConTeclado == true)
356   {
357     if (SDN.mapaAbiertoCerrado == "Abierto")
358     {
359       if (SDN.mapaGrande == true)
360       {
361         SDN.terrenoVistoMaximizado *= 1.5;
362       }
363       else
364       {
365         SDN.terrenoVisto *= 1.5;
366       }
367     }
368   }
369
370   if (Input.GetKeyDown(teclaAlejarMapa) && Time.timeScale != 0 &&
          permitirControlarMapaConTeclado == true)
371   {
372     if (SDN.mapaAbiertoCerrado == "Abierto")
373     {
374       if (SDN.mapaGrande == true)
375       {
376         SDN.terrenoVistoMaximizado /= 1.5;
```

```
377         }
378         else
379         {
380           SDN.terrenoVisto /= 1.5;
381         }
382       }
383     }
384
385     if (Input.GetKeyDown(teclaRestablecerMapa) && Time.timeScale != 0 &&
            permitirControlarMapaConTeclado == true)
386     {
387       if (SDN.mapaAbiertoCerrado == "Abierto")
388       {
389         if (SDN.mapaGrande == true)
390         {
391           SDN.terrenoVistoMaximizado = SDN.camaraMapa.GetComponent.<CamaraMapaControl>().
                terrenoVistoMaximizado;
392         }
393         else
394         {
395           SDN.terrenoVisto = SDN.camaraMapa.GetComponent.<CamaraMapaControl>().terrenoVisto
                ;
396         }
397       }
398     }
399
400
401     // ----------------------------
402     // SISTEMA DE NAVEGACIÓN GUIADA
403     // ----------------------------
404
405     // Si se ha activado la navegación guiada pero el intersectador no está activo, se
            activa.
406
407     // Nota: en aquellas áreas gestionadas por puntos de control el intersectador se
            desactivará automáticamente iniciando
408     // un nuevo ciclo de activación/desactivación mientras que el punto de control no haya
            asignado un valor de ubicación
409     // actual en "SDN.dondeEstoy", garantizando así la fiabilidad del sistema de navegación
            .
410
411     if (SDN.navegacionGuiada == true && SDN.indicadorAvatar.GetComponent.<
            IntersectadorCamaraMapaPerimetro>().enabled == false)
412     {
413       SDN.indicadorAvatar.GetComponent.<IntersectadorCamaraMapaPerimetro>().enabled = true;
414     }
415
416 }
417
418
419 function LateUpdate ()
420 {
421     // Presentador para la función de ubicación (dónde estoy).
422     // -------------------------------------------------------
423
424     // La solicitud de información "dónde estoy" la gestionamos en el ciclo "LateUpdate()"
            para permitr que en
425     // aquellas ubicaciones gestionadas por puntos de control estos registren la variable "
            SDN.dondeEstoy".
```

```
426
427   // La solicitud de información "dónde estoy" estará sólo disponible cuando la variable
          "SDN.dondeEstoy" esté
428   // asignada, de esta forma se resuelve concretamente el escenario en el que el usuario
          pulsara la tecla
429   // activadora mientras se carga un transporte a un área controlada por un punto de
          control, en cuyo caso,
430   // al resolverse el código asociado a la pulsación de la tecla nada más cargar la nueva
           escena la variable
431   // "SDN.dondeEstoy" aún no estaría registrada. También se comprueba que no se esté
          cargando una nueva escena,
432   // para evitar que se superponga la información de ubicación y el mensaje "cargando" en
           el modo de mapa completo.
433   if(SDN.dondeEstoy != "" && Time.timeScale != 0 && Application.isLoadingLevel != true)
434   {
435     if (Input.GetKeyDown(teclaDondeEstoy))
436     {
437       // Nos aseguramos de que no hay un presentador de nombres ya activo para evitar
              duplicados.
438       if (presentadorNombreDestinoClon != null)
439       {
440         Destroy(presentadorNombreDestinoClon);
441       }
442
443       // El presentador de nombres recoge el valor de la variable "TransporteSDN.
              nombreDestino" para
444       // saber qué mostrar por pantalla. Personalizamos este valor con el siguiente texto
              .
445       TransporteSDN.nombreDestino = Traductor.Traducir("Tu ubicación es:") + "\n" +
              Traductor.Traducir(SDN.dondeEstoy);
446
447       presentadorNombreDestinoClon = Instantiate(presentadorNombreDestino);
448     }
449   }
450
451   // Cuando el usuario solicite la información de ubicación pulsando el icono
          correspondiente en la interfaz del mapa
452   // se activa el siguiente bloque de código.
453   if(mostrarDondeEstoy == true)
454   {
455     // Nos aseguramos de que no hay un presentador de nombres ya activo para evitar
            duplicados.
456     if (presentadorNombreDestinoClon != null)
457     {
458       Destroy(presentadorNombreDestinoClon);
459     }
460
461     // El presentador de nombres recoge el valor de la variable "TransporteSDN.
            nombreDestino" para
462     // saber qué mostrar por pantalla. Personalizamos este valor con el siguiente texto.
463     TransporteSDN.nombreDestino = Traductor.Traducir("Tu ubicación es:") + "\n" +
            Traductor.Traducir(SDN.dondeEstoy);
464
465     presentadorNombreDestinoClon = Instantiate(presentadorNombreDestino);
466
467     mostrarDondeEstoy = false;
468   }
469 }
```

F.24. PosicionadorDesactivador.js

```
#pragma strict

/*#####################################################################

PosicionadorDesactivador.js
---------------------------

Guión encargado de asegurar que el objeto "H_Posicionador", de ayuda para el cálculo de las coordenadas
de los destinos del sistema de transporte SDN, esté siempre desactivado al cargar la escena. La función
del objeto "H_Posicionador" es la de ayudar al desarrollador visualmente en la tarea del cálculo de las
coordenadas de destino de un transporte, una vez cumplido su papel deberá permanecer desactivado.

#####################################################################*/


function Awake ()
{
  gameObject.SetActive (false);
}
```

F.25. PresentadorCargandoDestinoControl.js

```
#pragma strict

/*#####################################################################

PresentadorCargandoDestinoControl.js
------------------------------------

Guión encargado de gestionar el comportamiento del objeto presentador de los mensajes
genéricos "Cargando..." y "Cargando destino...", utilizados para informar al usuario de
que se está cargando una escena cuando el mapa se encuentra en modo de pantalla completa
el primero, y el segundo cuando el usuario activa un transporte SDN, para informarle de
que se está cargando el destino hasta que la nueva escena haya cargado por completo.

Este presentador se cargará cuando se esté cargando una nueva escena cumplidas las
circunstancias indicadas y se destruirá automáticamente una vez la nueva escena se
haya cargado por completo.

#####################################################################*/


var porcentajeAncho : float;
var porcentajeAlto : float;
var margenInferior : float;
var texturaFondo : Texture;
var estiloCargandoDestino : GUIStyle;

private var ancho : float;
private var alto : float;
```

```
29 private var margenAncho : float;
30 private var margenAlto : float;
31 private var alturaPantalla : float;
32 private var anchuraPantalla : float;
33
34
35 function Start ()
36 {
37   // Cálculo de las dimensiones para la representación del texto.
38
39   if (porcentajeAncho <= 0 || porcentajeAncho > 100)
40   {
41     Debug.Log("La anchura indicada para el menú selector de destinos está fuera de rango,
          se utilizará el valor predeterminado \"40\".");
42     porcentajeAncho = 65;
43   }
44
45   if (porcentajeAlto <= 0 || porcentajeAlto > 100)
46   {
47     Debug.Log("La altura indicada para el menú selector de destinos está fuera de rango,
         se utilizará el valor predeterminado \"80\".");
48     porcentajeAlto = 20;
49   }
50
51
52   if (margenInferior < 1 || margenInferior > 2)
53   {
54     Debug.Log("El valor indicado para el margen inferior entre el texto y el borde de la
         pantalla está fuera de rango. Se usará el valor predeterminado \"1.1\".");
55     margenInferior = 1.1;
56   }
57
58   alturaPantalla = Screen.height;
59   anchuraPantalla = Screen.width;
60
61   ancho = porcentajeAncho / 100 * anchuraPantalla;
62   alto = porcentajeAlto / 100 * alturaPantalla;
63
64   margenAncho = (anchuraPantalla - ancho) / 2;
65   margenAlto = (alturaPantalla - alto * margenInferior);
66 }
67
68
69 function Update ()
70 {
71   if (Application.isLoadingLevel == false)
72   {
73     Destroy(gameObject);
74   }
75
76
77   if (Screen.width != anchuraPantalla || Screen.height != alturaPantalla)
78   {
79     alturaPantalla = Screen.height;
80     anchuraPantalla = Screen.width;
81
82     ancho = porcentajeAncho / 100 * anchuraPantalla;
83     alto = porcentajeAlto / 100 * alturaPantalla;
84
```

```
85      margenAncho = (anchuraPantalla - ancho) / 2;
86      margenAlto = (alturaPantalla - alto * margenInferior);
87    }
88  }
89
90
91  function OnGUI ()
92  {
93    // Textura de fondo.
94
95    GUI.DrawTexture(Rect(margenAncho, margenAlto, ancho, alto), texturaFondo);
96
97
98    GUILayout.BeginArea(Rect(margenAncho, margenAlto, ancho, alto));
99
100
101     GUILayout.BeginHorizontal();
102
103       // Este presentador puede activarse tanto si el sistema de transporte SDN se activa
              como si
104       // el avatar cambia de escena estando el mapa maximizado. En el primer caso (
              posible sólo cuando
105       // el mapa no está maximizado, mostramos el texto específico "cargando destino", y
              en el segundo
106       // caso el texto genérico "cargando".
107
108       if (SDN.mapaGrande == false)
109       {
110         GUILayout.Label(Traductor.Traducir("Cargando destino..."), estiloCargandoDestino)
                ;
111       }
112       else
113       {
114         GUILayout.Label(Traductor.Traducir("Cargando..."), estiloCargandoDestino);
115       }
116
117     GUILayout.EndHorizontal();
118
119
120   GUILayout.EndArea();
121 }
```

F.26. PresentadorNombreDestinoControl.js

```
1  #pragma strict
2
3  /*#####################################################################
4
5  PresentadorNombreDestinoControl.js
6  ----------------------------------
7
8  Guión encargado de mostrar en pantalla el nombre del destino al que el avatar acaba
9  de transportarse mediante el sistema de navegación. Encargado también del gestionar
10 el tiempo durante el que se muestra el nombre mediante una cuenta atrás.
11
12 El sistema encargado de gestionar el tiempo que el nombre se muestra en pantalla incluye
13 también a las funciones "ActivarSelectorDestinos()", "DesactivarSelectorDestinos()", y
```

```
14 aquellas otras similares, como las del selector de idiomas o de apagado.
15
16 Si durante el tiempo que el nombre del destino está presente en pantalla el usuario
17 activa el selector de destinos (o similar), como éste detiene el tiempo, el nombre del
       destino
18 quedaría superpuesto en pantalla hasta que el selector se desactivara. Para evitarlo
19 este guión detectará cuándo se activa el selector y en ese instante se desactivará a sí
20 mismo, guardando una referencia del tiempo que ha transcurrido hasta que se activó el
21 selector. Si el usuario cancela la selección de destino, la función "
       DesactivarSelectorDestinos()"
22 reactivará este guión y mostrará el nombre del destino sólo durante el tiempo que faltara
23 hasta completar la cuenta atrás.
24
25 ##################################################################*/
26
27
28 var porcentajeAncho : float;
29 var porcentajeAlto : float;
30 var margenInferior : float;
31 var texturaFondo : Texture;
32 var estiloNombreDestino : GUIStyle;
33
34 // Variable que determina el tiempo que el presentador de nombres permanece activo en
       pantalla,
35 // y asignada a través del Inspector del prefab centralizador "H_CamaraMapa".
36 @HideInInspector var tiempoActivo : float;
37
38 private var ancho : float;
39 private var alto : float;
40 private var margenAncho : float;
41 private var margenAlto : float;
42 private var alturaPantalla : float;
43 private var anchuraPantalla : float;
44 private var tiempoInicial : float;
45 private var tiempoTranscurrido : float;
46 private var cuentaAtras : float;
47
48
49 function Start ()
50 {
51   var tiempoInicial = Time.deltaTime;
52
53   var tiempoTranscurrido = 0;
54
55   // Tomamos el valor del tiempo durante el que mostrar el nombre del destino en pantalla
         del guión
56   // "CamaraMapaControl.js" del prefab "CamaraMapa", desde donde podrá cambiarse
       cómodamente a través
57   // del Inspector.
58
59   var tiempoActivo = SDN.camaraMapa.GetComponent.<CamaraMapaControl>().
       tiempoMostrarNombreDestino;
60
61   if (tiempoActivo < 0 || tiempoActivo > 10)
62   {
63     Debug.Log("El tiempo introducido durante el que mostrar el nombre del destino en
         pantalla tras el transporte está fuera de rango. Se usará el valor predeterminado
          \"3\".");
64     tiempoActivo = 3;
```

```
65    }
66
67    cuentaAtras = tiempoActivo - tiempoTranscurrido;
68
69
70    // Cálculo de las dimensiones para la representación del nombre.
71
72    if (porcentajeAncho <= 0 || porcentajeAncho > 100)
73    {
74      Debug.Log("La anchura indicada para el menú selector de destinos está fuera de rango,
          se utilizará el valor predeterminado \"40\".");
75      porcentajeAncho = 65;
76    }
77
78    if (porcentajeAlto <= 0 || porcentajeAlto > 100)
79    {
80      Debug.Log("La altura indicada para el menú selector de destinos está fuera de rango,
          se utilizará el valor predeterminado \"80\".");
81      porcentajeAlto = 20;
82    }
83
84
85    if (margenInferior < 1 || margenInferior > 2)
86    {
87      Debug.Log("El valor indicado para el margen inferior entre el texto y el borde de la
          pantalla está fuera de rango. Se usará el valor predeterminado \"1.1\".");
88      margenInferior = 1.1;
89    }
90
91    alturaPantalla = Screen.height;
92    anchuraPantalla = Screen.width;
93
94    ancho = porcentajeAncho / 100 * anchuraPantalla;
95    alto = porcentajeAlto / 100 * alturaPantalla;
96
97    margenAncho = (anchuraPantalla - ancho) / 2;
98    margenAlto = (alturaPantalla - alto * margenInferior);
99  }
100
101
102 function Update ()
103 {
104   Destroy(gameObject, cuentaAtras);
105
106   if (SDN.interfazDestinos.activeInHierarchy)
107   {
108     tiempoTranscurrido = Time.deltaTime - tiempoInicial;
109     cuentaAtras = tiempoActivo - tiempoTranscurrido;
110
111     gameObject.SetActive(false);
112   }
113
114
115   if (Screen.width != anchuraPantalla || Screen.height != alturaPantalla)
116   {
117     alturaPantalla = Screen.height;
118     anchuraPantalla = Screen.width;
119
120     ancho = porcentajeAncho / 100 * anchuraPantalla;
```

```
121     alto = porcentajeAlto / 100 * alturaPantalla;
122
123     margenAncho = (anchuraPantalla - ancho) / 2;
124     margenAlto = (alturaPantalla - alto * margenInferior);
125   }
126 }
127
128
129 function OnGUI ()
130 {
131   // Textura de fondo.
132
133   GUI.DrawTexture(Rect(margenAncho, margenAlto, ancho, alto), texturaFondo);
134
135
136   GUILayout.BeginArea(Rect(margenAncho, margenAlto, ancho, alto));
137
138
139     GUILayout.BeginHorizontal();
140
141       GUILayout.Label(Traductor.Traducir(TransporteSDN.nombreDestino),
              estiloNombreDestino);
142
143     GUILayout.EndHorizontal();
144
145
146   GUILayout.EndArea();
147 }
```

F.27. PuntoAvatarSeguidor.js

```
1  #pragma strict
2
3  /*#####################################################################
4
5  PuntoAvatarSeguidor.js
6  ----------------------
7
8  Guión que controla la posición del objeto encargado de facilitar el cálculo
9  preciso de las intersecciones del avatar controlador con superficies que tengan
10 el valor "Is Trigger" activado.
11
12 #####################################################################*/
13
14
15 private var objetivo : Transform;
16
17
18 function Start ()
19 {
20   objetivo = SDN.avatar.transform;
21
22   transform.position = objetivo.position;
23 }
24
25
26 function LateUpdate ()
```

```
{
  // Vinculamos la posición a la del avatar controlador.

  transform.position = objetivo.position;

  // El centro de coordenadas del avatar se encuentra en sus pies, por lo que
  // elevamos la posición del concentrador de masa (este) hasta una altura
  // nominal por encima del suelo.

  transform.position.y += 0.5;
}
```

F.28. SDN.js

```
#pragma strict

/*#####################################################################

SDN.js
------

Guión con variables y funciones globales y persistentes útiles para todo el Proyecto H.


#####################################################################*/


static class SDN extends MonoBehaviour
{
  // ======================
  // VARIABLES PERSISTENTES
  // ======================

  // Elementos generales.
  // --------------------

  var camaraPrincipal : GameObject;
  var avatar : GameObject;
  var indicadorAvatar : GameObject;

  var persistente : GameObject;

  var camaraMapa : GameObject;
  var interfazMapa : GameObject;
  var interfazDestinos : GameObject;
  var interfazIdiomas : GameObject;
  var interfazApagado : GameObject;

  // Sistema de transporte.
  // ----------------------

  var presentadorNombreDestino : GameObject;
  var mostrarNombreDestino : boolean = false;

  // Sistema de cambio del tipo de vista.
  // ------------------------------------

```

```
44  var vistaOrbital : boolean = true;
45  var vistaOrbitalNoAlPrincipioEscena : boolean;
46
47  var idioma : String;
48
49  // Sistema de navegación guiada.
50  // -----------------------------
51
52  var navegacionGuiada : boolean = false;
53
54  var balizaActiva : String = "H_BalizaAulaUno";
55
56  var localizadorTipo : String = "normal";
57  var localizadorEnPosicion : boolean;
58
59  var dondeEstoy : String;
60  var dondeDeboEstar : String;
61  var puntoDeControlNavegacionGuiada : boolean;
62  var navegacionGuiadaForzarLlegada : boolean = false;
63
64  // Sistema de posicionamiento (Mapa).
65  // ----------------------------------
66
67  var mapaGrande : boolean = false;
68  var posicionMapa : String = "";
69  // Inicialmente la orientacion del mapa estará fijada al norte.
70  var bloqueoMapa : boolean = true;
71  // El estado del mapa, abierto o cerrado, queda determinado por la siguiente
72  // variable, gestionada entre escenas por el guión "CamaraMapaControl.js"
73  var mapaAbiertoCerrado : String = "";
74  var terrenoVisto : float;
75  var terrenoVistoMaximizado : float;
76
77
78  // =========
79  // FUNCIONES
80  // =========
81
82  // ----------------------------------
83  // ACTIVACIÓN Y DESACTIVACIÓN DEL MAPA
84  // ----------------------------------
85
86  // Funciones usadas por los selectores (de destino, de idioma, de apagado) para cerrar
87  // el mapa mientras se muestre la interfaz correspondiente, y reactivarlo cuando
88  // se cierre.
89
90  function ActivarMapa ()
91  {
92    if (SDN.mapaAbiertoCerrado == "Cerrado")
93    {
94      SDN.mapaAbiertoCerrado = "Abierto";
95
96      interfazMapa.SetActive(true);
97    }
98    else
99    {
100     camaraMapa.SetActive(true);
101
102     interfazMapa.SetActive(true);
```

```
103     }
104   }
105
106
107   function DesactivarMapa ()
108   {
109
110     camaraMapa.SetActive(false);
111
112     interfazMapa.SetActive(false);
113   }
114
115
116   // ----------------------------
117   // PAUSAR Y REANUDAR EL PROGRAMA
118   // ----------------------------
119
120   // Funciones usadas por los selectores (de destino, de idioma, de apagado) para pausar
121   // el programa mientras se muestre la interfaz correspondiente, y reactivarlo cuando
122   // se cierre.
123
124   function PausarJuego ()
125   {
126     SDN.camaraPrincipal.GetComponent.<MouseOrbit>().enabled = false;
127
128     Time.timeScale = 0;
129   }
130
131
132   function ReanudarJuego ()
133   {
134     if (SDN.vistaOrbital == true)
135     {
136       SDN.camaraPrincipal.GetComponent.<MouseOrbit>().enabled = true;
137     }
138
139     Time.timeScale = 1;
140   }
141
142
143   // --------------------
144   // SELECTOR DE DESTINOS
145   // --------------------
146
147   function ActivarSelectorDestinos ()
148   {
149     SDN.PausarJuego();
150     SDN.DesactivarMapa();
151
152     // Al activar el selector de destinos guardamos una referencia del objeto que muestra
          el nombre
153     // de destino cuando el avatar acaba de transportarse si éste existe en la escena. Se
          hace esto
154     // para poder reactivarlo si se cancela el selector a través de la función "SDN.
          DesactivarSelectorDestinos()",
155     // pues este objeto se desactiva automáticamente al detectar el selector. (Más
          información en el guión
156     // "PresentadorNombreDestinoControl.js").
157
```

```
158     SDN.presentadorNombreDestino = GameObject.Find("H_PresentadorNombreDestino(Clone)");
159
160     // Comprobamos que el desarrollador tiene una versión de Unity con licencia Pro, pues es
161     // es necesaria para añadir efectos a las cámaras (en este caso la cámara principal). En caso
162     // de que no disponga de esta licencia se le avisa y so continua sin efectos.
163
164     if ( !Application.HasProLicense() )
165     {
166       Debug.LogWarning("Los efectos de cámara sólo están disponible en una versión de Unity con licencia <b>Pro</b>.");
167     }
168     else
169     {
170       if (SDN.camaraPrincipal.GetComponent.<BlurEffect>() == null)
171       {
172         SDN.camaraPrincipal.AddComponent.<BlurEffect>();
173       }
174       else
175       {
176         camaraPrincipal.GetComponent.<BlurEffect>().enabled = true;
177       }
178     }
179
180     // Si se activa el selector de destinos mientras el nombre del último destino
181     // al que se transportó el avatar aparece en pantalla, se desactiva éste hasta
182     // que el usuario cancele la selección de destinos (o elija un destino, en cuyo
183     // caso se destruirá automáticamente) para evitar la superposición de ambos
184     // elementos en pantalla.
185
186     if (SDN.presentadorNombreDestino != null)
187     {
188       SDN.presentadorNombreDestino.SetActive(false);
189     }
190
191     interfazDestinos.SetActive(true);
192   }
193
194
195   function DesactivarSelectorDestinos ()
196   {
197     SDN.ReanudarJuego();
198     SDN.ActivarMapa();
199
200     // Si el desarrollador no dispone de una versión de Unity con licencia Pro no se le han
201     // añadido efectos a la cámara al activar el selector de Destinos. Si no se hace la siguiente
202     // comprobación Unity devolvería un error al tratar de desactivar un componente inexistente.
203
204     if ( SDN.camaraPrincipal.GetComponent.<BlurEffect>() != null  )
205     {
206       SDN.camaraPrincipal.GetComponent.<BlurEffect>().enabled = false;
207     }
208
209     // Si se cancela el selector de destinos y antes se estaba mostrando en pantalla
210     // el nombre del último destino al que se transportara el avatar, se vuelve a activar
```

```
211     // el nombre del destino. (La variable "SDN.presentadorNombreDestino" la ha definido
212     // previamente la función "ActivarSelectorDestinos()").
213
214     if (SDN.presentadorNombreDestino != null)
215     {
216       SDN.presentadorNombreDestino.SetActive(true);
217     }
218
219     interfazDestinos.SetActive(false);
220   }
221
222
223   // ----------------
224   // CAMBIO DE IDIOMA
225   // ----------------
226
227   function ActivarSelectorIdiomas ()
228   {
229     SDN.PausarJuego();
230     SDN.DesactivarMapa();
231
232     // Al activar el selector de destinos guardamos una referencia del objeto que muestra
          el nombre
233     // de destino cuando el avatar acaba de transportarse si éste existe en la escena. Se
          hace esto
234     // para poder reactivarlo si se cancela el selector a través de la función "SDN.
          DesactivarSelectorDestinos()",
235     // pues este objeto se desactiva automáticamente al detectar el selector. (Más
          información en el guión
236     // "PresentadorNombreDestinoControl.js").
237
238     SDN.presentadorNombreDestino = GameObject.Find("H_PresentadorNombreDestino(Clone)");
239
240     // Comprobamos que el desarrollador tiene una versión de Unity con licencia Pro, pues
          es
241     // es necesaria para añadir efectos a las cámaras (en este caso la cámara principal).
          En caso
242     // de que no disponga de esta licencia se le avisa y so continua sin efectos.
243
244     if ( !Application.HasProLicense() )
245     {
246       Debug.LogWarning("Los efectos de cámara sólo están disponible en una versión de
            Unity con licencia <b>Pro</b>.");
247     }
248     else
249     {
250       // Se comprueba que la cámara no tenga ya el efecto a añadir asociado, para evitar
251       // cualquier posibilidad de duplicados.
252       if (SDN.camaraPrincipal.GetComponent.<BlurEffect>() == null)
253       {
254         SDN.camaraPrincipal.AddComponent.<BlurEffect>();
255       }
256       else
257       {
258         camaraPrincipal.GetComponent.<BlurEffect>().enabled = true;
259       }
260     }
261
262     // Si se activa el selector de destinos mientras el nombre del último destino
```

```
263     // al que se transportó el avatar aparece en pantalla, se desactiva éste hasta
264     // que el usuario cancele la selección de destinos (o elija un destino, en cuyo
265     // caso se destruirá automáticamente) para evitar la superposición de ambos
266     // elementos en pantalla.
267
268     if (SDN.presentadorNombreDestino != null)
269     {
270       SDN.presentadorNombreDestino.SetActive(false);
271     }
272
273     interfazIdiomas.SetActive(true);
274   }
275
276
277   function DesactivarSelectorIdiomas ()
278   {
279     SDN.ReanudarJuego();
280     SDN.ActivarMapa();
281
282     // Si el desarrollador no dispone de una versión de Unity con licencia Pro no se le
          han
283     // añadido efectos a la cámara al activar el selector de Destinos. Si no se hace la
          siguiente
284     // comprobación Unity devolvería un error al tratar de desactivar un componente
          inexistente.
285
286     if ( SDN.camaraPrincipal.GetComponent.<BlurEffect>() != null  )
287     {
288       SDN.camaraPrincipal.GetComponent.<BlurEffect>().enabled = false;
289     }
290
291     // Si se cancela el selector de destinos y antes se estaba mostrando en pantalla
292     // el nombre del último destino al que se transportara el avatar, se vuelve a activar
293     // el nombre del destino. (La variable "SDN.presentadorNombreDestino" la ha definido
294     // previamente la función "ActivarSelectorIdiomas()").
295
296     if (SDN.presentadorNombreDestino != null)
297     {
298       SDN.presentadorNombreDestino.SetActive(true);
299     }
300
301     interfazIdiomas.SetActive(false);
302   }
303
304
305   // ------------------------------------
306   // APAGADO EN PLATAFORMAS "STAND-ALONE"
307   // ------------------------------------
308
309   function ActivarSelectorApagado ()
310   {
311     SDN.PausarJuego();
312     SDN.DesactivarMapa();
313
314     // Al activar el selector de destinos guardamos una referencia del objeto que muestra
          el nombre
315     // de destino cuando el avatar acaba de transportarse si éste existe en la escena. Se
          hace esto
```

```
316     // para poder reactivarlo si se cancela el selector a través de la función "SDN.
            DesactivarSelectorDestinos()",
317     // pues este objeto se desactiva automáticamente al detectar el selector. (Más
            información en el guión
318     // "PresentadorNombreDestinoControl.js").
319
320     SDN.presentadorNombreDestino = GameObject.Find("H_PresentadorNombreDestino(Clone)");
321
322     // Comprobamos que el desarrollador tiene una versión de Unity con licencia Pro, pues
            es
323     // es necesaria para añadir efectos a las cámaras (en este caso la cámara principal).
            En caso
324     // de que no disponga de esta licencia se le avisa y so continua sin efectos.
325
326     if ( !Application.HasProLicense() )
327     {
328       Debug.LogWarning("Los efectos de cámara sólo están disponible en una versión de
              Unity con licencia <b>Pro</b>.");
329     }
330     else
331     {
332       if (SDN.camaraPrincipal.GetComponent.<BlurEffect>() == null)
333       {
334         SDN.camaraPrincipal.AddComponent.<BlurEffect>();
335       }
336       else
337       {
338         camaraPrincipal.GetComponent.<BlurEffect>().enabled = true;
339       }
340     }
341
342     // Si se activa el selector de destinos mientras el nombre del último destino
343     // al que se transportó el avatar aparece en pantalla, se desactiva éste hasta
344     // que el usuario cancele la selección de destinos (o elija un destino, en cuyo
345     // caso se destruirá automáticamente) para evitar la superposición de ambos
346     // elementos en pantalla.
347
348     if (SDN.presentadorNombreDestino != null)
349     {
350       SDN.presentadorNombreDestino.SetActive(false);
351     }
352
353     interfazApagado.SetActive(true);
354   }
355
356
357   function DesactivarSelectorApagado ()
358   {
359     SDN.ReanudarJuego();
360     SDN.ActivarMapa();
361
362     // Si el desarrollador no dispone de una versión de Unity con licencia Pro no se le
            han
363     // añadido efectos a la cámara al activar el selector de Destinos. Si no se hace la
            siguiente
364     // comprobación Unity devolvería un error al tratar de desactivar un componente
            inexistente.
365
366     if ( SDN.camaraPrincipal.GetComponent.<BlurEffect>() != null   )
```

```
367     {
368       SDN.camaraPrincipal.GetComponent.<BlurEffect>().enabled = false;
369     }
370
371     // Si se cancela el selector de destinos y antes se estaba mostrando en pantalla
372     // el nombre del último destino al que se transportara el avatar, se vuelve a activar
373     // el nombre del destino. (La variable "SDN.presentadorNombreDestino" la ha definido
374     // previamente la función "ActivarSelectorApagado()").
375
376     if (SDN.presentadorNombreDestino != null)
377     {
378       SDN.presentadorNombreDestino.SetActive(true);
379     }
380
381     interfazApagado.SetActive(false);
382   }
383
384
385   // ---------------
386   // CAMBIO DE VISTA
387   // ---------------
388
389   function ActivarVistaOrbital ()
390   {
391     SDN.camaraPrincipal.transform.parent = SDN.avatar.transform;
392
393     SDN.camaraPrincipal.GetComponent.<MouseOrbit>().enabled = true;
394
395     SDN.vistaOrbital = true;
396
397     if (SDN.camaraPrincipal.GetComponent.<VistaTeclado>() != null)
398     {
399       SDN.camaraPrincipal.GetComponent.<VistaTeclado>().enabled = false;
400     }
401   }
402
403
404   function DesactivarVistaOrbital ()
405   {
406
407     SDN.camaraPrincipal.GetComponent.<MouseOrbit>().enabled = false;
408
409     SDN.vistaOrbital = false;
410
411     if (SDN.camaraPrincipal.GetComponent.<VistaTeclado>() == null)
412     {
413       SDN.camaraPrincipal.AddComponent.<VistaTeclado>();
414     }
415     else
416     {
417       camaraPrincipal.GetComponent.<VistaTeclado>().enabled = true;
418     }
419
420     // Informamos al guión controlador de la vista por teclado que la cámara no
421     // está en posición (altura y distancia indicadas tras el avatar).
422
423     SDN.camaraPrincipal.GetComponent.<VistaTeclado>().enPosicion = false;
424   }
425
```

```
426
427   // -----------------
428   // NAVEGACIÓN GUIADA
429   // -----------------
430
431   function ActivarNavegacionGuiada (destino : String)
432   {
433     SDN.dondeDeboEstar = destino;
434
435     LocalizadorDeBalizas.Localizar();
436
437     // Activamos la siguiente variable para asegurarnos de que el intersectador reprocesa
438     // toda la información al elegir un nuevo destino guiado aún cuando ya hubiera uno
439     // activado.
440     SDN.puntoDeControlNavegacionGuiada = true;
441
442     SDN.navegacionGuiada = true;
443   }
444
445   function DesactivarNavegacionGuiada ()
446   {
447     SDN.navegacionGuiada = false;
448   }
449
450 }
```

F.29. Traductor.js

```
1  #pragma strict
2
3  /*#####################################################################
4
5  Traductor.js
6  ------------
7
8  Guión encargado de gestionar el sistema traductor del Proyecto H.
9
10 Incluye la función "Traducir()", que admite como argumento la cadena de
11 texto que se desea traducir, la compara con aquellas que tiene recogidas y
12 en caso de encontrar coincidencia, devuelve el texto traducido en función
13 del idioma activo (variable "SDN.idioma".
14
15 Si el texto a traducir introducido no estuviera recogido, la función
16 "Traducir()" devolvería el texto introducido original.
17
18 La plantilla para añadir nuevas traducciones está recogida a continuación, en
19 ella deberán sustituirse los bloques en mayúsculas por los textos adecuados
20 y se copiará y pegará a continuación del punto indicado.
21
22 #####################################################################*/
23
24
25 static class Traductor extends MonoBehaviour
26 {
27
28   function Traducir (texto : String) : String
29   {
```

```
30    var textoTraducido : String;
31    var idioma : String = SDN.idioma;
32
33    switch (texto)
34    {
35
36      /* ----------------------------------------------------------------
37      --------- Plantilla para la introducción de traducciones ---------
38      ----------------------------------------------------------------
39      ------ Copiar, pegar a continuación y sustituir los valores ------
40      ----------------------------------------------------------------
41      ----------------------------------------------------------------
42      ----------- Nota: se puede usar el código \n para añadir ---------
43      ------------------------ una nueva línea -------------------------
44      ----------------------------------------------------------------
45
46      case "TEXTO A TRADUCIR":
47        switch (idioma)
48        {
49          case "ingles":
50            textoTraducido = "TEXTO TRADUCIDO AL INGLÉS";
51            break;
52          case "aleman":
53            textoTraducido = "TEXTO TRADUCIDO AL ALEMÁN";
54            break;
55          default:
56            textoTraducido = "TEXTO TRADUCIDO AL ESPAÑOL";
57            break;
58        }
59
60        break;
61
62      ----------------------------------------------------------------
63      ------------------- Pegar a partir de aquí ---------------------
64      ---------------------------------------------------------------- */
65
66
67
68
69
70      //
            ------------------------------------------------------------------------------------
71      // ------------------------------- Texto informativo
            ---------------------------------
72      //
            ------------------------------------------------------------------------------------
73
74
75      case "Cargando destino...":
76        switch (idioma)
77        {
78          case "ingles":
79            textoTraducido = "Loading destination...";
80            break;
81          case "aleman":
82            textoTraducido = "Die Destination wird geladen...";
83            break;
```

```
84            default:
85              textoTraducido = "Cargando destino...";
86              break;
87          }
88
89          break;
90
91
92        case "Cargando...":
93          switch (idioma)
94          {
95            case "ingles":
96              textoTraducido = "Loading...";
97              break;
98            case "aleman":
99              textoTraducido = "Wird geladen...";
100             break;
101           default:
102             textoTraducido = "Cargando...";
103             break;
104         }
105
106         break;
107
108
109       case "Has llegado al destino":
110         switch (idioma)
111         {
112           case "ingles":
113             textoTraducido = "You have arrived at your destination";
114             break;
115           case "aleman":
116             textoTraducido = "Sie haben Ihr Ziel erreicht";
117             break;
118           default:
119             textoTraducido = "Has llegado al destino";
120             break;
121         }
122
123         break;
124
125
126       case "Tu ubicación es:":
127         switch (idioma)
128         {
129           case "ingles":
130             textoTraducido = "Your location is:";
131             break;
132           case "aleman":
133             textoTraducido = "Ihr Standort ist:";
134             break;
135           default:
136             textoTraducido = "Tu ubicación es:";
137             break;
138         }
139
140         break;
141
142
```

```
143       case "PreguntaApagado":
144         switch (idioma)
145         {
146           case "ingles":
147             textoTraducido = "Click the button to exit the program";
148             break;
149           case "aleman":
150             textoTraducido = "Klicken Sie auf die Schaltfläche, um das Programm zu
                  beenden.";
151             break;
152           default:
153             textoTraducido = "Pulsa el botón para salir del programa";
154             break;
155         }
156
157         break;
158
159
160       //
              ----------------------------------------------------------------------------------

161       //  -------- Nombres de destino para el "presentador de nombres de destino"
              -----------
162       //
              ----------------------------------------------------------------------------------


163
164
165       case "Despacho de\nFernando Jorge Fraile Fernández":
166         switch (idioma)
167         {
168           case "ingles":
169             textoTraducido = "Fernando Jorge Fraile Fernández's\n Office";
170             break;
171           case "aleman":
172             textoTraducido = "Büro von\nFernando Jorge Fraile Fernández";
173             break;
174           default:
175             textoTraducido = "Despacho de\nFernando Jorge Fraile Fernández";
176             break;
177         }
178
179         break;
180
181
182       // Nota para las dos siguientes entradas: en inglés británico la planta baja y la
183       // planta uno son, respectivamente, "ground floor" y "first floor", mientras que en
184       // inglés americano son "first floor" y "second floor". En este caso se ha elegido
185       // la notación americana.
186
187       case "Escuela de ingenierías\nPlanta baja":
188         switch (idioma)
189         {
190           case "ingles":
191             // Notación americana.
192             textoTraducido = "School of Engineering\nFirst Floor";
193             break;
194           case "aleman":
195             textoTraducido = "Ingenieurschule\nErdgeschoss";
```

```
196             break;
197           default:
198             textoTraducido = "Escuela de ingenierías\nPlanta baja";
199             break;
200         }
201
202         break;
203
204
205       case "Escuela de ingenierías\nPlanta uno":
206         switch (idioma)
207         {
208           case "ingles":
209             // Notación americana.
210             textoTraducido = "School of Engineering\nSecond Floor";
211             break;
212           case "aleman":
213             textoTraducido = "Ingenieurschule\nErster Stock";
214             break;
215           default:
216             textoTraducido = "Escuela de ingenierías\nPlanta uno";
217             break;
218         }
219
220         break;
221
222
223       case "Secretaría":
224         switch (idioma)
225         {
226           case "ingles":
227             textoTraducido = "Administrative Office";
228             break;
229           case "aleman":
230             //textoTraducido = "Verwaltungsbüro";
231             textoTraducido = "Sekretariat";
232             break;
233           default:
234             textoTraducido = "Secretaría";
235             break;
236         }
237
238         break;
239
240
241       case "Salón de actos":
242         switch (idioma)
243         {
244           case "ingles":
245             textoTraducido = "Auditorium";
246             break;
247           case "aleman":
248             textoTraducido = "Hörsaal";
249             break;
250           default:
251             textoTraducido = "Salón de actos";
252             break;
253         }
254
```

```
255         break;
256
257
258       case "Aula 1":
259         switch (idioma)
260         {
261           case "ingles":
262             textoTraducido = "Classroom 1";
263             break;
264           case "aleman":
265             textoTraducido = "Klassenzimmer 1";
266             break;
267           default:
268             textoTraducido = "Aula 1";
269             break;
270         }
271
272         break;
273
274
275       case "Aula 2":
276         switch (idioma)
277         {
278           case "ingles":
279             textoTraducido = "Classroom 2";
280             break;
281           case "aleman":
282             textoTraducido = "Klassenzimmer 2";
283             break;
284           default:
285             textoTraducido = "Aula 2";
286             break;
287         }
288
289         break;
290
291
292       case "Aula 3":
293         switch (idioma)
294         {
295           case "ingles":
296             textoTraducido = "Classroom 3";
297             break;
298           case "aleman":
299             textoTraducido = "Klassenzimmer 3";
300             break;
301           default:
302             textoTraducido = "Aula 3";
303             break;
304         }
305
306         break;
307
308
309       case "Aula 4":
310         switch (idioma)
311         {
312           case "ingles":
313             textoTraducido = "Classroom 4";
```

```
314             break;
315           case "aleman":
316             textoTraducido = "Klassenzimmer 4";
317             break;
318           default:
319             textoTraducido = "Aula 4";
320             break;
321         }
322 
323         break;
324 
325 
326       case "Aula 204":
327         switch (idioma)
328         {
329           case "ingles":
330             textoTraducido = "Classroom 204";
331             break;
332           case "aleman":
333             textoTraducido = "Klassenzimmer 204";
334             break;
335           default:
336             textoTraducido = "Aula 204";
337             break;
338         }
339 
340         break;
341 
342 
343       case "Biblioteca\nPlanta baja":
344         switch (idioma)
345         {
346           case "ingles":
347             textoTraducido = "Library\nFirst Floor";
348             break;
349           case "aleman":
350             textoTraducido = "Bibliothek\nErdgeschoss";
351             break;
352           default:
353             textoTraducido = "Biblioteca\nPlanta baja";
354             break;
355         }
356 
357         break;
358 
359 
360       case "Biblioteca\nPlanta uno":
361         switch (idioma)
362         {
363           case "ingles":
364             textoTraducido = "Library\nSecond Floor";
365             break;
366           case "aleman":
367             textoTraducido = "Bibliothek\nErster Stock";
368             break;
369           default:
370             textoTraducido = "Biblioteca\nPlanta uno";
371             break;
372         }
```

```
373
374         break;
375
376
377       case "Escuela de ingenierías\nEntrada":
378         switch (idioma)
379         {
380           case "ingles":
381             textoTraducido = "School of Engineering\nEntrance (Outside)";
382             break;
383           case "aleman":
384             textoTraducido = "Ingenieurschule\nEingang (außerhalb)";
385             break;
386           default:
387             textoTraducido = "Escuela de ingenierías\nEntrada (exterior)";
388             break;
389         }
390
391         break;
392
393
394       case "Aula H3":
395         switch (idioma)
396         {
397           case "ingles":
398             textoTraducido = "Classroom H3";
399             break;
400           case "aleman":
401             textoTraducido = "Klassenzimmer H3";
402             break;
403           default:
404             textoTraducido = "Aula H3";
405             break;
406         }
407
408         break;
409
410
411       case "Aula 216":
412         switch (idioma)
413         {
414           case "ingles":
415             textoTraducido = "Classroom 216";
416             break;
417           case "aleman":
418             textoTraducido = "Klassenzimmer 216";
419             break;
420           default:
421             textoTraducido = "Aula 216";
422             break;
423         }
424
425         break;
426
427
428       case "Edificio tecnológico\nPlanta baja: Aseo (Chicas)":
429         switch (idioma)
430         {
431           case "ingles":
```

```
432             textoTraducido = "Technology Building\nFirst Floor: Restroom (Women)";
433             break;
434           case "aleman":
435             textoTraducido = "Technologiegebäude\nErdgeschoss: Toilette (Frauen)";
436             break;
437           default:
438             textoTraducido = "Edificio tecnológico\nPlanta baja: Aseo (Chicas)";
439             break;
440       }
441
442       break;
443
444
445     case "Edificio tecnológico\nPlanta baja":
446       switch (idioma)
447       {
448         case "ingles":
449           textoTraducido = "Technology Building\nFirst Floor";
450           break;
451         case "aleman":
452           textoTraducido = "Technologiegebäude\nErdgeschoss";
453           break;
454         default:
455           textoTraducido = "Edificio tecnológico\nPlanta baja";
456           break;
457       }
458
459       break;
460
461
462     case "Edificio tecnológico\nPlanta uno":
463       switch (idioma)
464       {
465         case "ingles":
466           textoTraducido = "Technology Building\nSecond Floor";
467           break;
468         case "aleman":
469           textoTraducido = "Technologiegebäude\nErster Stock";
470           break;
471         default:
472           textoTraducido = "Edificio tecnológico\nPlanta uno";
473           break;
474       }
475
476       break;
477
478
479     case "Edificio tecnológico\nPlanta dos":
480       switch (idioma)
481       {
482         case "ingles":
483           textoTraducido = "Technology Building\nThird Floor";
484           break;
485         case "aleman":
486           textoTraducido = "Technologiegebäude\nZweiter Stock";
487           break;
488         default:
489           textoTraducido = "Edificio tecnológico\nPlanta dos";
490           break;
```

```
491         }
492
493         break;
494
495
496       case "Edificio tecnológico\nPlanta tres":
497         switch (idioma)
498         {
499           case "ingles":
500             textoTraducido = "Technology Building\nFourth Floor";
501             break;
502           case "aleman":
503             textoTraducido = "Technologiegebäude\nDritten Stock";
504             break;
505           default:
506             textoTraducido = "Edificio tecnológico\nPlanta tres";
507             break;
508         }
509
510         break;
511
512
513       case "Edificio tecnológico\nPlanta cuatro (ala este)":
514         switch (idioma)
515         {
516           case "ingles":
517             textoTraducido = "Technology Building\nFifth Floor (East Wing)";
518             break;
519           case "aleman":
520             textoTraducido = "Technologiegebäude\nVierten Stock (Ostflügel)";
521             break;
522           default:
523             textoTraducido = "Edificio tecnológico\nPlanta cuatro (ala este)";
524             break;
525         }
526
527         break;
528
529
530       case "Edificio tecnológico\nPlanta cuatro (ala oeste)":
531         switch (idioma)
532         {
533           case "ingles":
534             textoTraducido = "Technology Building\nFifth Floor (West Wing)";
535             break;
536           case "aleman":
537             textoTraducido = "Technologiegebäude\nVierten Stock (Westflügel)";
538             break;
539           default:
540             textoTraducido = "Edificio tecnológico\nPlanta cuatro (ala oeste)";
541             break;
542         }
543
544         break;
545
546
547       //
            ------------------------------------------------------------------------------
```

```
548     //  -------------------------- Nombres de ubicaciones
            --------------------------------
549     //
            ------------------------------------------------------------------------------------------
550
551
552     case "TecnologicoDespachoFernandoJFF":
553       switch (idioma)
554       {
555         case "ingles":
556           textoTraducido = "Fernando Jorge Fraile Fernández's Office";
557           break;
558         case "aleman":
559           textoTraducido = "Büro von Fernando Jorge Fraile Fernández";
560           break;
561         default:
562           textoTraducido = "Despacho de Fernando Jorge Fraile Fernández";
563           break;
564       }
565
566       break;
567
568
569     // Nota para las dos siguientes entradas: en inglés británico la planta baja y la
570     // planta uno son, respectivamente, "ground floor" y "first floor", mientras que en
571     // inglés americano son "first floor" y "second floor". En este caso se ha elegido
572     // la notación americana.
573
574     case "EscuelaPlantaBaja":
575       switch (idioma)
576       {
577         case "ingles":
578           // Notación americana.
579           textoTraducido = "School of Engineering, First Floor";
580           break;
581         case "aleman":
582           textoTraducido = "Ingenieurschule, Erdgeschoss";
583           break;
584         default:
585           textoTraducido = "Escuela de ingenierías, planta baja";
586           break;
587       }
588
589       break;
590
591
592     case "EscuelaPlantaUno":
593       switch (idioma)
594       {
595         case "ingles":
596           // Notación americana.
597           textoTraducido = "School of Engineering, Second Floor";
598           break;
599         case "aleman":
600           textoTraducido = "Ingenieurschule, erster Stock";
601           break;
602         default:
603           textoTraducido = "Escuela de ingenierías, planta uno";
```

```
604             break;
605         }
606 
607         break;
608 
609 
610       case "Secretaria":
611         switch (idioma)
612         {
613           case "ingles":
614             textoTraducido = "Administrative Office";
615             break;
616           case "aleman":
617             //textoTraducido = "Verwaltungsbüro";
618             textoTraducido = "Sekretariat";
619             break;
620           default:
621             textoTraducido = "Secretaría";
622             break;
623         }
624 
625         break;
626 
627 
628       case "SalonDeActos":
629         switch (idioma)
630         {
631           case "ingles":
632             textoTraducido = "Auditorium";
633             break;
634           case "aleman":
635             textoTraducido = "Hörsaal";
636             break;
637           default:
638             textoTraducido = "Salón de actos";
639             break;
640         }
641 
642         break;
643 
644 
645       case "AulaUno":
646         switch (idioma)
647         {
648           case "ingles":
649             textoTraducido = "Classroom 1";
650             break;
651           case "aleman":
652             textoTraducido = "Klassenzimmer 1";
653             break;
654           default:
655             textoTraducido = "Aula 1";
656             break;
657         }
658 
659         break;
660 
661 
662       case "AulaDos":
```

```
663         switch (idioma)
664         {
665           case "ingles":
666             textoTraducido = "Classroom 2";
667             break;
668           case "aleman":
669             textoTraducido = "Klassenzimmer 2";
670             break;
671           default:
672             textoTraducido = "Aula 2";
673             break;
674         }
675
676         break;
677
678
679       case "AulaTres":
680         switch (idioma)
681         {
682           case "ingles":
683             textoTraducido = "Classroom 3";
684             break;
685           case "aleman":
686             textoTraducido = "Klassenzimmer 3";
687             break;
688           default:
689             textoTraducido = "Aula 3";
690             break;
691         }
692
693         break;
694
695
696       case "AulaCuatro":
697         switch (idioma)
698         {
699           case "ingles":
700             textoTraducido = "Classroom 4";
701             break;
702           case "aleman":
703             textoTraducido = "Klassenzimmer 4";
704             break;
705           default:
706             textoTraducido = "Aula 4";
707             break;
708         }
709
710         break;
711
712
713       case "AulaDoscientoscuatro":
714         switch (idioma)
715         {
716           case "ingles":
717             textoTraducido = "Classroom 204";
718             break;
719           case "aleman":
720             textoTraducido = "Klassenzimmer 204";
721             break;
```

```
722           default:
723             textoTraducido = "Aula 204";
724             break;
725         }
726 
727         break;
728 
729 
730       case "BibliotecaPlantaBaja":
731         switch (idioma)
732         {
733           case "ingles":
734             textoTraducido = "Library, First Floor";
735             break;
736           case "aleman":
737             textoTraducido = "Bibliothek, Erdgeschoss";
738             break;
739           default:
740             textoTraducido = "Biblioteca, planta baja";
741             break;
742         }
743 
744         break;
745 
746 
747       case "BibliotecaPlantaUno":
748         switch (idioma)
749         {
750           case "ingles":
751             textoTraducido = "Library, Second Floor";
752             break;
753           case "aleman":
754             textoTraducido = "Bibliothek, erster Stock";
755             break;
756           default:
757             textoTraducido = "Biblioteca, planta uno";
758             break;
759         }
760 
761         break;
762 
763 
764       case "Exterior":
765         switch (idioma)
766         {
767           case "ingles":
768             textoTraducido = "School of Engineering (Outside)";
769             break;
770           case "aleman":
771             textoTraducido = "Ingenieurschule (außerhalb)";
772             break;
773           default:
774             textoTraducido = "Escuela de ingenierías (exterior)";
775             break;
776         }
777 
778         break;
779 
780 
```

```
781       case "ExteriorUnion":
782         switch (idioma)
783         {
784           case "ingles":
785             textoTraducido = "School of Engineering (Passageway)";
786             break;
787           case "aleman":
788             textoTraducido = "Ingenieurschule (Durchgang)";
789             break;
790           default:
791             textoTraducido = "Escuela de ingenierías (pasaje)";
792             break;
793         }
794 
795         break;
796 
797 
798       case "AulaHacheTres":
799         switch (idioma)
800         {
801           case "ingles":
802             textoTraducido = "Classroom H3";
803             break;
804           case "aleman":
805             textoTraducido = "Klassenzimmer H3";
806             break;
807           default:
808             textoTraducido = "Aula H3";
809             break;
810         }
811 
812         break;
813 
814 
815       case "AulaDoscientosdieciseis":
816         switch (idioma)
817         {
818           case "ingles":
819             textoTraducido = "Classroom 216";
820             break;
821           case "aleman":
822             textoTraducido = "Klassenzimmer 216";
823             break;
824           default:
825             textoTraducido = "Aula 216";
826             break;
827         }
828 
829         break;
830 
831 
832       case "TecnologicoBanoChicasPlantaBaja":
833         switch (idioma)
834         {
835           case "ingles":
836             textoTraducido = "Technology Building, First Floor, Restroom (Women)";
837             break;
838           case "aleman":
839             textoTraducido = "Technologiegebäude, Erdgeschoss, Toilette (Frauen)";
```

```
840             break;
841           default:
842             textoTraducido = "Edificio tecnológico, planta baja, aseo (chicas)";
843             break;
844         }
845
846         break;
847
848
849       case "TecnologicoPlantaBaja":
850       case "TecnologicoPlantaBajaOeste":
851       case "TecnologicoPlantaBajaRellanoEscaleras":
852       case "TecnologicoPlantaBajaRellanoEscalerasOeste":
853         switch (idioma)
854         {
855           case "ingles":
856             textoTraducido = "Technology Building, First Floor";
857             break;
858           case "aleman":
859             textoTraducido = "Technologiegebäude, Erdgeschoss";
860             break;
861           default:
862             textoTraducido = "Edificio tecnológico, planta baja";
863             break;
864         }
865
866         break;
867
868
869       case "TecnologicoPlantaUno":
870       case "TecnologicoPlantaUnoOeste":
871       case "TecnologicoPlantaUnoRellanoEscaleras":
872       case "TecnologicoPlantaUnoRellanoEscalerasOeste":
873         switch (idioma)
874         {
875           case "ingles":
876             textoTraducido = "Technology Building, Second Floor";
877             break;
878           case "aleman":
879             textoTraducido = "Technologiegebäude, erster Stock";
880             break;
881           default:
882             textoTraducido = "Edificio tecnológico, planta uno";
883             break;
884         }
885
886         break;
887
888
889       case "TecnologicoPlantaDos":
890       case "TecnologicoPlantaDosOeste":
891       case "TecnologicoPlantaDosRellanoEscaleras":
892       case "TecnologicoPlantaDosRellanoEscalerasOeste":
893         switch (idioma)
894         {
895           case "ingles":
896             textoTraducido = "Technology Building, Third Floor";
897             break;
898           case "aleman":
```

```
899             textoTraducido = "Technologiegebäude, zweiter Stock";
900             break;
901           default:
902             textoTraducido = "Edificio tecnológico, planta dos";
903             break;
904         }
905 
906         break;
907 
908 
909       case "TecnologicoPlantaTres":
910       case "TecnologicoPlantaTresOeste":
911       case "TecnologicoPlantaTresRellanoEscaleras":
912       case "TecnologicoPlantaTresRellanoEscalerasOeste":
913         switch (idioma)
914         {
915           case "ingles":
916             textoTraducido = "Technology Building, Fourth Floor";
917             break;
918           case "aleman":
919             textoTraducido = "Technologiegebäude, dritten Stock";
920             break;
921           default:
922             textoTraducido = "Edificio tecnológico, planta tres";
923             break;
924         }
925 
926         break;
927 
928 
929       case "TecnologicoPlantaCuatroEste":
930         switch (idioma)
931         {
932           case "ingles":
933             textoTraducido = "Technology Building, Fifth Floor (East Wing)";
934             break;
935           case "aleman":
936             textoTraducido = "Technologiegebäude, vierten Stock (Ostflügel)";
937             break;
938           default:
939             textoTraducido = "Edificio tecnológico, planta cuatro (ala este)";
940             break;
941         }
942 
943         break;
944 
945 
946       case "TecnologicoPlantaCuatroOeste":
947         switch (idioma)
948         {
949           case "ingles":
950             textoTraducido = "Technology Building, Fifth Floor (West Wing)";
951             break;
952           case "aleman":
953             textoTraducido = "Technologiegebäude, vierten Stock (Westflügel)";
954             break;
955           default:
956             textoTraducido = "Edificio tecnológico, planta cuatro (ala oeste)";
957             break;
```

```
958         }
959
960         break;
961
962
963
964       //
              ----------------------------------------------------------------------------------------
965       //  ---------------------------------- Botones
              ----------------------------------------
966       //
              ----------------------------------------------------------------------------------------
967
968
969       case "BotonCancelarSeleccionDestino":
970         switch (idioma)
971         {
972           case "ingles":
973             textoTraducido = "Cancel Destination Selection";
974             break;
975           case "aleman":
976             textoTraducido = "Zielauswahl abbrechen";
977             break;
978           default:
979             textoTraducido = "Cancelar la selección de destino";
980             break;
981         }
982
983         break;
984
985       case "BotonDespachoFernandoJFF":
986         switch (idioma)
987         {
988           case "ingles":
989             textoTraducido = "Fernando Jorge Fraile Fernández's Office";
990             break;
991           case "aleman":
992             textoTraducido = "Büro von Fernando Jorge Fraile Fernández";
993             break;
994           default:
995             textoTraducido = "Despacho de Fernando Jorge Fraile Fernández";
996             break;
997         }
998
999         break;
1000
1001      case "BotonEscuelaPlantaBaja":
1002        switch (idioma)
1003        {
1004          case "ingles":
1005            // Notación americana.
1006            textoTraducido = "School of Engineering, First Floor";
1007            break;
1008          case "aleman":
1009            textoTraducido = "Ingenieurschule, Erdgeschoss";
1010            break;
1011          default:
```

```
1012            textoTraducido = "Escuela de ingenierías, planta baja";
1013            break;
1014        }
1015
1016        break;
1017
1018
1019      case "BotonEscuelaPlantaUno":
1020        switch (idioma)
1021        {
1022          case "ingles":
1023            // Notación americana.
1024            textoTraducido = "School of Engineering, Second Floor";
1025            break;
1026          case "aleman":
1027            textoTraducido = "Ingenieurschule, erster Stock";
1028            break;
1029          default:
1030            textoTraducido = "Escuela de ingenierías, planta uno";
1031            break;
1032        }
1033
1034        break;
1035
1036
1037      case "BotonSecretaria":
1038        switch (idioma)
1039        {
1040          case "ingles":
1041            textoTraducido = "Administrative Office";
1042            break;
1043          case "aleman":
1044            //textoTraducido = "Verwaltungsbüro";
1045            textoTraducido = "Sekretariat";
1046            break;
1047          default:
1048            textoTraducido = "Secretaría";
1049            break;
1050        }
1051
1052        break;
1053
1054
1055      case "BotonSalonDeActos":
1056        switch (idioma)
1057        {
1058          case "ingles":
1059            textoTraducido = "Auditorium";
1060            break;
1061          case "aleman":
1062            textoTraducido = "Hörsaal";
1063            break;
1064          default:
1065            textoTraducido = "Salón de actos";
1066            break;
1067        }
1068
1069        break;
1070
```

```
1071
1072        case "BotonAulaUno":
1073          switch (idioma)
1074          {
1075            case "ingles":
1076              textoTraducido = "Classroom 1";
1077              break;
1078            case "aleman":
1079              textoTraducido = "Klassenzimmer 1";
1080              break;
1081            default:
1082              textoTraducido = "Aula 1";
1083              break;
1084          }
1085
1086          break;
1087
1088
1089        case "BotonAulaDos":
1090          switch (idioma)
1091          {
1092            case "ingles":
1093              textoTraducido = "Classroom 2";
1094              break;
1095            case "aleman":
1096              textoTraducido = "Klassenzimmer 2";
1097              break;
1098            default:
1099              textoTraducido = "Aula 2";
1100              break;
1101          }
1102
1103          break;
1104
1105
1106        case "BotonAulaTres":
1107          switch (idioma)
1108          {
1109            case "ingles":
1110              textoTraducido = "Classroom 3";
1111              break;
1112            case "aleman":
1113              textoTraducido = "Klassenzimmer 3";
1114              break;
1115            default:
1116              textoTraducido = "Aula 3";
1117              break;
1118          }
1119
1120          break;
1121
1122
1123        case "BotonAulaCuatro":
1124          switch (idioma)
1125          {
1126            case "ingles":
1127              textoTraducido = "Classroom 4";
1128              break;
1129            case "aleman":
```

```
1130            textoTraducido = "Klassenzimmer 4";
1131            break;
1132          default:
1133            textoTraducido = "Aula 4";
1134            break;
1135        }
1136
1137        break;
1138
1139
1140      case "BotonAulaDoscientoscuatro":
1141        switch (idioma)
1142        {
1143          case "ingles":
1144            textoTraducido = "Classroom 204";
1145            break;
1146          case "aleman":
1147            textoTraducido = "Klassenzimmer 204";
1148            break;
1149          default:
1150            textoTraducido = "Aula 204";
1151            break;
1152        }
1153
1154        break;
1155
1156
1157      case "BotonBibliotecaPlantaBaja":
1158        switch (idioma)
1159        {
1160          case "ingles":
1161            textoTraducido = "Library, First Floor";
1162            break;
1163          case "aleman":
1164            textoTraducido = "Bibliothek, Erdgeschoss";
1165            break;
1166          default:
1167            textoTraducido = "Biblioteca, planta baja";
1168            break;
1169        }
1170
1171        break;
1172
1173
1174      case "BotonBibliotecaPlantaUno":
1175        switch (idioma)
1176        {
1177          case "ingles":
1178            textoTraducido = "Library, Second Floor";
1179            break;
1180          case "aleman":
1181            textoTraducido = "Bibliothek, erster Stock";
1182            break;
1183          default:
1184            textoTraducido = "Biblioteca, planta uno";
1185            break;
1186        }
1187
1188        break;
```

```
1189
1190
1191      case "BotonEscuelaExteriorEntrada":
1192        switch (idioma)
1193        {
1194          case "ingles":
1195            textoTraducido = "School of Engineering, Entrance (Outside)";
1196            break;
1197          case "aleman":
1198            textoTraducido = "Ingenieurschule, Eingang (außerhalb)";
1199            break;
1200          default:
1201            textoTraducido = "Escuela de ingenierías, entrada (exterior)";
1202            break;
1203        }
1204
1205        break;
1206
1207
1208      case "BotonAulaHacheTres":
1209        switch (idioma)
1210        {
1211          case "ingles":
1212            textoTraducido = "Classroom H3";
1213            break;
1214          case "aleman":
1215            textoTraducido = "Klassenzimmer H3";
1216            break;
1217          default:
1218            textoTraducido = "Aula H3";
1219            break;
1220        }
1221
1222        break;
1223
1224
1225      case "BotonAulaDoscientosdieciseis":
1226        switch (idioma)
1227        {
1228          case "ingles":
1229            textoTraducido = "Classroom 216";
1230            break;
1231          case "aleman":
1232            textoTraducido = "Klassenzimmer 216";
1233            break;
1234          default:
1235            textoTraducido = "Aula 216";
1236            break;
1237        }
1238
1239        break;
1240
1241
1242      case "BotonTecnologicoPlantaBajaAseoChicas":
1243        switch (idioma)
1244        {
1245          case "ingles":
1246            textoTraducido = "Technology Building, First Floor, Restroom (Women)";
1247            break;
```

```
1248            case "aleman":
1249              textoTraducido = "Technologiegebäude, Erdgeschoss, Toilette (Frauen)";
1250              break;
1251            default:
1252              textoTraducido = "Edificio tecnológico, planta baja, aseo (chicas)";
1253              break;
1254          }
1255
1256          break;
1257
1258
1259        case "BotonTecnologicoPlantaBaja":
1260          switch (idioma)
1261          {
1262            case "ingles":
1263              textoTraducido = "Technology Building, First Floor";
1264              break;
1265            case "aleman":
1266              textoTraducido = "Technologiegebäude, Erdgeschoss";
1267              break;
1268            default:
1269              textoTraducido = "Edificio tecnológico, planta baja";
1270              break;
1271          }
1272
1273          break;
1274
1275
1276        case "BotonTecnologicoPlantaUno":
1277          switch (idioma)
1278          {
1279            case "ingles":
1280              textoTraducido = "Technology Building, Second Floor";
1281              break;
1282            case "aleman":
1283              textoTraducido = "Technologiegebäude, erster Stock";
1284              break;
1285            default:
1286              textoTraducido = "Edificio tecnológico, planta uno";
1287              break;
1288          }
1289
1290          break;
1291
1292
1293        case "BotonTecnologicoPlantaDos":
1294          switch (idioma)
1295          {
1296            case "ingles":
1297              textoTraducido = "Technology Building, Third Floor";
1298              break;
1299            case "aleman":
1300              textoTraducido = "Technologiegebäude, zweiter Stock";
1301              break;
1302            default:
1303              textoTraducido = "Edificio tecnológico, planta dos";
1304              break;
1305          }
1306
```

```
1307        break;
1308
1309
1310      case "BotonTecnologicoPlantaTres":
1311        switch (idioma)
1312        {
1313          case "ingles":
1314            textoTraducido = "Technology Building, Fourth Floor";
1315            break;
1316          case "aleman":
1317            textoTraducido = "Technologiegebäude, dritten Stock";
1318            break;
1319          default:
1320            textoTraducido = "Edificio tecnológico, planta tres";
1321            break;
1322        }
1323
1324        break;
1325
1326
1327      case "BotonTecnologicoPlantaCuatroEste":
1328        switch (idioma)
1329        {
1330          case "ingles":
1331            textoTraducido = "Technology Building, Fifth Floor (East Wing)";
1332            break;
1333          case "aleman":
1334            textoTraducido = "Technologiegebäude, vierten Stock (Ostflügel)";
1335            break;
1336          default:
1337            textoTraducido = "Edificio tecnológico, planta cuatro (ala este)";
1338            break;
1339        }
1340
1341        break;
1342
1343
1344      case "BotonTecnologicoPlantaCuatroOeste":
1345        switch (idioma)
1346        {
1347          case "ingles":
1348            textoTraducido = "Technology Building, Fifth Floor (West Wing)";
1349            break;
1350          case "aleman":
1351            textoTraducido = "Technologiegebäude, vierten Stock (Westflügel)";
1352            break;
1353          default:
1354            textoTraducido = "Edificio tecnológico, planta cuatro (ala oeste)";
1355            break;
1356        }
1357
1358        break;
1359
1360
1361      //
             ------------------------------------------------------------------------------
1362      // ------------------------------ Selector de destinos
             ------------------------------
```

```
1363        //
                ----------------------------------------------------------------------------
1364
1365
1366        case "ELIGE UN DESTINO":
1367          switch (idioma)
1368          {
1369            case "ingles":
1370              textoTraducido = "CHOOSE YOUR DESTINATION";
1371              break;
1372            case "aleman":
1373              textoTraducido = "WÄHLEN SIE EIN BESTIMMUNGSORT";
1374              break;
1375            default:
1376              textoTraducido = "ELIGE UN DESTINO";
1377              break;
1378          }
1379
1380          break;
1381
1382
1383        case "InstruccionesSelectorDestino":
1384          switch (idioma)
1385          {
1386            case "ingles":
1387              textoTraducido = "The compass next to some destinations enables the <b>guided
                       navigation</b>. When active, the map will show the direction to follow
                      to get to the chosen destination. The direction indicator will change
                      color to indicate whether you have to go down, up or remain on the
                      current floor.";
1388              break;
1389            case "aleman":
1390              textoTraducido = "Der Kompaß neben einigen Bestimmungsörtern befähigt die <b>
                      geführte Navigation</b>. Wenn aktiv, wird die Karte die Richtung zeigen,
                      um zum gewählten Bestimmungsort zu kommen. Die Richtungsanzeige wird
                      seine Farbe ändern, um anzuzeigen, wenn Sie nach unten gehen, nach oben
                      gehen, oder auf dem gegenwärtigen Fußboden bleiben müssen.";
1391              break;
1392            default:
1393              textoTraducido = "La brújula que aparece junto a algunos destinos permite
                      activar la <b>navegación guiada</b>. Al activarla se indicará en el mapa
                      la dirección en la que se encuentra el destino elegido. El color del
                      indicador cambiará para señalar si se ha de bajar, subir o permanecer en
                      el mismo nivel.";
1394              break;
1395          }
1396
1397          break;
1398
1399
1400        case "InstruccionesLocalizadorSubir":
1401          switch (idioma)
1402          {
1403            case "ingles":
1404              textoTraducido = "<color=red>Red</color> if you are to go up.";
1405              break;
1406            case "aleman":
1407              textoTraducido = "<color=red>Rot,</color> wenn Sie hinaufgehen müssen.";
```

```
1408            break;
1409          default:
1410            textoTraducido = "<color=red>Rojo</color> si hay que subir.";
1411            break;
1412        }
1413
1414        break;
1415
1416
1417      case "InstruccionesLocalizadorBajar":
1418        switch (idioma)
1419        {
1420          case "ingles":
1421            textoTraducido = "<color=blue>Blue</color> if you are to go down.";
1422            break;
1423          case "aleman":
1424            textoTraducido = "<color=blue>Blau,</color> wenn Sie hinuntergehen müssen.";
1425            break;
1426          default:
1427            textoTraducido = "<color=blue>Azul</color> si hay que bajar.";
1428            break;
1429        }
1430
1431        break;
1432
1433
1434      case "InstruccionesLocalizadorNormal":
1435        switch (idioma)
1436        {
1437          case "ingles":
1438            textoTraducido = "<color=green>Green</color> if you are to remain on the
                  current floor.";
1439            break;
1440          case "aleman":
1441            textoTraducido = "<color=green>Grün,</color> wenn Sie auf der gegenwärtigen
                  Höhe bleiben müssen.";
1442            break;
1443          default:
1444            textoTraducido = "<color=green>Verde</color> si hay que seguir en el mismo
                  nivel.";
1445            break;
1446        }
1447
1448        break;
1449
1450
1451      case "DesactivarNavegacionGuiada":
1452        switch (idioma)
1453        {
1454          case "ingles":
1455            textoTraducido = "Disable Guided Navigation";
1456            break;
1457          case "aleman":
1458            textoTraducido = "Geführte Navigation ausschalten";
1459            break;
1460          default:
1461            textoTraducido = "Desactivar navegación guiada";
1462            break;
1463        }
```

```
1464
1465          break;
1466
1467
1468        //
              ----------------------------------------------------------------------------------
1469
1470        // Si la cadena de texto que se desea traducir no está aún recogida, se devuelve el
              texto
1471        // introducido tal cual.
1472
1473        default:
1474          textoTraducido = texto;
1475      }
1476
1477      if (textoTraducido != "")
1478      {
1479        return textoTraducido;
1480      }
1481      else
1482      {
1483        return texto;
1484      }
1485    }
1486  }
1487
1488
1489  /*
1490  // Nota: Si se usa el siguiente estilo de código Unity devuelve la siguiente advertencia:
1491  // BCW0015: WARNING: Unreachable code detected.
1492
1493  // Esto ocurre porque las instrucciones "return" ocultan el resto del código al salir
1494  // del bloque "switch" al llegar a ellas.
1495
1496  // Para solucionarlo sustituimos las instrucciones "return" por una asignación de
       variable
1497  // cuyo valor devolveremos con una única instrucción "return" global.
1498
1499  // Estilo de código a evitar:
1500
1501  case "Cargando destino...":
1502    switch (idioma)
1503    {
1504      case "ingles":
1505        return "Loading destination...";
1506        break;
1507      case "aleman":
1508        return "Die Destination wird geladen.";
1509      default:
1510        return "Cargando destino...";
1511        break;
1512    }
1513  */
```

F.30. TransporteSDN.js

```
#pragma strict

/*#####################################################################

TransporteSDN.js
----------------

Guión encargado de gestionar el transporte SDN.

El transporte SDN es un nuevo sistema proporcionado por el Proyecto H capaz de
gestionar cualquier reubicación del avatar, a cualquier punto del proyecto,
entre escenas o dentro de las mismas. Eficaz y eficiente, permite añadir fácilmente
nuevos destinos a los ya recogidos para un trasnporte automatizado mediante la función
de transporte general, así como definir las coordenadas del destino directamente mediante
la función de transporte manual; considera también la escena en la que se encuentra el
avatar para no recargar de nuevo esta si el transporte es local.

Funciones que incluye:

TransporteSDN.Destino("lugar")
TransporteSDN.Destino(X, Y, Z, rotacionY, "nombre escena", "etiqueta a mostrar").

La primera lee el destino de la base de datos de destinos y lee de ahí las coordenadas
de transporte. La segunda permite indicar manualmente las coordenadas X, Y y Z del punto
de destino, la rotación Y del avatar con la que aparecerá tras transportarse, el nombre
de la escena a la que trasportarse y el texto informativo que se mostrará en pantalla al
llegar al destino (gestionado éste por el presentador de nombres de destino
"PresentadorNombreDestinoControl.js". Ambas proporcionan al guión constructor "SustContr.
    js"
los valores de posición y rotación con los que clonar en avatar en la escena mediante la
asignación de las variables "TransporteSDN.posicion" y "TransporteSDN.rotacion", a las
    que
accederá el guión en el momento de la clonación.

La plantilla para añadir nuevos destinos automatizados está recogida a continuación, en
ella deberán añadirse las coordenadas X, Y y Z, la rotación Y, el nombre de la escena y
el texto informativo; y se copiará y pegará a continuación del punto indicado.

Los textos informativos serán traducidos por el Traductor si estos están recogidos en
su base de datos.

#####################################################################*/


static class TransporteSDN extends MonoBehaviour
{
  var activado : boolean = false;
  var posicion : Vector3;
  var rotacion : Quaternion;
  var nombreDestino : String;


  function Destino (lugar : String)
  {
    TransporteSDN.activado = true;
    nombreDestino = "";

    var posicionX : float = 0;
    var posicionY : float = 0;
```

```
58      var posicionZ : float = 0;
59      var rotacionY : float = 0;
60      var escena : String;
61
62      switch (lugar)
63      {
64
65        /* ----------------------------------------------------------------
66        ---------- Plantilla para la introducción de destinos ----------
67        ----------------------------------------------------------------
68        ------- Copiar, pegar a continuación y rellenar los valores ------
69        ----------------------------------------------------------------
70        ----------------------------------------------------------------
71        --------- Nota: en los nombres de Destino se puede usar el -------
72        ------------- código \n para añadir una nueva línea --------------
73              ----------------------------------------------------------------
74
75        case "":
76
77          nombreDestino = "";
78
79          posicionX = ;
80          posicionY = ;
81          posicionZ = ;
82
83          rotacionY = ;
84
85          escena = "";
86
87          break;
88
89        ----------------------------------------------------------------
90        ------------------- Pegar a partir de aquí ----------------------
91        ---------------------------------------------------------------- */
92
93
94
95
96
97        // ----------------------------------------------------------------
98
99        case "TecnologicoDespachoFernandoJFF":
100
101         nombreDestino = "Despacho de\nFernando Jorge Fraile Fernández";
102
103         posicionX = -30.51105;
104         posicionY = 10.39999;
105         posicionZ = -59.27818;
106
107         rotacionY = 180;
108
109         escena = "03b EdifTecnologicoP2";
110
111         break;
112
113       case "EscuelaPlantaBaja":
114
115         nombreDestino = "Escuela de ingenierías\nPlanta baja";
116
```

```
117         posicionX = 18.95518;
118         posicionY = 0.5336218;
119         posicionZ = -46.98695;
120 
121         rotacionY = 45;
122 
123         escena = "02a EscuelaPB";
124 
125         break;
126 
127       case "EscuelaPlantaUno":
128 
129         nombreDestino = "Escuela de ingenierías\nPlanta uno";
130 
131         posicionX = 36.6;
132         posicionY = 5.041617;
133         posicionZ = -37.5;
134 
135         rotacionY = 96.8;
136 
137         escena = "02b EscuelaP1";
138 
139         break;
140 
141       case "Secretaria":
142 
143         nombreDestino = "Secretaría";
144 
145         posicionX = 13.72361;
146         posicionY = 0.5109358;
147         posicionZ = -32.03547;
148 
149         rotacionY = 237.692;
150 
151         escena = "07 Secretaria";
152 
153         break;
154 
155       case "SalonDeActos":
156 
157         nombreDestino = "Salón de actos";
158 
159         posicionX = 56.01186;
160         posicionY = 0.5555206;
161         posicionZ = -55.93216;
162 
163         rotacionY = 135;
164 
165         escena = "04 SalondeActos";
166 
167         break;
168 
169       case "AulaUno":
170 
171         nombreDestino = "Aula 1";
172 
173         posicionX = 37.40165;
174         posicionY = 0.6196227;
175         posicionZ = -17.80277;
```

```
176
177         rotacionY = 22.98051;
178
179         escena = "06a Aula 01";
180
181         break;
182
183       case "AulaDos":
184
185         nombreDestino = "Aula 2";
186
187         posicionX = 26.046;
188         posicionY = 0.7136661;
189         posicionZ = -16.98843;
190
191         rotacionY = 54.51141;
192
193         escena = "06b Aula 02";
194
195         break;
196
197       case "AulaTres":
198
199         nombreDestino = "Aula 3";
200
201         posicionX = 38.27837;
202         posicionY = 0.5499984;
203         posicionZ = -17.05009;
204
205         rotacionY = 28.26678;
206
207         escena = "06c Aula 03";
208
209         break;
210
211       case "AulaCuatro":
212
213         nombreDestino = "Aula 4";
214
215         posicionX = 30.60505;
216         posicionY = 0.7136669;
217         posicionZ = -14.93155;
218
219         rotacionY = 330.3419;
220
221         escena = "06d Aula 04";
222
223         break;
224
225       case "AulaDoscientoscuatro":
226
227         nombreDestino = "Aula 204";
228
229         posicionX = 52.2;
230         posicionY = 4.884663;
231         posicionZ = -17.99036;
232
233         rotacionY = 54.9;
234
```

```
235         escena = "08 Aula 204";
236 
237         break;
238 
239       case "AulaDoscientosdieciseis":
240 
241         nombreDestino = "Aula 216";
242 
243         posicionX = 32.3904;
244         posicionY = 5.031725;
245         posicionZ = -51.33314;
246 
247         rotacionY = 198.1079;
248 
249         escena = "11 Aula 216";
250 
251         break;
252 
253       case "BibliotecaPlantaBaja":
254 
255         nombreDestino = "Biblioteca\nPlanta baja";
256 
257         posicionX = 54.50752;
258         posicionY = 5.090043;
259         posicionZ = -55.37032;
260 
261         rotacionY = 126.4512;
262 
263         escena = "05a BibliotecaAbajo";
264 
265         break;
266 
267       case "BibliotecaPlantaUno":
268 
269         nombreDestino = "Biblioteca\nPlanta uno";
270 
271         posicionX = 55.25053;
272         posicionY = 9.468655;
273         posicionZ = -60.50612;
274 
275         rotacionY = -257.5142;
276 
277         escena = "05b BibliotecaArriba";
278 
279         break;
280 
281       case "ExteriorEntrada":
282 
283         nombreDestino = "Escuela de ingenierías\nEntrada";
284 
285         posicionX = 11.66558;
286         posicionY = 0.01000023;
287         posicionZ = 56.59721;
288 
289         rotacionY = 245.7726;
290 
291         escena = "01 Exterior";
292 
293         break;
```

```
294
295       case "AulaHacheTres":
296
297         nombreDestino = "Aula H3";
298
299         posicionX = -2.780392;
300         posicionY = 5.506086;
301         posicionZ = -64.79453;
302
303         rotacionY = 237.6552;
304
305         escena = "09 Aula H3";
306
307         break;
308
309       case "AseoChicasTecnologicoPlantaBaja":
310
311         nombreDestino = "Edificio tecnológico\nPlanta baja: Aseo (Chicas)";
312
313         posicionX = -45.61321;
314         posicionY = 0.5556262;
315         posicionZ = -27.45911;
316
317         rotacionY = 232.3754;
318
319         escena = "10 BanoET";
320
321         break;
322
323       case "TecnologicoPlantaBaja":
324
325         nombreDestino = "Edificio tecnológico\nPlanta baja";
326
327         posicionX = -36.9;
328         posicionY = 0.609341;
329         posicionZ = -6.890869;
330
331         rotacionY = 180;
332
333         escena = "03a EdifTecnologicoP1";
334
335         break;
336
337       case "TecnologicoPlantaUno":
338
339         nombreDestino = "Edificio tecnológico\nPlanta uno";
340
341         posicionX = -31.24356;
342         posicionY = 5.606005;
343         posicionZ = -13.89931;
344
345         rotacionY = 205.33;
346
347         escena = "03a EdifTecnologicoP1";
348
349         break;
350
351       case "TecnologicoPlantaDos":
352
```

```
353         nombreDestino = "Edificio tecnológico\nPlanta dos";
354 
355         posicionX = -30.18819;
356         posicionY = 10.40001;
357         posicionZ = -13.18063;
358 
359         rotacionY = 203.9158;
360 
361         escena = "03b EdifTecnologicoP2";
362 
363         break;
364 
365       case "TecnologicoPlantaTres":
366 
367         nombreDestino = "Edificio tecnológico\nPlanta tres";
368 
369         posicionX = -30.56983;
370         posicionY = 14.57801;
371         posicionZ = -13.19398;
372 
373         rotacionY = 228.3628;
374 
375         escena = "03b EdifTecnologicoP2";
376 
377         break;
378 
379       case "TecnologicoPlantaCuatroEste":
380 
381         nombreDestino = "Edificio tecnológico\nPlanta cuatro (ala este)";
382 
383         posicionX = -32.79389;
384         posicionY = 18.756;
385         posicionZ = -11.17264;
386 
387         rotacionY = 222.6098;
388 
389         escena = "03b EdifTecnologicoP2";
390 
391         break;
392 
393       case "TecnologicoPlantaCuatroOeste":
394 
395         nombreDestino = "Edificio tecnológico\nPlanta cuatro (ala oeste)";
396 
397         posicionX = -32.50859;
398         posicionY = 18.756;
399         posicionZ = -59.721;
400 
401         rotacionY = 327.524;
402 
403         escena = "03b EdifTecnologicoP2";
404 
405         break;
406 
407       // default: quedarse donde está, activar variable de control y añadir un bloque if
              que, junto a la variable de control,
408       // determine se se realiza el transporte... o activar función de botón cancelar.
409     }
410 
```

```
411     posicion = Vector3(posicionX, posicionY, posicionZ);
412     rotacion = Quaternion.Euler(Vector3(0, rotacionY, 0));
413
414     // En el caso que se añadieran nuevos destinos al sistema de tranporte del SDN
415     // sin asignarles un nombre de destino para mostrar en pantalla, nos aseguramos
416     // de mostrar un nombre de destino genérico.
417
418     if (nombreDestino == "")
419     {
420       nombreDestino = "Has llegado al destino";
421     }
422
423     if (Application.loadedLevelName != escena)
424     {
425       Application.LoadLevel(escena);
426     }
427     // Si el avatar ya está dentro de la escena destino no recargaremos la escena, sólo
          reubicaremos
428     // al avatar en ella para ahorrar recursos, sin embargo, al no recargar la escena hay
           que realizar
429     // manualmente aquellas operaciones que en caso contrario se ejecutarían
          automáticamente al cargar
430     // una nueva escena.
431     else
432     {
433       SDN.avatar.transform.position = posicion;
434       SDN.avatar.transform.rotation = rotacion;
435
436       // Desactivamos manualmente la siguiente variable, pues en circunstancias normales
            sería el
437       // código del SDN en el guión constructor "SustContr.js" quien la desactivaría al
            clonar al
438       // avatar en la nueva escena.
439       TransporteSDN.activado = false;
440     }
441
442     SDN.DesactivarSelectorDestinos();
443
444     // Al activar el selector de destinos se pausa el juego, por tanto, tras completar
445     // el trasnporte lo reanudamos.
446
447     SDN.ReanudarJuego();
448
449     // Indicamos que, puesto que se ha realizado un transporte, se puede informar al
          usuario por
450     // pantalla del nombre del destino al llegar a éste. (Para determinar si se mostrará
          el nombre
451     // por pantalla se comprobarán, además de esta, las variables de control del guión
          maestro
452     // "CamaraMapaControl.js" ajustables en el Inspector del prefab "H_CamaraMapa").
453
454     SDN.mostrarNombreDestino = true;
455   }
456
457
458   function Destino (posicionXManual : float, posicionYManual : float, posicionZManual :
        float, rotacionYManual : float, escenaManual : String, etiquetaDestino : String)
459   {
460     TransporteSDN.activado = true;
```

```
461
462     var posicionX : float = posicionXManual;
463     var posicionY : float = posicionYManual;
464     var posicionZ : float = posicionZManual;
465     var rotacionY : float = rotacionYManual;
466     var escena : String = escenaManual;
467
468     posicion = Vector3(posicionX, posicionY, posicionZ);
469     rotacion = Quaternion.Euler(Vector3(0, rotacionY, 0));
470
471     nombreDestino = Traductor.Traducir(etiquetaDestino);
472
473     if (Application.loadedLevelName != escena)
474     {
475       Application.LoadLevel(escena);
476     }
477     // Si el avatar ya está dentro de la escena destino no recargaremos la escena, sólo
          reubicaremos
478     // al avatar en ella para ahorrar recursos, sin embargo, al no recargar la escena hay
           que realizar
479     // manualmente aquellas operaciones que en caso contrario se ejecutarían
          automáticamente al cargar
480     // una nueva escena.
481     else
482     {
483       SDN.avatar.transform.position = posicion;
484       SDN.avatar.transform.rotation = rotacion;
485
486       // Desactivamos manualmente la siguiente variable, pues en circunstancias normales
            sería el
487       // código del SDN en el guión constructor "SustContr.js" quien la desactivaría al
            clonar al
488       // avatar en la nueva escena.
489       TransporteSDN.activado = false;
490     }
491
492     SDN.DesactivarSelectorDestinos();
493
494     // Al activar el selector de destinos se pausa el juego, por tanto, tras completar
495     // el trasnporte lo reanudamos.
496
497     SDN.ReanudarJuego();
498
499     // Indicamos que, puesto que se ha realizado un transporte, se puede informar al
          usuario por
500     // pantalla del nombre del destino al llegar a éste. (Para determinar si se mostrará
          el nombre
501     // por pantalla se comprobarán, además de esta, las variables de control del guión
          maestro
502     // "CamaraMapaControl.js" ajustables en el Inspector del prefab "H_CamaraMapa").
503
504     SDN.mostrarNombreDestino = true;
505   }
506 }
```

F.31. VistaTeclado.js

```
#pragma strict

/*#################################################################

VistaTeclado.js
---------------

Guión encargado de suavizar los movimientos al desactivar la vista orbital y activar la vista
por teclado.

Durante la vista orbital la cámara principal, gestionada por el guión "MouseOrbit.js", rota
alrededor del avatar mientras este se mueve de forma independiente a ella. Este comportamiento
puede dificultar el manejo del avatar, aspecto que soluciona este guión ubicando la cámara siempre
detrás del avatar, a una distancia y una altura personalizables, pudiendo ver en todo momento hacia
dónde avanza este sin necesidad de tener que recolocar la cámara manualmente constantemente.

Las coordenadas de posición de la cámara mientras está controlada por el guión de "MouseOrbit.js" son
relativas a la posición del avatar y no globales, la cámara es "hija" del avatar y usa por tanto
coordenadas locales. Este guión utiliza en primer lugar las coordenadas locales para ubicar la cámara
detrás del avatar a la distancia y altura prefijadas. Este movimiento lo realiza de forma suavizada
conociendo la posición actual local de la cámara y la posicón local destino. Tras esto desempareja la
cámara del avatar para utilizar coordenadas globales referidas al centro de coordenadas general.
Ubicada la cámara tras el avatar el guión hará seguir la cámara al avatar manteniendo la distancia y
la altura indicadas y moviéndose de forma suavizada.

Las variables personalizables "suavizado", "altura" y "distancia" se introducirán, para comodidad del
desarrollador, en el guión "CámaraMapaControl.js" del prefab "CamaraMapa".

#################################################################*/


// Variables modificables desde el Inspector del prefab "CamaraMapa" que determinan
// la distancia y altura de la cámara tras el avatar.

var suavizado : float;
var altura : float;
var distancia : float;

private var posicionLocalActual : Vector3;
private var posicionLocalDestino : Vector3;

private var posicionActualGlobal : Vector3;
private var posicionDestinoGlobal : Vector3;

private var rotacionActualTotal : Quaternion;
```

```
46 private var rotacionDestinoTotal : Quaternion;
47
48 private var anguloActualEjeY : float;
49 private var anguloDestinoEjeY : float;
50
51 private var anguloSuavizadoEjeY : float;
52
53 private var rotacionSuavizadaEjeY : Quaternion;
54
55 // El estado de la siguiente variable será actualizado apropiadamente
56 // por la función "ActivarVistaOrbital()".
57
58 @HideInInspector var enPosicion : boolean;
59
60
61 function Start ()
62 {
63   // Recogemos los valores de suavizado, altura y distancia del guión "CamaraMapaControl.
       js" de la
64   // cámara del mapa, donde el usuario puede introducirlos centralizadamente.
65
66   // Nota: las comprobaciones de que los valores están dentro de rango se realizan en ese
        guión.
67
68   suavizado = SDN.camaraMapa.GetComponent.<CamaraMapaControl>().suavizadoVistaTeclado;
69   altura = SDN.camaraMapa.GetComponent.<CamaraMapaControl>().alturaVistaTeclado;
70   distancia = SDN.camaraMapa.GetComponent.<CamaraMapaControl>().distanciaVistaTeclado;
71
72   enPosicion = false;
73 }
74
75
76 function LateUpdate ()
77 {
78   // Si la cámara no está detrás del avatar a la distancia y altura especificadas no está
        entonces "en posición",
79   // y hará falta ubicarla en en esa posición de partida.
80
81   if (enPosicion == false)
82   {
83     // Si se acaba de cargar una escena y la vista de teclado estaba activada en la
         escena anterior, no
84     // suavizamos el movimiento de la cámara al colocarse en la posición de partida
         detrás del avatar, y
85     // hacemos que este cambió de posición sea instantáneo para que no haya saltos de
         cámara y la transición
86     // de escena sea completamente natural. Si, en cambio, se ha activado la vista de
         teclado en otro instante
87     // sí suavizamos el movimiento de la cámara hasta su posición de partida de la vista
         de teclado.
88
89     // Para este posicionamiento usamos variables locales referidas al avatar.
90
91     if (SDN.vistaOrbitalNoAlPrincipioEscena == true)
92     {
93       transform.localPosition = Vector3(0, 2, -2);
94       transform.localRotation = Quaternion.identity;
95
96       SDN.vistaOrbitalNoAlPrincipioEscena = false;
```

```
97      }
98      else
99      {
100       // Para el posicionamiento inicial detrás del avatar usamos coordenadas locales
              referidas a éste.
101
102       // Definimos su posición local actual y destino
103
104       posicionLocalActual = transform.localPosition;
105       posicionLocalDestino = Vector3(0, altura, - distancia);
106
107       // Suavizamos el movimiento de la cámara hacia la posición inicial.
108
109       transform.localPosition = Vector3.Lerp(posicionLocalActual, posicionLocalDestino,
              Time.deltaTime * suavizado);
110
111       // Definimos su rotación y la alcanzará de forma suavizada.
112
113       transform.localRotation = Quaternion.Lerp(transform.localRotation, Quaternion.
              identity, Time.deltaTime * suavizado);
114
115       enPosicion = true;
116
117     }
118
119     // Una vez en posición, desemparejamos la cámara del avatar para usar coordenadas
          globales.
120
121     SDN.camaraPrincipal.transform.parent = null;
122
123   }
124   else
125   {
126     // Definimos las rotaciones destino y actual ("quaternion"), así como la rotación
          específica
127     // sobre el eje Y ("float"), para suavizar no sólo la rotación, sino también la
          posición
128     // de la cámara.
129
130     rotacionActualTotal = SDN.camaraPrincipal.transform.rotation;
131     rotacionDestinoTotal = SDN.avatar.transform.transform.rotation;
132
133     anguloActualEjeY = SDN.camaraPrincipal.transform.eulerAngles.y;
134     anguloDestinoEjeY = SDN.avatar.transform.eulerAngles.y;
135
136     // Suavizamos la rotación específica sobre el eje Y.
137
138     anguloSuavizadoEjeY = Mathf.LerpAngle(anguloActualEjeY, anguloDestinoEjeY, Time.
          deltaTime * suavizado);
139
140     // Transformamos el ángulo ("float") en rotación ("quaternion").
141
142     rotacionSuavizadaEjeY = Quaternion.Euler(0, anguloSuavizadoEjeY, 0);
143
144     // Suavizamos la posición de la cámara teniendo en cuenta su rotación.
145
146     transform.position = SDN.avatar.transform.position;
147     transform.position -= rotacionSuavizadaEjeY * Vector3.forward * distancia;
148     transform.position.y = SDN.avatar.transform.position.y + altura;
```

```
149
150     // Suavizamos la rotación de la cámara.
151
152     transform.rotation = Quaternion.Lerp(rotacionActualTotal, rotacionDestinoTotal, Time.
            deltaTime * suavizado);
153   }
154
155   // -------------------------------------------------------
156   // Si no se desea suavizar el movimiento basta con este código
157   // y no hace falta desemparejar la cámara del avatar
158   // -------------------------------------------------------
159   //  transform.localPosition = Vector3(0, 2, -2);
160   //  transform.localRotation = Quaternion.identity;
161   // -------------------------------------------------------
162 }
```

F.32. PuntoDeControlTecnologicoDespachoFernandoJFF.js

```
1  #pragma strict
2
3  /*#####################################################################
4
5  PuntoDeControlTecnologicoDespachoFernandoJFF.js
6  ------------------------------------------------
7
8  Guión que gestiona el punto de control del despacho de Fernando Jorge Fraile Fernández.
9
10 #####################################################################*/
11
12
13 function OnTriggerEnter (colisionador : Collider)
14 {
15   if (colisionador.tag == "PlayerPunto")
16   {
17     SDN.puntoDeControlNavegacionGuiada = true;
18     SDN.dondeEstoy = "TecnologicoDespachoFernandoJFF";
19     LocalizadorDeBalizas.Localizar();
20   }
21 }
22
23
24 function OnTriggerExit (colisionador : Collider)
25 {
26   if (colisionador.tag == "PlayerPunto")
27   {
28     SDN.puntoDeControlNavegacionGuiada = true;
29     SDN.dondeEstoy = "TecnologicoPlantaDosOeste";
30     LocalizadorDeBalizas.Localizar();
31   }
32 }
```

F.33. PuntoDeControlTecnologicoPlantaBaja.js

```
1  #pragma strict
```

```
2
3  /*#####################################################################
4
5  PuntoDeControlTecnologicoPlantaBaja.js
6  ------------------------------------
7
8  Guión que gestiona el punto de control del ala este de la planta baja
9  del edificio tecnológico.
10
11 #####################################################################*/
12
13
14 function OnTriggerEnter (colisionador : Collider)
15 {
16   if (colisionador.tag == "PlayerPunto")
17   {
18     SDN.puntoDeControlNavegacionGuiada = true;
19     SDN.dondeEstoy = "TecnologicoPlantaBaja";
20     LocalizadorDeBalizas.Localizar();
21   }
22 }
```

F.34. PuntoDeControlTecnologicoPlantaBajaIntersticial.js

```
1  #pragma strict
2
3  /*#####################################################################
4
5  PuntoDeControlTecnologicoPlantaBajaIntersticial.js
6  -------------------------------------------------
7
8  Guión que gestiona el punto de control instersticial entre las alas este
       y oeste de
9  la planta baja del edificio tecnológico para resolver el escenario en el
       que el avatar
10 se transportara justa y directamente en el punto medio de las dos alas,
     donde los puntos
11 de control estándar no tienen control pero sí los puntos de control
     intersticiales.
12
13 #####################################################################*/
14
15
16 function OnTriggerEnter (colisionador : Collider)
17 {
18   if (colisionador.tag == "PlayerPunto")
19   {
20     // La siguiente instrucción "yield" sirve de seguro en caso de que
          el avatar se
```

```
21      // haya transportado de tal forma que su posición abarque tanto al
            punto de control
22      // intersticial como al punto de control de cualquiera de las dos
            alas. El código se
23      // ejecutará así al final del "frame", superado ya el ciclo "Update
            ()", dando tiempo
24      // al otro punto de control de asignar él la información relativa a
            la ubicación actual.
25
26      yield WaitForEndOfFrame;
27
28      // Si aún no se ha asignado una ubicación...
29      // (Nota: esto equivale a decir que el avatar se ha transportado al
            punto
30      // intermedio entre el ala este y el ala oeste).
31      if (SDN.dondeEstoy == "")
32      {
33        SDN.puntoDeControlNavegacionGuiada = true;
34        SDN.dondeEstoy = "TecnologicoPlantaBaja";
35        LocalizadorDeBalizas.Localizar();
36      }
37    }
38  }
```

F.35. PuntoDeControlTecnologicoPlantaBajaOeste.js

```
1  #pragma strict
2
3  /*#####################################################################
4
5  PuntoDeControlTecnologicoPlantaBajaOeste.js
6  -------------------------------------------
7
8  Guión que gestiona el punto de control del ala oeste de la planta baja
9  del edificio tecnológico.
10
11 Contiene un bloque de código especial, encargado de simular que el ala
12 oeste de la planta baja del edificio tecnológico es una extensión del
13 ala este cuando el destino de la navegación guiada sea la planta baja
14 de este edificio.
15
16 #####################################################################*/
17
18
19 function OnTriggerEnter (colisionador : Collider)
20 {
21   if (colisionador.tag == "PlayerPunto")
22   {
23     // El siguiente bloque de código indica al intersectador cuando la navegación guiada
24     // está activada y el destino elegido es la planta baja del edificio tecnológico que
```

```
25      // ya se ha llegado al destino, independientemente de si estamos en el ala este u
            oeste.
26      if (SDN.navegacionGuiada == true && SDN.dondeDeboEstar == "TecnologicoPlantaBaja")
27      {
28        SDN.navegacionGuiadaForzarLlegada = true;
29      }
30
31      SDN.puntoDeControlNavegacionGuiada = true;
32      SDN.dondeEstoy = "TecnologicoPlantaBajaOeste";
33      LocalizadorDeBalizas.Localizar();
34    }
35  }
```

F.36. PuntoDeControlTecnologicoPlantaBajaRellanoEscaleras.js

```
1  #pragma strict
2
3  /*#################################################################
4
5  PuntoDeControlTecnologicoPlantaBajaRellanoEscaleras.js
6  ------------------------------------------------------
7
8  Guión que gestiona el punto de control del rellano de las escaleras
9  del ala este de la planta baja del edificio tecnológico.
10
11 #################################################################*/
12
13
14 function OnTriggerEnter (colisionador : Collider)
15 {
16   if (colisionador.tag == "PlayerPunto")
17   {
18     SDN.puntoDeControlNavegacionGuiada = true;
19     SDN.dondeEstoy = "TecnologicoPlantaBajaRellanoEscaleras";
20     LocalizadorDeBalizas.Localizar();
21   }
22 }
```

F.37. PuntoDeControlTecnologicoPlantaBajaRellanoEscalerasOeste.js

```
1  #pragma strict
2
3  /*#################################################################
4
5  PuntoDeControlTecnologicoPlantaBajaRellanoEscalerasOeste.js
6  -----------------------------------------------------------
7
8  Guión que gestiona el punto de control del rellano de las escaleras
9  del ala oeste de la planta baja del edificio tecnológico.
10
11 #################################################################*/
12
13
```

```
14 function OnTriggerEnter (colisionador : Collider)
15 {
16   if (colisionador.tag == "PlayerPunto")
17   {
18     SDN.puntoDeControlNavegacionGuiada = true;
19     SDN.dondeEstoy = "TecnologicoPlantaBajaRellanoEscalerasOeste";
20     LocalizadorDeBalizas.Localizar();
21   }
22 }
```

F.38. PuntoDeControlTecnologicoPlantaCuatroEste.js

```
1  #pragma strict
2
3  /*#####################################################################
4
5  PuntoDeControlTecnologicoPlantaCuatroEste.js
6  ------------------------------------------
7
8  Guión que gestiona el punto de control del ala este de la planta cuatro
9  del edificio tecnológico.
10
11 #####################################################################*/
12
13
14 function OnTriggerEnter (colisionador : Collider)
15 {
16   if (colisionador.tag == "PlayerPunto")
17   {
18     SDN.puntoDeControlNavegacionGuiada = true;
19     SDN.dondeEstoy = "TecnologicoPlantaCuatroEste";
20     LocalizadorDeBalizas.Localizar();
21   }
22 }
```

F.39. PuntoDeControlTecnologicoPlantaCuatroOeste.js

```
1  #pragma strict
2
3  /*#####################################################################
4
5  PuntoDeControlTecnologicoPlantaCuatroOeste.js
6  ------------------------------------------
7
8  Guión que gestiona el punto de control del ala oeste de la planta cuatro
9  del edificio tecnológico.
10
11 #####################################################################*/
12
13
14 function OnTriggerEnter (colisionador : Collider)
15 {
16   if (colisionador.tag == "PlayerPunto")
17   {
```

```
    SDN.puntoDeControlNavegacionGuiada = true;
    SDN.dondeEstoy = "TecnologicoPlantaCuatroOeste";
    LocalizadorDeBalizas.Localizar();
  }
}
```

F.40. PuntoDeControlTecnologicoPlantaDos.js

```
#pragma strict

/*#####################################################################

PuntoDeControlTecnologicoPlantaDos.js
-------------------------------------

Guión que gestiona el punto de control del ala este de la planta dos
del edificio tecnológico.

#####################################################################*/


function OnTriggerEnter (colisionador : Collider)
{
  if (colisionador.tag == "PlayerPunto")
  {
    SDN.puntoDeControlNavegacionGuiada = true;
    SDN.dondeEstoy = "TecnologicoPlantaDos";
    LocalizadorDeBalizas.Localizar();
  }
}
```

F.41. PuntoDeControlTecnologicoPlantaDosIntersticial.js

```
#pragma strict

/*#####################################################################

PuntoDeControlTecnologicoPlantaDosIntersticial.js
-------------------------------------------------

Guión que gestiona el punto de control instersticial entre las alas este y oeste de
la planta dos del edificio tecnológico para resolver el escenario en el que el avatar
se transportara justa y directamente en el punto medio de las dos alas, donde los puntos
de control estándar no tienen control pero sí los puntos de control intersticiales.

#####################################################################*/


function OnTriggerEnter (colisionador : Collider)
{
  if (colisionador.tag == "PlayerPunto")
  {
    // La siguiente instrucción "yield" sirve de seguro en caso de que el avatar se
    // haya transportado de tal forma que su posición abarque tanto al punto de control
```

```
22      // intersticial como al punto de control de cualquiera de las dos alas.
23
24      yield WaitForEndOfFrame;
25
26      // Si aún no se ha asignado una ubicación...
27      // (Nota: esto equivale a decir que el avatar se ha transportado al punto
28      // intermedio entre el ala este y el ala oeste).
29      if (SDN.dondeEstoy == "")
30      {
31        SDN.puntoDeControlNavegacionGuiada = true;
32        SDN.dondeEstoy = "TecnologicoPlantaDos";
33        LocalizadorDeBalizas.Localizar();
34      }
35    }
36  }
```

F.42. PuntoDeControlTecnologicoPlantaDosOeste.js

```
1  #pragma strict
2
3  /*#####################################################################
4
5  PuntoDeControlTecnologicoPlantaDosOeste.js
6  ------------------------------------
7
8  Guión que gestiona el punto de control del ala oeste de la planta dos
9  del edificio tecnológico.
10
11 #####################################################################*/
12
13
14 function OnTriggerEnter (colisionador : Collider)
15 {
16   if (colisionador.tag == "PlayerPunto")
17   {
18     SDN.puntoDeControlNavegacionGuiada = true;
19     SDN.dondeEstoy = "TecnologicoPlantaDosOeste";
20     LocalizadorDeBalizas.Localizar();
21   }
22 }
```

F.43. PuntoDeControlTecnologicoPlantaDosRellanoEscaleras.js

```
1  #pragma strict
2
3  /*#####################################################################
4
5  PuntoDeControlTecnologicoPlantaDosRellanoEscaleras.js
6  ----------------------------------------------------
7
8  Guión que gestiona el punto de control del rellano de las escaleras
9  del ala este de la planta dos del edificio tecnológico.
10
11 #####################################################################*/
```

```
12
13
14 function OnTriggerEnter (colisionador : Collider)
15 {
16   if (colisionador.tag == "PlayerPunto")
17   {
18     SDN.puntoDeControlNavegacionGuiada = true;
19     SDN.dondeEstoy = "TecnologicoPlantaDosRellanoEscaleras";
20     LocalizadorDeBalizas.Localizar();
21   }
22 }
```

F.44. PuntoDeControlTecnologicoPlantaDosRellanoEscalerasOeste.js

```
1  #pragma strict
2
3  /*#####################################################################
4
5  PuntoDeControlTecnologicoPlantaDosRellanoEscalerasOeste.js
6  ----------------------------------------------------------
7
8  Guión que gestiona el punto de control del rellano de las escaleras
9  del ala oeste de la planta dos del edificio tecnológico.
10
11 #####################################################################*/
12
13
14 function OnTriggerEnter (colisionador : Collider)
15 {
16   if (colisionador.tag == "PlayerPunto")
17   {
18     SDN.puntoDeControlNavegacionGuiada = true;
19     SDN.dondeEstoy = "TecnologicoPlantaDosRellanoEscalerasOeste";
20     LocalizadorDeBalizas.Localizar();
21   }
22 }
```

F.45. PuntoDeControlTecnologicoPlantaTres.js

```
1  #pragma strict
2
3  /*#####################################################################
4
5  PuntoDeControlTecnologicoPlantaTres.js
6  --------------------------------------
7
8  Guión que gestiona el punto de control del ala este de la planta tres
9  del edificio tecnológico.
10
11 #####################################################################*/
```

```
12
13
14 function OnTriggerEnter (colisionador : Collider)
15 {
16   if (colisionador.tag == "PlayerPunto")
17   {
18     SDN.puntoDeControlNavegacionGuiada = true;
19     SDN.dondeEstoy = "TecnologicoPlantaTres";
20     LocalizadorDeBalizas.Localizar();
21   }
22 }
```

F.46. PuntoDeControlTecnologicoPlantaTresIntersticial.js

```
1  #pragma strict
2
3  /*#################################################################
4
5  PuntoDeControlTecnologicoPlantaTresIntersticial.js
6  ------------------------------------------------
7
8  Guión que gestiona el punto de control instersticial entre las alas este y oeste de
9  la planta tres del edificio tecnológico para resolver el escenario en el que el avatar
10 se transportara justa y directamente en el punto medio de las dos alas, donde los puntos
11 de control estándar no tienen control pero sí los puntos de control intersticiales.
12
13 ################################################################*/
14
15
16 function OnTriggerEnter (colisionador : Collider)
17 {
18   if (colisionador.tag == "PlayerPunto")
19   {
20     // La siguiente instrucción "yield" sirve de seguro en caso de que el avatar se
21     // haya transportado de tal forma que su posición abarque tanto al punto de control
22     // intersticial como al punto de control de cualquiera de las dos alas.
23
24     yield WaitForEndOfFrame;
25
26     // Si aún no se ha asignado una ubicación...
27     // (Nota: esto equivale a decir que el avatar se ha transportado al punto
28     // intermedio entre el ala este y el ala oeste).
29     if (SDN.dondeEstoy == "")
30     {
31       SDN.puntoDeControlNavegacionGuiada = true;
32       SDN.dondeEstoy = "TecnologicoPlantaTres";
33       LocalizadorDeBalizas.Localizar();
34     }
35   }
36 }
```

F.47. PuntoDeControlTecnologicoPlantaTresOeste.js

```
#pragma strict

/*#####################################################################

PuntoDeControlTecnologicoPlantaTresOeste.js
-------------------------------------

Guión que gestiona el punto de control del ala oeste de la planta tres
del edificio tecnológico.

#####################################################################*/


function OnTriggerEnter (colisionador : Collider)
{
  if (colisionador.tag == "PlayerPunto")
  {
    SDN.puntoDeControlNavegacionGuiada = true;
    SDN.dondeEstoy = "TecnologicoPlantaTresOeste";
    LocalizadorDeBalizas.Localizar();
  }
}
```

F.48. PuntoDeControlTecnologicoPlantaUno.js

```
#pragma strict

/*#####################################################################

PuntoDeControlTecnologicoPlantaUno.js
-------------------------------------

Guión que gestiona el punto de control del ala este de la planta uno
del edificio tecnológico.

#####################################################################*/


function OnTriggerEnter (colisionador : Collider)
{
  if (colisionador.tag == "PlayerPunto")
  {
    SDN.puntoDeControlNavegacionGuiada = true;
    SDN.dondeEstoy = "TecnologicoPlantaUno";
    LocalizadorDeBalizas.Localizar();
  }
}
```

F.49. PuntoDeControlTecnologicoPlantaUnoIntersticial.js

```
#pragma strict
```

```
2
3  /*#################################################################
4
5  PuntoDeControlTecnologicoPlantaUnoIntersticial.js
6  ------------------------------------------------
7
8  Guión que gestiona el punto de control instersticial entre las alas este y oeste de
9  la planta uno del edificio tecnológico para resolver el escenario en el que el avatar
10 se transportara justa y directamente en el punto medio de las dos alas, donde los puntos
11 de control estándar no tienen control pero sí los puntos de control intersticiales.
12
13 #################################################################*/
14
15
16 function OnTriggerEnter (colisionador : Collider)
17 {
18   if (colisionador.tag == "PlayerPunto")
19   {
20     // La siguiente instrucción "yield" sirve de seguro en caso de que el avatar se
21     // haya transportado de tal forma que su posición abarque tanto al punto de control
22     // intersticial como al punto de control de cualquiera de las dos alas.
23
24     yield WaitForEndOfFrame;
25
26     // Si aún no se ha asignado una ubicación...
27     // (Nota: esto equivale a decir que el avatar se ha transportado al punto
28     // intermedio entre el ala este y el ala oeste).
29     if (SDN.dondeEstoy == "")
30     {
31       SDN.puntoDeControlNavegacionGuiada = true;
32       SDN.dondeEstoy = "TecnologicoPlantaUno";
33       LocalizadorDeBalizas.Localizar();
34     }
35   }
36 }
```

F.50. PuntoDeControlTecnologicoPlantaUnoOeste.js

```
1  #pragma strict
2
3  /*#################################################################
4
5  PuntoDeControlTecnologicoPlantaUnoOeste.js
6  ------------------------------------------
7
8  Guión que gestiona el punto de control del ala oeste de la planta uno
9  del edificio tecnológico.
10
11 #################################################################*/
12
13
14 function OnTriggerEnter (colisionador : Collider)
15 {
16   if (colisionador.tag == "PlayerPunto")
17   {
18     SDN.puntoDeControlNavegacionGuiada = true;
19     SDN.dondeEstoy = "TecnologicoPlantaUnoOeste";
```

```
20      LocalizadorDeBalizas.Localizar();
21    }
22  }
```

F.51. PuntoDeControlTecnologicoPlantaUnoRellanoEscaleras.js

```
1  #pragma strict
2
3  /*#################################################################
4
5  PuntoDeControlTecnologicoPlantaUnoRellanoEscaleras.js
6  ----------------------------------------------------
7
8  Guión que gestiona el punto de control del rellano de las escaleras
9  del ala este de la planta uno del edificio tecnológico.
10
11 #################################################################*/
12
13
14 function OnTriggerEnter (colisionador : Collider)
15 {
16   if (colisionador.tag == "PlayerPunto")
17   {
18     SDN.puntoDeControlNavegacionGuiada = true;
19     SDN.dondeEstoy = "TecnologicoPlantaUnoRellanoEscaleras";
20     LocalizadorDeBalizas.Localizar();
21   }
22 }
```

F.52. PuntoDeControlTecnologicoPlantaUnoRellanoEscalerasOeste.js

```
1  #pragma strict
2
3  /*#################################################################
4
5  PuntoDeControlTecnologicoPlantaUnoRellanoEscalerasOeste.js
6  ----------------------------------------------------------
7
8  Guión que gestiona el punto de control del rellano de las escaleras
9  del ala oeste de la planta uno del edificio tecnológico.
10
11 #################################################################*/
12
13
14 function OnTriggerEnter (colisionador : Collider)
15 {
16   if (colisionador.tag == "PlayerPunto")
17   {
18     SDN.puntoDeControlNavegacionGuiada = true;
19     SDN.dondeEstoy = "TecnologicoPlantaUnoRellanoEscalerasOeste";
20     LocalizadorDeBalizas.Localizar();
21   }
22 }
```

F.53. CreadorDePlanos.cs

```
#if UNITY_EDITOR

/*##################################################################

CreadorDePlanos.cs
------------------

Guión encargado de crear los planos que servirán de contenedor a los mapas
capturados por el sistema de captura del Proyecto H.

Los planos creados por este guión están compuestos por un mínimo de dos
triángulos, aliviando así el consumo de recursos cuando se comparán con los
planos estándar consituidos por una rejilla de cien triángulos.

Los planos creados son configurables. Los valores configurables son los
siguientes: número de triángulos a lo ancho y a lo largo, dimensiones de
acho y largo, orientación (horizontal o vertical), posición del pivote
(ubicado en el centro, en las esquinas o en los puntos medios de los lados),
inclusión o no de un componente "collider", inclusión o no de un guión, opción
de crearlo directamente en el origen y nombre a asignar al plano al crearlo.

Si a la hora de definir las propiedades del plano se activa la casilla
"Capa Mapa H", se le asignará a la capa que permitira a la camara del mapa
verlo y mostrar su contenido.

Para acceder al menú de creación de estos planos personalizados hay que dirigirse
al menú: "GameObject/Create Other/Plano personalizado H" dentro del editor de Unity.


Autoría:

El creador del código original es Michael Garforth: http://wiki.unity3d.com/index.php/
    User:Mike

Código modificado, adaptado y traducido por mí (H.), para el Proyecto H.

##################################################################*/


using UnityEngine;
using UnityEditor;
using System.Collections;
using System.IO;


public class CreatePlane : ScriptableWizard
{

  public enum Orientacion
  {
    Horizontal,
    Vertical
  }

  public enum Pivote
  {
```

```
56      SuperiorIzquierda,
57      MitadSuperior,
58      SuperiorDerecha,
59      MitadDerecha,
60      InferiorDerecha,
61      MitadInferior,
62      InferiorIzquierda,
63      MitadIzquierda,
64      Centro
65    }
66
67    public int numSegmentosAncho = 1;
68    public int numSegmentosLargo = 1;
69    public float ancho = 1.0f;
70    public float largo = 1.0f;
71    public Orientacion orientacion = Orientacion.Horizontal;
72    public Pivote pivote = Pivote.Centro;
73    public bool agregarCollider = false;
74    public bool crearEnElOrigen = true;
75    public bool capaMapaH = false;
76    public string nombreDelPlano;
77
78    static Camera cam;
79    static Camera lastUsedCam;
80
81
82    [MenuItem("GameObject/Create Other/Plano personalizado H")]
83    static void CreateWizard()
84    {
85      cam = Camera.current;
86
87      // Si la ventana de escena no tiene el foco, "Camera.current" no devuelve la cámara
            del editor,
88      // se solventa con el siguiente código.
89
90      if (!cam)
91        cam = lastUsedCam;
92      else
93        lastUsedCam = cam;
94      ScriptableWizard.DisplayWizard("Crear plano personalizado H",typeof(CreatePlane));
95    }
96
97
98    void OnWizardUpdate()
99    {
100     numSegmentosAncho = Mathf.Clamp(numSegmentosAncho, 1, 254);
101     numSegmentosLargo = Mathf.Clamp(numSegmentosLargo, 1, 254);
102   }
103
104
105   void OnWizardCreate()
106   {
107     GameObject plane = new GameObject();
108
109     if (!string.IsNullOrEmpty(nombreDelPlano))
110       plane.name = nombreDelPlano;
111     else
112       plane.name = "Plano";
113
```

```
114     if (!crearEnElOrigen && cam)
115       plane.transform.position = cam.transform.position + cam.transform.forward*5.0f;
116     else
117       plane.transform.position = Vector3.zero;
118
119     Vector2 pivoteOffset;
120     string pivoteId;
121     switch (pivote)
122     {
123     case Pivote.SuperiorIzquierda:
124       pivoteOffset = new Vector2(-ancho/2.0f,largo/2.0f);
125       pivoteId = "SI";
126       break;
127     case Pivote.MitadSuperior:
128       pivoteOffset = new Vector2(0.0f,largo/2.0f);
129       pivoteId = "MS";
130       break;
131     case Pivote.SuperiorDerecha:
132       pivoteOffset = new Vector2(ancho/2.0f,largo/2.0f);
133       pivoteId = "SD";
134       break;
135     case Pivote.MitadDerecha:
136       pivoteOffset = new Vector2(ancho/2.0f,0.0f);
137       pivoteId = "MD";
138       break;
139     case Pivote.InferiorDerecha:
140       pivoteOffset = new Vector2(ancho/2.0f,-largo/2.0f);
141       pivoteId = "ID";
142       break;
143     case Pivote.MitadInferior:
144       pivoteOffset = new Vector2(0.0f,-largo/2.0f);
145       pivoteId = "MI";
146       break;
147     case Pivote.InferiorIzquierda:
148       pivoteOffset = new Vector2(-ancho/2.0f,-largo/2.0f);
149       pivoteId = "II";
150       break;
151     case Pivote.MitadIzquierda:
152       pivoteOffset = new Vector2(-ancho/2.0f,0.0f);
153       pivoteId = "MI";
154       break;
155     case Pivote.Centro:
156     default:
157       pivoteOffset = Vector2.zero;
158       pivoteId = "C";
159       break;
160     }
161
162     MeshFilter meshFilter = (MeshFilter)plane.AddComponent(typeof(MeshFilter));
163     plane.AddComponent(typeof(MeshRenderer));
164
165     string planeAssetName = plane.name + "_" + numSegmentosAncho + "x" +
            numSegmentosLargo + "A" + ancho + "L" + largo + (orientacion == Orientacion.
            Horizontal? "H" : "V") + pivoteId + ".asset";
166     Mesh m = (Mesh)AssetDatabase.LoadAssetAtPath("Assets/Editor/" + planeAssetName,typeof
            (Mesh));
167
168     if (m == null)
169     {
```

```
170         m = new Mesh();
171         m.name = plane.name;
172 
173         int hCount2 = numSegmentosAncho+1;
174         int vCount2 = numSegmentosLargo+1;
175         int numTriangles = numSegmentosAncho * numSegmentosLargo * 6;
176         int numVertices = hCount2 * vCount2;
177 
178         Vector3[] vertices = new Vector3[numVertices];
179         Vector2[] uvs = new Vector2[numVertices];
180         int[] triangles = new int[numTriangles];
181 
182         int index = 0;
183         float uvFactorX = 1.0f/numSegmentosAncho;
184         float uvFactorY = 1.0f/numSegmentosLargo;
185         float scaleX = ancho/numSegmentosAncho;
186         float scaleY = largo/numSegmentosLargo;
187         for (float y = 0.0f; y < vCount2; y++)
188         {
189           for (float x = 0.0f; x < hCount2; x++)
190           {
191             if (orientacion == Orientacion.Horizontal)
192             {
193               vertices[index] = new Vector3(x*scaleX - ancho/2f - pivoteOffset.x, 0.0f, y*
                      scaleY - largo/2f - pivoteOffset.y);
194             }
195             else
196             {
197               vertices[index] = new Vector3(x*scaleX - ancho/2f - pivoteOffset.x, y*scaleY
                      - largo/2f - pivoteOffset.y, 0.0f);
198             }
199             uvs[index++] = new Vector2(x*uvFactorX, y*uvFactorY);
200           }
201         }
202 
203         index = 0;
204         for (int y = 0; y < numSegmentosLargo; y++)
205         {
206           for (int x = 0; x < numSegmentosAncho; x++)
207           {
208             triangles[index]   = (y     * hCount2) + x;
209             triangles[index+1] = ((y+1) * hCount2) + x;
210             triangles[index+2] = (y     * hCount2) + x + 1;
211 
212             triangles[index+3] = ((y+1) * hCount2) + x;
213             triangles[index+4] = ((y+1) * hCount2) + x + 1;
214             triangles[index+5] = (y     * hCount2) + x + 1;
215             index += 6;
216           }
217         }
218 
219         m.vertices = vertices;
220         m.uv = uvs;
221         m.triangles = triangles;
222         m.RecalculateNormals();
223 
224         if (!Directory.Exists("Assets/Planos personalizados H"))
225           AssetDatabase.CreateFolder("Assets", "Planos personalizados H");
226 
```

```
227         AssetDatabase.CreateAsset(m, "Assets/Planos personalizados H/" + planeAssetName);
228         AssetDatabase.SaveAssets();
229       }
230
231       meshFilter.sharedMesh = m;
232       m.RecalculateBounds();
233
234       if (agregarCollider)
235         plane.AddComponent(typeof(BoxCollider));
236
237       if (capaMapaH)
238         plane.layer = LayerMask.NameToLayer("Mapa");
239
240       Selection.activeObject = plane;
241     }
242 }
243
244 #endif
```

F.54. SustContr.js

Guión del proyecto legado modificado por el presente.

```
1  #pragma strict
2
3  /*#####################################################################
4
5  SustContr.js
6  ------------
7
8  Guión altamente modificado del proyecto legado para el Proyecto H.
9  Encargado originalmente de sustuir el controlador estándar por el
10 controlador Mixano y reubicarlo en la escena, se le han añadido
11 ahora las funciones de transporte SDN del proyecto H al tiempo
12 que conserva armónicamente el sistema de transporte antiguo, mejorado
13 ahora éste para que todas sus variables se encuentren centralizadas
14 en este mismo guión, y reestructurado para que sus instrucciones
15 se ejecuten organizadamente en los ciclos de ejecución apropiados.
16
17 Las coordenadas de las que dependía el sistema de transporte antiguo
18 se han recalculado con precisión para evitar que el avatar aparezca por
19 encima del suelo como sucecía anteriormente, y se han añadido comentarios
20 a cada entrada para informar de la ubicación con la que se corresponden.
21
22 Asigna ahora también globalmente las variables correspondientes al avatar,
23 a la cámara principal, y a la cámara del mapa, de forma que estén accesibles
24 al resto de elementos del proyecto.
25
26 Se incluye al final la versión original del guión como referencia.
27
28 #####################################################################*/
29
30
31 var controlador1 : GameObject;
32 var controlador2 : GameObject;
```

```
33 var controlador3 : GameObject;
34 var controlador4 : GameObject;
35 var controlador5 : GameObject;
36 var controlador6 : GameObject;
37 var contrOrig : GameObject;
38
39 // Variables públicas a las que se le asignarán desde el Inspector, mediante el recurso "
       CambioControlador"
40 // incluido en la escena, los prefab de la cámara que gestionará el mapa y los
       indicadores de posición de
41 // los elementos en el mapa.
42
43 var camaraMapa : GameObject;
44
45 // Variables internas que gestionarán el seguimiento de los objetos clonados y que
       permitirán
46 // referenciarlos a la hora de trabajar con ellos.
47
48 private var controladorClon : GameObject;
49 private var camaraMapaClon : GameObject;
50
51
52 // El guión original ejecutaba sus instrucciones en el ciclo "Start()". Esto resultaba en
        falta de control
53 // del orden de ejecución en paralelo del código del resto de elementos y causaba
       comportamientos inesperados
54 // y no deseados. Para resolverlo se utiliza el ciclo "Awake()", anterior al "Star()", y
       se ha adaptado y
55 // mejorado todo el código que dependía originalmente de este guión para ajustar su
       funcionamiento al nuevo
56 // método.
57
58 function Awake ()
59 {
60   // (H.)
61
62   // En el sitema legado, la forma de colocar el avatar en la posición deseada tras un
        cambio de escenas
63   // consiste en reubicar el controlador predeterminado, destruirlo y clonar en esa
        posición el controlador
64   // Mixano. Los elementos encargados de realizar esta tarea están dispersos y el orden
        de la ejecución
65   // de estos pasos no está específicamente definido, por lo que puede llegar a resultar
        que el controlador
66   // Mixano sustituyera al predeterminado antes de haberse movido este.
67
68   // Para solucionar este problema sin modificar la esencia del sistema de transporte
        legado centralizaremos
69   // aquí las intrucciones y les asignaremos un orden perfectamente definido, para que
        cada paso se realice
70   // exactamente cuando debe con relación al resto.
71
72   // Si bien las coordenadas X y Z de la posición de transporte se han mantenido
        inalteradas, se ha aprovechado
73   // para modificar el valor de las coordenadas Y de cada posición, ajustándolas
        individualmente con precisión
74   // para evitar así que el avatar aparezca por encima del suelo y caiga hasta este al
        cargar una nueva escena,
75   // tal y como ocurría en el sistema legado.
```

```
76
77  // También se han independizado las rotaciones del avatar de las rotaciones del objeto "ControlProj" incluido
78  // en cada escena y que representa al controlador predeterminado. Se hace esto para evitar que la reubicación
79  // de este elemento en la escena afecte a la posición del avatar como sí lo haría de conservar la dependencia
80  // entre sus rotaciones del sistema anterior; para ello se han recalculado todas las rotaciones respecto al
81  // centro global de coordenadas.
82
83  // Por último se han añadido comentarios de ayuda para cada destino para ayudar a identificar el lugar al que
84  // corresponden sin necesidad de acudir a las coordenadas indicadas.
85
86  // Nota: Los casos "default" se correspondento con la posición del objeto "ControlProj" en la escena.
87
88  // (Nótese que para llamar a la variable "CambioPuerta.Puerta" se conserva la notación en mayúsculas
89  // utilizada en el sistema legado. La notación más adecuada para referise a una variable sería en
90  // minúsuclas "CambioPuerta.puerta". No se modifica no obstante para mantener la consistencia del sistema.
91
92
93  // --------
94  // FASE UNO
95  // --------
96
97  // Reposicionamiento del controlador predeterminando.
98
99
100 switch (Application.loadedLevelName)
101 {
102
103   case "01 Exterior":
104     switch (CambioPuerta.Puerta)
105     {
106       // Acceso sudoeste a la escuela desde la entrada principal.
107       case 1:
108         contrOrig.transform.position = Vector3(-10.5, 0.45, 53);
109         contrOrig.transform.rotation.eulerAngles = Vector3(0, 90, 0);
110
111         break;
112
113       // Acceso noroeste a la escuela desde la entrada principal.
114       case 2:
115         contrOrig.transform.position = Vector3(-10.5, 0.45, 55.4);
116         contrOrig.transform.rotation.eulerAngles = Vector3(0, 90, 0);
117
118         break;
119
120       // Acceso sudeste a la escuela desde el edificio tecnológico.
121       case 3:
122         contrOrig.transform.position = Vector3(-32, 0.48, 22.5);
123         contrOrig.transform.rotation.eulerAngles = Vector3(0, 135, 0);
124
125         break;
```

```
126
127         // Acceso noreste a la escuela desde el edificio tecnológico.
128         case 4:
129           contrOrig.transform.position = Vector3(-42.5, 0.63, 22);
130           contrOrig.transform.rotation.eulerAngles = Vector3(0, 225, 0);
131
132           break;
133
134         // Acceso sudeste del edificio tecnológico desde la escuela.
135         case 5:
136           contrOrig.transform.position = Vector3(-36, 0.63, -1.5);
137           contrOrig.transform.rotation.eulerAngles = Vector3(0, 0, 0);
138
139           break;
140
141         // Acesso noreste del edificio tecnológico desde la escuela.
142         case 6:
143           contrOrig.transform.position = Vector3(-39.5, 0.63, -1.5);
144           contrOrig.transform.rotation.eulerAngles = Vector3(0, 0, 0);
145
146           break;
147
148         // Posición a la que aparecerá el controlador en el exterior al iniciar el juego.
149
150         default:
151           contrOrig.transform.position = Vector3(18.76768, -0.1376796, 54.31792);
152           contrOrig.transform.rotation.eulerAngles = Vector3(0, 270, 0);
153       }
154
155       break;
156
157
158     case "02a EscuelaPB":
159
160       switch (CambioPuerta.Puerta)
161       {
162         // Acceso sudoeste a la escuela desde la entrada principal.
163         case 1:
164           contrOrig.transform.position = Vector3(14, 0.55, -50.2);
165           contrOrig.transform.rotation.eulerAngles = Vector3(0, 45, 0);
166
167           break;
168
169         // Acceso noroeste a la escuela desde la entrada principal.
170         case 2:
171           contrOrig.transform.position = Vector3(15.5, 0.55, -52.1);
172           contrOrig.transform.rotation.eulerAngles = Vector3(0, 45, 0);
173
174           break;
175
176         // Puerta sur de acceso a la escuela desde el edificio tecnológico.
177         case 3:
178           contrOrig.transform.position = Vector3(8.4, 0.55, -14.5);
179           contrOrig.transform.rotation.eulerAngles = Vector3(0, 90, 0);
180
181           break;
182
183         // Puerta norte de acceso a la escuela desde el edificio tecnológico.
184         case 4:
```

```
185           contrOrig.transform.position = Vector3(13.7, 0.55, -8.5);
186           contrOrig.transform.rotation.eulerAngles = Vector3(0, 180, 0);
187
188           break;
189
190         // Escaleras sur entre la planta baja y la uno de la escuela.
191         case 7:
192           contrOrig.transform.position = Vector3(28.3, 2.78, -37);
193           contrOrig.transform.rotation.eulerAngles = Vector3(0, -30, 0);
194
195           break;
196
197         // Escaleras sur entre la planta baja y la uno de la escuela.
198         case 8:
199           contrOrig.transform.position = Vector3(37.5, 2.78, -29);
200           contrOrig.transform.rotation.eulerAngles = Vector3(0, 150, 0);
201
202           break;
203
204         // Puertas de acceso al salón de actos.
205         case 9:
206           contrOrig.transform.position = Vector3(46, 0.55, -47);
207           contrOrig.transform.rotation.eulerAngles = Vector3(0, 315, 0);
208
209           break;
210
211         // Puerta de acceso al aula 04.
212         case 10:
213           contrOrig.transform.position = Vector3(31.4, 0.55, -18.9);
214           contrOrig.transform.rotation.eulerAngles = Vector3(0, 180, 0);
215
216           break;
217
218         // Puerta de acceso a la secretaría.
219         case 11:
220           contrOrig.transform.position = Vector3(16, 0.55, -30.3);
221           contrOrig.transform.rotation.eulerAngles = Vector3(0, 90, 0);
222
223           break;
224
225         // (Punto de transporte presente en el sistema legado pero no utilizado
226         // la ubicación se corresponde con un punto entre la conserjería y las
227         // escaleras sur de acceso a la planta uno).
228         case 12:
229           contrOrig.transform.position = Vector3(30, 0.55, -41.3);
230           contrOrig.transform.rotation.eulerAngles = Vector3(0, 90, 0);
231
232           break;
233
234         // Puerta de acceso al aula 02.
235         case 15:
236           contrOrig.transform.position = Vector3(46.9, 0.55, -31.5);
237           contrOrig.transform.rotation.eulerAngles = Vector3(0, 270, 0);
238
239           break;
240
241         // Puerta de acceso al aula 01.
242         case 16:
243           contrOrig.transform.position = Vector3(46.9, 0.55, -35);
```

```
244           contrOrig.transform.rotation.eulerAngles = Vector3(0, 270, 0);
245
246           break;
247
248         // Puerta de acceso al aula 03.
249         case 17:
250           contrOrig.transform.position = Vector3(34.6, 0.55, -19);
251           contrOrig.transform.rotation.eulerAngles = Vector3(0, 180, 0);
252
253           break;
254
255         default:
256           contrOrig.transform.position = Vector3(23.02298, 0.55, -44.69259);
257           contrOrig.transform.rotation.eulerAngles = Vector3(0, 0, 0);
258       }
259
260       break;
261
262
263     case "02b EscuelaP1":
264
265       switch (CambioPuerta.Puerta)
266       {
267         // Escaleras sur entre la planta baja y la uno de la escuela.
268         case 7:
269           contrOrig.transform.position = Vector3(29, 2.78, -37.7);
270           contrOrig.transform.rotation.eulerAngles = Vector3(0, 135, 0);
271
272           break;
273
274         // Escaleras sur entre la planta baja y la uno de la escuela.
275         case 8:
276           contrOrig.transform.position = Vector3(36.5, 2.78, -28.3);
277           contrOrig.transform.rotation.eulerAngles = Vector3(0, 315, 0);
278
279           break;
280
281         // Puerta de acceso a la biblioteca.
282         case 13:
283           contrOrig.transform.position = Vector3(47.9, 5.04, -48);
284           contrOrig.transform.rotation.eulerAngles = Vector3(0, 315, 0);
285
286           break;
287
288         // Puerta de acceso al aula 204.
289         case 14:
290           contrOrig.transform.position = Vector3(46.7, 5.04, -19.4);
291           contrOrig.transform.rotation.eulerAngles = Vector3(0, 225, 0);
292
293           break;
294
295         // Puerta de acceso al aula 216.
296         case 23:
297           contrOrig.transform.position = Vector3(30.5, 5.04, -49.7);
298           contrOrig.transform.rotation.eulerAngles = Vector3(0, 0, 0);
299
300           break;
301
302         default:
```

```
303             contrOrig.transform.position = Vector3(45.22293, 5.041617, -45.32293);
304             contrOrig.transform.rotation.eulerAngles = Vector3(0, 0, 0);
305         }
306 
307         break;
308 
309 
310     case "03a EdifTecnologicoP1":
311 
312         switch (CambioPuerta.Puerta)
313         {
314             // Acceso sudeste del edificio tecnológico desde la escuela.
315             case 5:
316                 contrOrig.transform.position = Vector3(-35.3, 0.625, -1.9);
317                 contrOrig.transform.rotation.eulerAngles = Vector3(0, 180, 0);
318 
319                 break;
320 
321             // Acesso noreste del edificio tecnológico desde la escuela.
322             case 6:
323                 contrOrig.transform.position = Vector3(-38.8, 0.625, -1.9);
324                 contrOrig.transform.rotation.eulerAngles = Vector3(0, 180, 0);
325 
326                 break;
327 
328             // Tramo más próximo a la entrada del rellano de las escaleras este
329             // de acceso a la planta dos.
330             case 15:
331                 contrOrig.transform.position = Vector3(-36.75, 7.96, -13.5);
332                 contrOrig.transform.rotation.eulerAngles = Vector3(0, 90, 0);
333 
334                 break;
335 
336             // Tramo más alejado a la entrada del rellano de las escaleras este
337             // de acceso a la planta dos.
338             case 16:
339                 contrOrig.transform.position = Vector3(-37.4, 7.96, -15.3);
340                 contrOrig.transform.rotation.eulerAngles = Vector3(0, 270, 0);
341 
342                 break;
343 
344             // Tramo más próximo a la entrada del rellano de las escaleras oeste
345             // de acceso a la planta dos.
346             case 17:
347                 contrOrig.transform.position = Vector3(-37.4, 7.96, -55.4);
348                 contrOrig.transform.rotation.eulerAngles = Vector3(0, 270, 0);
349 
350                 break;
351 
352             // Tramo más alejado a la entrada del rellano de las escaleras este
353             // de acceso a la planta dos.
354             case 18:
355                 contrOrig.transform.position = Vector3(-36.75, 7.96, -57.2);
356                 contrOrig.transform.rotation.eulerAngles = Vector3(0, 90, 0);
357 
358                 break;
359 
360             // Puerta de acceso al baño de chicas en la planta baja del edificio tecnológico.
361             case 19:
```

```
362           contrOrig.transform.position = Vector3(-45.5, 0.625, -25);
363           contrOrig.transform.rotation.eulerAngles = Vector3(0, 0, 0);
364
365           break;
366
367         // Puerta de acceso al aula H3.
368         case 21:
369           contrOrig.transform.position = Vector3(-4.6, 5.52, -61.5);
370           contrOrig.transform.rotation.eulerAngles = Vector3(0, 0, 0);
371
372           break;
373
374         default:
375           contrOrig.transform.position = Vector3(-37.40001, 0.609341, -42.11703);
376           contrOrig.transform.rotation.eulerAngles = Vector3(0, 0, 0);
377       }
378
379       break;
380
381
382     case "03b EdifTecnologicoP2":
383
384       switch (CambioPuerta.Puerta)
385       {
386         // Tramo más próximo a la entrada del rellano de las escaleras este
387         // de acceso a la planta uno.
388         case 15:
389           contrOrig.transform.position = Vector3(-37.4, 7.96, -13.5);
390           contrOrig.transform.rotation.eulerAngles = Vector3(0, 270, 0);
391
392           break;
393
394         // Tramo más alejado a la entrada del rellano de las escaleras este
395         // de acceso a la planta uno.
396         case 16:
397           contrOrig.transform.position = Vector3(-36.75, 7.96, -15.3);
398           contrOrig.transform.rotation.eulerAngles = Vector3(0, 90, 0);
399
400           break;
401
402         // Tramo más próximo a la entrada del rellano de las escaleras oeste
403         // de acceso a la planta uno.
404         case 17:
405           contrOrig.transform.position = Vector3(-36.75, 7.96, -55.4);
406           contrOrig.transform.rotation.eulerAngles = Vector3(0, 90, 0);
407
408           break;
409
410         // Tramo más alejado a la entrada del rellano de las escaleras oeste
411         // de acceso a la planta uno.
412         case 18:
413           contrOrig.transform.position = Vector3(-37.4, 7.96, -57.2);
414           contrOrig.transform.rotation.eulerAngles = Vector3(0, 270, 0);
415
416           break;
417
418         default:
419           contrOrig.transform.position = Vector3(-37.52436, 8.061043, -13.62617);
420           contrOrig.transform.rotation.eulerAngles = Vector3(0, 0, 0);
```

```
421       }
422
423       break;
424
425
426     case "05a BibliotecaAbajo":
427
428       switch (CambioPuerta.Puerta)
429       {
430         // Puerta de acceso a la planta uno de la escuela.
431         case 13:
432           contrOrig.transform.position = Vector3(48.8, 5.03, -49.3);
433           contrOrig.transform.rotation.eulerAngles = Vector3(0, 135, 0);
434
435           break;
436
437         // Segundo escalón del tramo de escaleras contiguo al segundo rellano
438         // hacia el primer piso.
439         case 19:
440           contrOrig.transform.position = Vector3(62.1, 7.17, -63.5);
441           contrOrig.transform.rotation.eulerAngles = Vector3(0, 270, 0);
442
443           break;
444
445         default:
446           contrOrig.transform.position = Vector3(62.1, 5.090043, -65.52473);
447           contrOrig.transform.rotation.eulerAngles = Vector3(0, 0, 0);
448
449       }
450
451       break;
452
453
454     case "09 Aula H3":
455
456       switch (CambioPuerta.Puerta)
457       {
458         // Puerta de acceso a la planta uno del edificio tecnológico.
459         case 21:
460           contrOrig.transform.position = Vector3(-4.5, 5.510001, -64);
461           contrOrig.transform.rotation.eulerAngles = Vector3(0, 180, 0);
462
463           break;
464
465         default:
466           contrOrig.transform.position = Vector3(-4.66824, 5.633972, -64.10133);
467           contrOrig.transform.rotation.eulerAngles = Vector3(0, 180, 0);
468
469       }
470
471       break;
472
473
474     case "11 Aula 216":
475
476       switch (CambioPuerta.Puerta)
477       {
478         // Puerta de acceso a la planta uno de la escuela.
479         case 24:
```

```
480           contrOrig.transform.position = Vector3(30.5, 5.04, -50.9);
481           contrOrig.transform.rotation.eulerAngles = Vector3(0, 180, 0);
482
483           break;
484
485         default:
486           contrOrig.transform.position = Vector3(30.77329, 5.088405, -50.88791);
487           contrOrig.transform.rotation.eulerAngles = Vector3(0, 180, 0);
488
489       }
490
491       break;
492
493
494     case "04 SalondeActos":
495
496       switch (CambioPuerta.Puerta)
497       {
498         // Puertas de acceso a la planta baja de la escuela.
499         case 25:
500           contrOrig.transform.position = Vector3(48.3, 0.55, -49.35);
501           contrOrig.transform.rotation.eulerAngles = Vector3(0, 135, 0);
502
503           break;
504
505         default:
506           contrOrig.transform.position = Vector3(48.33479, 0.5821109, -49.35332);
507           contrOrig.transform.rotation.eulerAngles = Vector3(0, 135, 0);
508
509       }
510
511       break;
512
513     /*
514     // Nota: Para aquellas escenas con una sólo acceso de entrada (ejemplo, aulas 01 o
          aula 214)
515     // el objeto "ControlProj" no se reubica y el controlador Mixano se clona en la
          posición que
516     // posee en la escena. Por esta razón no hay un caso "default" para el bloque "switch
          ",
517     // recogería precisamente el tipo de escenas mencionadas y su acción ya está definida
          .
518
519     default:
520
521       contrOrig.transform.position = ;
522       contrOrig.transform.rotation.eulerAngles = ;
523     */
524
525   }
526
527
528   // --------
529   // FASE DOS
530   // --------
531
532   // Tras la reubicación del controlador predeterminando, lo destruimos y clonamos en su
          misma
533   // posición el controlador Mixano.
```

```
534
535 // Nota: El sistema del transporte del SDN (proyecto H) no necesita reubicar el
        controlador
536 // predeterminado antes de destruirlo, es independiente de éste y crea directamente el
537 // controlador Mixano en la posición deseada, por tanto la fase uno sólamente es
        necesaria
538 // para el sistema legado.
539
540
541 // Sistema de transporte del SDN (Proyecto H)
542 if (TransporteSDN.activado == true)
543 {
544   switch (CambioControl.cambioContr)
545   {
546     case 1:
547       controladorClon = Instantiate(controlador1, TransporteSDN.posicion, TransporteSDN
              .rotacion);
548       Destroy(contrOrig);
549       break;
550     case 2:
551       controladorClon = Instantiate(controlador2, TransporteSDN.posicion, TransporteSDN
              .rotacion);
552       Destroy(contrOrig);
553       break;
554     case 3:
555       controladorClon = Instantiate(controlador3, TransporteSDN.posicion, TransporteSDN
              .rotacion);
556       Destroy(contrOrig);
557       break;
558     case 4:
559       controladorClon = Instantiate(controlador4, TransporteSDN.posicion, TransporteSDN
              .rotacion);
560       Destroy(contrOrig);
561       break;
562     case 5:
563       controladorClon = Instantiate(controlador5, TransporteSDN.posicion, TransporteSDN
              .rotacion);
564       Destroy(contrOrig);
565       break;
566     case 6:
567       controladorClon = Instantiate(controlador6, TransporteSDN.posicion, TransporteSDN
              .rotacion);
568       Destroy(contrOrig);
569       break;
570     default:
571       controladorClon = Instantiate(controlador1, TransporteSDN.posicion, TransporteSDN
              .rotacion);
572       Destroy(contrOrig);
573   }
574
575   TransporteSDN.activado = false;
576
577 }
578 // Sistema de transporte antiguo
579
580 // La clonación de los controladores en la versión anterior de este guión se realizaba
        mediante:
581 // Instantiate (controlador1,contrOrig.transform.position,contrOrig.transform.rotation)
        ;
```

```
582
583   // En esta versión se ha modificando esta instrucción para que quede de la siguiente
          forma:
584   // controladorClon = Instantiate(controlador1, contrOrig.transform.position, contrOrig.
          transform.rotation);
585   // de forma que ahora sea posible referenciar posteriormente a los controladores
          clonados.
586   else
587   {
588     if (CambioControl.cambioContr == 1)
589     {
590       controladorClon = Instantiate(controlador1, contrOrig.transform.position, contrOrig
              .transform.rotation);
591       Destroy(contrOrig, 0);
592     }
593
594     if (CambioControl.cambioContr == 2)
595     {
596       controladorClon = Instantiate(controlador2, contrOrig.transform.position, contrOrig
              .transform.rotation);
597       Destroy(contrOrig, 0);
598     }
599
600     if (CambioControl.cambioContr == 3)
601     {
602       controladorClon = Instantiate(controlador3, contrOrig.transform.position, contrOrig
              .transform.rotation);
603       Destroy(contrOrig, 0);
604     }
605
606     if (CambioControl.cambioContr == 4)
607     {
608       controladorClon = Instantiate(controlador4, contrOrig.transform.position, contrOrig
              .transform.rotation);
609       Destroy(contrOrig, 0);
610     }
611
612     if (CambioControl.cambioContr == 5)
613     {
614       controladorClon = Instantiate(controlador5, contrOrig.transform.position, contrOrig
              .transform.rotation);
615       Destroy(contrOrig, 0);
616     }
617
618     if (CambioControl.cambioContr == 6)
619     {
620       controladorClon = Instantiate(controlador6, contrOrig.transform.position, contrOrig
              .transform.rotation);
621       Destroy(contrOrig, 0);
622     }
623   }
624
625
626   // Asignamos una referencia global de los elementos principales, el nuevo avatar y la
          nueva cámara principal
627   // para accesos posteriores.
628
629   SDN.avatar = controladorClon;
630
```

```
631 // Para la asignación de la cámara principal recorremos los objetos hijo del
        controlador nuevo. Uno de ellos
632 // es la cámara principal, etiquetada con "MainCamera", que usaremos para la referencia
        global.
633
634 // (Nótese que la siguiente instrucción no sirve:
635 // SDN.camaraPrincipal = GameObject.FindWithTag("MainCamera");
636 // ... pues esa instrucción localiza la cámara principal anterior a la sustitución del
        controlador al tener
637 // ambas la misma etiqueta. Para asegurarnos de capturar la cámara correcta usamos el
        siguiete código)
638
639 /*
640   for (var objetoHijo : Transform in controladorClon.transform)
641 {
642   if (objetoHijo.CompareTag("MainCamera"))
643   {
644     SDN.camaraPrincipal = objetoHijo.gameObject;
645   }
646 }
647
648 // Nota: el código anterior genera en el compilador la siguiente advertencia:
649 // "WARNING: Implicit downcast from 'Object' to 'UnityEngine.Transform'"
650 // Ésta puede ignorarse por completo pues no supone error y la apunta el compilador
651 // cuando se utiliza la estructura "for (EQUIS in transform)". No obstante, se puede
652 // evitar el mensaje complicando el código (se opta por esta última opción para que
653 // posibles futuros desarrolladores, aun si el mensaje es inocuo, no tengan que
654 // preguntarse su origen):
655 */
656
657 for (objetoHijo in controladorClon.transform)
658 {
659   var objetoHijoAsignado : Transform = objetoHijo as Transform;
660
661   if (objetoHijoAsignado.CompareTag("MainCamera"))
662   {
663     SDN.camaraPrincipal = objetoHijoAsignado.gameObject;
664   }
665 }
666
667
668 // Clonamos en la escena la cámara del mapa, elemento central del sistema de navegación
        (Proyecto H).
669
670 if (camaraMapa != null)
671 {
672   camaraMapaClon = Instantiate(camaraMapa, transform.position, transform.rotation);
673
674   // Asignamos una referencia global de la cámara del mapa.
675
676   SDN.camaraMapa = camaraMapaClon;
677 }
678 else
679 {
680   Debug.LogWarning("No se ha asignado el <i>prefab</i> \"<b>H_CamaraMapa</b>\" o
        equivalente a la variable \"Camara Mapa\" del script \"Sust Contr\" dentro del
        objeto \"CambioControlador\" en la escena. El minimapa no se mostrará hasta que
        se asigne un <i>prefab</i> adecuado.");
681 }
```

```
682
683 }
684
685
686
687 /*
688 // ----------------------------
689 // GUIÓN ORIGINAL SIN MODIFICAR
690 // ----------------------------
691
692
693 var controlador1 : GameObject ;
694 var controlador2 : GameObject ;
695 var controlador3 : GameObject ;
696 var controlador4 : GameObject ;
697 var controlador5 : GameObject ;
698 var controlador6 : GameObject ;
699 var contrOrig : GameObject ;
700
701
702
703 function Start() {
704
705
706
707
708
709
710
711
712   if (CambioControl.cambioContr == 1) {
713
714     Instantiate (controlador1,contrOrig.transform.position,contrOrig.transform.rotation);
715     Destroy (contrOrig,0);
716     }
717
718   if (CambioControl.cambioContr == 2) {
719
720     Instantiate (controlador2,contrOrig.transform.position,contrOrig.transform.rotation);
721     Destroy (contrOrig,0);
722     }
723   if (CambioControl.cambioContr == 3) {
724
725     Instantiate (controlador3,contrOrig.transform.position,contrOrig.transform.rotation);
726     Destroy (contrOrig,0);
727     }
728   if (CambioControl.cambioContr == 4) {
729
730     Instantiate (controlador4,contrOrig.transform.position,contrOrig.transform.rotation);
731     Destroy (contrOrig,0);
732     }
733   if (CambioControl.cambioContr == 5) {
734
735     Instantiate (controlador5,contrOrig.transform.position,contrOrig.transform.rotation);
736     Destroy (contrOrig,0);
737     }
738   if (CambioControl.cambioContr == 6) {
739
740     Instantiate (controlador6,contrOrig.transform.position,contrOrig.transform.rotation);
```

```
741     Destroy (contrOrig,0);
742     }
743
744
745 }
746
747 */
```

F.55. StartVideo 1.js

Guión del proyecto legado modificado por el presente.

```
1  #pragma strict
2
3  /*#####################################################################
4
5  StartVideo 1.js
6  ---------------
7
8  Guión del proyecto legado modificado y mejorado para el Proyecto H.
9
10 Guión encargado de gestionar la presentación de los distintos vídeos habilitados en
11 el proyecto.
12
13 Se ha modificado para adaptarse a la directiva "#pragam strict" y las recomendaciones
14 indicadas en "http://docs.unity3d.com/Documentation/Components/class-MovieTexture.html":
15
16   "If you have #pragma strict enabled in your code a MovieTexture object
17   should be declared somewhere and the object should be initialized with
18   renderer.material.mainTexture. Then isPlaying, Play() and Stop() should
19   be called for this MovieTexture object."
20
21 Se incluye al final la versión original del guión como referencia.
22
23 #####################################################################*/
24
25
26 var presentacion : MovieTexture;
27
28 function Start ()
29 {
30   renderer.material.mainTexture = presentacion;
31 }
32
33 function Update ()
34 {
35   if (Input.GetButtonDown ("Jump"))
36   {
37     if (presentacion.isPlaying)
38     {
39       presentacion.Pause();
40     }
41     else
```

```
42      {
43        presentacion.Play();
44      }
45    }
46  }
47
48
49  /*
50  // ----------------------------
51  // GUIÓN ORIGINAL SIN MODIFICAR
52  // ----------------------------
53
54
55  function Update () {
56    if (Input.GetButtonDown ("Jump")) {
57      if (renderer.material.mainTexture.isPlaying) {
58        renderer.material.mainTexture.Pause();
59      }
60      else {
61        renderer.material.mainTexture.Play();
62      }
63    }
64  }
65
66  */
```

F.56. AbrirPuerta<...>.js

Guiones del proyecto legado modificados por el presente.

Esta sección sirve de modelo para todos los guiones modificados que comparten el prefijo «AbrirPuerta».

Los cambios en todos ellos han sido los mismos, sirva así, este, de ejemplo ilustrativo.

```
1  #pragma strict
2
3  /*#################################################################
4
5  AbrirPuerta<...>.js
6  -------------------
7
8  Guión del proyecto legado modificado y mejorado para el Proyecto H.
9
10 Guión encargado de gestionar la apertura y cierre de la puerta que
11 abarca el área activadora a la que está asociado.
12
13 Se ha añadido a los bloques "OnTriggerEnter()" y "OnTriggerExit()"
14 una comparación por etiqueta que permite acotar su funcionamiento
```

```
exclusivamente a la interacción del avatar con el área activadora.
Cualquier otro objeto diferente al avatar (identificado éste con la
etiqueta "Player") no desencadenará por tanto la acción del guión.
Se resuelven así todos aquellos escenarios en los que puediera
activarse la puerta repetidas veces para una misma sola acción de
entrada o salida.

##################################################################*/


var AngleY : float = 90.0;

private var targetValue : float = 90;
private var currentValue : float = 90;
private var easing : float = 0.05;

var puertaAbrir: AudioClip;
var puertaAbrir2: AudioClip;
var Target : GameObject;

function Update(){
  currentValue = currentValue + (targetValue - currentValue) * easing;
  Target.transform.rotation = Quaternion.identity; // set rotation to zero
  Target.transform.Rotate(0, currentValue, 0); // apply full Rotation
}

function OnTriggerEnter (other : Collider) {
  if (other.tag == "Player")
  {
    targetValue = AngleY;
    currentValue = 90;
    AudioSource.PlayClipAtPoint(puertaAbrir, transform.position);
  }
}

function OnTriggerExit (other : Collider) {
  if (other.tag == "Player")
  {
    currentValue = AngleY;
    targetValue = 90;
    AudioSource.PlayClipAtPoint(puertaAbrir2, transform.position);
  }
}
```

F.57. AbrirPuertaD277.js

Guión del proyecto legado modificado por el presente.

```
#pragma strict

/*##################################################################

AbrirPuertaD277.js
```

```
6  ------------------
7
8  Guión del proyecto legado modificado y mejorado para el Proyecto H.
9
10 Guión encargado de gestionar la apertura y cierre de la puerta de
11 acceso al despacho de Fernando Jorge Fraile Fernádez en la planta
12 segunda del edificio tecnológico.
13
14 Se incluye al final la versión original del guión como referencia.
15
16 ##################################################################*/
17
18
19 var AngleY : float = 90.0;
20
21 private var targetValue : float = 90;
22 private var currentValue : float = 90;
23 private var easing : float = 0.13;
24
25 // Variables de control para gestionar correctamente la
26 // apertura y cierre de la puerta cuando el avatar se
27 // transporta directamente dentro del área activadora
28 // "OnTriggerEnter".
29
30 private var avatarDentro : boolean = true;
31 private var avatarFuera : boolean = true;
32
33
34 var puertaAbrir: AudioClip;
35 var puertaAbrir2: AudioClip;
36 var Target : GameObject;
37
38
39 function Update()
40 {
41   currentValue = currentValue + (targetValue - currentValue) * easing;
42   // Ajuste inicial de la rotación a cero.
43   Target.transform.rotation = Quaternion.identity;
44   // Aplicación de la rotación calculada.
45   Target.transform.Rotate(0, currentValue, 0);
46 }
47
48 function OnTriggerEnter (other : Collider)
49 {
50   // Si el avatar se transporta directamente dentro del cubo activador
51   // la función "OnTriggerEnter()" se activa dos veces. Una al
52   // detectar que el avatar se ha transportado dentro, y dos,
53   // al comenzar a moverse el avatar.
54
55   // Este comportamiento se explica de la siguiente forma: el controlador
56   // Mixano no dispone de "RigidBody" y por ello la función "OnTrigger" lo
57   // detecta únicamente al moverse. Por otro lado el elemento "H_PlayerPunto"
58   // que el Proyecto H le asocia al controlador, precisamente para resolver
59   // eventos "OnTrigger" con precisión sí disponde de "RigidBody" y su presencia
60   // es detectada nada más transportarse y sin necesidad de moverse.
61
62   // Para resolver este funcionamiento sin modificar el proyecto legado
63   // establecemos una variable "avatarDentro" y un código de control que
64   // conseguirán que en cuanto "OnTriggerEnter" se haya ejecutado una vez, no
```

```
65    // vuelva a ejecutarse hasta que el avatar haya salido del cubo
66    // y vuelto a entrar en él.
67
68    // La variable "avatarFuera" funciona de forma equivalente a "avatarDentro",
69    // gestionando el cierre de la puerta en lugar de la apertura.
70
71    // Nota: acotando la función "OnTrigger" exclusivamente al avatar por mediante la
72    // etiqueta "Player" la puerta sólo se abre al moverse éste, y de acotarse al
73    // elemento "H_PlayerPunto" se resolvería el problema pero se alteraría el proyecto
74    // legado al convertir a este evento "OnTrigger" en dependiente de un elemento del
75    // Proyecto H.
76
77    if (avatarDentro == true)
78    {
79      targetValue = AngleY;
80      currentValue = 90;
81      AudioSource.PlayClipAtPoint(puertaAbrir, transform.position);
82
83      avatarDentro = false;
84      avatarFuera = true;
85    }
86  }
87
88  function OnTriggerExit (other : Collider)
89  {
90    if (avatarFuera == true)
91    {
92      currentValue = AngleY;
93      targetValue = 90;
94      AudioSource.PlayClipAtPoint(puertaAbrir2, transform.position);
95
96      avatarDentro = true;
97      avatarFuera = false;
98    }
99  }
100
101
102
103 /*
104 // ----------------------------
105 // GUIÓN ORIGINAL SIN MODIFICAR
106 // ----------------------------
107
108
109
110 var AngleY : float = 90.0;
111
112 private var targetValue : float = 90;
113 private var currentValue : float = 90;
114 private var easing : float = 0.05;
115
116 var puertaAbrir: AudioClip;
117 var puertaAbrir2: AudioClip;
118 var Target : GameObject;
119
120 function Update(){
121   currentValue = currentValue + (targetValue - currentValue) * easing;
122   Target.transform.rotation = Quaternion.identity; // set rotation to zero
123   Target.transform.Rotate(0, currentValue, 0); // apply full Rotation
```

```
124 }
125
126 function OnTriggerEnter (other : Collider) {
127   targetValue = AngleY;
128   currentValue = 90;
129   AudioSource.PlayClipAtPoint(puertaAbrir, transform.position);
130   }
131
132 function OnTriggerExit (other : Collider) {
133   currentValue = AngleY;
134   targetValue = 90;
135   AudioSource.PlayClipAtPoint(puertaAbrir2, transform.position);
136 }
137
138 */
```