Manual De Instrucciones

Manuel Jesús Corbacho Sánchez

26 de mayo de 2020

1. Manual de Instrucciones

Debido a que el desarrollo se ha apoyado en el motor Unity3D, se procede a explicar como puede integrarse el sistema desarrollado en el motor, y entre otras posibilidades, repetir los estudios de características y tiempos de ejecución llevados a cabo. Para ello seguiremos los siguientes pasos:

1. En primer lugar se descarga el proyecto del siguiente repositorio de **GitHub**: `https://github.com/mjcs-95/MazeGenerator`. Para ello hacermos click en *Clone or download* y luego a *Download ZIP*

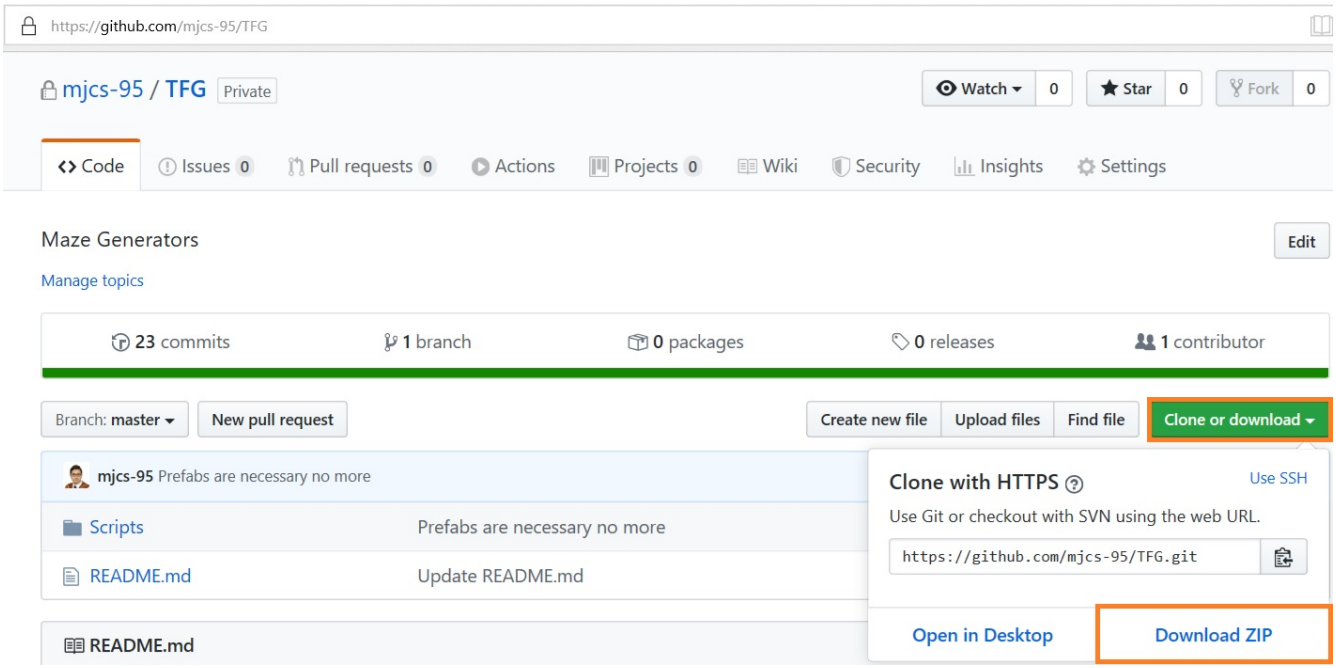
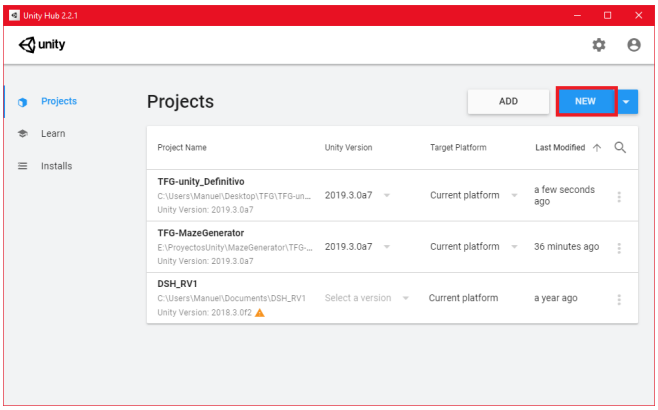


Figura 1: Repositorio de GitHub del proyecto.

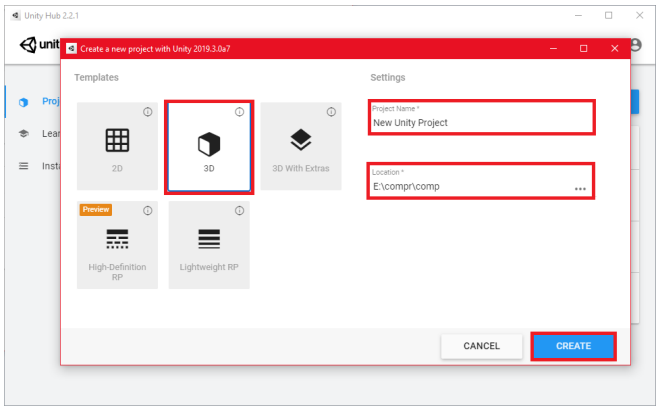
2. Creamos un nuevo proyecto desde UnityHub, Hacemos click sobre **NEW** 2a y seleccionamos un proyecto 3D. A continuación seleccionamos el nombre del proyecto y la ubicación del directorio del proyecto y por último hacemos click en **CREATE** 2b. La alternativa sería tener un proyecto ya creado, en ese caso podemos saltarnos este paso.

3. Extraemos la carpeta **Scripts** al directorio /**Assets** de nuestro proyecto en Unity3D. Para acceder al directorio /**Assets**, hacemos click derecho sobre el nombre de la carpeta en el editor de Unity3D y hacemos click en **Show in Explorer**. La carpeta /**Scripts** que hemos bajado contiene las clases del generador, una clase **MazeGenerator** que nos servirá de enlace para **Unity3D**, y una clase auxiliar **MazeGeneratorEditor** que es usada por la anterior para añadir la posibilidad de generar laberintos desde el editor, sin necesidad de entrar al modo de juego del motor. Además debido al diseño de la clase, crearemos una carpeta llamada *Resources* en el directorio *Assets* de nuestro proyecto, para ello hacemos click derechp sobre la carpeta *Assets* y pulsamos sobre la opción *Create Folder* 4.

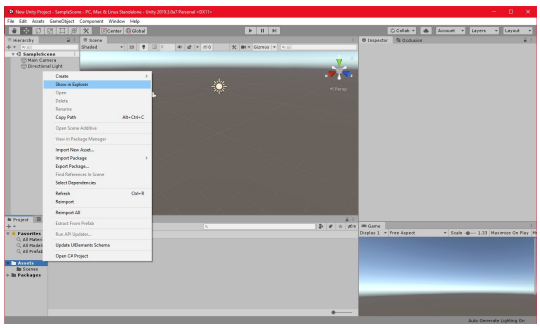
4. A continuación en el editor de Unity3D hacemos click sobre **GameObject** en la barra de herramientas superior, y en el menú desplegable hacemos click en **Create Empty** 5.



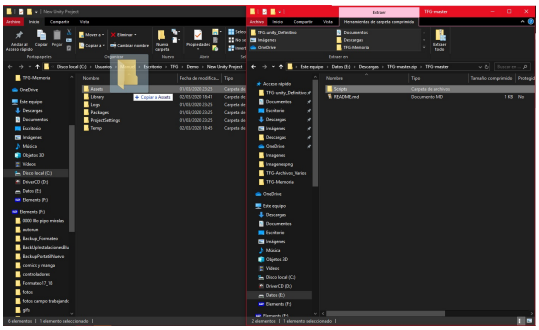
(a) Hacemos click en **NEW**.



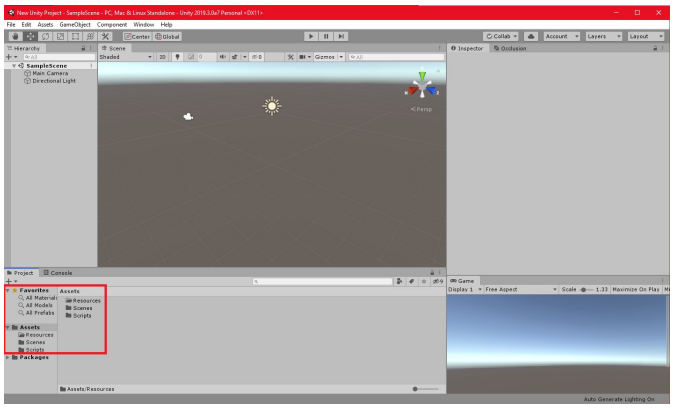
(b) Seleccionamos un proyecto 3D.



(a) Como abrir el directorio *Assets*



(b) Movemos la carpeta descargada del repositorio al directorio abierto



(c) Nuestro proyecto tras crear la carpeta *Resources*

Figura 3: Como introducir el código del repositorio a **Unity3D** en Windows 10.

Figura 4: Como crear un nuevo proyecto en Unity3D.

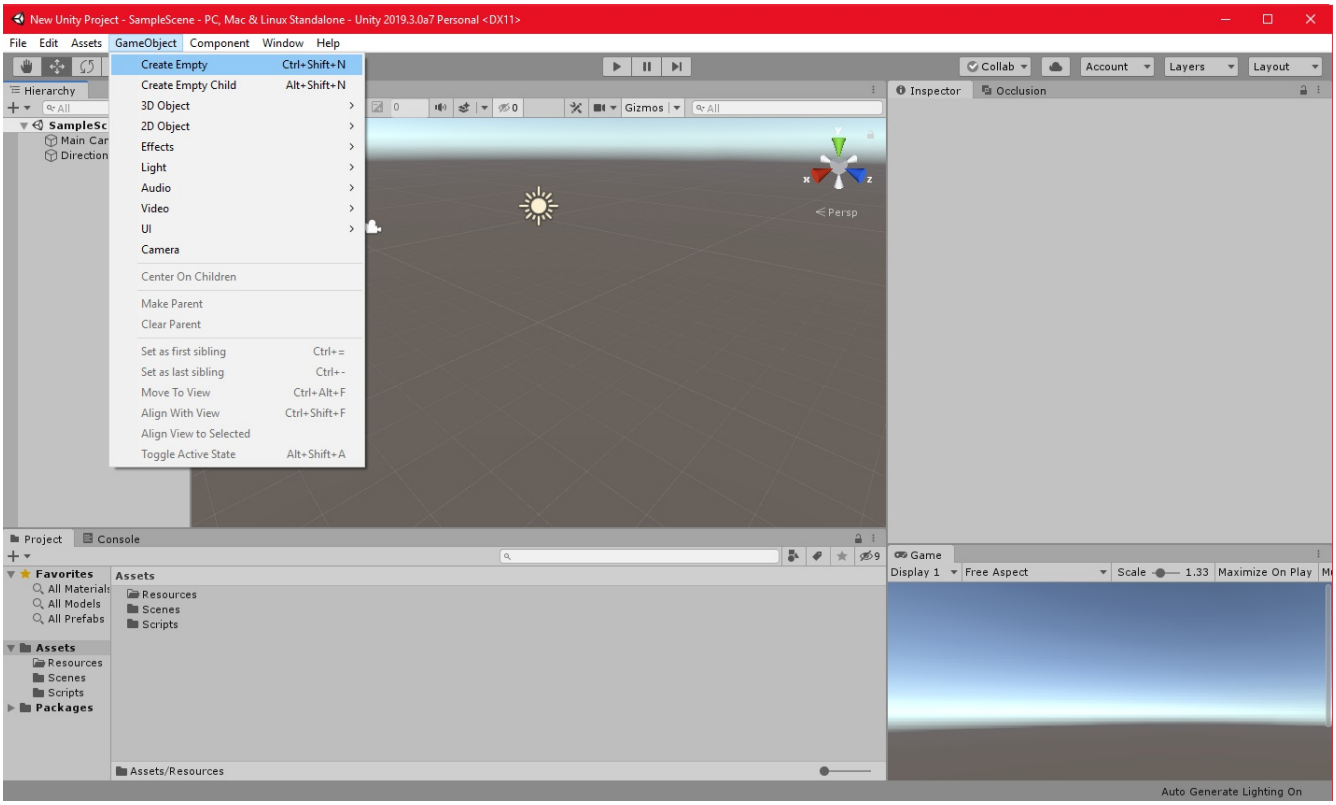


Figura 5: Creamos un GameObject.

5. Para el próximo pase en el navegador de Unity3D seleccionamos la carpeta **Scripts** de nuestro proyecto y arrastramos el archivo **MazeGenerator.cs** y sobre el *GameObject* que acabamos de crear 6.

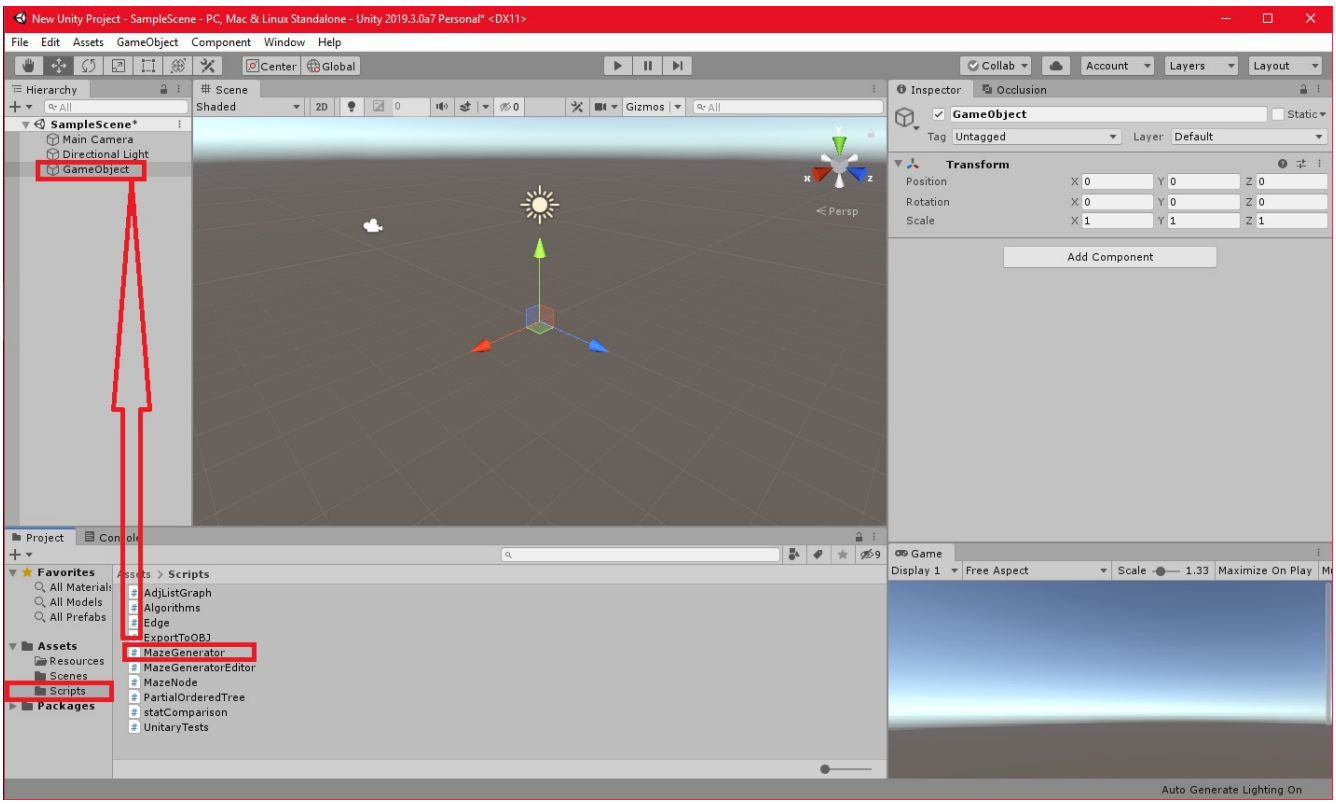


Figura 6: Cómo asignar un script a un GameObject.

6. Ahora que nos han aparecido los parámetros de configuración, hay que darles unos valores válidos. Para ello, primero establecemos las variables **Rows** y **Cols** a un número mayor que 0 7.

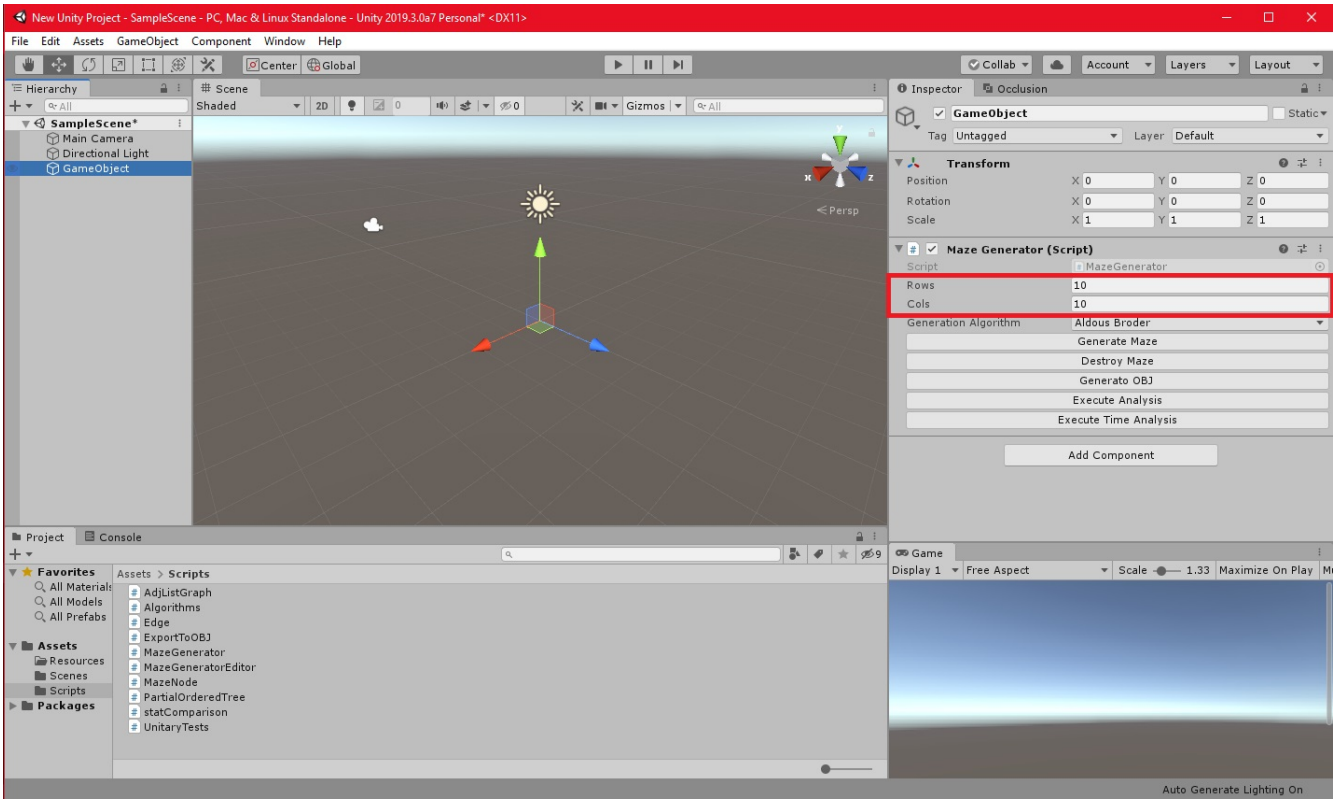


Figura 7: El generador actualmente sólo funcionará correctamente si los números son mayores que cero.

7. Ahora tenemos que seleccionar el algoritmo que deseemos de la lista desplegable de la variable **Generation Algorithm** 8.

8. Ahora podemos seleccionar pulsando sobre 1 de los 5 botones, qué acción queremos llevar a cabo.

   a) **Generate Maze** ejecuta el algoritmo seleccionado con los parametros que hemos configurado, y a continuación genera mediante los prefabs asignados las celdas del laberinto 9.

   b) **Destroy Maze** elimina el modelo del laberinto de Unity generado en Unity a partir del objeto al que hemos asignado el script **MazeGenerator**.Funciona eliminando los hijos del GameObject al que hemos asignado el script 10.

   c) **Execute Analysis** ejecutará varias veces cada algoritmo con unos parámetros preestablecidos, calculará las características de los laberintos generados, y almacenará los resultados en directorio /**Assets**, en un archivo con el nombre **analisis.csv** 11

   d) **Execute Time Analysis** ejecutará varias veces cada algoritmo, generando laberintos de distintos tamaños , midiendo el tiempo que tarda en ejecutarse cada algoritmo, y almacenará los resultados en directorio /**Assets**, en un archivo con el nombre **analisis.csv** . 11

   e) **Generate OBJ** exportará el laberinto actual a un archivo **.obj** en el directorio /**Assets**/**Resources** de nuestro proyecto. 11

   En 12 tenemos un ejemplo de cómo importar un **.obj** a blender.

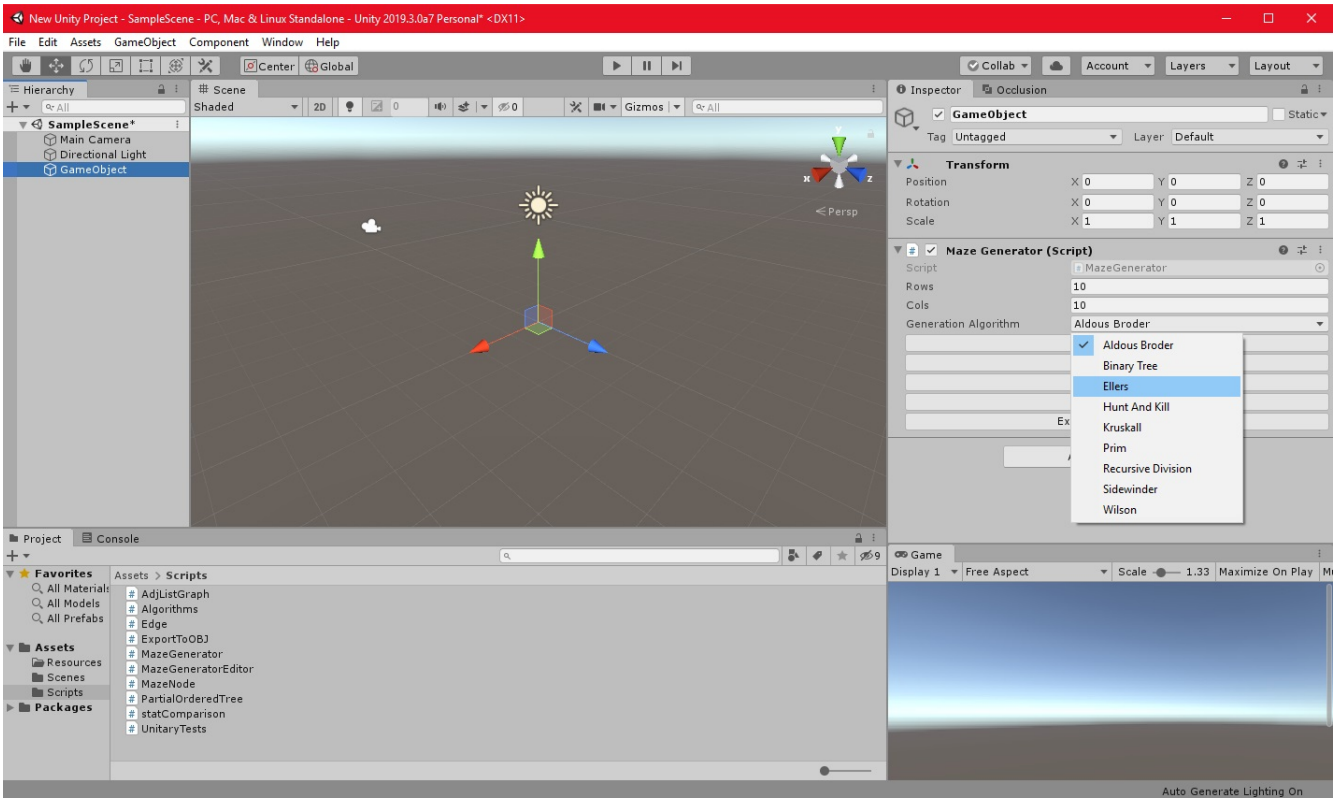


Figura 8: Lista desplegable actual, indicando los algoritmos disponibles.

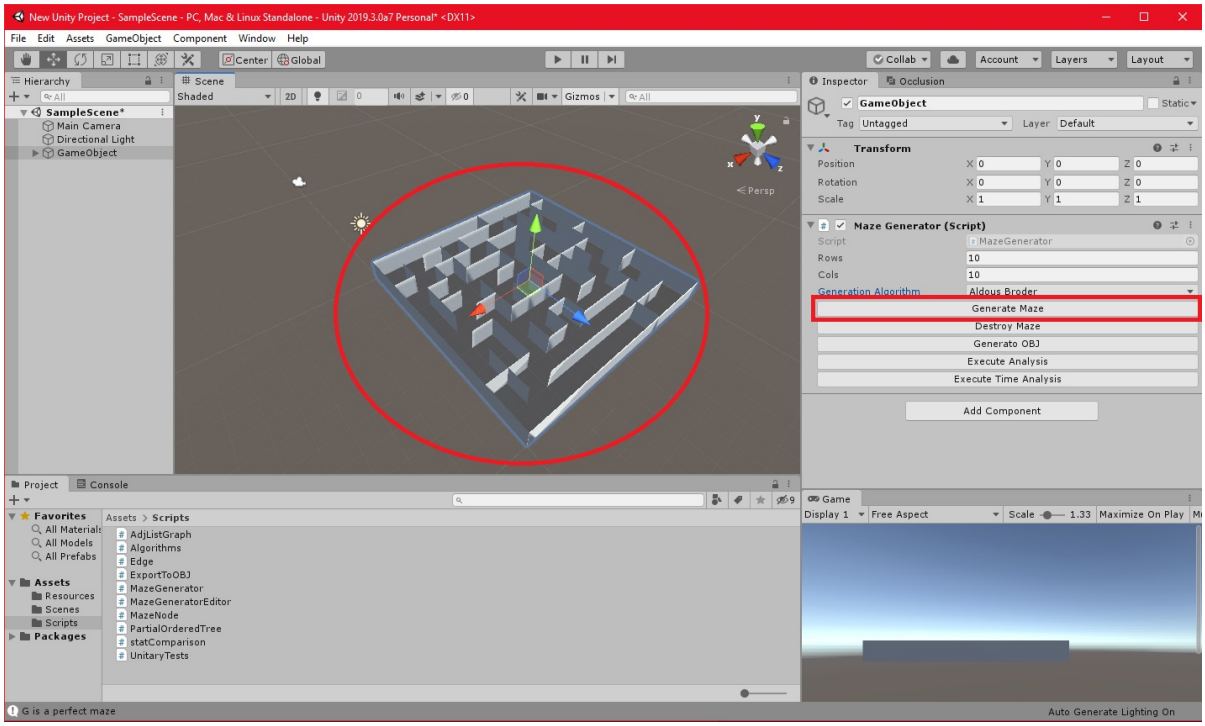


Figura 9: Haciendo click sobre el botón indicado, se genera un laberinto

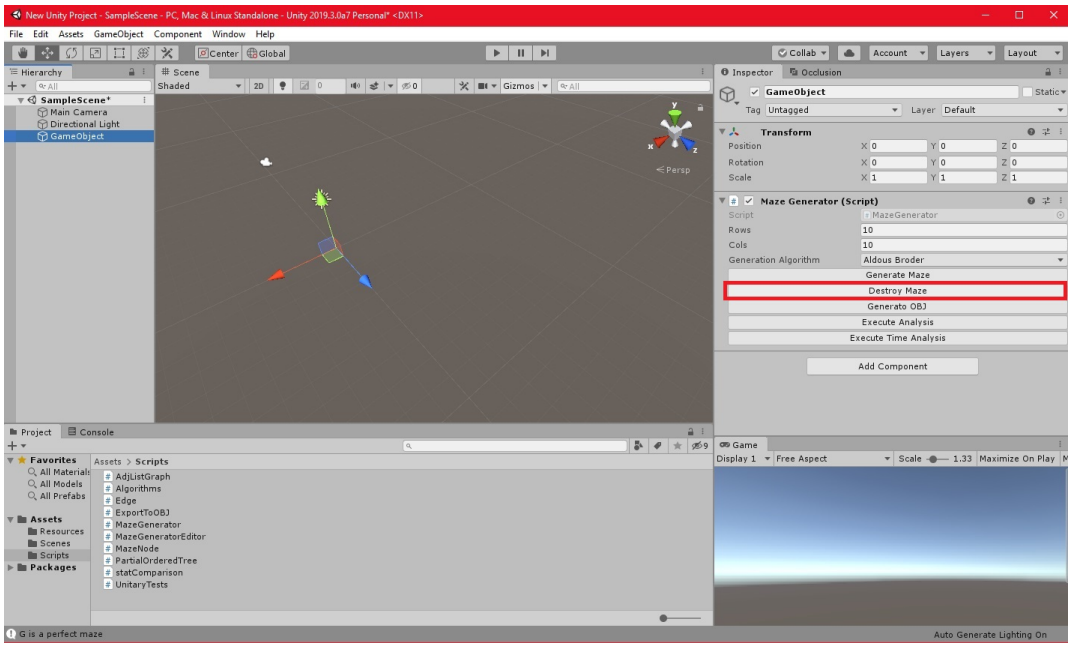


Figura 10: El laberinto se elimina correctamente.

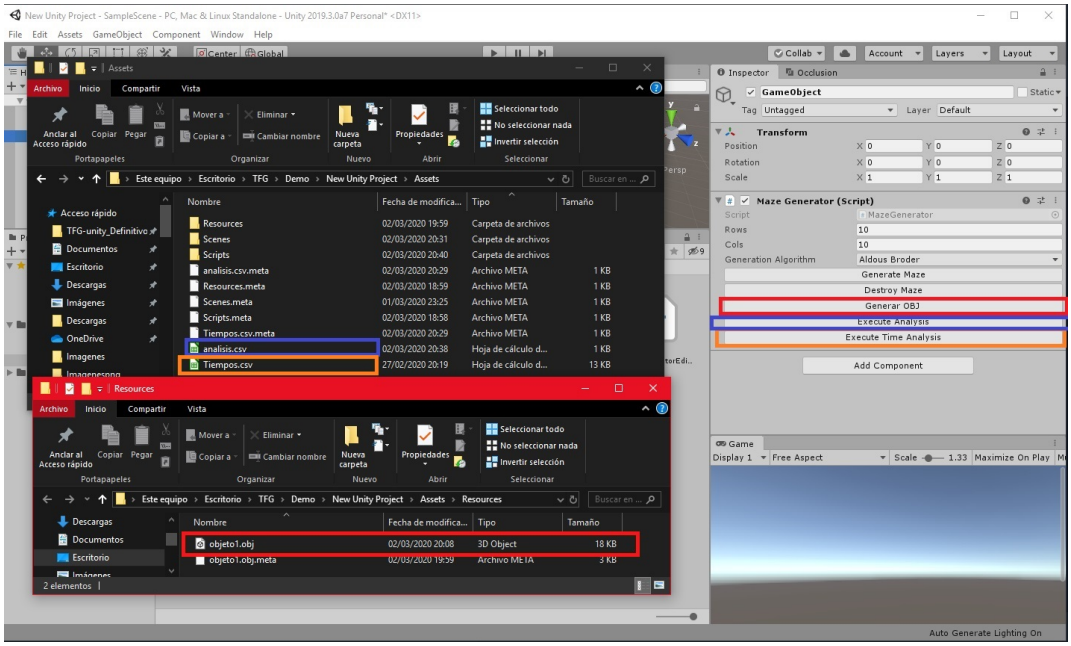
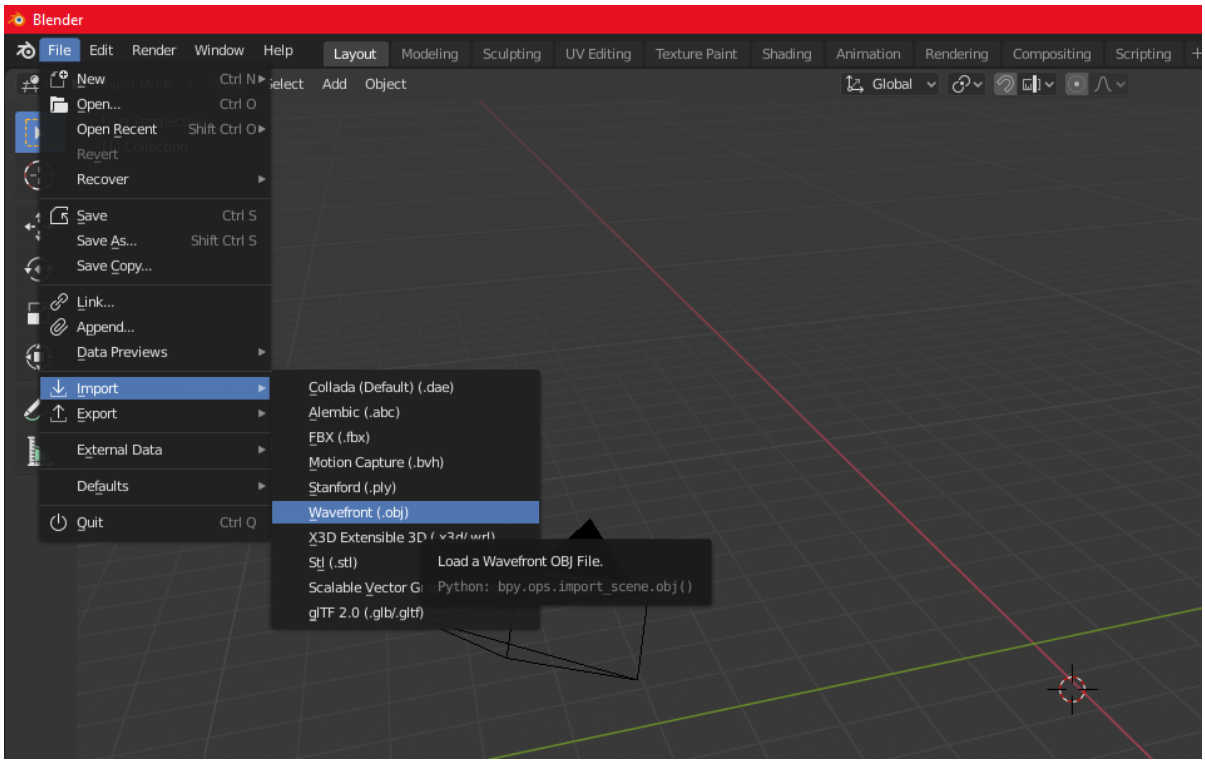
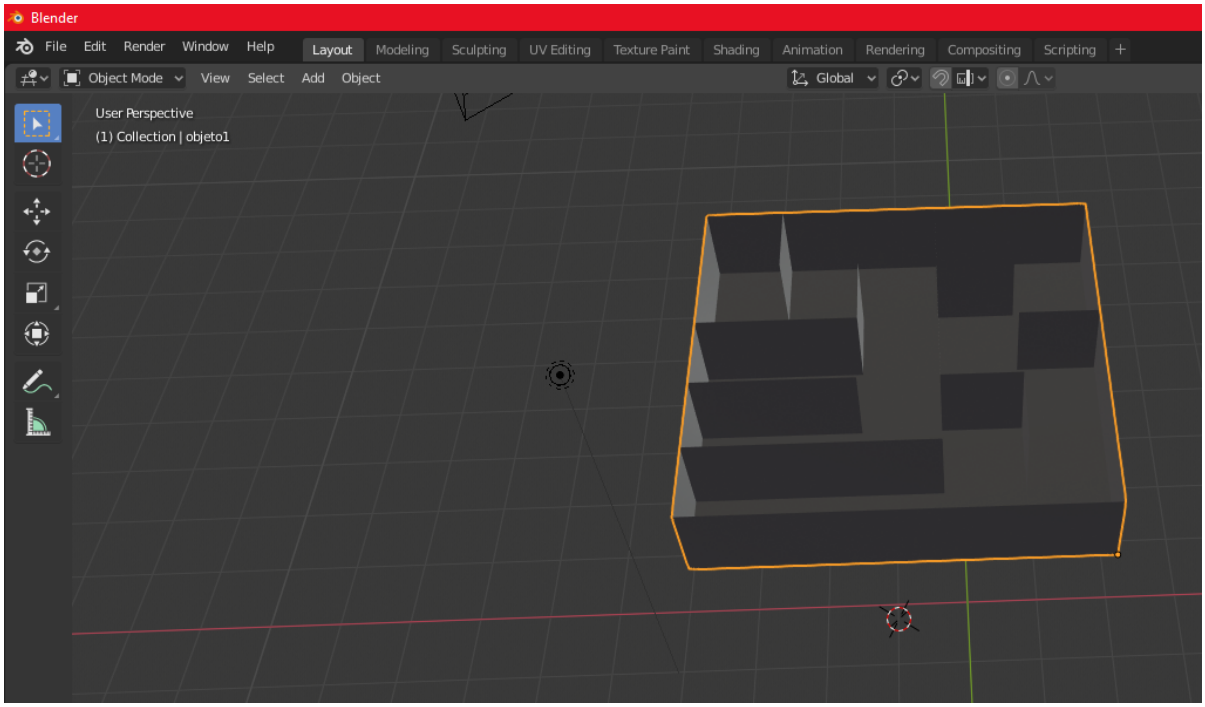


Figura 11: Archivos generados por cada método.



(a) Seleccionamos importar y seleccionarmos el archivo **.obj** que está en la carpeta *Resources* de nuestro proyecto.



(b) Nuestro laberinto cargado correctamente en blender.

Figura 12: ¡Importando un laberinto a Blender!