Capitulo 9

UD09. MOTORES DE XOGO

Resultados de avaliación

- **RA1.** Desenvolve programas que integren contidos multimedia, para o que analiza e emprega as tecnoloxías e as librarías específicas.
- **RA2.** Analiza a arquitectura de xogos 2D e 3D seleccionando e probando motores de xogos. SI
- **RA3.** Desenvolve xogos 2D e 3D sinxelos utilizando motores de xogos.

Criterios de avaliación

- CA1.4 Utilizáronse clases para crear e manipular figuras gráficas en 2D.
- CA1.5 Utilizáronse clases para reproducir e xestionar música e sons.
- CA1.6 Utilizáronse clases para a conversión de datos multimedia dun formato a outro.
- CA1.7 Utilizáronse clases para o control de eventos e excepcións, etc.
- CA1.8 Utilizáronse clases para a creación e o control de animacións.
- CA1.9 Utilizáronse clases para construír reprodutores de contidos multimedia.
- CA1.10 Depuráronse e documentáronse os programas desenvolvidos.
- CA2.1 Identificáronse os elementos da arquitectura dun xogo 2D e 3D.
- CA2.2 Analizáronse as funcións e os compoñentes dun motor de xogos.
- CA2.3 Analizáronse contornos de desenvolvemento de xogos.
- CA2.4 Analizáronse motores de xogos, as súas características e as súas funcionalidades.
- CA2.5 Identificáronse os bloques funcionais dun xogo.
- CA2.6 Definíronse e executáronse procesos de rénder.
- CA2.7 Recoñeceuse a representación lóxica e espacial dunha escena gráfica sobre un xogo existente.
- CA3.1 Deseñouse o proxecto de desenvolvemento dun xogo novo.
- CA3.2 Estableceuse a lóxica do xogo.
- CA3.3 Seleccionouse e instalouse o motor e o contorno de desenvolvemento.
- CA3.4 Creáronse obxectos e definíronse os fondos.
- CA3.5 Instaláronse e utilizáronse extensións para o manexo de escenas.
- CA3.6 Utilizáronse instrucións gráficas para determinar as propiedades finais da superficie dun obxecto ou dunha imaxe.
- CA3.7 Incorporóuselles son aos eventos do xogo.
- CA3.8 Desenvolvéronse e implantáronse xogos para dispositivos móbiles.
- CA3.9 Realizáronse probas de funcionamento e mellora dos xogos desenvolvidos.
- CA3.10 Documentáronse as fases de deseño e desenvolvemento dos xogos creados.

BC. DMotores de Xogo

- Reprodución e control de animacións.
- Procesamento e reprodución de obxectos multimedia: clases, estados, métodos e eventos.
- Documentación do desenvolvemento de aplicacións con contido multimedia.
- Programación de aplicacións con gráficos en dúas dimensións.

- Programación de aplicacións con música e sons.
- Conversión de información multimedia.
- Manexo de eventos e excepcións.
- Animación 2D e 3D.
- Arquitectura do xogo: compoñentes.
- Motores de xogos: tipos e uso.
- Áreas de especialización, librarías utilizadas e linguaxes de programación.
- Compoñentes dun motor de xogos.
- Librarías que proporcionan as funcións básicas dun motor 2D/3D.
- API dos gráficos 3D.
- Estudo de xogos existentes.
- Aplicación de modificacións sobre xogos.
- Proxecto de desenvolvemento: fases, estrutura e obxectivo.
- 0Incorporación de música e efectos sonoros.
- Desenvolvemento de xogos para dispositivos móbiles.
- Análise de execución. Optimización do código.
- Documentación do desenvolvemento dos xogos creados.
- Lóxica do xogo.
- Contornos de desenvolvemento para xogos
- Integración do motor de xogos en contornos de desenvolvemento.
- Obxectos gráficos.
- Escenas e fondos.
- Propiedades dos obxectos: luz, texturas, reflexos e sombras.
- Aplicación das funcións do motor gráfico. Renderización.
- Aplicación das funcións do grafo de escena. Tipos de nodos e o seu uso

Autor/a: Sabela Sobrino Última actualización: 08.02.2024

SUBSECCIONES DE UD09. MOTORES DE XOGO

Capitulo 1

UNITY

UNITY

Unity es un motor de juego multiplataforma utilizado para el desarrollo de videojuegos y aplicaciones interactivas. Es una herramienta versátil y ampliamente adoptada en la industria del desarrollo de juegos debido a su facilidad de uso, flexibilidad y capacidad para desplegar aplicaciones en una variedad de plataformas. Aquí

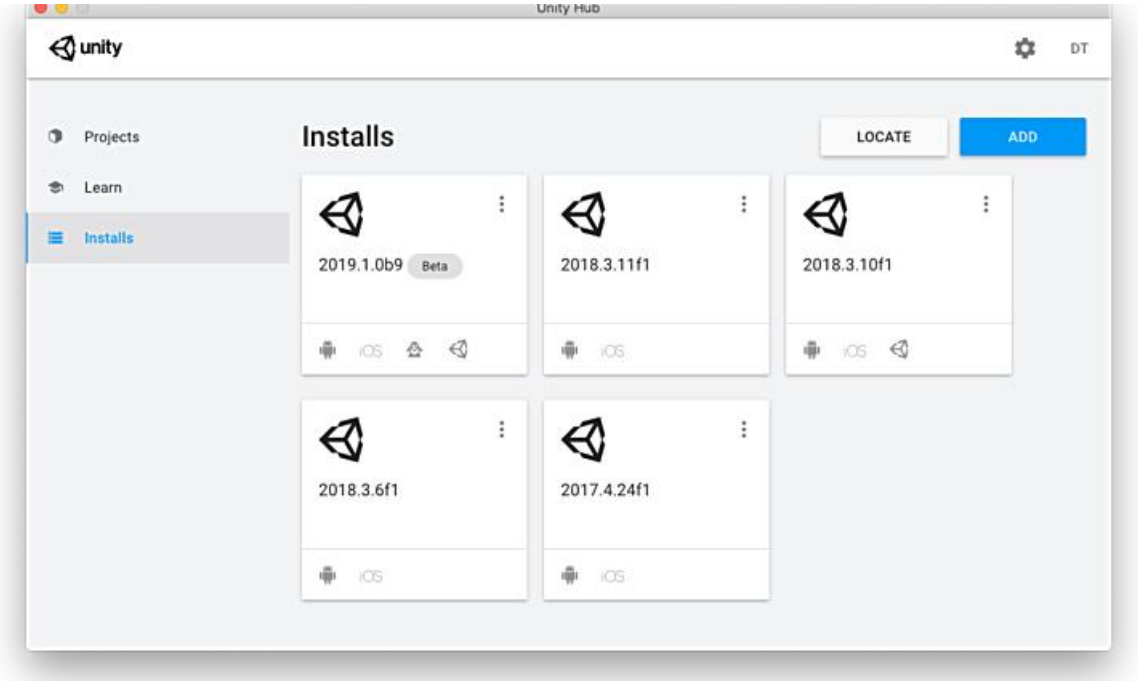
Autor/a: Sabela Sobrino Última actualización: 08.02.2024

SUBSECCIONES DE UNITY

INSTALACIÓN DE UNITY

Al acceder a la página de Unity, nos encontramos con dos opciones principales: Unity y Unity Hub. En este caso, optaremos por utilizar Unity Hub. Unity Hub es una aplicación que simplifica el proceso de descarga, instalación y gestión de Unity, así como la administración de diversos proyectos. Descarga la versión adecuada para tu sistema operativo (Windows, Linux o macOS). Ejecuta el instalador que has descargado y sigue las instrucciones del instalador para completar la instalación.

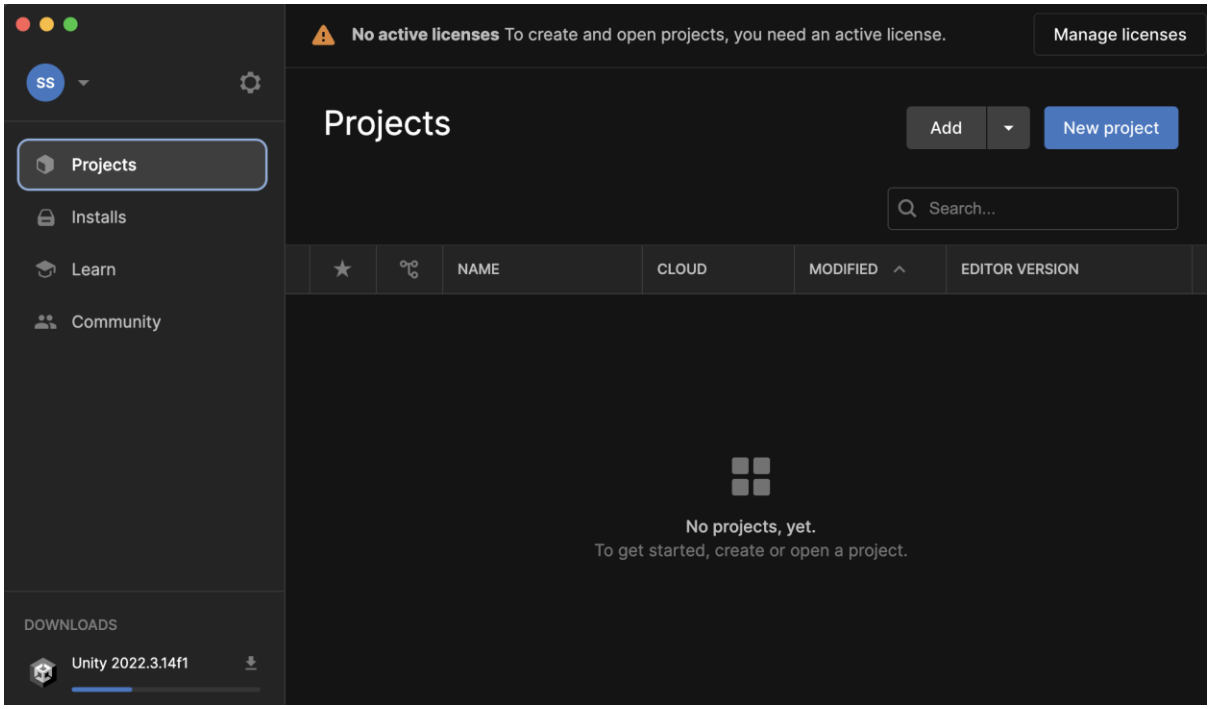
Unity Hub consta de tres secciones fundamentales:



1. **Projects:** Esta sección facilita la administración de nuestros proyectos, permitiéndonos organizarlos de manera eficiente.
2. **Learn:** Incluye proyectos y recursos de aprendizaje descargables, brindando a los usuarios la oportunidad de mejorar sus habilidades en Unity.
3. **Installs:** Aquí podemos gestionar las distintas versiones de Unity instaladas localmente, ofreciendo flexibilidad en el uso de diferentes versiones para diferentes proyectos.

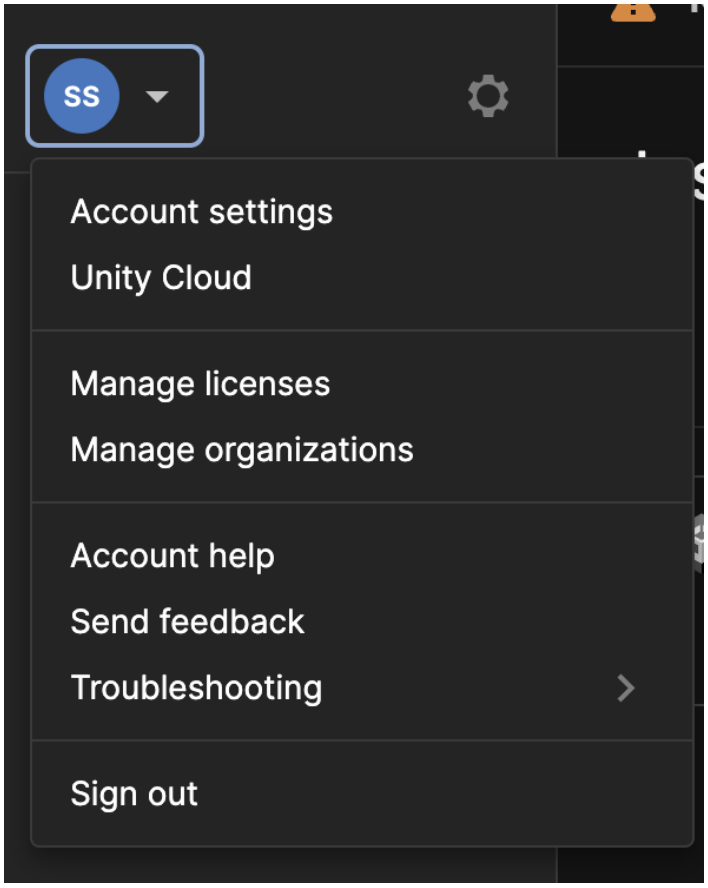
Desde Unity Hub, la descarga de diversas versiones de Unity y la gestión de proyectos resultan intuitivas y eficaces. Una vez descargadas las versiones necesarias, se requiere iniciar sesión para acceder a todas las funciones disponibles.

Posteriormente, al ingresar, se abrirá una pantalla que muestra los distintos proyectos disponibles, brindando una interfaz fácil de usar para la administración y selección de proyectos específicos.

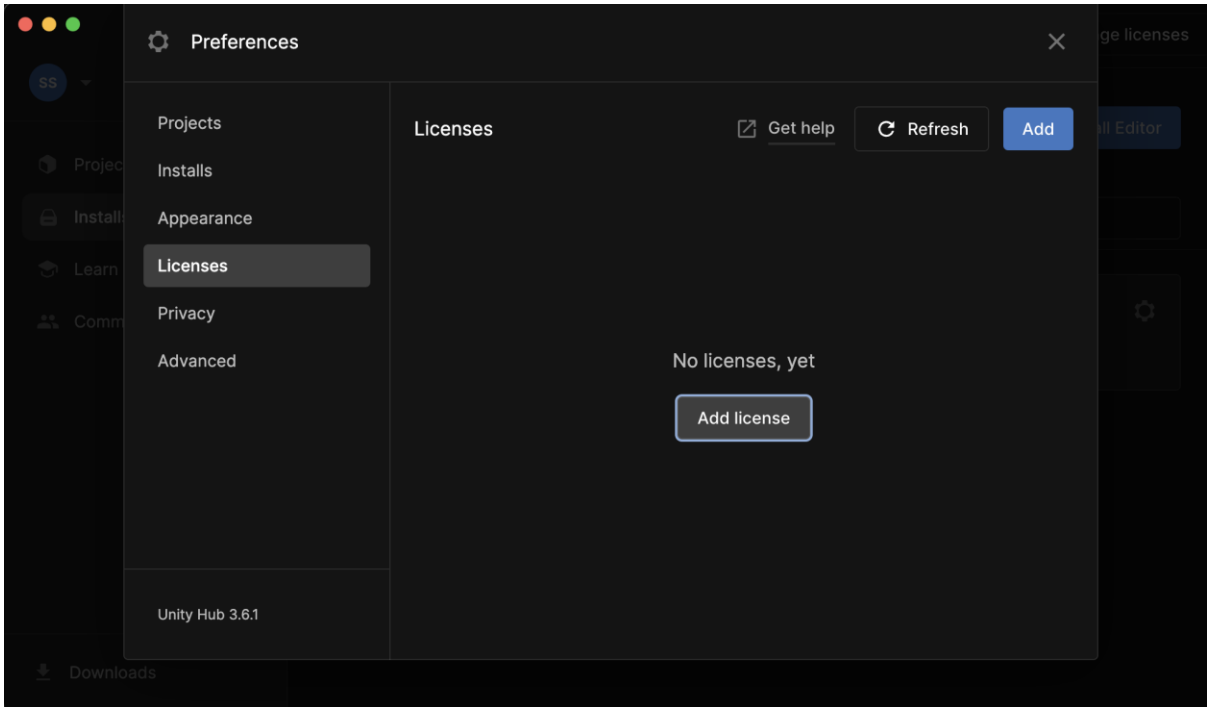


REGISTROS Y LICENCIAS

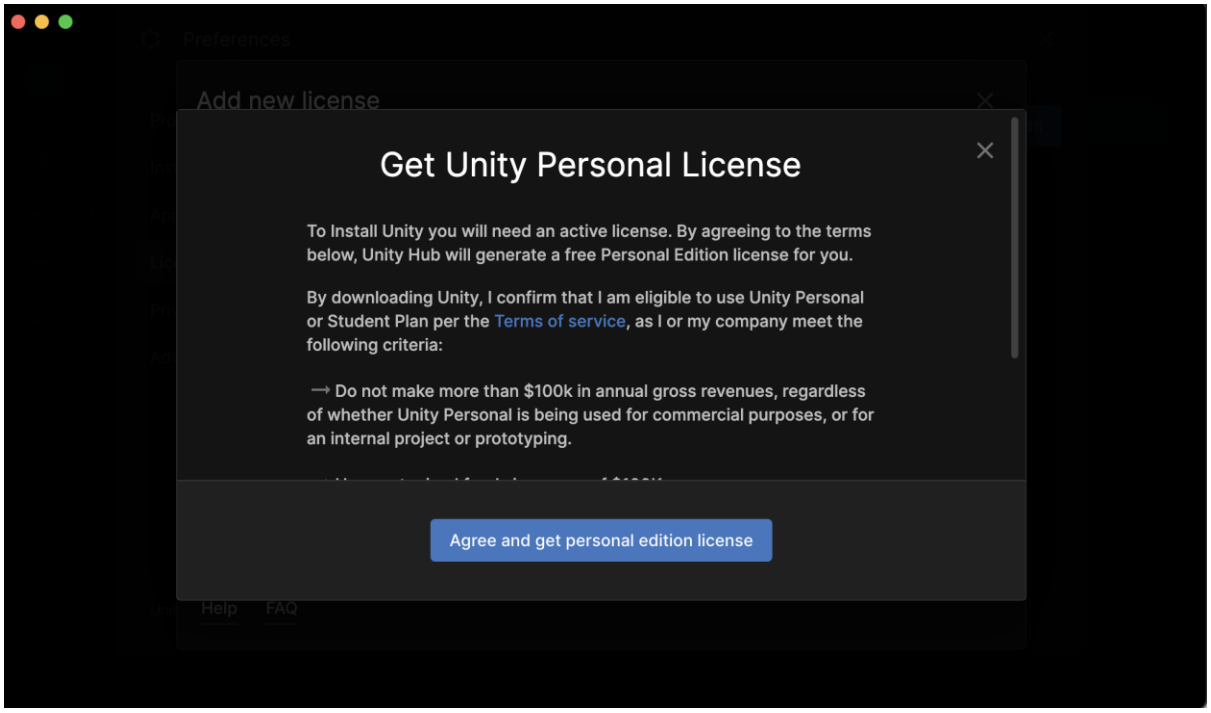
Para utilizar Unity, es necesario estar registrado en la plataforma. Puedes registrarte en la página oficial de Unity y crear tu cuenta:



En la sección de preferencias, puedes verificar el tipo de licencia asociada a tu cuenta.



Inicialmente, es posible que no tengas ninguna licencia asignada, pero puedes agregar una seleccionando la opción “Add” (“Añadir”). Seleccionaremos la opción “Get a Free Personal License” (“Obtener una licencia personal gratuita”):



Este proceso asegura que estás utilizando Unity con la licencia adecuada y te permite acceder a las funcionalidades correspondientes para tu proyecto.

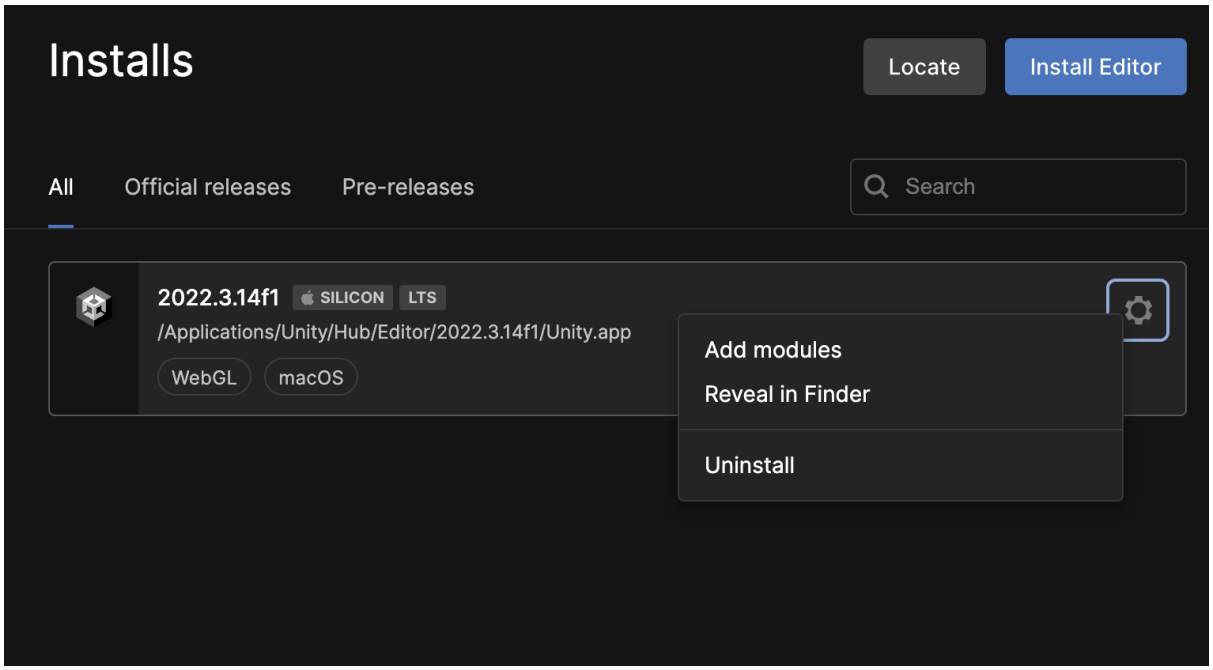
Autor/a: Sabela Sobrino Última actualización: 08.02.2024

EDITOR

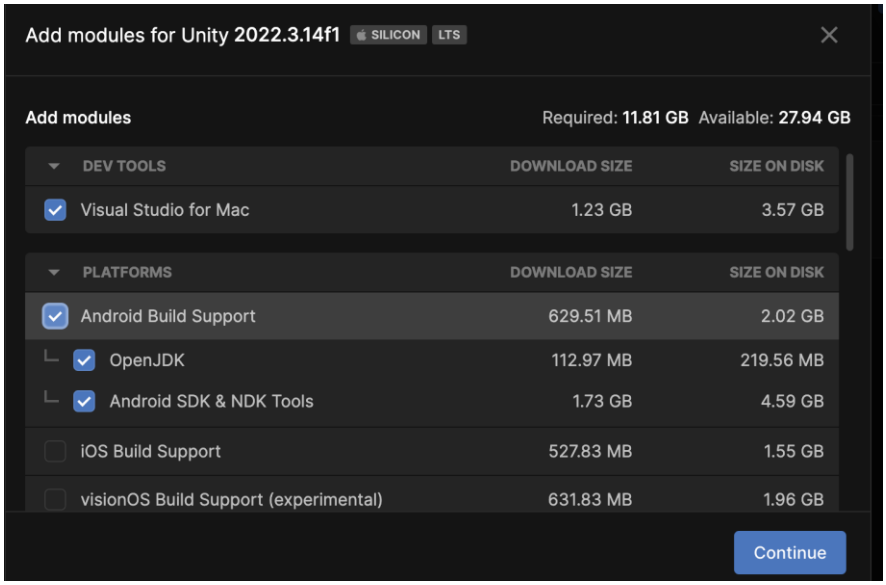
Instalación

Hasta ahora, hemos instalado Unity Hub, pero aún necesitamos instalar Unity en sí. Para lograrlo, dirigámonos a la sección de “Installs” y, en caso de no haberlo hecho previamente, añadamos una versión de Unity con soporte a largo plazo (LTS).

Una vez completada la instalación, podemos acceder a la configuración y agregar nuevos módulos:



En este punto, optaremos por seleccionar el módulo de Android, ya que estamos trabajando en un proyecto que se implementará en un dispositivo móvil:



Esta acción nos permite integrar el soporte necesario para el desarrollo y despliegue de proyectos específicos para Android en Unity. Asegúrate de tener los módulos adecuados instalados para satisfacer los requisitos de nuestro proyecto móvil.

Visual Studio Code

Unity es compatible con varios entornos de desarrollo integrado (IDE) y editores, entre los cuales se incluyen Microsoft Visual Studio, Visual Studio Code, JetBrains Rider, y más.

Visual Studio Code (VS Code): Este es un editor gratuito, ligero, multiplataforma y altamente extensible diseñado para el desarrollo de aplicaciones. Puedes descargarlo desde el siguiente enlace: Descargar VS Code.

Para configurar tu editor preferido en Unity, simplemente sigue estos pasos:

1. Accede a las preferencias de Unity.
2. Navega a External Tools -> External Script Editor.

Además, si estás utilizando Visual Studio Code con Unity, puedes realizar la configuración específica siguiendo las indicaciones proporcionadas en la documentación oficial de VS Code para Unity, disponible en este enlace: Configuración de VS Code para Unity.

Este proceso asegura que tu entorno de desarrollo esté configurado de manera óptima para trabajar con Unity, permitiéndote aprovechar al máximo las funcionalidades de tu editor preferido.

Editor

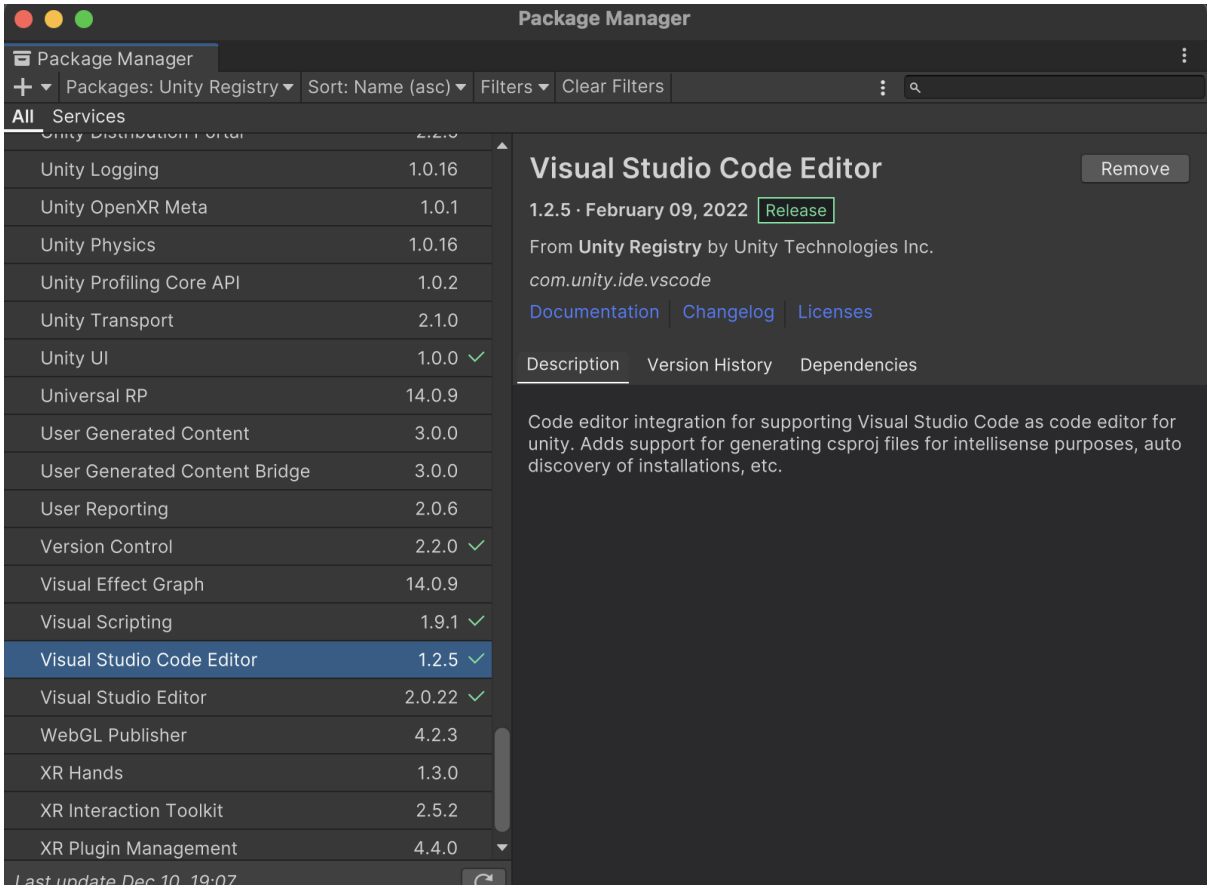
Unity es compatible con diversos entornos de desarrollo integrado (IDE) y editores, tales como Microsoft Visual Studio, Visual Studio Code, JetBrains Rider, entre otros.

- **Visual Studio Code (VS Code):**
  - Un editor gratuito, ligero, multiplataforma y extensible para el desarrollo de aplicaciones.
  - Descarga: VS Code
- **Selección del Editor en Unity:**
  - Configura tu editor preferido en Unity yendo a Preferences, External Tools -> External Script Editor.
- **Configuración de VS Code para Unity:**
  - Detalles sobre cómo configurar VS Code para trabajar con Unity: Configuración de VS Code

IntelliSense

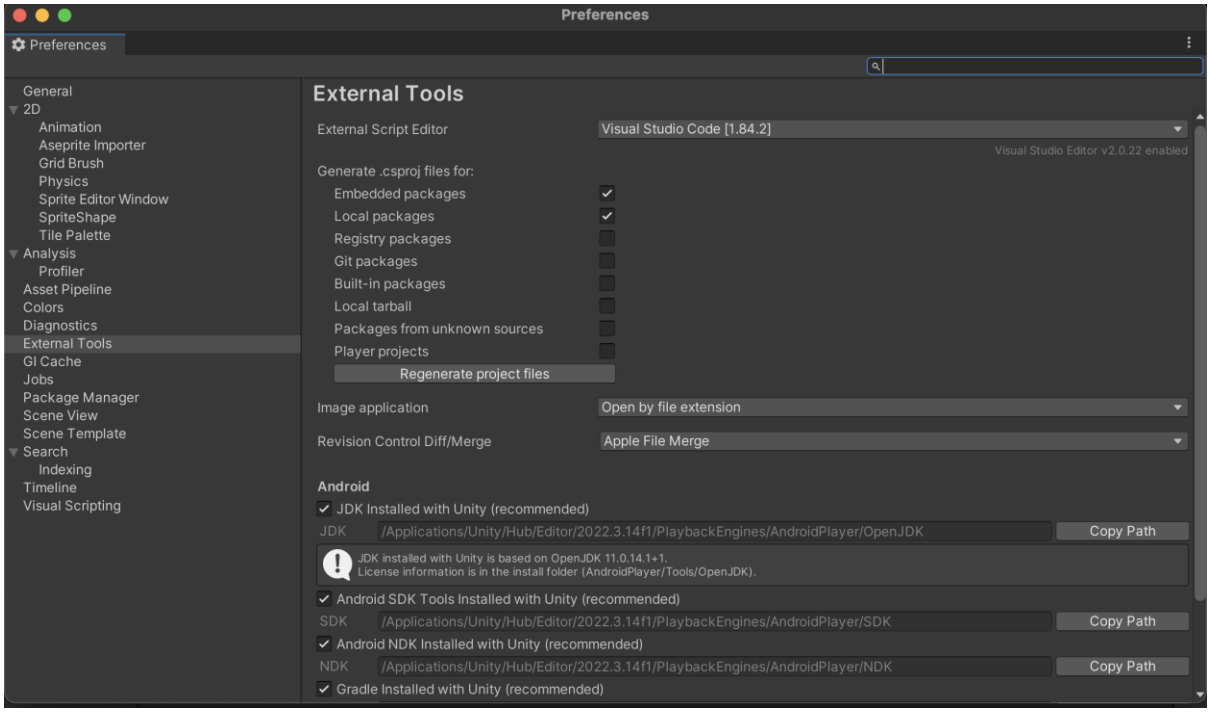
IntelliSense es una función de autocompletado utilizada en Microsoft Visual Studio y Visual Studio Code. Si encuentras problemas con IntelliSense, sigue estos pasos:

1. Verifica que el paquete “Visual Studio Code Editor” esté instalado en Unity Package Manager.
   - En Unity, ve a: Window > Package Manager. Selecciona “Packages: Unity Registry” y busca “Visual Studio Code Editor” en la lista. Asegúrate de que esté instalado (si aparece el botón “Remove”, ya está instalado).



Asegúrate de que VS Code esté seleccionado como el External Script Editor.

- Ve a: Edit > Preferences > External Tools y selecciona Visual Studio Code en la lista.
- Puede ser necesario hacer clic en “Regenerate project files” con la opción “Registry packages” seleccionada.



1. Si el problema persiste, asegúrate de tener instalado el .NET framework correcto en tu máquina.
    - Descarga la última versión de .NET Core SDK y .NET SDK desde: Descargar .NET
2. Abre VS Code y verifica la configuración del espacio de trabajo:
    - Pulsa Ctrl + Shift + P para abrir la paleta de comandos.
    - Escribe “Omnisharp: Select Project” y selecciona la solución del espacio de trabajo (.sln) para el proyecto.
3. Si aún así no funciona descarga la versión 3.1 de .NET Core SDK y .NET SDK desde: Descargar .NET

    DOTNET_FILE=dotnet-sdk-9.0.101-linux-x64.tar.gz export DOTNET_ROOT=$(pwd)/.dotnet mkdir -p “$DOTNET_ROOT” && tar zxf “$DOTNET_FILE” -C “$DOTNET_ROOT” export PATH=$PATH:$DOTNET_ROOT:$DOTNET_ROOT/tools export DOTNET_ROOT=$HOME/.dotnet dotnet

4. Añadir al perfil de usuario para que se mantenga cuando reiniciemos el equipo:

    nano ~/.profile dotnet source ~/.bashrc dotnet –list-sdks

if [-d “$HOME/.dotnet”] ; then

PATH="$HOME/.dotnet"

fi

Autor/a: Sabela Sobrino Última actualización: 20.01.2025

VIRTUALIZACIÓN

No se recomienda ejecutar Unity en un entorno virtualizado, ya que las máquinas virtuales limitan el acceso directo a la GPU, lo que impacta negativamente en la aceleración gráfica necesaria para un rendimiento fluido en la edición y el renderizado. Además, las máquinas virtuales suelen ofrecer menos recursos de hardware compartido, lo que puede generar latencia, ralentizaciones y una experiencia de usuario deficiente, especialmente en proyectos de mayor envergadura. Unity es intensivo en el uso de CPU, GPU y RAM, por lo que ejecutarlo en un sistema virtualizado puede dificultar el desarrollo y las pruebas en tiempo real.

Por este motivo, instalaremos Unity y el resto de herramientas en el sistema operativo nativo. Se recomienda utilizar la última versión de Lubuntu, concretamente la 24.04 LTS.

Autor/a: Sabela Sobrino Última actualización: 20.01.2025

GITIGNORE

Para poder tener una sincronización entre nuestros proyectos y github debemos tener este .gignore. Muy importante para no subir todo el contenido (muchos gigas):

```
# This .gitignore file should be placed at the root of your Unity project directory
#
# Get latest from https://github.com/github/gitignore/blob/main/Unity.gitignore
#
/[Ll]ibrary/
/[Tt]emp/
/[Oo]bj/
/[Bb]uild/
/[Bb]uilds/
/[Ll]ogs/
/[Uu]ser[Ss]ettings/

# MemoryCaptures can get excessive in size.
# They also could contain extremely sensitive data
/[Mm]emoryCaptures/

# Recordings can get excessive in size
/[Rr]ecordings/

# Uncomment this line if you wish to ignore the asset store tools plugin
# /[Aa]ssets/AssetStoreTools*

# Autogenerated Jetbrains Rider plugin
/[Aa]ssets/Plugins/Editor/JetBrains*

# Visual Studio cache directory
.vs/

# Gradle cache directory
.gradle/

# Autogenerated VS/MD/Consulo solution and project files
ExportedObj/
.consulo/
*.csproj
*.unityproj
*.sln
*.suo
*.tmp
*.user
*.userprefs
*.pidb
*.booproj
*.svd
*.pdb
*.mdb
*.opendb
*.VC.db

# Unity3D generated meta files
*.pidb.meta
*.pdb.meta
*.mdb.meta

# Unity3D generated file on crash reports
sysinfo.txt
```

```
# Builds
*.apk
*.aab
*.unitypackage
*.app

# Crashlytics generated file
crashlytics-build.properties

# Packed Addressables
/[Aa]ssets/[Aa]ddressable[Aa]ssets[Dd]ata/*/*.bin*

# Temporary auto-generated Android Assets
/[Aa]ssets/[Ss]treamingAssets/aa.meta
/[Aa]ssets/[Ss]treamingAssets/aa/*
```