«Talento Tech»

Desarrollo de Videojuegos

Unity 3D

Clase 15

Buenos Aires Ciudad

Clase N° 15 | Mobile

Temario:

- Adaptación de controles para Mobile
- Building para Mobile

Objetivos de la clase

- Crear Joysticks en la UI.
- Usar Input System
- Configurar y Buildear para Mobile.

Introducción a los controles en Mobile

Cuando adaptamos un juego de PC a plataformas móviles, una de las principales diferencias radica en **cómo el jugador interactúa con el juego**. En PC, solemos usar teclado, mouse o gamepads físicos. Sin embargo, en smartphones o tablets no contamos con botones físicos estándar, lo que nos obliga a **implementar controles completamente táctiles**.

Métodos de Entrada en Dispositivos Móviles

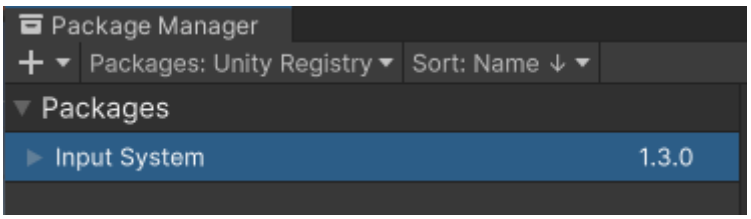
1. **Pantalla táctil (Touch Input)**
    - Es la forma más común de interacción en mobile.
    - Se usa para detectar toques (TouchPhase.Began), mantener presionado (TouchPhase.Stationary), y soltar (TouchPhase.Ended).
    - Puede usarse para botones virtuales, gestos de deslizamiento (swipe) o toques en la pantalla.
2. **Acelerómetro (Gyroscope)**
    - Se usa para detectar el movimiento del dispositivo (inclinación, rotación).
    - Se puede usar para juegos que requieren movimiento del dispositivo, como juegos de carreras o realidad aumentada.
3. **Joystick Virtual**
    - Se implementa con elementos de UI (Image, Button, EventTrigger).
    - Simula un joystick físico en la pantalla, permitiendo controlar el movimiento del personaje con el pulgar.
    - Se usa en juegos de aventura, RPGs y shooters en tercera persona.

Configuración Inicial: Activando el Nuevo Input System

Antes de crear nuestros joysticks, es fundamental asegurarnos de que Unity esté utilizando el Nuevo Sistema de Entradas (Input System), ya que este es más moderno y flexible que el antiguo.

Paso 1: Instalar el Paquete del Input System:

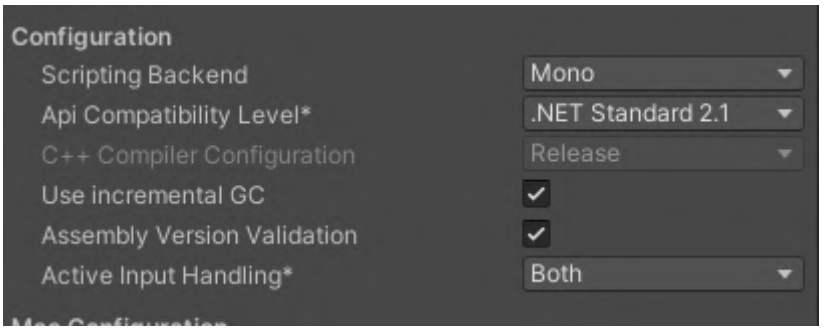
- Movete hasta ‘Window’ → ‘Package Manager’.
- En la lista desplegable ‘Packages: Unity Registry’, busca y selecciona ‘Input System’.
- Haz clic en el botón ‘Install’ (si no está ya instalado).



Antes de comenzar, asegúrate de que tu proyecto está usando el Input System en lugar del antiguo sistema de entrada.

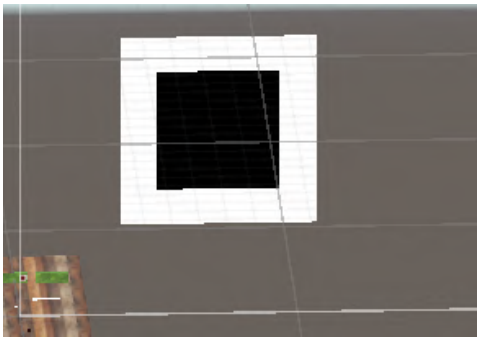
Paso 2: Activar el Input System en la Configuración del Proyecto:

- Movete a `Edit` → `Project Settings`
- En la ventana de Project Settings, selecciona la pestaña ‘Player’
- Desplázate hacia abajo hasta la sección ‘Other Settings’ y localiza la opción ‘Active Input Handling’
- Cámbialo a ‘Both’ (si queres usar ambos sistemas, el nuevo y el antiguo) o Input System Package (New) (si solo usarás el nuevo)
- Es probable que Unity te pida reiniciar el editor para aplicar los cambios.



Crear la estructura en UI:

- Añade un Canvas con Screen Space - Overlay.
- Dentro del Canvas, crea un Image (para la base del joystick).
- Dentro de la base, crea otro Image (para el "handle" o palanca).



Recordá que podes elegir distintas imágenes, esta es cuadrada a base de ejemplo.

Configurar RectTransform:

Selecciona el objeto "JoystickBase" en el Hierarchy y, en su componente RectTransform (en el Inspector):

1. Posición: Ubica el joystick en la esquina inferior izquierda de la pantalla o en la posición que mejor se adapte al diseño de tu juego.

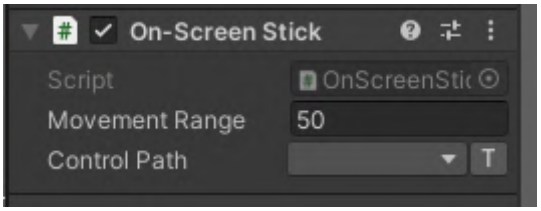
2. Anclajes (Anchors): Ajusta los anclajes (Anchors) para que el joystick se mantenga fijo en esa posición sin importar la resolución de la pantalla. Generalmente, para una esquina, usarás los anclajes correspondientes a esa esquina (por ejemplo, bottom-left para la esquina inferior izquierda).

3. Tamaño: Define el Width y Height del joystick para que tenga un tamaño cómodo para el uso táctil.

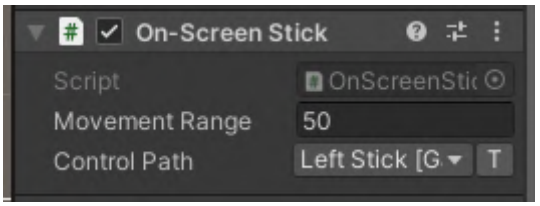
Agregar el component On-Screen Stick

Ahora, vamos a darle funcionalidad a nuestro joystick visual:

1. Selecciona el objeto "JoystickBase" en el Hierarchy.
2. En el Inspector, haz clic en *Add Component* y busca *On-Screen Stick.*

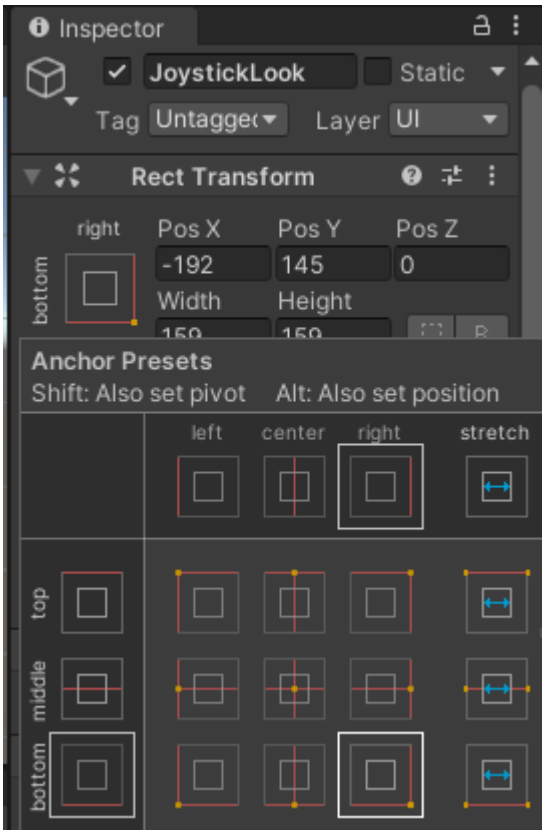


3. Este componente es clave, ya que es el que convierte nuestro elemento de UI en un joystick funcional cuando ejecutes el juego.
4. **Movement Range**: Esta variable controla qué tan lejos se puede mover la palanca (Handle) desde el centro de la base del joystick. Ajusta este valor para un control cómodo.
5. **Control Path:** Aquí es donde le indicamos al Input System a qué entrada virtual se va a mapear nuestro joystick.
    - Haz clic en el selector (T o el menú desplegable).



    - Navega a GamePad → Left Stick. Esto le dice a Unity que este joystick virtual actuará como el "stick izquierdo" de un gamepad, lo cual es esencial para que tu código de movimiento lo reconozca.

Joystick para el control de cámara

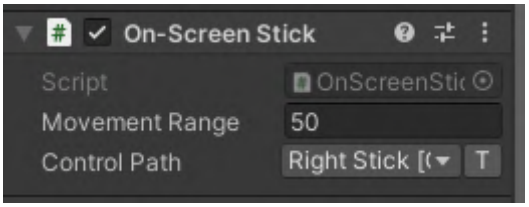


Para tener control sobre la cámara (o la "mirada" del personaje), duplicaremos el joystick que acabamos de crear y lo configuraremos para esa función:

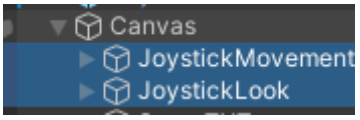
1. **Duplica el "JoystickMovement":** Selecciona el "*JoystickMovement*" en el Hierarchy y presiona Ctrl+D (o Cmd+D en Mac). Renómbralo a "*JoystickLook*".

2. **Posición y Anclaje**: Mueve "*JoystickLook*" a la esquina inferior derecha de la pantalla. Asegúrate de modificar sus anclajes (Anchors) para que se fije correctamente en esa nueva posición.

Con esto, tendremos nuestros dos joysticks virtuales perfectamente ubicados y configurados en el canvas: uno para el movimiento y otro para el control de la cámara.



Así tendremos los 2 Joystick en nuestro Canvas:

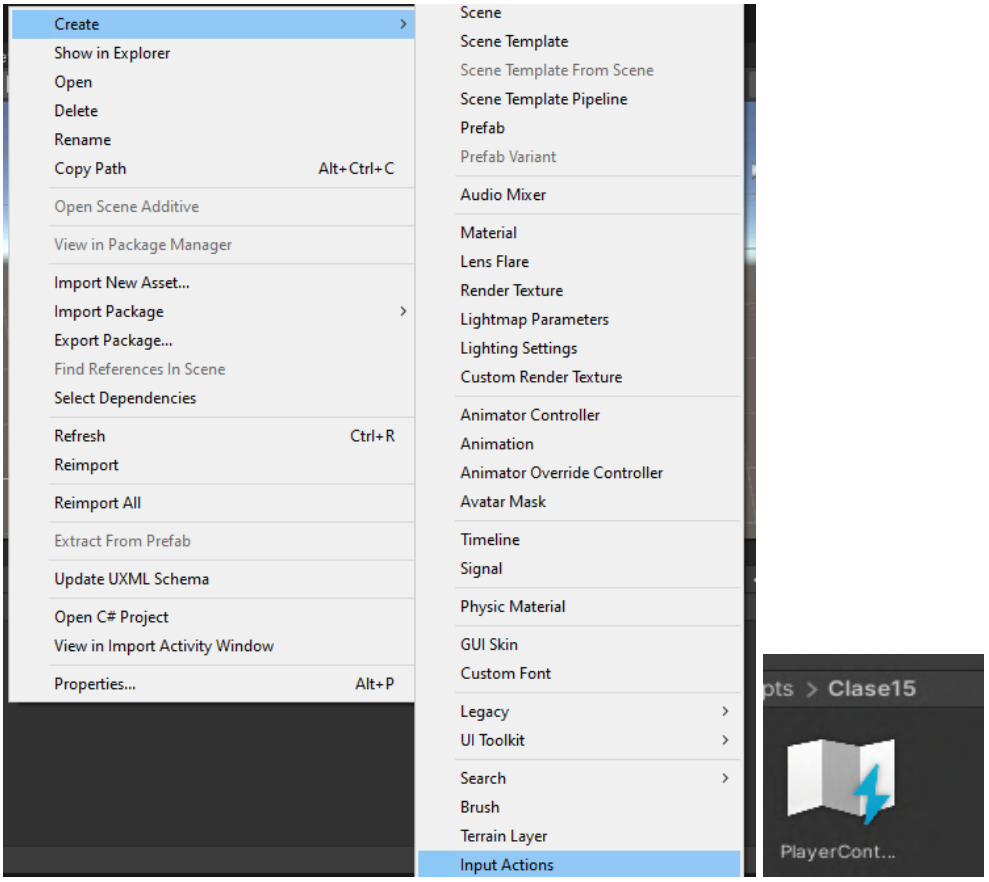


Input Action

Crearemos la conexión para que estos elementos sean reconocido como Inputs.

Paso 1: Crear el Input Action Asset:

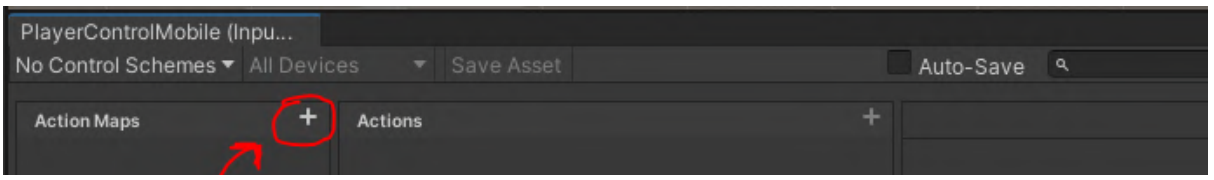
- En Project haremos Click Derecho -> Create -> Input Action



- Lo renombramos como “**PlayerControlMobile**”

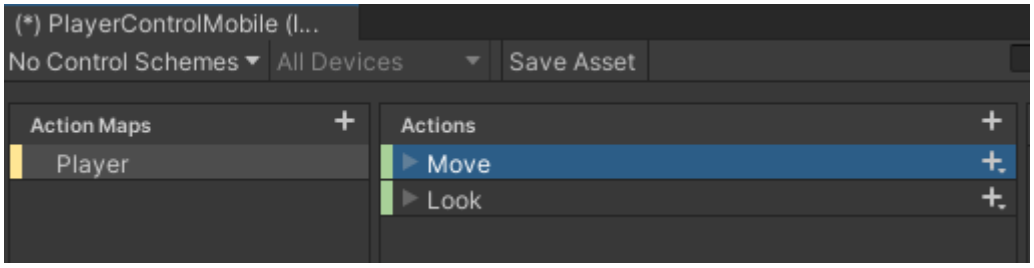
Paso 2: Definir las Acciones de Movimiento y Cámara:

- **Hacemos doble clic** en el asset "*PlayerControlMobile*" para abrir la ventana. Lo abriremos y luego buscaremos la opción para agregar un nuevo **Action Map**



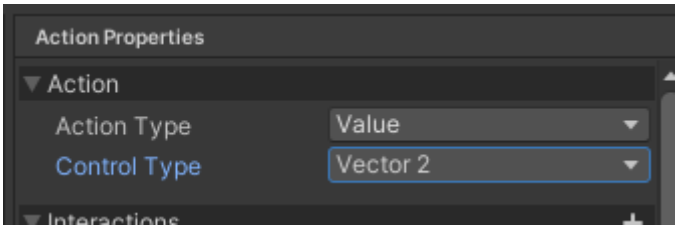
- **Crearemos uno y le agregaremos 2 acciones, “Move” y “Look”**
  En la columna “Actions” (dentro del Action Map "Player"), hacemos clic en el botón

“+” para añadir dos nuevas acciones: **"Move"** (para el movimiento del personaje) y **"Look"** (para el movimiento de la cámara/mirada).

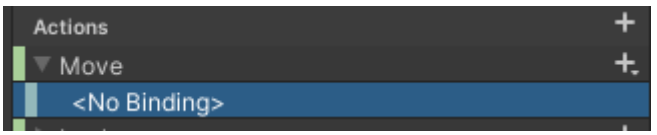


Paso 3: Configurar la Acción "Move":

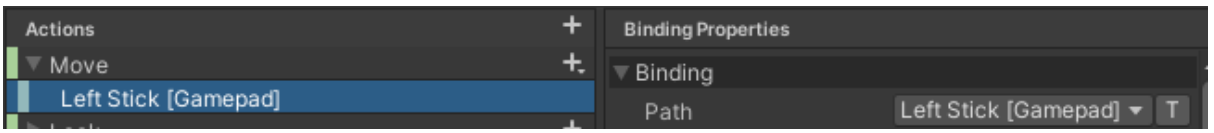
- Le damos click en Move y vamos a Action Properties para cambiar las variables **Action Type = Value** y **Control Type = Vector 2**.



- Debajo de "Move", verás un Binding que dice "<No Binding>". Haz clic en él.

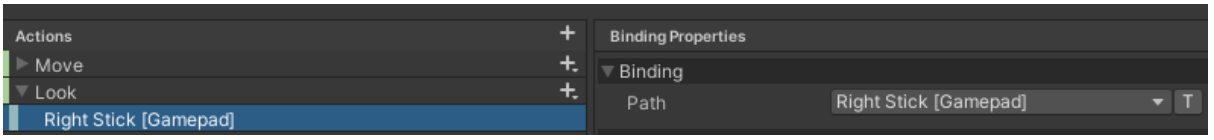


- En la ventana Binding Properties (lado derecho), en Path, selecciona: GamePad → Left Stick. Esto conecta nuestra acción "Move" con la entrada del joystick izquierdo del gamepad virtual.



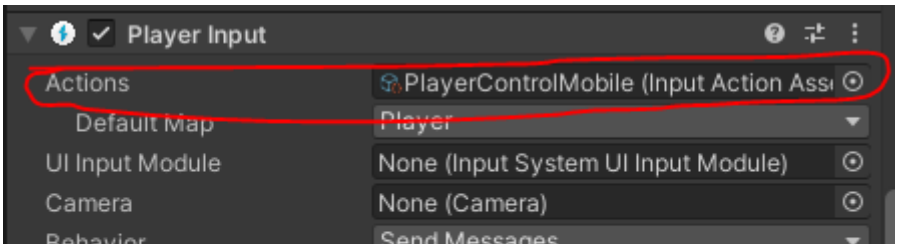
Paso 4: Configurar la Acción "Look"

- Repite el proceso para la acción "Look", pero esta vez, en Path, selecciona: GamePad → Right Stick.



Paso 5: Asignar el Input Action al Player

En nuestro Player pondremos el Component "**Player Input**" y le arrastraremos nuestra Action Map a la variable Actions.



Código

Para el código, modificaremos los **Scripts** de movimiento y de la camara para que acepte los Inputs nuevos.

Script de Movimiento - Move

Tendremos que colocar la librería **using UnityEngine.InputSystem;**

```
using UnityEngine.InputSystem;
```

Crearemos una variable de referencia de nuestro sistema de inputs y le asignaremos el componente en el **Start()**

```
PlayerInput pi; // Variable de referencia al componente PlayerInput

void Start()
{
    pi = GetComponent<PlayerInput>();
}
```

Reemplazamos los valores de los inputs

```
void FixedUpdate() {
    // Lee el valor del joystick "Move" como un Vector2
    Vector2 input = pi.actions["Move"].ReadValue<Vector2>();

    // Convierte el Vector2 de input a un Vector3 para el movimiento 3D
(ignorando el eje Y de input para altura)
    Vector3 movementeMobile = new Vector3(input.x, 0, input.y);

      // Normaliza el vector para evitar movimientos más rápidos en
diagonal
    movementeMobile = movementeMobile.normalized;

    // Aplica la velocidad al Rigidbody
         rb.velocity = new Vector3(movementeMobile.x * moveSpeed,
rb.velocity.y, movementeMobile.z * moveSpeed);
}
```

En la línea “ `Vector2 input = pi.actions["Move"].ReadValue<Vector2>();` ” Accedemos al Action **“Move”** que creamos y leemos el valor del Vector2 del joystick izquierdo que se genera constantemente.

Script de control de cámara - Look

1. **Añade la librería y referencia:** Repite los pasos 1 y 2 del script de movimiento en tu script de control de cámara.
2. **Lee el Input y Rota la Cámara:** Dentro del `Update()`:

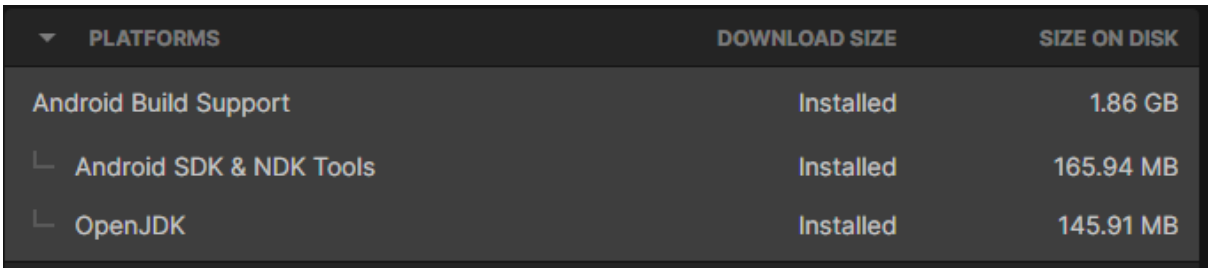
```
void Update() {
    // Lee el valor del joystick "Look" como un Vector2
    Vector2 input = pi.actions["Look"].ReadValue<Vector2>();

    // Ajusta la rotación de la cámara usando los valores del joystick
    Vector3 rotateMobile = new Vector3((input.y), (input.x * -1), 0);
// Notar el -1 para la rotación horizontal
    transform.eulerAngles = transform.eulerAngles - rotateMobile; //
Restamos para una rotación intuitiva
}
```

La lógica es muy similar al movimiento, pero aquí accedemos a la acción "Look" (asociada al joystick derecho). El ajuste de input.x * -1 es común para lograr la rotación horizontal deseada de la cáma

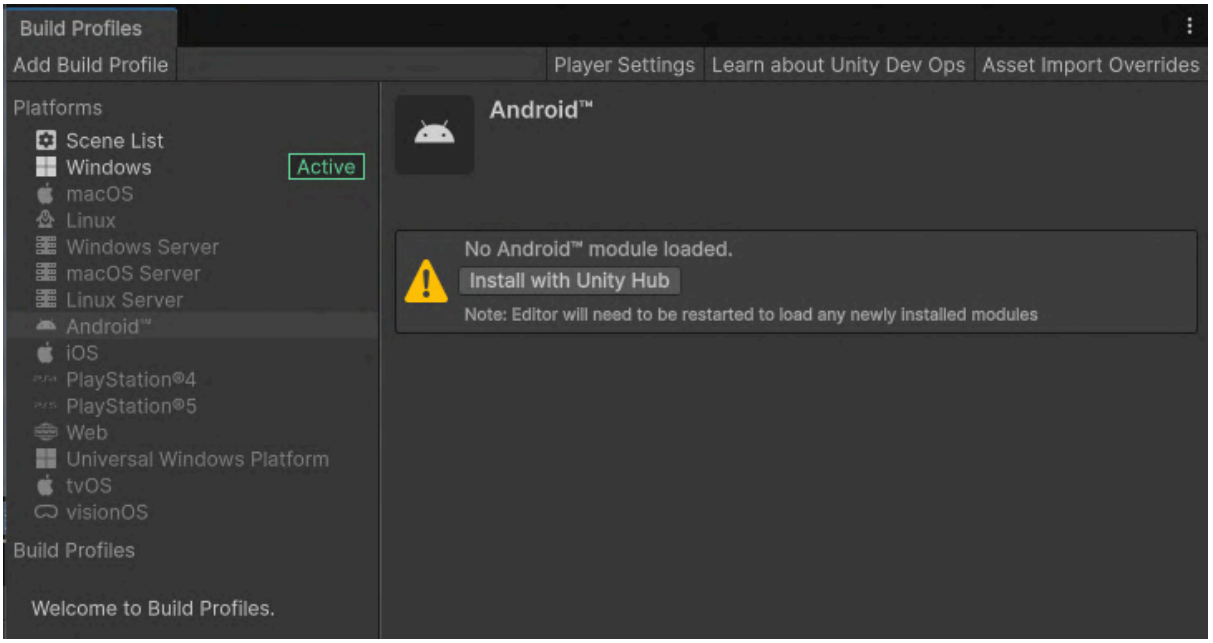
Build para Mobile

Paso 1: Verificamos el Módulo de Android

Para hacer un build destinado a mobile primero tenemos que estar seguros de tener el módulo de **Android instalado en Unity**.
Podemos verificarlo y añadirlo a través de Unity Hub o en Window → Package Manager → Unity Registry (Android Build Support).

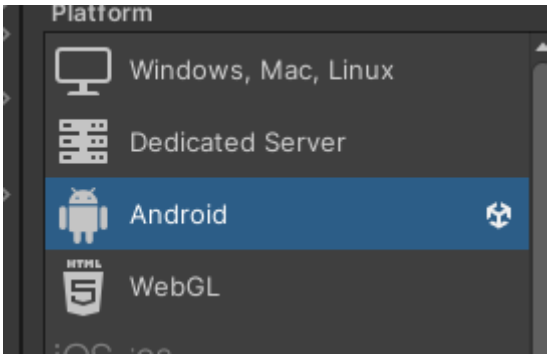


Si no lo tenemos instalado, vamos a file -> Build Profiles -> Buscamos Android y clickeamos. Nos mostrará una pestaña para instalar con Unity Hub.

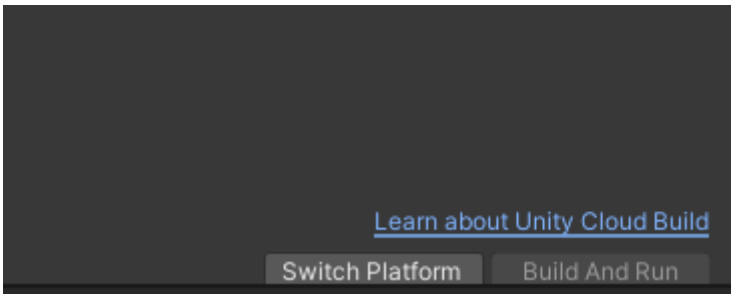


Paso 2: Cambiamos la Plataforma de Build

- Vamos a File → Build Settings
- En la lista de Platform, seleccionamos Android.

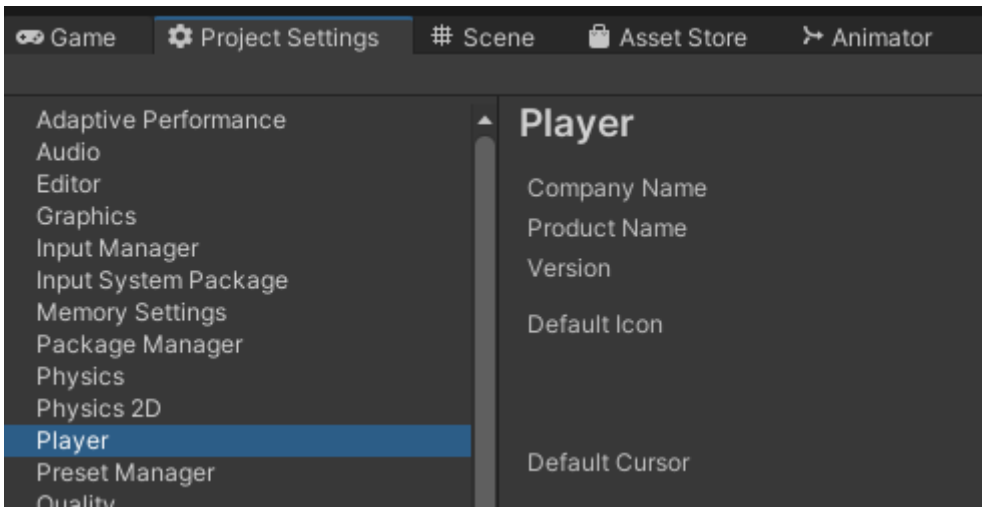


- Hacemos clic en el botón Switch Platform (ubicado en la parte inferior derecha). Esto tomará un tiempo, ya que Unity reimportará los assets para la nueva plataforma.

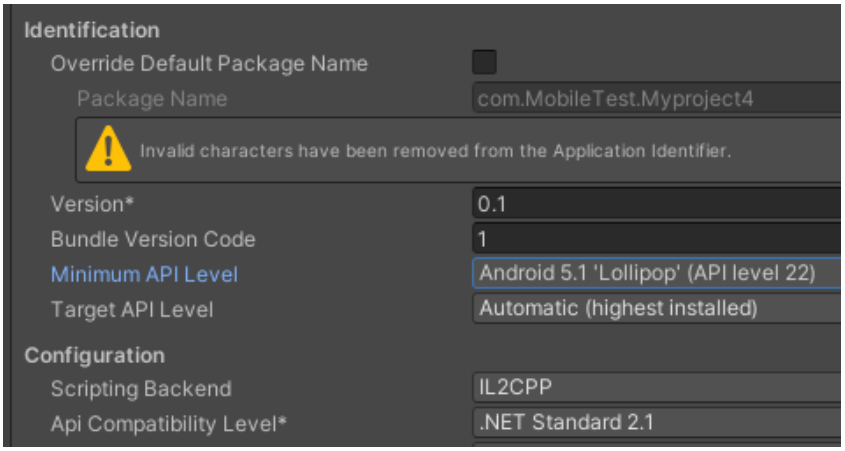


Paso 3: Configurar los Ajustes del Player para Android

1. Una vez que la plataforma haya cambiado, vamos a Edit → Project Settings
2. En la ventana de Project Settings, seleccionamos la pestaña **Player**.



3. Dentro de la sección **Identification**:
    a. **Minimum API Level:** Configúralo al **nivel de API más bajo posible** que desees soportar (esto ampliará la compatibilidad con dispositivos más antiguos).



    b. **`Target API Level`:** Déjalo en **`Automatic (highest installed)`** para que Unity use el SDK más reciente.

4. Dentro de la sección **`Configuration`**:
    a. **`Scripting Backend`:** Cambia el valor de `Mono` a **`IL2CPP`**.

    `IL2CPP` ofrece mejor rendimiento y compatibilidad en dispositivos móviles.

Paso 4: Generar la APK (o AAB):

1. Volvemos a File → Build Settings.
2. Ahora, en lugar de Switch Platform, veremos el botón Build (o Build And Run por si queremos instalarlo directamente en un dispositivo conectado).
3. Hacemos click en Build, elegimos una carpeta para guardar tu archivo, y Unity generará la APK (el paquete de instalación de Android) de tu juego.

Blenvenido al mundo Mobile:

*A medida que **Nexus** avanza, el equipo de **TalentoLab** se enfrenta a un nuevo desafío: expandir el juego a un público más amplio. Con la creciente demanda de juegos móviles, ha llegado el momento de **adaptar el proyecto para plataformas móviles**. Esto implica rediseñar algunas mecánicas y ajustar los controles para ofrecer una experiencia fluida y atractiva en dispositivos táctiles.*

*"Nuestro juego está tomando forma. Ahora necesitamos que todo el mundo, sin importar la plataforma, pueda disfrutar de **Nexus**. ¡Es hora de llevarlo a las manos de los jugadores!"*

*Este es un paso clave en el desarrollo del proyecto, y el equipo debe aprender a manejar los **inputs para mobile**, usando el **Input System** de Unity para que el control sea intuitivo y cómodo.*

Ejercicios prácticos:

Con el juego **Nexus** tomando forma en las plataformas móviles, es el momento de asegurarse de que los jugadores puedan interactuar de manera cómoda y fluida. Para ello, debemos adaptar las acciones del juego a botones táctiles, algo clave para brindar una experiencia completa en dispositivos móviles.

"Recuerda, en dispositivos móviles no contamos con el mismo tipo de controles que en una PC o consola. Necesitamos ofrecer controles intuitivos, accesibles, y con una respuesta inmediata para mantener la experiencia envolvente."

🎯 **Objetivo:** Implementar un botón en la interfaz móvil que permita ejecutar una acción específica dentro del juego.

✔ **Diseñar un botón** en la interfaz móvil que realice una acción que elijas dentro del juego (por ejemplo, un ataque, salto, o interacción).
✔ **Configurar el Input System** para que interactúe correctamente con las mecánicas del juego, activando la acción seleccionada al tocarlo.
✔ **Asegurarte de que sea visible y fácil de usar**, ubicándolo de forma que no interfiera con otros controles, y testeando su funcionamiento en un dispositivo móvil real.
✔ **Personalizarlo** para que tenga una estética adecuada al estilo de **Nexus**, manteniendo la coherencia visual con el resto del juego.

Materiales y recursos adicionales.

Input System
https://docs.unity3d.com/Packages/com.unity.inputsystem@1.13/manual/index.html

Desarrollando con Android
https://docs.unity3d.com/es/530/Manual/android-GettingStarted.html

---

Preguntas para reflexionar.

1. ¿Qué diferencias se plantean a la hora de hacer un juego para PC y para Mobile? Algunos temas centrales: Controles, Flow, curvas de interacción y optimización.

Próximos pasos.

En la próxima clase reflexionaremos sobre el juego creado, compartiendo ideas, experiencias y opiniones para enriquecer el proceso de diseño y desarrollo. Será una oportunidad para identificar fortalezas, áreas de mejora y aprender de los enfoques de cada uno.


Buenos Aires
aprende
Agencia de Habilidades para el Futuro
Buenos
Aires
Ciudad