Building XR Experiences for Meta Quest 3 with Godot Engine 4.5

Adriano Venturini

Dario Cavada

SUGGESTO srl
Trento (Italy)

Our Goal

Our objective is to enable **augmented reality experiences** through **headsets and smart glasses**, using **open-source software and standards**.

Dedicated AR glasses are not yet available based on open software, we have chosen the **Meta Quest 3 headset**, which currently support the development of mixed-reality applications based on open standards.

References:

- Meta Quest 3
- Godot Engine 4.5
- Official Documentation (Godot XR Tools)

What is Extended Reality (AR/MR/XR)

Augmented Reality (AR) overlays **digital elements** like 3D objects, text, or images onto the real world.

Mixed Reality (MR): blends digital content with the physical environment, allowing virtual objects to interact with and respond to real-world spaces in real time

Extended Reality (XR) combines **AR, VR, and MR** to create immersive spaces where real and virtual blend together.

Some Use Cases - Cultural Heritage

Museums & Cultural Heritage

Goal: Enrich physical exhibitions through digital overlays.

Concept: Using **QR anchors**, artworks or artifacts overlay interactive 3D layers - reconstructions, animations, or guided storytelling.

Interaction: Visitors can explore and interact naturally using **hand tracking**.

Example: “Augmented Sculpture Gallery” - missing parts of statues digitally reconstructed and anchored via QR markers.

Industrial Training & Maintenance

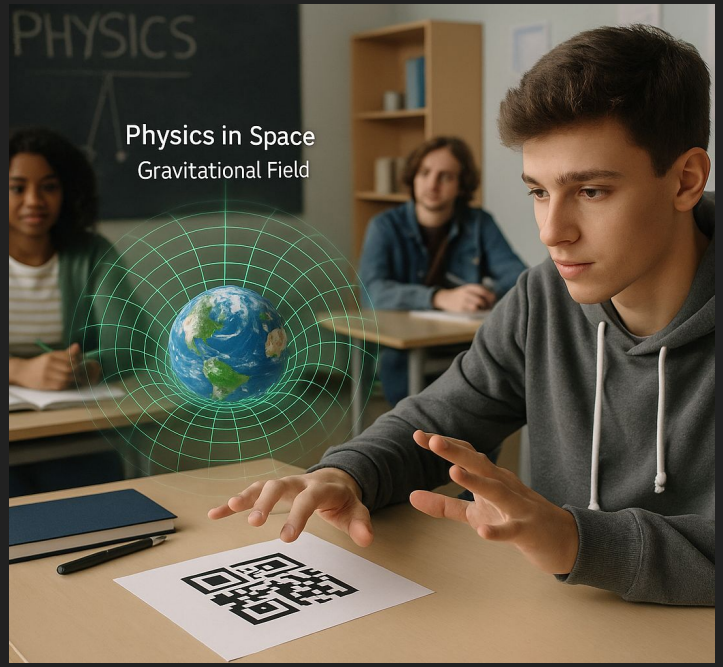
Goal: Make learning safer and more immersive in real workspaces.

Concept: Combine **spatial anchors** and **QR codes** to attach training steps or maintenance instructions directly to real machines.

Interaction: Workers can visualize, follow, and confirm each step hands-free through **Meta Quest 3**.

Example: “XR Maintenance Assistant” - real-time guidance for assembly or repairs.

Education & Science



Goal: Turn abstract lessons into tangible experiences.

Concept: Place **interactive 3D models** on desks or lab tables through **QR anchors**

Interaction: Students can manipulate models safely with **hand tracking** for better understanding.

Example: “Physics in Space” - visualize gravitational fields anchored in the classroom.

OpenXR and Android XR

OpenXR

An open standard API that provides a unified framework for XR development. It allows to build applications for multiple devices and platforms.

Already supports a wide range of headsets and systems (e.g., Meta Quest, HTC Vive, Pico, Varjo).

Android XR

Google’s operating system designed to unify XR device development. It provides a common platform for building immersive experiences and powering next-generation XR hardware.

Android XR natively supports OpenXR applications, ensuring cross-device compatibility and streamlined development workflows.

Samsung Galaxy XR

First Android XR headset built by Samsung, Google & Qualcomm.

Powered by Snapdragon XR2+ Gen 2, dual 4K micro-OLED, field of view ~109°/100°.

Runs Android XR, supports OpenXR, runs Android apps in mixed reality

Compatible with Godot Engine

SUGGESTO
SMART experience

PICO 4 Ultra Enterprise

Headset VR/MR standalone

Designed for professional use

Qualcomm Snapdragon XR2 Gen 2

2160 × 2160 pixels per eye

4K+ binocular resolution

105° field of view

Dual 32 MP HD cameras

(for improved MR/XR performance)

OpenXR support, compatible with Godot Engine

https://business.picoxr.com/

Beyond Headsets: The Future of XR Devices

[Aura Project](#) (2026)

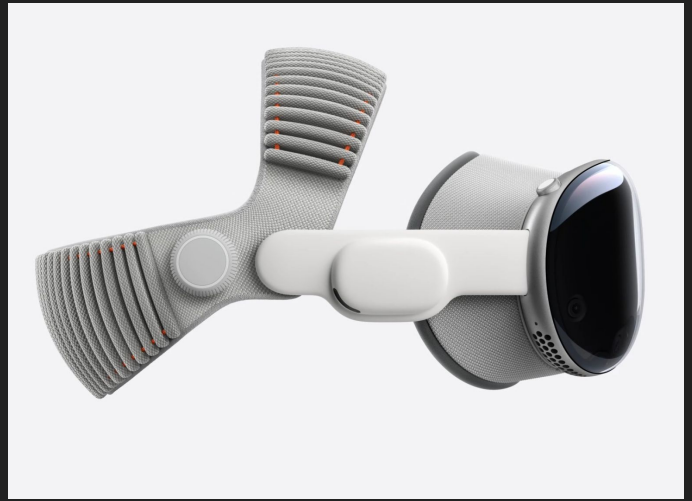
Apple and smart glasses

Apple is developing its own AR glasses, marking a major step toward mainstream adoption.

Two models are in progress: one AI-powered and lightweight, another with a built-in display.

With Apple entering the field, AR competition and innovation are accelerating, shaping the next era of mixed reality.

Apple is also contributing official VisionOS support to Godot, enabling native Vision Pro development within the open-source engine



Released the new version on the 15th of october 2025

Without display
(Camera, Mic, Speaker)

With display
(fixed in the space)

6 DoF AR
for Spatial Experiences

Link to the complete list with details

XR Frameworks Overview

Aspect	OpenXR	Meta Quest 3	AndroidXR	ARCore	ARKit
Provider	Khronos Group	Meta / Reality Labs	Google + Samsung	Google	Apple
Type	Cross-platform XR API	Standalone XR headset	XR OS for devices	AR SDK for Android	AR framework for iOS
Platforms	Multi-device / cross-vendor	Quest devices only	Upcoming XR headsets / glasses	Android phones & tablets	iPhone / iPad / Vision Pro
XR Scope	AR + VR	VR + Passthrough	AR + VR	AR only	AR only
Tracking	6DoF + standard inputs	6DoF + hand tracking	6DoF + controller support	6DoF camera tracking	6DoF + face / body tracking
Anchors & Mapping	Plane, spatial anchors standardized	Room scan + anchors	Plane, spatial (via ARCore)	Plane anchors, Cloud Anchors, Image Anchors	Plane, image anchors, object anchors, world map
Dev Tools	Godot / Unity / Unreal via API	Godot / Unity / Unreal / native SDK	Android Studio + Engines + Godot expected soon	Unity / Unreal SDK	RealityKit / SceneKit / Unity
Use cases	Cross-platform XR apps	Immersive VR / MR games	Unified XR ecosystem	Mobile AR experiences	High-quality iOS AR apps

XR SW Platform Comparison

Engines	Godot Engine	Unity	Unreal Engine	Meta Horizon Studio	Lens Studio
License/Cost	MIT, free	Proprietary, plans	Royalty-based	Free, platform-bound	Free, platform-bound
Quest Deployment	APK via OpenXR (+ Vendors)	APK via OpenXR/Meta	APK via OpenXR/Meta	In-platform (no APK)	Not for Quest APK
MR Capabilities	Passthrough, spatial anchors	Broad MR stack	High-fidelity MR stack	Social worlds, interactions	Mobile AR effects, tracking
Best for	Open, rapid prototyping	Production MR at scale	AAA visual/sims	Multiuser UGC experiences	Campaigns, try-ons, filters

MR (Mixed Reality): real and virtual worlds interacting together.

AAA Visuals / Sims: high-fidelity graphics and realistic simulation.

Why Meta Quest 3 + Godot

Meta Quest 3 supports **OpenXR**, enabling **mixed reality through passthrough cameras**, a key step toward developing experiences ready for future **AR glasses**.

With **Godot Engine 4.5**, developers can directly build and deploy XR applications for the Quest 3, including **passthrough-based AR**.

The **OpenXR plugin for Godot** is actively maintained by the **Godot XR Team** (Bastiaan Olij and others) and the **Godot Foundation**, ensuring strong community support and continuous updates.

Bastiaan Olij - Godot XR Team Lead

What is Godot

Godot is a **free and open-source game engine** for creating **2D, 3D, and XR projects**.

It's **lightweight**, **cross-platform**, and offers a **node-based system** where every element (object, light, script) fits into a clear scene structure.

It includes **GDScript**, an easy scripting language similar to Python, and a large, active community.

Why we use it:

Fast to prototype, fully customizable, and ideal for all skill levels.

Projects and Scenes in Godot

Every project in Godot is a complete application.

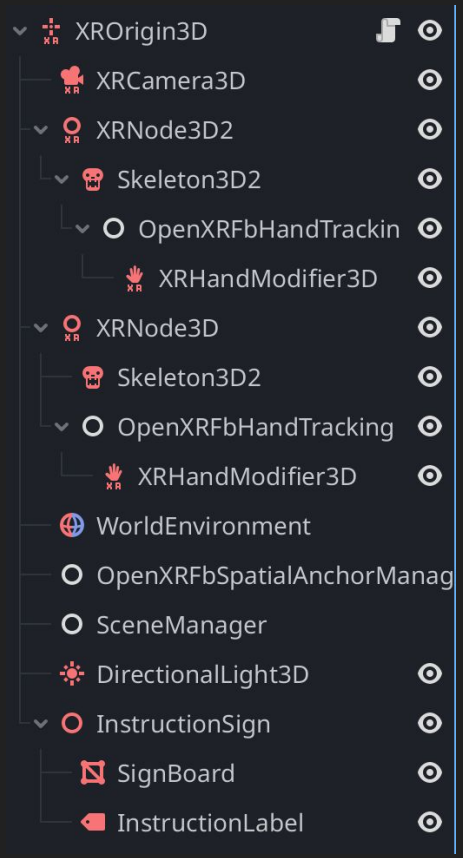
A project is composed of scenes, which are made of nodes.

Each scene can represent an object, a character, or even a complete environment.

Scenes can be nested or reused, allowing you to maintain a clean and modular structure.

What is a node?

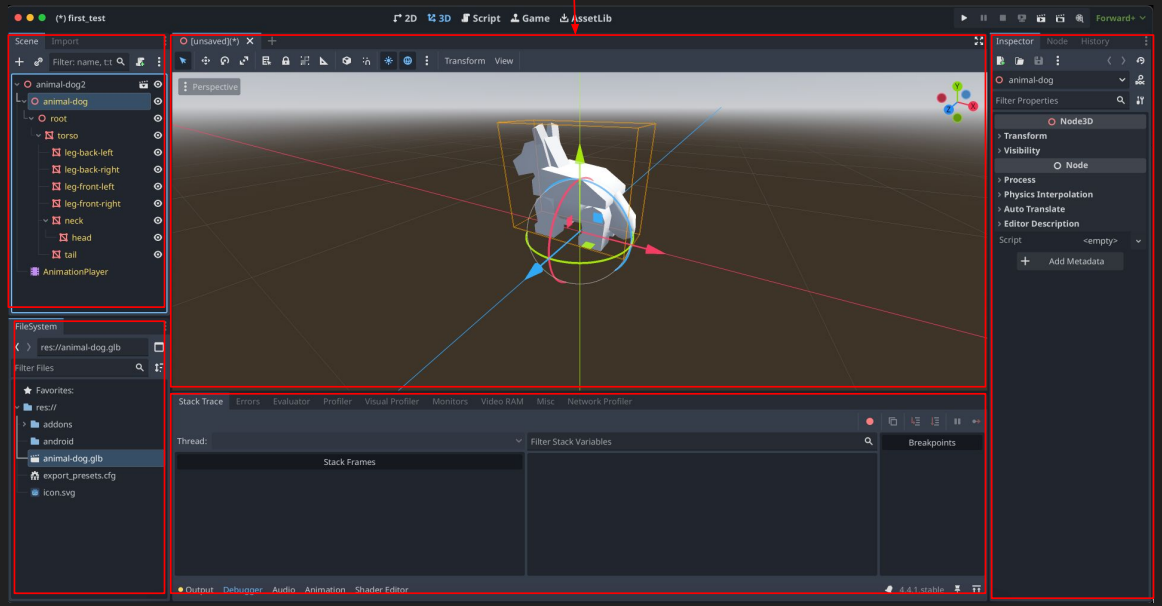
- A **Node** is the **fundamental unit** inside a scene.
- Each Node has a specific role, for example a model, light, camera, or script.
- Nodes are organized in a **tree structure**, making scenes clear and modular.
- In **XR**, Nodes represent **devices or spatial elements** like hands, controllers, or anchors, acting as the **bridge between OpenXR and the virtual world**.



Interface

1.Viewport: the main area where you **see and edit the XR environment**.

2.Scene Dock: where you **build the spatial scene**, adding and organizing nodes (headset, controllers, anchors, 3D objects).

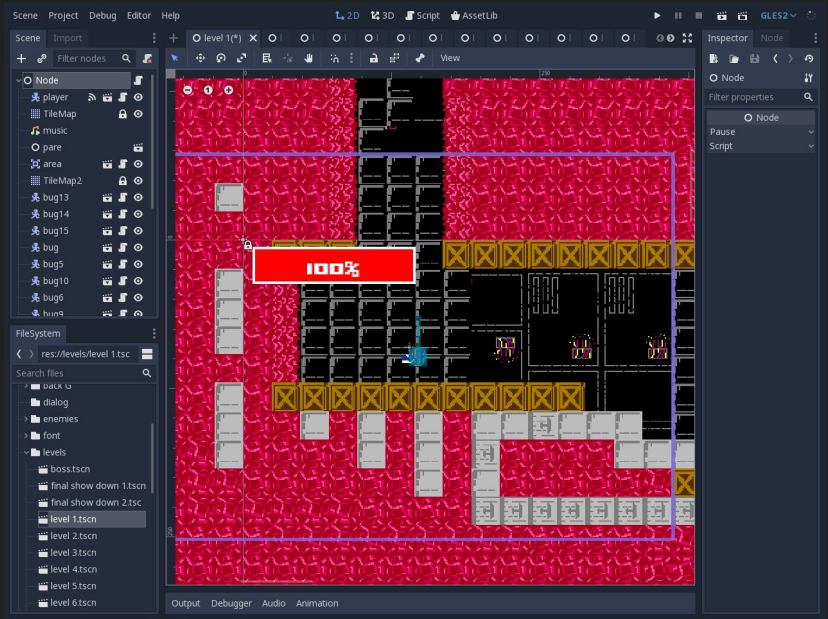


3.FileSystem: project file management (assets, scripts,addons, like XR plugin).

4.Inspector: shows and lets you **modify properties** of the selected node (e.g., tracking origin, controller input, passthrough).

Bottom Panel: area for **error messages, XR logs, performance metrics, or level editors, animation editor**.

2D Development in Godot



Godot makes it easy to create **2D games and interfaces** in the same environment used for 3D.

It's perfect for **platformers, mobile games, UI design, or educational projects**, thanks to its optimized **physics and animation tools**.

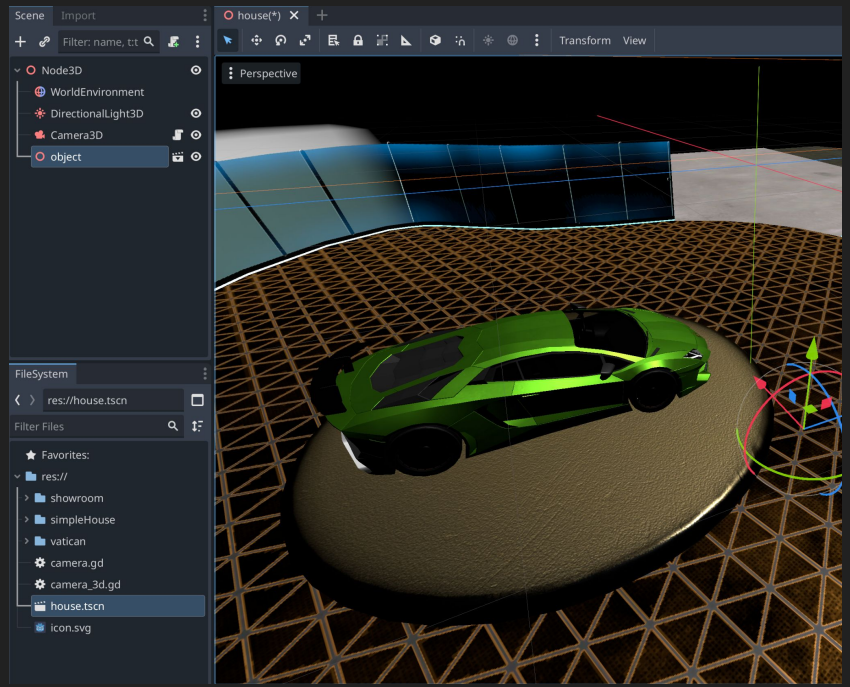
Scenes are light, fast, and easy to build using **sprites, tilemaps, and controls**.

3D Development and XR

In **3D**, Godot supports **models, lighting, shaders, physics, and animations** to create immersive worlds.

The same engine powers **XR development**, where **3D defines the spatial environment**, the world the user moves in, while **2D is used for interfaces and overlays** inside that space.

You can combine both seamlessly in one scene.



3D Model Support in Godot

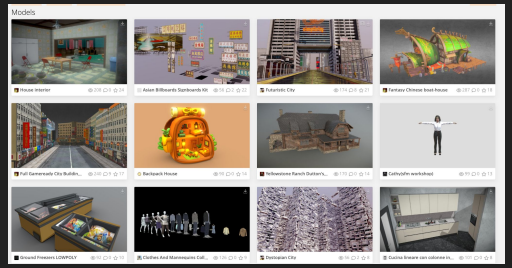
- Godot natively supports **common 3D formats** such as **GLTF**, **FBX**, **OBJ**, and **DAE**.
- **GLTF** is the **recommended format** for XR, fast loading, optimized for real-time rendering, and fully compatible with materials and animations.
- **Blender integration** is seamless: models can be exported directly to GLTF and imported into Godot without conversion.

SUGGESTO
SMART experience

External assets can be used from platforms like

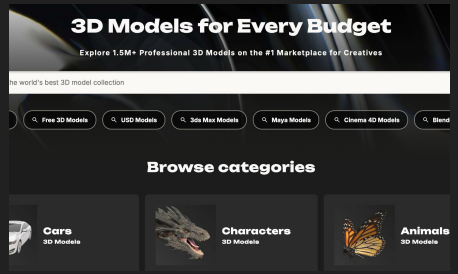
Sketchfab

3D model sharing



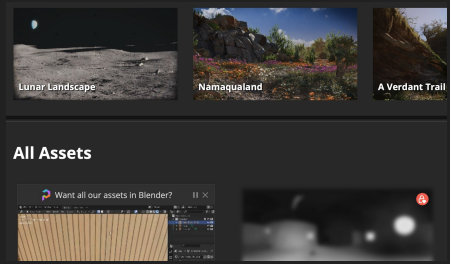
TurboSquid

Premium 3D marketplace



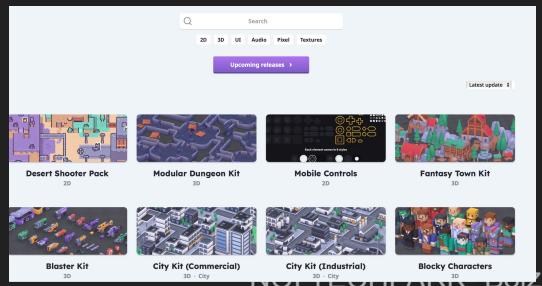
Poly Haven

Free HDRI & assets



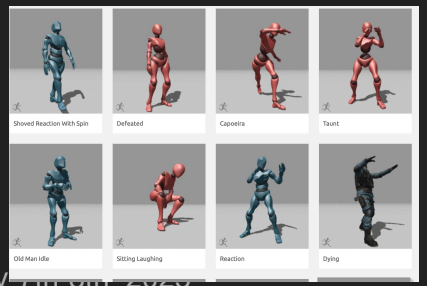
Kenney

Free game assets

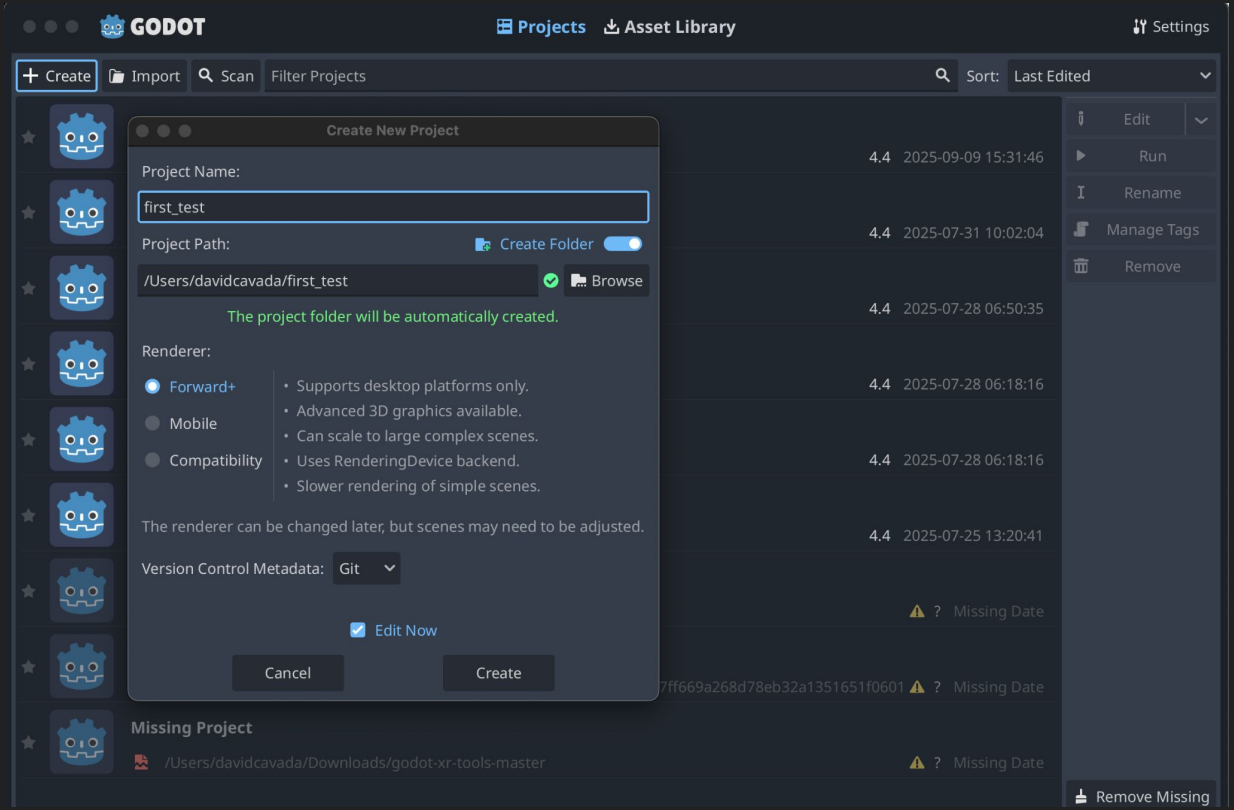


Mixamo

Auto-rigged 3D animations

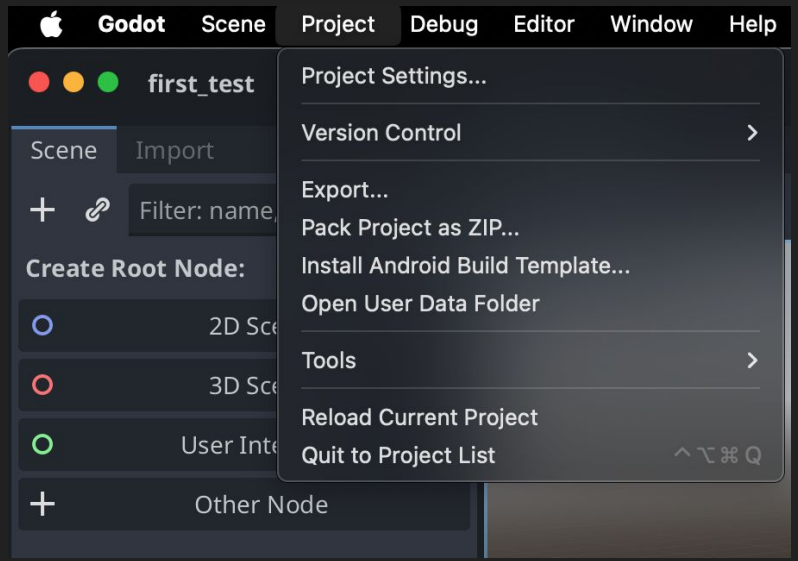


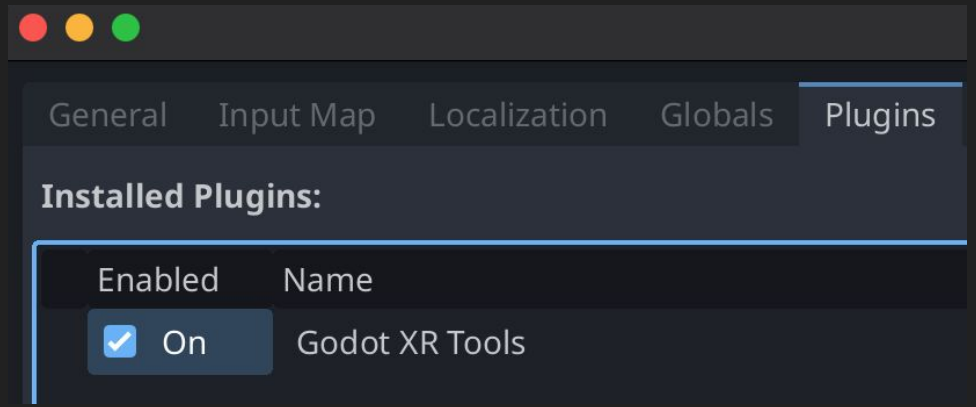
Instructions to create a Project



1. Download our basic demo (available in github)
   a. You have to add the OpenXR Plugin and Vendors Plugin (see readme.MD)
   b. Enable the plugin by Projects > Project Settings > Plugins -> Enable Plugin

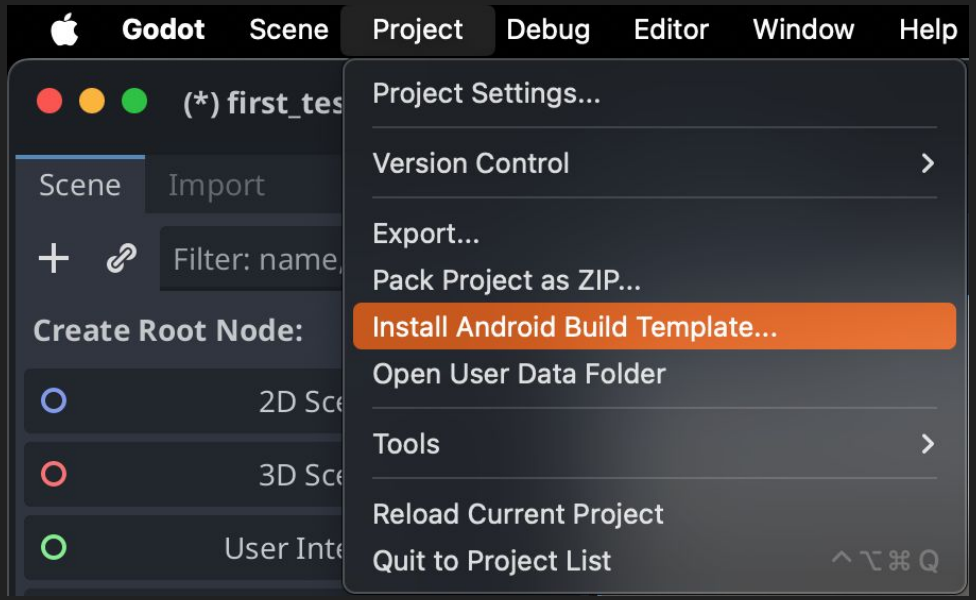
OpenXR plugin → enables XR support in Godot.





Install the Android compilation module

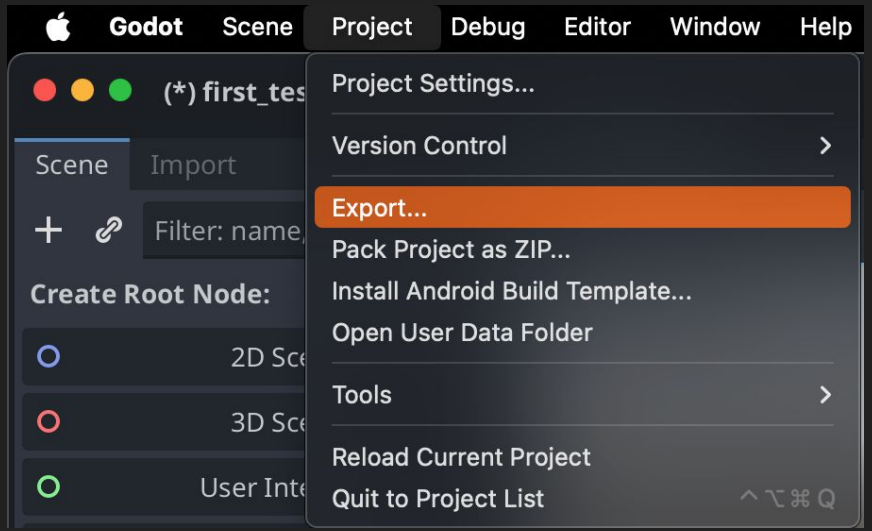
The **Android build module** is required because the **Meta Quest 3 runs on Android**, allowing Godot to **compile and export** the project to the headset.

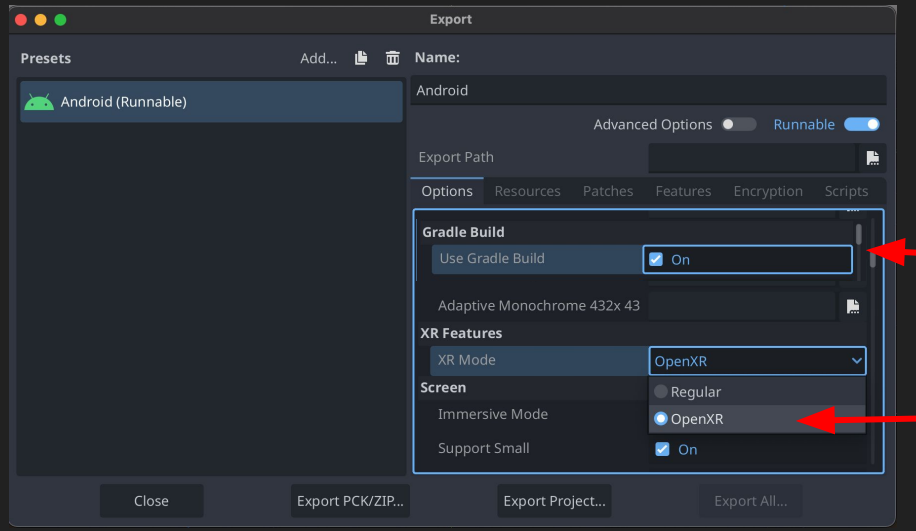


Enable OpenXR mode

Project > Export > Android -> XR Mode > OpenXR

-> Gradle Build > Use build Gradle

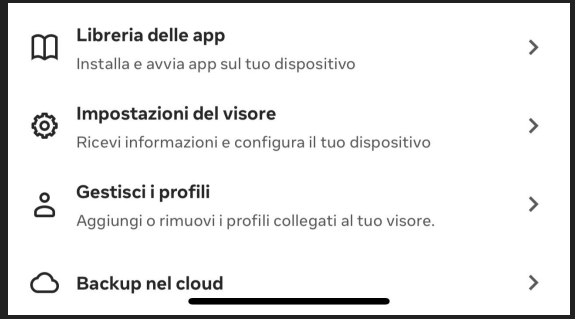


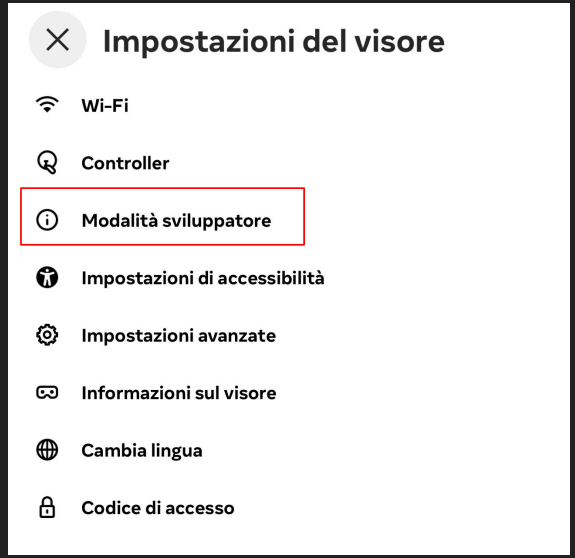


Connect the Meta Quest 3

Setting Up Developer Mode

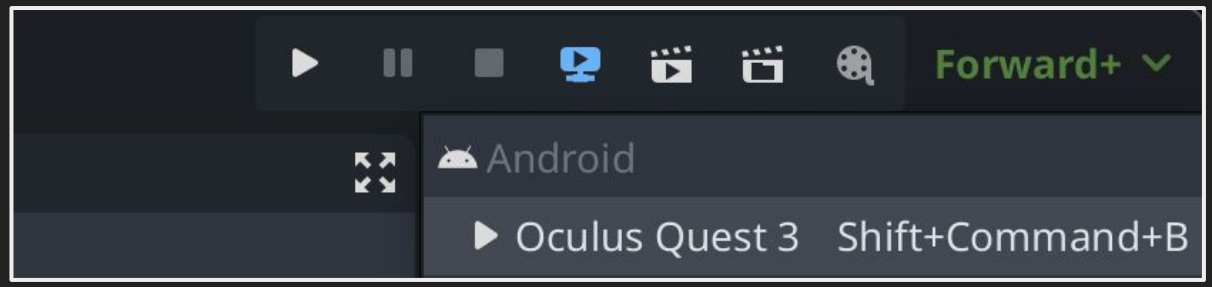
1. Open the **Horizon app** on your phone.
2. Go to **Devices → Developer Mode**.
3. **Enable Developer Mode** to allow testing and app deployment.
4. Restart the headset if needed.





Connecting the Meta Quest 3 to Your Computer

1. Use a **USB-C cable** to connect the headset to your computer.
2. Inside the headset, **allow USB debugging** when prompted.
3. Godot will automatically detect the device once permissions are granted.
4. You can now **run or export** the XR scene directly to the Quest.



There is a chance that the OpenXR plugin may be blocked due to being an unverified component.
To fix this → Go to **Device System Settings > Privacy & Security** and allow the use of the plugin.

Supported Vendors

OpenXR provides a **common standard** for XR devices, but each manufacturer (*vendor*) can add **custom features** beyond the standard.

The **Godot OpenXR Vendors Plugin** extends Godot's OpenXR support to include these **vendor-specific features**, such as:

- **Meta**: passthrough, dynamic resolution, hand tracking
- **HTC**: eye tracking, controller extensions
- **Pico / Magic Leap**: additional input and environment data

This plugin ensures **compatibility and advanced functionality** across multiple devices, including **Meta Quest, Pico 4 Ultra, HTC Vive XR Elite**, and **Magic Leap 2**.

Supported Anchors on Meta Quest 3 (Godot + OpenXR)

- **Spatial Anchors**: fix virtual objects to real-world positions; supported through Meta's OpenXR extension.
- **Plane Anchors**: experimental; detect flat surfaces such as tables, walls, or floors.
- **Hand / Controller Anchors**: track user hands or controllers for interaction and input.
- **Scene Anchors**: conceptually supported via vendor extensions, but not yet fully integrated in Godot's OpenXR plugin.

XR Godot Limitations

Some XR features are **not natively supported** and require **custom extensions or external SDKs**:

- **QR Code Anchors**: not available in OpenXR or Meta SDK; require computer vision integration.
- **Image Anchors**: no native API for image-based tracking in Godot or Quest.
- **Object Anchors**: would need custom 3D recognition or ML-based tracking.
- **Visual Recognition**: possible only through external AI or CV frameworks.
- **Chatbot / LLM Integration**: achievable via APIs, but not part of XR/OpenXR.

These features can be implemented through custom plugins or hybrid pipelines combining Godot with AI or computer-vision modules.

Demo

Basic demo with object pick up and rendering

Basic configuration of the XR environment to enable direct interaction with virtual objects.

Example of a GLB character model with animation imported from Mixamo, demonstrating movement within the scene.

Use of asset packs from https://kenney.nl/, showing how objects are positioned and rendered correctly in 3D space.

Live scene editing: as objects are moved or adjusted in the Godot editor, the changes are reflected in real time on the Meta Quest without needing to recompile.

Basic Demo: Hand Tracking & Object Interaction

User **hand tracking** allows direct interaction with virtual objects.

In this demo, a **simple 3D ball** reacts to the user's hands and you can place the ball in the environment

WORK IN PROGRESS

Our Demos on github will be available in the future

We are currently finalizing additional demos, which will be made available on GitHub, including:

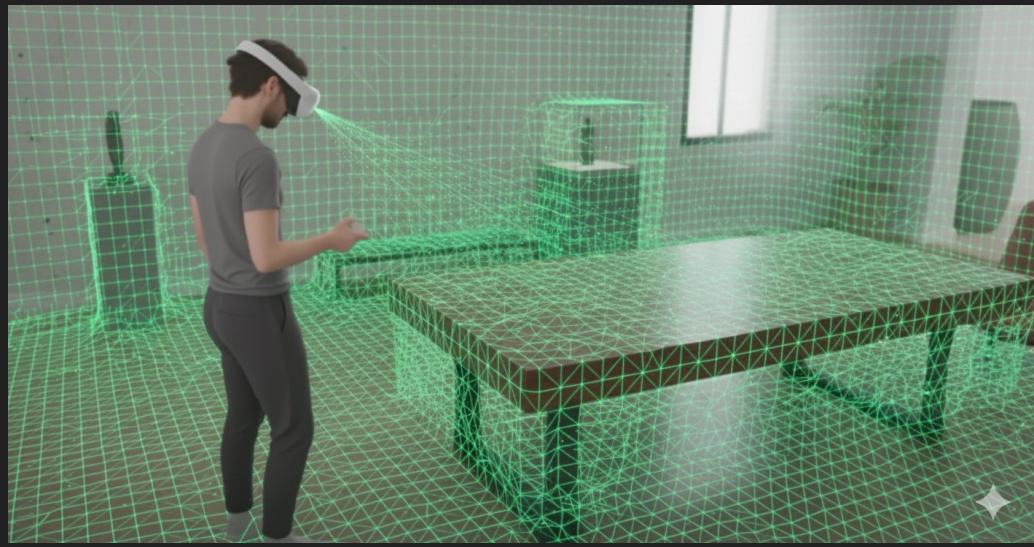
- **Room Scanning** and spatial understanding with hand tracking and gesture-based interaction for object Interaction
- **Visual Anchors** using a QR-Based Anchoring Plugin
- **Camera Server Pipeline**: capturing camera frames, sending them to a server, and receiving back scene descriptions or object recognition results
- More demonstrations and tutorials will be released in the near future.

Room Scanning, Hand Tracking & Object Interaction

(You can try it at our stand in the Crane Hall)

Using the **Meta OpenXR Vendor extension**, we scan the physical room and retrieve **3D meshes** representing walls, floors, and objects.

These meshes form a **virtual replica of the environment**, enabling realistic spatial awareness inside Godot.



SUGGESTO
SMART experience

VisualAnchors: QR-Based Anchoring Plugin

(You can try it at our stand in the Crane Hall)

Developed by our team, **VisualAnchors** brings **QR-code tracking and anchoring** to **Godot 4.5 on Meta Quest 3**.

It uses the headset's **camera passthrough** to detect markers in real time and place virtual objects **exactly where the QR codes are**.

VisualAnchors Plugin (in development): How It Works

- Built as a **native Android plugin** using **Camera2** and **ZXing** for QR detection.
- Computes each marker's **pose** (position + rotation) directly from the camera feed.
- Sends detections to Godot via a **singleton**, wrapped by VisualAnchorsManager.
- Developers can **bind any 3D node** to a QR payload to keep it aligned in real space.
- Supports **smoothing**, **offsets**, and **locking** for stable tracking.
- Currently available for **Meta Quest**, as it relies on **Camera2 passthrough access**, but it can be extended to other XR devices by integrating their **native video streams**.
- Enables **reliable visual anchoring**, a feature **not natively available** on Meta Quest 3.

SUGGESTO
SMART experience

Camera Server

(Work in progress)

Demo Overview

This demo demonstrates how to use the **Meta Quest camera API** to create **personalized experiences**.

- Users send an image to a server, by double pinch.
- The server responds with **a description of the image using AI**.

Server

Examples of Servers That Can Be Used

Local server options:

- **Moondream** (Visual Language Model): Generates a textual description of what the user is currently seeing.
- **DinoV3** embeddings: Recognizes specific images (e.g., paintings or artifacts) and returns contextual descriptions.

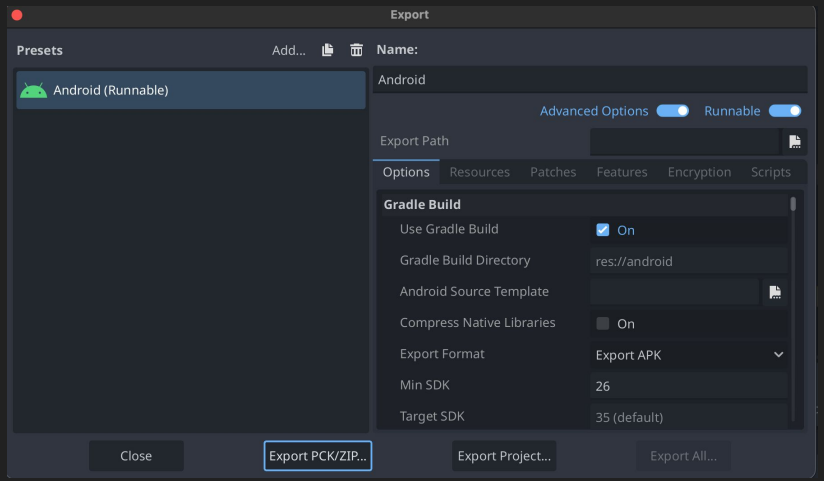
Cloud-based options:

- **Gemini** 2.5 Flash
- **OpenAI** Vision / Multimodal
- **Moondream** Cloud
- (and others depending on deployment requirements)

Custom / Sideload Deployment from Godot

Steps:

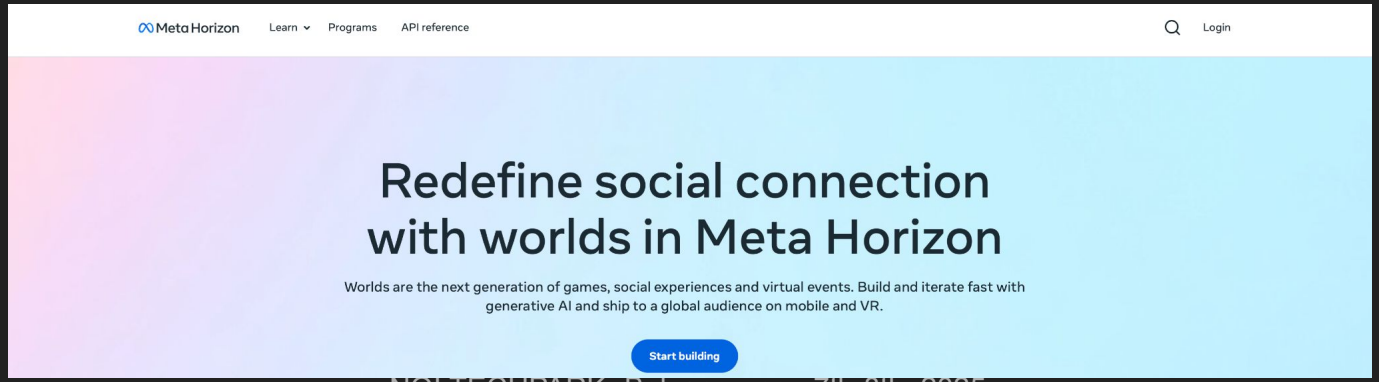
1. Enable **Developer Mode** on Quest (via Meta app) and allow **Unknown Sources**.
2. Connect the Quest to your PC via **USB-C** cable.
3. Export your Godot project as **Android (APK)**, targeting the headset.
4. Install the APK manually (via **adb**, SideQuest, or file transfer).
5. The app appears under **“Unknown Sources”** in the Quest menu.



Publishing to Meta Horizon Store / App Lab

Steps:

1. Enable **developer account / organization** in Meta Dashboard.
2. Prepare your APK: signed, optimized, meets **VRC (Virtual Reality Checks)**.
3. Upload the build via **Meta Quest Developer Hub** or CLI.
4. Choose **Release Channel / App Lab / Public listing**.
5. Submit for review: Meta checks stability, performance, permissions.



Additional Use Cases

- **QR Code Anchors**: place content relative to scanned codes.
- **Image Anchors**: attach objects to recognized 2D images.
- **Object Anchors**: lock content to known 3D objects.
- **Visual Recognition Anchors**: use ML to dynamically anchor to real-world features.
- **Chatbot with LLM**: augment XR scenes with conversational agents that respond to context.

LET'S STAY IN TOUCH!

Source code coming soon on GitHub

Contact us:

https://www.linkedin.com/in/adrianoventurini/

https://www.linkedin.com/in/dariocavada/

Scan the QRcode for the presentation and our contacts


Thank
You