VRinVET

Co-funded by the European Union

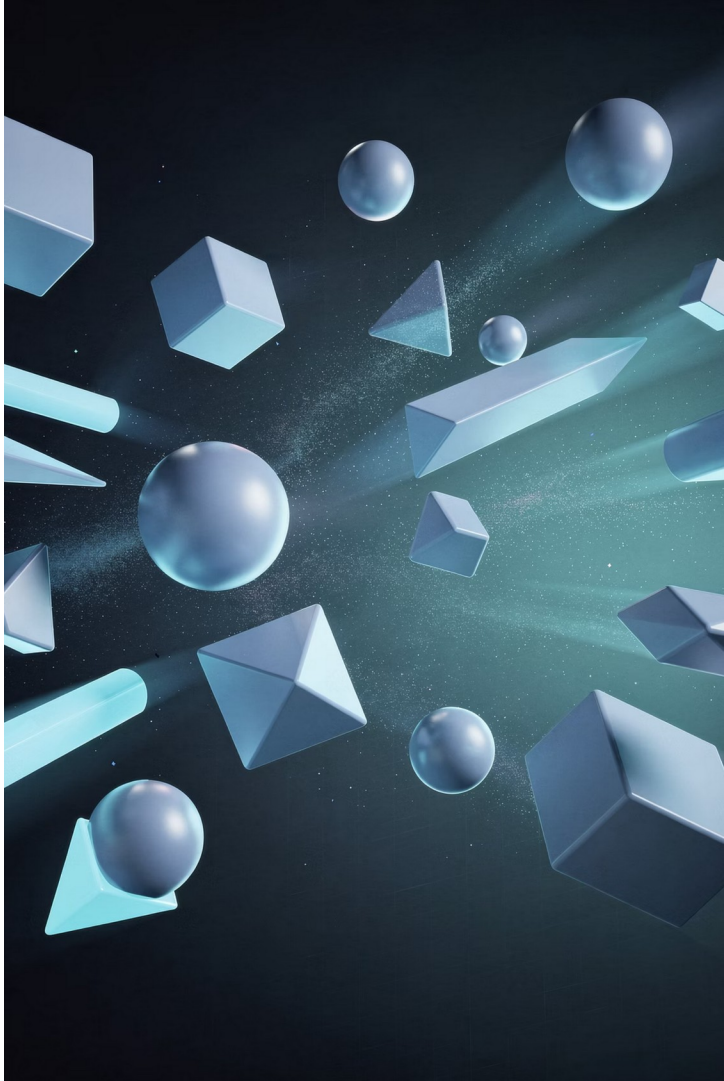
Three.js: Bringing 3D to Life

One of the world's most popular JavaScript libraries for creating immersive 3D graphics, WebXR applications, browser-based VR, interactive simulations, and educational visualizations — all running directly inside a web browser.

THREEJS.ORG

WEBGL · WEBGPU · JAVASCRIPT

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.



What Is Three.js?

The Big Idea

Three.js is an open-source JavaScript library that sits on top of WebGL and WebGPU, giving developers a high-level, approachable framework for building 3D scenes, animations, lighting, cameras, materials, and immersive environments — all without writing a single line of raw GPU shader code.

The Problem It Solves

Without Three.js, developers must write complex, verbose low-level WebGL code to render even the simplest 3D object. Three.js abstracts that complexity away, making browser-based 3D development accessible to a much broader audience of web developers and designers.

Open-source · Free · Community-supported · 100K+ GitHub stars

Why Three.js Matters

Three.js democratized browser-based 3D by making it **accessible, scalable, interactive, and cross-platform**. It is now a foundational technology across a wide range of industries and application types.

WebXR & Immersive Web

Browser-based VR and AR without app installs.

Digital Twins

Real-time visualization of machines and factories.

Education & VET

Interactive simulations and virtual laboratories.

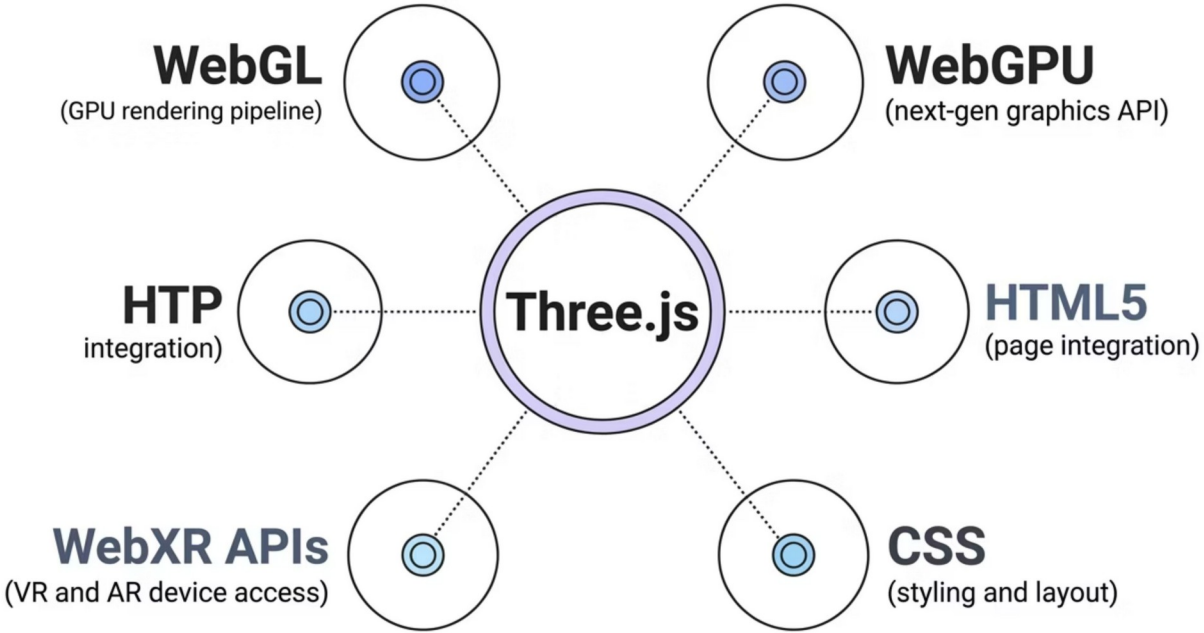
Metaverse Platforms

Persistent virtual worlds and social spaces.

Under the Hood

Core Technologies Behind Three.js

Three.js acts as an intelligent bridge between JavaScript developers and the GPU-accelerated rendering power built into every modern browser. It coordinates multiple web standards simultaneously to deliver smooth, high-fidelity 3D experiences.



Main Features of Three.js

3D Rendering

Real-time 3D graphics via WebGL and WebGPU with support for shadows, reflections, and post-processing effects.

Cameras & Scene Management

Perspective and orthographic cameras, fog, environment maps, and full virtual environment organization.

Lighting Systems

Directional, point, spot, and ambient lights create realistic illumination and dynamic shadows.

Materials & Textures

PBR materials, UV mapping, bump maps, and environment textures for photorealistic surfaces.

Animation Systems

Keyframe, skeletal, and morph target animations for dynamic objects and characters.

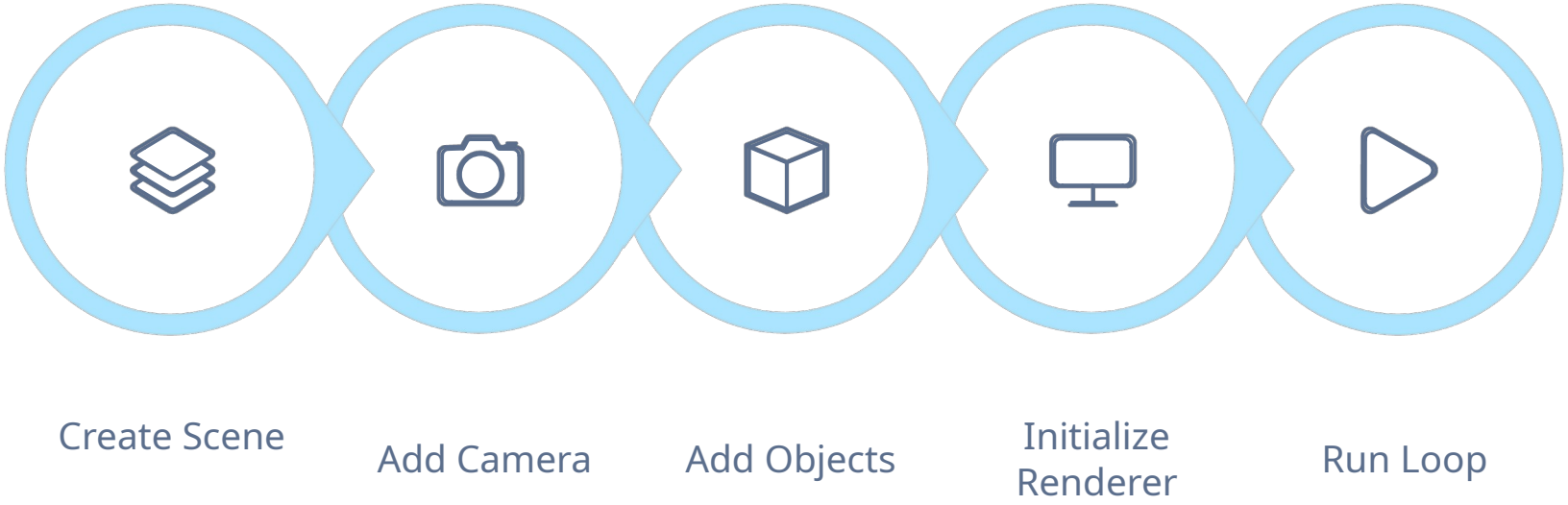
WebXR Integration

First-class support for browser-based VR and AR experiences across headsets and mobile devices.

Architecture

Basic Three.js Application Structure

Every Three.js project follows a clear, predictable architecture. Understanding this core structure is the foundation for building anything — from a simple rotating cube to a full-scale WebXR environment.



The **animation loop** is the heartbeat of any Three.js app — it calls `renderer.render(scene, camera)` on every frame, typically at 60fps, keeping your 3D world alive and responsive to user input.

Three.js and WebXR

Three.js has first-class, production-ready support for the **WebXR Device API**, enabling fully immersive VR and AR experiences delivered directly through a web browser — no app store, no download, no installation required.

Supported Headsets

Meta Quest, HTC Vive, Pico, and any OpenXR-compatible device.

Mobile AR

Smartphone-based immersive experiences via WebXR on Android and iOS browsers.

Desktop Browsers

3D previews and non-immersive XR fallbacks for standard desktop viewing.

Three.js in VR & AR Development

Virtual Reality

Three.js powers fully immersive VR environments accessible via browser. Developers build virtual training labs, educational simulations, industrial safety courses, virtual museums, and serious games — all deployable with a single URL.

- Industrial safety training
- Engineering simulations
- Virtual classrooms & labs
- Immersive virtual museums

Augmented Reality

Three.js also enables browser-based AR using WebXR's hit-testing and image-tracking capabilities. Digital content overlays the real world on mobile and compatible desktop browsers — no native app required.

- Product visualization
- Educational overlays
- Maintenance instructions
- Interactive learning systems

Education Focus

Three.js and Vocational Education & Training

Three.js is a natural fit for **VET institutions** because it eliminates the hardware and software barriers that make traditional XR prohibitively expensive. Learners access immersive simulations through any browser — no headset required to start.

CNC Simulations

Safe, repeatable machine operation practice before touching real equipment.

Electrical Systems

Interactive circuit visualization and fault diagnosis training.

Smart Factory

Digital twin interaction and Industry 4.0 process understanding.

Machine Guidance

Step-by-step 3D operation walkthroughs for complex industrial equipment.

Three.js and Digital Twins

A **digital twin** is a real-time virtual replica of a physical asset — a machine, a production line, a factory floor, or an entire smart infrastructure system. Three.js provides the 3D visualization layer that makes digital twins intuitive and interactive for operators and engineers.

By connecting Three.js to live IoT data streams, developers can render machine behavior, track sensor readings, highlight anomalies, and simulate failure scenarios — all inside a standard web browser.

- Real-time data visualization
- Machine behavior monitoring
- IoT & SCADA integration
- Industrial process simulation

Advantages & Limitations

✅ Strengths

Open-Source & Free

Community-supported, no licensing costs.

Browser-Based

Zero installation for end users — just a URL.

Cross-Platform

Runs on desktop, mobile, and XR devices.

Large Ecosystem

Extensive docs, examples, and community plugins.

⚠️ Limitations

Requires JavaScript

Not a no-code tool — programming knowledge needed.

Graphics Complexity

Advanced scenes require careful optimization.

No Built-in Physics

Physics require external libraries like Cannon.js.

Not a Game Engine

Lacks level editors, asset pipelines, and built-in AI.

Three.js vs. Unity vs. Unreal Engine

Each platform excels in different contexts. Three.js wins on accessibility and web integration; Unity and Unreal Engine win on raw graphical power and built-in tooling for complex games and simulations.

Feature	Three.js	Unity	Unreal Engine
Platform	Browser	Native Apps	Native Apps
Language	JavaScript	C#	C++ / Blueprint
Installation	Not required	Required	Required
Graphics Power	Moderate–Strong	Strong	Very Strong
WebXR Support	Excellent	Moderate	Moderate
Accessibility	Very High	Medium	Medium

For browser-based educational and VET applications, Three.js offers the best accessibility-to-capability ratio of any platform available today.

The Ecosystem

Popular Libraries That Extend Three.js

Three.js has a rich ecosystem of companion libraries that accelerate development. These tools layer additional abstractions — from React integration to physics engines — on top of the Three.js core.

React Three Fiber

A React renderer for Three.js that lets you declare 3D scenes as JSX components. Ideal for developers already working in the React ecosystem. docs.pmnd.rs/react-three-fiber

Drei

A rich collection of ready-to-use helper components for React Three Fiber — cameras, controls, loaders, shaders, and more. Dramatically speeds up scene composition. github.com/pmndrs/drei

Cannon.js

A lightweight, pure-JavaScript physics engine that integrates with Three.js to add gravity, collisions, rigid body dynamics, and constraint systems to 3D scenes. schteppe.github.io/cannon.js

Three.js, AI, and the Metaverse

AI Integration

AI layers intelligence onto Three.js experiences — enabling adaptive learning pathways, intelligent tutoring agents, voice interaction, object recognition in AR, procedural content generation, and real-time learner analytics that personalize challenge and pacing.

Metaverse Applications

Three.js is a core rendering technology for browser-based metaverse platforms. It powers persistent virtual worlds, spatial collaboration spaces, avatar systems, and multiplayer XR environments accessible through any web browser.

Looking Ahead

The Future of Three.js

Three.js continues to evolve rapidly alongside the web platform itself. Several emerging capabilities are set to dramatically expand what's possible in the browser over the next few years.

1 WebGPU Integration

Next-generation GPU API delivering dramatically faster rendering and compute shaders in the browser.

2 AI-Enhanced XR

Intelligent agents, adaptive environments, and AI-driven procedural content embedded in immersive scenes.

3 Cloud Rendering

Offloading complex 3D computation to the cloud, enabling console-quality graphics on low-end devices.

4 Industrial Metaverse

Scalable digital twin networks and spatial computing environments for smart manufacturing and Industry 5.0.

Three.js is expected to remain a foundational technology for browser XR, immersive web systems, and scalable digital learning for years to come.

Why Three.js for Erasmus+ and VET Projects

The Strategic Case

Three.js uniquely solves the **distribution problem** of immersive educational content. Experiences built with Three.js require no software installation — they are shared as a URL, embedded in an LMS, linked from a QR code, or hosted on a project portal. This dramatically lowers the barrier for learners and institutions across international partnerships.

Distribution Methods

- 📎 Direct URL sharing
- 📱 QR code access on printed materials
- 🎓 LMS embedding (Moodle, Canvas, etc.)
- 🌐 Online portals and project websites

Key Benefits for VET

- Low-cost immersive learning at scale
- No per-seat licensing or hardware lock-in
- Multilingual and accessible by design

Key Resources & Links

Three.js Official

threejs.org — Home, docs, and live examples gallery.

WebXR API (MDN)

MDN WebXR Device API — Complete browser XR reference.

React Three Fiber

docs.pmnd.rs/react-three-fiber — React renderer for Three.js.

Babylon.js

babylonjs.com — Alternative 3D web engine with strong XR support.

A-Frame

aframe.io — HTML-based WebXR framework built on Three.js.

OpenXR Standard

khronos.org/openxr — The open standard unifying XR device APIs.