

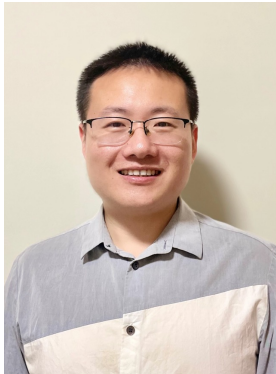


Build a brand new ecosystem with OpenXR



This work is licensed under a Creative Commons Attribution 4.0 International License

Speaker



Shuai Liu

Working for PICO as XRRuntime tech lead, currently, focusing on OpenXR runtime implementation and OpenXR API designing.

Participate in PICO / EXT OpenXR extension developing, PICO XRRuntime standardization, and XRRuntime performance optimization.

Agenda

- **Who am I?**
- **Introduction of OpenXR**
- **What can OpenXR do?**
- **OpenXR API overview**
- **OpenXR 1.1 updates**
- **What is coming soon...**
- **Our experience**

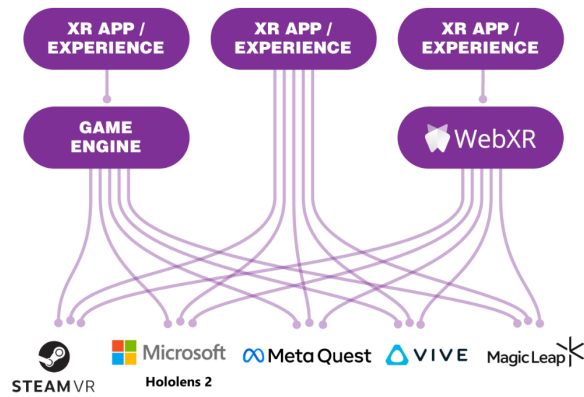
What is OpenXR

OpenXR is a set of functions that interface with a runtime to perform commonly required operations such as accessing controller/peripheral state, getting current and/or predicted tracking positions, and submitting rendered frames.

Empowering developers to create cross-platform, immersive spatial computing experiences

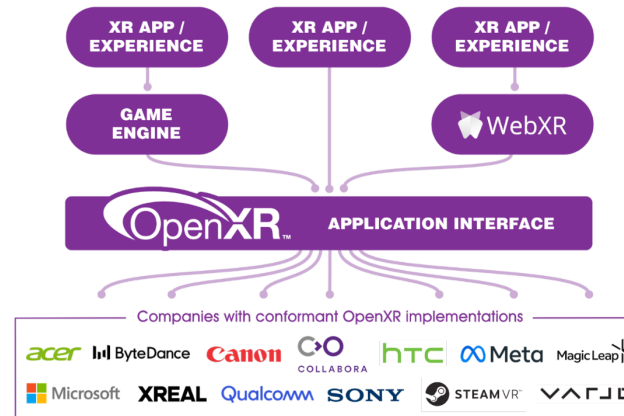


OpenXR Cross-Platform Portability




























Before OpenXR: Applications and engines needed separate proprietary code for each device on the market.

Applications and engines can portably access any OpenXR-conformant hardware












OpenXR provides a single cross-platform, high-performance API between applications and all conformant devices.

OpenXR Device Support

 	 	 
HoloLens and Mixed Reality Headsets. Hand and eye tracking extensions	Rift S, Quest 3, Quest 2 and Quest Pro Meta Deprecated own API for OpenXR	Vive Focus 3, Vive Cosmos, Vive XR Elite, Vive Wave Runtime
 	 	 
Valve Index Valve Deprecated OpenVR APIs for OpenXR	All Varjo Headsets are fully compliant XR-3, XR-4	MREAL X1
 	 	 
Magic Leap 2	XREAL Air 2, Air 2 Pro, Air 2 Ultra	Qualcomm Snapdragon Spaces XR Development Platform
 	  	 
Spatial Labs Display Series	Neo 3 and Pico 4	Spatial Reality Displays

This work is licensed under a Creative Commons Attribution 4.0 International License

Engines, Browsers, and Libraries with OpenXR

		
<p>Unreal has been providing support since 4.24. UE 5.0 supports OpenXR</p>	<p>Unity's OpenXR plugin available since 2020 LTS</p>	<p>Godot provides OpenXR support since March 2023 (Core 4.0 Alpha 4)</p>
		
<p>OpenXR supported since VRED 2023.4</p>	<p>NVIDIA Omniverse and CloudXR Platforms</p>	<p>WebXR in Chrome, Edge, and Firefox uses OpenXR as the default backend</p>
		
<p>Open-source OpenXR Implementation</p>	<p>A lightweight XR Meta XR Simulator to Speed Unity OpenXR Development</p>	<p>Autodesk open-source mixed reality library for building HoloLens and VR applications</p>

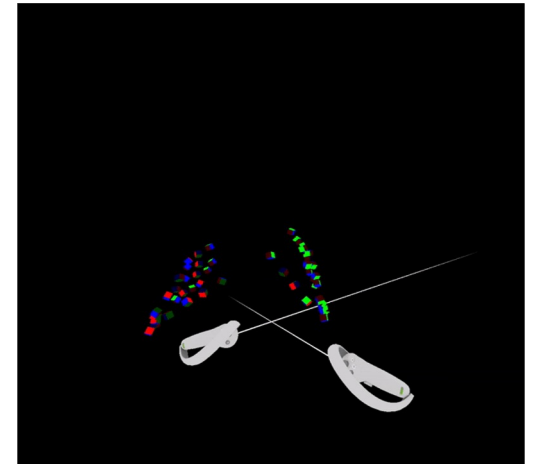
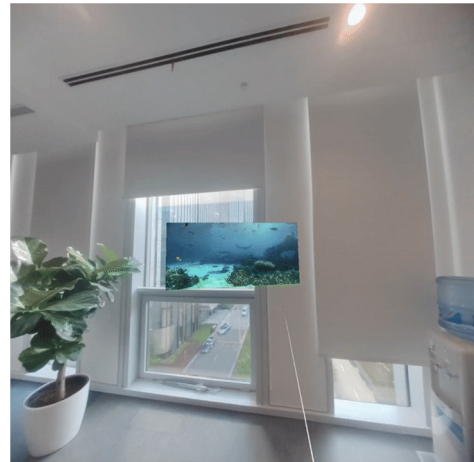
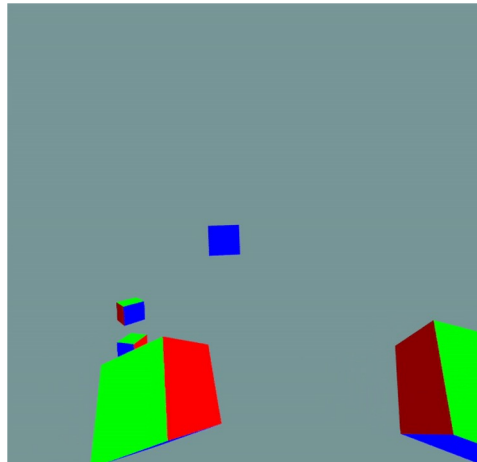
OpenXR Games and Apps

				
Blender uses OpenXR for native scene inspection in VR	Adobe Substance 3D Modeller uses OpenXR for VR support	Kitware's Paraview uses OpenXR for VR support	Meta Horizon Workrooms	OpenBrush uses OpenXR for Desktop and Quest support
				
War Thunder now uses OpenXR	Cubism uses OpenXR for VR support	Vermillion uses OpenXR for VR support	The Light Brigade uses OpenXR for VR support	XPlane12 uses OpenXR for VR support
				
Minecraft uses OpenXR for desktop VR support	Microsoft Flight Simulator uses OpenXR for VR support	Supports over 27 devices thanks to OpenXR	Phasmophobia switched from OpenVR to OpenXR	Beat Saber Alpha branch uses OpenXR

This work is licensed under a Creative Commons Attribution 4.0 International License

What can OpenXR do?

- View
- Input
- Haptics
- Layer
- Passthrough
- Extensions
- Hand Tracking
- Eye Interaction
- Face Tracking

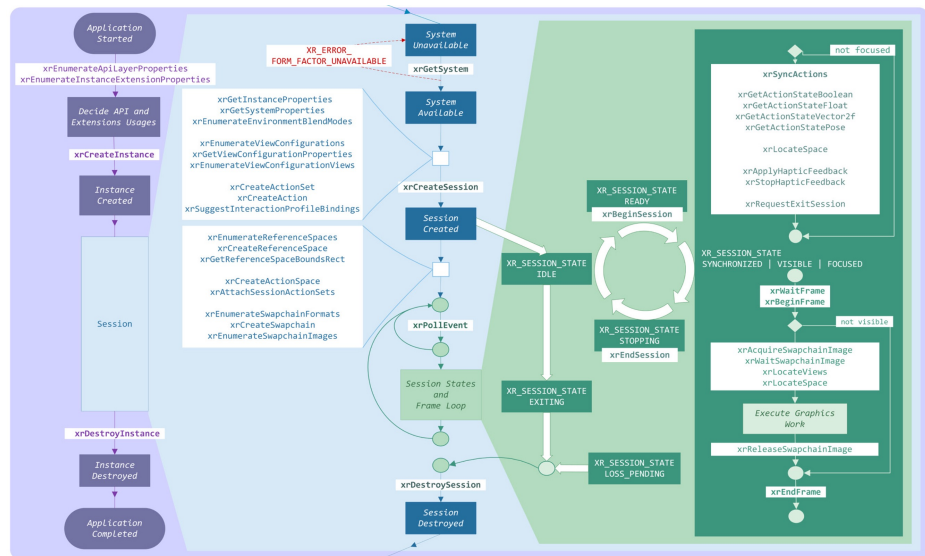


OpenXR1.0 concept

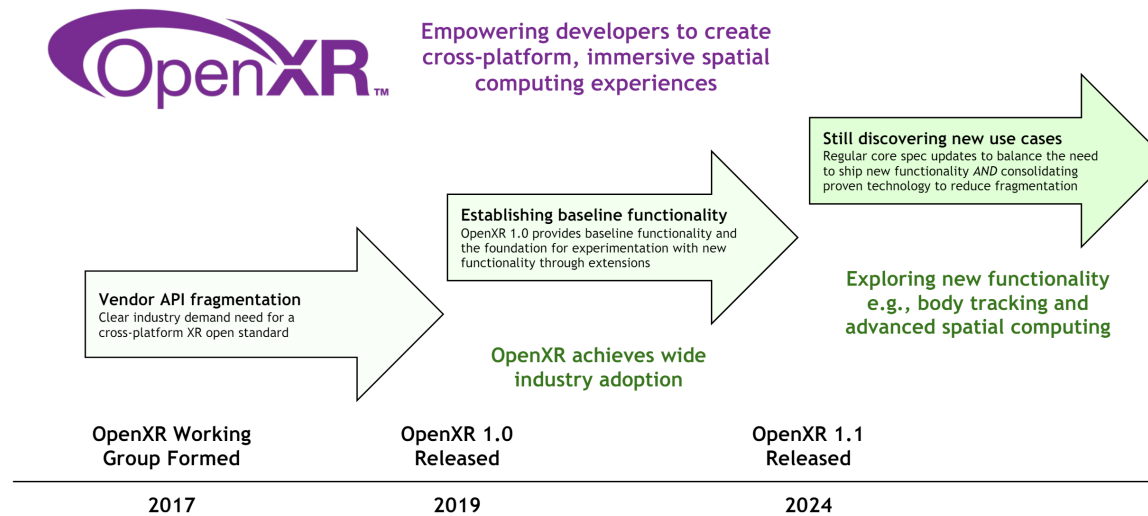
- **Extensions**
 - **xrEnumerateInstanceExtensionProperties**
- **Instance**
- **Session**
- **Interaction Profile**
 - `/interaction_profiles/vendor_x/profile_x`
 - `.../input/<identifier>[_<location>][!/<component>]`
 - `.../output/<output_identifier>[_<location>]`
- **ActionSet**
 - **xrSyncActions**
 - **xrGetActionState***
- **Reference Space**
 - XR_REFERENCE_SPACE_TYPE_VIEW
 - XR_REFERENCE_SPACE_TYPE_LOCAL
 - XR_REFERENCE_SPACE_TYPE_STAGE
 - **xrLocateSpace**
- **View Configuration**
 - XR_VIEW_CONFIGURATION_TYPE_PRIMARY_MONO
 - XR_VIEW_CONFIGURATION_TYPE_PRIMARY_STEREO
 - **xrEnumerateViewConfigurations**
- **FrameLoop**
 - **xrWaitFrame**
 - **xrBeginFrame**
 - **xrEndFrame**
- **Event**
 - **xrPollEvent**
- **Swapchain**
 - **xrEnumerateSwapchainFormats**
 - **xrCreateSwapchain**
 - **xrAcquireSwapchainImage**
 - **xrWaitSwapchainImage**
 - **xrReleaseSwapchainImage**

OpenXR API Overview

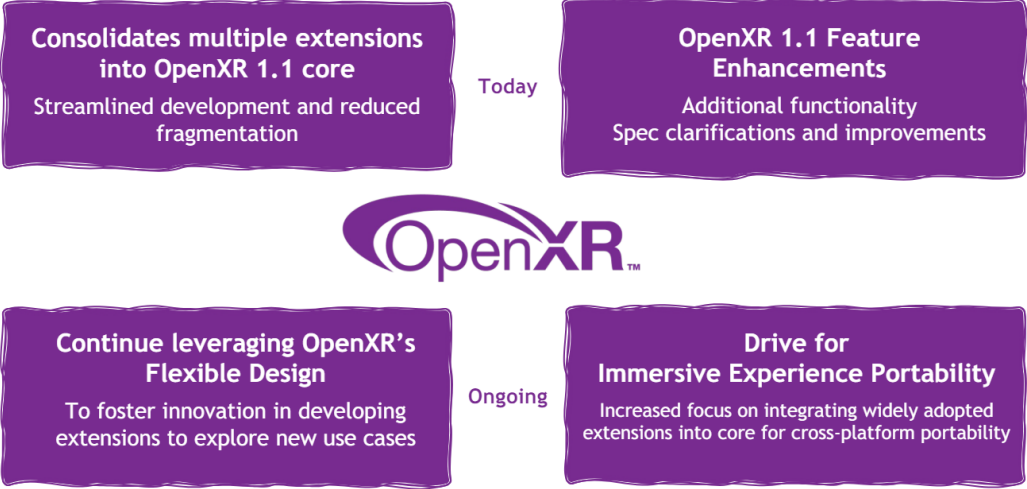
- **Get started**
 - Create Instance
 - Choose extensions, layers, bind to graphics API
- **Find out where/how to run**
 - Get HMD characteristics - mono/stereo, form factor etc..
- **Set up your interaction/input handles**
 - Create Action Sets, Actions
 - Suggest bindings
- **Prepare your immersive experience**
 - Create Session
 - Attach action sets
 - Create Reference and Action Spaces
 - Create Swapchain
- **Participate in the frame loop**
 - handle input and haptics
 - manage swapchain to drive imagery to the display
 - poll for events



The OpenXR Story So Far...



OpenXR 1.1



OpenXR 1.1 updates

[OpenXR1.1](#) Release at 04.15.2024

- Extensions Promoted to the OpenXR 1.1 Core Specification
- Interaction Profile Improvements
- Fundamentals
- New Error Codes
- Specification Refinement
- Conformance Test Suite Enhancements



Extensions Promoted to the OpenXR 1.1 Core Specification

Local Floor (promoted from XR_EXT_local_floor)

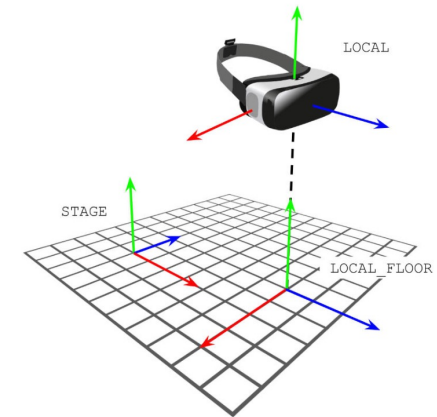
provides a new [Reference Space](#) with a gravity-aligned world-locked origin for standing-scale content that can be recentered to the current user position at the press of a button without a calibration procedure. It also has an estimated floor height built-in.

`XR_REFERENCE_SPACE_TYPE_LOCAL_FLOOR` was added in OpenXR1.1.

Stereo with Foveated Rendering (promoted from XR_VARJO_quad_views)

provides a [Primary View Configuration](#) to realize eye-tracked foveated rendering or fixed foveated rendering for XR headsets across multiple graphics rendering APIs. Its use is especially beneficial for efficiently rendering high-pixel-count displays, which put a heavy load on the GPU.

`XR_VIEW_CONFIGURATION_TYPE_PRIMARY_STEREO_WITH_FOVEATED_INSET`.



Extensions Promoted to the OpenXR 1.1 Core Specification

Grip Surface (promoted from XR_EXT_palm_pose)

provides a [Standard Pose Identifier](#) that reliably anchors visual content relative to the user's physical hand, whether the hand position is tracked directly, or inferred from a physical controller's position and orientation.

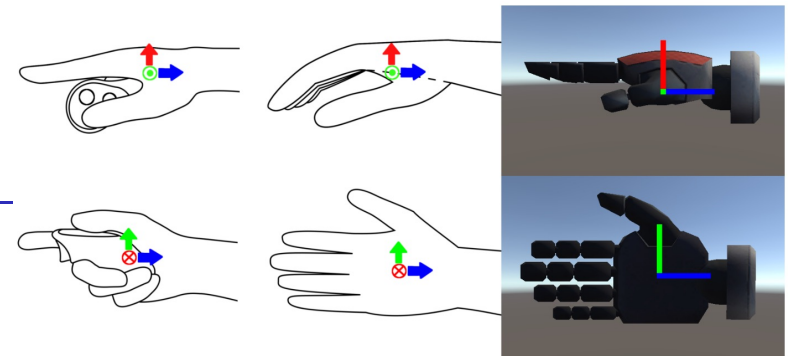
../input/grip_surface/pose

XrUuid (promoted from XR_EXT_uuid)

provides a [Common Data Type](#) to hold a Universally Unique Identifier that follows the IETF [RFC 4122](#).

xrLocateSpaces (corresponding extension equivalent XR_KHR_locate_spaces)

provides a [Locating Spaces](#) function to improve performance and simplify application code by enabling an application to locate an array of spaces in a single function call populating an "array of structures" (AoS), instead of being limited to locating a single space per function call.



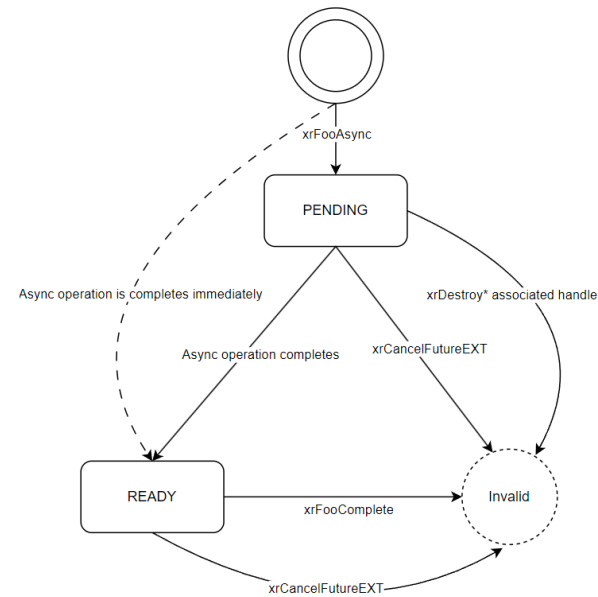
New extension

- XR_EXT_future

In XR systems there are certain operations that are long running and do not reasonably complete within a normal frame loop. This extension introduces the concept of a future which supports creation of asynchronous (async) functions for such long running operations. This extension does not include any asynchronous operations: it is expected that other extensions will use these futures and their associated conventions in this extension to define their asynchronous operations.

XrResult **xrPollFutureEXT**(XrInstance instance, const XrFuturePollInfoEXT* pollInfo, XrFuturePollResultEXT* pollResult);

XrResult **xrCancelFutureEXT**(XrInstance instance, const XrFutureCancelInfoEXT* cancelInfo);



Extensions(Deprecated)

- **XR_KHR_locate_spaces**
- **XR_KHR_maintenance1**
- XR_EXT_hp_mixed_reality_controller
- XR_EXT_local_floor
- XR_EXT_palm_pose
- XR_EXT_Samsung_odyssey_controller
- XR_EXT_uuid
- XR_BD_controller_interaction
- XR_HTC_vive_cosmos_controller_interaction
- XR_HTC_vive_focus3_controller_interaction
- XR_ML_ml2_controller_interaction
- XR_MND_swapchain_usage_input_attachment_bit
- XR_OCULUS_android_session_state_enable
- XR_VARJO_quad_view

Feature Enhancements

Interaction Profile Paths: 13 new interaction profiles have been added to the OpenXR 1.1 core specification, most promoted from vendor extensions

XR_BD_controller_interaction:

/interaction_profiles/bytedance/pico_neo3_controller
/interaction_profiles/bytedance/pico4_controller
/interaction_profiles/bytedance/pico_g3_controller

Standard Identifiers: includes thumb_resting_surfaces, stylus, trigger_curl, and trigger_slide.

Standard Component: adds proximity.

Output Paths: haptic_trigger and haptic_thumb



Fundamentals

- Improvements in OpenXR 1.1 provide developers with an extended set of universal tools for building enhanced XR experiences:
- **XrDuration:**
 - Clarify behavior for a negative duration: A timeout with a duration that refers to the past (that is, a negative duration) **must** be interpreted as a timeout of [XR_NO_DURATION](#)
- **Event Polling:**
 - Clarify runtime and application behavior for polling: events are placed in a queue within the runtime.
- **Two-Call Idiom:**
 - Precise explanation for “buffer size”,
- **New Structures:**
 - Added color without alpha channel (XrColor3f), rectangular prism (XrExtent3Df), oriented sphere (XrSpheref), oriented box (XrBoxf), and frustum (XrFrustum)



New Error Codes & Specification Refinement

- New error code help with application debugging.
 - XR_ERROR_INSUFFICIENT_PERMISSIONS only used in vendor extensions
 - XR_ERROR_DEPENDENCY_NOT_ENABLED
- OpenXR 1.1 provides clearer explanations of specification intent, while removing ambiguity, and strengthening the preciseness of normative language. Affected chapters include:
 - [Spaces](#): OpenXR 1.1 adds links to Reference Spaces to easily identify them in the text.
 - [Rendering](#): the XR_COMPOSITION_LAYER_CORRECT_CHROMATIC_ABERRATION_BIT is deprecated since it is not used in any shipping runtimes
 - [Input](#): mentions of “Default Bindings” are removed and timing requirements for reading input action state are clarified
 - [Appendix](#): Updated list of contributors
 - [Versions](#): A new chapter to show promoted extensions between versions

Coming soon

- **Extending hand tracking**
 - To include full body tracking
- **Enhanced handling of spatial entities**
 - Standardized methods to interact with the user's environment
 - Support for advanced spatial computing applications
- **Expanded haptics support**
 - Support immersive experiences through PCM, vibrotactiles, and transients
- **Controller render models (glTF)**
 - Dynamically highlight pressed buttons or show menus pointing to buttons
- **Increased Accessibility**
 - Input rebinding UI: input re-binding at runtime - Use any button/interaction mapping you wish
- **Metal (Mac OS) Support**
 - Provide swapchain images on Metal

Our experience

- **Pico has supported standard OpenXR loader since rom 5.9.0**
- **Pico Unity/Unreal OpenXR Plugin**
- **More and more developers want to develop their apps with OpenXR**
- **OpenXR APIs are becoming more easy to use, more easy for portability and more flexible.**
- **OpenXR is focusing on solving development pain points.**
- **PICO will also fully support OpenXR 1.1 in our next rom version.**



More Information

- Landing page khronos.org/openxr
- Tutorial <https://openxr-tutorial.com/>
- Api registry khronos.org/registry/openxr
- Source and issue trackers github.com/KhronosGroup?q=openxr
- Learn the open source runtime gitlab.freedesktop.org/monado
- Recommend to run `hello_xr` and experiment for more low-level OpenXR examples. You can download [PICO OpenXR SDK](#) and refer to [PICO OpenXR Demos](#) for more features like hand-tracking, eye-tracking, body-tracking, passthrough, etc.
- PICO Unity OpenXR SDK, PICO Unreal OpenXR SDK, PICO Native OpenXR SDK
<https://developer.picoxr.com/resources/>



Thank you