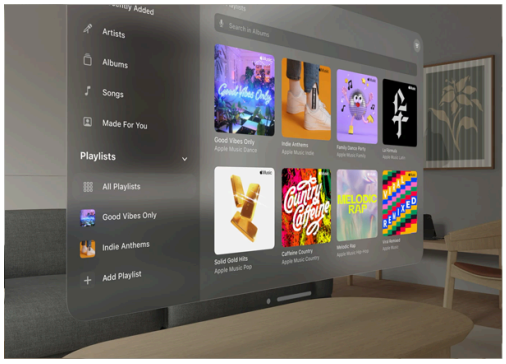
visionOS

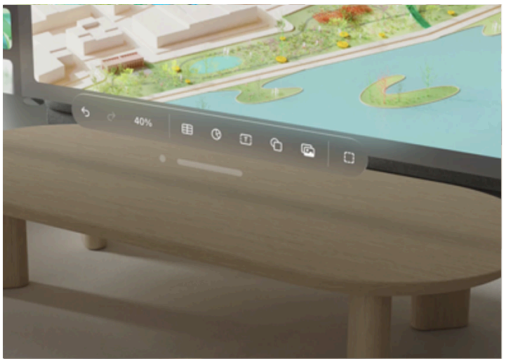
Create a new universe of apps and games for Apple Vision Pro.

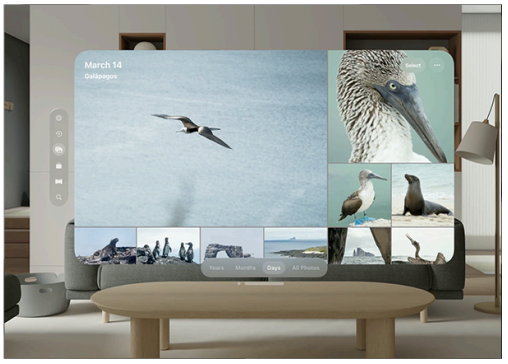
visionOS 1.0+

Overview

visionOS is the operating system that powers Apple Vision Pro. Use visionOS together with familiar tools and technologies to build immersive apps and games for spatial computing.





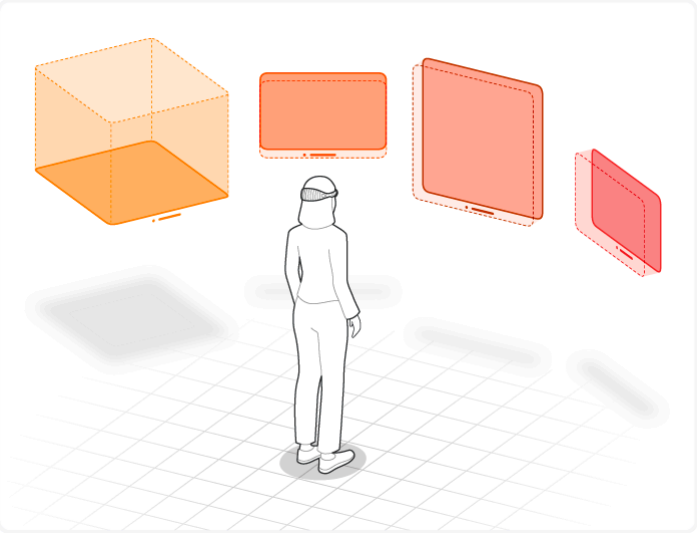


Developing for visionOS requires a Mac with Apple silicon. Create new apps using SwiftUI to take full advantage of the spectrum of immersion available in visionOS. If you have an existing iPad or iPhone app, add the visionOS destination to your app's target to gain access to the standard system appearance, and add platform-specific features to create a compelling experience. To provide continuous access to your content in the meantime, deliver a compatible version of your app that runs in visionOS.

Expand your app into immersive spaces

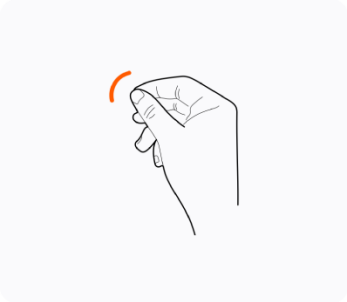
Start with a familiar window-based experience to introduce people to your content. From there, add SwiftUI scene types specific to visionOS, such as volumes and spaces. These scene types let you incorporate depth, 3D objects, and immersive experiences.

Build your app's 3D content with RealityKit and Reality Composer Pro and display it with a `RealityView`. In an immersive experience, use ARKit to integrate your content with the person's surroundings.

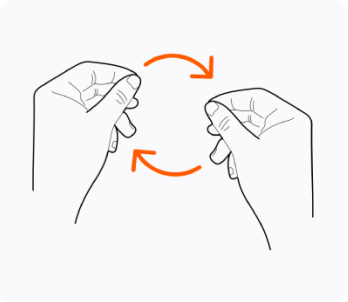


Explore new kinds of interaction

People can select an element by looking at it and tapping their fingers together. They can also pinch, drag, zoom, and rotate objects using specific hand gestures. SwiftUI provides built-in support for these standard gestures, so rely on them for most of your app's input. When you want to go beyond the standard gestures, use ARKit to create custom gestures.



Tap to select



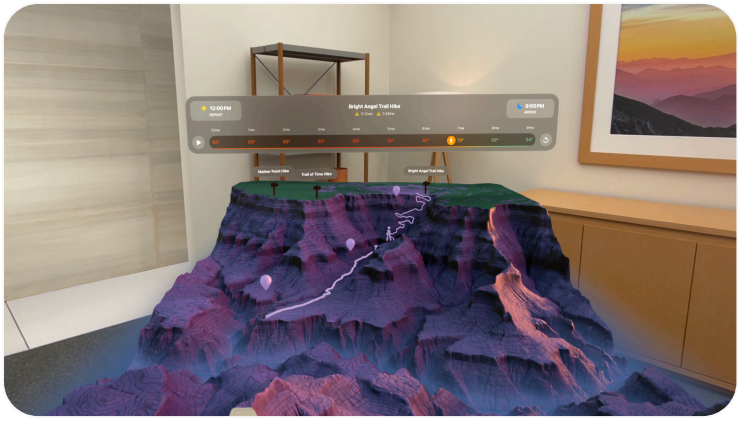
Pinch to rotate

Manipulate objects

Create custom gestures

Dive into featured sample apps

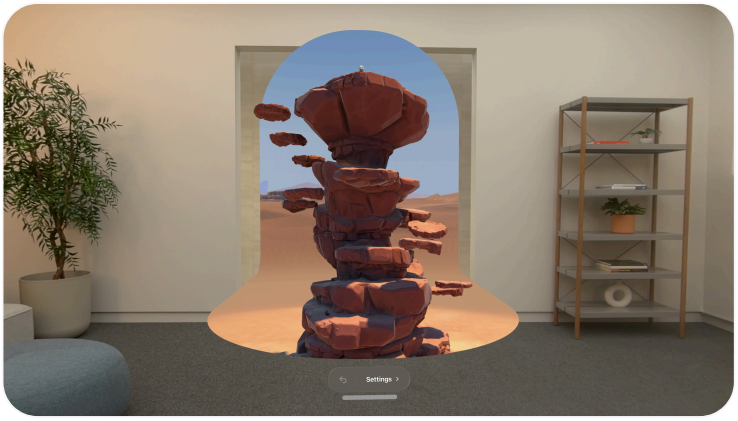
Explore the core concepts for all visionOS apps with Hello World. Understand how to detect custom gestures using ARKit with Happy Beam. Discover streaming 2D and stereoscopic media with Destination Video. And learn how to build 3D scenes with RealityKit and Reality Composer Pro with Diorama and Swift Splash.



Canyon Crosser: Building a volumetric hike-planning app

Create a hike planning app using SwiftUI and RealityKit.

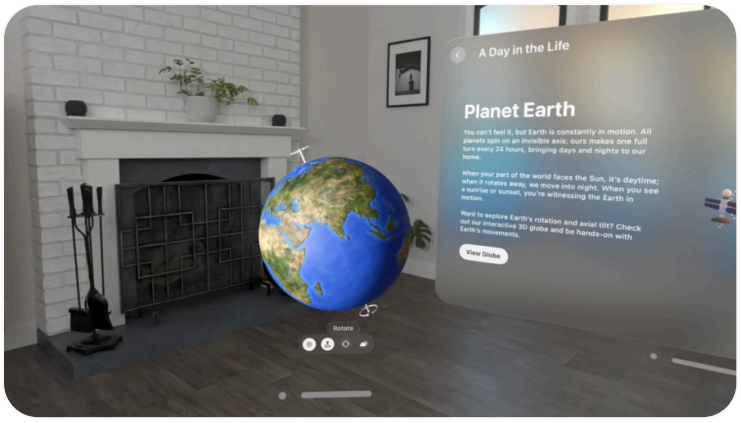
View sample code ›



Petite Asteroids: Building a volumetric visionOS game

Use the latest RealityKit APIs to create a beautiful video game for visionOS.

View sample code ›



Hello World

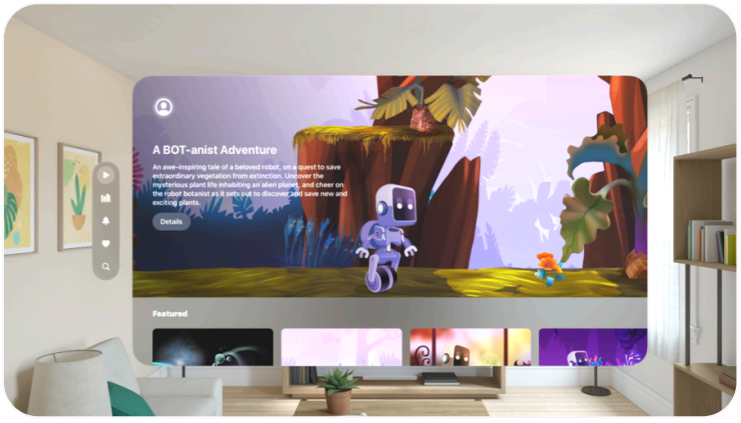
Use windows, volumes, and immersive spaces to teach people about the Earth.

View sample code ›

BOT-anist

Build a multiplatform app that uses windows, volumes, and animations to create a robot botanist's greenhouse.

View sample code ›



Destination Video

Happy Beam

Leverage SwiftUI to build an immersive media experience in a multiplatform app.

View sample code >

Leverage a Full Space to create a fun game using ARKit.

View sample code >

Diorama

Design scenes for your visionOS app using Reality Composer Pro.

View sample code >

Swift Splash

Use RealityKit to create an interactive ride in visionOS.

View sample code >

Topics

App construction

- Creating your first visionOS app

  Build a new visionOS app using SwiftUI and add platform-specific features.

- Adding 3D content to your app

  Add depth and dimension to your visionOS app and discover how to incorporate your app's content into a person's surroundings.

- Creating fully immersive experiences in your app

  Build fully immersive experiences by combining spaces with content you create using RealityKit or Metal.

- Drawing sharp layer-based content in visionOS

  Deliver text and vector images at multiple resolutions from custom Core Animation layers in visionOS.

- Introductory visionOS samples

Learn the fundamentals of building apps for visionOS with beginner-friendly sample code projects.

Design

- Designing for visionOS

  When people wear Apple Vision Pro, they enter an infinite 3D space where they can engage with your app or game while staying connected to their surroundings.

- Adopting best practices for privacy and user preferences

  Minimize your use of sensitive information and provide a clear statement of what information you do use and how you use it.

- Improving accessibility support in your visionOS app

  Update your code to ensure everyone can access your app's content in visionOS.

SwiftUI

- Canyon Crosser: Building a volumetric hike-planning app

  Create a hike planning app using SwiftUI and RealityKit.

- Hello World

  Use windows, volumes, and immersive spaces to teach people about the Earth.

- Presenting windows and spaces

  Open and close the scenes that make up your app's interface.

- Positioning and sizing windows

  Influence the initial geometry of windows that your app presents.

- Adopting best practices for persistent UI

  Create persistent and contextually relevant spatial experiences by managing scene restoration, customizing window behaviors, and surface snapping data.

RealityKit and Reality Composer Pro

- Reality Composer Pro

  Build, create, and design 3D content for your RealityKit apps.

- Petite Asteroids: Building a volumetric visionOS game

  Use the latest RealityKit APIs to create a beautiful video game for visionOS.

{} BOT-anist

Build a multiplatform app that uses windows, volumes, and animations to create a robot botanist's greenhouse.

{} Swift Splash

Use RealityKit to create an interactive ride in visionOS.

{} Diorama

Design scenes for your visionOS app using Reality Composer Pro.

{} Building an immersive media viewing experience

Add a deeper level of immersion to media playback in your app with RealityKit and Reality Composer Pro.

{} Enabling video reflections in an immersive environment

Create a more immersive experience by adding video reflections in a custom environment.

{} Combining 2D and 3D views in an immersive app

Use attachments to place 2D content relative to 3D content in your visionOS app.

Understanding the modular architecture of RealityKit

Learn how everything fits together in RealityKit.

Using transforms to move, scale, and rotate entities

Learn how to use Transforms to move, scale, and rotate entities in RealityKit.

Capturing screenshots and video from Apple Vision Pro for 2D viewing

Create screenshots and record high-quality video of your visionOS app and its surroundings for app previews.

Implementing object tracking in your visionOS app

Create engaging interactions by training models to recognize and track real-world objects in your app.

{} Placing entities using head and device transform

Query and react to changes in the position and rotation of Apple Vision Pro.

ARKit

{} Happy Beam

Leverage a Full Space to create a fun game using ARKit.

Setting up access to ARKit data

Check whether your app can use ARKit and respect people's privacy.

Incorporating real-world surroundings in an immersive experience

Create an immersive experience by making your app's content respond to the local shape of the world.

Placing content on detected planes

Detect horizontal surfaces like tables and floors, as well as vertical planes like walls and doors.

Tracking specific points in world space

Retrieve the position and orientation of anchors your app stores in ARKit.

Tracking preregistered images in 3D space

Place content based on the current position of a known image in a person's surroundings.

Exploring object tracking with ARKit

Find and track real-world objects in visionOS using reference objects trained with Create ML.

Object tracking with Reality Composer Pro experiences

Use object tracking in visionOS to attach digital content to real objects to create engaging experiences.

Building local experiences with room tracking

Use room tracking in visionOS to provide custom interactions with physical spaces.

Placing entities using head and device transform

Query and react to changes in the position and rotation of Apple Vision Pro.

Video playback

Destination Video

Leverage SwiftUI to build an immersive media experience in a multiplatform app.

Playing immersive media with RealityKit

Create an immersive video playback experience with RealityKit.

Rendering stereoscopic video with RealityKit

Render stereoscopic video in visionOS with RealityKit.

Creating a multiview video playback experience in visionOS

Build an interface that plays multiple videos simultaneously and handles transitions to different experience types gracefully.

- Configuring your app for media playback

  Configure apps to enable standard media playback behavior.

- Adopting the system player interface in visionOS

  Provide an optimized viewing experience for watching 3D video content.

- Controlling the transport behavior of a player

  Play, pause, and seek through a media presentation.

- Monitoring playback progress in your app

  Observe the playback of a media asset to update your app's user-interface state.

- Trimming and exporting media in visionOS

  Display standard controls in your app to edit the timeline of the currently playing media.

Xcode and Simulator

- Configuring your app icon using an asset catalog

  Add app icon variations to an asset catalog that represents your app in places such as the App Store, the Home Screen, Settings, and search results.

- Diagnosing and resolving bugs in your running app

  Inspect your app to isolate bugs, locate crashes, identify excess system-resource usage, visualize memory bugs, and investigate problems in its appearance.

- Diagnosing issues in the appearance of a running app

  Inspect your running app to investigate issues in the appearance and placement of the content it displays.

- Running your app in Simulator or on a device

  Launch your app in a simulated iOS, iPadOS, tvOS, visionOS, or watchOS device, or on a device connected to a Mac.

- Interacting with your app in the visionOS simulator

  Use your Mac to navigate spaces and control interactions with your visionOS apps in Simulator.

Performance

- Creating a performance plan for your visionOS app

  Identify your app's performance and power goals and create a plan to measure and assess them.

- Analyzing the performance of your visionOS app

  Use the RealityKit Trace template in Instruments to evaluate and improve the performance of your visionOS app.

- Reducing the rendering cost of your UI on visionOS

  Optimize your 2D user interface rendering on visionOS.

- Reducing the rendering cost of RealityKit content on visionOS

  Optimize your app's 3D augmented reality content to render efficiently on visionOS.

- Understanding the visionOS render pipeline

  Compare how visionOS handles events and manages its rendering loop differently from other Apple platforms.

iOS migration and compatibility

- Determining whether to bring your app to visionOS

  Decide whether to bring your existing iPadOS or iOS app to visionOS.

- Bringing your existing apps to visionOS

  Build a version of your iPadOS or iOS app using the visionOS SDK, and update your code for platform differences.

- Bringing your ARKit app to visionOS

  Update an iPadOS or iOS app that uses ARKit, and provide an equivalent experience in visionOS.

- Making your existing app compatible with visionOS

  Modify your iPadOS or iOS app to run successfully in visionOS as a compatible app.

Enterprise APIs for visionOS

- {} Accessing the main camera

  Add camera-based features to enterprise apps.

- Building spatial experiences for business apps with enterprise APIs for visionOS

Grant enhanced sensor access and increased platform control to your visionOS app by using entitlements.

{} Displaying video from connected devices

Show video from devices connected with the Developer Strap in your visionOS app.

{} Locating and decoding barcodes in 3D space

Create engaging, hands-free experiences based on barcodes in a person’s surroundings.