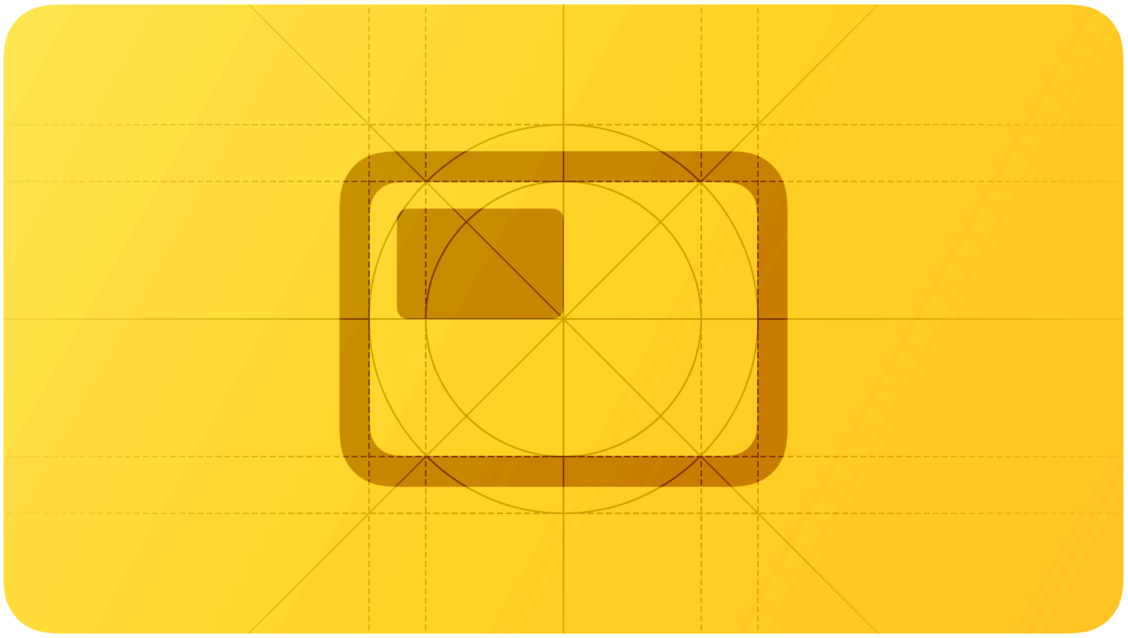
June 9, 2025
Added guidance for Liquid Glass.

Layout

A consistent layout that adapts to various contexts makes your experience more approachable and helps people enjoy their favorite apps and games on all their devices.



Your app's layout helps ground people in your content from the moment they open it. People expect familiar relationships between controls and content to help them use and discover your app's features, and designing the layout to take advantage of this makes your app feel at home on the platform.

Apple provides templates, guides, and other resources that can help you integrate Apple technologies and design your apps and games to run on all Apple platforms. See Apple Design Resources.

Best practices

Group related items to help people find the information they want. For example, you might use negative space, background shapes, colors, materials, or separator lines to show when elements are related and to separate information into distinct areas. When you do so, ensure that content and controls remain clearly distinct.

Make essential information easy to find by giving it sufficient space. People want to view the most important information right away, so don't obscure it by crowding it with nonessential details. You can make secondary information available in other parts of the window, or include it in an additional view.

Extend content to fill the screen or window. Make sure backgrounds and full-screen artwork extend to the edges of the display. Also ensure that scrollable layouts continue all the way to the bottom and the sides of the device screen. Controls and navigation components like sidebars and tab bars appear on top of content rather than on the same plane, so it's important for your layout to take this into account.

When your content doesn't span the full window, use a background extension view to provide the appearance of content behind the control layer on either side of the screen, such as beneath the sidebar or inspector. For developer guidance, see `backgroundExtensionEffect()` and `UIBackgroundExtensionView`.

Visual hierarchy

Differentiate controls from content. Take advantage of the Liquid Glass material to provide a distinct appearance for controls that's consistent across iOS, iPadOS, and macOS. Instead of a background, use a scroll edge effect to provide a transition between content and the control area. For guidance, see Scroll views.

Place items to convey their relative importance. People often start by viewing items in reading order — that is, from top to bottom and from the leading to trailing side — so it generally works well to place the most important items near the top and leading side of the window, display, or field of view. Be aware that reading order varies by language, and take right to left languages into account as you design.

Align components with one another to make them easier to scan and to communicate organization and hierarchy. Alignment makes an app look neat and organized and can help people track content while scrolling or moving their eyes, making it easier to find information. Along with indentation, alignment can also help people understand an information hierarchy.

Take advantage of progressive disclosure to help people discover content that's currently hidden. For example, if you can't display all the items in a large collection at once, you need to indicate that there are additional items that aren't currently visible. Depending on the platform, you might use a disclosure control, or display parts of items to hint that people can reveal additional content by interacting with the view, such as by scrolling.

Make controls easier to use by providing enough space around them and grouping them in logical sections. If unrelated controls are too close together — or if other content crowds them — they can be difficult for people to tell apart or understand what they do, which can make your app or game hard to use. For guidance, see Toolbars.

Adaptability

Every app and game needs to adapt when the device or system context changes. In iOS, iPadOS, tvOS, and visionOS, the system defines a collection of *traits* that characterize variations in the device environment that can affect the way your app or game looks. Using SwiftUI or Auto Layout can help you ensure that your interface adapts dynamically to these traits and other context changes; if you don't use these tools, you need to use alternative methods to do the work.

Here are some of the most common device and system variations you need to handle:

- Different device screen sizes, resolutions, and color spaces
- Different device orientations (portrait/landscape)
- System features like Dynamic Island and camera controls
- External display support, Display Zoom, and resizable windows on iPad
- Dynamic Type text-size changes
- Locale-based internationalization features like left-to-right/right-to-left layout direction, date/time/number formatting, font variation, and text length

Design a layout that adapts gracefully to context changes while remaining recognizably consistent. People expect your experience to work well and remain familiar when they rotate their device, resize a window, add another display, or switch to a different device. You can help ensure an adaptable interface by respecting system-defined safe areas, margins, and guides (where available) and specifying layout modifiers to fine-tune the placement of views in your interface.

Be prepared for text-size changes. People appreciate apps and games that respond when they choose a different text size. When you support Dynamic Type — a feature that lets people choose the size of visible text in iOS, iPadOS, tvOS, visionOS, and watchOS — your app or game can respond appropriately when people adjust text size. To support Dynamic Type in your Unitybased game, use Apple's accessibility plug-in (for developer guidance, see Apple – Accessibility). For guidance on displaying text in your app, see Typography.

Preview your app on multiple devices, using different orientations, localizations, and text sizes. You can streamline the testing process by first testing versions of your experience that use the largest and the smallest layouts. Although it's generally best to preview features like wide-gamut color on actual devices, you can use Xcode Simulator to check for clipping and other layout issues. For example, if your iOS app or game supports landscape mode, you can use Simulator to make sure your layouts look great whether the device rotates left or right.

When necessary, scale artwork in response to display changes. For example, viewing your app or game in a different context — such as on a screen with a different aspect ratio — might make your artwork appear cropped, letterboxed, or pillarboxed. If this happens, don't change the aspect ratio of the artwork; instead, scale it so that important visual content remains visible. In visionOS, the system automatically scales a window when it moves along the z-axis.

Guides and safe areas

A *layout guide* defines a rectangular region that helps you position, align, and space your content on the screen. The system includes predefined layout guides that make it easy to apply standard margins around content and restrict the width of text for optimal readability. You can also define custom layout guides. For developer guidance, see `UILayoutGuide` and `NSLayoutGuide`.

A *safe area* defines the area within a view that isn't covered by a toolbar, tab bar, or other views a window might provide. Safe areas are essential for avoiding a device's interactive and display features, like Dynamic Island on iPhone or the camera housing on some Mac models. For developer guidance, see `SafeAreaRegions` and Positioning content relative to the safe area.

Respect key display and system features in each platform. When an app or game doesn't accommodate such features, it doesn't feel at home in the platform and may be harder for people to use. In addition to helping you avoid display and system features, safe areas can also help you account for interactive components like bars, dynamically repositioning content when sizes change.

For templates that include the guides and safe areas for each platform, see Apple Design Resources.

Platform considerations

iOS

Aim to support both portrait and landscape orientations. People appreciate apps and games that work well in different device orientations, but sometimes your experience needs to run in only portrait or only landscape. When this is the case, you can rely on people trying both orientations before settling on the one you support — there's no need to tell people to rotate their device. If your app or game is landscape-only, make sure it runs equally well whether people rotate their device to the left or the right.

Prefer a full-bleed interface for your game. Give players a beautiful interface that fills the screen while accommodating the corner radius, sensor housing, and features like Dynamic

Island. If necessary, consider giving players the option to view your game using a letterboxed or pillarboxed appearance.

Avoid full-width buttons. Buttons feel at home in iOS when they respect system-defined margins and are inset from the edges of the screen. If you need to include a full-width button, make sure it harmonizes with the curvature of the hardware and aligns with adjacent safe areas.

Hide the status bar only when it adds value or enhances your experience. The status bar displays information people find useful and it occupies an area of the screen most apps don't fully use, so it's generally a good idea to keep it visible. The exception is if you offer an in-depth experience like playing a game or viewing media, where it might make sense to hide the status bar.

iPadOS

People can freely resize windows down to a minimum width and height, similar to window behavior in macOS. It's important to account for this resizing behavior and the full range of possible window sizes when designing your layout. For guidance, see Multitasking and Windows.

As someone resizes a window, defer switching to a compact view for as long as possible. Design for a full-screen view first, and only switch to a compact view when a version of the full layout no longer fits. This helps the UI feel more stable and familiar in as many situations as possible. For more complex layouts such as split views, prefer hiding tertiary columns such as inspectors as the view narrows.

Test your layout at common system-provided sizes, and provide smooth transitions. Window controls provide the option to arrange windows to fill halves, thirds, and quadrants of the screen, so it's important to check your layout at each of these sizes on a variety of devices. Be sure to minimize unexpected UI changes as people adjust down to the minimum and up to the maximum window size.

Consider a convertible tab bar for adaptive navigation. For many apps, you don't need to choose between a tab bar or sidebar for navigation; instead, you can adopt a style of tab bar that provides both. The app first launches with your choice of a sidebar or a tab bar, and then people can tap to switch between them. As the view resizes, the presentation style changes to fit the width of the view. For guidance, see Tab bars. For developer guidance, see `sidebarAdaptable`.

macOS

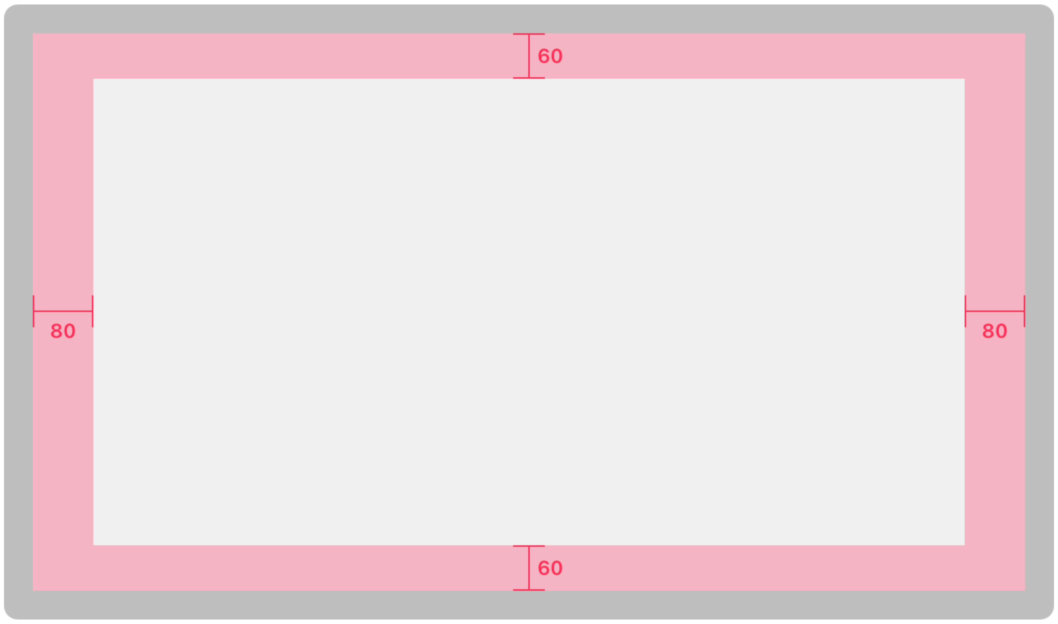
Avoid placing controls or critical information at the bottom of a window. People often move windows so that the bottom edge is below the bottom of the screen.

Avoid displaying content within the camera housing at the top edge of the window. For developer guidance, see `NSPrefersDisplaySafeAreaCompatibilityMode`.

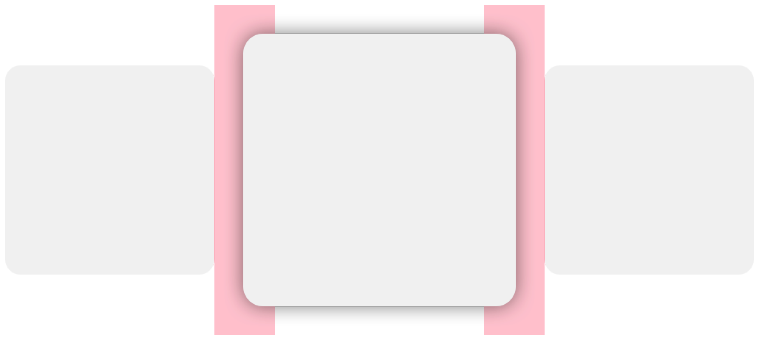
tvOS

Be prepared for a wide range of TV sizes. On Apple TV, layouts don't automatically adapt to the size of the screen like they do on iPhone or iPad. Instead, apps and games show the same interface on every display. Take extra care in designing your layout so that it looks great in a variety of screen sizes.

Adhere to the screen's safe area. Inset primary content 60 points from the top and bottom of the screen, and 80 points from the sides. It can be difficult for people to see content that close to the edges, and unintended cropping can occur due to overscanning on older TVs. Allow only partially displayed offscreen content and elements that deliberately flow offscreen to appear outside this zone.



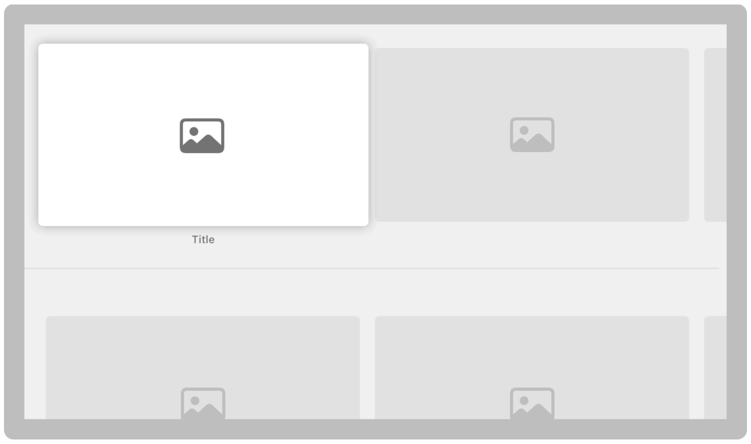
Include appropriate padding between focusable elements. When you use UIKit and the focus APIs, an element gets bigger when it comes into focus. Consider how elements look when they're focused, and make sure you don't let them overlap important information. For developer guidance, see About focus interactions for Apple TV.



Grids

The following grid layouts provide an optimal viewing experience. Be sure to use appropriate spacing between unfocused rows and columns to prevent overlap when an item comes into focus.

If you use the UIKit collection view flow element, the number of columns in a grid is automatically determined based on the width and spacing of your content. For developer guidance, see `UICollectionViewFlowLayout`.



- Two-column
- Three-column
- Four-column
- Five-column
- Six-column
- Seven-column
- Eight-column
- Nine-column

Two-column grid

Attribute	Value
Unfocused content width	860 pt
Horizontal spacing	40 pt
Minimum vertical spacing	100 pt

Include additional vertical spacing for titled rows. If a row has a title, provide enough spacing between the bottom of the previous unfocused row and the center of the title to avoid crowding. Also provide spacing between the bottom of the title and the top of the unfocused items in the row.

Use consistent spacing. When content isn't consistently spaced, it no longer looks like a grid and it's harder for people to scan.

Make partially hidden content look symmetrical. To help direct attention to the fully visible content, keep partially hidden offscreen content the same width on each side of the screen.

visionOS

The guidance below can help you lay out content within the windows of your visionOS app or game, making it feel familiar and easy to use. For guidance on displaying windows in space and best practices for using depth, scale, and field of view in your visionOS app, see Spatial layout. To learn more about visionOS window components, see Windows > visionOS.

> Note
>
> When you add depth to content in a standard window, the content extends beyond the window's bounds along the z-axis. If content extends too far along the z-axis, the system clips it.

Consider centering the most important content and controls in your app or game. Often, people can more easily discover and interact with content when it's near the middle of a window, especially when the window is large.

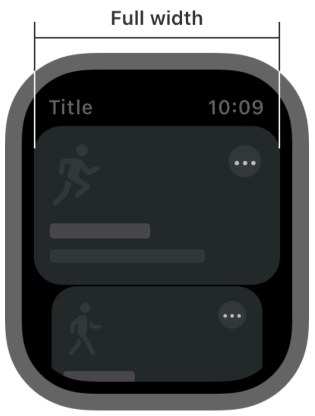
Keep a window's content within its bounds. In visionOS, the system displays window controls just outside a window's bounds in the XY plane. For example, the Share menu appears above the window and the controls for resizing, moving, and closing the window appear below it. Letting 2D or 3D content encroach on these areas can make the system-provided controls, especially those below the window, difficult for people to use.

If you need to display additional controls that don't belong within a window, use an ornament. An ornament lets you offer app controls that remain visually associated with a window without interfering with the system-provided controls. For example, a window's toolbar and tab bar appear as ornaments. For guidance, see Ornaments.

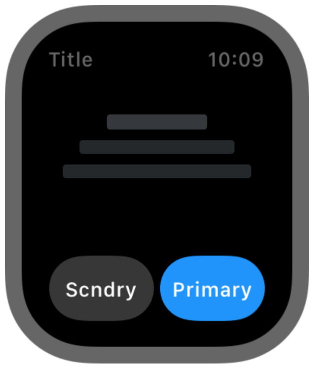
Make a window's interactive components easy for people to look at. You need to include enough space around an interactive component so that visually identifying it is easy and comfortable, and to prevent the system-provided hover effect from obscuring other content. For example, place buttons so their centers are at least 60 points apart. For guidance, see Eyes, Spatial layout, and Buttons > visionOS.

watchOS

Design your content to extend from one edge of the screen to the other. The Apple Watch bezel provides a natural visual padding around your content. To avoid wasting valuable space, consider minimizing the padding between elements.



Avoid placing more than two or three controls side by side in your interface. As a general rule, display no more than three buttons that contain glyphs — or two buttons that contain text — in a row. Although it's usually better to let text buttons span the full width of the screen, two side-by-side buttons with short text labels can also work well, as long as the screen doesn't scroll.



Support autorotation in views people might want to show others. When people flip their wrist away, apps typically respond to the motion by sleeping the display, but in some cases it makes sense to autorotate the content. For example, a wearer might want to show an image to a friend or display a QR code to a reader. For developer guidance, see `isAutorotating`.

Specifications

iOS, iPadOS device screen dimensions

Model	Dimensions (portrait)
iPad Pro 12.9-inch	1024x1366 pt (2048x2732 px @2x)
iPad Pro 11-inch	834x1194 pt (1668x2388 px @2x)
iPad Pro 10.5-inch	834x1194 pt (1668x2388 px @2x)
iPad Pro 9.7-inch	768x1024 pt (1536x2048 px @2x)
iPad Air 13-inch	1024x1366 pt (2048x2732 px @2x)
iPad Air 11-inch	820x1180 pt (1640x2360 px @2x)
iPad Air 10.9-inch	820x1180 pt (1640x2360 px @2x)
iPad Air 10.5-inch	834x1112 pt (1668x2224 px @2x)
iPad Air 9.7-inch	768x1024 pt (1536x2048 px @2x)
iPad 11-inch	820x1180 pt (1640x2360 px @2x)
iPad 10.2-inch	810x1080 pt (1620x2160 px @2x)
iPad 9.7-inch	768x1024 pt (1536x2048 px @2x)
iPad mini 8.3-inch	744x1133 pt (1488x2266 px @2x)
iPad mini 7.9-inch	768x1024 pt (1536x2048 px @2x)
iPhone 16 Pro Max	440x956 pt (1320x2868 px @3x)
iPhone 16 Pro	402x874 pt (1206x2622 px @3x)
iPhone 16 Plus	430x932 pt (1290x2796 px @3x)
iPhone 16	393x852 pt (1179x2556 px @3x)
iPhone 16e	390x844 pt (1170x2532 px @3x)
iPhone 15 Pro Max	430x932 pt (1290x2796 px @3x)
iPhone 15 Pro	393x852 pt (1179x2556 px @3x)
iPhone 15 Plus	430x932 pt (1290x2796 px @3x)
iPhone 15	393x852 pt (1179x2556 px @3x)
iPhone 14 Pro Max	430x932 pt (1290x2796 px @3x)

Model	Dimensions (portrait)
iPhone 14 Pro	393x852 pt (1179x2556 px @3x)
iPhone 14 Plus	428x926 pt (1284x2778 px @3x)
iPhone 14	390x844 pt (1170x2532 px @3x)
iPhone 13 Pro Max	428x926 pt (1284x2778 px @3x)
iPhone 13 Pro	390x844 pt (1170x2532 px @3x)
iPhone 13	390x844 pt (1170x2532 px @3x)
iPhone 13 mini	375x812 pt (1125x2436 px @3x)
iPhone 12 Pro Max	428x926 pt (1284x2778 px @3x)
iPhone 12 Pro	390x844 pt (1170x2532 px @3x)
iPhone 12	390x844 pt (1170x2532 px @3x)
iPhone 12 mini	375x812 pt (1125x2436 px @3x)
iPhone 11 Pro Max	414x896 pt (1242x2688 px @3x)
iPhone 11 Pro	375x812 pt (1125x2436 px @3x)
iPhone 11	414x896 pt (828x1792 px @2x)
iPhone XS Max	414x896 pt (1242x2688 px @3x)
iPhone XS	375x812 pt (1125x2436 px @3x)
iPhone XR	414x896 pt (828x1792 px @2x)
iPhone X	375x812 pt (1125x2436 px @3x)
iPhone 8 Plus	414x736 pt (1080x1920 px @3x)
iPhone 8	375x667 pt (750x1334 px @2x)
iPhone 7 Plus	414x736 pt (1080x1920 px @3x)
iPhone 7	375x667 pt (750x1334 px @2x)
iPhone 6s Plus	414x736 pt (1080x1920 px @3x)
iPhone 6s	375x667 pt (750x1334 px @2x)
iPhone 6 Plus	414x736 pt (1080x1920 px @3x)
iPhone 6	375x667 pt (750x1334 px @2x)
iPhone SE 4.7-inch	375x667 pt (750x1334 px @2x)
iPhone SE 4-inch	320x568 pt (640x1136 px @2x)
iPod touch 5th generation and later	320x568 pt (640x1136 px @2x)

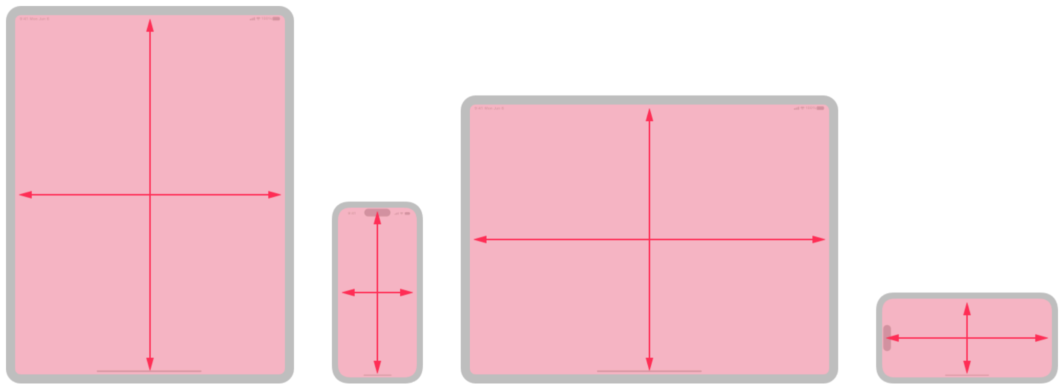
Note

All scale factors in the table above are UIKit scale factors, which may differ from native scale factors. For developer guidance, see `scale` and `nativeScale`.

iOS, iPadOS device size classes

A size class is a value that's either regular or compact, where *regular* refers to a larger screen or a screen in landscape orientation and *compact* refers to a smaller screen or a screen in portrait orientation. For developer guidance, see `UserInterfaceSizeClass`.

Different size class combinations apply to the full-screen experience on different devices, based on screen size.



Model	Portrait orientation	Landscape orientation
iPad Pro 12.9-inch	Regular width, regular height	Regular width, regular height
iPad Pro 11-inch	Regular width, regular height	Regular width, regular height
iPad Pro 10.5-inch	Regular width, regular height	Regular width, regular height
iPad Air 13-inch	Regular width, regular height	Regular width, regular height
iPad Air 11-inch	Regular width, regular height	Regular width, regular height
iPad 11-inch	Regular width, regular height	Regular width, regular height
iPad 9.7-inch	Regular width, regular height	Regular width, regular height
iPad mini 7.9-inch	Regular width, regular height	Regular width, regular height
iPhone 16 Pro Max	Compact width, regular height	Regular width, compact height
iPhone 16 Pro	Compact width, regular height	Compact width, compact height
iPhone 16 Plus	Compact width, regular height	Regular width, compact height
iPhone 16	Compact width, regular height	Compact width, compact height
iPhone 16e	Compact width, regular height	Compact width, compact height
iPhone 15 Pro Max	Compact width, regular height	Regular width, compact height

Model	Portrait orientation	Landscape orientation
iPhone 15 Pro	Compact width, regular height	Compact width, compact height
iPhone 15 Plus	Compact width, regular height	Regular width, compact height
iPhone 15	Compact width, regular height	Compact width, compact height
iPhone 14 Pro Max	Compact width, regular height	Regular width, compact height
iPhone 14 Pro	Compact width, regular height	Compact width, compact height
iPhone 14 Plus	Compact width, regular height	Regular width, compact height
iPhone 14	Compact width, regular height	Compact width, compact height
iPhone 13 Pro Max	Compact width, regular height	Regular width, compact height
iPhone 13 Pro	Compact width, regular height	Compact width, compact height
iPhone 13	Compact width, regular height	Compact width, compact height
iPhone 13 mini	Compact width, regular height	Compact width, compact height
iPhone 12 Pro Max	Compact width, regular height	Regular width, compact height
iPhone 12 Pro	Compact width, regular height	Compact width, compact height
iPhone 12	Compact width, regular height	Compact width, compact height
iPhone 12 mini	Compact width, regular height	Compact width, compact height
iPhone 11 Pro Max	Compact width, regular height	Regular width, compact height
iPhone 11 Pro	Compact width, regular height	Compact width, compact height
iPhone 11	Compact width, regular height	Regular width, compact height
iPhone XS Max	Compact width, regular height	Regular width, compact height
iPhone XS	Compact width, regular height	Compact width, compact height
iPhone XR	Compact width, regular height	Regular width, compact height
iPhone X	Compact width, regular height	Compact width, compact height
iPhone 8 Plus	Compact width, regular height	Regular width, compact height
iPhone 8	Compact width, regular height	Compact width, compact height
iPhone 7 Plus	Compact width, regular height	Regular width, compact height
iPhone 7	Compact width, regular height	Compact width, compact height
iPhone 6s Plus	Compact width, regular height	Regular width, compact height
iPhone 6s	Compact width, regular height	Compact width, compact height
iPhone SE	Compact width, regular height	Compact width, compact height
iPod touch 5th generation and later	Compact width, regular height	Compact width, compact height

watchOS device screen dimensions

Series	Size	Width (pixels)	Height (pixels)
10	42mm	374	446
10	46mm	416	496
Apple Watch Ultra (all generations)	49mm	410	502
7, 8, and 9	41mm	352	430
7, 8, and 9	45mm	396	484
4, 5, 6, and SE	40mm	324	394
4, 5, 6, and SE	44mm	368	448
1, 2, and 3	38mm	272	340
1, 2, and 3	42mm	312	390

Resources

Related

Right to left

Spatial layout

Layout and organization

Developer documentation

Composing custom layouts with SwiftUI — SwiftUI

Videos

Get to know the new design system

Compose custom layouts with SwiftUI

Essential Design Principles

Change log

Supported platforms

Layout
Best practices
Visual hierarchy
Adaptability
Guides and safe areas
Platform considerations
Specifications
Resources
Change log

Date	Changes
June 9, 2025	Added guidance for Liquid Glass.
March 7, 2025	Added specifications for iPhone 16e, iPad 11-inch, iPad Air 11-inch, and iPad Air 13-inch.
September 9, 2024	Added specifications for iPhone 16, iPhone 16 Plus, iPhone 16 Pro, iPhone 16 Pro Max, and Apple Watch Series 10.
June 10, 2024	Made minor corrections and organizational updates.
February 2, 2024	Enhanced guidance for avoiding system controls in iPadOS app layouts, and added specifications for 10.9-inch iPad Air and 8.3-inch iPad mini.
December 5, 2023	Clarified guidance on centering content in a visionOS window.
September 15, 2023	Added specifications for iPhone 15 Pro Max, iPhone 15 Pro, iPhone 15 Plus, iPhone 15, Apple Watch Ultra 2, and Apple Watch SE.
June 21, 2023	Updated to include guidance for visionOS.
September 14, 2022	Added specifications for iPhone 14 Pro Max, iPhone 14 Pro, iPhone 14 Plus, iPhone 14, and Apple Watch Ultra.