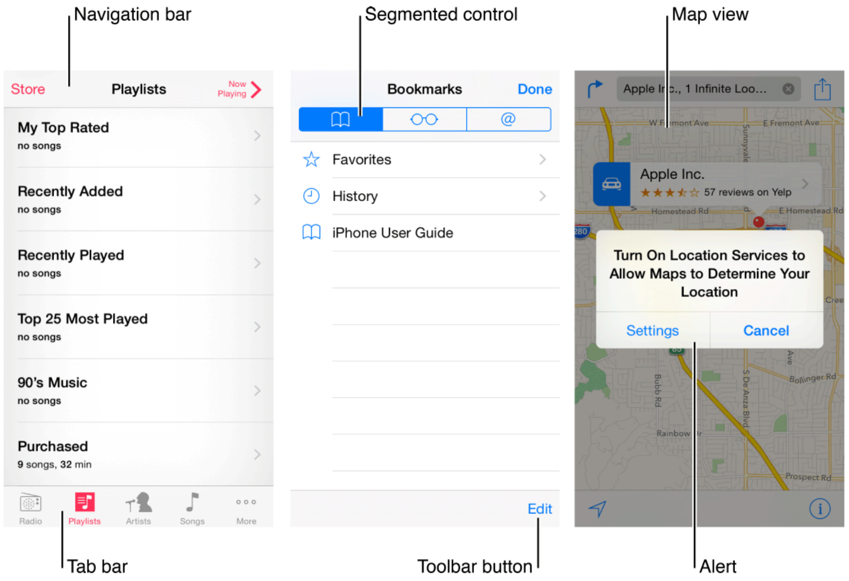
iOS App Anatomy

Almost all iOS apps use at least some of the UI components defined by the UIKit framework. Knowing the names, roles, and capabilities of these basic components helps you make informed decisions as you design the UI of your app.



The UI elements provided by UIKit fall into four broad categories:

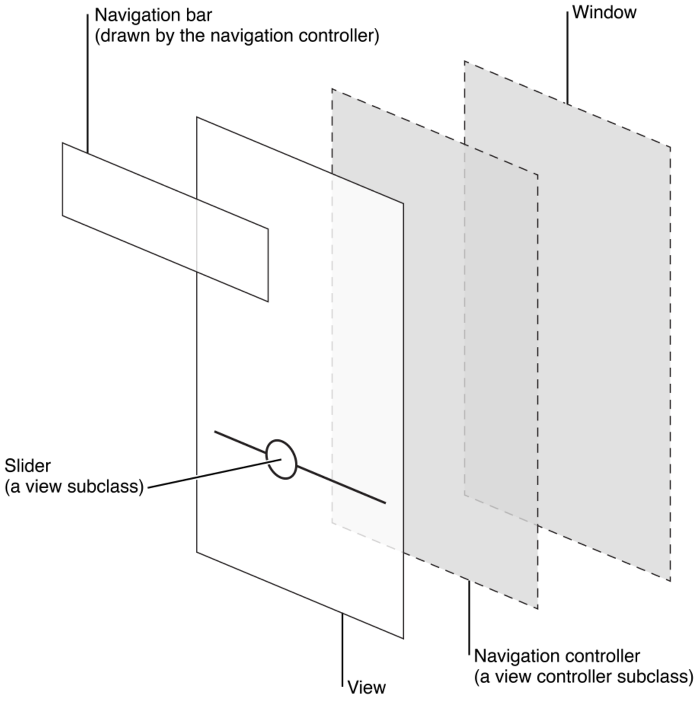
- **Bars.** Bars contain contextual information that tells users where they are and controls that help users navigate or initiate actions.
- **Content views.** Content views contain app-specific content and can enable behaviors such as scrolling, insertion, deletion, and rearrangement of items.
- **Controls.** Controls perform actions or display information.
- **Temporary views.** Temporary views appear briefly to give users important information or additional choices and functionality.

In addition to defining UI elements, UIKit defines objects that implement functionality, such as gesture recognition, drawing, accessibility, and printing support.

Programmatically, a UI element is a type of *view* because it inherits from `UIView`. A view knows how to draw itself onscreen, and it knows when a user touches within its bounds. Controls (such as buttons and sliders), content views (such as collection views and table views), and temporary views (such as alerts and action sheets) are all types of views.

To manage a set or hierarchy of views in your app, you typically use a *view controller*. A view controller coordinates the display of views, implements the functionality behind user interactions, and can manage transitions from one screen to another. For example, Settings uses a navigation controller to display its hierarchy of views.

Here's an example of how views and view controllers can combine to present the UI of an iOS app.



Although developers think in terms of views and view controllers, users tend to experience an iOS app as a collection of screens. From this perspective, a *screen* generally corresponds to a distinct visual state or mode in an app.

> NOTE
>
> An iOS app includes a window. But—unlike a window in a computer app—an iOS window has no visible parts and it can't be moved to another location on the display. Most iOS apps contain only one window; apps that support an external display can have more than one.

In *iOS Human Interface Guidelines*, the word *screen* is used as it's understood by most users. As a developer, you might also read about screens in other contexts, where the term refers to the `UIScreen` object you can use to access an external display screen.