* [sourced from iOS Human Interface Guidelines]
* Designing for iOS
* iOS 7 embodies the following themes:
  + **Deference.** The UI helps users understand and interact with the content, but never competes with it.
  + **Clarity.** Text is legible at every size, icons are precise and lucid, adornments are subtle and appropriate, and a sharpened focus on functionality motivates the design.
  + **Depth.** Visual layers and realistic motion impart vitality and heighten users’ delight and understanding.
* **Take advantage of the whole screen.**
* **Use plenty of negative space.** Negative space makes important content and functionality more noticeable and easier to understand.
* **Let color simplify the UI.** A key color—such as yellow in Notes—highlights important state and subtly indicates interactivity.
* **Embrace borderless buttons.**
* **Use Depth to Communicate.** Calendar app zooms in/out from year to month, month to week, week to day.
* Starting and Stopping
* **Start Instantly**
  + **As much as possible, avoid displaying a splash screen or other startup experience.** Avoid asking people to supply setup information
    - **Focus on the needs of 80 percent of your users.**
    - **Get as much information as possible from other sources.**
  + **Delay a login requirement for as long as possible.** It’s best when users can navigate through much of your app and use some of its functionality without logging in. For example, App Store doesn’t ask users to log in until they decide to buy something. Users often abandon apps that force them to log in before they can do anything useful.
* **Always Be Prepared to Stop**
  + **Save user data as soon as possible and as often as reasonable.**
  + **Save the current state when stopping at the finest level of detail possible.**
  + **If all app features are unavailable, display a screen that describes the situation and suggests a correction.**
    - If only some app features are unavailable, display either a screen or an alert when people try to use the feature.
* Layout
* **Make it easy for people to interact with content and controls by giving each interactive element ample spacing.**
  + Give tappable controls a hit target of about 44 x 44 points.
* **Make it easy to focus on the main task by elevating important content or functionality.**
  + Some good ways to do this are to place principal items in the upper half of the screen and—in left-to-right cultures—near the left side of the screen
* **Use visual weight and balance to show users the relative importance of onscreen elements.**
  + Large items catch the eye and tend to appear more important than smaller ones. Larger items are also easier for users to tap, which makes them especially useful in apps—such as Phone and Clock—that users often use in distracting surroundings.
* **Use alignment to ease scanning and communicate groupings or hierarchy.**
  + Alignment tends to make an app look neat and organized and it gives users places to focus while they scroll through screenfuls of information.
* **Make sure that users can understand primary content at its default size.**
  + For example, users shouldn’t have to scroll horizontally to read important text, or zoom to see primary images.
* **As much as possible, avoid inconsistent appearances in your UI.**
  + In general, elements that have similar functions should also look similar.
* Navigation
* Broadly speaking, there are three main styles of navigation, each of which is well suited to a specific app structure:
  + Hierarchical (Settings, Mail)
  + Flat (Music, App Store)
  + Content or experience driven (games; books slide to new pages)
* **Users should always know where they are in your app and how to get to their next destination.**
* **Use a navigation bar to give users an easy way to traverse a hierarchy of data.** The navigation bar’s title can show users their current position in the hierarchy; the back button makes it easy to return to the previous level.
* **Use a tab bar to display several peer categories of content or functionality.** A tab bar is a good way to support a flat information architecture and its persistence lets people switch between categories regardless of their current location.
* **In general, it’s best to give users one path to each screen.** If there’s one screen that users need to see in more than one context, consider using a temporary view, such as a modal view, action sheet, or alert.
* UIKit also provides the following related controls:
  + [Segmented Control](https://developer.apple.com/library/ios/documentation/userexperience/conceptual/MobileHIG/Controls.html#//apple_ref/doc/uid/TP40006556-CH15-SW27). A segmented control can give users a way to see different categories or aspects of the content on the screen; it doesn’t enable navigation to a new screen.
  + [Toolbar](https://developer.apple.com/library/ios/documentation/userexperience/conceptual/MobileHIG/Bars.html#//apple_ref/doc/uid/TP40006556-CH12-SW4). Although a toolbar looks similar to a navigation bar or a tab bar, it doesn’t enable navigation. Instead, a toolbar gives users controls that act on the contents of the current screen.
* **734-721-2500**
* Interactivity and Feedback
* To signal interactivity, the built-in apps use a variety of cues:
  + color
  + location
  + context
  + meaningful icons and labels
  + Users rarely need additional decorations to show them that an onscreen element is interactive or to suggest what it does.
* **In a content area, add a button border or background only if necessary.**
  + Buttons in bars, action sheets, and alerts don’t need borders because users know that most of the items in these areas are interactive. In a content area, on the other hand, a button might need a border or a background to distinguish it from the rest of the content.
* **As much as possible, integrate status and other relevant feedback information into your UI.**
  + It’s best when users can get this type of information without taking action or being distracted from their content. For example, Mail displays the update status in the toolbar where it doesn’t compete with the user’s content.
* **Avoid unnecessary alerts.**
  + An alert is a powerful feedback mechanism, but it should be used only to deliver important—and ideally actionable—information. If users see too many alerts that don’t contain essential information, they quickly learn to ignore all alerts.
* Inputting information takes time and attention, whether people tap controls or use the keyboard. **When an app slows people down by asking for a lot of user input before anything useful happens, people can feel discouraged from using it.**
* **Make it easy for users to make choices.**
  + For example, you can use a picker or a table view instead of a text field, because most people find it easier to select an item from a list than to type words.
* **Balance a request for input by giving users something useful in return.**
  + A sense of give and take helps people feel they’re making progress as they move through your app.