

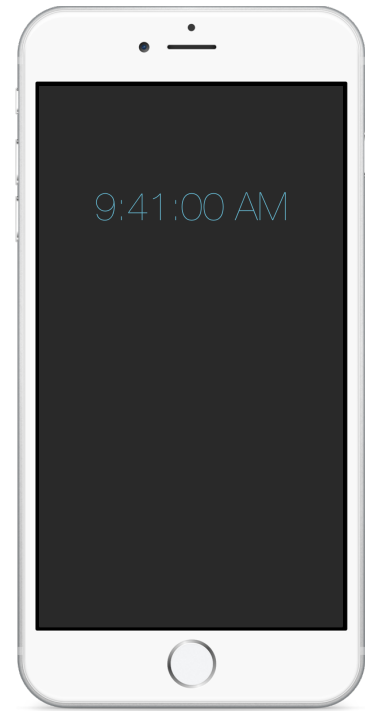
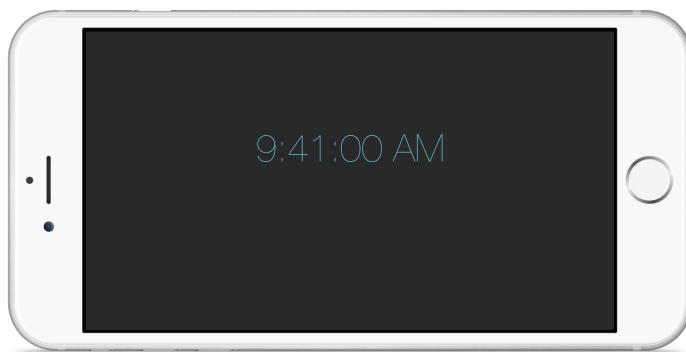
Clock

Lesson 6



Description

Hide the status bar to make the app feel "full screen," and support all device orientations.



Learning Outcomes

- Assess iOS best practices and app design requirements.
- Employ project configuration and controller methods to handle interface orientations.

Vocabulary

Human Interface Guidelines	iOS Status Bar	property list file
orientation	type conversion	bit mask
UIInterfaceOrientationMask	Xcode project	

Materials

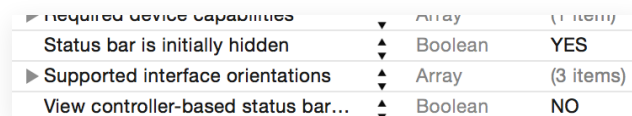
- **Clock Lesson 6** Xcode project

Opening

How can we get our app to display correctly when the device is rotated?

Agenda

- Explain how the iOS Human Interface Guidelines, or "HIG," describes best practices for consistent, high quality user experience.
- Discuss the best practice of not hiding the iOS status bar, but making the design decision to hide the status bar for this app in order to remove the redundancy of the status bar's time display.
- Explain how an individual view controller can override a `prefersStatusBarHidden` method, and how the status bar can be disabled application-wide through configuration instead of code.
- Using the Project Navigator (⌘1), select **Info.plist**, add a new `Boolean` item called `Status bar is initially hidden` and assign it the value `YES`. Add a second `Boolean` item called `View controller-based status bar appearance` and assign it the value `NO`.



Required device capabilities	Type	Value
Status bar is initially hidden	Boolean	YES
Supported interface orientations	Array	(3 items)
View controller-based status bar appearance	Boolean	NO

- Explain the significance of an app's main property list file for configuration.
- Run the app (⌘R), and observe how the status bar is now hidden.
- With the app still running in the Simulator, use the *Hardware* menu (or ⌘→ and ⌘←) to rotate the device, and observe how the interface does not change when in the upside down orientation.
- Discuss the need to change the orientation of the interface when the device orientation changes.
- Using the Xcode Documentation and API Reference (⇧⌘0), explore the `UIViewController` class reference and the `supportedInterfaceOrientations` method.
- Express the controller's support of multiple orientations by overriding the `supportedInterfaceOrientations` method.

```
override func supportedInterfaceOrientations() -> Int {  
    return Int(UIInterfaceOrientationMask.All.rawValue)  
}
```

- Briefly explain how this method returns an `Int` and uses the Swift type conversion idiom before returning.
- Using the Xcode Documentation and API Reference (⇧⌘0), search the documentation for `UIInterfaceOrientationMask` and observe the static properties declared in the structure.
- Explain how supported orientations in an iOS app are represented as bit masks.
- Using the Project Navigator (⌘1), select the **Clock** project, observe the Device Orientation configuration, and ensure that the Upside Down orientation is checked.
- Explain how the project configuration setting describes the possible orientations supported throughout an entire app, while view controllers override `supportedInterfaceOrientations` to indicate which of the possible orientations they support.
- Run the app (⌘R) and use the Simulator's Hardware menu (or ⌘→ and ⌘←) to rotate the device.

Closing

What other features might we implement to improve our simple clock?

Modifications and Extensions

- Make the colons blink every second.
- Make the text bigger when the device is in landscape orientation.
- Analyze and critique the accuracy of the displayed time and improve it.

Resources

iOS Human Interface Guidelines: The Status Bar <https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/MobileHIG/Bars.html>

Information Property List Key Reference: About Information Property List Files <https://developer.apple.com/library/ios/documentation/General/Reference/InfoPlistKeyReference/Articles/AboutInformationPropertyListFiles.html>

Information Property List Key Reference: iOS Keys <https://developer.apple.com/library/ios/documentation/General/Reference/InfoPlistKeyReference/Articles/iPhoneOSKeys.html>

Start Developing iOS Apps Today: Finding Information <https://developer.apple.com/library/ios/referencelibrary/GettingStarted/RoadMapiOS/FindingInformation.html>

Searching Developer Documentation http://developer.apple.com/library/ios/recipes/xcode_help-documentation_organizer/SearchingDocumentation/SearchingDocumentation.html

UIViewController Class Reference https://developer.apple.com/library/ios/documentation/UIKit/Reference/UIViewController_Class/

UIApplication Class Reference: UIInterfaceOrientationMask https://developer.apple.com/library/ios/documentation/UIKit/Reference/UIApplication_Class/index.html#//apple_ref/c/tdef/UIInterfaceOrientationMask

View Controller Programming Guide for iOS: Supporting Multiple Interface Orientations <https://developer.apple.com/library/ios/featuredarticles/ViewControllerPGforiPhoneOS/RespondingtoDeviceOrientationChanges/RespondingtoDeviceOrientationChanges.html>