

# Stopwatch

## Lesson 1

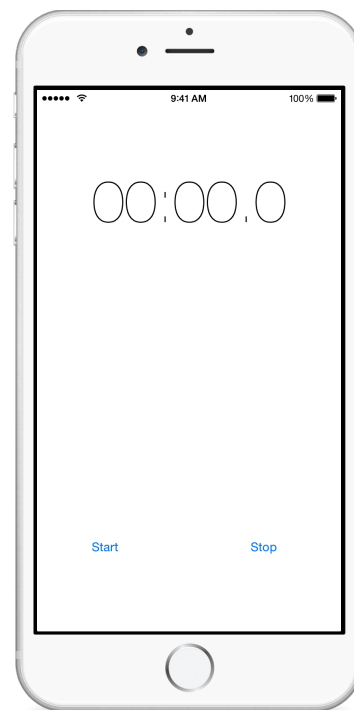


### Description

Build the application interface using Interface Builder, and learn how to adjust interface elements with the Size Inspector and Auto Layout constraints.

### Learning Outcomes

- Describe the Model-View-Controller pattern and identify the view and controller in an Xcode project.
- Analyze a user interface and subdivide it into components.
- Assemble an iOS app interface using Interface Builder.
- Use Auto Layout constraints to position multiple interface elements.
- Experiment with Interface Builder previews.



### Vocabulary

user interface	Model-View-Controller	view
view controller	Interface Builder	IB Object Library
Auto Layout	constraint	Attributes Inspector
Size Inspector		

### Materials

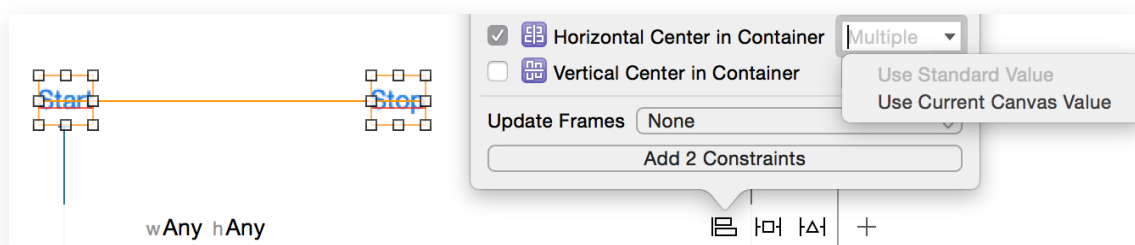
- **Stopwatch Lesson 1** Xcode project
- **Model-View-Controller** presentation

## Opening

Once we have a clear idea for an app, how do we start building the user interface?

## Agenda

- Present the concepts of Model-View-Controller, focusing on views and view controllers, and how they work together to present the UI of an iOS app.
- With Interface Builder, use the Object Library (⌘L) to add a Label for displaying the elapsed time, and change its contents to **00:00.0**.
- Use the Attributes Inspector (⌘4) and Size Inspector (⌘5) to adjust the Label position, size, color and typeface.
- Add constraints to the Label by Control-dragging upwards from the Label to the main view, and selecting *Center Horizontally in Container* to create a Center X Alignment constraint. Use the Pin tool (⇧⌘=) to create a top Vertical Space constraint.
- Resolve any constraint issues using the menu item *Editor > Resolve Auto Layout Issues > Update Frames* (⌘⇧=).
- Use the Object Library (⌘L) again to add two Buttons, labeled **Start** and **Stop**, to the main view. Adjust their size and position with the Size Inspector (⌘5).
- Add a Center Y Alignment constraint to the Buttons by Control-dragging from the Start button to the Stop button, and selecting *Center Y*. Add a Vertical Space constraint by Control-dragging downward from the Start button to the main view, and selecting *Bottom Space to Bottom Layout Guide*.
- Add Center X Alignment constraints to both buttons by selecting them both together (⌘click), and using the Align tool (⇧⌘=) to select *Horizontal Center in Container* and *Use Current Canvas Value*.



- Resolve any constraint issues using the menu item *Editor > Resolve Auto Layout Issues > Update Frames* (⌘⇧=).
- Using the Assistant Editor (⌘⇧↔), add different devices to the Preview, and observe how the interface elements adapt to the different size classes.
- Run the app (⌘R) to witness the interface displayed within the iOS Simulator.

## Closing

What are the benefits and drawbacks to starting the construction of an app with the interface first?

## Modifications and Extensions

- Delete the existing constraints, and experiment with using different constraints to achieve a desired layout that works well in multiple size classes. Use the Assistant Editor Preview to view the results.
- Add the label and buttons to the interface using only code, and critique the benefits and drawbacks of this procedural approach with the declarative approach supported by Interface Builder.

## Resources

Cocoa Core Competencies: Model-View-Controller <https://developer.apple.com/library/ios/documentation/General/Conceptual/DevPedia-CocoaCore/MVC.html>

Interface Builder Help: Building User Interfaces [https://developer.apple.com/library/ios/recipes/xcode\\_help-interface\\_builder/Chapters/AboutInterfaceBuilder.html](https://developer.apple.com/library/ios/recipes/xcode_help-interface_builder/Chapters/AboutInterfaceBuilder.html)

Start Developing iOS Applications Today: Designing the User Interface <https://developer.apple.com/library/ios/referencelibrary/GettingStarted/RoadMapiOS/DesigningaUserInterface.html>

Adaptive User Interfaces <https://developer.apple.com/design/adaptivity/>

Adding Auto Layout Constraints with the Pin and Align Tools [https://developer.apple.com/library/ios/recipes/xcode\\_help-IB\\_auto\\_layout/chapters/pin-constraints.html](https://developer.apple.com/library/ios/recipes/xcode_help-IB_auto_layout/chapters/pin-constraints.html)

Auto Layout Guide: Working with Constraints in Interface Builder <https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/AutolayoutPG/WorkingwithConstraints/WorkingwithConstraints.html>

About Designing for Multiple Size Classes [https://developer.apple.com/library/ios/recipes/xcode\\_help-IB\\_adaptive\\_sizes/chapters/AboutAdaptiveSizeDesign.html](https://developer.apple.com/library/ios/recipes/xcode_help-IB_adaptive_sizes/chapters/AboutAdaptiveSizeDesign.html)

View Programming Guide for iOS [https://developer.apple.com/library/ios/documentation/WindowsViews/Conceptual/ViewPG\\_iPhoneOS/](https://developer.apple.com/library/ios/documentation/WindowsViews/Conceptual/ViewPG_iPhoneOS/)

Interface Builder Help: Previewing Your Layout [https://developer.apple.com/library/ios/recipes/xcode\\_help-interface\\_builder/Chapters/PreviewingLayouts.html#//apple\\_ref/doc/uid/TP40009971-CH5-SW1](https://developer.apple.com/library/ios/recipes/xcode_help-interface_builder/Chapters/PreviewingLayouts.html#//apple_ref/doc/uid/TP40009971-CH5-SW1)