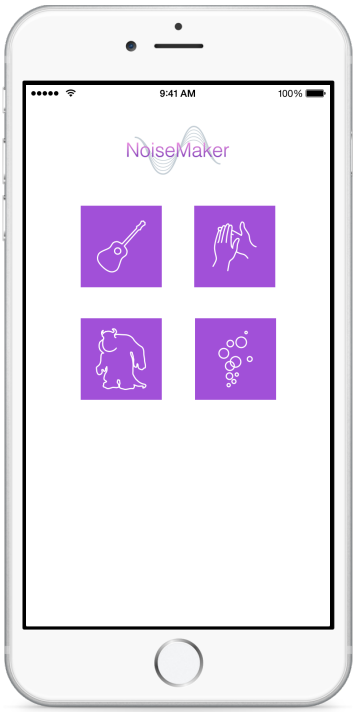
# NoiseMaker
# Lesson 10

## Description

Add a custom title image to the interface and customize each button's appearance and accessibility attributes.




## Learning Outcomes

- Discover how Xcode and iOS manage image asset to accommodate different size classes.
- Practice creating image sets and adding image files to an image set.
- Practice customizing the appearance of buttons with images.
- Assess the usability of an interface, and apply accessibility features to accommodate a wide audience of users.

## Vocabulary

| | | |
|---|---|---|
| asset catalog | size class | image set |
| Project Navigator | resolution | launch screen |
| Auto Layout constraint | Attributes Inspector | accessibility |
| Identity Inspector | | |

## Materials

- **NoiseMaker Lesson 10** Xcode project
- **NoiseMaker Images** archive

# Opening

How can we add our own images to the app and customize how the buttons look?

# Agenda

- Explain how Xcode provides asset catalogs to group and manage assets such as images, that can adapt to different size classes.
- Using the Project Navigator (⌘1), select the **Images.xcassets** asset catalog.
- Use the Add control (+) to add a new image set to the catalog.
- Rename the image set to **ApplauseIcon**.
- With the ApplauseIcon image set selected, observe how Xcode displays three different wells for three different sizes of a single image.
- Discuss how the different resolutions of devices and size classes might use the different images within an image set.
- Drag the supplied **Applause.png** , **Applause@2x.png**, and **Applause@3x.png** image files from within the OS X Finder to the **1x** , **2x** and **3x** wells in the ApplauseIcon image set.
- Create four more image sets called **BubblesIcon**, **GuitarIcon**, **MonsterIcon**, and **Title**, and drag each image file into their respective **1x**, **2x** and **3x** image wells.
- Using the Project Navigator (⌘1), select **LaunchScreen.xib**, and observe the default app launch screen appear in Interface Builder.
- Delete the NoiseMaker label, and use the Object Library (⌥⌘L) to add an Image View to the center of the launch screen.
- Select the Image View on the canvas, use the Attributes Inspector (⌥⌘4) to set the **Image** attribute to **Title**, and observe that the Title image appears on the canvas.
- Use the Align control (⫿) to center the image view horizontally and vertically, and use the menu item *Editor* > *Resolve Auto Layout Issues* > *Update Frames* (⌥⌘=) to establish appropriate constraints.
- Run the app (⌘R), and observe that the Title image appears on the launch screen in the Simulator.
- Using the Project Navigator (⌘1), select **Main.storyboard**, and use the Object Library (⌥⌘L) to add an Image View to the middle of the top of the interface.
- Select the Image View on the canvas, use the Attributes Inspector (⌥⌘4) to set the **Image** attribute to **Title**, and observe that the Title image appears on the canvas.
- Use the Align control (⫿) to add a horizontal alignment constraint, Control-drag upwards from the image view to the main view to create a vertical spacing constraint, and use the menu item *Editor* > *Resolve Auto Layout Issues* > *Update Frames* (⌥⌘=) to correct the size.

- Run the app (⌘R), and observe the image appear in the interface.
- Using Interface Builder, select a button and open the Attributes Inspector (⌥⌘4). Set the **Image** attribute to the corresponding image (e.g., GuitarIcon). Within the **View** panel of the Attributes Inspector (⌥⌘4), change the **Background Color** attribute to a desired color. Delete the word in the **Title** field (e.g., Guitar), and adjust the position and size of the button to a square large enough to accommodate the image (e.g., 100 x 100).
- Use the menu item *Editor* > *Resolve Auto Layout Issues* > *Update Constraints* (⇧⌘=) to accommodate the new size and position. Use the Pin control (⊦□⊦) to add height and width constraints (e.g. 100), and use the menu item *Editor* > *Resolve Auto Layout Issues* > *Update Frames* (⌥⌘=) to resolve any remaining Auto Layout issues.
- Repeat the above actions to update each button's appearance, adjust each button position as necessary, and use the menu item *Editor* > *Resolve Auto Layout Issues* > *Update Constraints* (⇧⌘=) to match the constraints to the visible layout on the canvas.
- Run the app (⌘R), observe the customized buttons, and tap each button to play a sound.
- Discuss how the buttons now lack text, and how this might affect accessibility.
- Present the concept of user experience and accessibility, via the Apple Accessibility web resources.
- Using Interface Builder, select each button and open the Identity Inspector (⌥⌘3). In the Accessibility Pane, provide each button a **Label** (e.g. Guitar), **Hint** (e.g. Plays guitar sound), and ensure that the **Plays Sound** Trait is checked.
- Explain how iOS accessibility features can use the button Accessibility attributes to describe each button for visually-impaired users.

# Closing

What other button attributes can we customize, and how do they affect the button appearance? How might you add your own sounds and button images to the app?

# Modifications and Extensions

- Investigate the purpose of the **@2x** and **@3x** image files and how iOS uses them.
- Use the different button State Configurations within the Attributes Inspector to change the way the buttons appear when tapped.
- Change the device orientation within the iOS Simulator and modify the app so that the interface appears correctly in different orientations.
- Explore the Apple Human Interface Guidelines (HIG) and critique the user NoiseMaker interface.

- Add your own custom sounds and buttons to create a musical instrument, such as a digital drum machine. Ensure that the app is accessible as possible to all users.

# Resources

Asset Catalogs https://developer.apple.com/library/ios/recipes/xcode_help-image_catalog-1.0/Recipe.html

Interface Builder Help: Configuring Object Attributes https://developer.apple.com/library/ios/recipes/xcode_help-IB_objects_media/Chapters/ObjectAttributes.html

iOS Human Interface Guidelines https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/MobileHIG/index.html

iOS Human Interface Guidelines: Icon and Image Sizes https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/MobileHIG/IconMatrix.html

UIKit User Interface Catalog: Buttons https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/UIKitUICatalog/UIButton.html

UIButton Class Reference https://developer.apple.com/library/ios/documentation/UIKit/Reference/UIButton_Class/index.html

Accessibility for Developers: Accessibility on iOS https://developer.apple.com/accessibility/ios/