Mobile Application Development
Aileen Pierce

ADAPTIVE LAYOUT

Adaptive Apps

- Adaptive apps are flexible enough to be run on different devices and in different orientations
- Design your user interface in Interface Builder to adapt to any device size or orientation
  - Auto layout
  - Constraints
  - Size classes

Device Orientation

- iOS devices can be used in portrait or landscape mode.
- iPhone
  - Most apps support autorotation but not all
  - Upside down portrait is not usually supported
- iPad
  - Recommended that all apps support every orientation
- If autorotation makes sense for your app you should support it.

Device Orientation

- Interface orientation can be set for the whole application that will be the default for all view controllers
- You can also set valid orientations for each view controller.
- When the user rotates the device the active view controller will be asked if the new orientation is supported.
- If it is then the app's window and view will be rotated and resized to fit the new orientation.

Device Orientation

- **`supportedInterfaceOrientations()`** is called when a user rotates the device
  - Called on the active view controller
  - Can return acceptable orientations
  - Every view controller can implement this differently
  - Allows you to decide acceptable orientations for different view controllers

Device Orientation

- There are four defined iOS interface orientations values for UIInterfaceOrientation
  - **`UIInterfaceOrientationUnknown`**
  - **`UIInterfaceOrientationPortrait`**
  - **`UIInterfaceOrientationPortraitUpsideDown`**
  - **`UIInterfaceOrientationLandscapeLeft`**
  - **`UIInterfaceOrientationLandscapeRight`**

Auto Layout

- The system that helps you design the layout of your user interface
- Creates a user interface that responds to different screen sizes, orientations, and localization
- Auto layout enables you to design by intent
  - Express what you want your interface to accomplish but not how
  - Create relationships between UI elements called constraints

Constraints

- Constraints express rules for the relationship between UI elements in your interface
- Auto Layout uses these rules to automatically create your user interface when your app runs
- Creating constraints
  - IB can add suggested or missing constraints based on your interface layout
  - Create your own constraints

Constraints

- For each element there must be enough constraints to determine its size, horizontal position, and vertical position
- Constraint properties
  - Attributes: leading and trailing are the default
  - Values: size or offset of constraint in points
  - Relation: type of relationship between elements
  - Priority: required, high, or low

Constraints

- Align: align edges or centers of elements to each other or within the container
- Pin: sets width or height spacing for an element or to a view
  - Leading space (from left for English)
  - Trailing space (from right for English)
  - Top
  - Bottom

Constraints

- Constraints are color coded
  - Blue: constraints are valid
  - Orange: constraints are misplaced or ambiguous
    - Dotted orange box shows constraints with mismatched views
  - Red: constraints are conflicting
- Constraint issues are also listed in the document outline and provides suggested fixes

Editing Constraints

- Change the constant, relation, and priority of an existing constraint
  - Double click on the constraint
  - Select an element in IB and view all its constraints in the size inspector
  - Select the constraint and edit its properties in the attributes inspector

Resolving Auto Layout Issues

- Add missing constraints to have Auto Layout add constraints based on your layout
- Clear or reset constraints
- Update frames to satisfy existing constraints
- Update constraints to update them to match your current layout
- Delete a conflicting constraint

Auto Layout

- Preview assistant editor let's you preview devices and orientations in Xcode without running the simulator
- Test your user interface in all device sizes and orientations your app will be supporting
- Choose the resizeable iPhone or iPad scheme when running the simulator

Size classes

- Size classes enable a storyboard to work with all iOS device screen sizes and orientations
- Build your interface as it will look on most devices and orientations
- Change the interface for different size classes

Size classes

- Change your interface for specific form factors
  - Size or position of a view
  - Add or remove a view or constraint
  - Change fonts
- Class widths
  - Any
  - Regular
  - Compact

Size classes

- When you change the size class any changes you make are only for that size class
  - Base values: w Any h Any
  - iPhone portrait: w Compact h Regular
  - iPhone landscape: w Any h Compact
  - iPad: w Regular h Regular