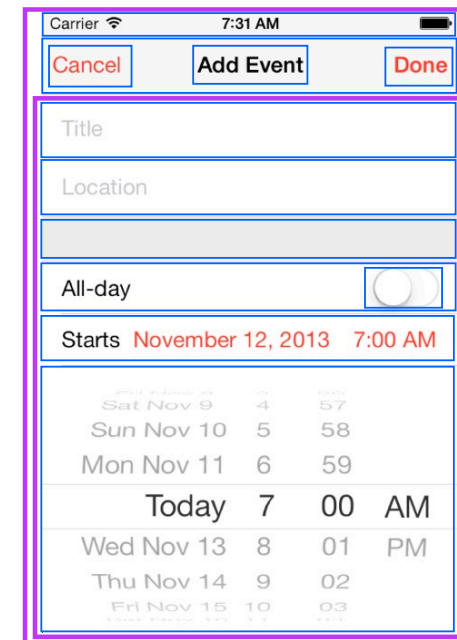
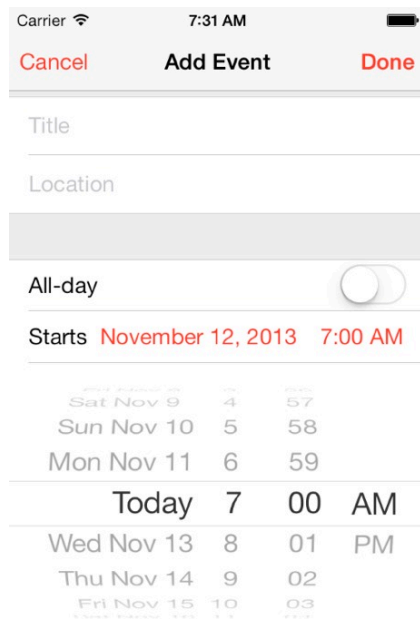


View Concepts

iPhone Application Programming Lecture 4: User Interface Design

Chat Wacharamanotham
Media Computing Group
RWTH Aachen University
Winter Semester 2013/2014
<http://hci.rwth-aachen.de/iphone>

- SDK provide many types of Views to show your content
- At run-time Views are organized as a tree
- Use Interface Builder to design your UI and connect it to code
- Geometry of Views are determined by constraints



Finding the Right View

Bars
The Status Bar
Navigation Bar
Toolbar
Toolbar and Navigation Bar Buttons
Tab Bar
Tab Bar Icons
Search Bar
Scope Bar
Content Views
Activity
Activity View Controller
Collection View
Container View Controller
Image View
Map View
Rage View Controller
Popover (iPad Only)
Scroll View
Split View Controller (iPad Only)
Table View
Text View
Web View
Controls
Activity Indicator
Contact Add Button
Date Picker
Detail Disclosure Button
Info Button
Network Activity Indicator
Page Control
Picker
Progress View
Refresh Control
Rounded Rectangle Button
Segmented Control
Slider
Stepper
Switch
System Button
Text Field
Temporary Views
Alert
Action Sheet
Modal View

Label

A label displays static text.

Create a stream or join one to share your best shots and enjoy friends' comments and contributions right in the iOS photos app.

API NOTE

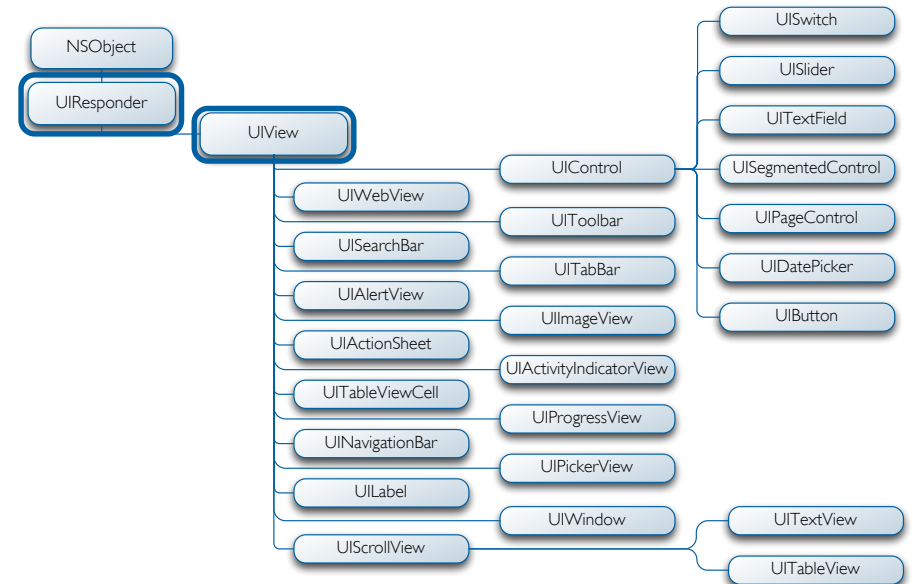
To learn more about defining labels in your code, see [UILabel Class Reference](#).

A label:

- Displays any amount of static text
- Doesn't allow user interaction except, potentially, to copy the text

Use a label to name or describe parts of your UI or to provide short messages to the user. A label is best suited for displaying a relatively small amount of text.

Take care to make your labels legible. It's best to support Dynamic Type and use the `UIFont` method `preferredFontForTextStyle` to get the text for display in a label. If you choose to use custom fonts, don't sacrifice clarity for fancy lettering or showy colors. (For guidelines about using text in an app, see [Colors and Typography](#); to learn more about Dynamic Type, see "Text Styles" in [Text Programming Guide for iOS](#).)



View Concepts

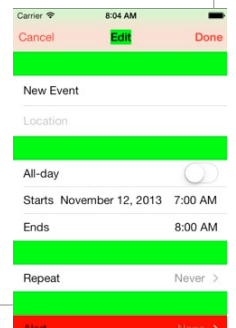
✓ SDK provide many types of Views to show your content

- At run-time Views are organized as a tree
- Use Interface Builder to design your UI and connect it to code
- Geometry of Views are determined by constraints

Demo: Hacking Calendar

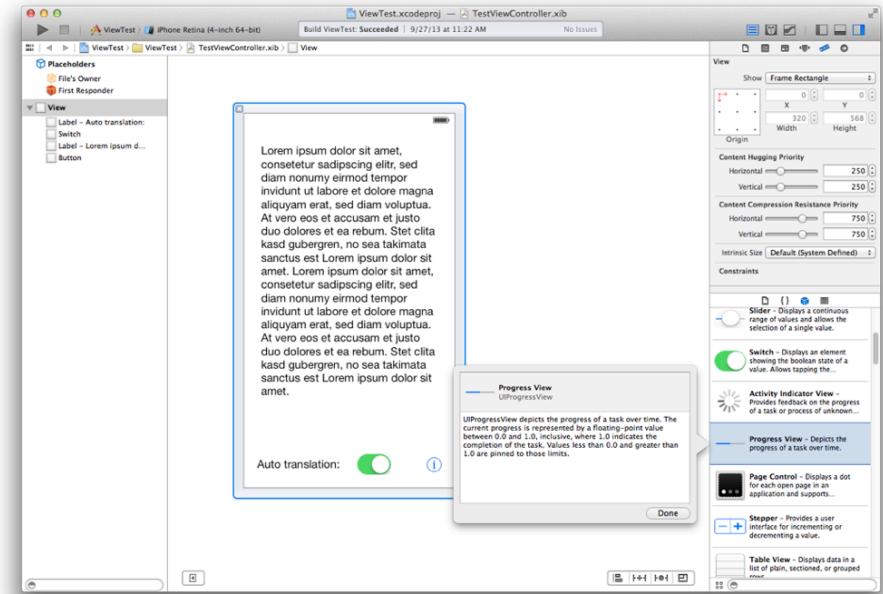
```

(lldb) po [[UIWindow keyWindow] recursiveDescription]
<UIWindow
| <UILayoutContainerView
| | <UINavigationController
| | | <UIViewControllerWrapperView
| | | | <EKCalendarItemEditorTableView: 0xf870200; baseClass = UITableView;
| | | | | <UITableViewWrapperView
| | | | | | <UITableViewCell: 0xb19ad90
| | | | | ...
| | | <UINavigationController
| | | ...
| | | <UINavigationControllerItemView
| | | | <UILabel
| | | | <UINavigationControllerItemView
| | | | ...
(lldb) expr ((UIView *)0xb19ad90).backgroundColor = [UIColor redColor]
(lldb) expr ((UIView *)0xf870200).backgroundColor = [UIColor greenColor]
    
```



View Concepts

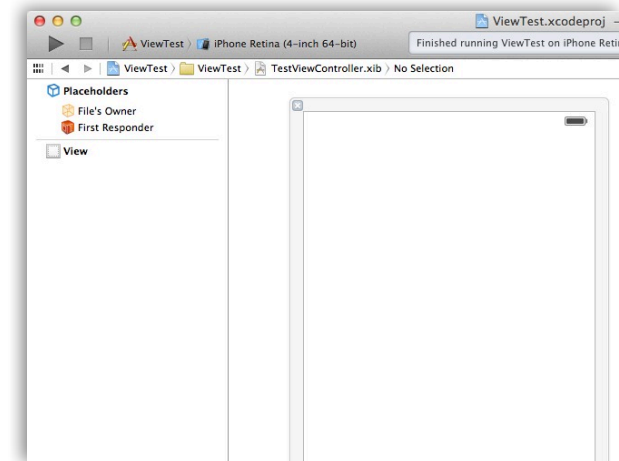
- ✓ SDK provide many types of Views to show your content
- ✓ At run-time Views are organized as a tree
 - Use Interface Builder to design your UI and connect it to code
 - Geometry of Views are determined by constraints



Interface Builder

- Graphical tool to layout user interfaces
- Create the widget hierarchy
- Set attributes of widgets
- Set up connections between the widgets
- Store these informations in nib files

The Anatomy of a xib File



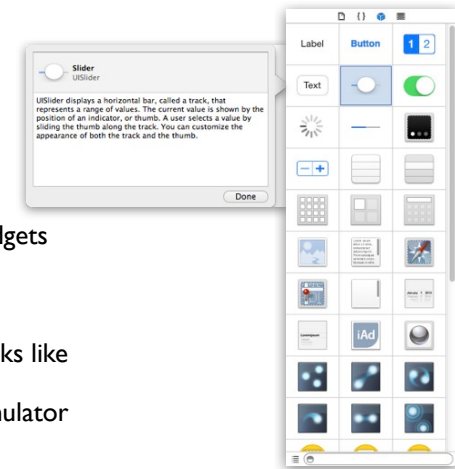
The Source of a xib

```
<?xml version="1.0" encoding="UTF-8"?>
<archive type="com.apple.InterfaceBuilder3.CocoaTouch.XIB" version="8.00">
  <data>
    <int key="IBDocument.SystemTarget">1280</int>
    <string key="IBDocument.SystemVersion">11C74</string>
    <string key="IBDocument.InterfaceBuilderVersion">1938</string>
    <string key="IBDocument.AppKitVersion">1138.23</string>
    <string key="IBDocument.HIToolboxVersion">567.00</string>
    <object class="NSMutableDictionary" key="IBDocument.PluginVersions">
      <string key="NS.key.0">com.apple.InterfaceBuilder.IBCocoaTouchPlugin</string>
      <string key="NS.object.0">933</string>
    </object>
    <array key="IBDocument.IntegratedClassDependencies">
      <string>IBUITextView</string>
      <string>IBUISwitch</string>
      <string>IBUIButton</string>
      <string>IBUIView</string>
      <string>IBUILabel</string>
      <string>IBProxyObject</string>
    </array>
    <array key="IBDocument.PluginDependencies">
      <string>com.apple.InterfaceBuilder.IBCocoaTouchPlugin</string>
    </array>
    <object class="NSMutableDictionary" key="IBDocument.Metadata">
      <string key="NS.key.0">PluginDependencyRecalculationVersion</string>
      <integer value="1" key="NS.object.0">
    </object>
    <array class="NSMutableArray" key="IBDocument.RootObjects" id="1000">
      <object class="IBProxyObject" id="841351856">
        <string key="IBProxiedObjectIdentifier">IBFilesOwner</string>
        <string key="targetRuntimeIdentifier">IBCocoaTouchFramework</string>
      </object>
      <object class="IBProxyObject" id="371349661">
        <string key="IBProxiedObjectIdentifier">IBFirstResponder</string>
        <string key="targetRuntimeIdentifier">IBCocoaTouchFramework</string>
      </object>
      <object class="IBUIView" id="474857837">
        <reference key="NSNextResponder">
        <int key="NSvFlags">292</int>
        <array class="NSMutableArray" key="NSSubviews">
          <object class="IBUITextView" id="694905917">
            <reference key="NSNextResponder" ref="474857837">
            <int key="NSvFlags">274</int>
            <string key="NSFrame">{{20, 20}, {280, 385}}</string>
            <reference key="NSSuperview" ref="474857837">
          </object>
        </array>
      </object>
    </array>
  </data>
</archive>
```

13 iPhone Application Programming • Prof. Jan Borchers



Laying out the User Interface

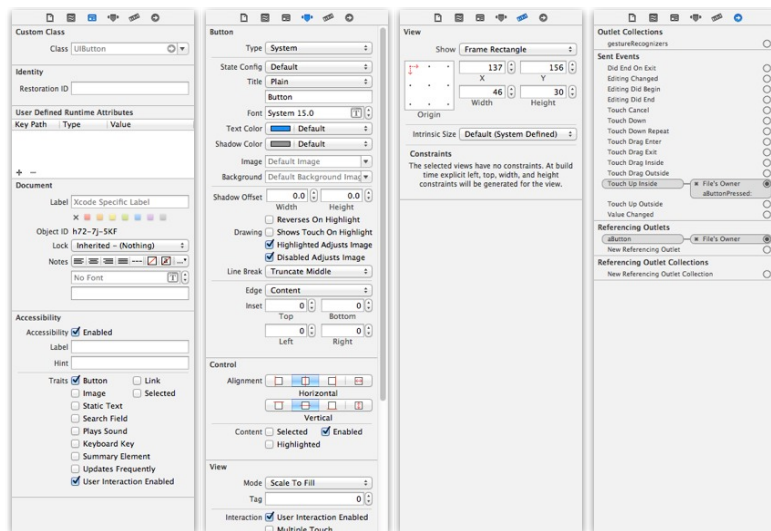


- The library contains all UI Widgets
- Drag them to your view
- See instantly what your UI looks like
- Test your UI in the iPhone Simulator

14 iPhone Application Programming • Prof. Jan Borchers



Set Widget Attributes



15 iPhone Application Programming • Prof. Jan Borchers



Connecting Widgets and Code

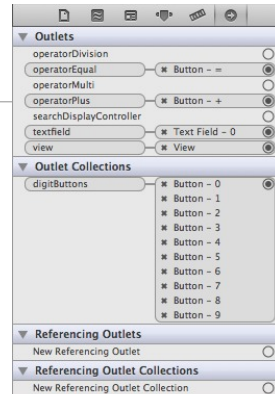
- IBActions
 - Tags a method as a target for an interface action
- IBOutlets
 - IBOutletCollection
 - Variables to populate with objects from a nib file

16 iPhone Application Programming • Prof. Jan Borchers



Connecting Widgets and Code

```
@interface iCalcView: UIView {  
}  
- (IBAction)addDigit:(id)sender;  
- (IBAction)calculateResult:(id)sender;  
  
//declared properties  
//textfield  
@property (nonatomic,weak) IBOutlet UITextField *textfield;  
//operators: + =  
@property (nonatomic,weak) IBOutlet UIButton *operatorPlus  
@property (nonatomic,weak) IBOutlet UIButton *operatorEqual;  
//digits  
@property (nonatomic,weak) IBOutletCollection (UIButton) NSArray* digitButtons;
```



Interface Builder Demo

View Concepts

- ✓ SDK provide many types of Views to show your content
- ✓ At run-time Views are organized as a tree
- ✓ Use Interface Builder to design your UI and connect it to code
- Geometry of Views are determined by constraints

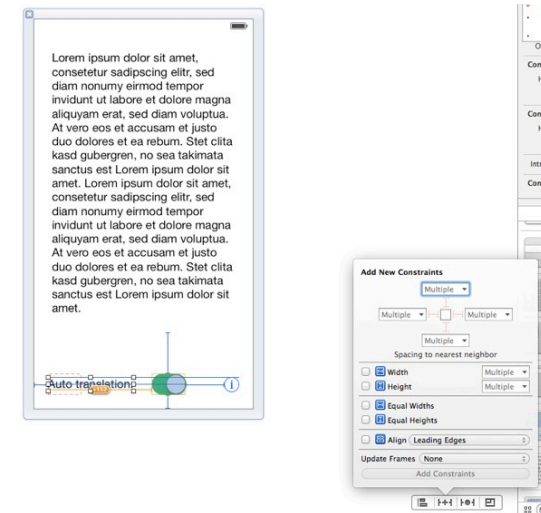
Auto Layout

- Preferred layout management
- Allows you to create views that work both in portrait and landscape mode
- Available in iOS 6 and higher
- Spatial relationships expressed by constraints

Auto Layout Constraints

- Constraints are mathematical expressions
 - \leq , $=$, \geq
- Constraints have a priority level
- The runtime tries to solve the system of equations

Adding Constraints



View Concepts

- ✓ SDK provide many types of Views to show your content
- ✓ At run-time Views are organized as a tree
- ✓ Geometry of Views are determined by constraints
- ✓ Use Interface Builder to design your UI and connect it to code

UIView

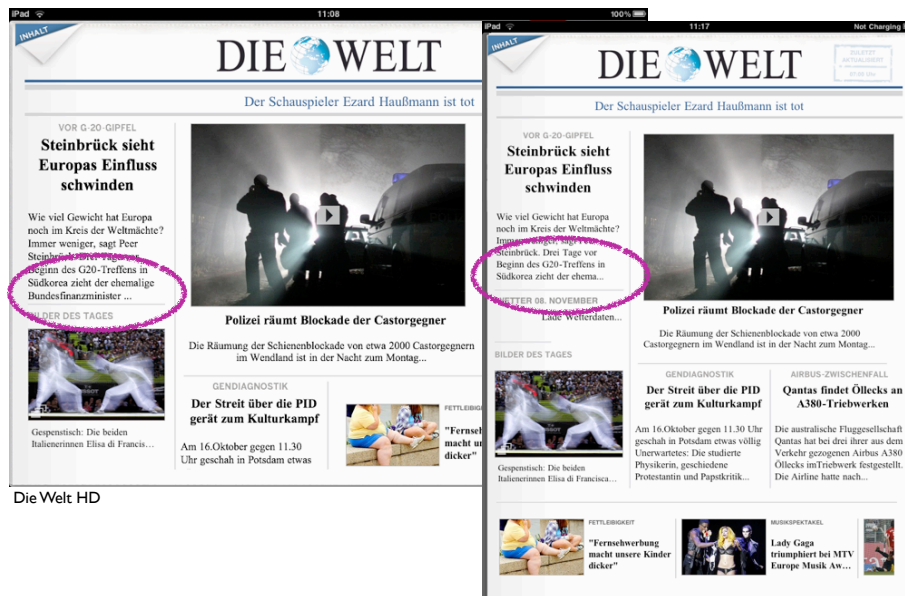
- Defines a rectangular area on the screen
- Two responsibilities
 - Render content
 - React to user input
 - Manage subviews
- Layout as view hierarchy



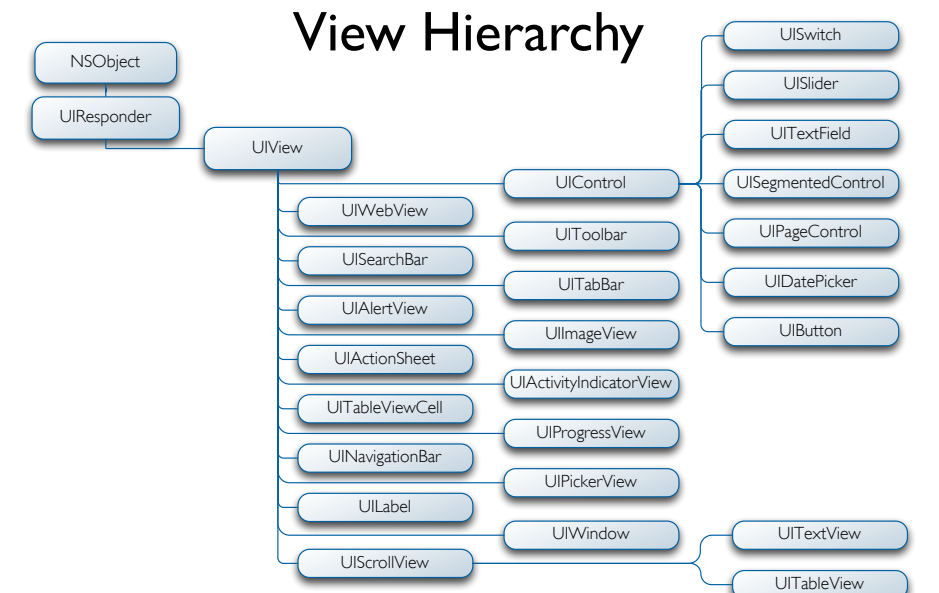
25 iPhone Application Programming • Prof. Jan Borchers



26 iPhone Application Programming • Prof. Jan Borchers

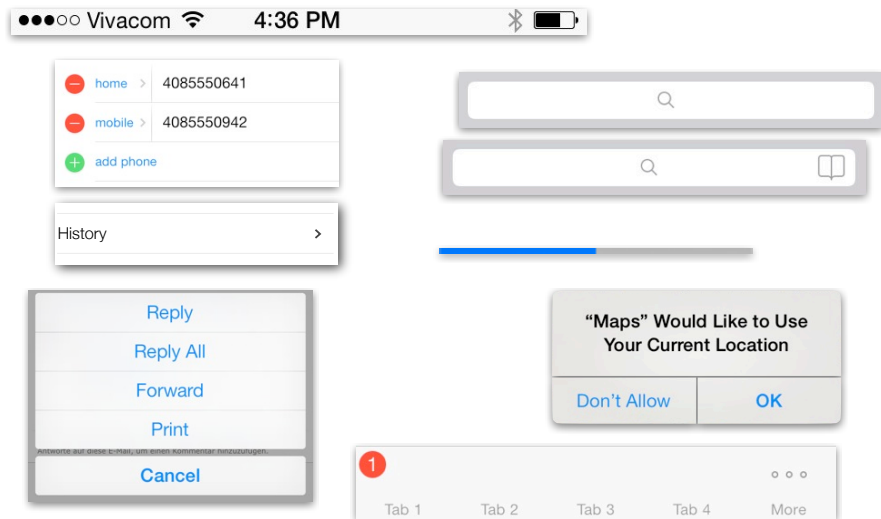


27 iPhone Application Programming • Prof. Jan Borchers



28 iPhone Application Programming • Prof. Jan Borchers

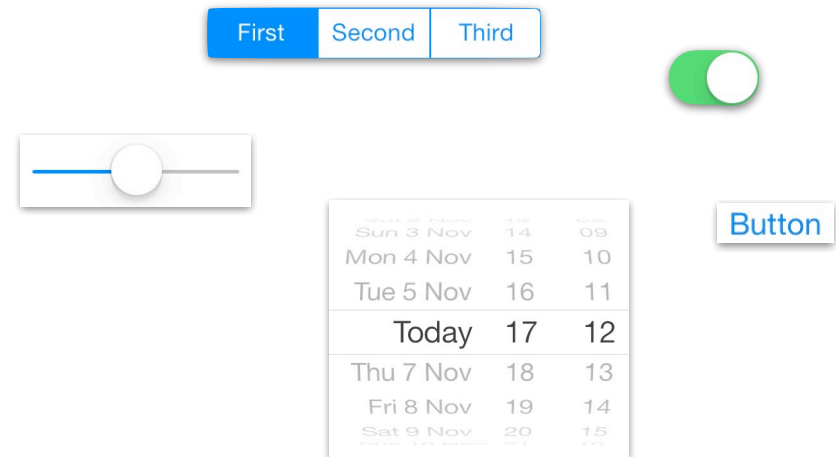
UI Widgets



29 iPhone Application Programming • Prof. Jan Borchers



Control Widgets



30 iPhone Application Programming • Prof. Jan Borchers



View Categories: Examples

- Containers: [UIScrollView](#)
- Controls: [UIButton](#)
- Display views: [UIImageView](#)
- Text and web views: [UITextView](#)
- Alert views: [UIAlertView](#)
- Navigation views: [UITabBar](#)
- UIWindow

31 iPhone Application Programming • Prof. Jan Borchers



UIViewController

- Manages typically one screen
- Flushes the view on low-memory situations
- Resizes the view on orientation change
- Creates modal views on top of the current view

32 iPhone Application Programming • Prof. Jan Borchers

