



# STORYBOARDING

OUDL iOS mini-workshop  
December 6, 2012 @ [TheBoxJelly.com](http://TheBoxJelly.com)



How to Build HTTP-compliant Client/Server



git &  
github



The  
**Lambda**  
CALCULUS

Introduction to Haskell



THINGS  
IDIOMAS 有用的  
делать

# THANKS



**HIC**  **PACITY**

**HTDC** *Cultivating Hawaii's tech sector*  
HIGH TECHNOLOGY DEVELOPMENT CORPORATION

*Aloha*  
*Startups*

# Greetings

Kyle Oba

@mudphone

mudphone on #hicapacity

# OBJECTIVES

- PROTOTYPING: intro, tool, techniques
- WHY
- WHAT: storyboard basics
- BUILD: & sprinkled throughout
- !!! CODE !!!
- RESOURCES

# PROTOTYPING



Tuesday, December 11, 12

## INTRODUCTION – Main Objectives:

- Prototypes for iOS apps
- Learn Storyboarding

# Why use prototyping?

- Communication
- Common Model
- Dog & Pony
- UX Testing

Tuesday, December 11, 12

Communication – Get an idea out and communicate it to your team.

Common Model – Teams work better if they have a common model. A common visual vocabulary.

Dog & Pony – Sharing a concept with clients, customers, investors, God.

UX Testing – Checking the feel of a UI, interactions.

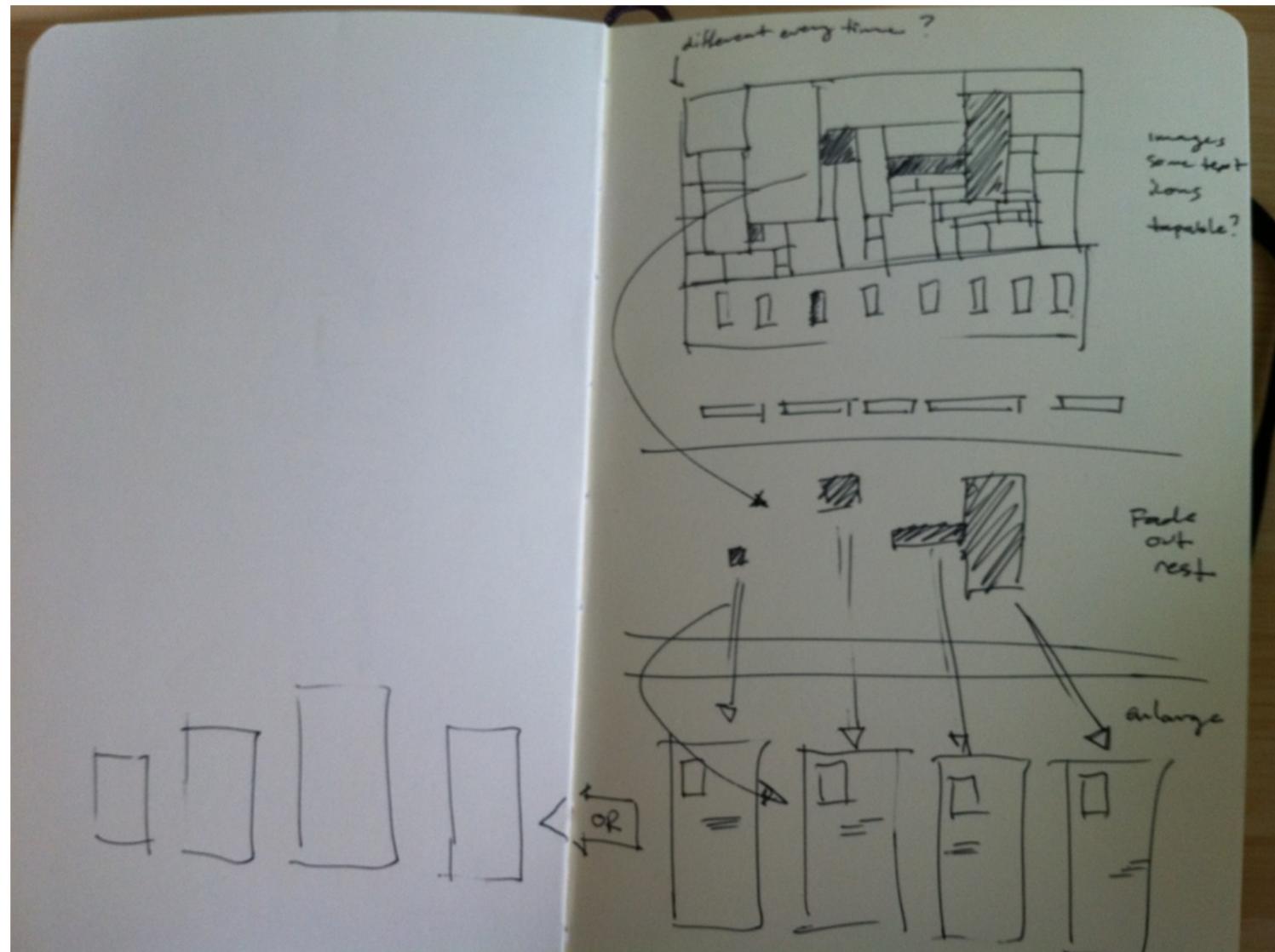
# A desirable process

- Easy Creation
- Accessible
- Fast Output
- Low Cost

Tuesday, December 11, 12

Sometimes these conflicts with your goals.  
They're not necessarily intrinsic in prototyping.  
Compromises

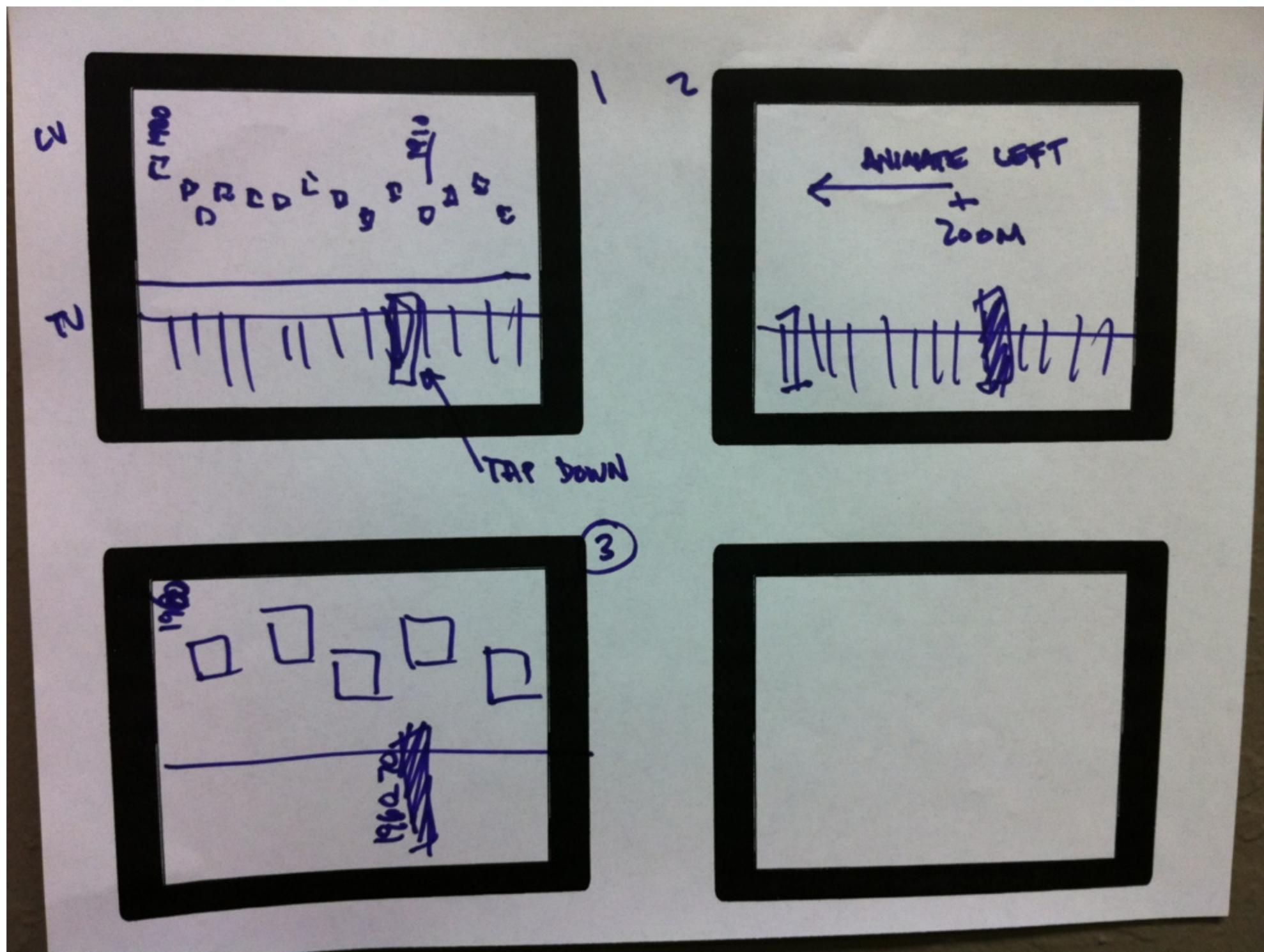
# Dead Trees



Tuesday, December 11, 12

Moleskine optional

# Homemade Worksheets

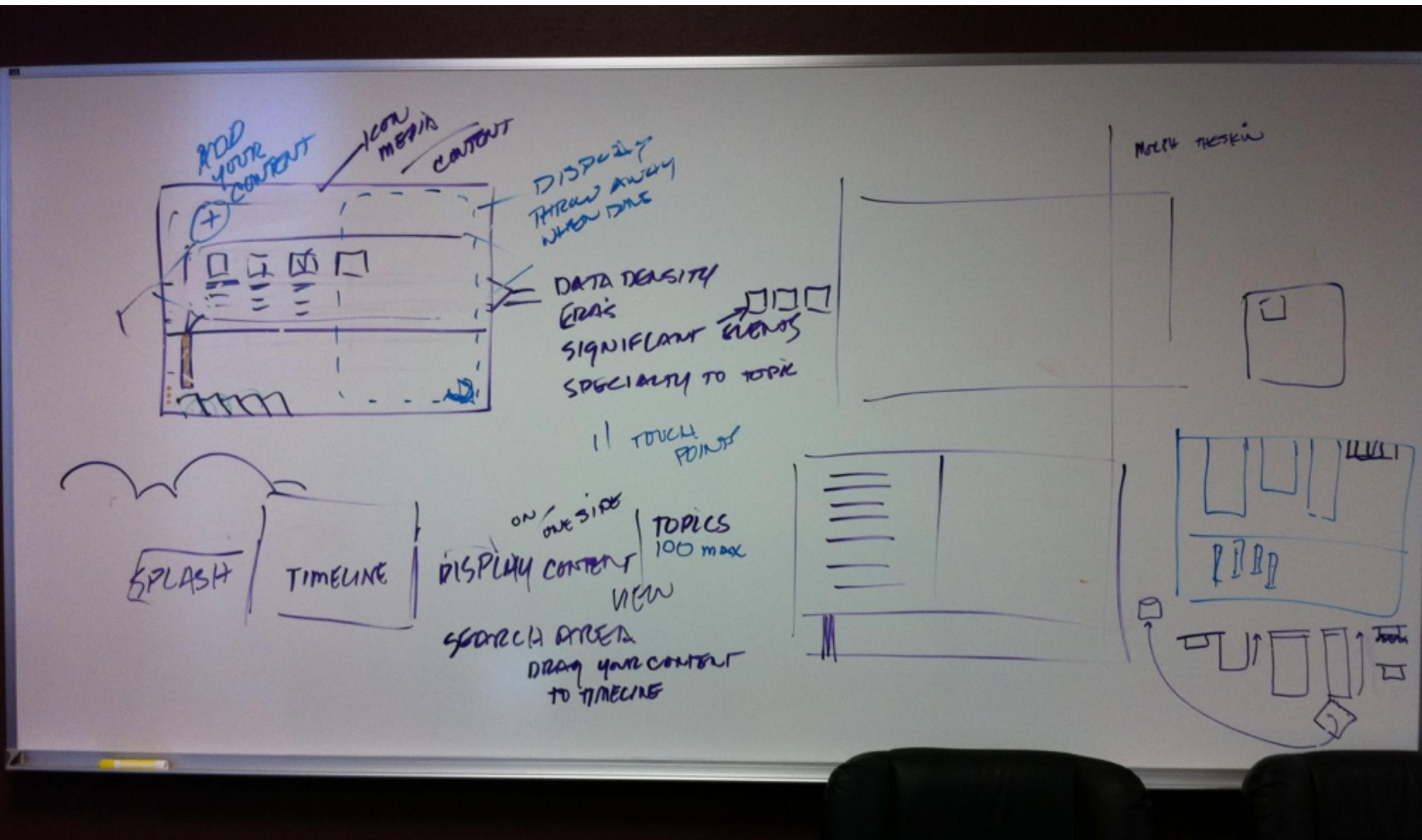


Tuesday, December 11, 12

Downloadable templates, for printing, photocopying

[UIStencils.com](http://UIStencils.com)

# White Boards



Tuesday, December 11, 12

Showerboard can be had for less than \$20 a panel at HD/Lowe's

# UI Stencils



Tuesday, December 11, 12

stencils, mockup notepads, and sticky notes

See also, AI and PS stencils

# Balsamiq Mockups



Tuesday, December 11, 12

# Keynotopia



Tuesday, December 11, 12

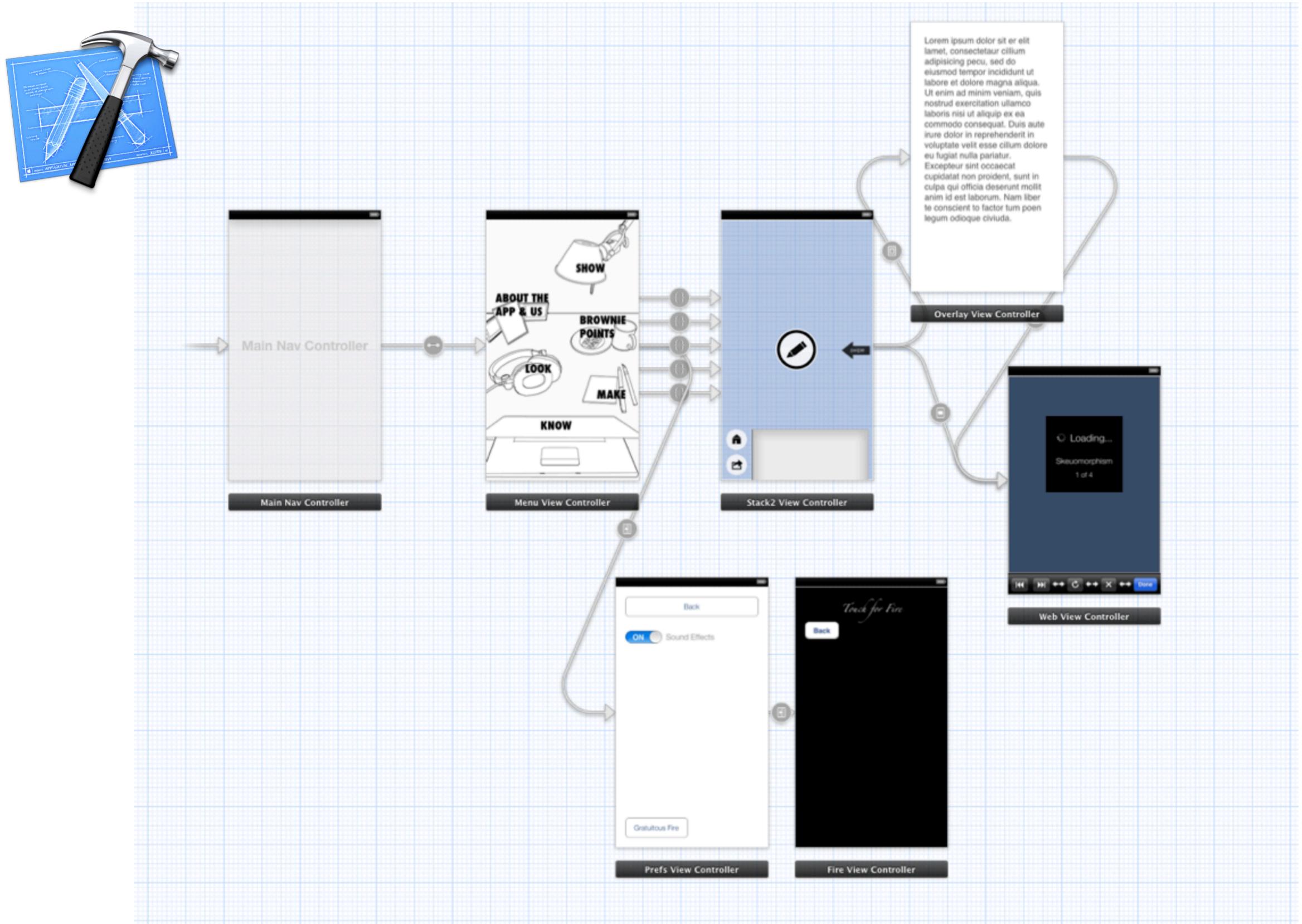
Requires Keynote on Mac and on device

# Storyboards, in general...

Tuesday, December 11, 12

Creates a visual outline describing scenes  
and relationships between scenes

# Xcode Storyboards



Tuesday, December 11, 12

Uses scenes and segue/transition metaphor of movie storyboards

This is a real app. It uses the same transitions and UI elements

Interactions are real (slider, button press)

Animations are real (switch)

Transitions have the same timing

Can be used as a basis for the final product

# Xcode Storyboards

**“The right tool is the one you will use.”**

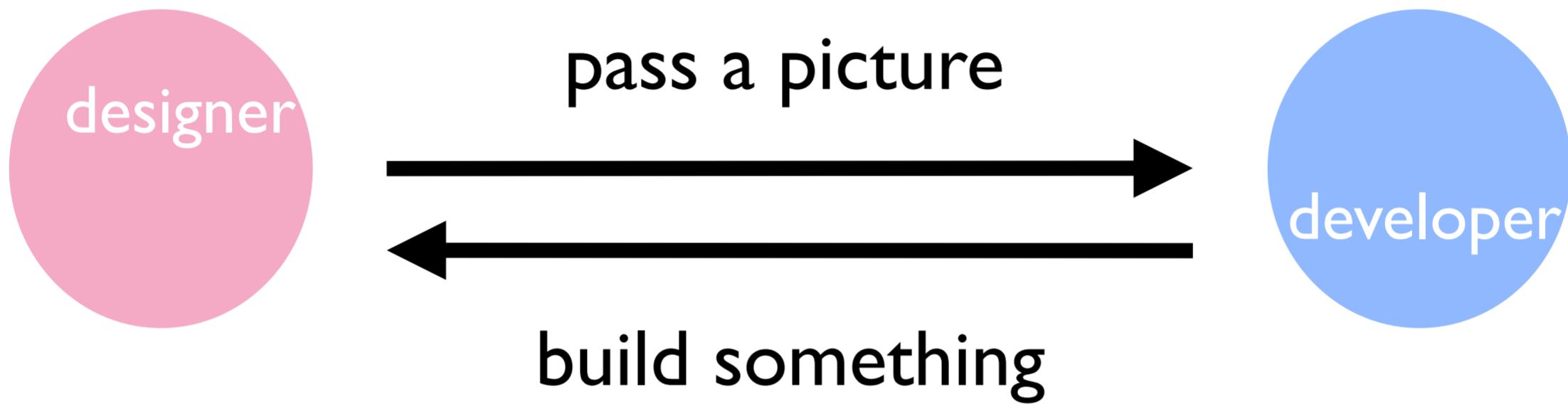
<https://blog.bloomfire.com/posts/487150-the-right-tool-is-the-one-you-will-use/public>

Tuesday, December 11, 12

We use some combination of all these tools.

If your team doesn't learn Storyboarding, you have to pass a picture from the design team to the development team.

# This happens



Tuesday, December 11, 12

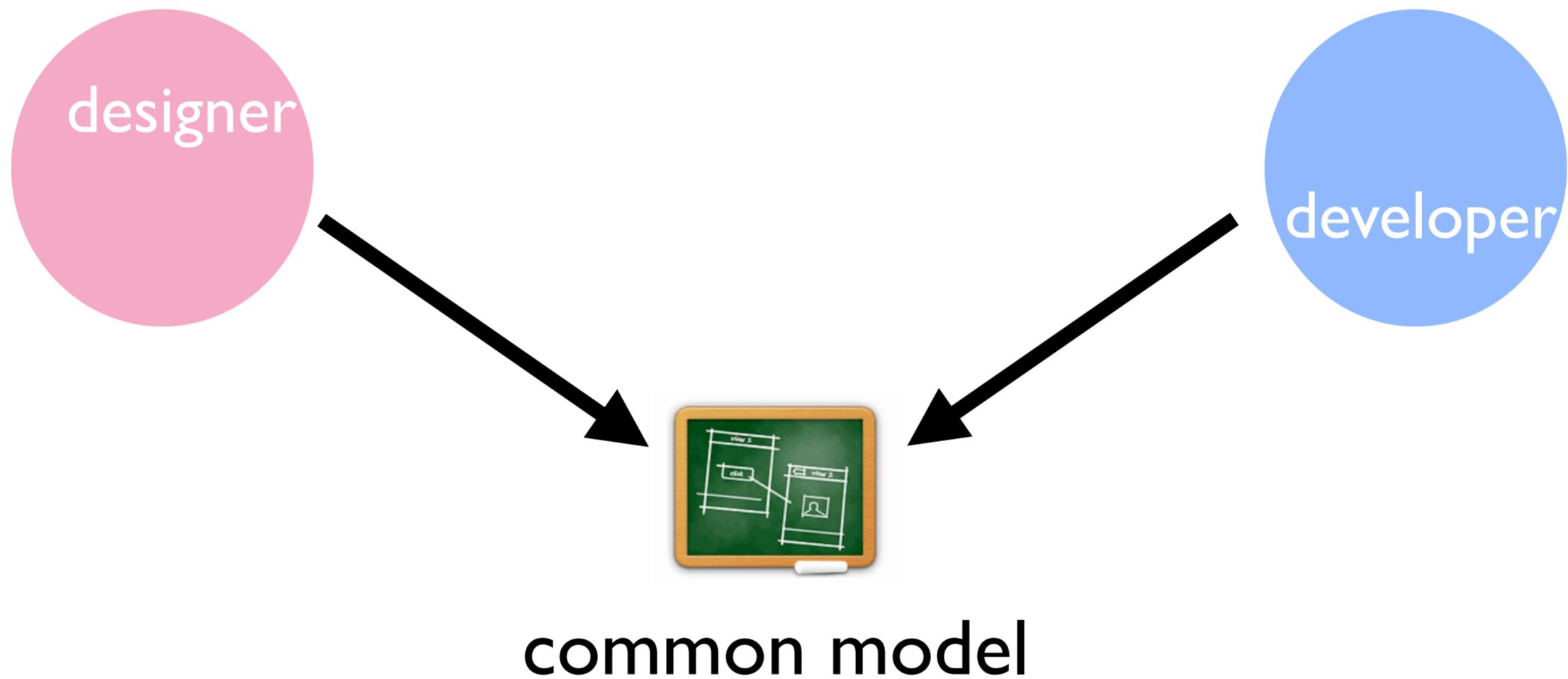
This happens (and it's fine)

Rinse, repeat

This is why the apartment I live in is horrible

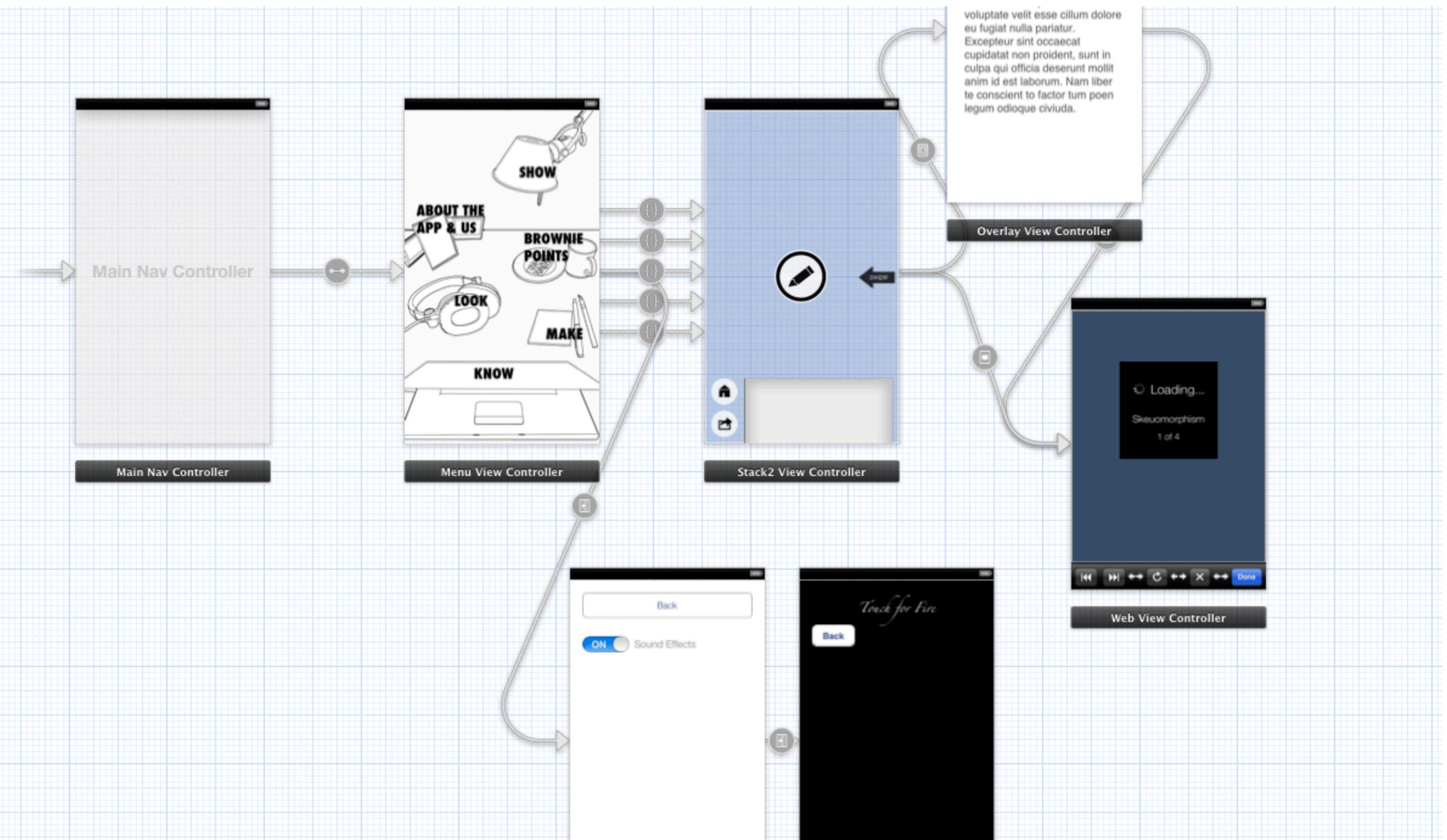
The greatness of our product is limited by these cycles

# This is better

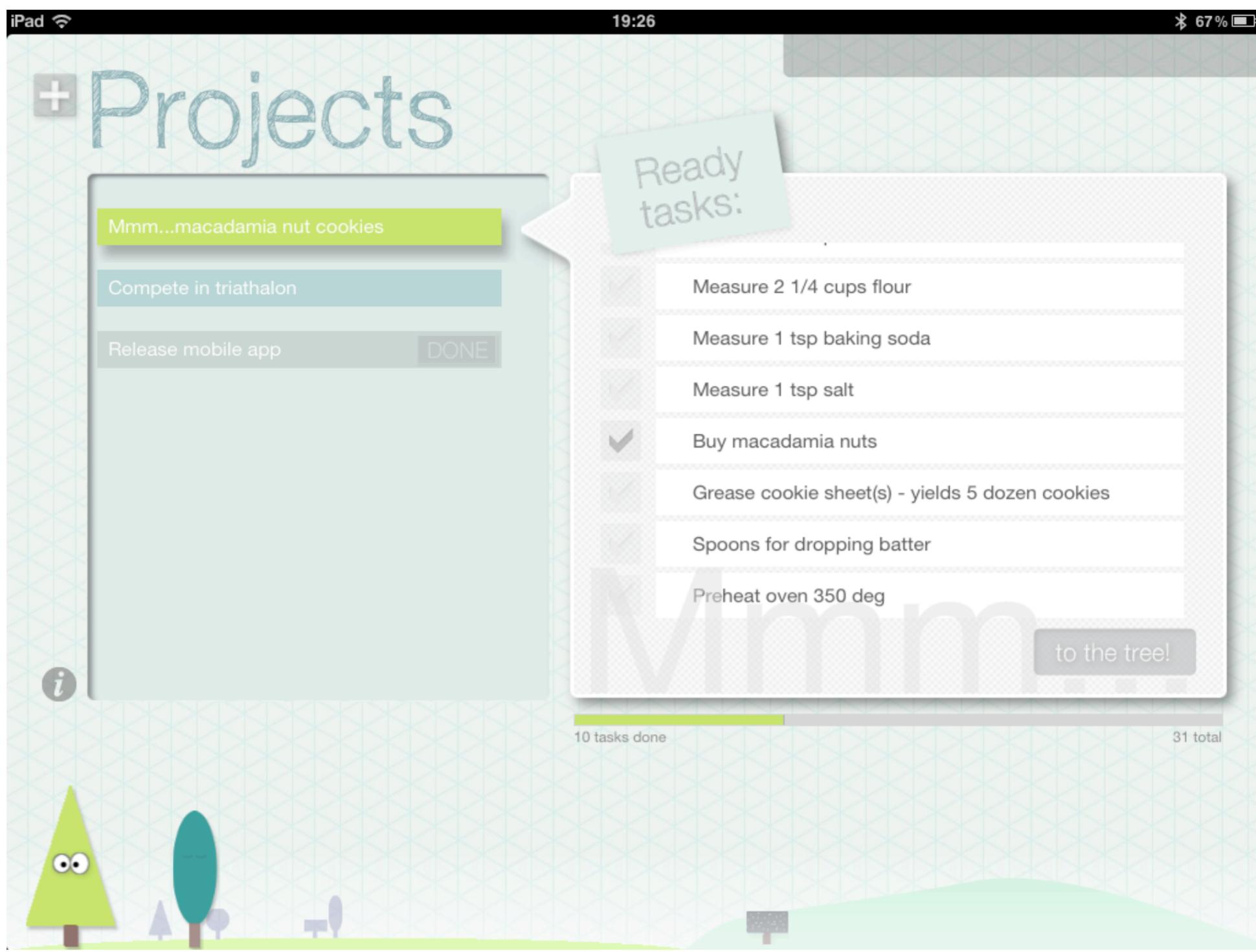


Tuesday, December 11, 12

Communication required here.  
We have to know where the other side is coming from.



# I'm also a customer.



# I also convert (to iOS 6)

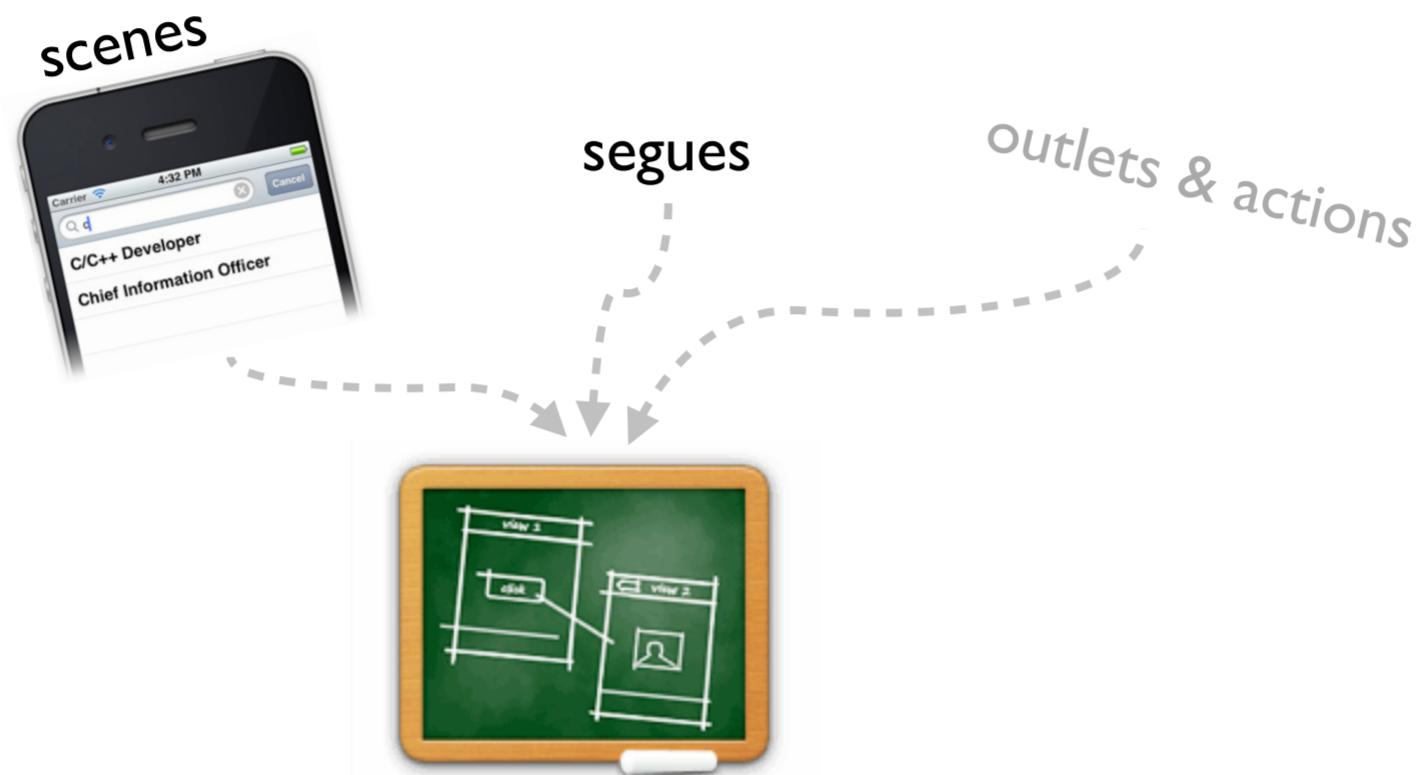
Tuesday, December 11, 12

Treed originally released for iOS 4, upgraded to iOS 5  
Now revising for iOS 6 w/ Storyboards

# XCODE STORYBOARDS

**Exactly what are they?**

# Xcode Storyboards



Tuesday, December 11, 12

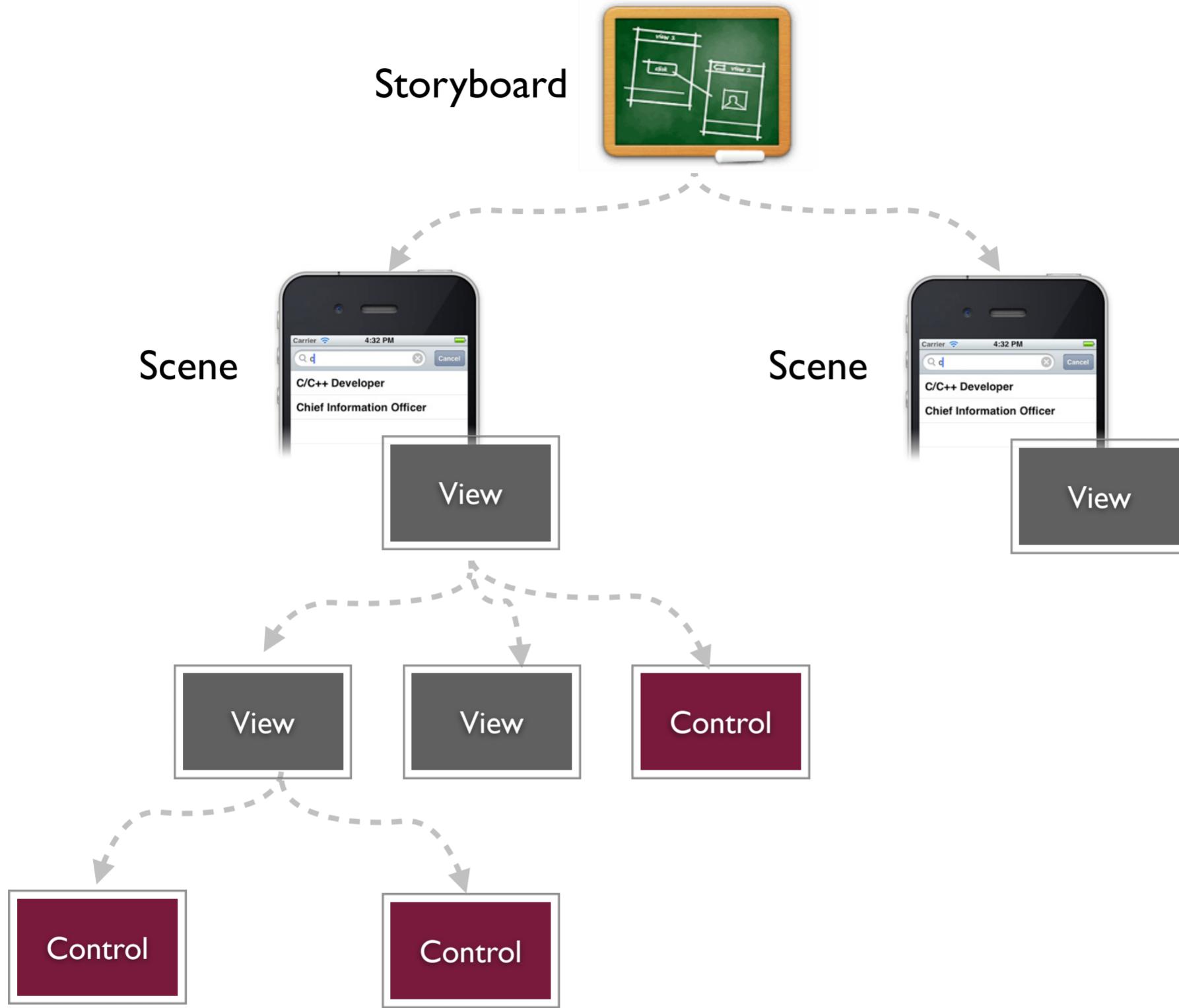
A scene is like a screen (or page)

Scenes are related by segues

Transitions are specific types of segue

Outlets & actions make coding easier (not covered)

# A collection of scenes



Tuesday, December 11, 12

View is a visual element, like a table view or red square.

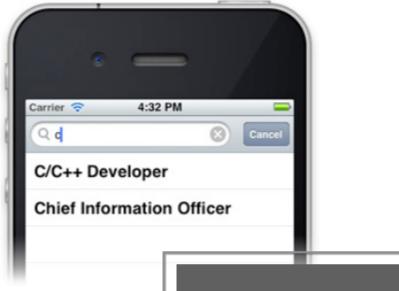
- Contain basics for display and interaction.

Control, like a button (is a specific type of view)

Layers omitted for brevity.

# Views and Controls

## Scene



	<b>Table View</b> – Displays data in a list of plain, sectioned, or grouped rows.
	<b>Table View Cell</b> – Defines the attributes and behavior of cells (rows) in a table view.
	<b>Collection View</b> – Displays data in a collection of cells.
	<b>Collection View Cell</b> – Defines the attributes and behavior of cells in a collection view.
	<b>Collection Reusable View</b> – Defines the attributes and behavior of reusable views in a collection view, such as a...
	<b>Image View</b> – Displays a single image, or an animation described by an array of images.
	<b>Text View</b> – Displays multiple lines of editable text and sends an action message to a target object when...
	<b>Web View</b> – Displays embedded web content and enables content navigation.
	<b>Map View</b> – Displays maps and provides an embeddable interface to navigate map content.
	<b>ScrollView</b> – Provides a mechanism to display content that is larger than the size of the application's window.
	<b>Date Picker</b> – Displays multiple rotating wheels to allow users to select dates and times.
	<b>Picker View</b> – Displays a spinning-wheel or slot-machine motif of values.



	<b>Label</b> – A variably sized amount of static text.
	<b>Round Rect Button</b> – Intercepts touch events and sends an action message to a target object when it's tapped.
	<b>Segmented Control</b> – Displays multiple segments, each of which functions as a discrete button.
	<b>Text Field</b> – Displays editable text and sends an action message to a target object when Return is tapped.
	<b>Slider</b> – Displays a continuous range of values and allows the selection of a single value.
	<b>Switch</b> – Displays an element showing the boolean state of a value. Allows tapping the control to toggle the value.
	<b>Activity Indicator View</b> – Provides feedback on the progress of a task or process of unknown duration.
	<b>Progress View</b> – Depicts the progress of a task over time.
	<b>Page Control</b> – Displays a dot for each open page in an application and supports sequential navigation...
	<b>Stepper</b> – Provides a user interface for incrementing or decrementing a value.

# Segues

Scene:

login screen



button pressed



Scene:

home screen

side menu



swipe



hidden content

custom control



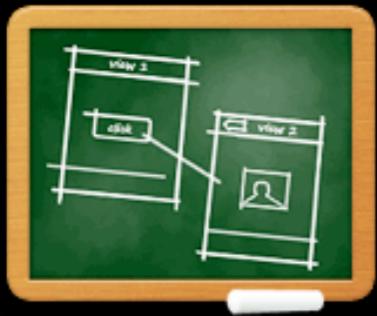
touch



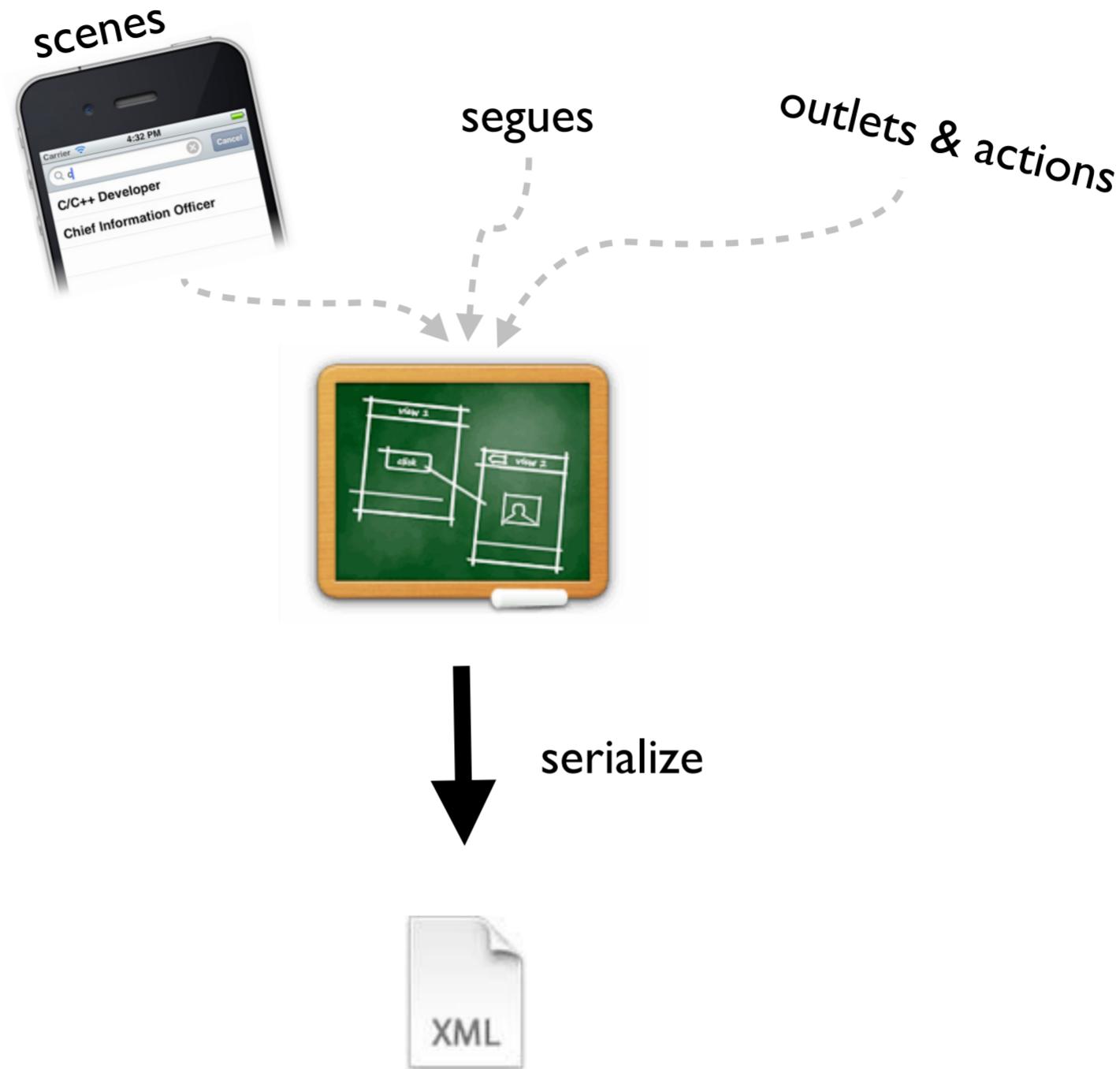
birds in slingshot

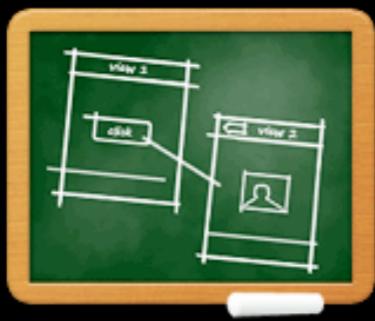
Tuesday, December 11, 12

Other segue triggers: shake, rotation, timing, geo fencing, etc.



# Freeze Dried





# Rehydration



Tuesday, December 11, 12

Can be faster than other editors which emit code to create objects.

# PROJECT TEMPLATES

## iPhone

- Single View
- Master-Detail
- Page Based
- Tabbed

## iPad

- Single View
- Master-Detail

Tuesday, December 11, 12

### iPhone

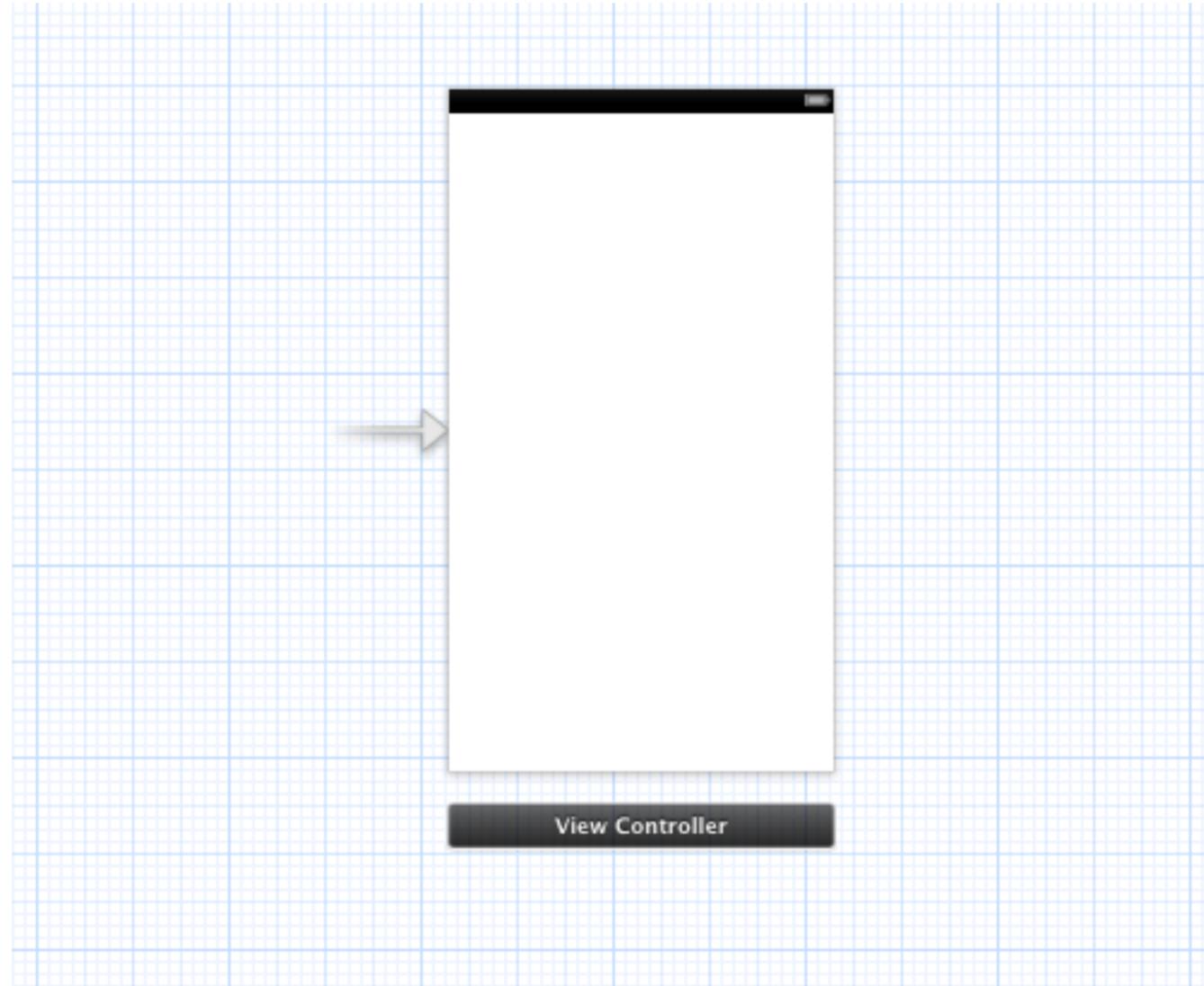
- Single View: This is a blank slate; we'll use this one
- Master-Detail Application (Dynamic UITableView): Just take a look
- Page Based: Just take a look
- Tabbed: Add another tab
  - Add new view controller
  - Ctrl drag from UITabViewController to new view controller

### iPad

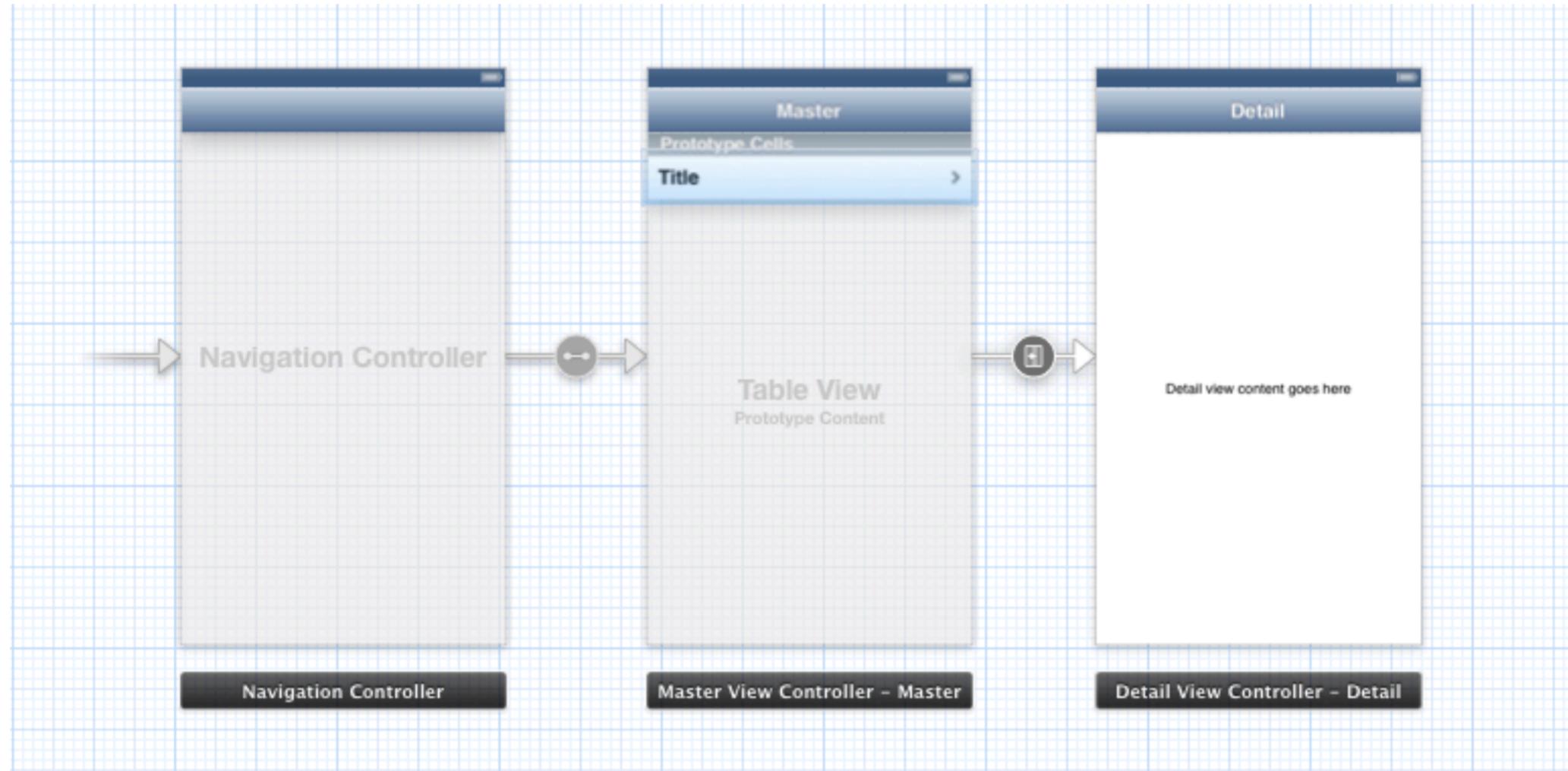
- Master-Detail Application (Split View)
- Single View: Demo's here

# What should we build?

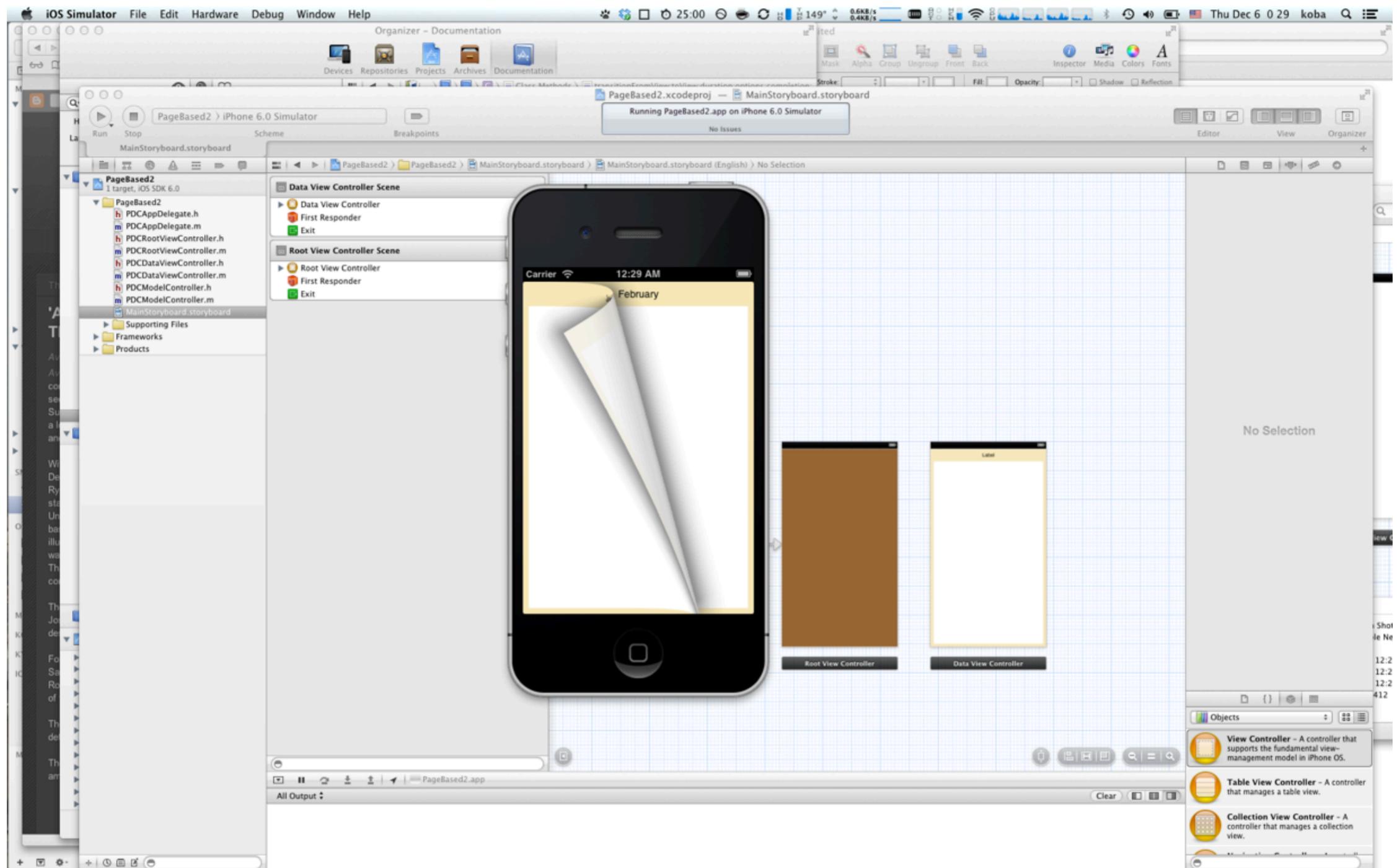
# Single View



# Master-Detail iPhone

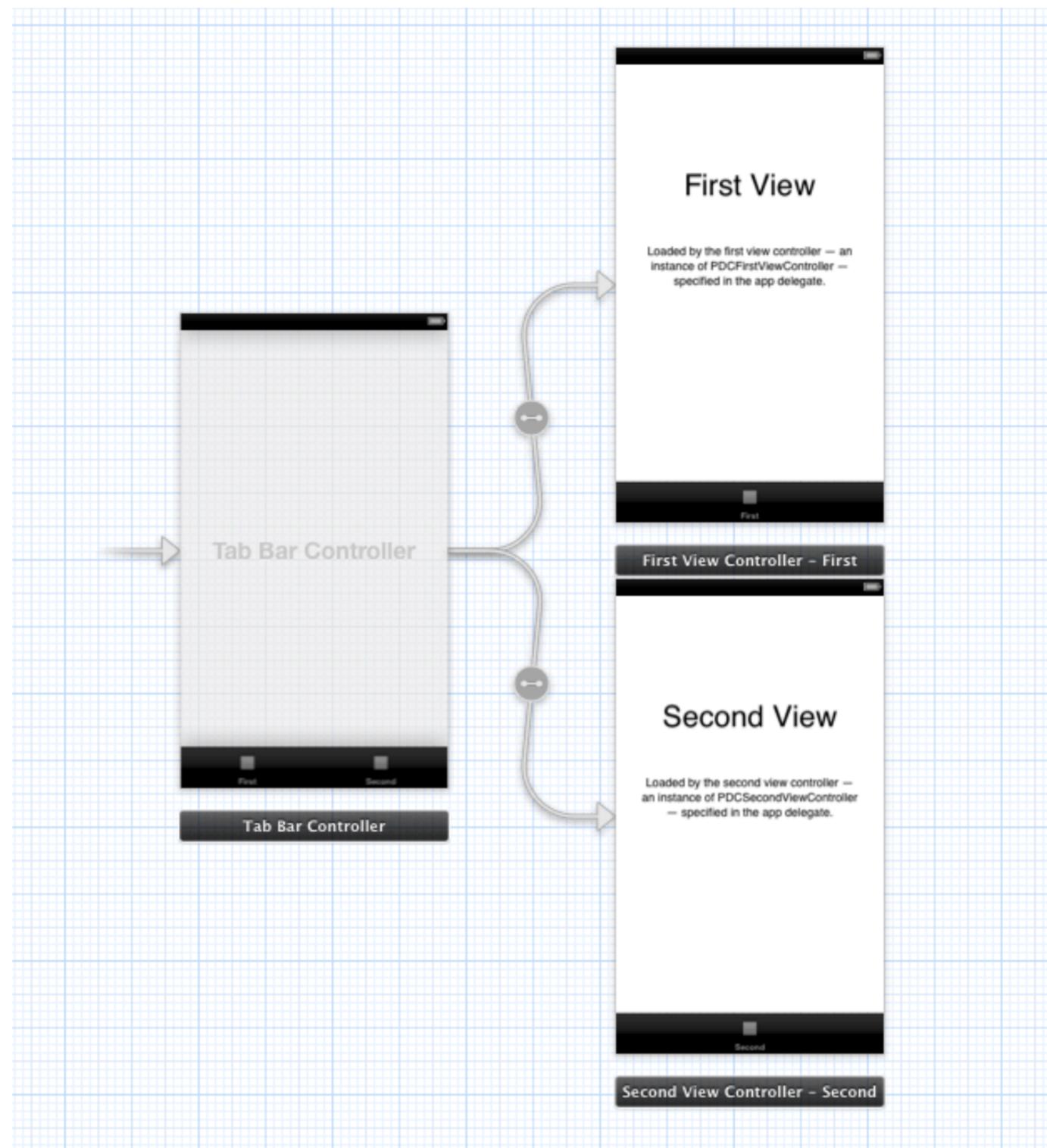


# Page-Based

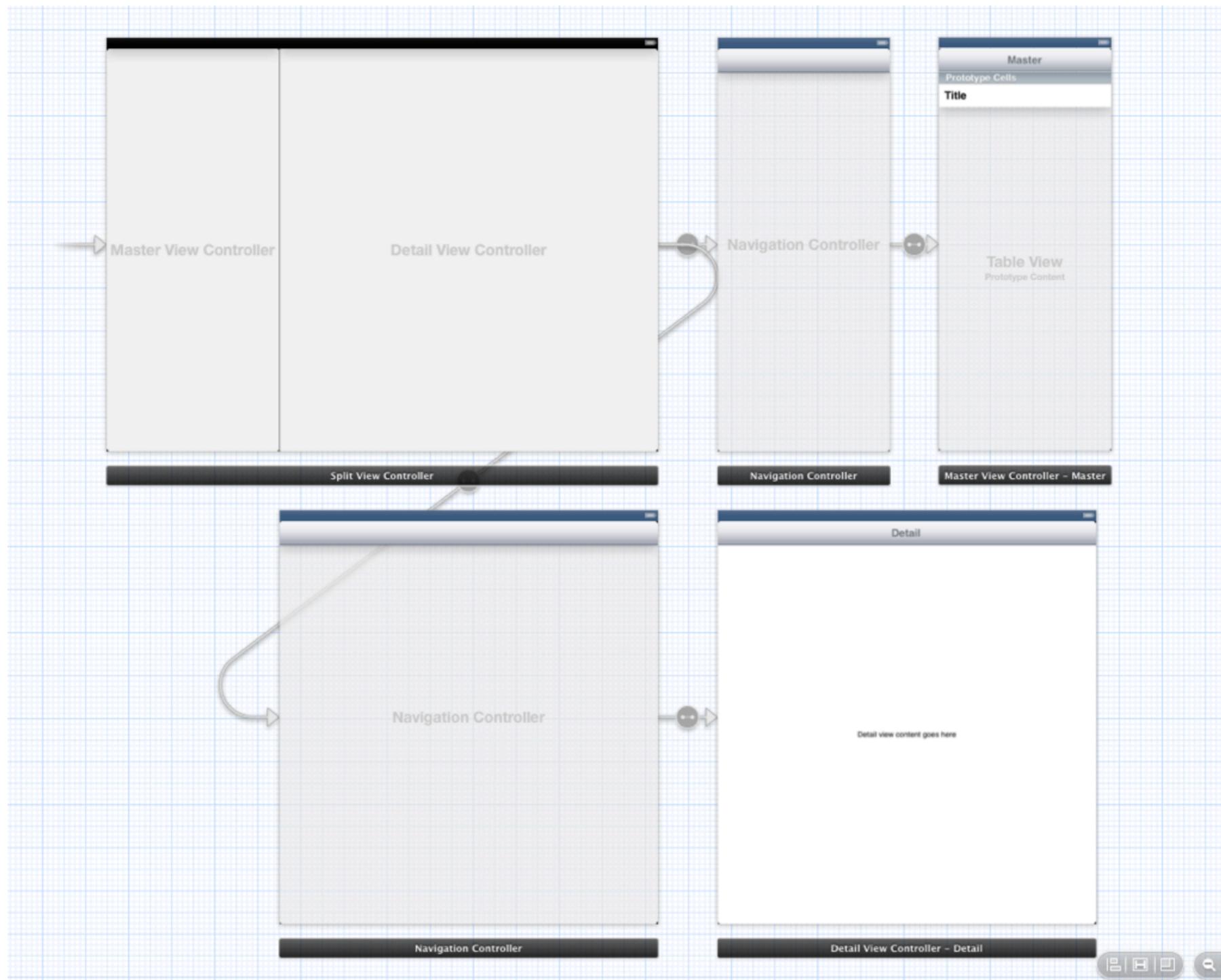


Tuesday, December 11, 12

# Tabbed



# Master-Detail iPad



Tuesday, December 11, 12

# What should we build? (Insta-Kitteh)

# LABS OVERVIEW

- Create projects
- Run app on simulator
- Add views, such as buttons and labels
- Create new scenes
- Move from scene to scene via segues, triggered by buttons or gestures
- Create a configuration screen, using a static table view
- Use container views, such as tab bars and navigations controllers

# LAB I: Single View

1. Create a new folder on your desktop: OUDL
2. Create a new “Single View” Project, saving it in the new OUDL folder
3. Drag a UILabel onto the scene
4. Change the label text
5. Run this project in the simulator

Tuesday, December 11, 12

Show all the steps of creating a new project.

Show each pane of Xcode.

Show all objects that could be dragged in.

Show attributes panel, change color.

Run in simulator.

Show how to change device type.

Show how to re-open a project after it is closed.

# LAB2: Modal Segue

1. Create a new “Single View” Project, or use the old one
2. Drag a button onto the scene
3. Drag another UIViewController onto the scene
4. Change the background color of the new view controller (anything but white)
5. Control-click from the button to the new view controller. When you release, select the “modal” segue option.
6. Run this project in the simulator

Tuesday, December 11, 12

This is a modal segue, so you can't get back (unless you use code).  
Check out the other modal segue types.  
Show what happens when you try a push (boom!).

# LAB3: Nav Controller

1. Create a new “Single View” Project
2. Drag a UINavigationController onto the storyboard
3. Drag “initial scene” arrow to point to “Navigation Controller”
4. Control-click and from the navigation controller to the original scene (not the Table View). When you release, select the “root view controller” segue option.
5. Click on the very top of the Table View scene to select it. Then delete it.
6. Drag a UIButton onto the original scene (not the Navigation Controller)
7. Add another UIViewController onto the storyboard
8. Add a UILabel to the new UIViewController
9. Control-click and from the button to the new UIViewController (the one with the label). When you release, select the “push” segue option.
10. Run simulator, tap button to go to new screen (tap “back” to go back)

Tuesday, December 11, 12

Show “Is Initial View Controller” checkbox in Attributes Inspector

CTRL-OPT-CMD 3: focus on object library’s search field

CMD-R: to start simulator

Now you can get back.

Explain the container view type.

# LAB4: Table View

- I. Create a new “Single View” Project
2. Drag a UITableViewController (not a UITableView) onto the storyboard
3. Drag “initial scene” arrow to point to “Table View Controller”
4. In Attributes Inspector, change Table View Content type from “Dynamic Prototypes” to “Static Cells”
5. Change the Style to “Grouped”
6. Change Sections to 2
7. In the Document Outline, select the first section. Change the number of rows to 5
8. Give the first section a “Header” and “Footer”
9. Run simulator

Tuesday, December 11, 12

Show change to cell selection color  
Show cell accessory types  
Show cell style types

# LAB5: Configuration Screen

- I. Create a new “Single View” Project
2. Drag a UINavigationController onto the storyboard
3. Change the Table View Controller’s table view content type to “Static Cells”
4. Change the Style to “Grouped”
5. Change the first cell’s style to “Basic”
6. Double click the text in the first cell and change it to “My Setting”
7. Change the first cell’s Accessory to “Disclosure Indicator”
8. Drag a UIViewController onto the storyboard
9. Control-click and from the first cell to the new view controller. When you release, select the Selection Segue “push” option.
10. Run simulator (tap on the first cell)

Tuesday, December 11, 12

Change “Root View Controller” title

Add another UIViewController (add a different label to it), use detail disclosure to trigger segue

# LAB6: Tabs

1. Create a new “Tabbed Application” Project
2. Run this project in the simulator

# LAB7: More Tabs

1. Reuse your old “Tabbed Application” Project
2. Drag a new UIViewController to the storyboard
3. Control-click from the Tab Bar Controller to the new view controller. When you release, select the relationship segue “view controllers” option.
4. Change the new view controller’s Bar Item - Title (the default value is “Item”)
5. Add a label to the new view controller
6. Run this project in the simulator

Tuesday, December 11, 12

Explain that the Tab Bar Controller is another container view controller

Extra Credit: Embed the new view controller in a navigation controller, then add a button that pushes another view controller

# LAB8: Gestures

1. Create a new “Single View” Project
2. Drag a UINavigationController onto the storyboard
3. Delete the TableView Controller
4. Drag “initial scene” arrow to point to the “Navigation Controller”
5. Make the original blank view controller the root view controller of the navigation controller (i.e. embed the original blank view controller in the navigation controller)
6. Drag a Swipe Gesture Recognizer onto the original blank view controller. Change the swipe direction to “Left”
7. Drag a new UIViewController onto the storyboard
8. Change to the Top Bar - Navigation Bar attribute of the new view controller (found in the Simulated Metrics section of the Attributes Inspector).
9. Double click the middle of the new navigation bar, and change the title to “Swipe!”
10. Find the Swipe Gesture Recognizer in the Document Outline. Control-click and from the Swipe Gesture Recognizer to the “Swipe!” view controller. When you release, select the Action Segue “push” option.
11. Run simulator (Swipe to the left to trigger the segue)

Tuesday, December 11, 12

Change the swipe direction.

Try this with other gestures.

# Doing it in code.



Display a red square in code.

Display a red square with a Storyboard.

Tuesday, December 11, 12

Code is sometimes required.  
It is certainly required for dynamic interactivity.  
But, do as much as you can in the Storyboard.

# Things not appearing...

- Scroll Views, Web Views, etc.
- Layers, Core Graphics, Core Image, etc.
- Rotation
- Auto Layout, a.k.a. handling rotation & different screen sizes, resolutions
- Dynamic UI elements, a.k.a. interactivity

# Resources

- <http://fortawesome.github.com/Font-Awesome/>
- <http://www.glyphish.com>
- WWDC 2011, 2012 videos having anything to do with storyboarding
- <http://www.raywenderlich.com/tutorials>
- <http://placekitten.com>

# Merci!



**HIC**  **PACITY**

**HTDC** *Cultivating Hawaii's tech sector*  
HIGH TECHNOLOGY DEVELOPMENT CORPORATION

*Aloha*  
**Startups**

# Appendix

# LAB0: Deploying to a device

- I. Register as a developer at:
  - I. <https://developer.apple.com/devcenter/ios/index.action>
2. After registering, you will have access to the documentation. You can then follow these instructions to prepare your development environment for builds on a device:
  - I. [http://developer.apple.com/library/ios/#documentation/Xcode/Conceptual/ios\\_development\\_workflow/10-Configuring\\_Development\\_and\\_Distribution\\_Assets/identities\\_and\\_devices.html#/apple\\_ref/doc/uid/TP400079](http://developer.apple.com/library/ios/#documentation/Xcode/Conceptual/ios_development_workflow/10-Configuring_Development_and_Distribution_Assets/identities_and_devices.html#/apple_ref/doc/uid/TP400079)
3. You'll need to pay \$99 to become a licensed iOS developer. This fee is also required if you would like to publish your app on the iOS App Store. Please note that there are two Apple developer programs, one is for iOS (iPhone, iPad, iPod Touch) apps, the other is for Mac (desktop) apps. Be sure to sign up and pay for the correct one.
4. After registering, you will have access to the documentation. You can then follow these instructions to prepare your development environment for builds on a device:
  - I. [http://developer.apple.com/library/ios/#documentation/Xcode/Conceptual/ios\\_development\\_workflow/10-Configuring\\_Development\\_and\\_Distribution\\_Assets/identities\\_and\\_devices.html#/apple\\_ref/doc/uid/TP400079](http://developer.apple.com/library/ios/#documentation/Xcode/Conceptual/ios_development_workflow/10-Configuring_Development_and_Distribution_Assets/identities_and_devices.html#/apple_ref/doc/uid/TP400079)
5. Read:
  - I. <https://developer.apple.com/ios/manage/overview/index.action> (You can't read this without a developer account.)