iPhone Development

Mark Brindle

6 April 2011

To get started, you need:

- an intel based Macintosh running OS X (10.5.6 or later)
- to sign up to become an iOS Developer
- to download XCode 4 & the iOS SDK
- an iPhone, iPod Touch or an iPad
- get & install a Developer Certificate
- get & install a Provisioning Profile
-

Developing your App

- Decide what kind of app you want
- Create an XCode Project
- Design your App's cool interface
- Write some amazing code (helps to know Objective-C)
- Test your App (naturally)
- Submit to Apple

What kind of App

Utility

- Single screen
- ‘Deck of cards’ analogy
- Easily discernible data
- Small number of ‘cards’ (20 max)

What kind of App

“Immersive”

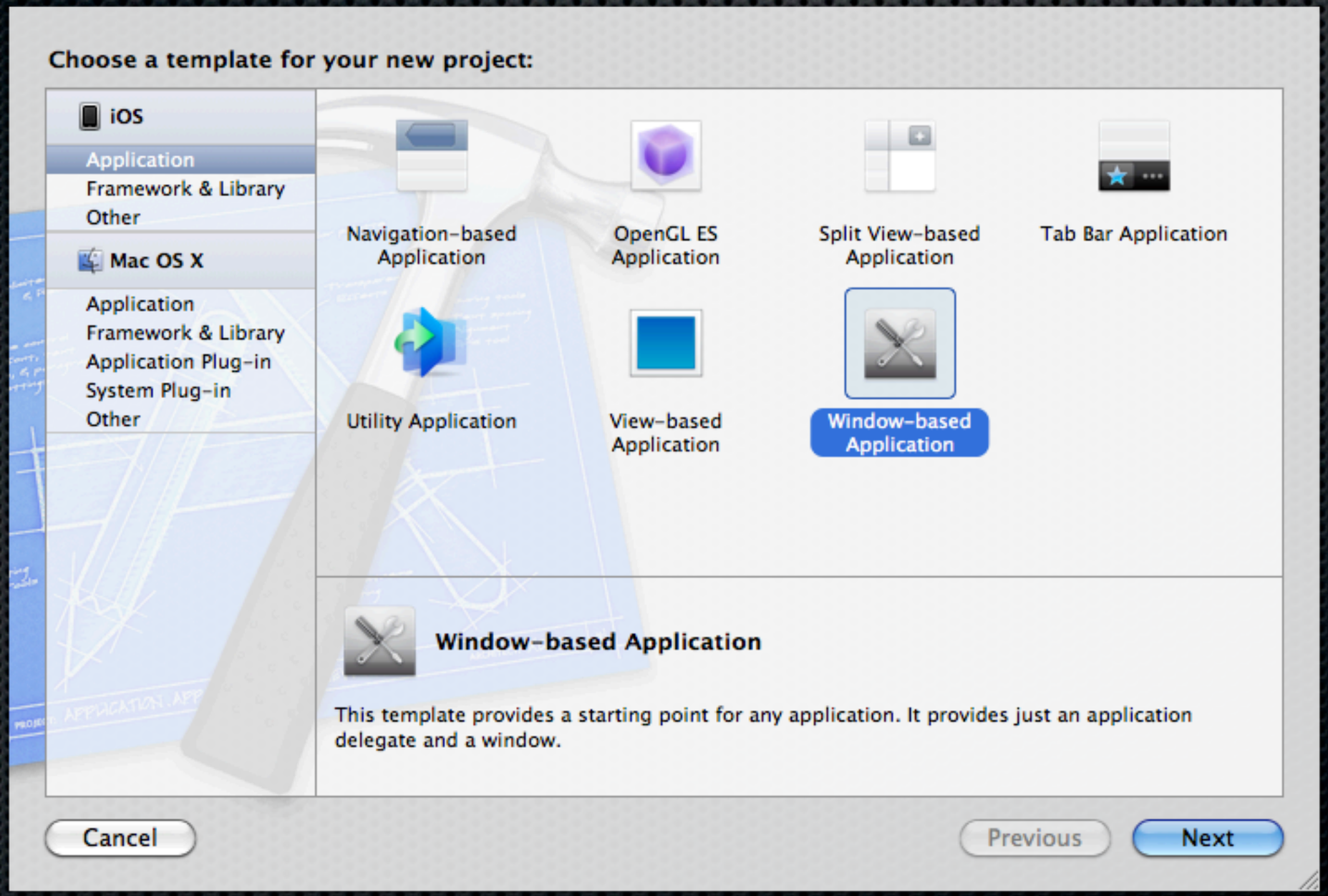
- Often whole screen
- Usually highly graphical
- Primarily games

What kind of App

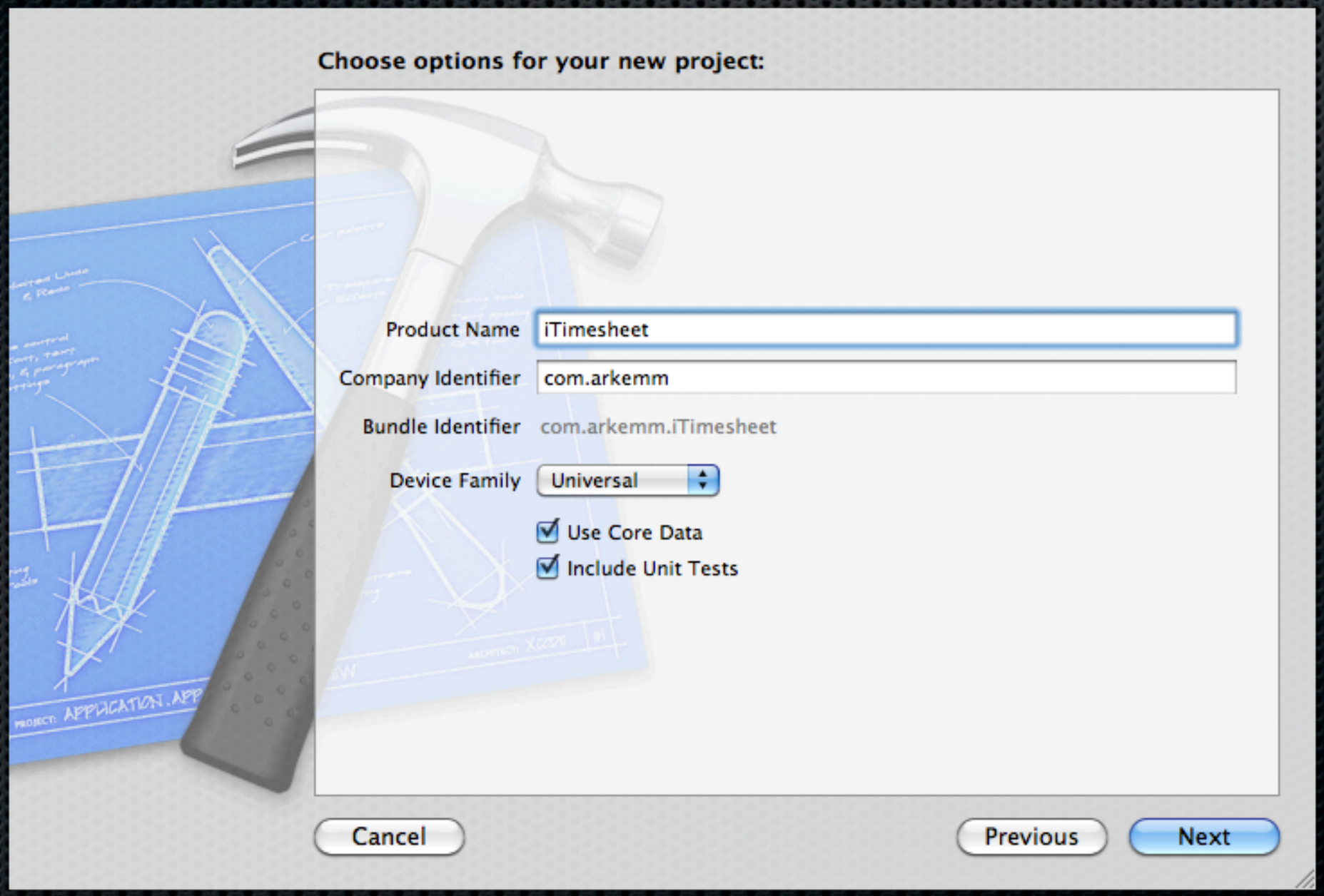
Navigation-based

- Lists
- Tab bar
- Tree structure

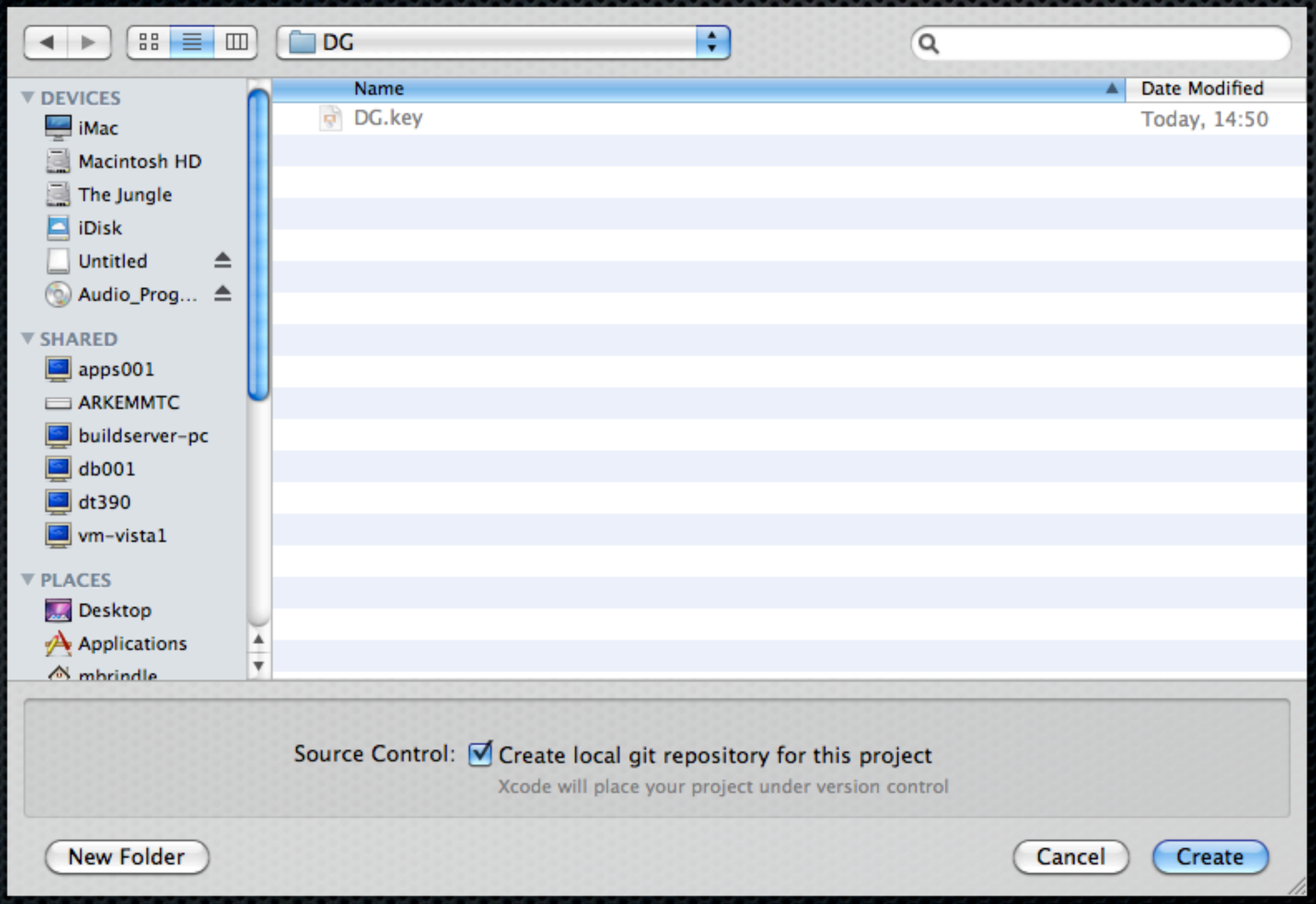
Pick your poison



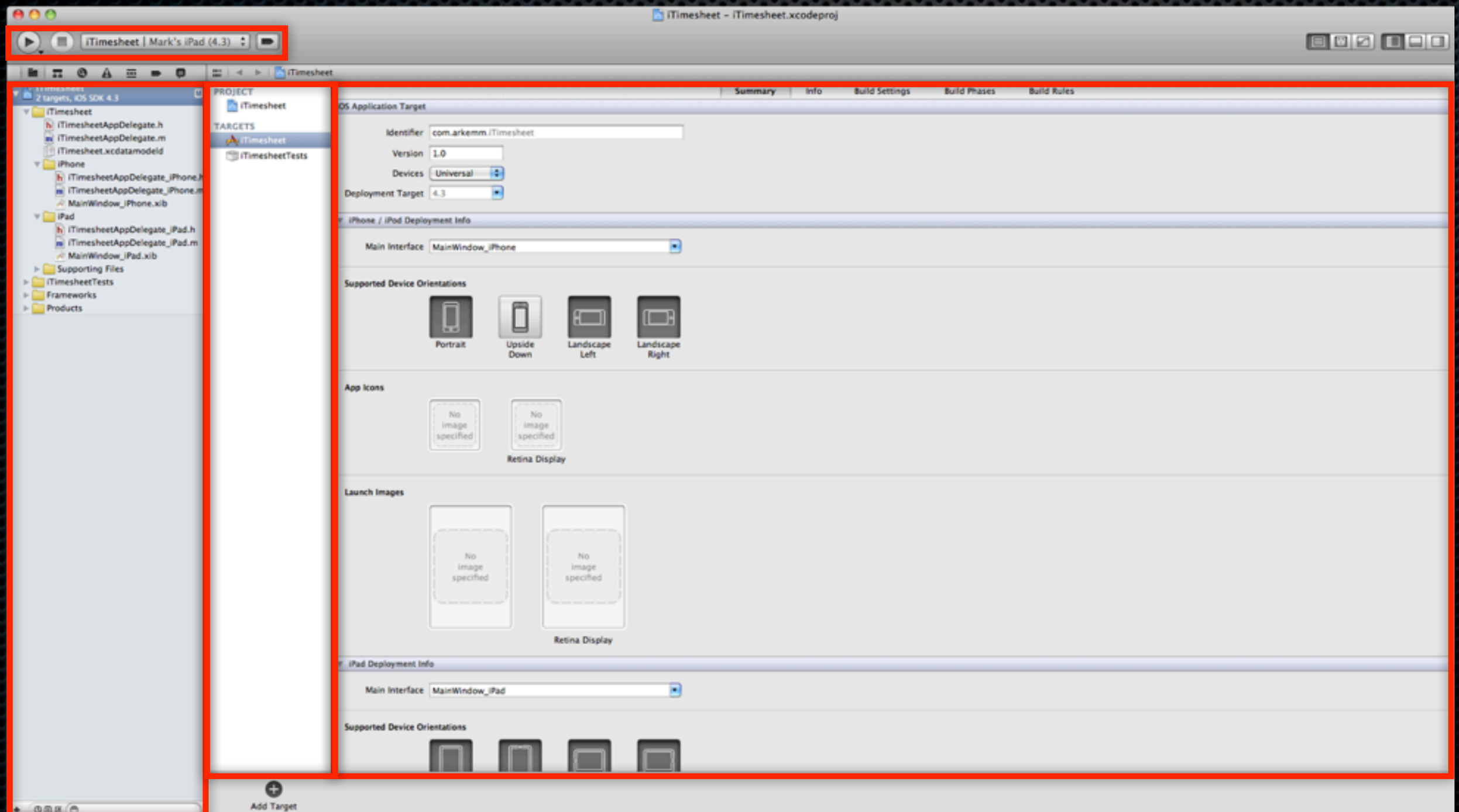
Give your project a name!



Put it somewhere



XCode desktop



Lift-off

- Select a host to run on (depending on what's connected)
- Run - not a lot going on as yet!

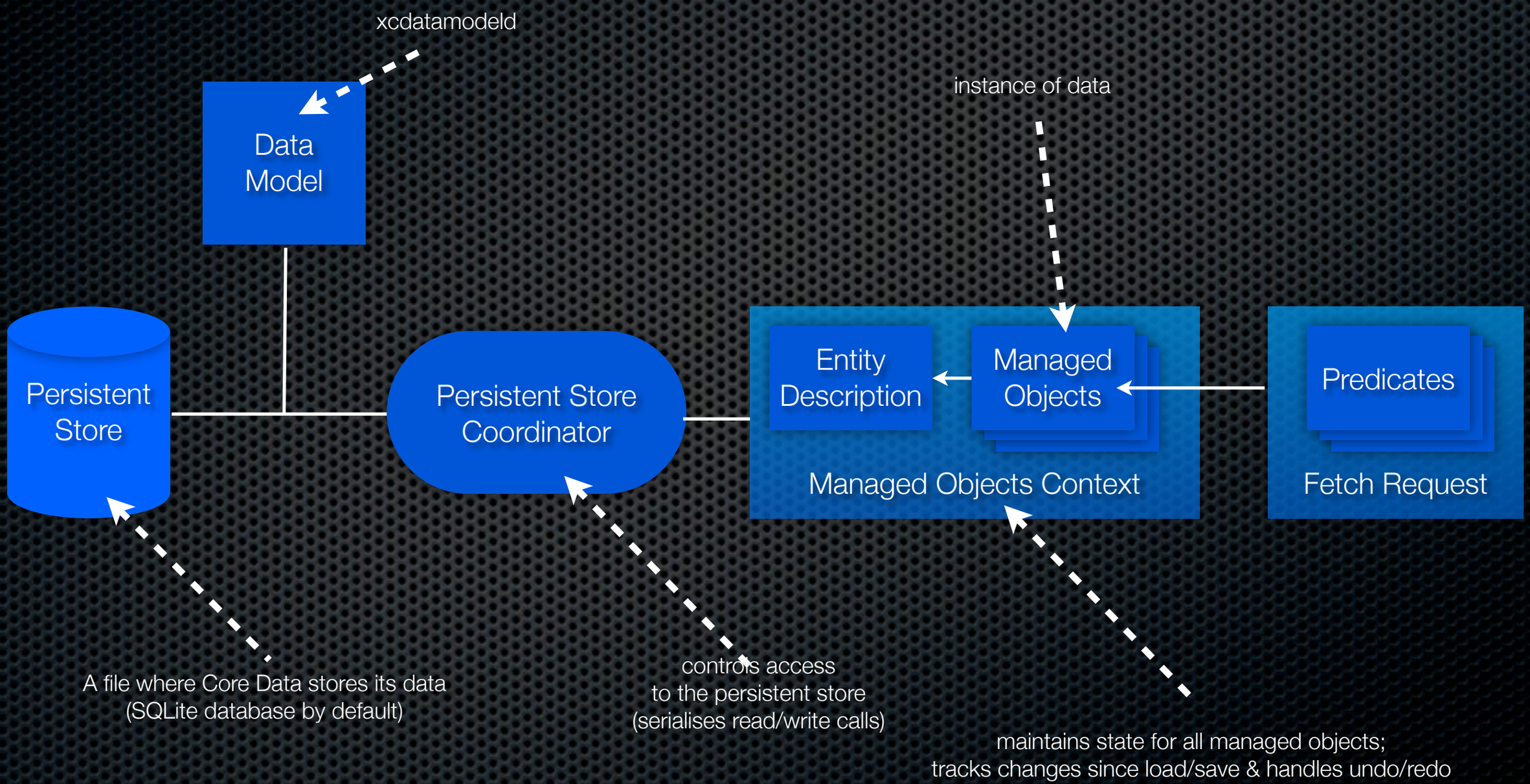
The brief (OK I made this up)

- I want to record time spent working on jobs for different projects.
- I want to quickly start/stop recording time spent on a job.
- I want to easily see how much time I have spent on a job or project.
- It would be nice to assign a cost centre to each project and maybe each job; that way a price can be calculated for time spent.
- I want to see my data on other devices.

Our Design

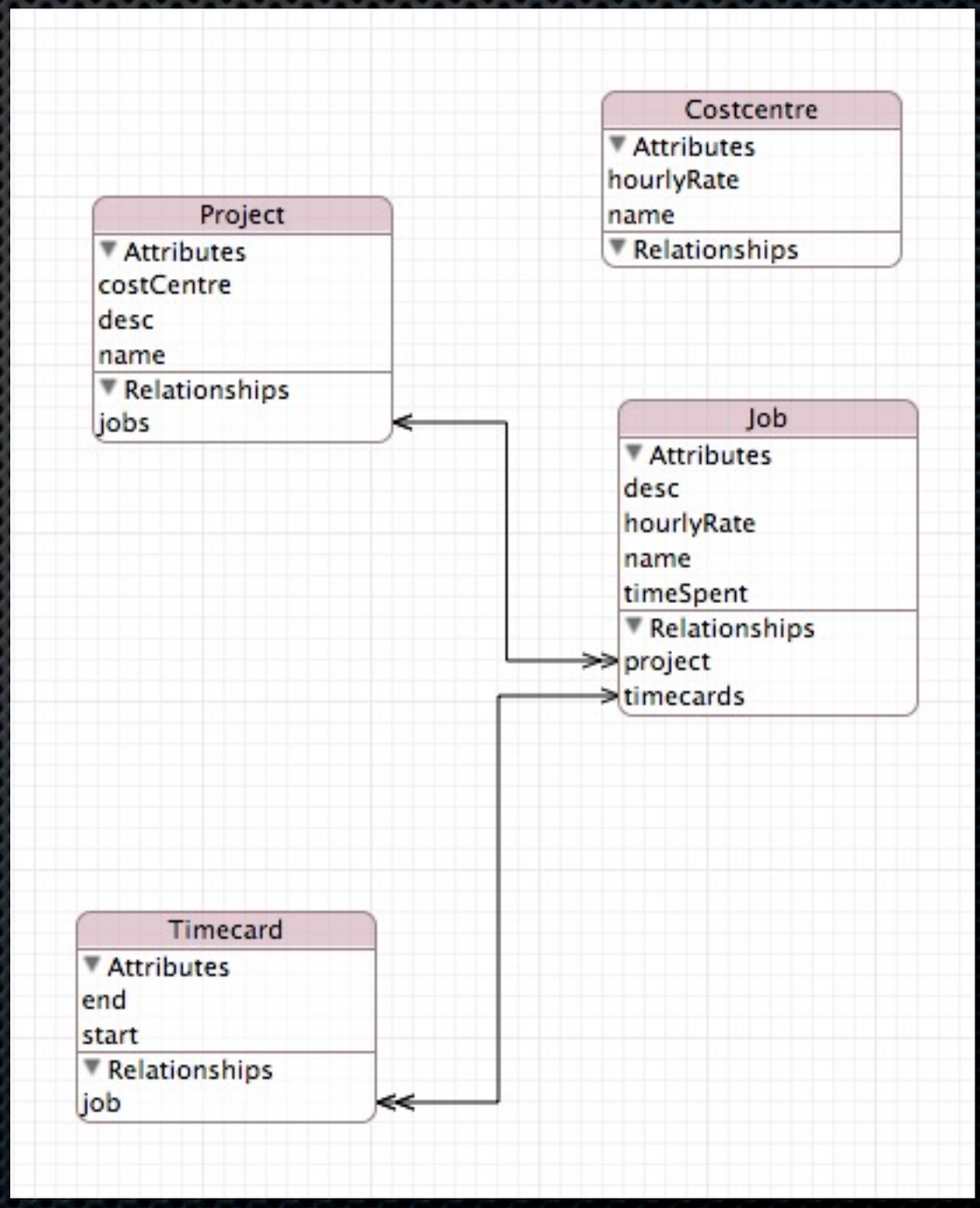
- We need some classes to represent Project, Job, Timecard and Cost Centre.
- We need some way to present this information.
- We need some local storage; we are going to use Core Data, which requires that some entities be defined.

Core Data


xcdatamodeld
Data Model
instance of data
Persistent Store
Persistent Store Coordinator
Entity Description
Managed Objects
Predicates
Managed Objects Context
Fetch Request
A file where Core Data stores its data
(SQLite database by default)
controls access
to the persistent store
(serialises read/write calls)
maintains state for all managed objects;
tracks changes since load/save & handles undo/redo

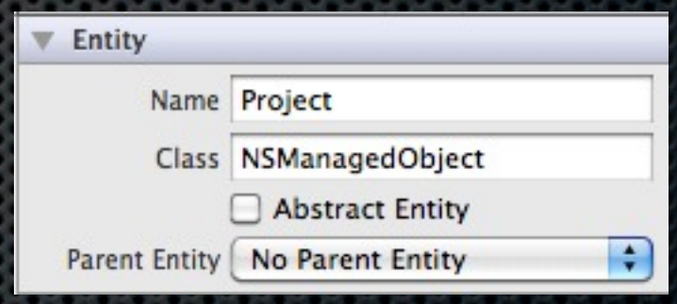
Create a data model

- xcdatamodeld holds our model definition.
- Add entities
- Add attributes to entities
- Add relationships between entities.
- Can auto generate classes to add business rules.



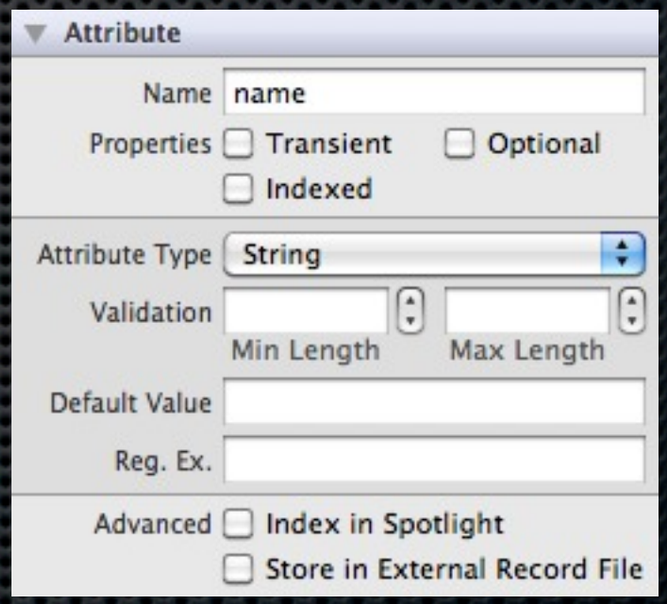
Core Data Entities

- May be abstract; though not in our case.
- Can be derived from another parent entity; again, not for us!
- Derive from NSManagedObject or descendent.



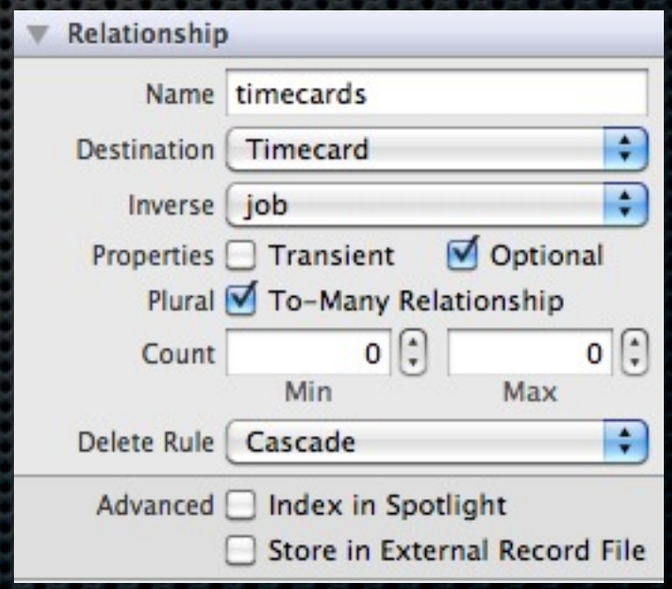
Core Data Attributes

- Different types of data can be represented.
- Attribute details differ depending on type.
- Can set default values as required.



Core Data Relationships

- Destination / Inverse
- To-One / To-Many Relationship
- Delete rule
  - Cascade
  - Nullify
  - Deny
  - No Action



KVC - key-value coding

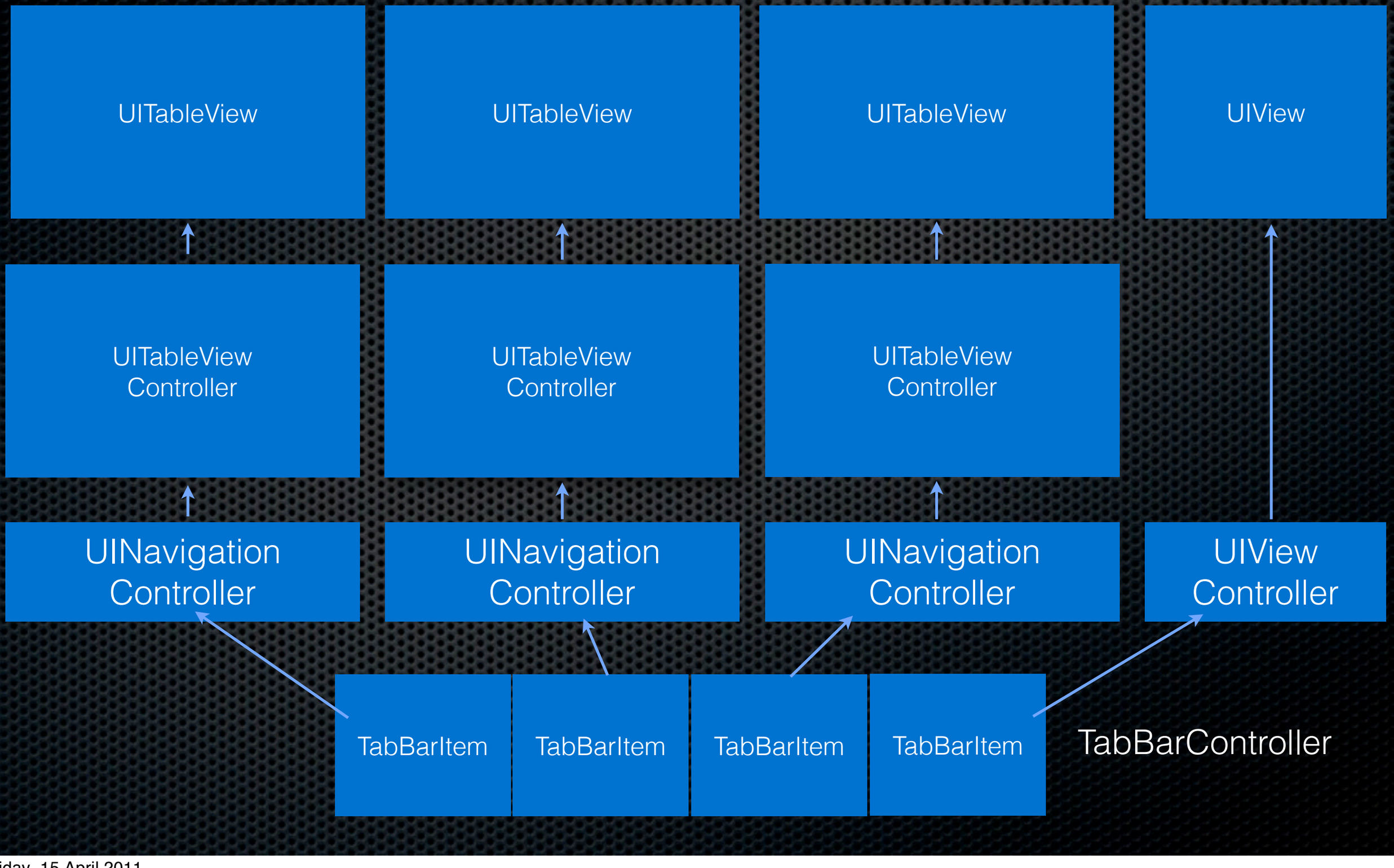
- Access attributes of an object without calling the accessors of that object directly.
- Implemented through an informal protocol on NSObject.
- -valueForKey:
- -setValue:forKey:

KVO - key-value observing

- Sister API to KVC.
- Request notifications when an attribute has changed.
- React when those attributes are changed.

- Implemented via an informal protocol on NSObject.
- -addObserver:forKeyPath:options:context:
- -removeObserver:forKeyPath:
- Although, likeKVC, there are other methods involved in the protocol, these are the primary two used.

UI Elements


UITableView
UITableView
UITableView
UIView
UITableView Controller
UITableView Controller
UITableView Controller
UINavigation Controller
UINavigation Controller
UINavigation Controller
UIView Controller
TabBarItem
TabBarItem
TabBarItem
TabBarItem
TabBarController

Projects Tab

White text shows vanilla objects
Yellow text shows modified objects

Project TableViewCell

UITableView

ProjectTable ViewController (displays Projects)

UIView

InsertProject ViewController (creates a new Project)

ProjectJob TableViewCell

UITableView

JobTable ViewController (displays Jobs; also creates Timecards)

UIView

InsertJob ViewController (creates a new Job)

UITableView

TimecardTable ViewController (displays Timecards)

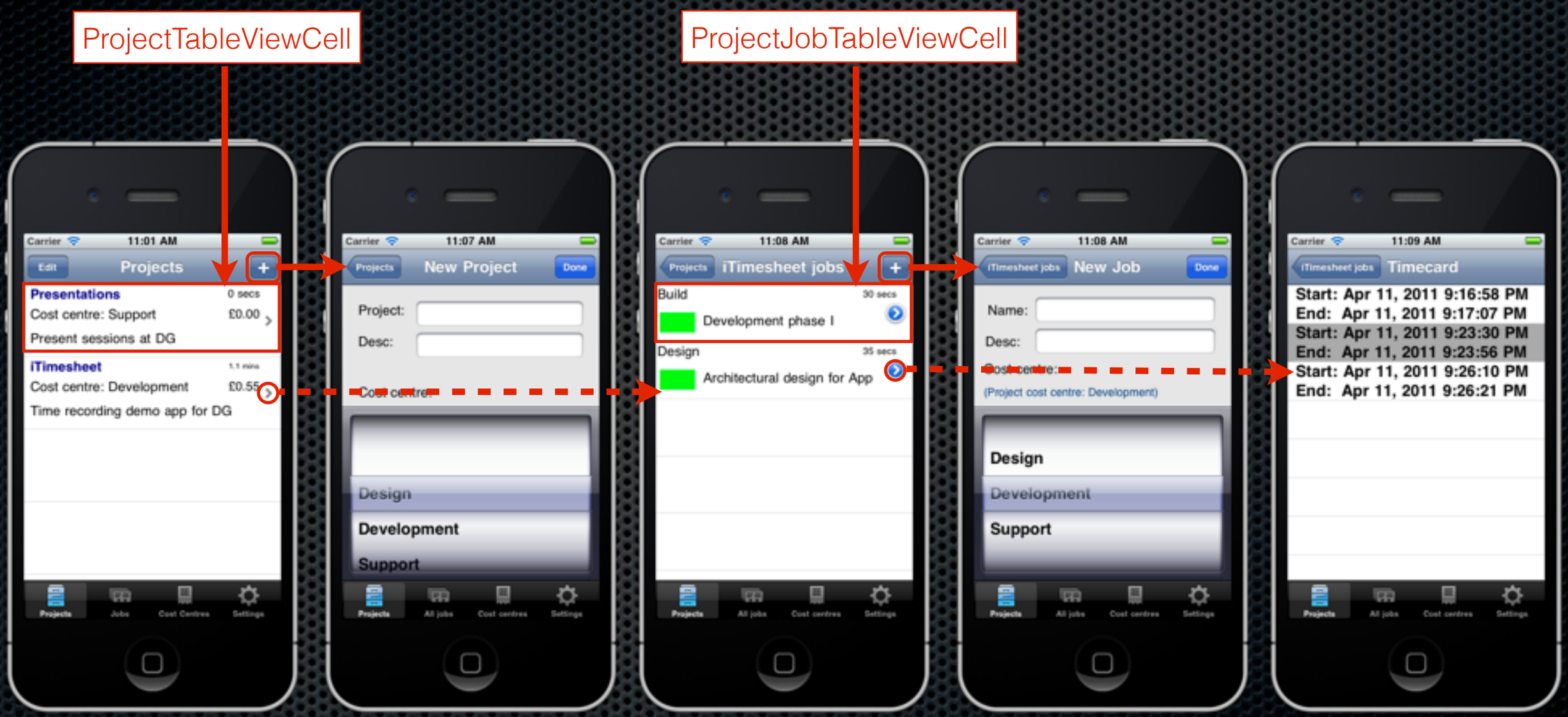
UINavigationController

Projects | Jobs | Cost Centres | Settings

TabBarController

(root Controller)

Projects UI


ProjectTableViewCell
ProjectJobTableViewCell
Carrier 11:01 AM
Edit
Projects
Presentations 0 secs
Cost centre: Support £0.00
Present sessions at DG
iTimesheet 1.1 mins
Cost centre: Development £0.55
Time recording demo app for DG
Projects Jobs Cost Centres Settings
Carrier 11:07 AM
Projects
New Project
Done
Project:
Desc:
Cost centre:
Design
Development
Support
Projects All jobs Cost centres Settings
Carrier 11:08 AM
Projects
iTimesheet jobs
Build 30 secs
Development phase I
Design 35 secs
Architectural design for App
Projects All jobs Cost centres Settings
Carrier 11:08 AM
iTimesheet jobs
New Job
Done
Name:
Desc:
Cost centre:
(Project cost centre: Development)
Design
Development
Support
Projects All jobs Cost centres Settings
Carrier 11:09 AM
iTimesheet jobs
Timecard
Start: Apr 11, 2011 9:16:58 PM
End: Apr 11, 2011 9:17:07 PM
Start: Apr 11, 2011 9:23:30 PM
End: Apr 11, 2011 9:23:56 PM
Start: Apr 11, 2011 9:26:10 PM
End: Apr 11, 2011 9:26:21 PM
Projects All jobs Cost centres Settings
InsertProjectViewController
InsertJobViewController
ProjectTableViewController
JobTableViewController
TimecardTableViewController

Jobs Tab

White text shows vanilla objects
Yellow text shows modified objects

Job TableViewCell

UITableView

AllJobsTable ViewController (displays Jobs)

UITableView

TimecardTable ViewController (displays Timecards)

UINavigationController

Projects | **Jobs** | Cost Centres | Settings

TabBarController

(root Controller)

All Jobs UI

AllJobsTableViewController TimecardTableViewController

Cost Centres Tab

White text shows vanilla objects
Yellow text shows modified objects

UITableView

CostCentreTable ViewController (displays Cost centres)

UIView

InsertCostCentre ViewController (creates a new cost centre)

UINavigationController

Projects | Jobs | Cost Centres | Settings

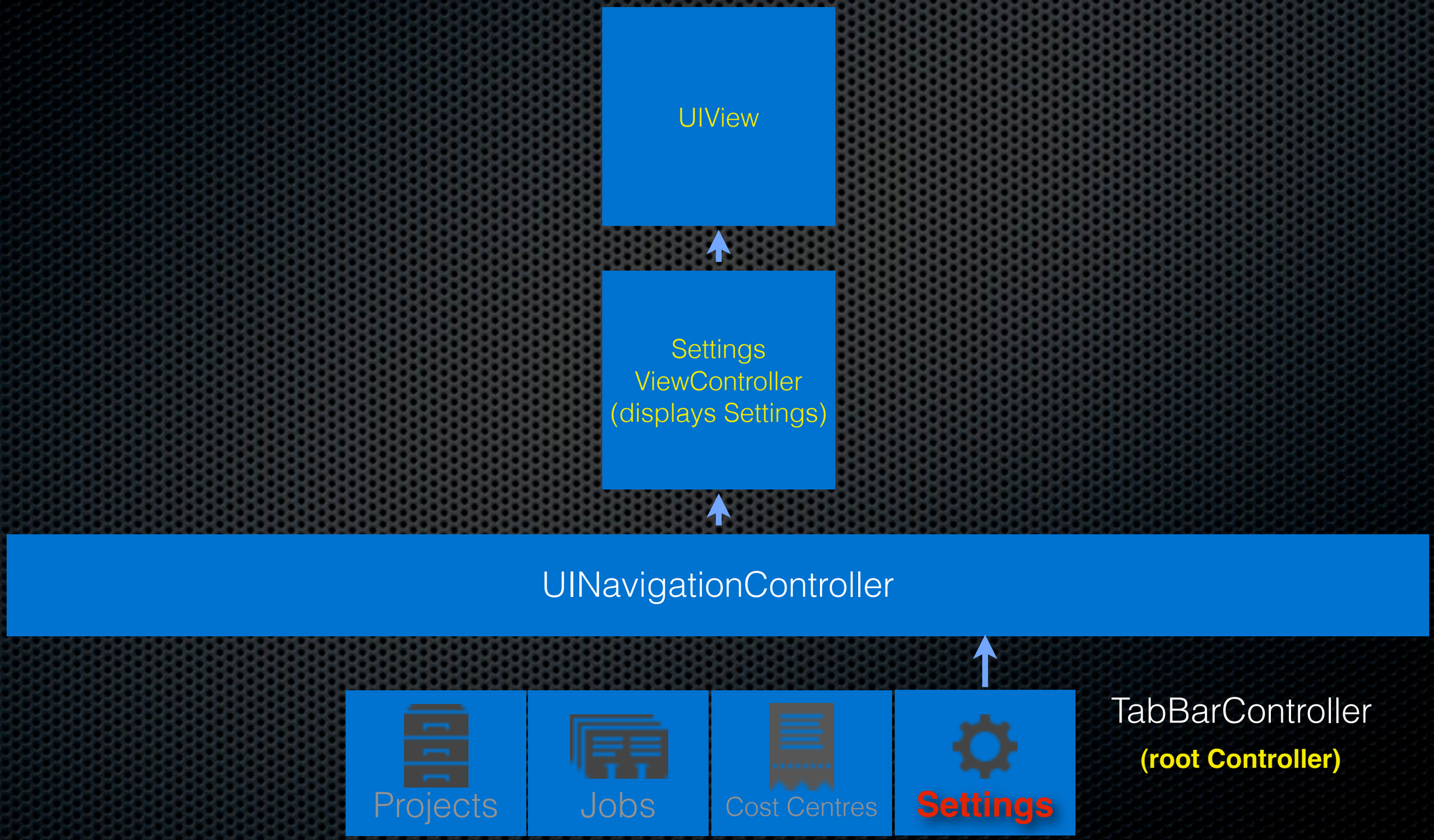
TabBarController

(root Controller)

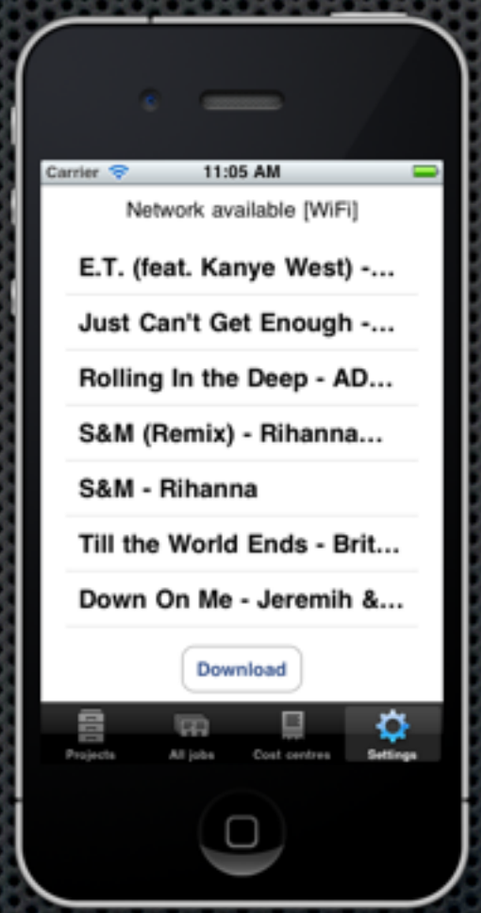
Cost Centres UI

Settings Tab

White text shows vanilla objects
Yellow text shows modified objects



Settings UI



SettingsViewController

Write some code

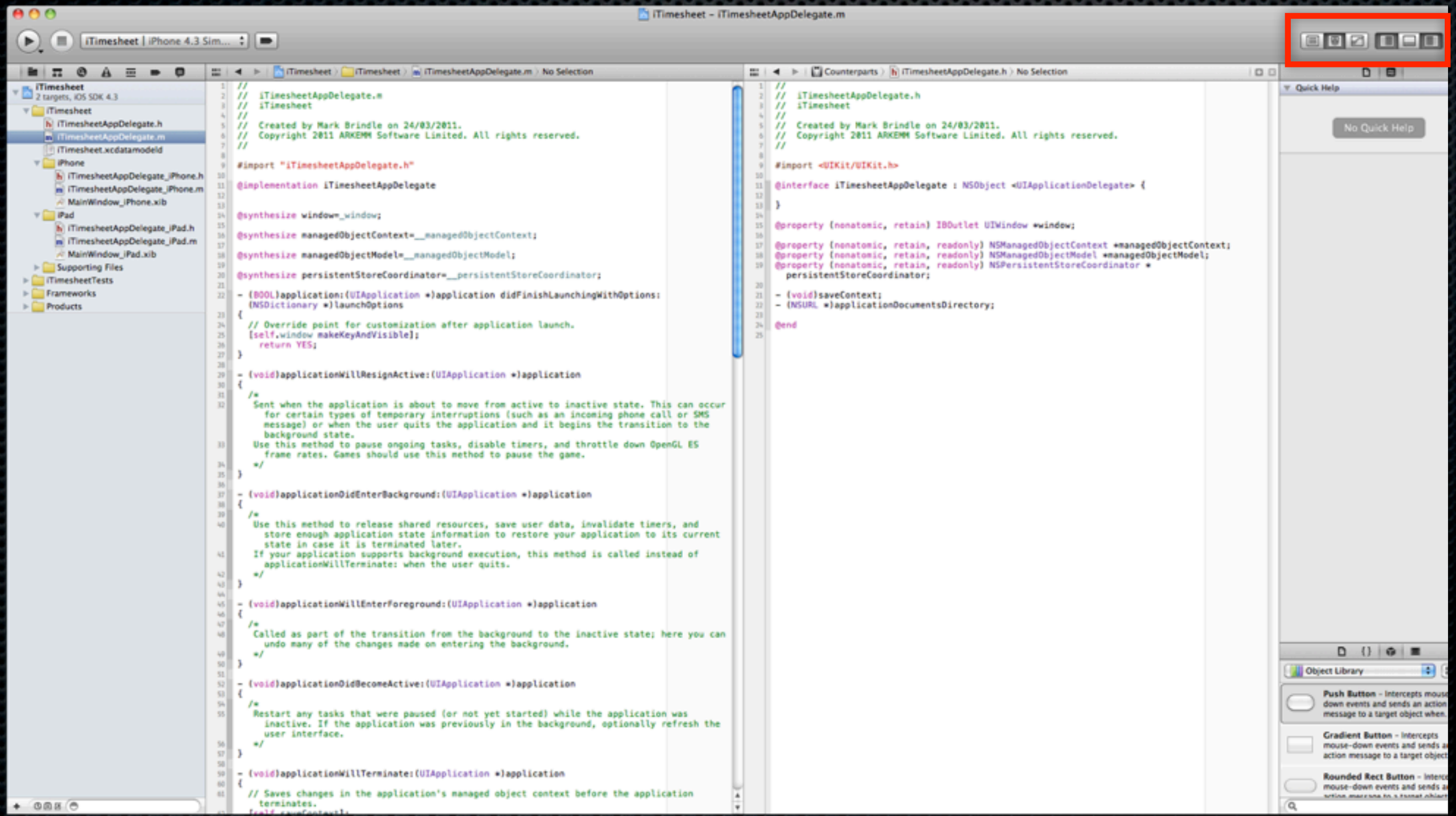
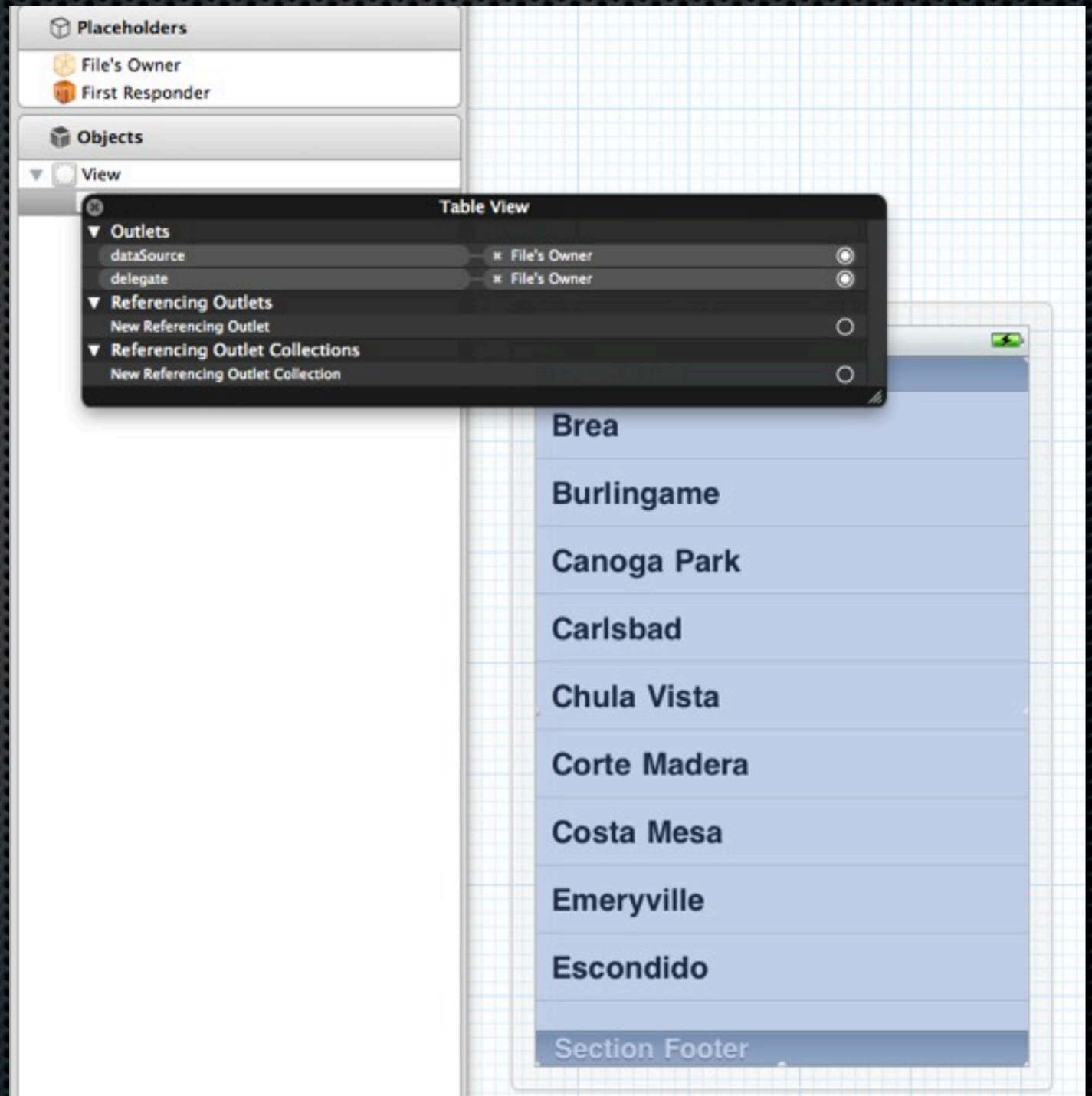
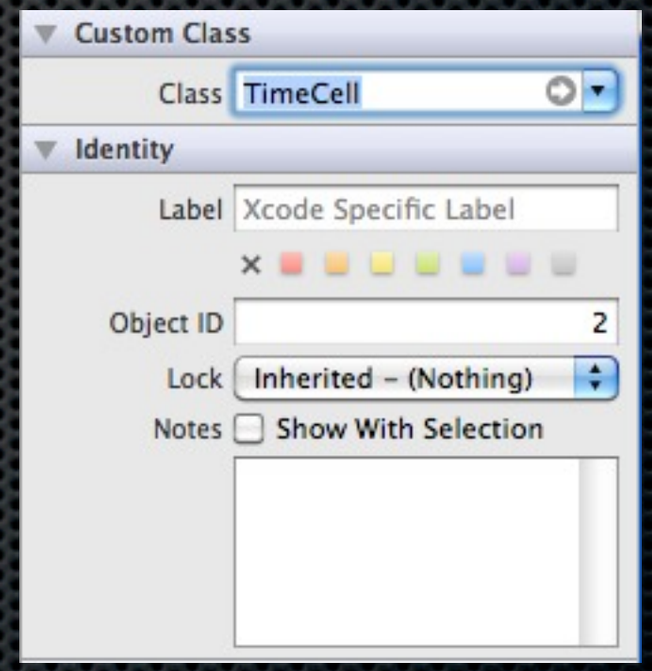


Table view

- Add a table view
- Set delegate to File's owner
- Set datasource to File's owner



TableViewCell

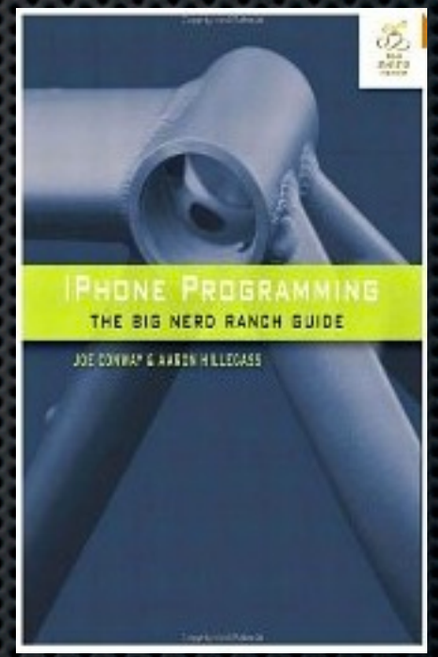


Networking

- 3 different Interfaces
  - WiFi, BlueTooth & Cellular
- Bonjour
- Uses TCP/IP or UDP under the covers
- iPhone SDK uses sockets
- BSD Socket API (Berkeley Software Distribution)
- 3 Frameworks
  - CFNetwork, NSNetServices & GameKit

Networking (continued)

- Checking Network Status
  - Test Network connection type (use lower quality data if on cellular connection; higher on WiFi)
- Networking code examples are adapted from examples in a couple of very useful books



Interrupts

- A number of events cause program interrupts, eg:
  - Phone call or Text message
  - Calendar event or other alerts
  - Switch to another app or press home button
- applicationWillResignActive - pause & save state
- applicationDidBecomeActive - load state & resume

Gotcha's

- "The model used to open the store is incompatible with the one used to create the store"; Fix this by deleting the app & re-running. (but only if you don't mind losing data on the device)