课程介绍

授课：白天
baitian@ustc.edu.cn
sseustc@163.com
中国科技大学软件学院

iOS概述

一、什么是iOS？

iOS是由苹果公司开发的移动操作系统

iPHONE, iPAD, iPOD TOUCH， APPLE TV
iWatch

二、iOS的发展史

1、智能手机的发展史

- 1994年　IBM Simon
- 1996年　Nokia 9000 Communicator
- 1997年　GS88
  首次出现了“智能手机”一词
- 2000-2006年　快速发展期
- 2007年后　普及期

二、智能手机的发展史

- 1994年　IBM Simon

Zaurus OS

- 1996年　Nokia 9000 Communicator

GEOS

那些年我们一起追过的移动OS

Symbian、Meego、 Palm 、 Windows CE、BlackBerry

目前三足鼎立

iOS, android , winphone

二、iOS的发展史

3、iOS发展史

iOS 1：	2007	
iOS 2：	2008	App Store
iOS 3：	2009	
iOS 4：	2010	Retina, 多任务
iOS 5:	2011	Siri
iOS 6:	2012	苹果地图
iOS 7:	2013	扁平化风格 指纹
iOS 8:	2014	

iOS的生态系统

智能手机市场占有率：

Samsung	32.6%	25.2%
Apple	13.4%	11.9%
Huawei	4.8%	6.8%
Lenovo	4.8%	5.4%
Xiaomi	1.8%	5.1%
LG	5.2%	4.9%
Others	37.4%	40.6%

2014 Q3 苹果智能手机运营利润　市场份额为86%

iOS的生态系统

2012年用户在App Store的消费总额45亿美元左右

2013年用户在App Store的消费总额超过100亿美元

2014年Q1国内IOS游戏市场已突破12亿的季度收入

2014年 Q1： 苹果App Store应用下载量比Google Play低45%，然而收入超过后者85%，App Store中国区收入比上季度增长70%。

2014年全年： 150亿美元

iOS的生态系统

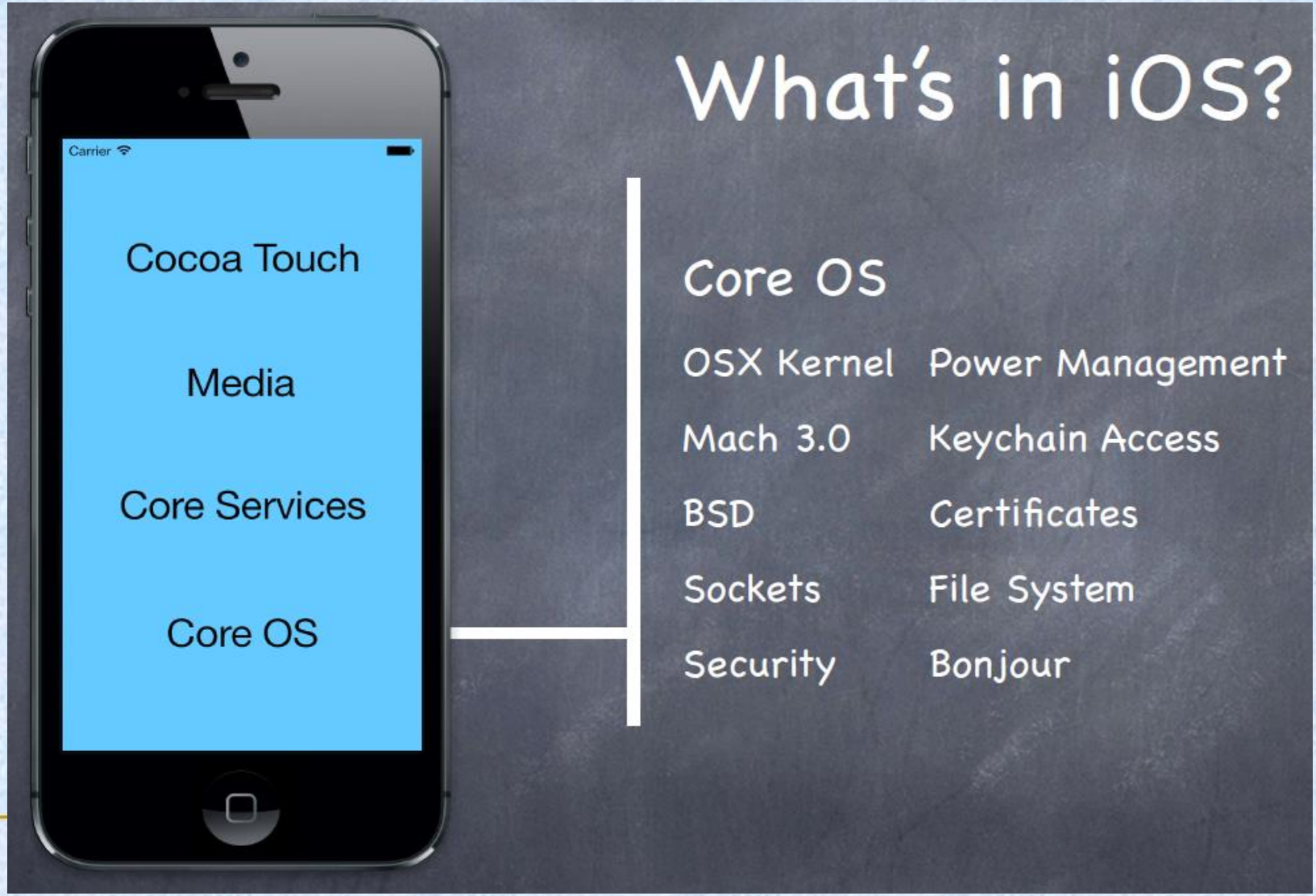
IOS生态系统不仅仅是指产品，更重要的是指iPhone/iPad/iPod/Mac/iWatch/iTV +iCloud+App整个系统，包括Siri（部分设备不支持）、FaceTime、Safari、Game Center、地图、Passbook、电话、邮件。

苹果所有移动设备都使用自己的OS，且不能更换电池，不能插内存卡，数据线连电脑后只能通过自家软件iTunes进行文件的传输，只能使用经过苹果认可的软件，且软件也只能从App Store下载，这就组成了一个封闭的生态圈。

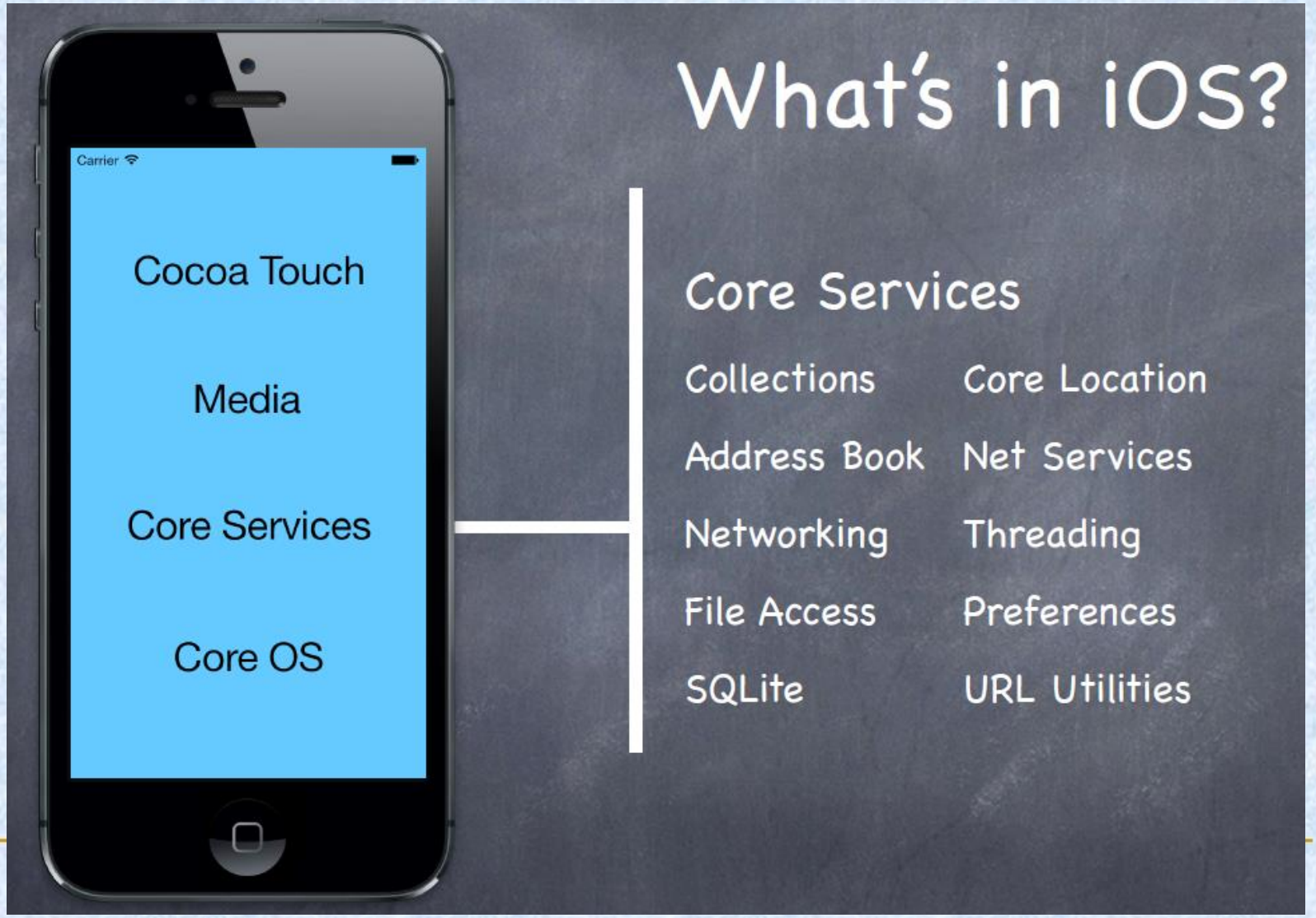
三、iOS架构


Cocoa Touch
UIKit
Media
Core Graphics
Open GL ES
Core Animation
Core Services
Core Data
Foundation
Core OS

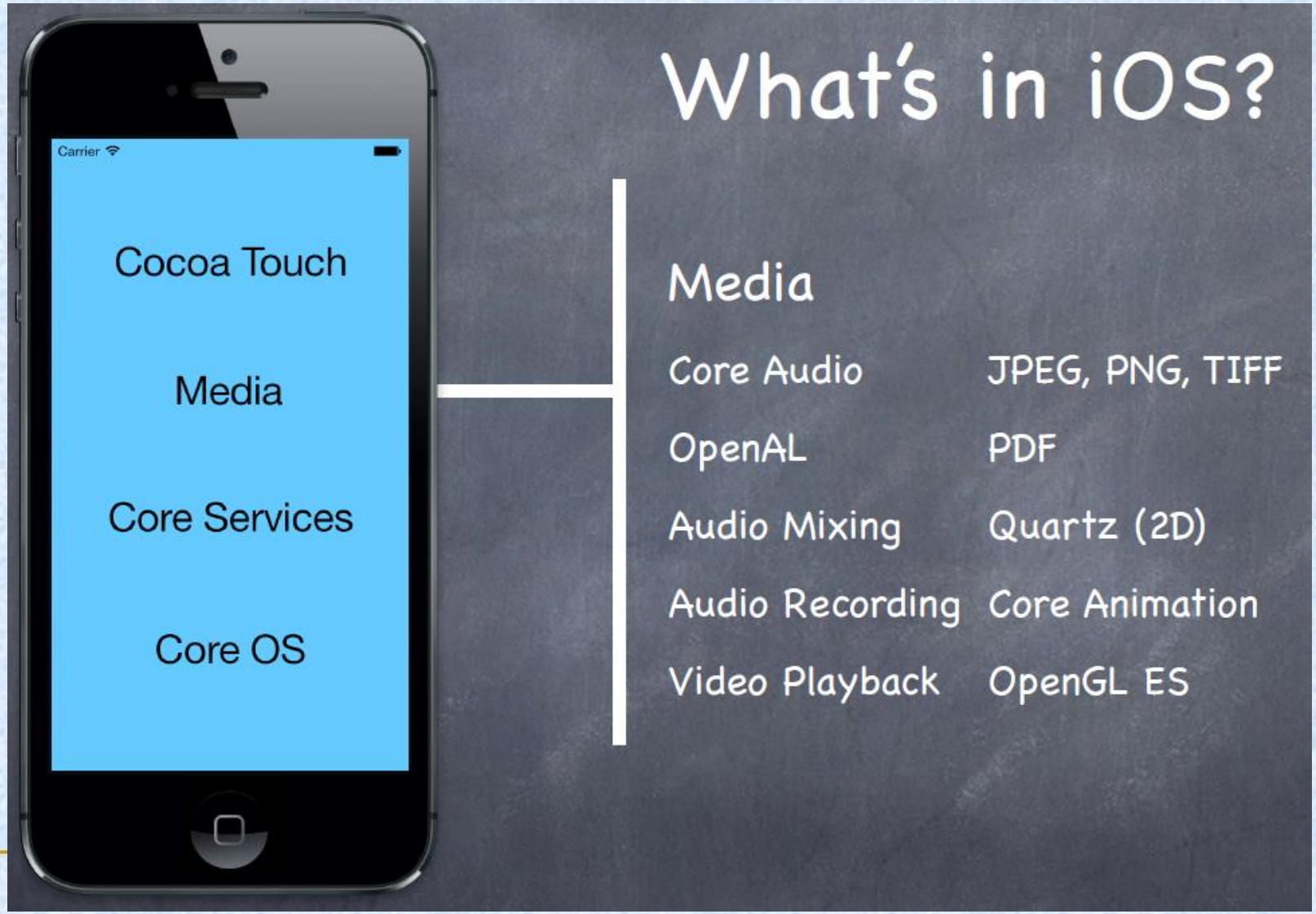
三、iOS架构


Carrier
Cocoa Touch
Media
Core Services
Core OS
What's in iOS?
Core OS
OSX Kernel
Mach 3.0
BSD
Sockets
Security
Power Management
Keychain Access
Certificates
File System
Bonjour

三、iOS架构


Carrier
Cocoa Touch
Media
Core Services
Core OS
What's in iOS?
Core Services
Collections
Core Location
Address Book
Net Services
Networking
Threading
File Access
Preferences
SQLite
URL Utilities

三、iOS架构？


Carrier
Cocoa Touch
Media
Core Services
Core OS
What's in iOS?
Media
Core Audio
OpenAL
Audio Mixing
Audio Recording
Video Playback
JPEG, PNG, TIFF
PDF
Quartz (2D)
Core Animation
OpenGL ES

三、iOS架构


Carrier
Cocoa Touch
Media
Core Services
Core OS
What's in iOS?
Cocoa Touch
Multi-Touch
Core Motion
View Hierarchy
Localization
Controls
Alerts
Web View
Map Kit
Image Picker
Camera

四、MAC OS 与 iOS

MAC OS X 10.10 Yosemite
MAC OS X 10.9 Mavericks

iOS 9
iOS 8

五、 iOS框架

框架包含了框架资源库供应用程序调用

Cocoa

AddressBookUI	使用通讯录的UI框架，以显示联系人的通讯录数据库中的数据。
EventKitUI	创建的用户界面，用于查看和编辑日历数据与事件套件UI框架。
GameKit	在应用程序与游戏Kit框架添加网络功能。
iAd	IAD框架，在您的应用程序放置全屏幕广告或横幅广告。
MapKit	提供应用程序嵌入地图的接口
MessageUI	提供一个用于邮箱发送的ViewController的用户界面接口
Twitter	Twitter API
UIKit	提供应用程序用户界面基础元件的管理功能

五、 IOS框架

Media

AssetsLibrary	获得用户媒体库和数据库的框架
AudioToolbox	录制或播放音频，转换格式，解析音频流，并配置您的音频会议的音频工具箱框架不透明类型
AudioUnit	打开连接，使用音频插件被称为音频单元与音频单元框架，。
AVFoundation	录制，编辑和播放音频和视频，配置您的音频会议，并在设备中的音频环境变化的响应AVFoundation框架。
CoreAudio	表示具有从核心音频框架的基本数据类型的音频流，复杂的缓冲区，和时间值。
CoreGraphics	处理2D渲染的任务，使用核心图形框架。使用这一基于C的API，这是基于Quartz的绘图引擎，路径为基础的绘图，抗锯齿渲染，渐变，图像，色彩管理，和PDF文件处理。
CoreImage	执行图像处理和视频图像处理的核心映像框架。
CoreMIDI	MIDI设备，包括硬件键盘和频率合成器器，使用核心MIDI框架，进行交流。使用Dock连接器或网络连接。
CoreText	布局文本和执行的核心文本框架的字体处理。文本布局API提供高品质的排版，包括字符，字形的线条和段落字形的转换和定位。互补的字体技术提供功能，如自动字体替换（级联），字体描述符和集合，并容易获得字体度里和字形数据。
CoreVideo	电影播放和过程，访问单个帧，与核心的视频框架。这种基于C的框架提供了一个低级别的电影工作，管道的API。您可以使用它的工作与像素的缓冲区，OpenGL的缓冲区，和OpenGL纹理。
GLKit	创建使用GLKit框架的OpenGL ES应用程序所需的时间缩短。 GLKit包括数学库，一个标准视图和视图控制器来实现你的渲染循环，背景纹理加载和预先创建的着色效果。
ImageIO	大多数图像文件格式的读取和写入图像I / O框架。这种基于C的框架还支持色彩管理和图像元数据的访问。
MediaPlayer	查找和播放用户安装媒体项目，包括歌曲，音频播客，有声读物，并与媒体播放器框架。您还可以用它来播放定制的电影文件，如那些用于削减在游戏场景，。
OpenAL	使用低延迟，音频播放位置，建立为iOS引人注目的游戏。 OpenAL的使您沉浸在定向声音跟踪屏幕上的动画的用户。内置的Core Audio后，在iOS的OpenAL提供高性能和出色的音频质量。
OpenGLES	使用一个紧凑，高效的移动设备上的二维和三维绘图的OpenGL API的子集。 OpenGL ES的框架包括EAGL，基于C的API，支持OpenGL ES的渲染核心动画层和UIKit意见的整合。您还可以使用EAGL渲染像素缓冲区
QuartzCore	使用Quartz的核心框架，呈现最佳性能，然后在硬件配置的动画和效果。该框架包含了先进的的动画和合成技术为核心动画。

五、 IOS框架

Core Services

Accounts	管理用户帐户的外部账户使用框架。
AddressBook	使用通讯簿框架，以获得访问中央数据库，用于存储用户的联系人。该数据库，被称为"地址簿"，是使用的应用程序，如邮件和信息，目前已知和未知的人士的信息。
CFNetwork	提供对系统网络服务和配置的访问接口。
CoreData	使用一般化和自动化解决方案与对象生命周期和对象图的管理，包括持久相关的共同任务中的核心数据框架。
CoreFoundation	使用所有iOS应用程序的基本系统服务的核心基础框架。核心基金提供了常见的数据类型的抽象，它有利于国际化与Unicode字符串存储，它提供了一个套件，例如公用事业插件支持，XML属性列表，URL资源的访问，和喜好。
CoreLocation	使用的核心位置框架，以确定当前的纬度和经度和设备配置和调度提供位置相关的事件。该框架使用可用的硬件三角用户的位置，附近的信号信息的基础上。
CoreMedia	核心媒体框架的基本数据类型的基于时间的音像数据。
CoreMotion	接收和处理的核心运动框架的加速度计和其他运动事件。
CoreTelephony	访问蜂窝电话的地位和移动电话服务提供商的信息与核心电话框架。
EventKit	读，写的日历数据的Event Kit框架。
Foundation	该基金会框架提供您需要实现图形，事件驱动的iOS应用程序的基本工具和基础设施。
MobileCoreServices	访问标准类型和常数与移动核心服务框架。依赖上UTI的类型信息的其他框架一起使用。
NewsstandKit	下载和处理杂志的问题或其他书报亭的内容，使用的报刊亭框架。您还可以使用此框架来管理下载的问题。
QuickLook	显示基于视图的Quick Look框架的项目预览。
StoreKit	嵌入在您的应用程序的存储，使用Store Kit框架。用它来处理与购买内容和服务，从您的应用程序相关的金融交易。
SystemConfiguration	使用系统配置框架，以确定网络的可用性和设备状态。系统配置框架声明的功能，类型，以及网络可达性有关的常数。
UIAutomation	这个文件集为UI自动化功能，它允许你编写的测试脚本，行使您的应用程序的用户界面元素，作为连接的设备上运行的应用程序的API参考。你写的UI自动化API，模拟与应用程序的用户交互，运行日志信息，并返回到主机，因为它在JavaScript测试。

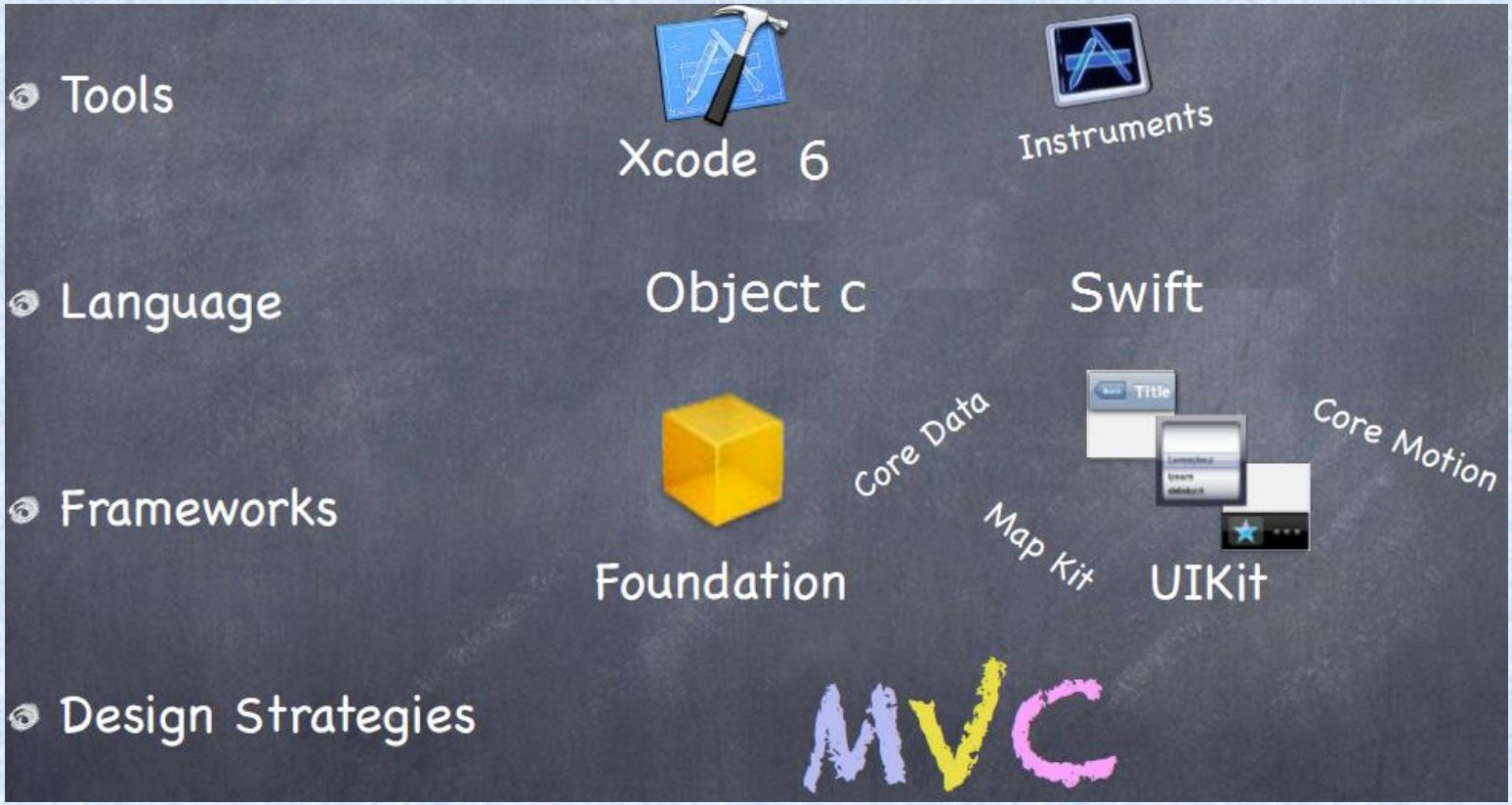
五、 IOS框架

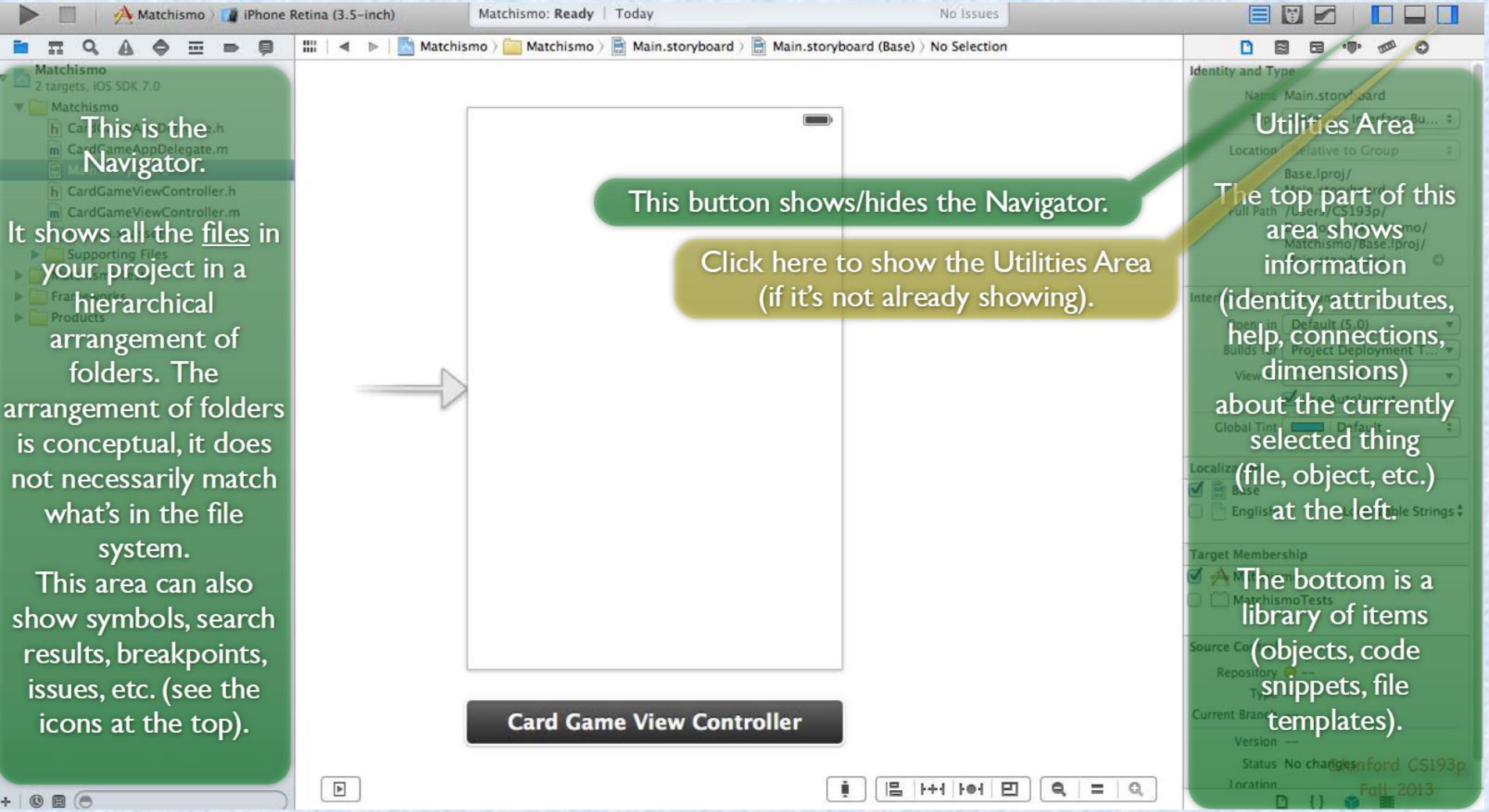
Core OS

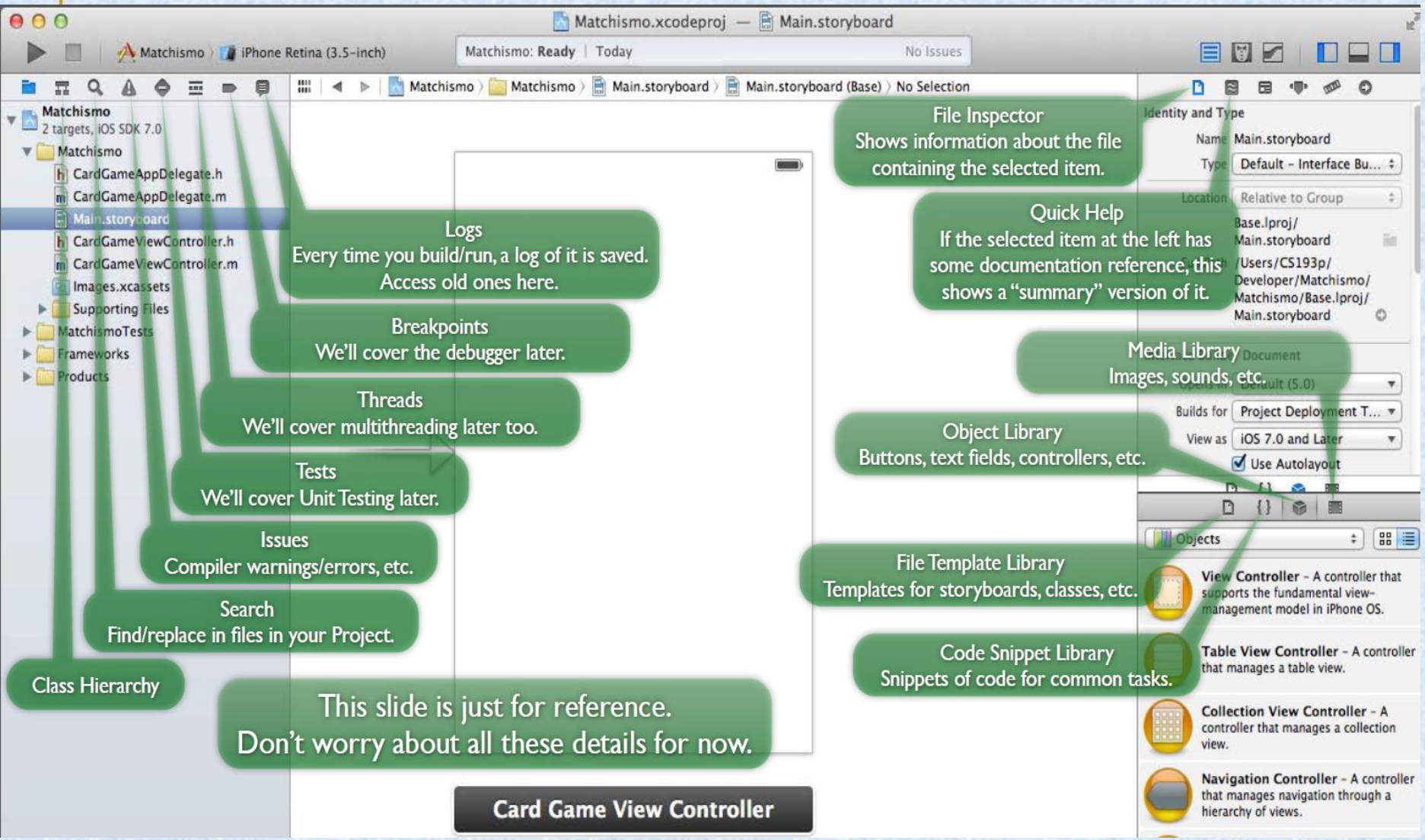
Accelerate	执行复杂的数学与加速框架或图像计算。
CoreBluetooth	核心蓝牙框架
ExternalAccessory	沟通与外部附件框架连接到基于IOS的设备配件。用它来通过30针的基座接口连接的配件，或通过蓝牙互动。
Security	使用安全框架，以确保您的应用程序管理的数据。这个框架定义为保护信息和控制访问软件的C接口。
System	制度的框架提供了一个BSD和POSIX功能，如UNIX系统调用和C库函数的一个子集，。

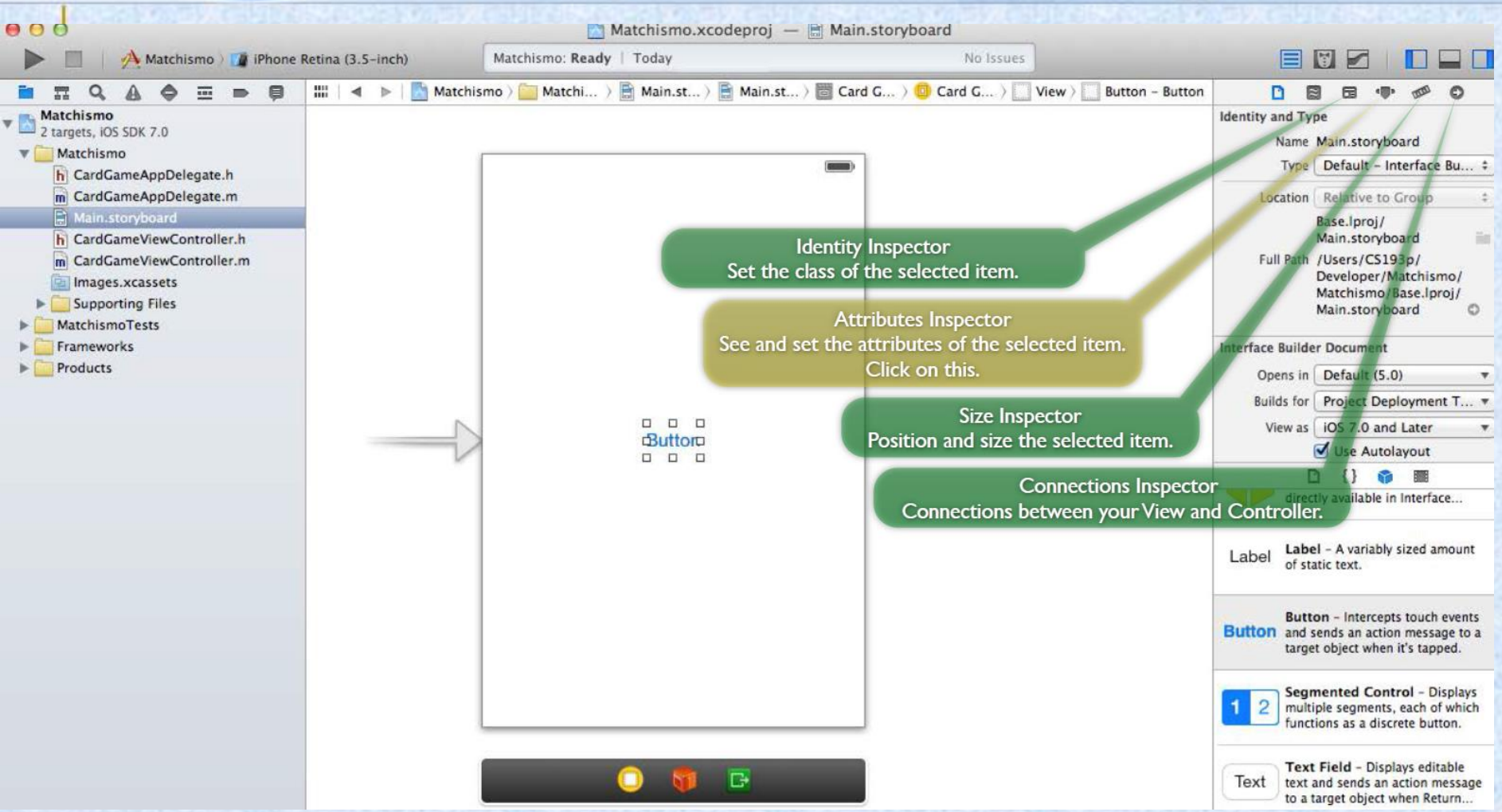
注意：尽可能使用高层框架

六、 IOS应用开发工具


Tools
Xcode 6
Instruments
Language
Object c
Swift
Frameworks
Foundation
Core Data
Map Kit
UIKit
Core Motion
Design Strategies
MVC


This is the Navigator.
It shows all the files in your project in a hierarchical arrangement of folders. The arrangement of folders is conceptual, it does not necessarily match what's in the file system.
This area can also show symbols, search results, breakpoints, issues, etc. (see the icons at the top).
This button shows/hides the Navigator.
Click here to show the Utilities Area (if it's not already showing).
Utilities Area
The top part of this area shows information (identity, attributes, help, connections, dimensions) about the currently selected thing (file, object, etc.) at the left.
The bottom is a library of items (objects, code snippets, file templates).
Card Game View Controller
Matchismo
iPhone Retina (3.5-inch)
Matchismo: Ready
Today
No Issues
Main.storyboard
Main.storyboard (Base)
No Selection
Identity and Type
Target Membership
Stanford CS193p
Fall 2013


Matchismo.xcodeproj — Main.storyboard
Matchismo: Ready | Today
No Issues
Matchismo 〉 Matchismo 〉 Main.storyboard 〉 Main.storyboard (Base) 〉 No Selection
Matchismo
2 targets, iOS SDK 7.0
Matchismo
CardGameAppDelegate.h
CardGameAppDelegate.m
Main.storyboard
CardGameViewController.h
CardGameViewController.m
Images.xcassets
Supporting Files
MatchismoTests
Frameworks
Products
File Inspector
Shows information about the file containing the selected item.
Quick Help
If the selected item at the left has some documentation reference, this shows a "summary" version of it.
Identity and Type
Name Main.storyboard
Type Default – Interface Bu...
Location Relative to Group
Base.lproj/
Main.storyboard
/Users/CS193p/
Developer/Matchismo/
Matchismo/Base.lproj/
Main.storyboard
Logs
Every time you build/run, a log of it is saved.
Access old ones here.
Breakpoints
We'll cover the debugger later.
Threads
We'll cover multithreading later too.
Tests
We'll cover Unit Testing later.
Issues
Compiler warnings/errors, etc.
Search
Find/replace in files in your Project.
Class Hierarchy
Media Library
Images, sounds, etc.
Object Library
Buttons, text fields, controllers, etc.
Builds for Project Deployment T...
View as iOS 7.0 and Later
Use Autolayout
Objects
File Template Library
Templates for storyboards, classes, etc.
Code Snippet Library
Snippets of code for common tasks.
View Controller – A controller that supports the fundamental view-management model in iPhone OS.
Table View Controller – A controller that manages a table view.
Collection View Controller – A controller that manages a collection view.
Navigation Controller – A controller that manages navigation through a hierarchy of views.
This slide is just for reference.
Don't worry about all these details for now.
Card Game View Controller



七、 MVC模式

目的

- 将人机交互从核心功能中分离出来
- 模型对用户来说是不可见的，用户只需要观察视图
- 用户与模型的交互通过控制器提供的安全方法来实现

- MVC（Model-View-Controller）将一个交互式应用程序分成3个组件
  - 模型：包含核心功能和数据 （核心业务逻辑）
  - 视图：向用户显示信息
  - 控制器：处理用户输入
- 变更-传播机制保证了模型和用户界面之间的一致性

模型（Model）

- 封装了内核功能和数据
- 模型对于用户来说是不可见的(M与V独立)
- 模型独立于特定输出表示或者输入方式(M与C独立)
- 用户只能通过控制器操作模型(C是M与V之间的桥梁)

变更-传播机制

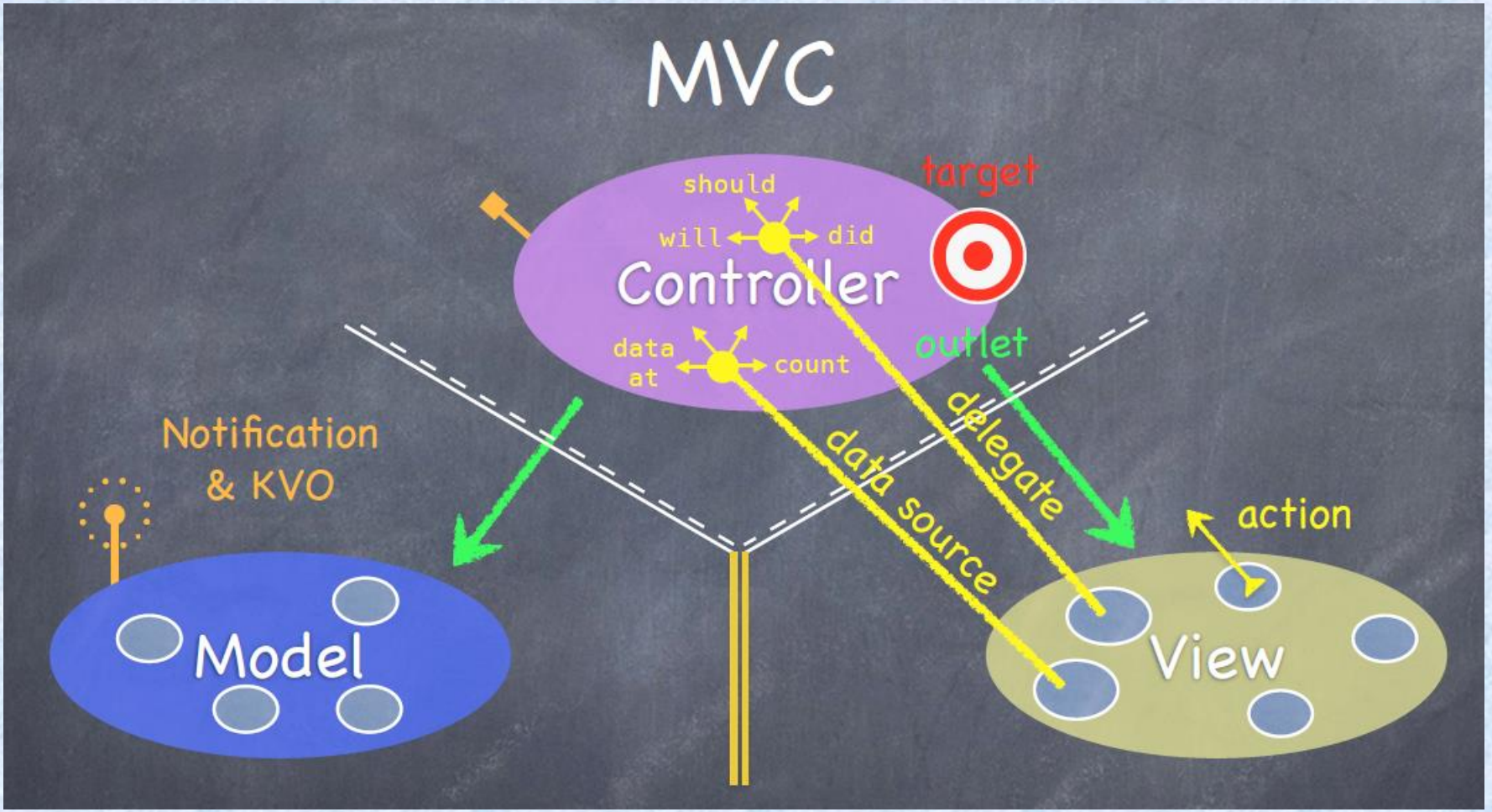
- 一个模型可对应多个视图
  - 如果用户通过一个视图的控制器改变了模型中的数据，那么依赖于该数据的其他视图也应该反映出这样的变化
  - 一旦模型的数据发生了变化，模型需要通知所有相关的视图做出相应的变化
- 维护数据的一致性

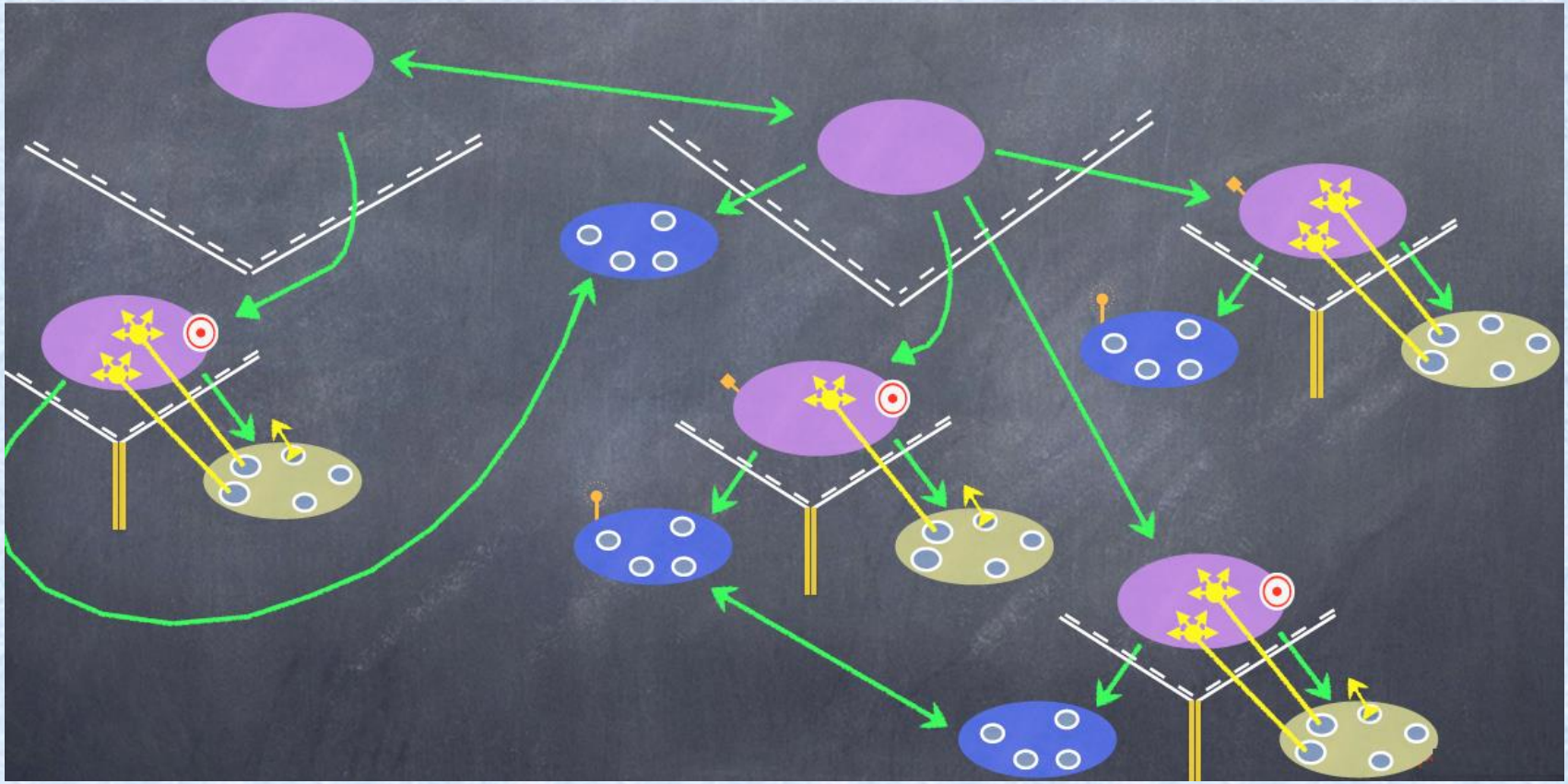
视图（View）

- 向用户显示信息
  - 不同的视图使用不同的方法呈现信息
  - 每个视图组件都有一个更新函数，这个函数被模型变更通知激活
  - 这个函数被激活（此时模型已经改变）后，将使得视图重新和模型一致
  - 在初始化阶段，视图向模型登记请求变更通知（表）
- 从模型获得数据
  - 通过状态查询函数实现
  - 例如：定时刷新

控制器（Controller）

- 每个视图有一个相关的控制器组件(一一对应)
- 控制器组件接受事件，并翻译成输入
  - 事件如何发送到控制器由用户界面平台决定
  - 事件被翻译成为对模型或者视图的请求
  - 如果控制器的行为依赖于模型的状态，那么控制器也需要向模型登记请求变更通知
  - 例如：用户点击按钮，按钮的事件响应函数将采取相应的措施处理用户要求
- 用户仅仅通过控制器与系统交互


MVC
should
will
did
Controller
target
data
at
count
outlet
Notification
& KVO
data source
delegate
action
Model
View



多个MVC协同工作

八、 参考资料

- Xcode联机文档
- 斯坦福大学iOS7,iOS8公开课 (视频)