Creating Modern Cocoa Apps

Session 227
Tony Parker
Software Engineer, Cocoa Frameworks
A Modern Cocoa App
A Modern Cocoa App
Introducing Lister

Built with
• Storyboards
• Auto Layout
• NSDocument
OS X and iOS
Objective-C and Swift
Adding to do items:

- Outline
- Demos
- Slides
- Software
adding to do items

- Outline
- Demos
- Slides
- Software
Learn about Cocoa
Meet engineers at labs
Beer Bash
Keynote
There are no items in your list.
There are no items in your list.
- That new requirement
- That one with the wrong behavior
- That one with the crash
That new requirement
That one with the wrong behavior
That one with the crash
File

New
Add Files...
Open...
Open Recent
Open Quickly...
Close Window
Close Tab
Close Document
Close Workspace

Save
Duplicate...
Revert to Saved...
Unlock...
Export...
Show in Finder
Open with External Editor

Project...

Window

File...
Target...
Workspace...
Group
Group from Selection
Choose a template for your new project:

iOS
- Application
- Framework & Library
- Other
- Apple Internal

OS X
- Application
- Framework & Library
- System Plug-in
- Other

Cocoa Application
This template creates a Cocoa application for the OS X platform.
Choose options for your new project:

Product Name:ModernCocoaApp
Organization Name:My Company
Organization Identifier:com.mycompany
Bundle Identifier:com.mycompany.ModernCocoaApp
Language:Swift

- Use Storyboards
- Create Document-Based Application

Document Extension:mydoc
- Use Core Data
Your document contents here
Agenda
Agenda

Getting started

Your document contents here
Agenda

Getting started
Adding more features
Agenda

Getting started
Adding more features
Where to go next
Getting Started
Model, View, Controller
Model, View, Controller
Model, View, Controller

View

Model
Model, View, Controller

View

Controller

Model
Storyboards
Storyboards

Starting point for creating views and controllers
Storyboards

Starting point for creating views and controllers
Consist of two parts
Storyboards

Starting point for creating views and controllers
Consist of two parts
- Scenes
  - Part of your user interface
Storyboards

Starting point for creating views and controllers
Consist of two parts

• Scenes
  - Part of your user interface

• Segues
  - Transition from one scene to next
Your document contents here
```swift
// LoginViewController.swift
// ModernCocoaApp

import Cocoa

class LoginViewController: NSViewController {
    init(nibName nibNameOrNil: String?, bundle nibBundleOrNil: NSBundle) {
        super.init(nibName:nibNameOrNil, bundle: nibBundleOrNil)
    }
}
```
Storyboards

Great for rapid prototyping
Storyboards

Great for rapid prototyping

Compose to form more complicated interfaces
<table>
<thead>
<tr>
<th>Related Sessions</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Storyboards and Controllers for OS X</td>
</tr>
</tbody>
</table>
Auto Layout
Auto Layout

How views are placed in user interface
Auto Layout

How views are placed in user interface
Changes size and placement of views as content changes
Auto Layout

How views are placed in user interface
Changes size and placement of views as content changes
Specify relationships between views using constraints
Auto Layout

Allows flexibility as design changes
Auto Layout

 Allows flexibility as design changes
 Simplifies localization
Auto Layout

Allows flexibility as design changes
Simplifies localization
Xcode has rich interface for working with Auto Layout
• Adding new constraints
• Understanding how constraints interact
• Previewing results
• Debugging issues
Related Sessions

- Taking Control of Auto Layout in Xcode 5
  WWDC 2013
Documents and Data
Documents and Data

The reason to use your app
Documents and Data

The reason to use your app
Deciding how to store data will drive design
Documents and Data

The reason to use your app
Deciding how to store data will drive design
Two major kinds of data storage in a Cocoa app
• Shoebox
• Document-based
Shoebox Apps
Shoebox Apps

Store data in library or container

• Usually hidden from user
Shoebox Apps

Store data in library or container
• Usually hidden from user
Presents data in single window
Shoebox Apps

Store data in library or container
  • Usually hidden from user
Presents data in single window
Great for mix and match of data
Shoebox Apps

Store data in library or container
• Usually hidden from user

Presents data in single window

Great for mix and match of data
Cocoa provides Core Data
Core Data
Core Data

Generalized object graph and persistence framework
Core Data

Generalized object graph and persistence framework
Efficient performance
Core Data

Generalized object graph and persistence framework
Efficient performance
Change tracking and undo
Core Data

Generalized object graph and persistence framework
Efficient performance
Change tracking and undo
Relationship maintenance
Core Data

Generalized object graph and persistence framework
Efficient performance
Change tracking and undo
Relationship maintenance
Schema migration
Core Data

Generalized object graph and persistence framework
Efficient performance
Change tracking and undo
Relationship maintenance
Schema migration
Query compilation
## Related Sessions

<table>
<thead>
<tr>
<th>Session</th>
<th>Location</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>What’s New in Core Data</td>
<td>Pacific Heights</td>
<td>Thursday 9:00AM</td>
</tr>
</tbody>
</table>
Document-Based Apps
Document-Based Apps

Stores user data in named file container
Document-Based Apps

Stores user data in named file container
Documents may be local or in iCloud
Document-Based Apps

Stores user data in named file container
Documents may be local or in iCloud
Documents are not generally related to each other
Document-Based Apps

Stores user data in named file container
Documents may be local or in iCloud
Documents are not generally related to each other
Users are interested in a few at a time
Document-Based Apps

Stores user data in named file container
Documents may be local or in iCloud
Documents are not generally related to each other
Users are interested in a few at a time
Cocoa provides a document architecture
Cocoa Document Architecture
Cocoa Document Architecture

Provides core features with little or no additional code
• Auto Save and Versions
• iCloud
• Asynchronous reading and writing
• Undo
Cocoa Document Architecture

Provides core features with little or no additional code

- Auto Save and Versions
- iCloud
- Asynchronous reading and writing
- Undo

Customizable to your app’s needs
Cocoa Document Architecture

Provides core features with little or no additional code
- Auto Save and Versions
- iCloud
- Asynchronous reading and writing
- Undo

Customizable to your app’s needs

Start by subclassing NSDocument
- Creates, presents, and stores document data
NSDocument

Subclass and enable iCloud, Auto Save, and Versions
NSDocument
Subclass and enable iCloud, Auto Save, and Versions

class ListDocument : NSDocument {

class ListDocument : NSDocument {
    // Default is an empty list
    var list = List()
class ListDocument : NSDocument {
    // Default is an empty list
    var list = List()

    override class func autosavesInPlace() -> Bool {
        return true
    }
}
override func `dataOf`Type(typeName: String,
     error outError: NSErrorPointer) -> NSData? {

}
override func `dataOfT`ype(typeName: String, error outError: NSErrorPointer) -> NSData? {
    if let data = NSKeyedArchiver.archivedDataWithRootObject(list) {
        return data
    }
}
override func `dataOfType`(typeName: String, 
    error outError: NSErrorPointer) -> NSData? {
    if let data = NSKeyedArchiver.archivedDataWithRootObject(list) {
        return data
    }
    // Set error here
    return nil;
}
override func `readFromData`(data: NSData, ofType typeName: String, error outError: NSErrorPointer) -> Bool {
}

NSDocument
Create document from data on disk

override func readFromData(data: NSData, ofType typeName: String, error outError: NSErrorPointer) -> Bool {
    if let deserializedList = NSKeyedUnarchiver.unarchiveObjectWithData(data) as? List {
        
    } else {
        
    }
}
override func readFromData(data: NSData, ofType typeName: String, error outError: NSErrorPointer) -> Bool {
    if let deserializedList = 
        NSKeyedUnarchiver.unarchiveObjectWithData(data) as? List
    {
        list = deserializedList
        return true
    } else {
        
    }
}
override func `readFromData`(data: NSData, ofType typeName: String, error outError: NSErrorPointer) -> Bool {
    if let deserializedList = 
        NSKeyedUnarchiver.unarchiveObjectWithData(data) as? List {
        list = deserializedList
        return true
    } else {
        // Set error here
        return false
    }
}
Related Sessions

- Auto Save and Versions in Mac OS X 10.7 Lion  

WWDC 2011
Other User Data
Other User Data

Preferences—NSUserDefaults

• Application preferences
• System preferences
  - Language
  - Locale
Other User Data

Preferences—NSUserDefaults
• Application preferences
• System preferences
  - Language
  - Locale

Network—NSURLSession
Other User Data

Preferences—NSUserDefaults
• Application preferences
• System preferences
  - Language
  - Locale

Network—NSURLSession

iCloud key-value store—NSUbiquitousKeyValueStore
• Small amounts of data, available on all iCloud devices
Other User Data

Preferences—NSUserDefaults
- Application preferences
- System preferences
  - Language
  - Locale

Network—NSURLSession

iCloud key-value store—NSUbiquitousKeyValueStore
- Small amounts of data, available on all iCloud devices

CloudKit
Demo

Getting started in Lister

Alex Migicovsky
Developer Publications Engineer
Adding More Features
Auto Localization
Auto Localization

Built on top of Auto Layout
Auto Localization

Built on top of Auto Layout
Shares the same UI design but replaces the strings
Auto Localization

Built on top of Auto Layout
Shares the same UI design but replaces the strings
Resizes views appropriately when words are different lengths
Auto Localization

Built on top of Auto Layout
Shares the same UI design but replaces the strings
Resizes views appropriately when words are different lengths
Support both left to right and right to left
Application Bundle Layout

Lister.app/
Contents/
Resources/
Application Bundle Layout

Lister.app/
Contents/
Resources/

Base.lproj/
Main.storyboardc  UI, development language
Application Bundle Layout

Lister.app/
  Contents/
    Resources/

Base.lproj/
  Main.storyboardc  UI, development language

es.lproj/
  Main.strings      Spanish strings
Strings File

"key" = "value";
Strings File

"9eV-3V-M02.title" = "Artstay:";}
Strings File

"9eV-3V-M02.title" = "Artstay:";
Strings File

"9eV-3V-M02.title" = "Artstay:";
Strings File

"9eV-3V-M02.title" = "Artstay:";
Strings File

"9eV-3V-M02.title" = "Artstay:";

"P3b-i2-9LW.title" = "Endway:";
Handoff

Start working on one device and continue on another
Handoff

Start working on one device and continue on another
Support is built in to NSDocument and UIDocument
Handoff

Start working on one device and continue on another
Support is built in to NSDocument and UIDocument
Requires only a small amount of code to enable
<table>
<thead>
<tr>
<th>Key</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>▼ Information Property List</td>
<td>Dictionary</td>
<td>(19 items)</td>
</tr>
<tr>
<td>▼ Localization native development region</td>
<td>String</td>
<td>en</td>
</tr>
<tr>
<td>▼ NSUserActivityTypes</td>
<td>Array</td>
<td>(1 item)</td>
</tr>
<tr>
<td>Item 0</td>
<td>String</td>
<td>com.example.apple-samplecode.Lister</td>
</tr>
<tr>
<td>▼ Document types</td>
<td>Array</td>
<td>(1 item)</td>
</tr>
<tr>
<td>▼ Item 0 (list)</td>
<td>Dictionary</td>
<td>(8 items)</td>
</tr>
<tr>
<td>Handler rank</td>
<td>String</td>
<td>Owner</td>
</tr>
<tr>
<td>Role</td>
<td>String</td>
<td>Editor</td>
</tr>
<tr>
<td>Icon File Name</td>
<td>String</td>
<td>ListerDocumentIcon</td>
</tr>
<tr>
<td>Document Type Name</td>
<td>String</td>
<td>list</td>
</tr>
<tr>
<td>▼ Document Content Type UTIs</td>
<td>Array</td>
<td>(1 item)</td>
</tr>
<tr>
<td>Cocoa NSDocument Class</td>
<td>String</td>
<td>ListerOSX.ListDocument</td>
</tr>
<tr>
<td>Document is a package or bundle</td>
<td>Boolean</td>
<td>NO</td>
</tr>
<tr>
<td>NSUbiquitousDocumentUserActivityType</td>
<td>String</td>
<td>com.example.apple-samplecode.Lister</td>
</tr>
<tr>
<td>Key</td>
<td>Type</td>
<td>Value</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>Information Property List</td>
<td>Dictionary</td>
<td>(19 items)</td>
</tr>
<tr>
<td>Localization native development region</td>
<td>String</td>
<td>en</td>
</tr>
<tr>
<td>NSUserActivityTypes</td>
<td>Array</td>
<td>(1 item)</td>
</tr>
<tr>
<td>Item 0</td>
<td>String</td>
<td>com.example.apple-samplecode.Lister</td>
</tr>
<tr>
<td>Document type</td>
<td>Dictionary</td>
<td>(8 items)</td>
</tr>
<tr>
<td>Item 0 (list)</td>
<td>String</td>
<td>Owner</td>
</tr>
<tr>
<td>Role</td>
<td>String</td>
<td>Editor</td>
</tr>
<tr>
<td>Icon File Name</td>
<td>String</td>
<td>ListDocumentIcon</td>
</tr>
<tr>
<td>Document Type Name</td>
<td>String</td>
<td>list</td>
</tr>
<tr>
<td>Document Content Type UTIs</td>
<td>Array</td>
<td>(1 item)</td>
</tr>
<tr>
<td>Cocoa NSDocument Class</td>
<td>String</td>
<td>ListerOSX.ListDocument</td>
</tr>
<tr>
<td>Document is a package or bundle</td>
<td>Boolean</td>
<td>NO</td>
</tr>
<tr>
<td>NSUbiquitousDocumentUserActivityType</td>
<td>String</td>
<td>com.example.apple-samplecode.Lister</td>
</tr>
<tr>
<td>Key</td>
<td>Type</td>
<td>Value</td>
</tr>
<tr>
<td>-------------------------------------------------</td>
<td>----------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>▼ Information Property List</td>
<td>Dictionary (19 items)</td>
<td></td>
</tr>
<tr>
<td>Localization native development region</td>
<td>String</td>
<td>en</td>
</tr>
<tr>
<td>▼ NSUserActivityTypes</td>
<td>Array</td>
<td>(1 item)</td>
</tr>
<tr>
<td>Item 0</td>
<td>String</td>
<td>com.example.apple-samplecode.Lister</td>
</tr>
<tr>
<td>▼ Document types</td>
<td>Array</td>
<td>(1 item)</td>
</tr>
<tr>
<td>Item 0 (list)</td>
<td>Dictionary (8 items)</td>
<td></td>
</tr>
<tr>
<td>Handler rank</td>
<td>String</td>
<td>Owner</td>
</tr>
<tr>
<td>Role</td>
<td>String</td>
<td>Editor</td>
</tr>
<tr>
<td>Icon File Name</td>
<td>String</td>
<td>ListerDocumentIcon</td>
</tr>
<tr>
<td>Document Type Name</td>
<td>String</td>
<td>list</td>
</tr>
<tr>
<td>▼ Document Content Type UTIs</td>
<td>String</td>
<td>ListerOSX.ListDocument</td>
</tr>
<tr>
<td>Cocoa NSDocument Class</td>
<td>String</td>
<td>ListerOSX.ListDocument</td>
</tr>
<tr>
<td>Document is a package or bundle</td>
<td>Boolean</td>
<td>NO</td>
</tr>
</tbody>
</table>
Handoff

Lower-level API—NSUserActivity

- Native app to web app
- Web app to native app
- Provide additional state beyond document content
- Send custom data using streams
## Related Sessions

- **Adopting Handoff on iOS and OS X**
  - Mission
  - Wednesday 2:00PM
Sharing

Provide a way to share with social networks

• Twitter
• Facebook
• …and more
• Single sign-on
Sharing

Provide a way to share with social networks

• Twitter
• Facebook
• …and more
• Single sign-on

Integration with extensions from other applications
- What do you call 1000 Apple engineers at a conference?
- There once was a man from Cupertino...
- An engineer and a mathematician walk into a bar...
What do you call 1000 Apple engineers at a conference?

There once was a man from Cupertino...

An engineer and a mathematician walk into a bar...
NSSharingServicePicker
NSSharingServicePicker

Shares NSImage, NSURL, NSString, NSAttributedString
NSSharingServicePicker

Shares NSImage, NSURL, NSString, NSAttributedString

Two steps to sharing

• Put a share button in your app
• Present the picker and specify the data to share
NSSharingServicePicker

-(IBAction)shareIt:(NSButton *)sender {

}
NSSharingServicePicker

-(IBAction)shareIt:(NSButton *)sender {

    NSString *greeting = @"Hello, world!";

}

NSSharingServicePicker

-(IBAction)shareIt:(NSButton *)sender {

    NSString *greeting = @"Hello, world!";

    NSSharingServicePicker *picker =
    [[NSSharingServicePicker alloc] initWithItems:@[greeting]];
}

NSSharingServicePicker

-(IBAction)shareIt:(NSButton *)sender {

    NSString *greeting = @"Hello, world!";

    NSSharingServicePicker *picker =
    [[NSSharingServicePicker alloc] initWithItems:@[greeting]];

    [picker showRelativeToRect:NSZeroRect
    ofView:sender
    preferredEdge:NSMinYEdge];
}

More About Sharing

Your app can provide extensions too

- Action
- Finder
- Sharing
- Today
## Related Sessions

<table>
<thead>
<tr>
<th>Session</th>
<th>Type</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Creating Extensions for iOS and OS X, Part 1</td>
<td>Mission</td>
<td>Tuesday 2:00PM</td>
</tr>
<tr>
<td>Creating Extensions for iOS and OS X, Part 2</td>
<td>Mission</td>
<td>Wednesday 11:30AM</td>
</tr>
<tr>
<td>Integrating with Facebook, Twitter, and Sina Weibo</td>
<td></td>
<td>WWDC 2012</td>
</tr>
</tbody>
</table>
Demo
Sharing in Lister

Alex Migicovsky
Developer Publications Engineer
What to Do Next
Adding More Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Cocoa Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Undo</td>
<td>NSUndoManager</td>
</tr>
<tr>
<td>Drag &amp; Drop, Copy &amp; Paste</td>
<td>NSPasteboard</td>
</tr>
<tr>
<td>Printing</td>
<td>NSPrinting</td>
</tr>
</tbody>
</table>
## Adding More Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Cocoa Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Undo</td>
<td>NSUndoManager</td>
</tr>
</tbody>
</table>
## Adding More Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Cocoa Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Undo</td>
<td>NSUndoManager</td>
</tr>
<tr>
<td>Drag &amp; Drop, Copy &amp; Paste</td>
<td>NSPasteboard</td>
</tr>
</tbody>
</table>
## Adding More Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Cocoa Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Undo</td>
<td>NSUndoManager</td>
</tr>
<tr>
<td>Drag &amp; Drop, Copy &amp; Paste</td>
<td>NSPasteboard</td>
</tr>
<tr>
<td>Printing</td>
<td>NSPrinting</td>
</tr>
</tbody>
</table>
Adding More Features

Energy efficiency
Adding More Features

Energy efficiency
Three key rules for better battery life
• Stay idle as long as possible
• Avoid unnecessary work
• Return to idle as quickly as possible
## Related Sessions

<table>
<thead>
<tr>
<th>Session Title</th>
<th>Location</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Writing Energy Efficient Code, Part 1</td>
<td>Russian Hill</td>
<td>Wednesday 10:15AM</td>
</tr>
</tbody>
</table>
## Adding More Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Cocoa Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full screen</td>
<td>NSWindow</td>
</tr>
<tr>
<td>Resume</td>
<td>NSWindowRestoration</td>
</tr>
<tr>
<td>Progress reporting</td>
<td>NSProgress</td>
</tr>
<tr>
<td>Gestures</td>
<td>NSGestureRecognizer</td>
</tr>
<tr>
<td>Accessibility</td>
<td>NSAccessibility</td>
</tr>
</tbody>
</table>
## Adding More Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Cocoa Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full screen</td>
<td>NSWindow</td>
</tr>
</tbody>
</table>
## Adding More Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Cocoa Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full screen</td>
<td>NSWindow</td>
</tr>
<tr>
<td>Resume</td>
<td>NSWindowRestoration</td>
</tr>
</tbody>
</table>
## Adding More Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Cocoa Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full screen</td>
<td>NSWindow</td>
</tr>
<tr>
<td>Resume</td>
<td>NSWindowRestoration</td>
</tr>
<tr>
<td>Progress reporting</td>
<td>NSProgress</td>
</tr>
</tbody>
</table>
## Adding More Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Cocoa Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full screen</td>
<td>NSWindow</td>
</tr>
<tr>
<td>Resume</td>
<td>NSWindowRestoration</td>
</tr>
<tr>
<td>Progress reporting</td>
<td>NSProgress</td>
</tr>
<tr>
<td>Gestures</td>
<td>NSGestureRecognizer</td>
</tr>
</tbody>
</table>
## Adding More Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Cocoa Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full screen</td>
<td>NSWindow</td>
</tr>
<tr>
<td>Resume</td>
<td>NSWindowRestoration</td>
</tr>
<tr>
<td>Progress reporting</td>
<td>NSProgress</td>
</tr>
<tr>
<td>Gestures</td>
<td>NSGestureRecognizer</td>
</tr>
<tr>
<td>Accessibility</td>
<td>NSAccessibility</td>
</tr>
</tbody>
</table>
Summary

Get started with a great foundation
Summary

Get started with a great foundation
Take advantage of the features the frameworks provide
Summary

Get started with a great foundation
Take advantage of the features the frameworks provide
Your app will be in great shape for the future
Jake Behrens
App Frameworks Evangelist
behrens@apple.com

Documentation
Start Developing Mac Apps Today
http://developer.apple.com

Lister

Apple Developer Forums
http://devforums.apple.com
<table>
<thead>
<tr>
<th>Session</th>
<th>Location</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accessibility on OS X</td>
<td>Russian Hill</td>
<td>Tuesday 2:00PM</td>
</tr>
<tr>
<td>Full Screen and Aqua Changes</td>
<td>WWDC 2011</td>
<td></td>
</tr>
<tr>
<td>Resume and Automatic Termination in Lion</td>
<td>WWDC 2011</td>
<td></td>
</tr>
</tbody>
</table>
## Labs

<table>
<thead>
<tr>
<th>Lab</th>
<th>Location</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>View Controllers and Cocoa Lab</td>
<td>Frameworks Lab B</td>
<td>Thursday 1:30PM</td>
</tr>
<tr>
<td>Extensions Lab</td>
<td>Frameworks Lab B</td>
<td>Thursday 2:00PM</td>
</tr>
<tr>
<td>Cocoa Lab</td>
<td>Frameworks Lab B</td>
<td>Thursday 4:30PM</td>
</tr>
</tbody>
</table>