

Building iAd Rich Media Ads with iAd Producer

Introducing iAd Producer

Session 506

Mark Malone

iAd

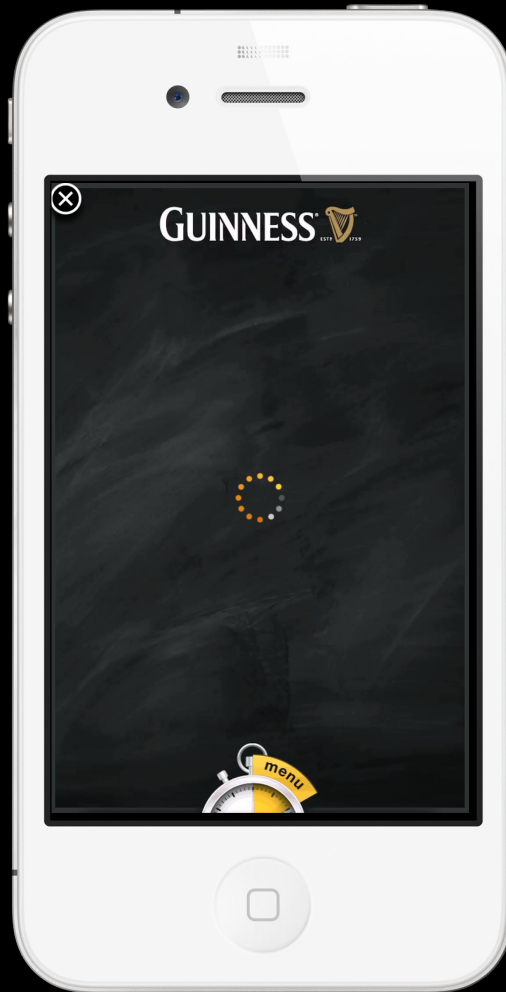
These are confidential sessions—please refrain from streaming, blogging, or taking pictures

Agenda

What we'll cover

- The iAd experience
- Code-free design
- Going beyond the built-in
- Sharing with others

iAd Experience











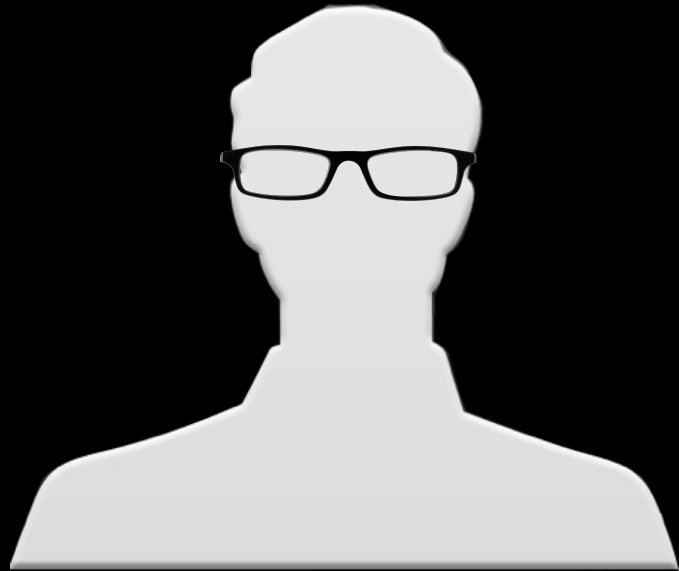






iAd Producer

Designed for...



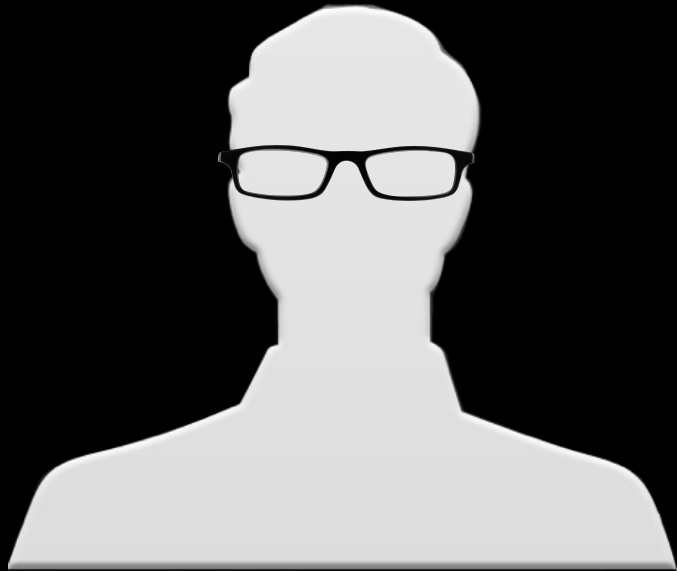
Creative Designer



Creative Developer

iAd Producer

Designed for designers



Creative Designer

- Interactive prototyping
- Collaborative design sessions
- Template pages, animation, and effects
- Demo on device
- Fast-track to development

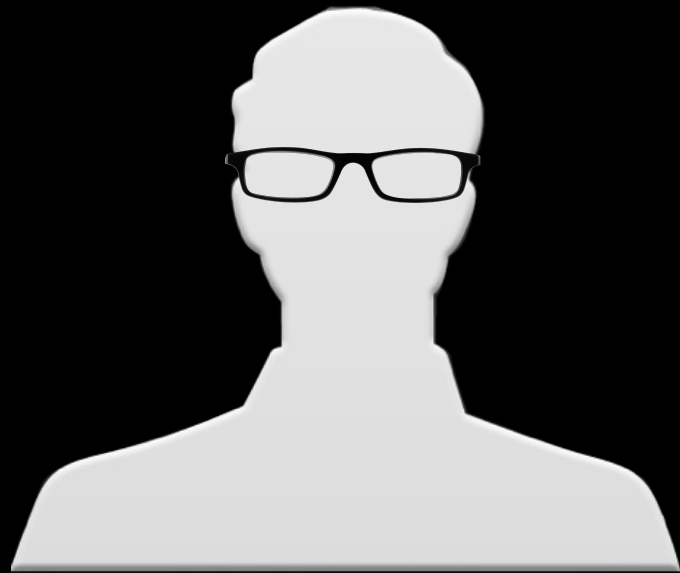
iAd Producer

Designed for developers

- Built-in performance and compatibility
- Extendable built-in objects
- Code import and custom plug-ins
- Test and debug in iPhone simulator
- Package and publish in one step



Creative Developer



Creative Designer



Creative Developer

iAd Producer

Designed for you



Designer / Developer

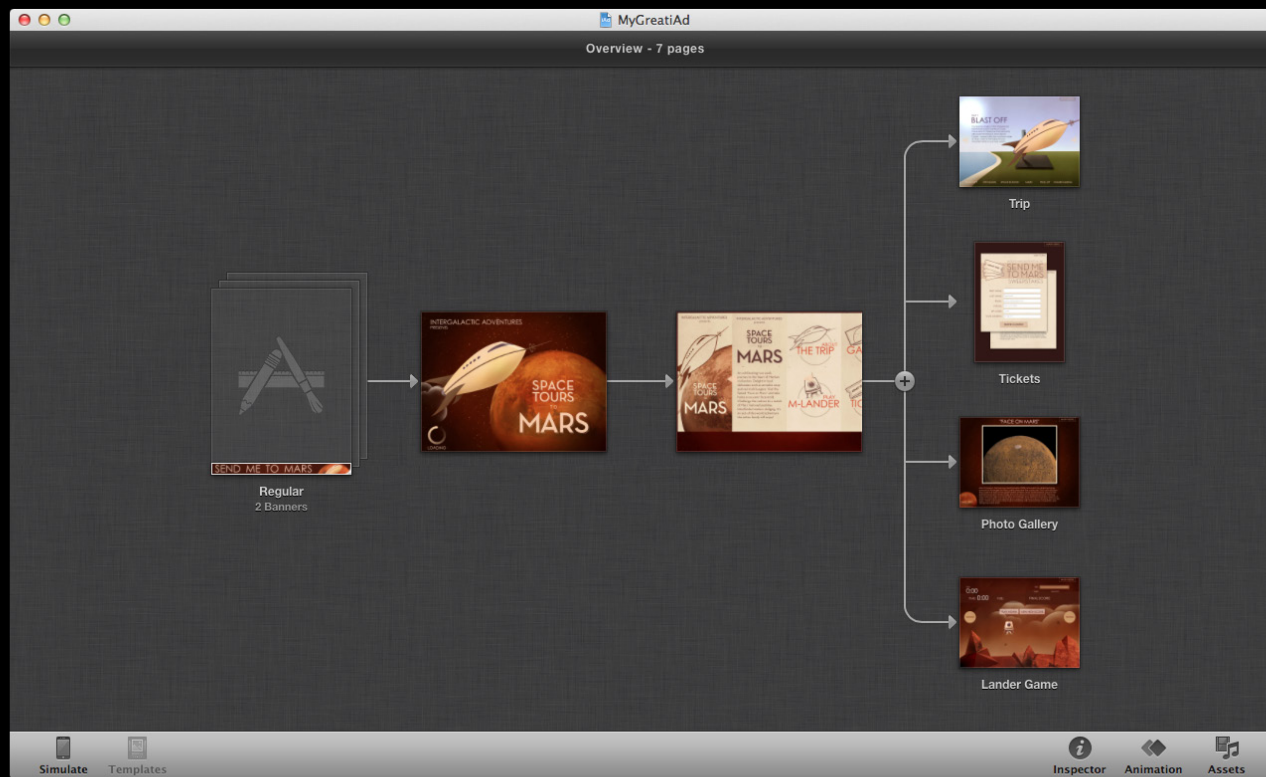


Design → Develop → Deploy



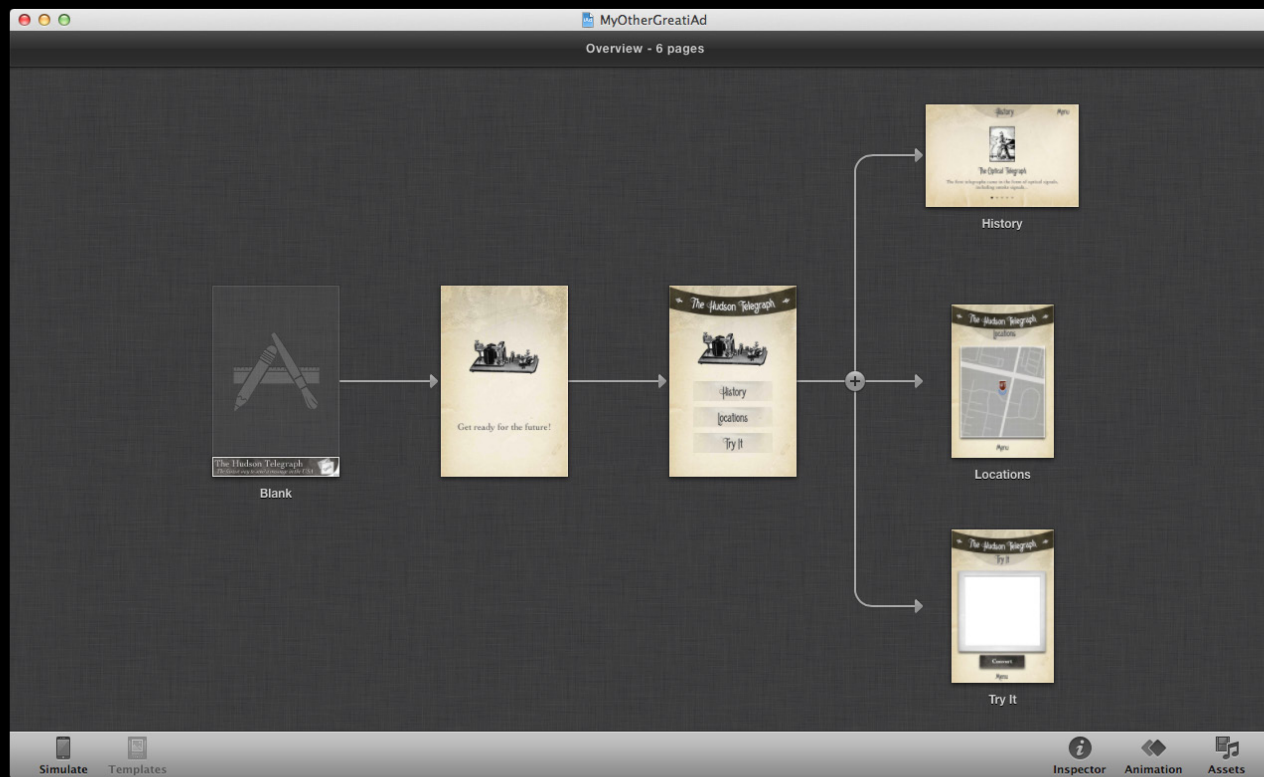
Design → Develop → Deploy

Design iAd for iPad



Design

iAd for iPhone and iPod touch

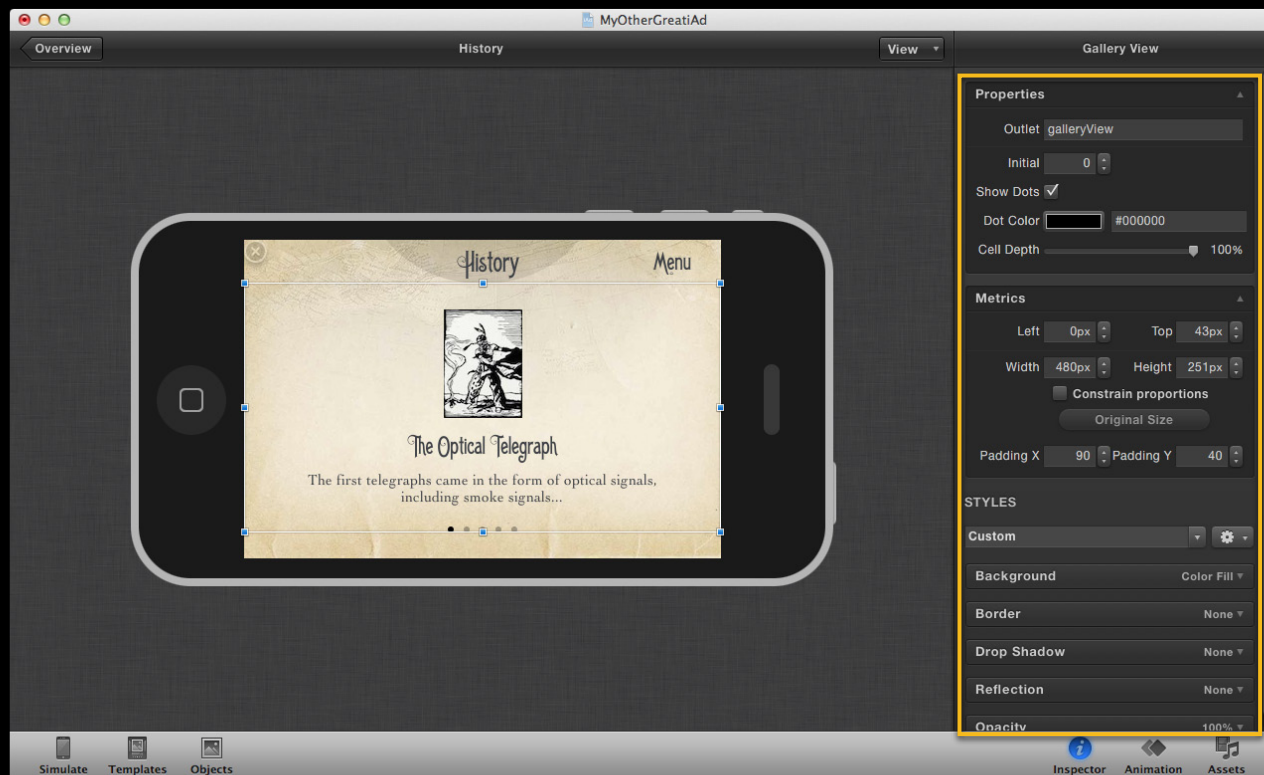


Design

Using pre-built objects



Design Code-free style



Design

Code-free animation

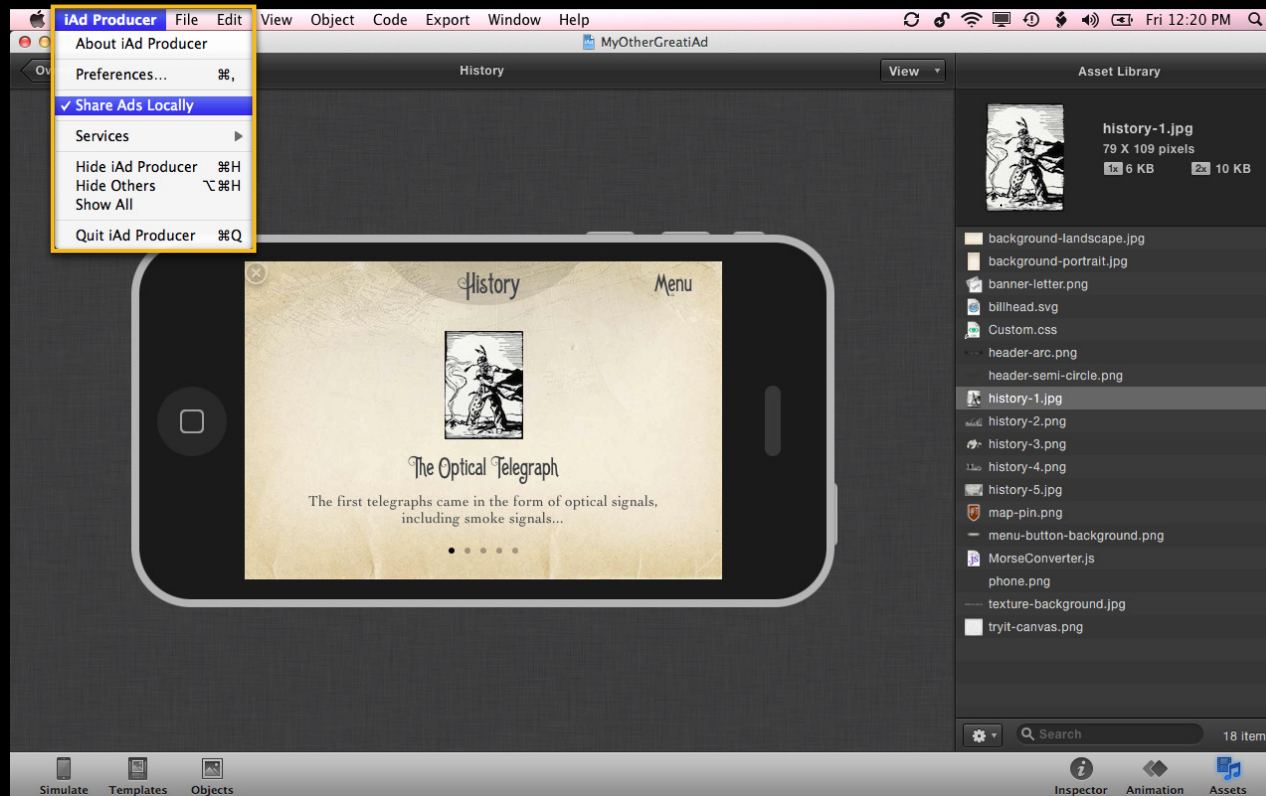


Design

Organized assets



Design Experience on device



Design

Experience on device



Demo

Design an iAd



Design → Develop → Deploy



Design —————→ **Deploy**



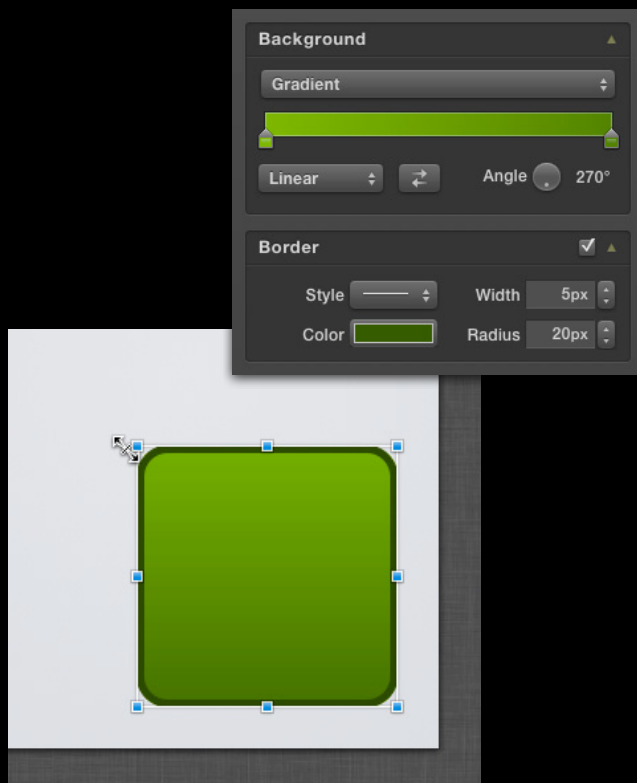
Design → **Develop** → Deploy





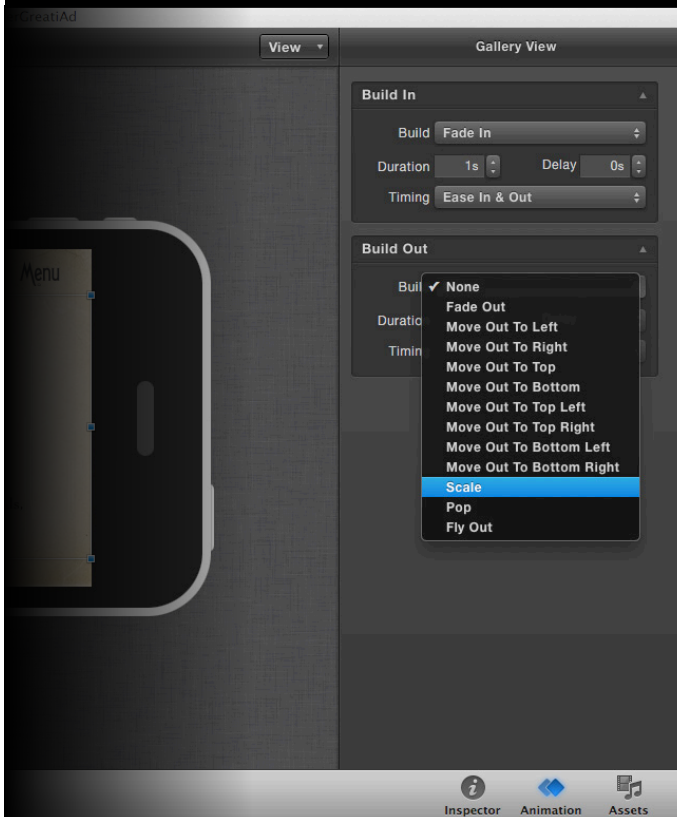


Develop CSS for style



```
#green-box {  
  background: -webkit-gradient(  
    linear, 0% 0%, 0% 100%, color-stop(0%, #5da200), color-stop(100%, #366301)  
  );  
  border-radius: 20px;  
  border-color: #213d00;  
  border: 5px solid #213d00;  
  width: 167px;  
  height: 167px;  
  left: 127px;  
  top: 287px;  
}
```

Develop CSS for animation



```
#History [ad-outlet="HistoryFlow"]  
{  
  -webkit-transition-property: top;  
  -webkit-transition-duration: 2s;  
  top: 0px;  
}
```



HTML @font-face CORS/Access Control
Web Workers DOM Manifests JPG
CSS Animation SVG DOM Storage
WebGL CSS PNG MPEG CSS
Selectors API Drag and Drop Touch
Events Media Queries <audio> CSS
Transitions Web SQL Database
JavaScript data:URIs Cross-document
Messaging Canvas Web SQL Database
Offline Applications CSS Transforms
GeoLocation Native JSON WAI-ARIA
<video> Flexible Box Layout

Develop

Leverage core web technologies



Inline Audio/Video



CSS Visual Effects



Device Orientation



Gyro



Accelerometer



WebGL



Geolocation



Touch Events

Develop

Web technologies



+



Develop iAd JS language features

- Subclassing
- Mix-ins
- Synthesized properties
- Property observing
- View controllers
- View hierarchy
- Event handling
- System bindings

iAd JS Developer Library

iAd JS Programming Guide

Table of Contents

- Introduction
- ▶ Creating a Simple Ad: A Tutorial
- ▶ Anatomy of an Ad
- ▶ The iAd Bundle
- The Banner
- ▼ iAd JS Programming
 - Subclassing
 - Using Mixins
 - Synthesizing Properties
 - Observing Property Changes
 - Using View Controllers
 - Creating and Displaying Views
- ▼ Handling Events
 - Observing Control Events
 - Registering for Events
 - Dispatching Events
- ▶ iAd JS Declarative Interface
- ▼ iAd System Bindings
 - Saving Images
 - ▶ Purchasing Content from iTunes
 - Adding Calendar Events
 - Composing Email and SMS Messages
 - Handling Shake Events
 - Handling Orientation Changes
 - ▶ Displaying Annotated Maps
 - Playing Videos
 - Performance Tuning
 - ▶ Creating Universal Ads
- Appendix A: Fallback Map API

Previous Next

iAd JS Programming

To master iAd JS programming and the declarative interface, you need to become familiar with the iAd JS language features and classes. iAd JS provides additional features such as subclassing, mixins, and synthesized properties to improve JavaScript programming. The object-oriented design of its classes is very similar to UIKit used by native applications. Every iAd JS ad uses controls, views, and view controllers to build the user interface. A view controller is used to manage a full-screen view and its subviews. iAd JS uses a delegate design pattern and dispatches high-level events so your ad can be more responsive to users.

This chapter describes the programming language features and core iAd JS JavaScript classes. You should read this chapter even if you use iAd Producer or primarily the declarative interface because it provides a description of the core objects you use to build your ad.

Read ["iAd JS Declarative Interface"](#) for more information on the declarative approach to creating views, adding visual effects, and accessing view controllers.

Subclassing

iAd JS classes can extend existing classes through a common object-oriented language mechanism called **subclassing**. The new class you define **inherits** methods and properties from its parent class. The new class simply adds to or modifies the inherited methods. It doesn't need to duplicate inherited code.

All iAd JS classes inherit from some class forming a class hierarchy with a single root. Every class has a single superclass and may have multiple subclasses. The root class is `iAd.Object` which provides common methods such as `callSuper` and others in support of key-value observing. [Figure 5-1](#) shows just the `iAd.View` portion of the iAd JS class hierarchy. For example, `iAd.CarouselView` extends `iAd.View`—that is, it inherits all properties and methods from `iAd.View` and adds additional class-specific ones. It may also override inherited methods.

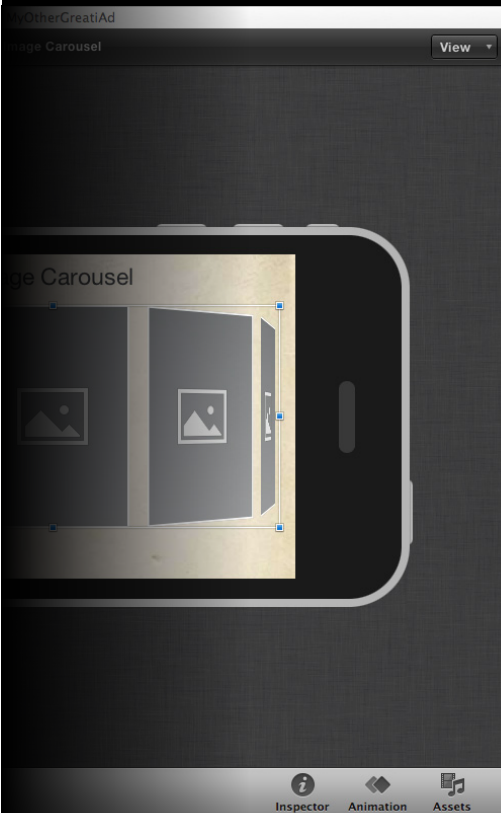
Figure 5-1 The View class hierarchy

```
graph TD; iAdControl[iAd.Control] --- iAdCarouselView[iAd.CarouselView]; iAdControl --- iAdPageControl[iAd.PageControl]; iAdButton[iAd.Button] --- iAdPageControl;
```

App Download Carousel View
Animations Search Bar Movie
Download Segmented Control
Picker View Set Wallpaper Switch
View Shake Event Map View App
Download Scroll View View
Transforms Map Annotation Flow
View Set Home Screen Image View
Slider Control User Location Table
View Camera Multiple Orientation
Page Control Tab Bar View Tool Bar
Touch Events Photo Library Access
Progress View Music Download



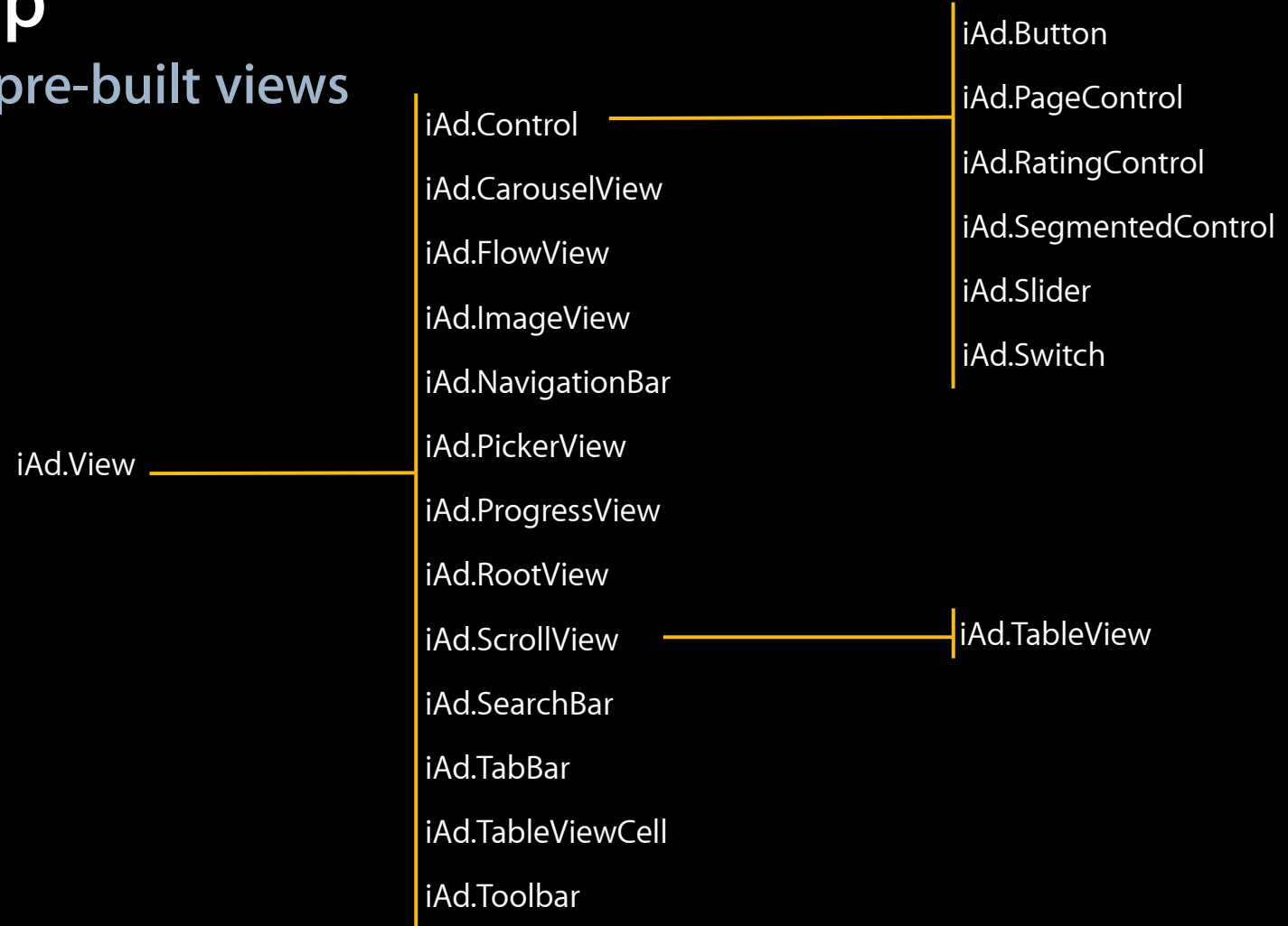
Develop iAd JS for high-level objects



```
controller.init = function () {  
    carousel = new iAd.CarouselView();  
    carousel.orientation = HORIZONTAL_ORIENTATION;  
    carousel.size = new iAd.Size(480, 100);  
    carousel.position = new iAd.Point(50, 50);  
};
```

Develop

Subclass pre-built views



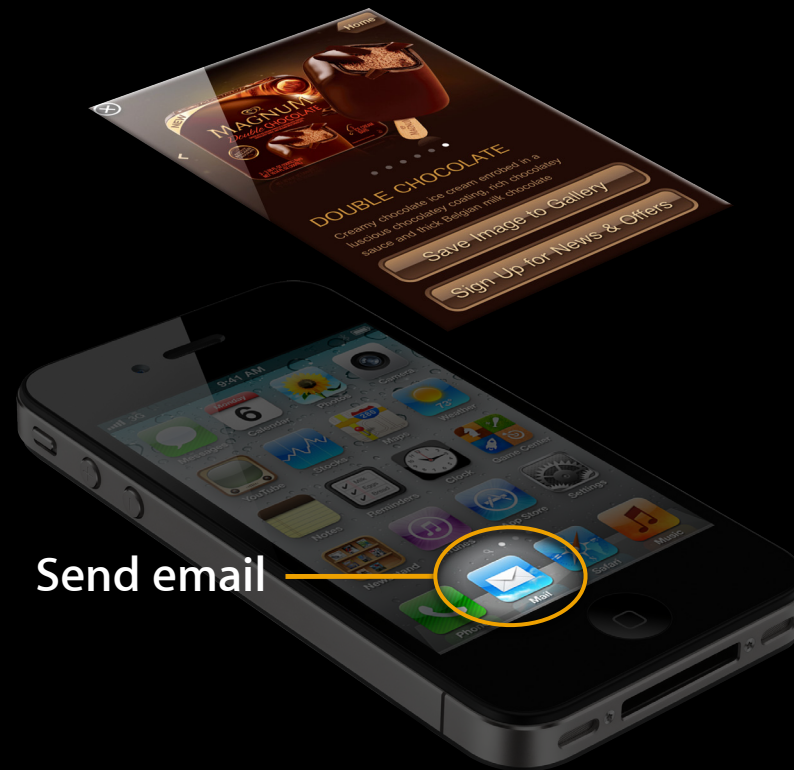
Develop

Add device integration



Develop

Add device integration



Develop

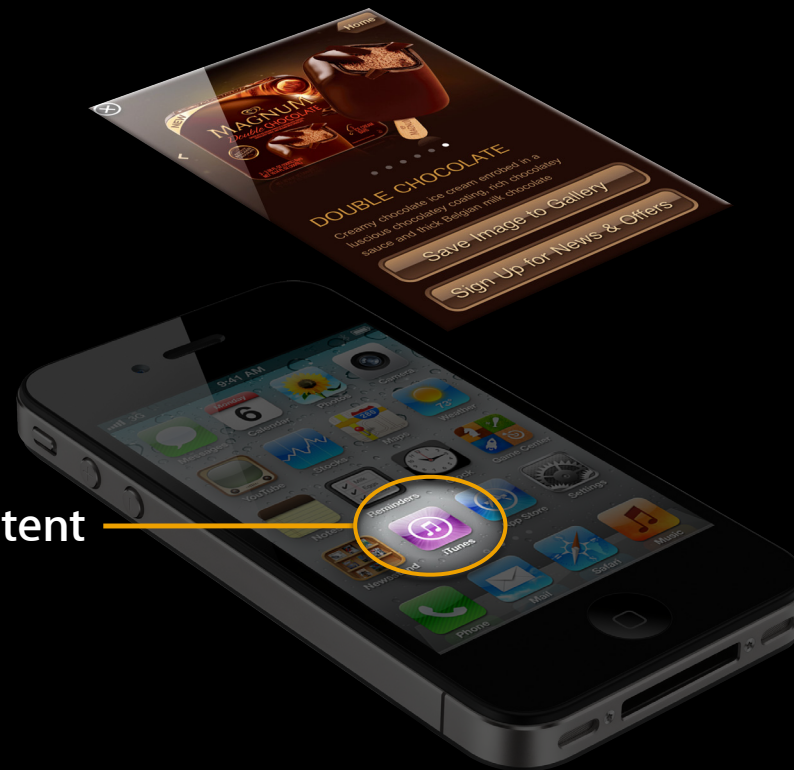
Add device integration



Download apps

Develop

Add device integration



Download iTunes content

Develop

Add device integration

Save to photos
Set wallpaper



Develop

Add device integration



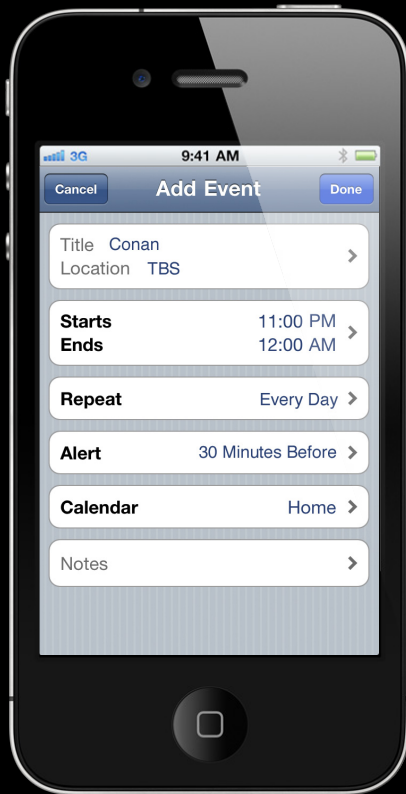
Develop

Add device integration



Develop

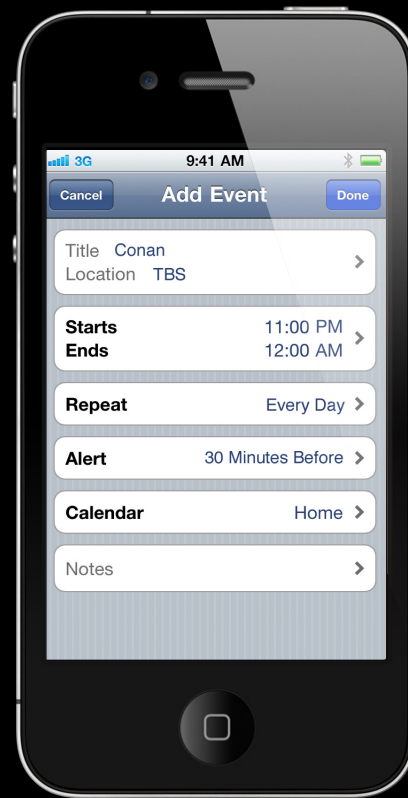
Add device integration



```
// Event details
var startDate = new Date().toISOString();
var endDate = new Date().toISOString();
var myEvent = { "description": "My Event",
                "location": "Cupertino, CA",
                "summary": "Free parking available.",
                "start": startDate,
                "end": endDate
              };
// prompt user
window.ad.calendar.presentComposer(myEvent, listener);
```

Develop

Add device integration



Develop

Web technologies



+



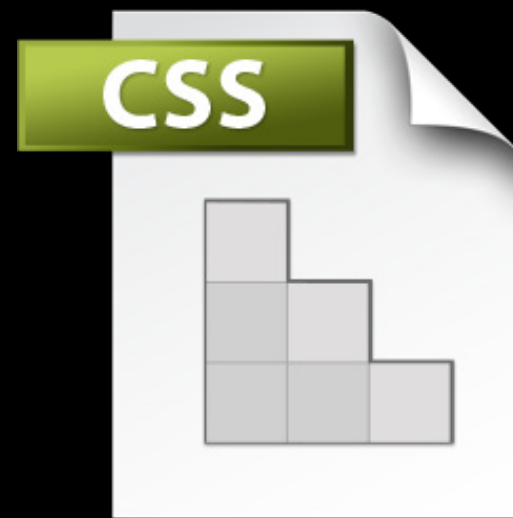
Develop

Web technologies



Develop

Import code assets



Develop

Create iAd Producer plug-ins



Develop

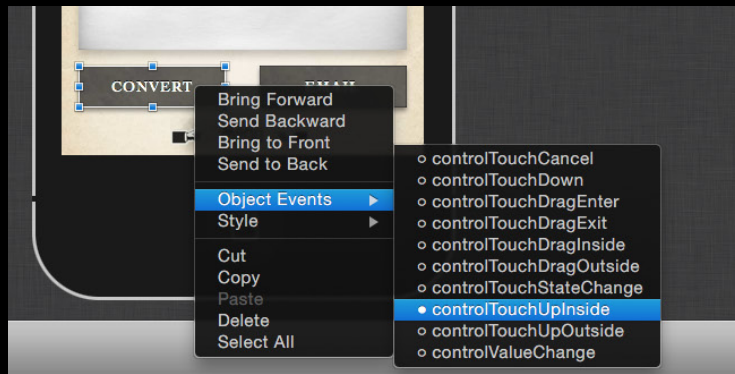
iAd Producer plug-ins

- Create drop-in/reusable widgets for designers
- Deliver custom objects and page templates
- Provide inspectors for visual editing



Develop

Hook into events



```
Hudson Telegraph - TryItSconvertButtonDelegate.js
js TryItSconvertButtonDelegate.js onControlTouchUpInside Available Events
this.onControlTouchUpInside = function(event) {
    var converter = this.viewController.textConverter;
    var textField = this.viewController.textField;
    var emailButton = this.viewController.outlets.emailButton;
    var beepAudio = this.viewController.outlets.beepAudio;
    // exit when the text area is blank:
    if (!converter.element.value.length) {
        return;
    }
    // load the current message and play the conversion audio
    // when the converter is going to convert text to morse codes:
    if (converter.isText) {
        converter.loadMessage();
        beepAudio.play();
        emailButton.enabled = true;
    }
    // run the conversion:
    converter.convertMessage();
    // toggle the disabled attribute of the text field:
    // (the text field should be disabled for editing when it is showing morse codes)
    if (textField.disabled) {
        textField.removeAttribute('disabled');
    } else {
        textField.setAttribute('disabled', 'disabled');
    }
};
```

Develop Debugger (just in case)

The screenshot shows a web browser's developer debugger interface. The main pane displays JavaScript code for a Morse code converter. The code includes a `loadMessage` function that processes text into Morse code and a `convertMessage` function that displays the result. The code is as follows:

```
75 loadMessage: function() {  
76     var message = this.element[this.textAttr];  
77     this.textMessage = message.replace(/\s+|\s+$/g, '').split('');  
78     this.morseMessage = [];  
79     for (var i = 0, len = this.textMessage.length; i < len; i++) {  
80         var code = (HT.morse.letterCodes[ this.textMessage[i].toUpperCase() ] ||  
81         ' ') + ' '; this.morseMessage.push(code);  
82     }  
83 }  
84 // convert the message between text and Morse codes and display it in the  
element:  
85 convertMessage: function() {  
86     var message = this.isText ? this.morseMessage : this.textMessage;  
87     this.isText = !this.isText;  
88     this.element[this.textAttr] = message.join('');  
89 }  
90 }  
91 }  
92 }
```

The interface also shows a Breakpoint list with a breakpoint set at line 85, a Stackframes & Variables pane, an Evaluator, and a Run Log at the bottom.



Demo

Adding custom code and style



Design → **Develop** → Deploy



Design → Develop → **Deploy**

Deployment

Sharing your iAd

- Locally via Wi-Fi



Deployment

Sharing your iAd

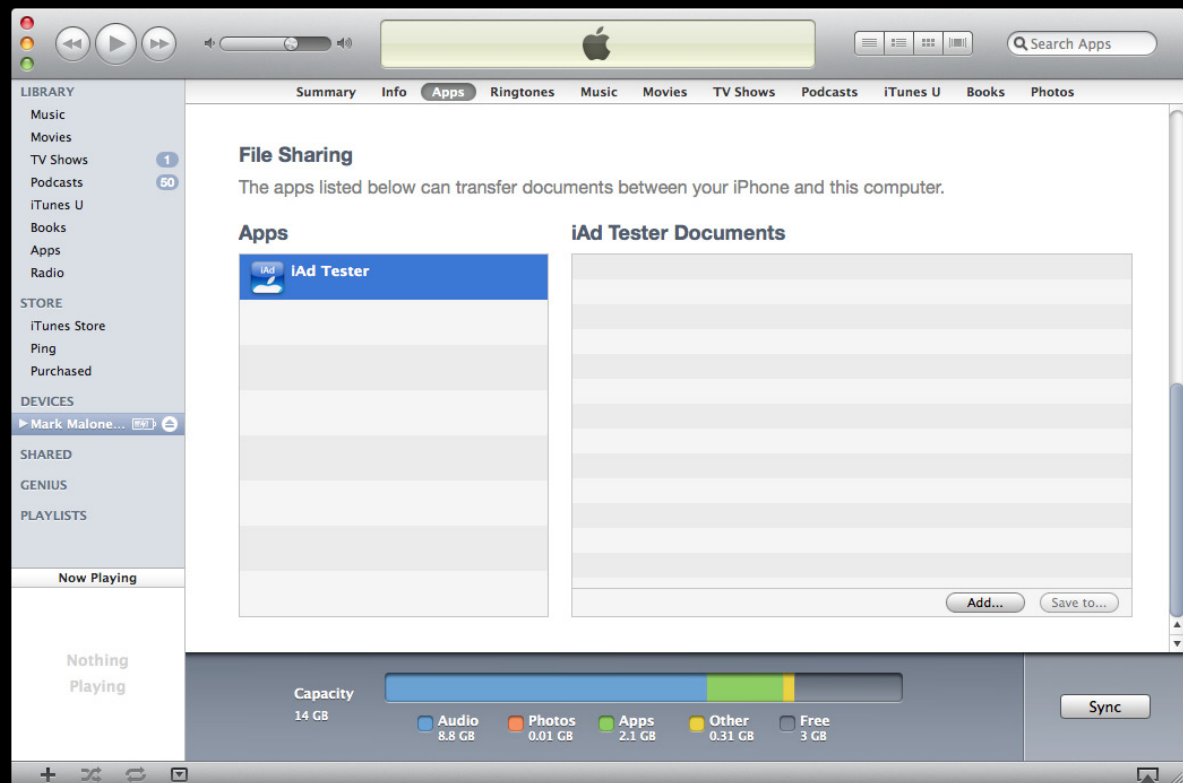
- Locally via Wi-Fi
- Installed via iTunes



Deployment

Sharing your iAd

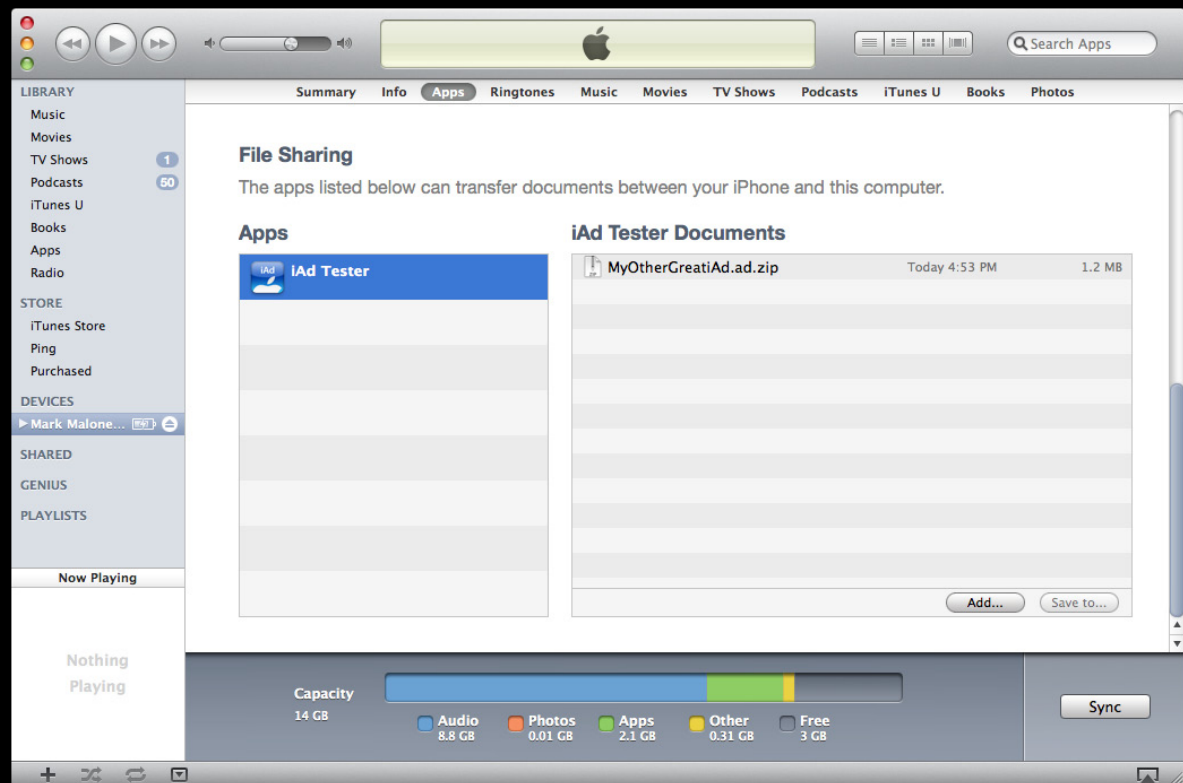
- Locally via Wi-Fi
- Installed via iTunes



Deployment

Sharing your iAd

- Locally via Wi-Fi
- Installed via iTunes



Deployment

Sharing your iAd

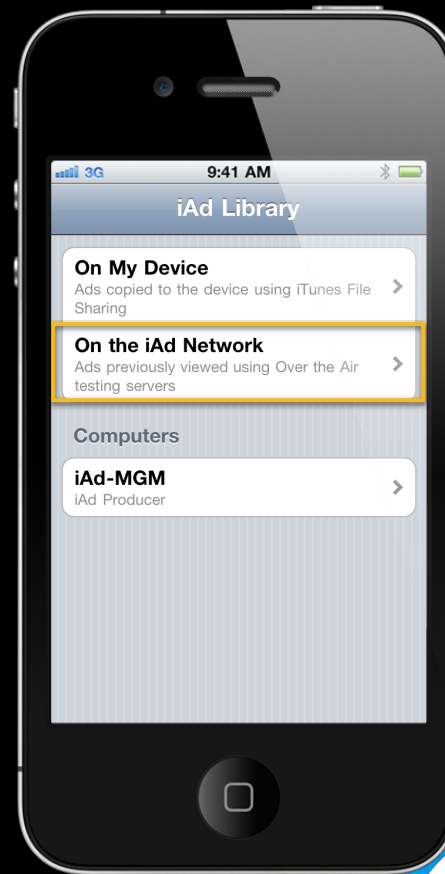
- Locally via Wi-Fi
- Installed via iTunes



Deployment

Sharing your iAd

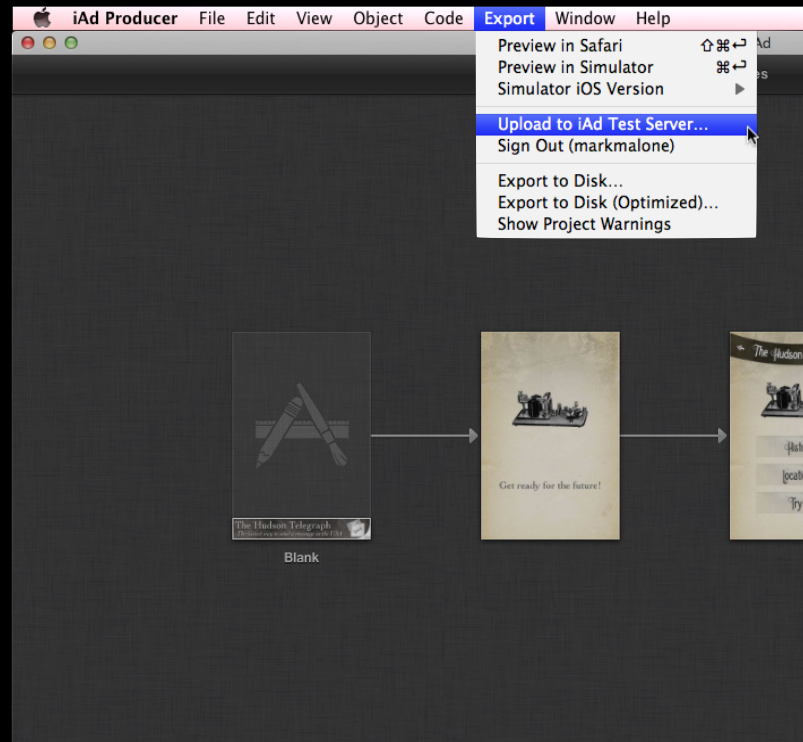
- Locally via Wi-Fi
- Installed via iTunes
- iAd test server



Deployment

Sharing your iAd

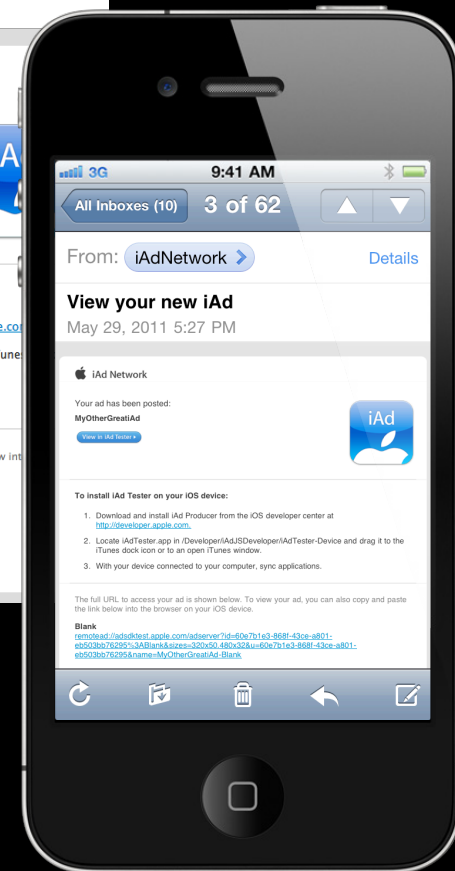
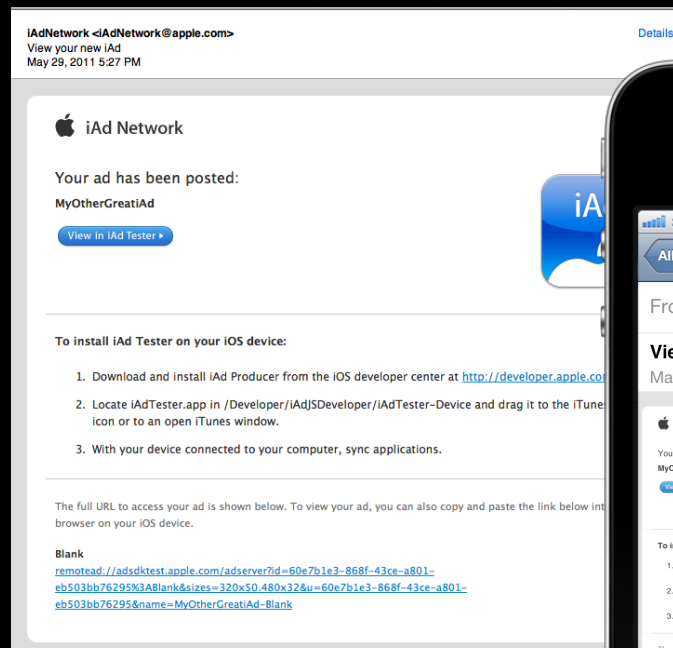
- Locally via Wi-Fi
- Installed via iTunes
- iAd test server



Deployment

Sharing your iAd

- Locally via Wi-Fi
- Installed via iTunes
- iAd test server





Design → Develop → Deploy

Wrapping Up

Summary

- Makes incredible—easy
 - Page templates and objects
 - Code free style and animation
- Unlimited creativity via customization
 - Bring your own fonts, code, and style
 - Custom templates via iAd Producer plug-ins
- Reduced deployment surprises
 - Experience on device—design through deployment
 - Mobile optimized templates



More Information

Mark Malone

iAd Professional Services
mgm@apple.com

iAd Producer and Information

<http://advertising.apple.com/>
<http://developer.apple.com/iad/iadproducer/>

Documentation

iAd Producer User Guide
iAd Producer Tutorial
iAd JS Reference Library
<http://developer.apple.com/>

Apple Developer Forums

<https://devforums.apple.com/community/ios/iadproduction/iadproducer>

Related Sessions

iAd Implementation Best Practices

Presidio
Wednesday 10:15AM

Understanding And Optimizing Web Graphics

Marina
Wednesday 3:15PM

Labs

iAd Producer Lab

Internet and Web Lab A
Wednesday 2:00PM

iAd Integration Lab

Internet and Web Lab B
Wednesday 2:00PM

