

Titanium Certified Expert (TCE) Training

ANDREW MCELROY

@Sophrinix



WELCOME TO

TITANIUM CERTIFIED EXPERT (TCE) TRAINING

- ▶ Labs @ Appcelerator Wiki (included code)
- ▶ PDF version of this presentation available by Instructor.

INTRODUCTION

TITANIUM CERTIFIED EXPERT (TCE) TRAINING

Briefly please:

- ▶ Name
- ▶ What's your background?
- ▶ What's your Titanium experience?

- ▶ Class Hours
- ▶ Parking
- ▶ Restrooms
- ▶ Meals
- ▶ Network Connectivity
- ▶ Phones
- ▶ Smoking
- ▶ Local Emergencies
- ▶ Fire Exits

COURSE OBJECTIVES

- ▶ Understand the core principles of mobile user experience design as expressed by Apple's Human Interface Guidelines and Google's Android Design guides.
- ▶ Deepen knowledge of the Titanium platform by covering advanced use cases, performance tips, UI configurations, and device APIs.
- ▶ Use platform-specific device APIs and configuration to deliver a best of breed experience on each mobile platform.

SCHEDULE



<u>UI/UX Design Principles</u>	<u>iOS Deep Dive</u>
<u>Orientation & Gestures</u>	<u>Android Deep Dive</u>
<u>Animation</u>	<u>App Interconnections</u>
<u>TableViews & ListViews</u>	<u>Performance Optimization</u>
<u>Input and Navigation</u>	<u>Using Modules</u>
<u>Custom UI Components</u>	
<u>Q&A, Hacking</u>	<u>Q&A, Exam</u>

Q&A

UI/UX DESIGN PRINCIPLES

TITANIUM CERTIFIED EXPERT (TCE) TRAINING

IN THIS LESSON, YOU WILL:

- ▶ Explore the importance of UI/UX design
- ▶ Identify the principles of mobile UX design
- ▶ Explore input/output options available in a mobile app
- ▶ Incorporate brand & personality in your app
- ▶ Identify tools for UI/UX design
- ▶ Identify best practices for testing and the development cycle



UI/UX DESIGN IS CRITICAL

BASICS OF UX

The image displays two screenshots side-by-side, each showing a user interface with a modal dialog box overlaid.

Screenshot 1 (Left): A mobile application interface for tracking travel expenses. It includes fields for Distance (Business, Charity, Medical, Other), Type (Business, Charity, Medical, Other), Destination (Apple Cupertino), Purpose (Consulting), Frequent Trips (Post Office (3.2), FedEx (4.4), Fry's Electronics (20.7), Santa Clara Valley Audubon (6.7)), and a \$ Spent button. At the bottom, there are Save Data and Clear buttons. A numeric keypad is also present. A modal dialog box is overlaid, showing a small profile picture of a man and a woman, followed by the text: "An app on my phone just asked me... 'Do you want to cancel?' My options are 'Yes' and 'Cancel'... I'm totally stumped." Below this text is the timestamp "2 hours ago".

Screenshot 2 (Right): A Windows application window titled "Generate Bitstream". It contains an informational message: "Create a programming (.bit) file for the design, use IMPACT to program the FPGA device or generate a PRCH programming file from the generated bitstream." Below this is a "Options" section with a list of checkboxes labeled: -d, -l, -b, -c, -m, -r, -v, -z, -d, -r, -d, -d. A "More Options" link is visible. At the bottom of the window is a note: "Select an option above to see description of it." At the very bottom are OK and Cancel buttons.

Can you answer the following:

- ▶ **What problem(s) am I solving?**
- ▶ **Who are my key users?**
- ▶ **What are their needs?**

- ▶ Segment Your Customers
- ▶ **Focus on your primary consumer!**
(The one that makes you profitable)
- ▶ **Remove your personal opinion!**
(You're not the target consumer)

PRINCIPLES OF UI/UX DESIGN

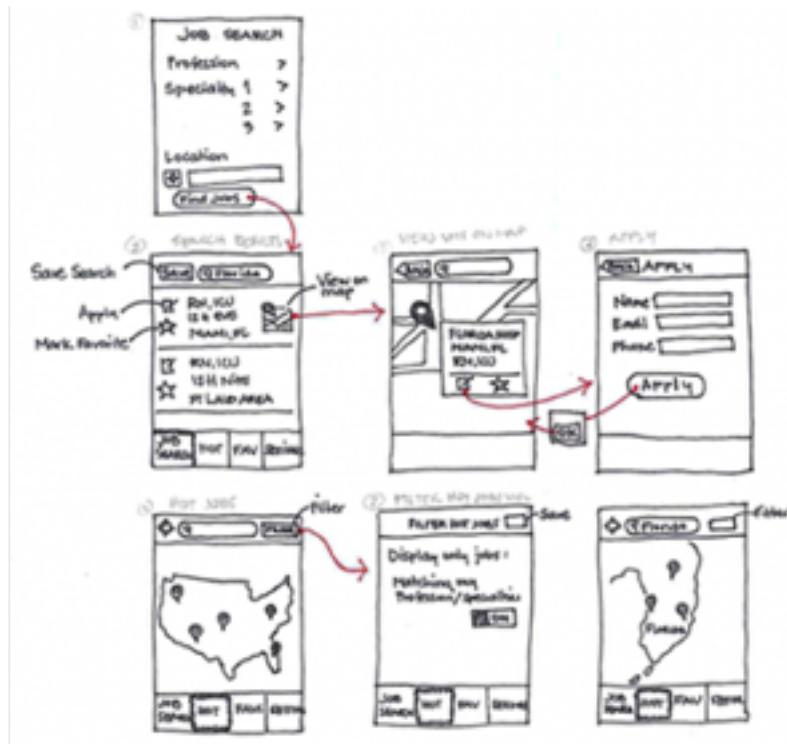
- ▶ Mobile computing is fundamentally different than desktop/laptop
- ▶ Design for real mobile interfaces
(voice, messaging, location...)
- ▶ Users take their mobile devices everywhere
- ▶ People's identities are wrapped up in their mobile devices

- ▶ Mobile hardware is confined by nature
- ▶ Freedom in the framework
- ▶ Target 'lowest common' hardware

- ▶ Size and density variations
- ▶ But that's an opportunity
- ▶ Target your primary consumers, accommodate your secondary consumers
- ▶ Don't design once, use everywhere

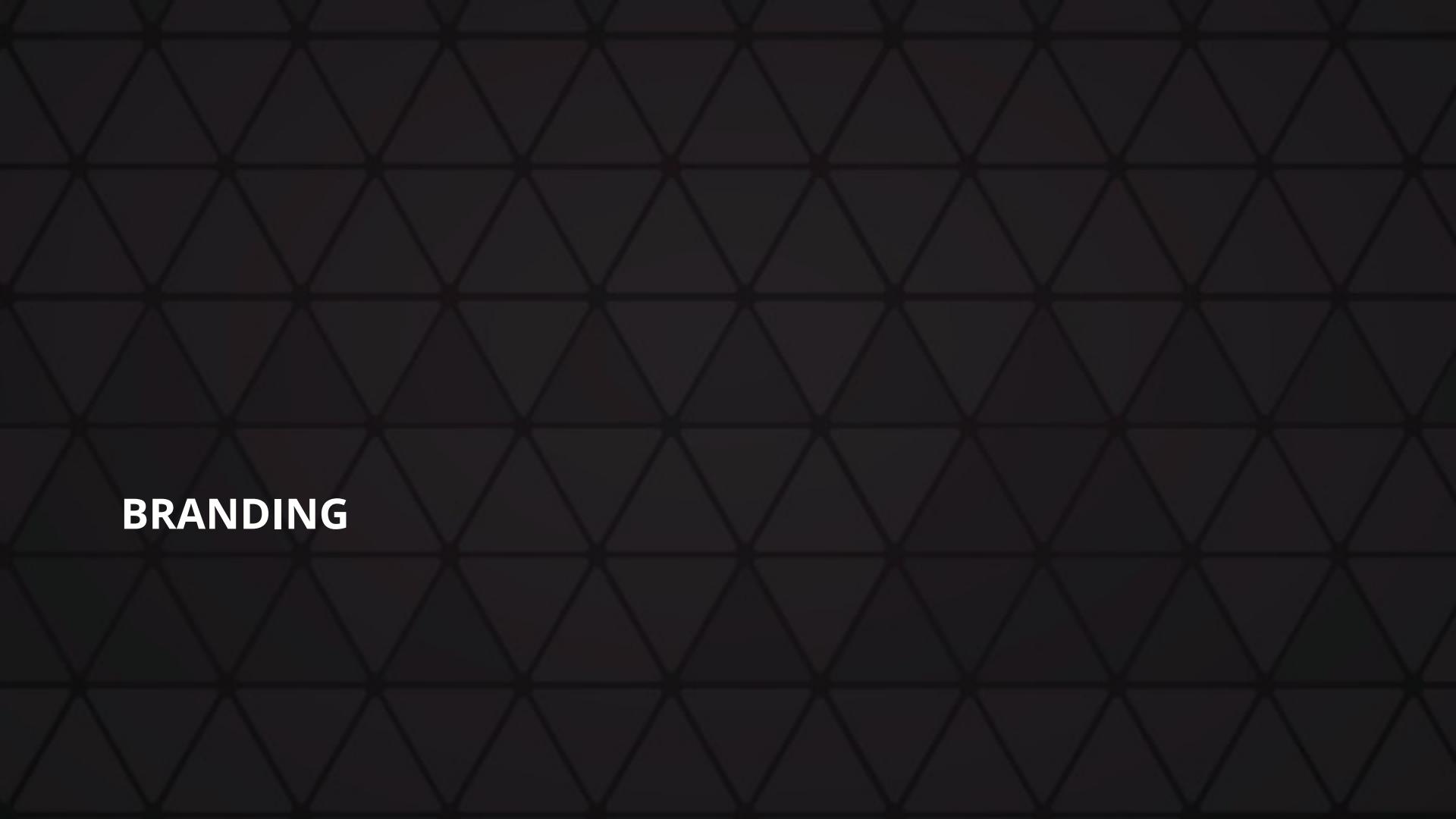
- ▶ Handset usage: One-handed, on the go
- ▶ Tablet usage: Two-handed, lean-back
- ▶ Users can, and do rotate their devices for various reasons
- ▶ Plan for abandon

SKETCH IT!



REMEMBER YOUR MVP

- ▶ Minimum Viable Product = just the required features
- ▶ More is not better
- ▶ Watch out for feature creep
- ▶ Target your primary audience

The background of the slide is a dark, almost black, color. It features a subtle, light-colored geometric pattern consisting of a grid of thin, light gray lines forming a series of small, equilateral triangles. This pattern is more prominent in the center and towards the bottom of the slide, creating a sense of depth.

BRANDING

- ▶ Understanding Your Consumer
- ▶ Develop a Single Page Brand
- ▶ Know the Emotion, Style & Location

- ▶ App Icons & Default Screens
- ▶ Colors & UX
- ▶ UI Components

- ▶ Quick & easy way to reinforce brand
- ▶ Don't overuse
- ▶ Adhere to the brand
- ▶ TTF and OTF fonts support
- ▶ Google Fonts, FontSquirrel.com, and others
- ▶ Watch licensing issues

CUSTOM FONTS

```
  ".container": {  
    backgroundColor:"white"  
  },  
  "Label": [  
    width: Ti.UI.SIZE,  
    height: Ti.UI.SIZE,  
    color: "#000",  
    font: {  
      fontFamily: 'FirecatMedium',  
      fontSize: '32dp'  
    }  
  ]
```

- ▶ Put in assets/fonts or assets/platform/fonts
- ▶ For fontFamily (default):
 - ▶ Android use file name
 - ▶ iOS use PostScript name
- ▶ Tip: rename file to PostScript name:
 1. Install font
 2. Open FontBook & find font
 3. Press CMD + i to find PostScript name

INPUTS & OUTPUTS

- ▶ Directional Information
- ▶ Use in conjunction with Compass
- ▶ Directional Scrolling
- ▶ Dictates Orientation



You must lock your orientation, or design for both!

User's rotate device for a variety of reasons, your app
must work in both orientations

Better yet, design for both!

ORIENTATION

The image shows a mobile application interface for grocery shopping and meal preparation.

Left Screen (Shopping List):

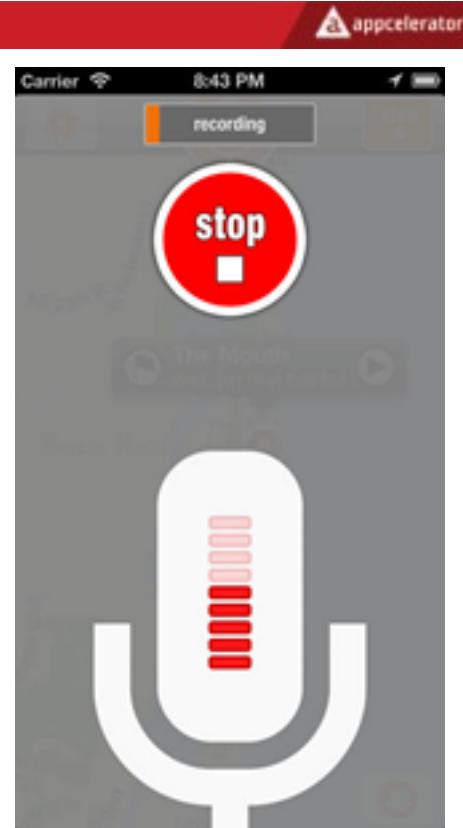
- Aisle: Recipe
- Recipe: Lemon & herb chicken with mash (For two people)
- Ingredients:
 - 2 skinless chicken breasts
preferably higher welfare
 - 400g new potatoes
 - 2 ripe tomatoes
 - 1 small fresh basil bunch
 - 4 small spring onions bunch
- Bottom navigation: Recipes, Shopping (with a red notification dot), Videos, Essentials, More

Right Screen (Recipe Card):

- Recipe: Pork and broccoli stir-fry
- Step: Re-fill and boil the kettle. Trim off and discard the tough ends of the broccoli spears then slice them in half lengthways.
- Ingredient: 100g purple sprouting broccoli
- Progress: 4 of 14

MICROPHONE & AUDIO

- ▶ Input & Output
- ▶ Guide a user with instruction
- ▶ Provide positive/negative feedback
- ▶ Unique and interesting UX





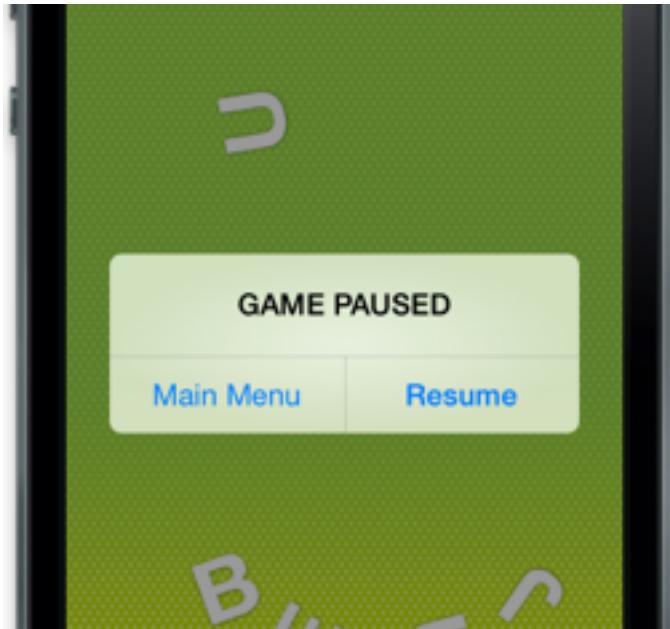
- ▶ Record & Edit
- ▶ Instant Result & Gratification
- ▶ Multiple Uses
- ▶ Augment Experience

LOADING INDICATORS



- ▶ Provide feedback to user
- ▶ Provide information
- ▶ Show progress
- ▶ Pass/Fail

FEEDBACK



- ▶ Alerts are ok
- ▶ Action screens are good
- ▶ Active results are best

- ▶ Only provide it when necessary
- ▶ Don't remind me constantly
- ▶ Are you serving the needs?
- ▶ Does the data really fit?
- ▶ Are you just showing off?

TOOLS OF THE TRADE

UX DESIGN TOOLS



- ▶ Paper & pencil !
- ▶ Balsamiq
- ▶ Fluid
- ▶ What's your favorite?

- ▶ Raster: Photoshop, Pixelmator, Gimp, etc.
 - ▶ Vector: Fireworks, Inkscape, etc.
 - ▶ Other: Keynote/PowerPoint
-
- ▶ Start with a template or follow Apple/Android human interface guidelines

LIVE VIEW TOOLS



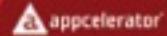
- ▶ Quick device feedback
- ▶ Check sizes and layout
- ▶ Great for app demos
- ▶ iOS: Reflector, AirServer, LiveView
- ▶ Android: Miracast, ChromeCast, Mirror



DEVELOPMENT & TESTING

- ▶ Low number of your primary audience
- ▶ LiveView apps work well
- ▶ Do not 'own' the project

USING TITANIUM FOR RAPID TESTING



- ▶ Bring your UI to life
- ▶ Bring important Design elements
(Fonts, colors, positioning)
- ▶ Place dead images
- ▶ Use test content ('Lorem Ipsum')

- ▶ Change early and often
- ▶ Ensure your changes are justified
- ▶ Pivoting 180 degrees is not bad
- ▶ Ask yourself the three questions...

- ▶ Keep testing hypotheses
- ▶ You are never complete!
- ▶ Remember your MVP

In this lesson, you:

- ▶ Explored the importance of UI/UX design
- ▶ Identified the principles of mobile UX design
- ▶ Explored input/output options available in a mobile app
- ▶ Incorporated brand & personality in your app
- ▶ Identified tools for UI/UX design
- ▶ Identified best practices for testing and the development cycle

Q&A

ORIENTATION AND GESTURES

TITANIUM CERTIFIED EXPERT (TCE) TRAINING

IN THIS LESSON, YOU WILL:

- ▶ Configure orientation support and react to orientation changes
- ▶ Respond to gestures other than 'click'

ORIENTATION

ORIENTATION SUPPORT



- ▶ Fixing orientation for the whole app
- ▶ Setting supported orientation for each window
- ▶ Reacting to orientation changes dynamically

- ▶ App-level settings either fix orientation for the entire app
- ▶ Or specify the possible orientations that the app supports
- ▶ Controls splash screen orientation (on tablets)
- ▶ Techniques differ for iOS & Android

APP-LEVEL ORIENTATION SUPPORT - IOS

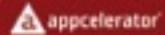


In tiapp.xml:

```
<ios>
<plist>
<dict>
    <key>UISupportedInterfaceOrientations</key>
    <array>
        <string>UIInterfaceOrientationPortrait</string>
    </array>
    <key>UISupportedInterfaceOrientations~ipad</key>
    <array>
        <string>UIInterfaceOrientationPortrait</string>
        <string>UIInterfaceOrientationPortraitUpsideDown</string>
        <string>UIInterfaceOrientationLandscapeLeft</string>
        <string>UIInterfaceOrientationLandscapeRight</string>
    </array>
</dict>
</plist>
</ios>
```

- ▶ By default, supports Portrait and Landscape
- ▶ No notion of upside-down or landscape-left/right
- ▶ Configure manifest to remove orientation-change support

LOCKING ORIENTATION ON ANDROID



1. Copy <application> node & all its <activity> tags from build/android/AndroidManifest.xml

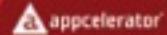
1. Paste into tiapp.xml, between <android> <manifest></manifest> </android> tags

1. Specify desired app orientation by adding screenOrientation attribute

```
<android xmlns:android="http://schemas.android.com/apk/res/android">
    <manifest>
        <application>
            <activity
                android:name="org.appcelerator.titanium.TiActivity"
                android:configChanges="keyboardHidden|orientation"
                android:screenOrientation="portrait"
            />
            <activity
                android:name="ti.modules.titanium.ui.TiTabActivity"
                android:configChanges="keyboardHidden"
            />
        </application>
    </manifest>
</android>
```

By default, Android supports all orientations

SETTING ORIENTATION PER WINDOW



Window Orientation Modes

Supported values include:

- `PORTRAIT` / `UPSIDE_PORTRAIT`
- `LANDSCAPE_LEFT` / `LANDSCAPE_RIGHT`

```
// traditional API code
var win = Ti.UI.createWindow({
    orientationModes: [Ti.UI.PORTRAIT, Ti.UI.LANDSCAPE_LEFT]
});

/* With Alloy, can define in the TSS file */
"#window": {
    orientationModes: [Ti.UI.PORTRAIT,
Ti.UI.LANDSCAPE_LEFT]
}
```

ORIENTATION EVENTS



- orientationchange event in the Ti.Gesture namespace
- This is an app-level, global event
- Event properties and methods:

```
Ti.Gesture.addEventListener('orientationchange',
function(e) {
    // current device orientation
    // Ti.Gesture.orientation

    // or, get orientation from event object
    // e.orientation

    // also, there are two helpers:
    // e.source.isPortrait()
    // e.source.isLandscape()
});
```

EXAMPLE



```
Ti.Gesture.addEventListener('orientationchange', function(e)
{
    if(e.source.isPortrait()) {
        $.someView.applyProperties({
            top: 10, left: 10
        });
    } else if(e.source.isLandscape()) {
        // explicitly test to avoid spurious orientation
        changes on iOS
    }
});
```

- ▶ Don't put *orientationchange* event listeners in your view controllers
- ▶ They are a potential memory leak
- ▶ Instead, put a single *orientationchange* event listener in index.js
- ▶ Export a function from your other controllers, which you call from index.js

BEST PRACTICE EXAMPLE



http://docs.appcelerator.com/titanium/3.0/#!/guide/Orientation-section-29004932_Orientation-Reactingtoorientationchanges

```
//index.js
Ti.Gesture.addEventListener('orientationchange',function(e) {
    Ti.App.fireEvent('orient', e.source);
});

//someview.js

Ti.App.addEventListener('orient', function(e) {
    if(e.portrait) {
        someview.left = 10;
    } else if(e.landscape) {
        someview.left = 50;
    }
});
```

GESTURES

GESTURES

- ▶ Shake
- ▶ Swipe
- ▶ Touch start, end, move, & cancel
- ▶ Pinch
- ▶ Long press

SHAKE



```
Ti.Gesture.addEventListerner('shake', function(e) {  
    // e.timestamp is only property  
    alert('it worked');  
});
```

```
<!-- how hard a shake is needed -->  
<property name="ti.android.shake.factor">1.3</property>  
<!-- how long must it continue -->  
<property name="ti.android.shake.active.milliseconds">1000</property>  
<!-- how long between shakes -->  
<property name="ti.android.shake.quiet.milliseconds">500</property>
```

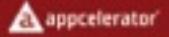
SWIPES



- ▶ Built-in event on most Ti.UI elements
- ▶ Event object properties:
 - ▶ direction
 - ▶ source
 - ▶ x/y coords

```
$.someView.addEventListener('swipe', function(e) {  
    alert('You swiped ' + e.direction);  
});
```

TOUCHES



- Built-in event on most Ti.UI elements
- Subtypes: touchstart, touchmove, touchend, touchcancel
- touchmove fires continuously during event
- Event object properties: source, x/y coords

```
$.someView.addEventListener('touchstart',
  function(e) {
    alert('You touched at coordinates ' + e.x +
      '/' + e.y);
});
```

PINCH



- ▶ iOS & Android
- ▶ Zoom only (no rotation)

```
// $.image is some large image, shown at 400w x 300h
// $.view is a view, also shown at 400w x 300h
// $.image is inside the $.view

var maxWidth = 1600,
    maxHeight = 1200,
    origWidth = 400,
    origHeight = 300;

$.view.addEventListener('pinch', function(e){
    var newWidth = $.image.width * e.scale;
    newWidth = (newWidth < origWidth) ? origWidth : newWidth;
    newWidth = (newWidth > maxWidth) ? maxWidth : newWidth;
    var newHeight = $.image.height * e.scale;
    newHeight = (newHeight < origHeight) ? origHeight : newHeight;
    newHeight = (newHeight > maxHeight) ? maxHeight : newHeight;
    $.image.applyProperties({
        width: Math.round(newWidth),
        height: Math.round(newHeight)
    });
})
```

LONG PRESS



- Supported by most UI elements
- Keep in mind native UI conventions for long presses

```
$.someView.addEventListener('longpress',
function(e) {
    alert('You pressed me');
});
```

In this lesson, you:

- ▶ Configured orientation support and react to orientation changes
- ▶ Responded to gestures other than 'click'

Q&A

LAB GOALS



- ▶ Update simple app to add orientation support
- ▶ Update data based on swipe and shake events

wiki.appcelerator.org/display/td/o2++Handling+Gestures+and+Orientation+Changes

ANIMATION

TITANIUM CERTIFIED EXPERT (TCE) TRAINING

IN THIS LESSON, YOU WILL:

- ▶ Know key concepts of animation
- ▶ Know basic properties of animations
- ▶ Know 2D and 3D animation arrays

BASIC ANIMATION

- ▶ Animations can help a user distinguish changes to an application
- ▶ You can help to make your application more attractive and dynamic (ie 'fashionable')
- ▶ Essential for games (use less often in Titanium, but increasingly popular)

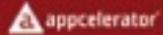
BASIC ANIMATIONS



For simple animations,
use the
obj.animate property ()

```
// $.circle is a Titanium View
$.circle.animate({
    top: 200,
    right: 30,
    duration: 500
}, function() {
    // optional callback, invoked when anim is
    complete
    $.circle.animate({
        top: 0,
        left: 0,
        duration: 500
    });
});
```

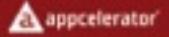
BASIC ANIMATIONS WITH OBJECTS



To reuse the animations, you can create an object and assign it an animate ():

```
// $.circle is a Titanium View
var a = Ti.UI.createAnimation({
    backgroundColor: '#ff0000',
    autoreverse: true,
    repeat: 3
});
a.duration = 1000;
$.circle.animate(a);
```

USING THE OPACITY



- Opacity is an important property in animation
- 0 = invisible; 1 = fully opaque
- Dimming objects indicates they are unavailable
- Fading in/out views makes UI state changes less jarring

```
// fade out
$.view.animate({
    opacity: 0,
    duration: 500
});
```

MATRIX TRANSFORMATIONS

MATRIX ANIMATION



What is a matrix?

For animations, is a set of properties that describe the transformation of an object in a manner that a computer can process and execute

It offers more options for animation

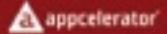
[http://en.wikipedia.org/wiki/Matrix_\(mathematics\)](http://en.wikipedia.org/wiki/Matrix_(mathematics)) if you're interested

MATRIX 2D ANIMATIONS



- ▶ Transform objects in 2D space
- ▶ Rotate (rotate)
- ▶ Scale (scalar)
- ▶ Skew (bias)
- ▶ Associated with animation object

EXAMPLE MATRIX 2D ANIMATION



```
// a 2D matrix object
var twoD = Ti.UI.create2DMatrix();
twoD = twoD.rotate(20);
twoD = twoD.scale(1.5);

// which is used in an animation
object
var a = Ti.UI.createAnimation();
a.transform = twoD;
a.duration = 3000;
a.autoreverse = true;
a.repeat = 3;

// set it all in motion
$.cloud.animate(a);
// $.cloud is an ImageView to animate
```

MATRIX 3D ANIMATION (IOS)



- ▶ Transforming objects in 3D space
- ▶ Rotate (rotate)
- ▶ Scale (scalar)
- ▶ Skew (bias)
- ▶ Associated with animation object

EXAMPLE MATRIX 3D ANIMATION



```
// a 3D matrix object -- note namespace!
var threeD = Ti.UI.iOS.create3DMatrix();
threeD = threeD.rotate(180, 1, 1, 0);
threeD = threeD.scale(2.0, 2.0, 2.0);
// modify value at a specific matrix location
threeD.m34 = 1.0/-1500;

// use the matrix in an animation object
var a = Ti.UI.createAnimation();
a.transform = threeD;
a.duration = 3000;
a.autoreverse = true;
a.repeat = 3;

// set it all in motion
$.cloud.animate(a); // $.cloud is an ImageView to animate
```

- ▶ iOS only feature
- ▶ Built-in 3D transformations
- ▶ Page curl, flip, etc.
- ▶ Apply to Window or View

- ▶ Pre-built animation effects
- ▶ require() in the animation library, then call on your View(s)
- ▶ crossFade(), fadeAndRemove(), fadeIn(), and more

See <https://github.com/appcelerator/alloy/blob/master/Alloy/builtins/animation.js>

EXAMPLE



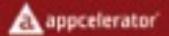
```
// in the view controller

// require in the animation built-in
var animation = require('alloy/animation');

$.loginButton.addEventListener('click', function() {
    var loginFailed = validate(); // contrived example
    if(loginFailed) {
        animation.shake($.loginButton, 30); // ← here's the animation
    } else {
        // do whatever
    }
});
```

iOS 7-SPECIFIC ANIMATIONS

IOS 7 ANIMATED TRANSITIONS



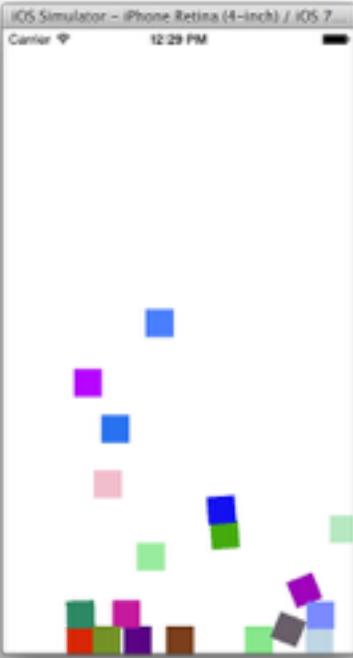
```
var transition =
Ti.UI.iOS.createTransitionAnimation({
    duration: 300,
    // The show transition makes the window
    // opaque and rotates it upright
    transitionTo: {
        opacity: 1,
        duration: 300,
        transform: Ti.UI.create2DMatrix()
    },
    // The hide transition makes the window
    transparent
    // and rotates it upside down
    transitionFrom: {
        opacity: 0,
        duration: 300 / 2,
        transform:
Ti.UI.create2DMatrix().rotate(180),
    }
});
```

```
$.openButton.addEventListener('click', function(){
    var win2 = Alloy.createController("win2", {trans:
transition});
    $.navWindow.openWindow(win2);
});

$.navWindow.open();
```

- ▶ Titanium.UI.iOS.Animator object
- ▶ Add physics-related capabilities and animations using the iOS physics engine
- ▶ Behaviors - define rules of the animator
- ▶ Items - assigned one or more behaviors
- ▶ Start with **startAnimator()**

ANIMATOR EXAMPLE



```
// Create an Animator object using the window as  
// the coordinate system  
var animator =  
Ti.UI.iOS.createAnimator({referenceView:  
$.win});  
  
// Create default collision behavior, window  
// edges as boundaries  
var collision =  
Ti.UI.iOS.createCollisionBehavior();  
  
// Simulate Earth's gravity  
var gravity = Ti.UI.iOS.createGravityBehavior({  
    gravityDirection: {x: 0.0, y: 1.0}  
});
```

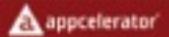
```
// Create a bunch of random blocks; add to the  
// window and behaviors  
var blocks = [];  
for (var i = 0; i < 20; i++) {  
    blocks[i] = Alloy.createController('box').getView();  
    $.win.add(blocks[i]);  
    collision.addItem(blocks[i]);  
    gravity.addItem(blocks[i]);  
}  
animator.addBehavior(collision);  
animator.addBehavior(gravity);  
  
// Start the animation when the window opens  
.win.addEventListener('open', function(e){  
    animator.startAnimator();  
});  
  
.win.open();
```

In this lesson, you:

- ▶ Learned key animation concepts
- ▶ Animated the properties of views
- ▶ Animated views with 2D and 3D matrix operations

Q&A

LAB OBJECTIVES



- ▶ Implement basic animations
- ▶ Compare the animation capabilities across platforms

wiki.appcelerator.org/display/td/340+-+Animation+API+Deep+Dive

TABLEVIEWS AND LISTVIEWS

TITANIUM CERTIFIED EXPERT (TCE) TRAINING

IN THIS LESSON, YOU WILL:

- ▶ Identify the features and uses of the list-like views
- ▶ Implement TableViews
- ▶ Implement ListViews
- ▶ Compare and contrast ListViews and TableViews

LIST-LIKE VIEWS

LIST-LIKE VIEWS

- ▶ TableView
- ▶ ListView
- ▶ ScrollView
- ▶ ScrollableView

TABLEVIEW CHARACTERISTICS

- ▶ Vertically scrolling container of data presented as rows
- ▶ Tables can contain sections, which contain rows, which can contain views
- ▶ There's an implicit section if you don't add one
- ▶ Basic row contents: title, leftImage, rightImage, indicators
- ▶ Optional: add Views as children
- ▶ Event listeners on table (preferred), section, rows, or child views

LISTVIEW CHARACTERISTICS



- ▶ Vertically scrolling container of data presented as items
- ▶ ListViews contain sections, which contain ListItems, which render ListDataItems
- ▶ ListSection is required; add ListItems to sections not the list itself
- ▶ ListItem = ListDataItem + ItemTemplate
- ▶ Event listeners on list itself or on child components within the ListItems

SCROLLVIEW CHARACTERISTICS



- ▶ Vertically or horizontally scrolling container of child views
- ▶ ScrollViews contain child Views of nearly any type
- ▶ Don't put TableViews or ListViews within a ScrollView
- ▶ Put event listeners on the child views

WHY USE ONE OVER THE OTHER?



- ▶ **TableViews** - simpler API and usage
- ▶ **TableViews** - direct manipulation of child views
- ▶ **TableViews** - lots of code examples in docs and sample apps
- ▶ **ScrollViews** - horizontal scrolling
- ▶ **ListViews** - *speed!* Faster rendering and smoother scrolling even with large data sets

TABLEVIEWS

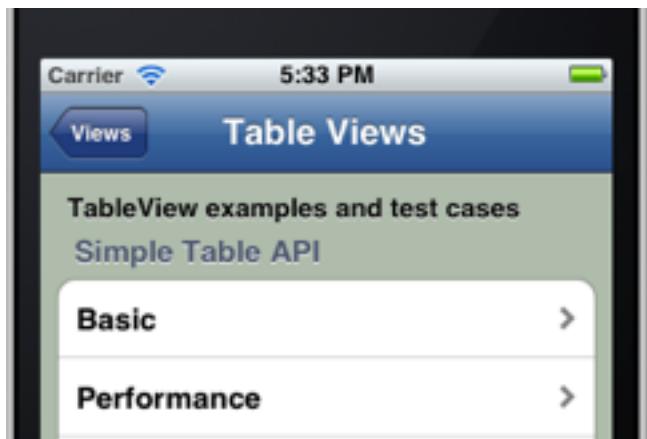
TABLEVIEW BASICS



```
<TableView id="table">
    <TableViewRow
        title="row"/>
</TableView>
```

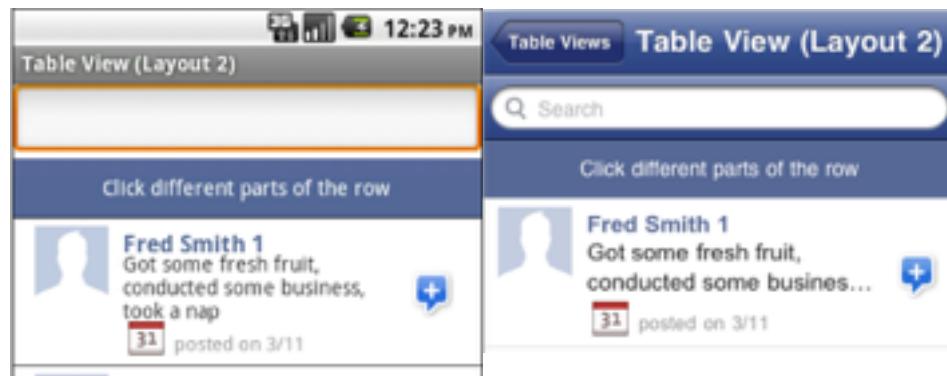
```
// eventing (use `table` rather than `$.table` for traditional example)
$.table.addEventListener('click', function(e) {
    // e.row == the row that was clicked
    // e.rowData == custom properties added to the row
});
```

HEADERS AND FOOTER VIEWS



```
$.table.header = 'TableView examples and test  
cases';  
$.table.headerView = $.simpleTableAPIView;  
$.table.footerView = $.anotherView;
```

TABLE SEARCH



```
var searchbar = Ti.UI.createSearchBar({  
    barColor:'#385292',  
    showCancel:false  
});  
  
.table.search = searchbar;
```

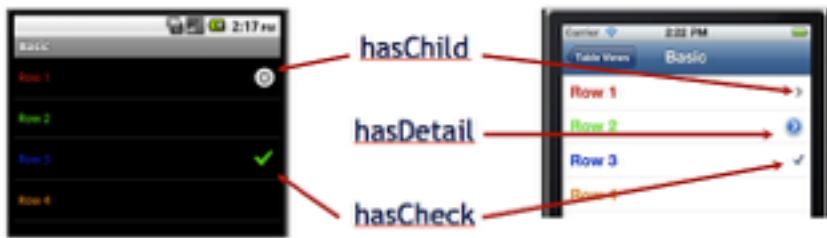
ACCESSING CHILD VIEWS



```
<TableView id="table">
  <TableViewRow title="row">
    <Label text="A label"></Label>
    <ImageView image="/someimage.png"/>
  </TableViewRow>
</TableView>
```

```
$.table.addEventListener('click', function(e)
{
  // e.row.children[1] == the ImageView
})
```

Row INDICATORS

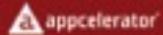


```
/* in the TSS */
"#someRow": {
  hasChild: true,
  hasDetail: true,
  hasCheck: true
}
```

- ▶ setData() or table.data faster than append()
- ▶ Load only as much data as needed
- ▶ If your table has 1,000s of custom rows, you might need to rethink your UX design
- ▶ Don't use child Views, just title & row properties

LISTVIEWS

LISTVIEW SAMPLE (ALLOY)

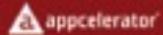


```
<ListView id="listView"
defaultItemTemplate="template">
    <Templates>
        <ItemTemplate name="template">
            <ImageView bindId="pic" id="icon" />
            <Label bindId="info" id="title" />
        </ItemTemplate>
    </Templates>

    <ListSection headerTitle="Fruit">
        <ListItem info:text="Apple" pic:image="/apple.png" />
        <ListItem info:text="Banana" pic:image="/banana.png" />
    </ListSection>
</ListView>
```

```
// in associated TSS file:
"#icon" : {
    width: '50dp', height: '50dp', left: 0
},
"#title" : {
    color: 'black',
    font: { fontFamily:'Arial', fontSize: '20dp',
fontWeight:'bold' },
    left: '60dp', top: 0
}
```

LISTVIEW SAMPLE (ALLOY)



```
<ListView id="listView"
defaultItemTemplate="template">
    <Templates>
        <ItemTemplate name="template">
            <ImageView bindId="pic" id="icon" />
            <Label bindId="info" id="title" />
        </ItemTemplate>
    </Templates>

    <ListSection headerTitle="Fruit">
        <ListItem info:text="Apple" pic:image="/apple.png" />
        <ListItem info:text="Banana" pic:image="/banana.png" />
    </ListSection>
</ListView>
```

```
// in associated TSS file:
"#icon" : {
    width: '50dp', height: '50dp', left: 0
},
"#title" : {
    color: 'black',
    font: { fontFamily:'Arial', fontSize: '20dp',
fontWeight:'bold' },
    left: '60dp', top: 0
}
```

- ▶ You must define a template for the ListView
- ▶ A ListView can have multiple templates, but one must be the defaultTemplate
- ▶ Each template can have child templates, which define what components are contained in the ListItem (row)
- ▶ There are built-in templates supported on iOS

ITEM ACCESSORIES

Ti.UI.LIST_ACCESSORY_TYPE_

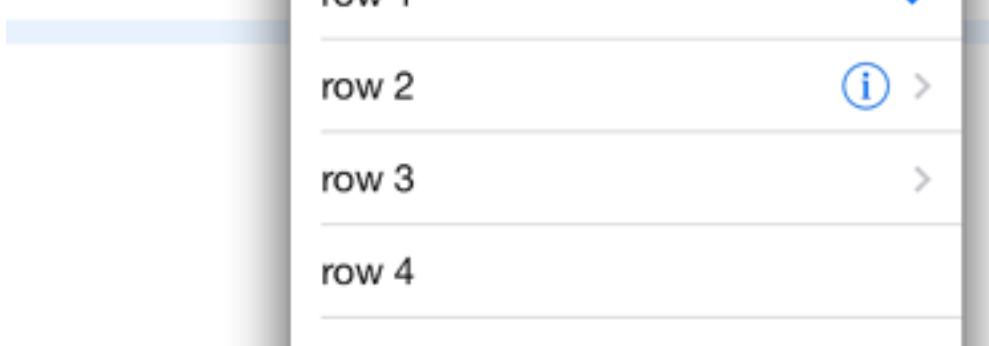
CHECKMARK

DETAIL

DISCLOSURE

NONE

```
<ListView id="list" onItemclick="onItemClick">
  <ListSection>
    <ListItem title="row 1" accessoryType="Ti.UI.LIST_ACCESSORY_TYPE_CHECKMARK"/>
    <ListItem title="row 2" accessoryType="Ti.UI.LIST_ACCESSORY_TYPE_DETAIL"/>
    <ListItem title="row 3" accessoryType="Ti.UI.LIST_ACCESSORY_TYPE_DISCLOSURE"/>
    <ListItem title="row 4" accessoryType="Ti.UI.LIST_ACCESSORY_TYPE_NONE"/>
    <ListItem title="row 5" accessoryType="Ti.UI.LIST_ACCESSORY_TYPE_CHECKMARK"/>
    <ListItem title="row 6" accessoryType="Ti.UI.LIST_ACCESSORY_TYPE_DETAIL"/>
    <ListItem title="row 7" accessoryType="Ti.UI.LIST_ACCESSORY_TYPE_DISCLOSURE"/>
    <ListItem title="row 8" accessoryType="Ti.UI.LIST_ACCESSORY_TYPE_NONE"/>
  </ListSection>
</ListView>
```



LISTVIEW EVENTING

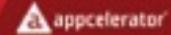


```
// use the special 'itemclick' event with ListViews
listView.addEventListener('itemclick', function(e){
    // grab a handle to the list item (row) that was clicked
    var item = e.section.getItemAt(e.itemIndex);
    // modify properties of list item:
    if (item.properties.accessoryType == Ti.UI.LIST_ACCESSORY_TYPE_NONE) {
        item.properties.accessoryType = Ti.UI.LIST_ACCESSORY_TYPE_CHECKMARK;
    } else {
        item.properties.accessoryType = Ti.UI.LIST_ACCESSORY_TYPE_NONE; }

    // once you've modified the properties, you must update the item
    e.section.updateItemAt(e.itemIndex, item);

    // use bindId (not bindID!) to determine which child component was clicked
});
```

EVENTING, PART TWO



```
var someTemplate = {
    childTemplates: [
        {
            type: 'Ti.UI.Button', // Use a button
            bindId: 'button', // Bind ID for this button
            properties: { // Sets several button properties
                width: '80dp',
                height: '30dp',
                right: '10dp',
                title: 'press me'
            },
            events: { click : report } // Binds a callback to the button's
click event
....
```

- ▶ Header and footer views
- ▶ Search views
on Android, the search box can be shown outside the ListView
- ▶ iOS: pull to refresh support, native editing support

ONE MORE THING



- ▶ Read this:
- ▶ [https://github.com/appcelerator-services/
ElementsOfListView](https://github.com/appcelerator-services/ElementsOfListView)

COMPARE & CONTRAST

TABLEVIEW VS. LISTVIEW

TableView	ListView
<i>View-oriented architecture for maximum flexibility</i>	<i>Data-oriented architecture for maximum performance</i>
<i>Sections implicit & can generally be ignored</i>	<i>Sections required</i>
<i>Bind events to table, sections, rows, or child Views</i>	<i>Bind events to ListView or child components (via template)</i>
<i>Relatively easier to code</i>	<i>Faster</i>

In this lesson, you:

- ▶ Identified the features and uses of the list-like views
- ▶ Implemented UITableViews
- ▶ Implemented ListViews
- ▶ Compared and contrasted ListViews and UITableViews

Q&A

In this lab, you will:

- ▶ Complete a two-tabbed application that implements both TableView and ListView
- ▶ Compare the features and capabilities of those components
- ▶ Manage eventing with both components
- ▶ Compare the performance of the TableView and ListView components

Lab: wiki.appcelerator.org/display/td/o4+-+TableViews+and+ListViews

SOLUTION WALK THRU

USER INPUT COLLECTION

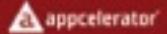
TITANIUM CERTIFIED EXPERT (TCE) TRAINING

IN THIS LESSON, YOU WILL:

- ▶ Set TextField and TextArea options
- ▶ Handle keyboard and layout issues
- ▶ Explore the available UI components

TEXTFIELD AND TEXTAREA OPTIONS

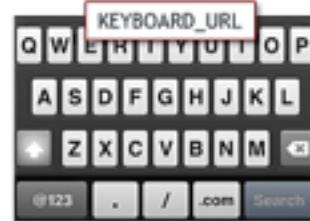
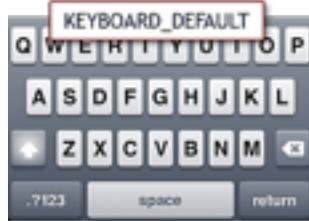
AUTOCORRECTION AND AUTOCAPITALIZATION



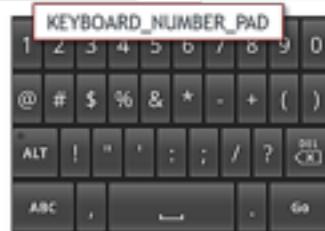
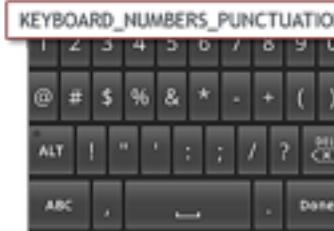
```
"#someField": {  
    autocorrection: false,  
    autocapitalization: Ti.UI.TEXT_AUTOCAPITALIZATION_WORDS  
}  
  
// others include:  
Ti.UI.TEXT_AUTOCAPITALIZATION_DEFAULT  
Ti.UI.TEXT_AUTOCAPITALIZATION_ALL  
Ti.UI.TEXT_AUTOCAPITALIZATION_SENTENCES  
Ti.UI.TEXT_AUTOCAPITALIZATION_NONE
```

- ▶ Keyboard Type — **field.keyboardType**
- ▶ Assigning the Return key — **field.returnKeyType**
- ▶ Keyboard Toolbars

KEYBOARD TYPES - iPHONE



KEYBOARD TYPES - ANDROID



KEYBOARD_DECIMAL_PAD



RETURN KEY OPTIONS

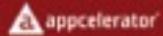


supported returnKeyTypes

- Titanium.UI.RETURNKEY_DEFAULT
- Titanium.UI.RETURNKEY_GO
- Titanium.UI.RETURNKEY_GOOGLE
- Titanium.UI.RETURNKEY_JOIN
- Titanium.UI.RETURNKEY_NEXT
- Titanium.UI.RETURNKEY_ROUTE
- Titanium.UI.RETURNKEY_SEARCH
- Titanium.UI.RETURNKEY_SEND
- Titanium.UI.RETURNKEY_YAHOO
- Titanium.UI.RETURNKEY_DONE
- Titanium.UI.RETURNKEY_EMERGENCY_CALL

KEYBOARD TOOLBARS

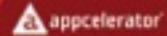
iOS only feature



```
<TextField platform="ios">
  <!-- Sets the keyboardToolbar property -->
  <KeyboardToolbar>
    <Toolbar>
      <Items>
        <FlexSpace/>
        <Button systemButton="Ti.UI.iPhone.SystemButton.CAMERA" />
        <FlexSpace/>
        <TextField />
        <FlexSpace/>
        <Button>Search</Button>
        <FlexSpace/>
      </Items>
    </Toolbar>
  </KeyboardToolbar>
</TextField>
```

SOFT KEYBOARD HANDLING

HIDING THE SOFT KEYBOARD



```
$.someField.blur(); // cross-platform method
if(OS_ANDROID) {
    // android only method
    Ti.UI.Android.hideSoftKeyboard();
}

if(OS_IOS) {
    Ti.App.addEventListener('keyboardframechanged', function(){
        Ti.API.info('Soft keyboard shown');
    });
}
```

PREVENT SOFT KEYBOARD FROM COVERING INPUT



- Put field in a scrolling view (i.e. ScrollView)
- On Android, set:

```
$.win.windowSoftInputMode = Ti.UI.Android.SOFT_INPUT_ADJUST_PAN;
```

UI COMPONENTS

DIALOG 2

```
<!-- in the View XML -->  
<AlertDialog id='alert' />  
  
/* in the TSS */di"#alert": {  
    cancel: 1, buttonNames: ['Yes',  
    'No'],  
    message: 'Delete the file?',  
    title: 'Delete'  
}
```

```
// in the controller  
$.alert.addEventListener('click',  
function(e)  
{  
    if (e.index === e .source.cancel){  
        Ti.API.info('Cancel was  
clicked');  
    }  
});  
$.alert.show();
```

UI: SWITCH

- Presents two mutually exclusive choices
- On iOS, user can turn on indicators: | (on) and o (off)

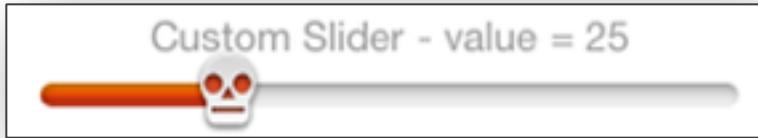


```
<!-- in the View XML -->
<Switch id='theSwitch'/>

/* in the TSS */
#theSwitch[platform=android]: {
    style: Ti.UI.Android.SWITCH_STYLE_CHECKBOX
    /* or Ti.UI.Android.SWITCH_STYLE_TOGGLEBUTTON */
}

// in the controller
$.theSwitch.addEventListener('change', function(e){
    if(e.value === true) {
        // switch is on ...
    }
});
```

UI: SLIDER



```
<!-- in the View XML -->
<Label id='customLabel'/>
<Slider id='theSlider'/>

/* in the TSS */
'#theSlider': {
    min: 0,
    max: 100,
    value: 3,
    thumbImage: 'images/skull.png'
}
```

PICKER

	red
Bananas	green
Strawberries	blue
Mangos	orange
Grapes	

	red
Bananas	green
Strawberries	blue
Mangos	orange

```
<Alloy>
  <Window backgroundColor="white">
    <Picker id="picker" selectionIndicator="true" useSpinner="true">
      <PickerColumn id="column1">
        <PickerRow title="Bananas"/>
        <PickerRow title="Strawberries"/>
        <PickerRow title="Mangos"/>
        <PickerRow title="Grapes"/>
      </PickerColumn>
      <!-- Picker shorthand notation -->
      <Column id="column2">
        <Row title="red"/>
        <Row title="green"/>
        <Row title="blue"/>
        <Row title="orange"/>
      </Column>
    </Picker>
  </Window>
</Alloy>
```

PICKER EVENTS



- Listen on 'change'
- e.selectedValue
- e.row
- e.value

```
$.picker.addEventListener('change',function(e){  
    Ti.API.info((e.selectedValue).join(', '));  
    Ti.API.info(e.row.title);  
});
```

DATE PICKER



```
<Alloy>
```

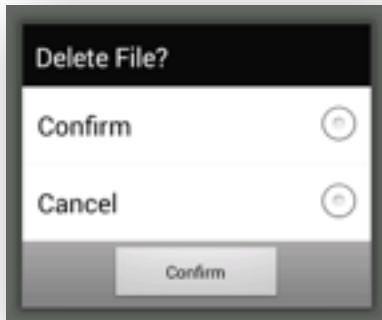
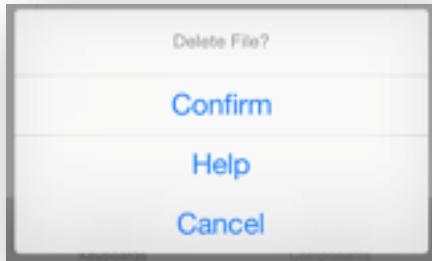
```
  <Window backgroundColor="white">
    <Picker id="picker" selectionIndicator="true"
    type="Ti.UI.PICKER_TYPE_DATE">
      </Picker>
    </Window>
  </Alloy>
```

```
// in controller
```

```
$picker.value = new Date();
```

```
$.picker.addEventListener('change',function(e){
  Ti.API.info(e.value.toLocalizedString())
});
```

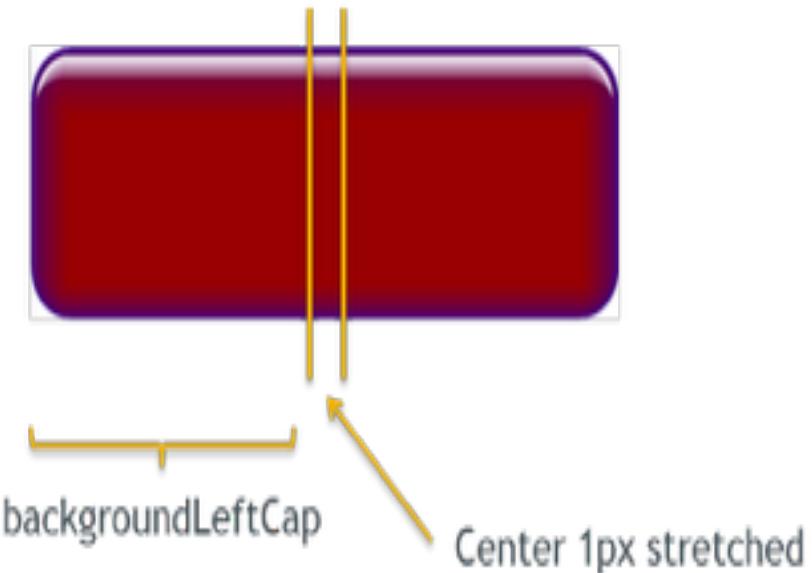
OPTION DIALOG



```
<Alloy>
<Window backgroundColor="white" layout="vertical" exitOnClose="true">
  <OptionDialog id="dialog" title="Delete File?">
    <Options>
      <Option>Confirm</Option>
      <Option platform="ios">Help</Option>
      <Option>Cancel</Option>
    </Options>
    <!-- The ButtonNames tag sets the Android-only buttonNames property. -->
    <ButtonNames>
      <ButtonName>Confirm</ButtonName>
    </ButtonNames>
  </OptionDialog>
</Window>
</Alloy>

// in controller
$.dialog.show();
```

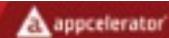
STYLING INPUT COMPONENTS



- ▶ Background colors, graphics, gradients
- ▶ Border width, color, radius
- ▶ iOS: backgroundLeftCap, backgroundTopCap

SHADOWS

Add shadows to Labels (cross-platform) and Buttons (Android only)



A screenshot of the Appcelerator Studio interface. On the left, the code editor shows an index.css file with CSS rules for a container and a label. The label rule includes a shadow color (#cccc), a shadow offset (x: 2, y: 2), and a shadow radius (2). On the right, there are two side-by-side device simulators. Both show a white rectangular box containing the text "I'm a shady character". The simulator on the left is an iPhone Retina (4-inch) running iOS 7, and the one on the right is an Android device running version 4.4. Both screens show a black header bar with the "LiveView" logo.

```
Studio (extended) = LiveView/app/styles/index.css = Appcelerator Studio = /Users/tipoulsen/Dot...
```

```
index.css 24 index.xml
1 ".container" {
2     backgroundColor:"white"
3 },
4 "#label": {
5     text: "I'm a shady character",
6     width: Ti.UI.SIZE,
7     height: Ti.UI.SIZE,
8     color: "#000",
9     shadowColor: "#cccc",
10    shadowOffset: [
11        x: 2, y: 2
12    ]
13 }
14 "#label[platform=android]": {
15     shadowOffset: [
16         x: 5, y: 5
17     ],
18     shadowRadius: 2
19 }
```

I'm a shady character

I'm a shady character

In this lesson, you:

- ▶ Set TextField and TextArea options
- ▶ Handled keyboard and layout issues
- ▶ Explored the available UI components

Q&A

You will set various keyboard options and conveniences to ease data entry

- ▶ Set a different keyboard type for a set of input fields
- ▶ Prevent the soft keyboard from covering inputs
- ▶ Hide the keyboard when user taps the window

wiki.appcelerator.org/display/td/04+-+User+Input+Collection

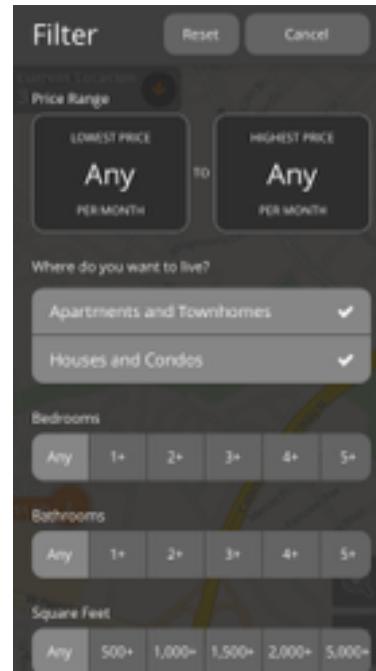
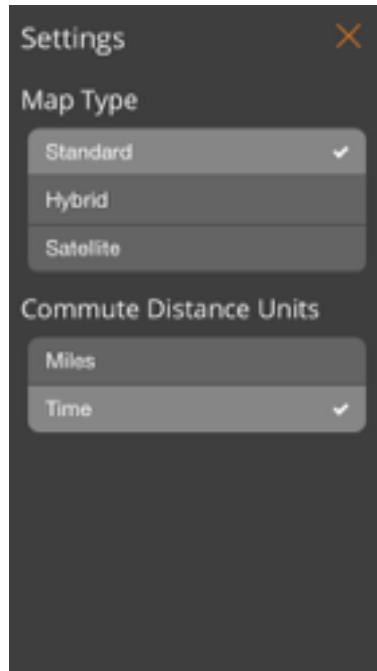
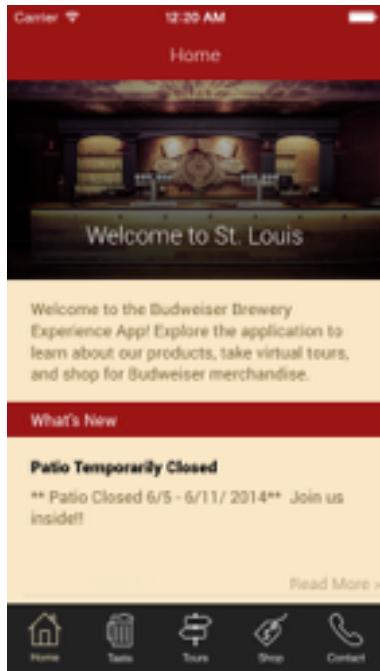
CUSTOM UI COMPONENTS

TITANIUM CERTIFIED EXPERT (TCE) TRAINING

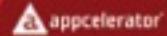
IN THIS LESSON, YOU WILL:

- ▶ Identify the building blocks of custom UI components
- ▶ Implement a custom component built with traditional API techniques
- ▶ Implement an Alloy widget

EXAMPLES

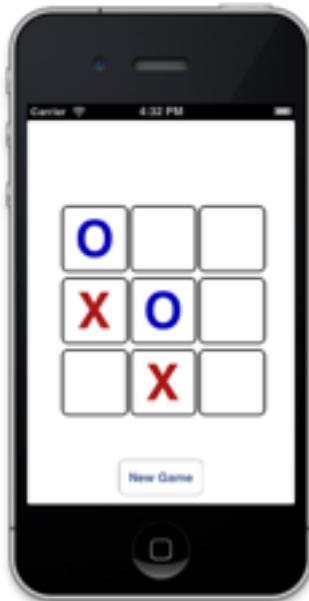


WHY CUSTOM COMPONENTS?



- ▶ Create unified UI/UX across platforms
- ▶ Apply brand styling to your UI components
- ▶ Simplify reusability
- ▶ Invent your own UX/UI component
- ▶ Cons: More work, more maintenance

CREATING CUSTOM COMPONENTS

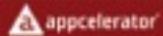


- ▶ Building blocks: Views, ImageViews, Buttons, Labels
- ▶ Which? The one that's closest to your needs
- ▶ Set styles, add eventListeners
- ▶ Generalize them as much as possible and DESIGN THE API
- ▶ Navigation components (e.g. custom tab controllers) are extra work

TRADITIONAL API-BASED COMPONENTS

- ▶ CommonJS module
- ▶ Use in either traditional or Alloy project
- ▶ Many community-contributed components
- ▶ **require()** in and then use like built-in components

CREATING A TRADITIONAL COMPONENT



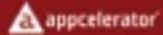
- Use the 'factory' pattern with your CommonJS module
- Don't extend the Titanium proxies
- Instantiate with new

```
var foo = 'common to all instances';
function MyComponent() {
    var bar = 'redefined w/each instance';
    var self = {};
    self.view = Ti.UI.createView();
    self.doSomething = function(args) {
        // some method of your custom component
    }
    return self;
}
module.exports = MyComponent;
```

ALLOY WIDGETS

- ▶ Self-contained components
- ▶ Can be easily dropped into an Alloy project
- ▶ Write your own, or use the ones published by
Appcelerator [github.com/AppceleratorSolutions/
AlloyWidgets](https://github.com/AppceleratorSolutions/AlloyWidgets)
- ▶ 100's of community built widgets
<http://gitt.io>

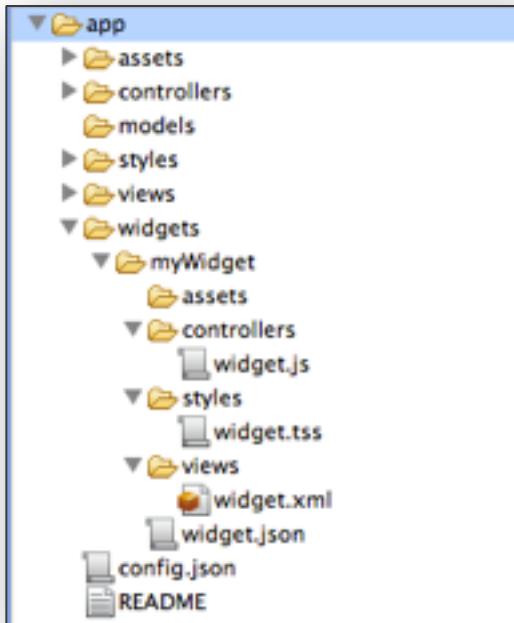
USING WIDGETS



1. Configure dependency in app/config.json
 2. Download code to app/widgets folder*
 3. Add tag to appropriate view
 4. Call widget's init() method in controller
- * Except those distributed with Alloy

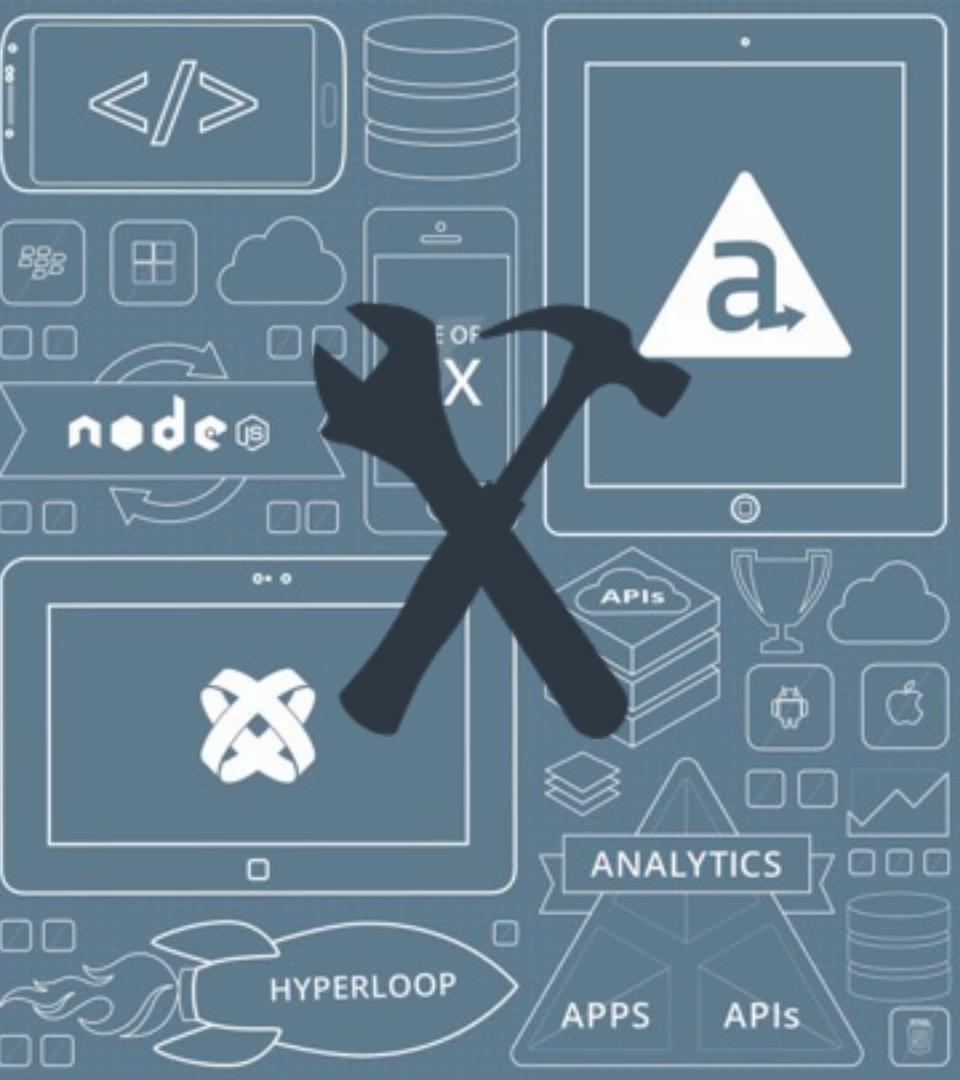
```
// config.json
{
  "global": {},
  "env:development": {},
  "env:test": {},
  "env:production": {},
  "os:ios": {},
  "os:android": {},
  "dependencies": {
    "com.appcelerator.butongrid": "1.0"
  }
}
// index.xml
<Widget id="butongrid" src="com.appcelerator.butongrid"/>
```

WIDGET CREATION PROCESS

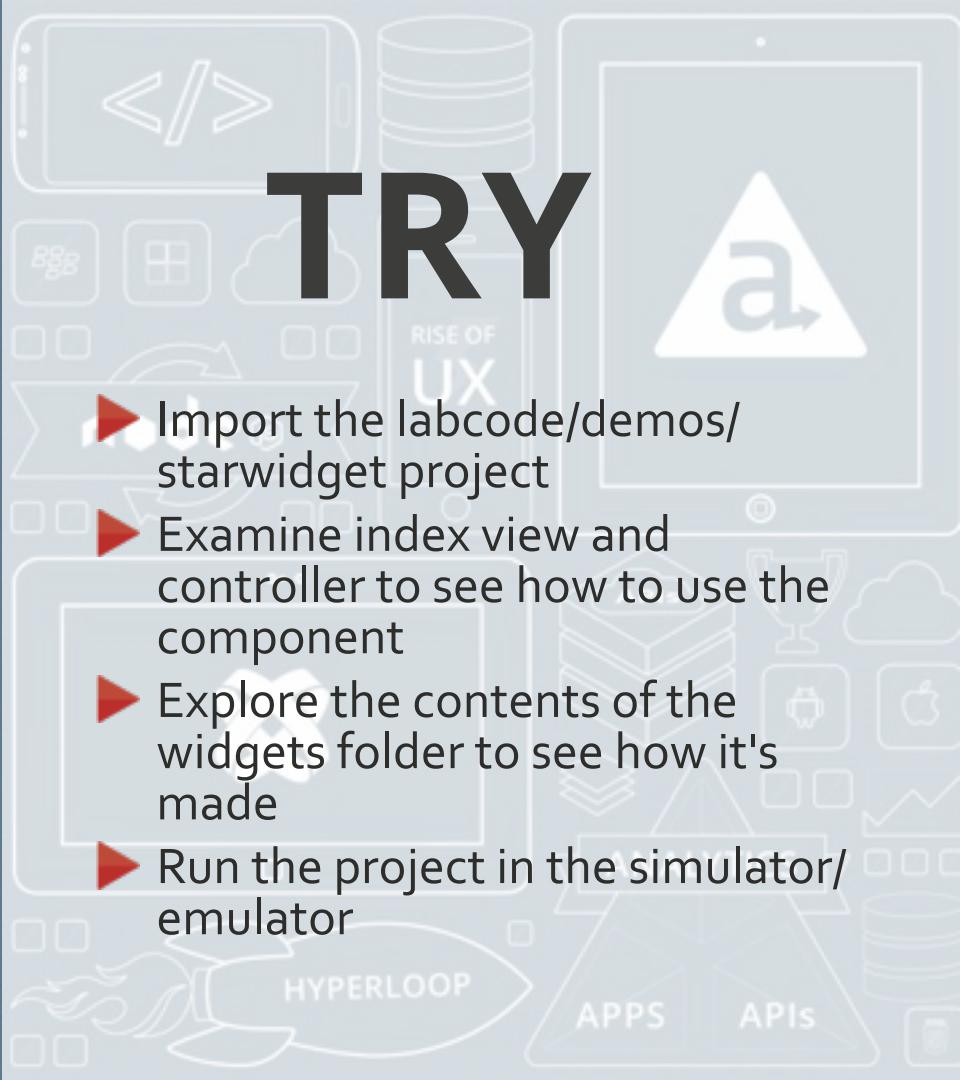


1. Create a standard mobile project
2. Right-click **app**, choose **New Widget**
3. Edit `widget.json` as needed
4. Implement your functionality in the widget's `views`, `styles`, and `controllers`
5. Test by building your mobile project
6. Distribute by sharing your widget's folder

- ▶ Main controller is `widget.js/widget.xml` rather than `index.js/index.xml`
- ▶ Instantiate "sub-views" with **Alloy.createWidget('myWidget', 'myButton')**
- ▶ Controller methods are private unless you prefix with **exports**
- ▶ You must use **WPATH('images/foo.png')** to reference assets



TRY

- ▶ Import the labcode/demos/starwidget project
 - ▶ Examine index view and controller to see how to use the component
 - ▶ Explore the contents of the widgets folder to see how it's made
 - ▶ Run the project in the simulator/emulator
- 

In this lesson, you:

- ▶ Identified the building blocks of custom UI components
- ▶ Implemented a custom component built with traditional API techniques
- ▶ Implemented an Alloy widget

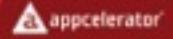
Q&A

In this lab, you will:

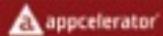
- ▶ Build a simple Alloy widget
- ▶ You will implement basic UI functionality (no backend / networking functionality)
- ▶ Implement that widget in a project

Lab: wiki.appcelerator.org/display/td/05+-+Creating+a+Widget

SOLUTION WALKTHROUGH



AGENDA – Day 2

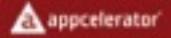


- ▶ iOS Deep Dive
- ▶ Android Deep Dive
- ▶ App Interconnections
- ▶ Performance Optimization
- ▶ Using Modules in your App

IOS DEEP DIVE

TITANIUM CERTIFIED EXPERT (TCE) TRAINING

IN THIS LESSON, YOU WILL:



- ▶ Explore iOS Platform Characteristics
- ▶ Explore Key UI and non-UI APIs

PLATFORM CHARACTERISTICS

PLATFORM CHARACTERISTICS

-  Display Is Paramount, (Regardless of Its Size)
-  Device Orientation Can Change
-  Apps Respond to Gestures, Not Clicks
-  People Interact with One App at a Time
-  Preferences Are Available in Settings
-  An App Has a Single Window
-  No hardware buttons (menu, back, keyboard)

iOS 6 vs iOS 7

iO7



iOS6



iO6



iOS7



Source: <http://osxdaily.com/2013/06/11/ios-7-vs-ios-6-visual-comparison/>

- ▶ Reduced bevels, shadows, rounding
- ▶ Reduced textures, bitmaps —; no more woodgrain or leather
- ▶ Depth (and context) provided by layering, subtle shadows, parallax and 3D effects
- ▶ Brighter & lighter design —; brighter colors, thinner fonts, line-art iconography

WINDOWS "EXTEND" TO FILL THE SCREEN



The screenshot shows an iPhone Retina (4-inch) simulator running iOS 7.0. The status bar at the top displays "Carrier" and the time "5:12 PM". Below the status bar is a red header bar with the text "Testing extendEdges". The main content area has a light blue background with a white grid pattern. A single red rectangular box is positioned in the center of the grid.

```
index.xml index.js index.ts
```

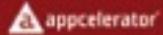
```
1
2  "#knowwin": {
3
4  }
5  "#win": {
6    backgroundColor: "transparent",
7    backgroundImage: "/gridlines.png",
8    backgroundRepeat: true,
9    title: "Testing extendEdges",
10   translucent: true, /* default is true */
11   extendEdges:[Ti.UI.EXEND_EDGE_TOP, Ti.UI.EXEND_EDGE_BOTTOM],
12   /*
13    Supported Ti.UI.EXEND_EDGE_NONE (default), Ti.UI.EXEND_EDGE_TOP, Ti.UI.EXEND_EDGE_LEFT,
14    Ti.UI.EXEND_EDGE_BOTTOM, Ti.UI.EXEND_EDGE_RIGHT or Ti.UI.EXEND_EDGE_ALL
15   */
16   top: 20
17 }
18 "#box":{
19   backgroundColor: 'red',
20   height: 30, width: 100,
21   top: 30
22 }
```

Applies "key" color to elements, such as button text foreground color in a navigation bar

- ▶ TabGroup.tabsTintColor - sets the key color for the active tab in the tab group
- ▶ Window.navTintColor - sets the key color for the window's navigation bar
- ▶ View.tintColor - sets the key color for the UI element.
- ▶ Toolbar.tintColor - sets the color of button text on the toolbar

KEY UI APIs

NAVIGATIONWINDOW



- Replaces the NavigationGroup (removed and unsupported on iOS 7)
- Is a top-level container (don't put it in a window)
- Windows within a NavigationWindow have a NavigationBar (title bar)
- Add/remove new windows with openWindow() and closeWindow()

```
<!-- Titanium.UI.iOS.NavigationWindow
-->
<Alloy>
    <NavigationWindow id="navwin"
platform="ios">
    <Window id="win">
        <View id="box"/>
    </Window>
    </NavigationWindow>
</Alloy>
```

```
// in associated controller, add a
second window
var newWin =
Alloy.createController('newWin');
$.navwin.openWindow(newWin.getView());
```

NAVIGATIONBAR



- Present in tab groups and NavigationWindows
- Modal windows have them, but without nav buttons
- String, image, or View in center
- Left/Right nav buttons for in-app navigation

```
<Alloy>
    <NavigationWindow>
        <Window>
            <!-- proxy properties for the window -->
            <LeftNavButton>
                <Button>Base UI</Button>
            </LeftNavButton>
            <TitleControl>
                <ImageView image="/images/skull.png"/>
            </TitleControl> <!-- rest of the Window markup --
        >
        </Window>
    </NavigationWindow>
</Alloy>
```

TOOLBARS



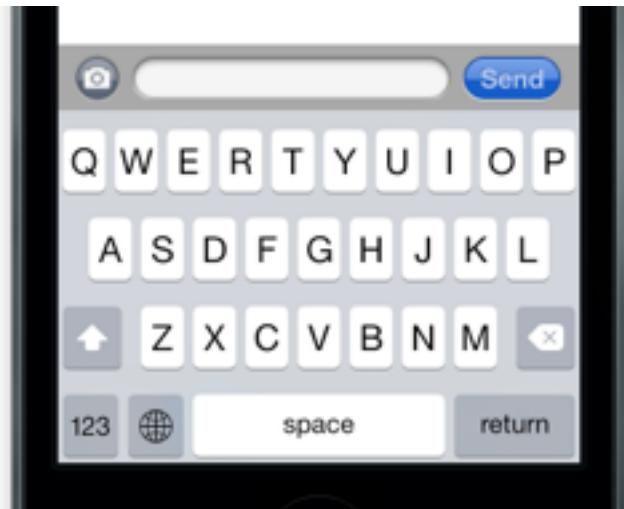
- ▶ Buttons perform actions related to current context
- ▶ Create button objects first, store in an array
- ▶ Pass the array to the Toolbar object
- ▶ Aim for 44×44 point hit area for buttons

TOOLBARS



```
<Toolbar platform="ios">
  <Items>
    <Button id="camera"/>
    <FlexSpace/>
    <TextField id="tf"/>
    <FlexSpace/>
    <Button id="send"/>
  </Items>
</Toolbar>
```

KEYBOARD TOOLBARS

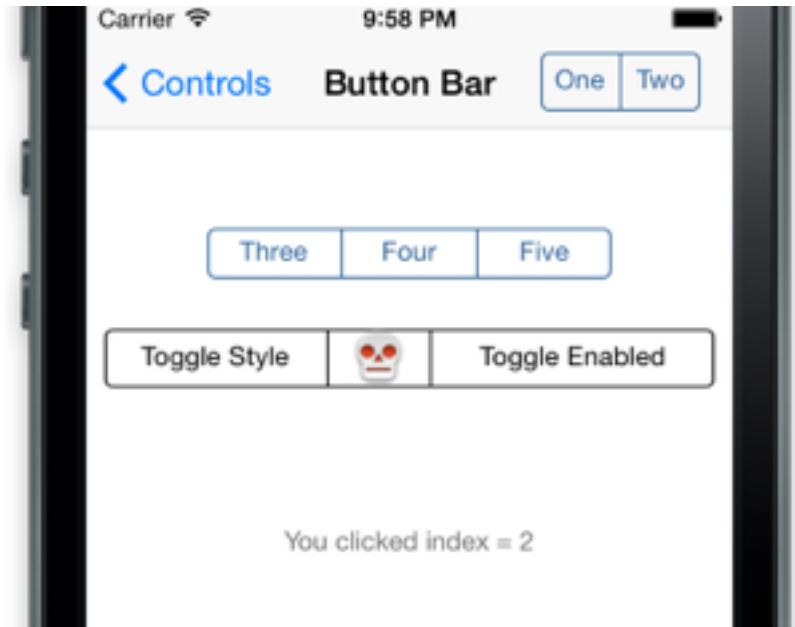
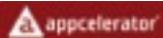


```
// not supported through Alloy markup yet
// create buttons to go in the tool bar
var camera = Ti.UI.createButton();

$.textfield.keyboardToolbar = [camera, flexSpace, tf, flexSpace, send];
$.textfield.keyboardToolbarColor = '#999';
```

BUTTON BAR

Related controls that don't maintain state



```
<ButtonBar platform="ios" onClick="doSomething">
```

```
  <Labels>
```

```
    <!-- Specify text with node text or "title" attribute. -->
```

```
    <Label>One</Label>
```

```
    <Label title="Two"/>
```

```
    <Label title="Three"/>
```

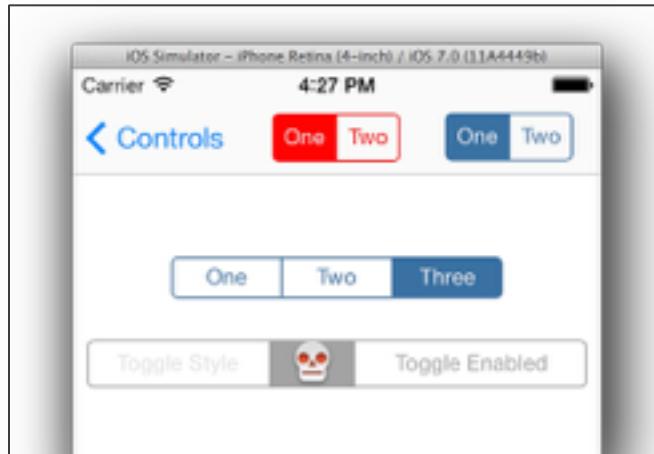
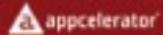
```
    <Label title="Four"/>
```

```
    <Label title="Five"/>
```

```
  </Labels>
```

TABBED BAR

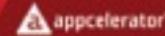
Like radio buttons, maintain state



```
<TabbedBar platform="ios" onClick="doSomething">
  <Labels>
    <!-- Specify text with node text or "title" attribute. -->
    <Label>One</Label>
    <Label title="Two"/>
    <Label title="Three"/>
  </Labels>
</TabbedBar>
// in the controller
function doSomething(e){
  switch(e.index) {
    case 0:
      // do stuff
      break;
  }
});
```

ATTRIBUTED STRINGS

Apply fonts and styles to ranges of characters in a Label, TextArea, or TextField

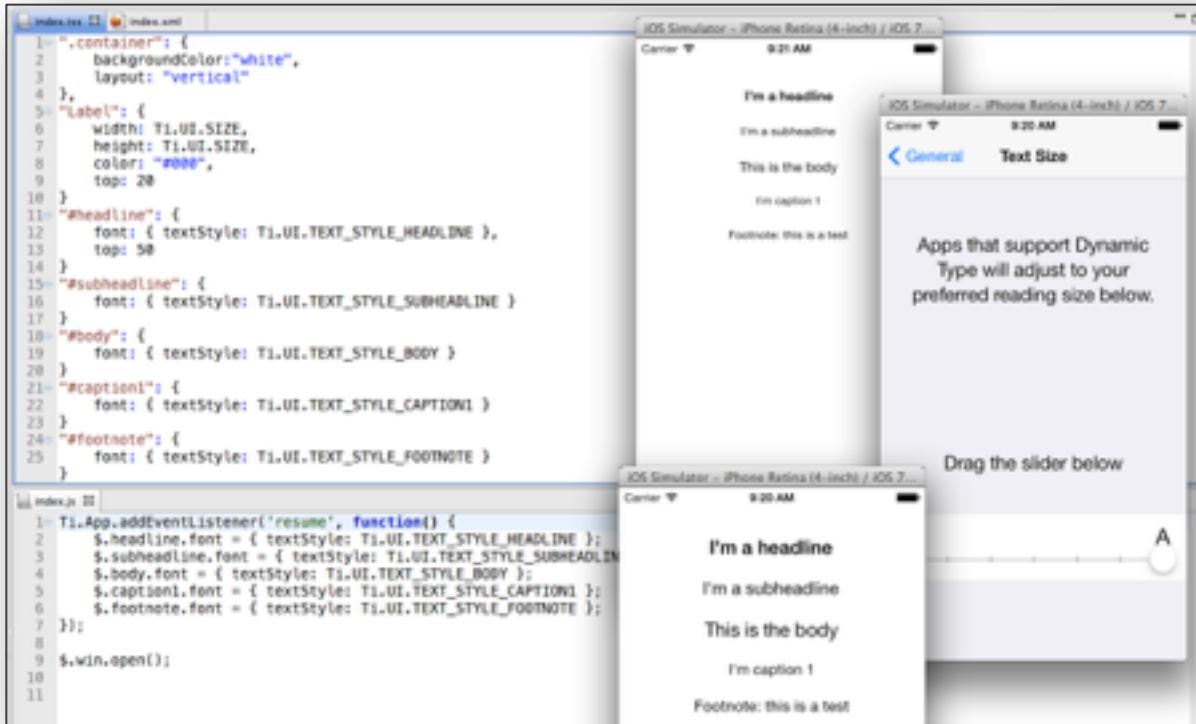


The screenshot shows the Appcelerator Studio interface. On the left, there's a code editor window titled 'index.js' with the following code:

```
1 var text = "Have you tried hyperloop yet?";
2 var attr = Ti.UI.iOS.createAttributedString({
3   text: text,
4   attributes: [
5     {
6       type: Ti.UI.iOS.ATTRIBUTE_BACKGROUND_COLOR,
7       value: "yellow",
8       range: [text.indexOf('Have'), ('Have').length]
9     },
10    {
11      type: Ti.UI.iOS.ATTRIBUTE_FONT,
12      value: {fontSize: 24, fontFamily: 'Chalkduster' },
13      range: [text.indexOf('hyperloop'), ("hyperloop").length]
14    }
15   /*
16    Other attributes include:
17    Ti.UI.iOS.ATTRIBUTE_FOREGROUND_COLOR
18    Titanium.UI.iOS.ATTRIBUTE_LIGATURE
19    Ti.UI.iOS.ATTRIBUTE_KERN
20    Ti.UI.iOS.ATTRIBUTE_STRIKETHROUGH_STYLE
21    Ti.UI.iOS.ATTRIBUTE_UNDERLINES_STYLE
22    Ti.UI.iOS.ATTRIBUTE_STROKE_COLOR
23    Ti.UI.iOS.ATTRIBUTE_SHADOW
24    Ti.UI.iOS.ATTRIBUTE_TEXT_EFFECT
25    Ti.UI.iOS.ATTRIBUTE_WRITING_DIRECTION
26    Ti.UI.iOS.ATTRIBUTE_LINK
27    Ti.UI.iOS.ATTRIBUTE_BASELINE_OFFSET
28    Ti.UI.iOS.ATTRIBUTE_OBLIQUENESS
29    Ti.UI.iOS.ATTRIBUTE_EXPANSION
30   */
31 }
32 });
33 $label.attributedString = attr;
34 $index.open();
```

On the right, there's an 'iOS Simulator - iPhone Retina (4-inch) / iOS 7...' window showing the text "Have you tried **hyperLoop** yet?" where the word "hyperLoop" is displayed in a larger, bold, black font.

DYNAMIC TEXT (TEXTSTYLE)



The image displays a developer's workspace with two code editors and two iOS simulator windows.

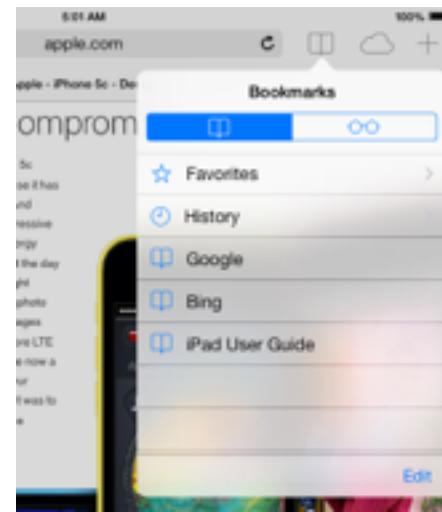
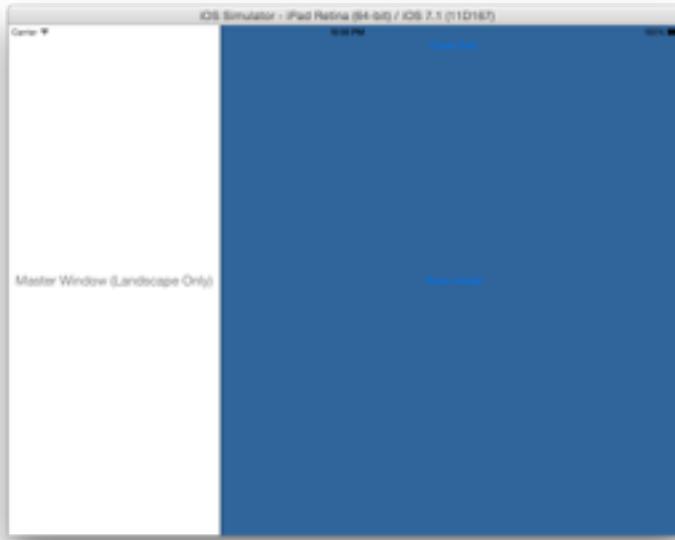
Code Editors:

- index.ios.js**: Contains CSS-like styles for various text components. It includes rules for a container, labels, and five different text styles: headline, subheadline, body, caption, and footnote, each with specific font sizes and colors.
- index.js**: Contains JavaScript code that adds an event listener for the 'resume' event. Inside the event handler, it sets the font for each of the five text styles defined in the CSS to Ti.UI.TEXT_STYLE_HEADLINE, Ti.UI.TEXT_STYLE_SUBHEADLINE, Ti.UI.TEXT_STYLE_BODY, Ti.UI.TEXT_STYLE_CAPTION1, and Ti.UI.TEXT_STYLE_FOOTNOTE respectively. Finally, it opens a window.

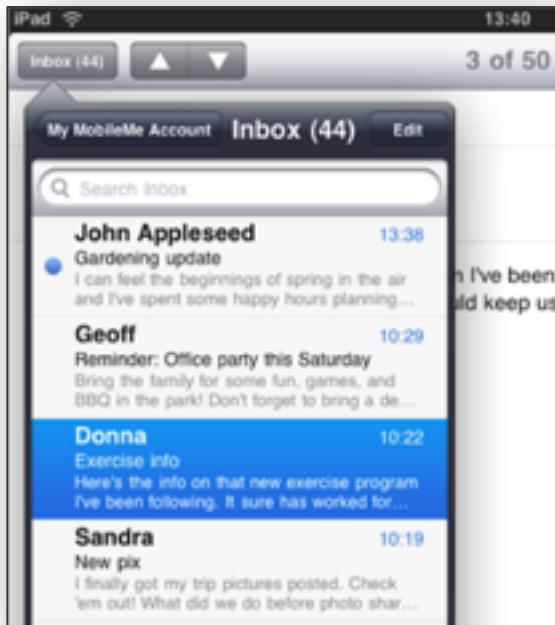
iOS Simulators:

- Carrier** 9:21 AM: Shows the application's interface with five text elements: "I'm a headline", "I'm a subheadline", "This is the body", "I'm caption 1", and "Footnote: this is a test".
- Carrier** 9:20 AM: A settings screen titled "Text Size" with the sub-section "General". It contains the text "Apps that support Dynamic Type will adjust to your preferred reading size below." Below this text is a slider with a central point labeled "A". The slider has arrows on both ends and a track with tick marks.
- Carrier** 9:20 AM: Another instance of the application's interface, identical to the first one, showing the same five text elements.

Ti.UI.iPad.Popover Ti.UI.iPad.SplitWindow



KEY APIs - TITANIUM.UI.IPAD.POPOVER

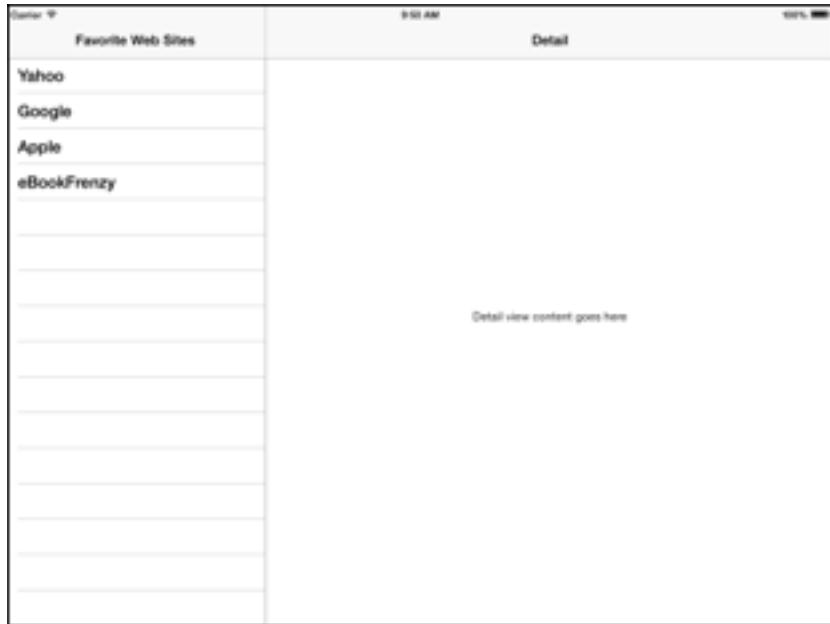


- ▶ Transient view revealed by tap on control
- ▶ Hovers over app content

```
<Popover title="Inbox">
  <ContentView>
    <NavigationWindow>
      <LeftNavBar id="mobileMe"/>
      <RightNavBar id="edit"/>
      <Window>
        <TableView id="tbl"/>
      </Window>
    </NavigationWindow>
  </ContentView>
</Popover>
```

```
// view property controls where the
// arrow points
var pop = Alloy.createController('popover').getView();
pop.show({view:$button});
```

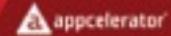
KEY APIs - TITANIUM.UI.IPAD.SPLITWINDOW



- ▶ Full-screen, two-pane view
- ▶ Left pane fixed at 320 points
- ▶ User cannot resize panes

KEY APIs - TITANIUM.UI.IPAD.SPLITWINDOW

SplitWindow is a top-level container; it doesn't go inside a window.

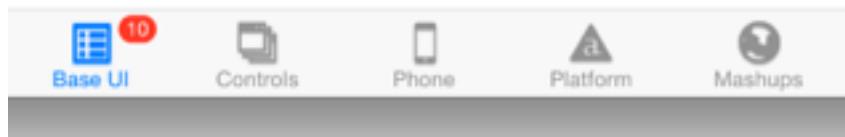
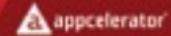


```
<Alloy>
    <SplitWindow platform="ios"
formFactor="tablet">
    <!-- The 'masterView' window -->
    <Window id="master">
        <Label>This is the master</Label>
    </Window>

    <!-- The 'detailView' window -->
    <Window id="detail">
        <Label>This is the detail</Label>
    </Window>
</SplitWindow>
</Alloy>
```

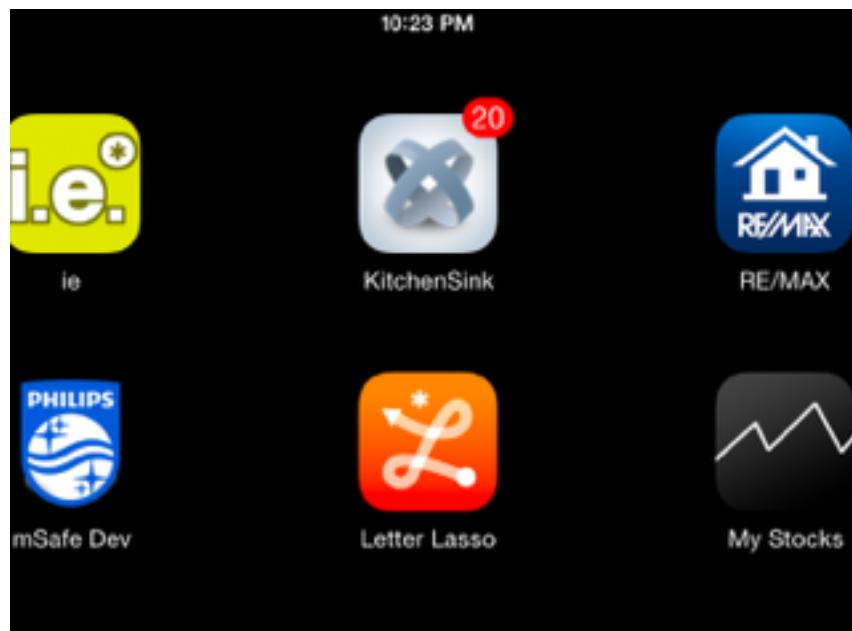
TAB BADGE

Communicates status to the user, their attention is needed in part of your app



```
var tab = Ti.UI.currentTab;  
tab.badge = 10; // set the badge  
tab.badge = null; // to remove it
```

APP BADGE



```
Ti.UI.iPhone.appBadge = 20;  
Ti.UI.iPhone.appBadge = null;
```

COVERFLOW VIEW



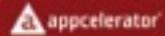
```
<Alloy>
  <Window id="window">
    <CoverFlowView platform="ios" backgroundColor="#000">
      <!-- The Images tag sets the CoverFlowView.images property. -->
      <Images>
        <!-- Assign the image by node text or the image attribute. -->
        <!-- Can also specify the width and height attributes. →
        <Image>a.png</Image>
        <Image>b.png</Image>
        <Image>c.png</Image>
      </Images>
      <!-- Place additional views for the CoverFlowView here. →
    </CoverFlowView>
  </Window>
</Alloy>
```

DASHBOARD VIEW



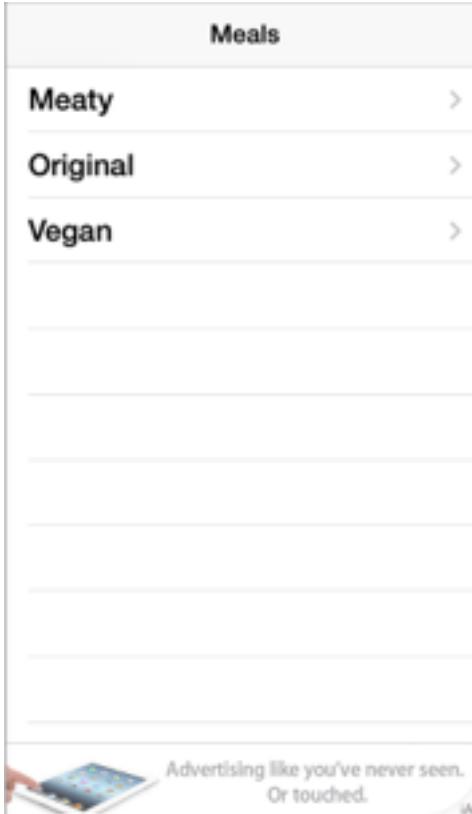
- ▶ Springboard-like view
- ▶ Use as app home screen
- ▶ User can rearrange or remove icons (if you enable it)
- ▶ Scrolling container

DASHBOARD VIEW



```
<Window>
  <RightButton>
    <Button id="editButton" onClick="toggleEditMode">Edit</Button>
  </RightButton>
</Toolbar>
<DashboardView id="dash" onEdit="handleEdit">
  <DashboardItem image="acct_off.png" selectedImage="acct_on.png" label="account"/>
  <DashboardItem image="calls_off.png" selectedImage="calls_on.png" label="calls"/>
  <DashboardItem image="cases_off.png" selectedImage="cases_on.png" label="cases"/>
  <DashboardItem image="tasks_off.png" selectedImage="tasks_on.png" label="tasks"/>
</DashboardView>
</Window>

// in the controller
function toggleEditMode(e) {
  isEditable ? $.dash.stopEditing() : $.dash.startEditing();
}
```



Meals

Meaty >

Original >

Vegan >

Advertising like you've never seen.
Or touched.

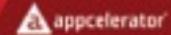
Ad

- ▶ Include ads in your app
- ▶ Banner or full screen
- ▶ Must join Apple's iAd program
- ▶ Get test ads only during development

```
var iad = Ti.UI.iOS.createAdView({
    width: Ti.UI.FILL,
    height: Ti.UI.SIZE,
    adSize: Ti.UI.iOS.AD_SIZE_PORTRAIT,
    bottom: -100
});
iad.addEventListener('load', function(){
    iad.animate({
        bottom: 0,
        duration: 500
    });
});
iad.addEventListener('action', function(){
    mygame.pause(); // do something when ad is clicked
});
$.win.add(iad);
```

FUNCTIONALITY APIs

BACKGROUND SERVICES



```
<iOS>
<plist>
<dict>
  <key>UIBackgroundModes</key>
  <array>
    <string>audio</string>
    <string>location</string>
    <string>voip</string>
    <string>newsstand-content</string>
    <string>external-accessory</string>
    <string>bluetooth-central</string>
  </array>
</dict>
</plist>
</iOS>
```

```
var svc =
Ti.App.iOS.registerBackgroundService({
  url: 'bg.js'
});
```

- Extend runtime for limited duration
- Long-running geolocation or music playback

LOCAL NOTIFICATIONS



- Runs on device, not push
- Scheduled action
- Background service alert the user

```
// schedule the notification
Titanium.App.iOS.scheduleLocalNotification({
    alertBody:"View Alarm",
    alertAction:"You set an alarm",
    userInfo:{data: 'Data to pass'},
    date:new Date(new Date().getTime() + 3000)
});
// listen for the notification
Ti.App.iOS.addEventListenert('notification',
    function(e) {
        Ti.API.info('Local notification received: ' +
e.data);
});
```

- ▶ Register with Apple
- ▶ Create APNS certificate
- ▶ Need fully-qualified app name, not wildcard
- ▶ Need server-side to generate notification (e.g. ACS, custom)

PUSH NOTIFICATIONS - CODE



```
Ti.Network.registerForPushNotifications({
    types: [
        Ti.Network.NOTIFICATION_TYPE_BADGE,
        Ti.Network.NOTIFICATION_TYPE_ALERT
    ],
    success:function(e) {
        var deviceToken = e.deviceToken;
        Ti.API.info("Push notification device token is: "+deviceToken);
    },
    error:function(e) {
        Ti.API.info("Error during registration: "+e.error);
    },
    callback:function(e) {
        // called when a push notification is received.
        var data = JSON.parse(e.data);
        if(data.badge > 0) {
            Ti.UI.iPhone.appBadge = data.badge;
        }
        if(data.message) {
            alert(data.message);
        }
    }
});
```

- ▶ New APIs to support iOS 7 "coalesced" (background) network activity
- ▶ Download large files using **UrlSession**
- ▶ Perform actions when the app is awoken to perform background downloads by listening for the **backgroundfetch** event
- ▶ Requires an add-on (though included) module, com.appcelerator.urlSession
- ▶ See the [iOS Background Services](#) guide for all the details

IMPLEMENTING APPLICATION PREFERENCES



1. Create a Settings Bundle
2. Access settings as properties with `Ti.App.Properties`
3. Do clean-build of your project

CREATING THE SETTINGS BUNDLE



1. Copy KitchenSink/platform/iphone to your project
2. In Finder, right-click Settings.bundle and choose **Show Package Contents**
3. Open Root.plist
4. Edit as necessary, then save

1. Copy KitchenSink/platform/iphone to your project
2. In Finder, right-click Settings.bundle and choose **Show Package Contents**
3. Open Root.plist
4. Edit as necessary, then save

<https://developer.apple.com/library/mac/documentation/Cocoa/Conceptual/UserDefaults/Preferences/Preferences.html>

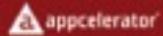
HANDLING IOS 6 AND 7 IN THE SAME APP

VERSION DETECTION



```
if(OS_IOS && parseInt(Ti.Platform.version, 10) >= 7))  
{  
    // iOS 7 code  
} else {  
    // everything else  
}
```

HANDLING WINDOW TOP



```
// in alloy.js
Alloy.Globals.winTop = (OS_IOS &&
parseInt(Ti.Platform.version, 10) >= 7) ? 20 : 0;

// then in app.tss
"Window[platform=ios)": {
    top: Alloy.Globals.winTop,
    ...
}
```

HANDLING STATUS BAR COLOR



```
// in alloy.js
Ti.UI.backgroundColor = "#fff"; // sets bar color, text remains black
```

```
// if you want a dark status bar, get white text with:
$.win.statusBarStyle = Titanium.UI.iPhone.StatusBar.LIGHT_CONTENT;
// iOS 7 only // for an iOS 6 and 7 solution, try
$.win.statusBarStyle = Titanium.UI.iPhone.StatusBar.TRANSLUCENT_BLACK;
```

```
<!-- App-wide: white status bar text -->
<ios>
  <plist>
    <dict>
      <key>UIStatusBarStyle</key>
      <string>UIStatusBarStyleLightContent</string>
      <key>UIStatusBarHidden</key><true/>
    </dict>
  </plist>
</ios>
```

In this lesson, you:

- ▶ Explore iOS Platform Characteristics
- ▶ Explore Key UI and non-UI APIs

Q&A

Add settings to an iOS application

Enable dynamic text that responds to changes made in
the Settings app

wiki.appcelerator.org/display/tdd/07---+iOS+Deep+Dive

ANDROID API DEEP DIVE

TITANIUM CERTIFIED EXPERT (TCE) TRAINING

IN THIS LESSON, YOU WILL:

- ▶ Identify the strength and weaknesses of the Android platform
- ▶ Explore Android components and vocabulary
- ▶ Configure Android apps using native configuration techniques
- ▶ Implement Android UI APIs
- ▶ Implement Android non-visual APIs

PLATFORM CHARACTERISTICS

- ▶ Open nature (hackable)
- ▶ Variety of app distribution methods
- ▶ Range of devices: low-cost to high-end
- ▶ Java-based environment (common skill set)
- ▶ Strongly integrated into the Google ecosystem
(identity, Google Apps, data sharing)

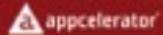
- ▶ Slow OS upgrades on user devices
- ▶ Carrier themes add a layer of complexity in UI design
- ▶ Large distribution of device screen types, hardware capabilities, etc.
- ▶ Open nature (hackable)
- ▶ Less active app economy (fewer 99-cent purchases)

Phones, tablets, various operating system versions,
vendor skins, carrier add-ons, forked versions

Need to test as widely as possible ... on device

ANDROID COMPONENTS

ANDROID APPLICATION KEY COMPONENTS



- ▶ Activities
- ▶ Services
- ▶ Broadcast Receivers
- ▶ Intents
- ▶ Pending Intents

It is necessary to understand and implement these in Ti to provide a truly native experience

<http://developer.android.com/guide/topics/fundamentals.html>

'An activity is a single, focused thing that the user can do. Almost all activities interact with the user, so the Activity class takes care of creating a window for you in which you can place your UI'

<http://developer.android.com/reference/android/app/Activity.html>

'A Service is an application component representing either an application's desire to perform a longer-running operation while not interacting with the user or to supply functionality for other applications to use.'

<http://developer.android.com/reference/android/app/Service.html>

'A broadcast receiver is a component that responds to system-wide broadcast announcements'

<http://developer.android.com/reference/android/content/BroadcastReceiver.html>

'Three of the core components of an application - activities, services, and broadcast receivers - are activated through messages, called intents.'

<http://developer.android.com/guide/topics/intents/intents-filters.html>

A Pending Intent is an intent you give to another application to perform on your app's behalf at a future time.

<http://developer.android.com/reference/android/app/PendingIntent.html>

ALL OF THESE WORK IN TITANIUM

The screenshot shows the Appcelerator Titanium 3.0 documentation interface. The top navigation bar includes icons for Home, Help, Recent, and a search bar. The active tab is 'Android'. The left sidebar contains a tree view of modules: Global, Alloy, Modules, Titanium (expanded), Accelerometer, Analytics, Android (expanded), Calendar, NotificationManager, ActionBar, Activity, Intent, Menu, MenuItem, Notification, PendingIntent, R, RemoteViews, Service, and ABI. The main content area is titled 'Titanium.Android' [Submodule of Titanium]. It features tabs for 'Properties 185' and 'Methods 10'. Below the tabs, the full path is shown: Titanium.Proxy > Titanium.Module > Titanium.Android. A descriptive text block states: 'The top level Android module.' It explains that the currentActivity property provides a reference to the context's current Activity. Activity objects can start a new activity from an existing activity using Activity.startActivity or Activity.startActivityForResult. Another text block notes that almost all constants defined in this module correspond directly to Android constants, with a link to the Android developer's guide for more information. A 'See also:' section lists 'Intents and Intent Filters in the Android Developer's Guide'.

Titanium.Android [Submodule of Titanium]

Properties 185 Methods 10

Titanium.Proxy > Titanium.Module > Titanium.Android

The top level Android module.

The `currentActivity` property provides a reference to the context's current `Activity`. `Activity` objects can start a new activity from an existing activity using `Activity.startActivity` or `Activity.startActivityForResult`.

Almost all of the constants defined in this module correspond directly to Android constants, and a link to the Android developer's guide for more information.

See also:

- Intents and Intent Filters in the Android Developer's Guide

ANDROID CONFIGURATION

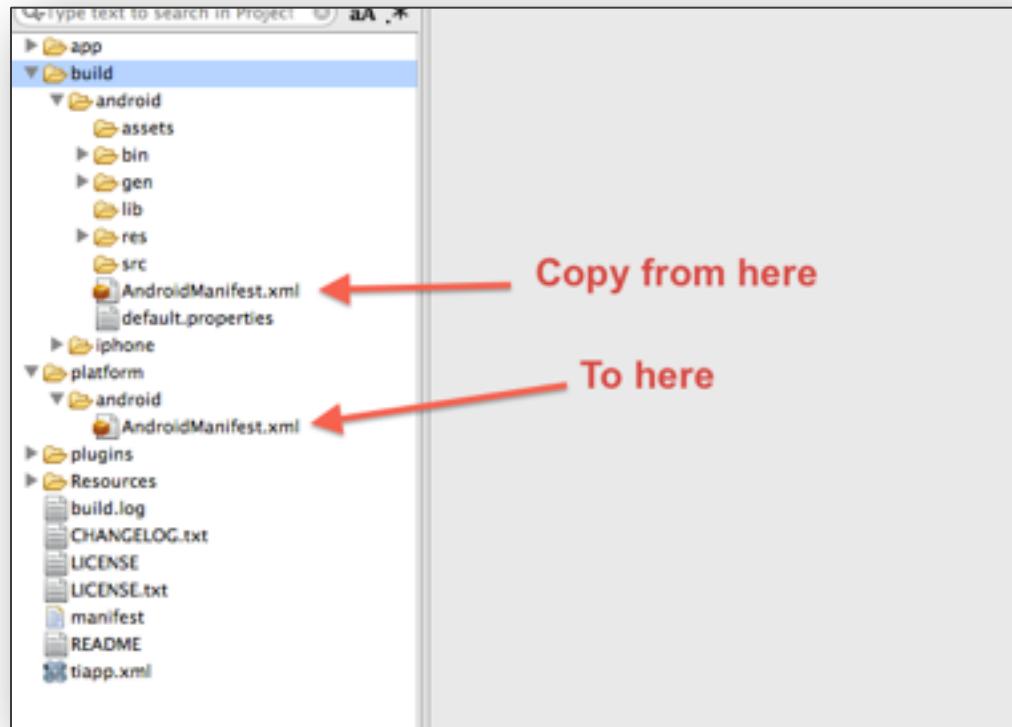
- ▶ Native configurations done in AndroidManifest.xml
- ▶ Most of those properties can be set in the tiapp.xml
- ▶ Or, use a custom AndroidManifest.xml
- ▶ Examples: app permissions, orientation handling, version number customization, native themes, tooling, etc.

CONFIGURATION IN TIAPP.XML

```
48- <android xmlns:android="http://schemas.android.com/apk/res/android">
49-     <manifest>
50-         <uses-sdk android:minSdkVersion="14" android:targetSdkVersion="17"/>
51-         <application android:debuggable="false"
52-             android:icon="@drawable/appicon"
53-             android:label="URLSchemes" android:name="URLSchemesApplication">
54-             <!-- all the activity tags and more go here -->
55-         </application>
56-     </manifest>
57- </android>
```

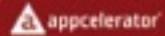
1. Build once to generate stock build/
android/AndroidManifest.xml
2. Copy pertinent tags from generated file
(e.g. all of the **<application>** block)
3. In tiapp.xml, edit **<android .../>** to be
block rather than closed tag
4. Add **<manifest></manifest>** tags
5. Paste in generated code, edit as needed

CUSTOM ANDROID MANIFEST



ANDROID UI APIs

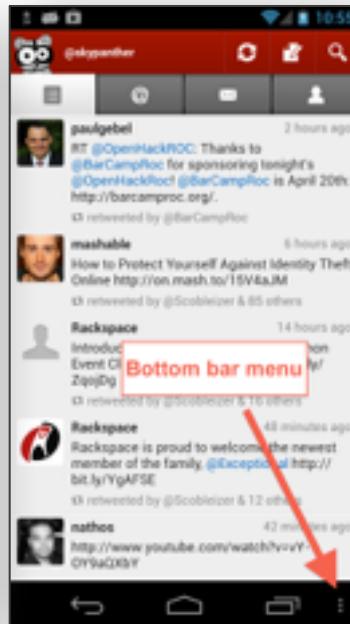
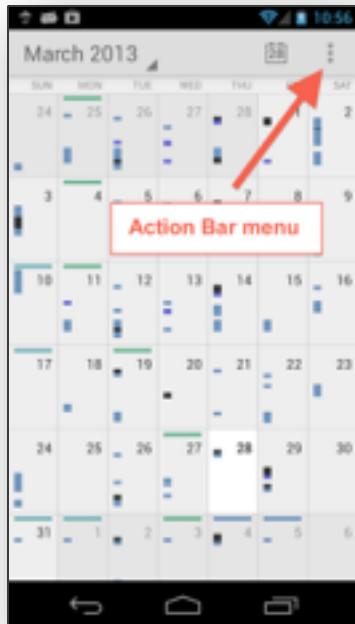
WINDOWS AND ACTIVITIES



- Starting with Ti3.2, a Window == Activity
- Menus, title bars, and more are associated with the activity
- Enter & exit animations on activities
- Older SDKs distinguished between heavyweight (activities) and lightweight (full-screen views) windows
- In a TabGroup, the activity is associated with the TabGroup not the windows

```
$.win.open({  
    activityEnterAnimation:  
    Ti.Android.R.anim.slide_in_left,  
    activityExitAnimation:  
    Ti.Android.R.anim.slide_out_right  
});
```

HARDWARE MENU



- ▶ Menu of options displayed when "hardware" button is pressed
- ▶ Associated with activity (heavyweight window)
- ▶ Pre-ICS: shown at bottom of screen as slide up tray
- ▶ ICS: either in Action Bar or in bottom bar as ellipsis button

MENU EXAMPLE

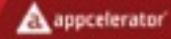


```
<Menu>
    <MenuItem title="Compose" icon="/images/compose_icon.png"
        showAsAction="Ti.Android.SHOW_AS_ACTION_IF_ROOM "
        onClick="doSomething"/>
</Menu> ;
```

MENU, CONTINUED

To get the ActionBar style menu, you just need to build with a current SDK, which is the

default



```
<android xmlns:android="http://schemas.android.com/apk/res/  
    android">  
  
    <manifest>  
  
        <!-- target SDK 14+ (Android 4.0) to get ActionBar style  
        menus  
  
            default if you omit uses-sdk is to target highest  
            installed -->  
  
            <uses-sdk android:minSdkVersion="10"  
                android:targetSdkVersion="14"/>  
  
            <!-- other manifest entries including <application> tag-->  
    </manifest>  
  
</android>
```

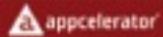
MENUS IN TABGROUPS



- TabGroups are the activity (not the windows)
- So menu code goes inside the tab group
- One menu for whole app
- Call `Ti.Android.invalidateOptionsMenu()` within Window to re-create a menu (i.e. have separate menus for each window)

```
<Alloy>
    <TabGroup id='tabGroup'>
        <Require src="Fugitives" id="fugitivesTab"/>
        <Require src="Captured" id="capturedTab"/>
        <Menu>
            <MenuItem title="Add" onClick="addNewFugitive"/>
        </Menu>
    </TabGroup>
</Alloy>
```

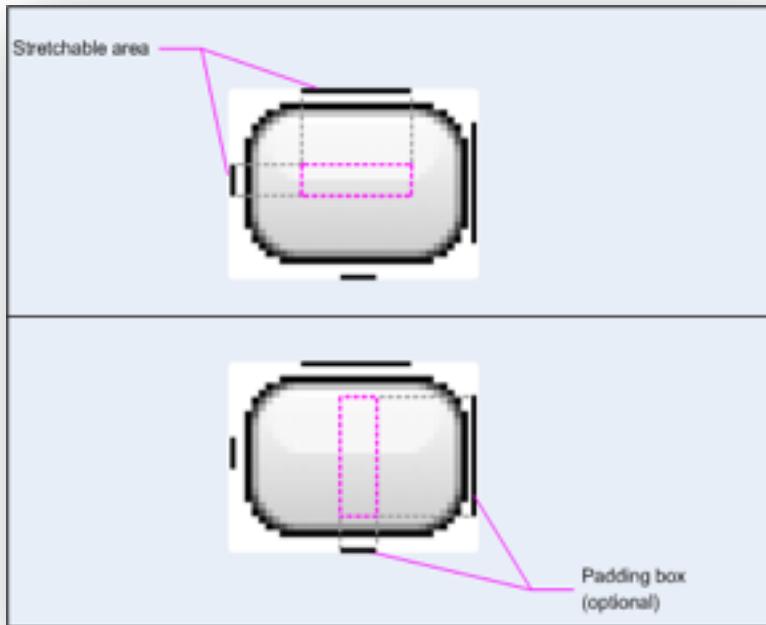
INSTALLING TO THE SD CARD



- Must build with SDK tools 8 or higher (default is to use highest installed)
- Set minSdkVersion to 10 or higher
- Options are preferExternal, preferInternal, or auto (default)

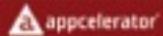
```
<android xmlns:android="http://schemas.android.com/apk/res/android">
    <manifest android:installLocation="preferExternal">
        <uses-sdk android:minSdkVersion="10" />
    </manifest>
</android>
```

NINEPATCH IMAGES



- ▶ Avoid stretched splash screens
- ▶ Define stretchable areas of your graphic
- ▶ Enable through Android theme modification

USING A NINEPATCH FOR THE SPLASH SCREEN



1. Create a NinePatch image using draw9patch utility
2. Copy to platform/android/res/drawable[-xdpi]
3. Rename it to splash.9.png
4. Create a theme.xml file, put it in platform/android/res/values

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <style name="Theme.Titanium" parent="android:Theme">
        <item name="android:windowBackground">@drawable/splash</item>
        <item name="android:windowNoTitle">true</item>
    </style>
</resources>
```

- ▶ Use Label.html to apply formatting like you would use NSAttributedString on iOS
- ▶ Use Label.autoLink to "linkify" URLs, phone numbers, address, etc.
- ▶ autoLink also works on TextAreas

TOAST NOTIFICATIONS



- ▶ Simple text display over all activities
- ▶ Can control positioning on screen
- ▶ Rendering will be different based on OS version and skin

```
var n = Ti.UI.createNotification({message:"Howdy folks"});  
n.duration = Ti.UI.NOTIFICATION_DURATION_LONG;  
// or NOTIFICATION_DURATION_SHORT  
  
// Setup the X & Y Offsets  
n.offsetX = 100;  
n.offsetY = 75;  
  
n.show();
```

- ▶ Override default Back button behavior
- ▶ Example: wizard interface where you want to go back a screen
- ▶ Careful about user expectations
- ▶ Demo: AndroidBackDemo

NON-VISUAL APIs

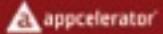
- ▶ You can launch other apps (activities) from JS
- ▶ Need to have an intent object to pass
- ▶ Many built in intents to use
- ▶ Forging Titanium #9 - Android Intents Cookbook

<http://bit.ly/ryOSW4>

In the Android SDK directory, under /platforms/android-x/data
(x being the API level):

- ▶ activity_actions.txt
- ▶ broadcast_actions.txt
- ▶ categories.txt
- ▶ features.txt

EXAMPLE

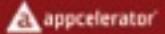


```
// create an Android intent whose action is to send plain text data
var intent = Ti.Android.createIntent({
    action: Ti.Android.ACTION_SEND,
    type: 'text/plain'
});
// define two extra fields for the intent
intent.putExtra(Ti.Android.EXTRA_SUBJECT, 'Isn\'t This Cool!');
intent.putExtra(Ti.Android.EXTRA_TEXT, $.message.value);

try {
    Ti.Android.currentActivity.startActivity(intent);
} catch (ex) {
    /* Handle Exception if no suitable apps installed */
    Ti.UI.createNotification({ message : 'No sharing apps installed!' }).show();
}
```

- ▶ Run JavaScript-based services in the background
- ▶ Must be started by your app, but can survive when you exit the app
- ▶ Runs on an interval you specify
- ▶ Can access many non-UI Titanium APIs (networking, eventing, etc.)

EXAMPLE



```
// app/lib/logservice.js
// This is the service, use non-UI
Ti APIs Ti.API.info("Hello world, I'm a service");
```

```
<!-- tiapp.xml -->
<android xmlns:android="http://schemas.android.com/apk/res/android">
    <services>
        <service url="logservice.js"
            type="interval"/>
    </services>
</android>
```

```
// in index.js or other controller
var SECONDS = 10; // every 10 seconds
var intent = Titanium.Android.createServiceIntent({
    url: 'logservice.js'
});
intent.putExtra('interval', SECONDS * 1000); // Needs to be milliseconds
Ti.Android.stopService(intent); // later, to stop the service!
```

- ▶ JS access to R.java - <http://developer.android.com/reference/android/R.html>
- ▶ R.drawable - built in icons for ImageView, etc.
- ▶ R.string - OS localized string for 'OK', 'Cancel', etc
- ▶ Android docs required to see properties

In this lesson, you:

- ▶ Identified the strength and weaknesses of the Android platform
- ▶ Explored Android components and vocabulary
- ▶ Configured Android apps using native configuration techniques
- ▶ Implemented Android UI APIs
- ▶ Implemented Android non-visual APIs

Q&A

- ▶ Share data with an Intent
- ▶ Hide the keyboard using the Android-specific technique
- ▶ Use the Android-specific technique to prevent the soft keyboard from covering the text area
- ▶ Install the app to the external storage location

wiki.appcelerator.org/display/td/07+-+Android+Deep+Dive

APP INTERCONNECTIONS

TITANIUM CERTIFIED EXPERT (TCE) TRAINING

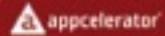
IN THIS LESSON, YOU WILL:

- ▶ Launch native and third-party apps
- ▶ Configure your app's URL scheme
- ▶ Pass data between apps

LAUNCH OTHER APPS

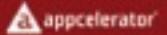
- ▶ **Ti.Platform.openURL()**
- ▶ Need to know the "URL scheme"
- ▶ On Android, launch an intent directly

COMMON URL SCHEMES



App	Scheme	Example
Maps	iOS: maps:// comgooglemaps:// Android: geo:	<i>maps://maps.google.com/maps?q=440+Bernardo+Ave,+Mountain+View comgooglemaps://? center=40.76581,-73.97586&zoom=14 geo:0,0?q=440+Bernardo+Ave,+Mountain+View,+CA http://maps.google.com/maps?q=440+Bernardo+Ave,</i>
SMS	sms:	<i>sms:+1234567890 (no // is used)</i>
Phone	tel:	<i>tel:408-555-1212 (no // is used)</i>
Mail	mailto://	<i>mailto://fake@example.com?subject=Hello</i>
App Store	iOS: http:// Android: market://	<i>http://itunes.apple.com/us/app/legoland-california-official/id452395530?mt=8 market://details?id=com.zc.android</i>

EXAMPLE



Open the native maps app

```
function doClickMaps(e) {  
    if(OS_IOS) {  
        Ti.Platform.openURL('maps://maps.google.com/maps?q=440+Bernardo+Ave,+Mountain  
+View,+CA');  
    } else {  
        // no slashes, 0,0 could be a lat/long or use address as shown  
        Ti.Platform.openURL('geo:0,0?q=440+Bernardo+Ave,+Mountain+View,+CA');  
        // on Android, can also use http:// ...  
        Ti.Platform.openURL('http://maps.google.com/maps?q=440+Bernardo+Ave,+Mountain  
+View,+CA');  
    }  
}
```

DRIVING DIRECTIONS (NAVIGATION)



// iOS - Apple maps

<http://maps.apple.com/?daddr=San+Francisco,+CA&saddr=cupertino>

// iOS - Google maps

<comgooglemaps://?daddr=440+Bernardo+Ave,+Mountain+View,+CA&directionsmode=driving>

// Android - Google maps

<http://maps.google.com/maps?daddr=440+Bernardo+Ave,+Mountain+View,+CA&directionsmode=driving>

// optional, add saddr for source 'address', can use lat/long instead of address

- ▶ [http://wiki.akosma.com/IPhone URL Schemes](http://wiki.akosma.com/IPhone_URL_Schemes)
- ▶ Apple: <http://goo.gl/TgRoO>
- ▶ <http://handleopenurl.com>
- ▶ <http://applookup.com>
- ▶ <http://schemes.zwapp.com>
- ▶ <http://www.openintents.org/en/> (Android)

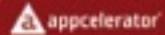
TRY

- ▶ Create a new project
- ▶ Add one button
- ▶ When clicked, open the native maps app to a location you choose
- ▶ Account for Android/iOS differences
- ▶ Test it in the simulator/emulator or on a device

CUSTOM URL SCHEMES

- ▶ Let other apps launch yours
- ▶ Need to select your own 'myappsscheme://' URL scheme string
- ▶ No registration or oversight, so choose carefully to avoid conflicts
- ▶ Requires you to configure the tiapp.xml with custom platform-specific application properties

CUSTOM IOS URL SCHEMES



- App ID must match your app's App ID
- 'Urlschemes' is the custom URL Scheme, should be lead-capped

```
<ios>
<plist><dict>
<key>CFBundleURLTypes</key>
<array>
<dict>
<key>CFBundleURLName</key>
<string>com.company.yourapp</string>
<key>CFBundleURLSchemes</key>
<array>
<string>Urlschemes</string>
</array>
</dict>
</array>
</dict></plist>
</ios>
```

- ▶ Modify AndroidManifest.xml
- ▶ Either via tiapp.xml or /platform/android/AndroidManifest.xml
- ▶ Modify the activity node that you want launched — it doesn't have to be your main activity
- ▶ Lots of extra options — see the Google docs

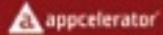
MANIFEST EXAMPLE



- You're enabling a special intent filter
- Use action type VIEW to launch your app
- 'urlschemes' is your URL scheme, by convention it's lowercase

```
<activity android:name=".UrlschemesActivity"
    android:label="URLSchemes" android:theme="@style/Theme.Titanium"
    android:configChanges="keyboardHidden|orientation">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
        <action android:name="android.intent.action.VIEW"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <category android:name="android.intent.category.BROWSABLE"/>
        <data android:scheme="urlschemes" android:host="" />
    </intent-filter>
</activity>
```

CHECKING IF TARGET APP EXISTS



- ▶ iOS: Use `Ti.Platform.canOpenURL()`
- ▶ Android: `Ti.Platform.openURL()` returns a Boolean; check for **false** to know if app was opened
- ▶ Then, prompt user to download the necessary app
- ▶ Could open the app store for them!

EXAMPLE

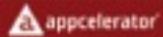


```
function doClick(e) {  
    if(OS_IOS) {  
        if(Ti.Platform.canOpenURL('Urlschemes://')) {  
            Ti.Platform.openURL('Urlschemes://');  
        } else {  
            alert('You must install the Target app first');  
        }  
    } else {  
        var didItWork = Ti.Platform.openURL('urlschemes://');  
        if(!didItWork) {  
            alert('You need to install the Target app');  
        }  
    }  
}
```

APP ARGUMENTS

- ▶ Pass data to another application when you launch it via its URL scheme
- ▶ Work like query parameters on a 'normal' URL
- ▶ You get the entire URL in the target app & need to parse out the arguments
- ▶ Of course, how you do this varies by platform

LAUNCH ARGUMENTS ON IOS

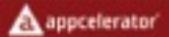


- Use Ti.App.getArguments() in the target application
- Use a 'resumed' app event listener to get arguments when app launched from backgrounded state

```
function grabArguments() {  
    var args = Ti.App.getArguments();  
    // data validation, parsing  
}  
grabArguments();  
Ti.App.addEventListener(  
    'resumed', grabArguments);
```

- ▶ Access via the current Activity
- ▶ But, need to grab the activity reference in alloy.js
- ▶ From the activity, you get a handle to the Intent that launched the activity
- ▶ From that intent, you can call getData()
- ▶ Data available even if app launched from backgrounded state

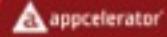
EXAMPLE



```
// alloy.js
if(OS_ANDROID) {
    Alloy.Globals.action = '';
    var activity = Ti.Android.currentActivity;
    function doIt(e) {
        var args = activity.getIntent().getData();
        if(args && args != 'urlschemes://') {
            // parse the url string and arguments
            var parsedArgs = urlParser(args);
            switch(parsedArgs[0]) {
                case 'maps':
                    Alloy.Globals.action = 'maps';
                    break;
                case 'youtube':
                    Alloy.Globals.action = 'youtube';
                    break;
            }
        }
    }
    activity.addEventListener("start", doIt);
}
```

```
// index.js
if(OS_ANDROID) {
    $.index.addEventListener('open', function() {
        switch(Alloy.Globals.action) {
            case 'maps':
                doClickMaps();
                break;
            case 'youtube':
                doClickYouTube();
                break;
        }
    });
}
```

PARSING ARGUMENTS

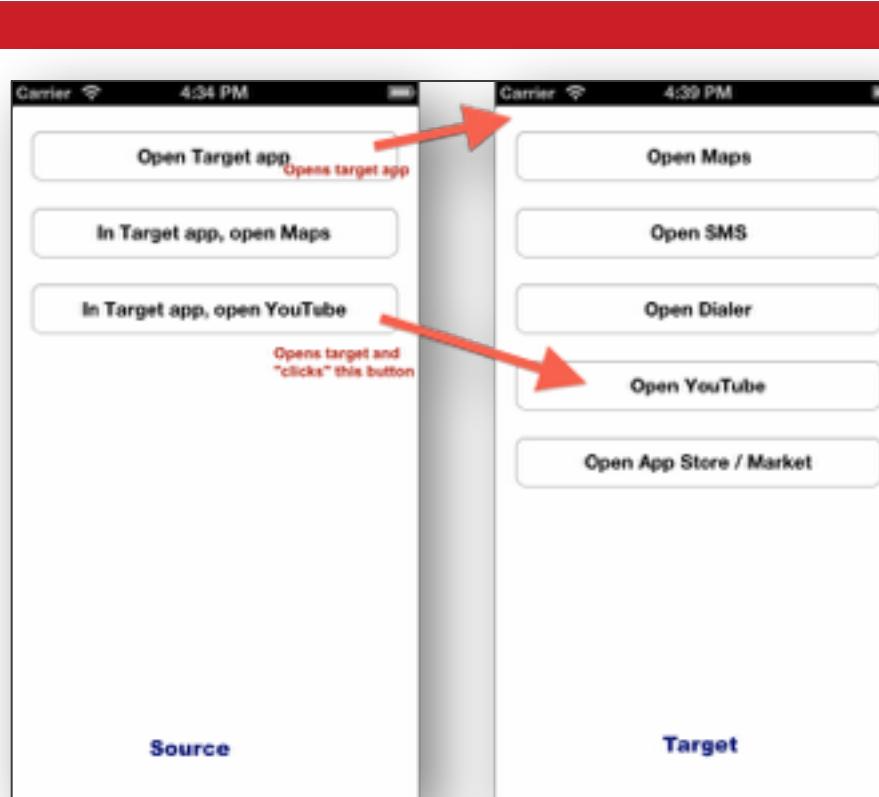


```
function urlParser(url) {  
    // strips out urlschemes:// (and Urlschemas://)  
    url = url.replace(/[Uu]rlschemas:\//\?/, "");  
    // splits and returns an array of arguments  
    return url.split('&');  
}
```

Just like with web applications, you should:

- ▶ Check for correct data types
- ▶ Don't 'blindly' pass arguments to database queries
- ▶ Confirm permissions and security
- ▶ Don't expose destructive actions via launch arguments

DEMO



Some links work only on device

In this lesson, you:

- ▶ Launched native and third-party apps
- ▶ Configured your app's URL scheme
- ▶ Passed data between apps

Q&A

In this lab, you will:

- ▶ Create the two apps just demoed
- ▶ Target app itself can launch various native apps
- ▶ Source app will launch target
- ▶ Buttons in Source app will launch the native apps via target app

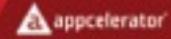
Lab: wiki.appcelerator.org/display/td/o8++App+Interconnections

SOLUTION WALK THRU

PERFORMANCE AND OPTIMIZATION

TITANIUM CERTIFIED EXPERT (TCE) TRAINING

IN THIS LESSON, YOU WILL:



- ▶ Manage and profile memory in a Titanium app
- ▶ Identify performance best practices

MEMORY MANAGEMENT

MEMORY LIMITATIONS



Computation is usually not the bottleneck, RAM is

iOS	Android
<ul style="list-style-type: none">iPhone ~12 MBiPad ~30-50 MB	<ul style="list-style-type: none">Froyo/Gingerbread (2.2/2.3) - 24 MB
	<ul style="list-style-type: none">ICS - 24 MB
<i>* Apple does not publish information about their app termination threshold</i>	<ul style="list-style-type: none">ICS w/ "large heap" - 128 MB

- ▶ Understand JavaScript garbage collection
- ▶ Understand native-layer memory deallocation
- ▶ Write high-quality code
- ▶ Profile your app

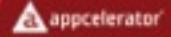
- ▶ Automatic (no manual allocation/deallocation)
- ▶ 'Mark and Sweep' stops execution periodically & briefly
- ▶ Interpreter decides when and how
- ▶ Objects collected when no references remain
- ▶ Nothing in the global space is ever collected!
- ▶ "Force" collection by encapsulating within functions or modules that will go out of scope
- ▶ Or, set variables to **null**

Object collected/nulled in JS signals native to deallocate

```
$ .win.close(); // ← its children can be collected  
$.imageview = null; // ← native proxy can be deallocated  
(function() {  
    var win = Ti.UI.createWindow();  
    win.open(); // ← when user closes this window, it will be  
    // collected  
})();
```

- ▶ Don't store variables in the global scope
- ▶ Set objects = null when no longer needed
- ▶ Watch for "hidden" references in functions, closures, event listeners, accidental global assignments

LIMIT IMAGES



- Images are bitmap in memory
- No compression in memory
- File format doesn't matter
- Lazy load images – when you need them
- Then unload / destroy images you no longer need them

200 by 200 px image

= 40,000 pixels x 32 bits per pixel

= 1,280,000 bits

= 160,000 Bytes

= 156.25 KB

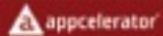
= ~1/6th MB

- ▶ iOS: Use Instruments — either Allocations or Leaks
- ▶ Android: Use Monitor (more difficult)

PERFORMANCE TIPS

- ▶ Defer loading a script until it is *actually needed*
- ▶ Use **require()** to capitalize on caching
- ▶ Avoid **Ti.include()**, remember **eval()** is "evil"
- ▶ Use Alloy's conditionals, e.g. **IS_IOS**, **IS_ANDROID**

JAVASCRIPT OPTIMIZATIONS

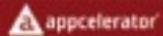


Avoid trips across the bridge!

```
var table = Ti.UI.createTableView();
var data = [];
for(var i = 0; i < 10000; i++){
    data.push(
        Ti.UI.createTableViewRow({
            title:'Row #' + i,
            selectionStyle: Ti.UI.iPhone.TableViewCellStyle.BLUE
        });
}
table.setData(data);

// References: https://gist.github.com/2989311
// and https://gist.github.com/3056032
```

JAVASCRIPT OPTIMIZATIONS

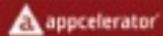


Avoid trips across the bridge!

```
var table = Ti.UI.createTableView();
var data = [];
for(var i = 0; i < 10000; i++){
    // generic object references can be faster
    data.push({
        title:'Row #' + i,
        selectionStyle: Ti.UI.iPhone.TableViewCellSelectionStyle.BLUE
    });
}
table.setData(data);

// References: https://gist.github.com/2989311
// and https://gist.github.com/3056032
```

JAVASCRIPT OPTIMIZATIONS



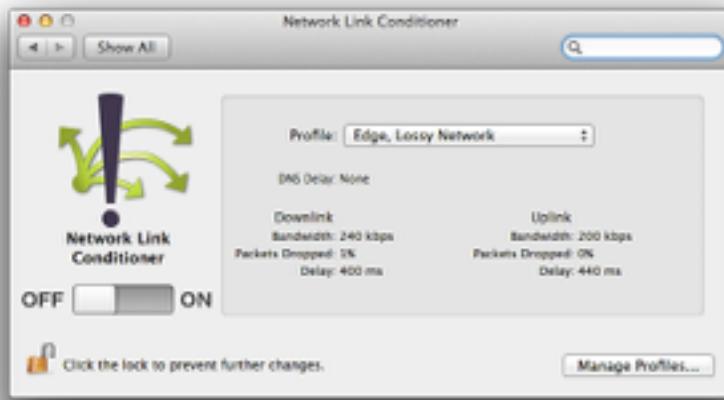
And this is even faster!

```
var table = Ti.UI.createTableView();
var data = [];
// create local references to Ti properties
var SEL_STYLE_BLUE = Ti.UI.iPhone.TableViewCellStyle.BLUE;
for(var i = 0; i < 10000; i++){
    data.push({
        title:'Row #' + i,
        selectionStyle: SEL_STYLE_BLUE
    });
}
table.setData(data);

// References: https://gist.github.com/2989311
// and https://gist.github.com/3056032
```

- ▶ SQL Queries are fast - use these rather than in-memory sorting of JS arrays, etc.
- ▶ Wrap multiple inserts in a transaction
- ▶ Avoid loading Blobs into memory — don't store in database
- ▶ Open/close connections with each use
- ▶ SQLite Optimization FAQ - <http://j.mp/HFxVZ8>

NETWORK PERFORMANCE



- ▶ Don't assume WiFi — test on cell networks
- ▶ OS X: Network Link Conditioner
- ▶ Windows: Monitor/DDMS
- ▶ Field testing best
- ▶ Decrease service payload sizes
- ▶ ... but make as few requests as possible
- ▶ Use JSON as transport
set up server-side proxy for SOAP/XML services

INSTALLING NETWORK LINK CONDITIONER



1. Open Xcode
2. Choose Xcode > Open Developer Tool > More Developer Tools
3. Download **Hardware IO Tools for Xcode**
4. Double-click **Network Link Conditioner.prefpane**

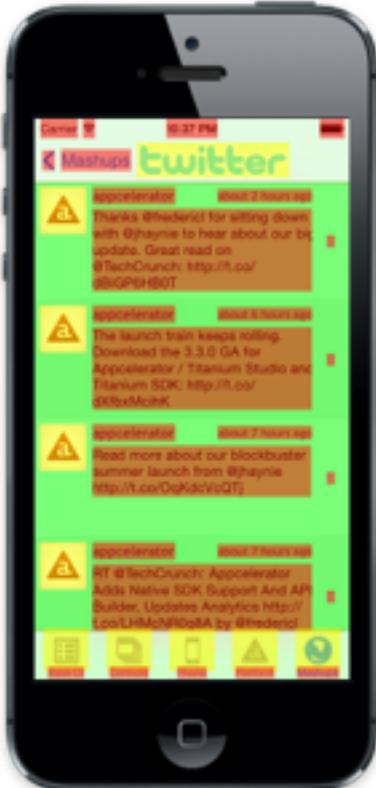
BATTERY OPTIMIZATION



- ▶ Critical on Android, important on iOS too
- ▶ Network and geolocation most critical concerns
- ▶ More info at:
 - ▶ Android — <http://j.mp/IDMk5H>
 - ▶ iOS — <http://j.mp/IDMaeG>

- ▶ Avoid compositing operations: transparency, rounded corners, gradients, drop shadows
- ▶ Set a BG color to match parent container — default is transparent
- ▶ Avoid multiple layout passes: Ti.UI.SIZE/FILL or '80%'
- ▶ Position at integer coordinates, e.g.
`$view.top = Math.round(calculatedPosition)`
- ▶ Use correctly sized images to limit runtime scaling

VIEWING INEFFICIENCIES IN IOS SIMULATOR



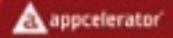
- ▶ Debug > Color Blended Layers
shows transparent/blended layers
- ▶ Debug > Color Misaligned Images
Magenta means subpixel
misalignments, yellow means
stretched/scaled images

In this lesson, you:

- ▶ Managed and profile memory in a Titanium app
- ▶ Identified performance best practices

Q&A

LAB GOALS



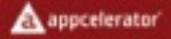
- ▶ Analyze a memory leak
- ▶ Correct the leak and test the app

wiki.appcelerator.org/display/td/09+-+Finding+and+correcting+memory+leaks

USING MODULES IN YOUR APPS

TITANIUM CERTIFIED EXPERT (TCE) TRAINING

IN THIS LESSON, YOU WILL:



- ▶ Explore the purpose of modules and identify sources of modules
- ▶ Install a module for use in a Titanium project

MODULE OVERVIEW & QUICK-TOUR

TITANIUM MODULES



- ▶ Add 'missing' features
- ▶ Narrow use cases
- ▶ Appcelerator & Community created
- ▶ Free, one-time, & subscription-based

- ▶ Some distributed with Titanium, just not installed
- ▶ Via Marketplace
- ▶ Also, Gitt.io, GitHub, Blogs, etc.
- ▶ Variety of licensing options

"BUILT-IN" MODULES

The screenshot shows the Appcelerator Documentation website for Titanium 3.X. The top navigation bar includes links for Documentation, Titanium 3.X (selected), and a search bar. The left sidebar has icons for Home, Settings, Modules (selected), and Help. The main content area is titled "Modules" and describes optional Titanium modules. It lists "Facebook", "Map", "Nfc", and "Titanium". A note states: "Optional Titanium modules. Modules in this section are optional modules that are packaged with the Titanium SDK. These modules are not part of the Titanium namespace, and must be added to the `titanium.xmll` file and imported using `require`". At the bottom, there are links for "By Package" and "By Inheritance". The footer contains links for platform, cloud services, marketplace, think mobile, get help, partners, company, developers, Privacy Policy, Legal Information, and a note about Appcelerator being a registered trademark.

Documentation Titanium 3.X Search

Modules

Global

Alloy

Modules

- Facebook
- Map
- Nfc
- Titanium

Modules (Object)

Optional Titanium modules.

Modules in this section are optional modules that are packaged with the Titanium SDK. These modules are not part of the Titanium namespace, and must be added to the `titanium.xml` file and imported using `require`.

By Package By Inheritance

platform | cloud services | marketplace | think mobile | get help | partners | company | developers
Privacy Policy | Legal Information © 2008–2013 Appcelerator Inc. All rights reserved. Appcelerator is a registered trademark.
We believe in the power of open source. This site is generated using the Simple JavaScript Documentation generator — JSDuck

No download required for Facebook module

APPCELERATOR OPEN SOURCE MODULES



github Explore GitHub Search Features Blog Sign up for free Sign in

PUBLIC appcelerator / titanium_modules

Code Network Pull Requests Graphs

Modules for Appcelerator's Titanium — Read more <http://developers.appcelerator.com>

Clone in Mac ZIP HTTP SSH Git Read-Only https://github.com/appcelerator/titanium_modules.git Read-Only access

branch: master Files Commits Branched Tags

titanium_modules / 276 commits

Merge pull request #198 from jonalter/MOD-1265

jonalter authored a day ago latest commit 1379ffeb0

File	When	What
lib/admin	a month ago	[TMOOPEN-818] Admin: iOS: updating SDK to 6.3.1 and building with 2.1.3-GA [Jon Alter]
lib/airprint	a month ago	[TMOOPEN-818] Fixing Copyright in Source [Jon Alter]
lib/barcode	a day ago	[MOD-1265] Barcode: building iOS with 2.1.3-GA and open sourcing [jonalter]
lib/box	a month ago	[MOD-945] Box: building with 2.1.3-GA and open sourcing [Jon Alter]
lib/box2d	9 days ago	[MOD-1087] Box2d: adding attribution [jonalter]
lib/bump	9 days ago	[MOD-1266] Bump: removing .jar from dist folder [jonalter]
lib/canvas	a month ago	[TMOOPEN-818] Fixing Copyright in Source [Jon Alter]
lib/charts	22 days ago	[MOD-1120] Charts: building with 2.1.3-GA and open sourcing [Jon Alter]
lib/columns	22 days ago	[MOD-937] Columns: building with 2.1.3-GA and open-sourcing [Jon Alter]
lib/compression	22 days ago	[MOD-938] Compression: building with 2.1.3-GA and open sourcing [Jon Alter]

APPCELERATOR MARKETPLACE



Products Developers Enterprise Partners Customers Company Contact

shop purchased products account sell

Product Market > Search Results

Search Results

Showing 57 results for "appcelerator".

Sort By: Compare

SalesForce Toolkit for Appcelerator
Connector to force.com database by salesForce.com
REST API wrapper to provide read/write access to Force.com and Database.com including OAuth support for OAuth 2. [Learn More](#)

Compression Module
Compress your files to reduce storage by Appcelerator
Create compressed zip archives to reduce storage requirements. Supports both zip and unzip. Get a great deal on access to all Appcelerator modules at <http://www.appcelerator.com/products/plans-pricing>. Get a great... [Learn More](#)

Ifurry Module
Meet your App Audience by Appcelerator
Ifurry Analytics helps mobile application developers make better apps, deeper consumer engagement and improve monetization of their applications. The service is free and able to handle data loads of any size applicati... [Learn More](#)

StoreKit module

gitTio!

Search & Install all 769 Titanium Modules and 277 Alloy Widgets on GitHub



New modules

	androidmodule	1.1.6	Jul 17
	SimPushServer	1.0	Jul 17
	CrittercismModule	1.0.0	Jul 17
	iOS	1.0.0	Jul 17

New widgets

	info.liquanium.Image	1.0	Jul 12
	info.liquanium.InfiniteScroll	1.0	Jul 12
	Commutr Logo	1.0	Jul 10
	Commutr Trip Row	1.0	Jul 10

PAYPAL (IOS AND ANDROID)



- ▶ Mobile Payments Library
- ▶ Requires PayPal merchant account
- ▶ In development, uses sandbox environment, in production uses live servers

- ▶ StoreKit — iTunes App Store integration
- ▶ In-App Billing — Google Play integration
- ▶ StoreKit can be used for in-app purchases, digital goods
- ▶ Free from Appcelerator

- ▶ Admob, Millennial Media, Greystripe, and InMobi from Appcelerator
- ▶ Tap for Tap, Flurry/AppCircle, and more from third-parties
- ▶ Some free, some not

- ▶ Flurry Analytics
- ▶ Google Analytics
- ▶ Appio Localytics

COMMUNITY CONTRIBUTIONS



github Explore GitHub Search Features Blog Sign up for free Sign in

Follow

Filter benbahrenburg's repositories... All Sources Forks Mirrors

benCoding.Android.Extras A few tools to make working with Titanium Android a little easier Last updated 6 days ago JavaScript ★ 2 ⚡ 0

benCoding.PDF Titanium module to make working with PDFs easier Last updated 5 days ago Python ★ 0 ⚡ 0

Ti.OneTok Titanium Modules for Ti.OneTok Last updated 7 days ago Perl ★ 12 ⚡ 0

Dossier Makes moving and copying Titanium Directories easy Last updated 9 days ago JavaScript ★ 4 ⚡ 0

Securely Last updated 10 days ago Objective-C ★ 10 ⚡ 0

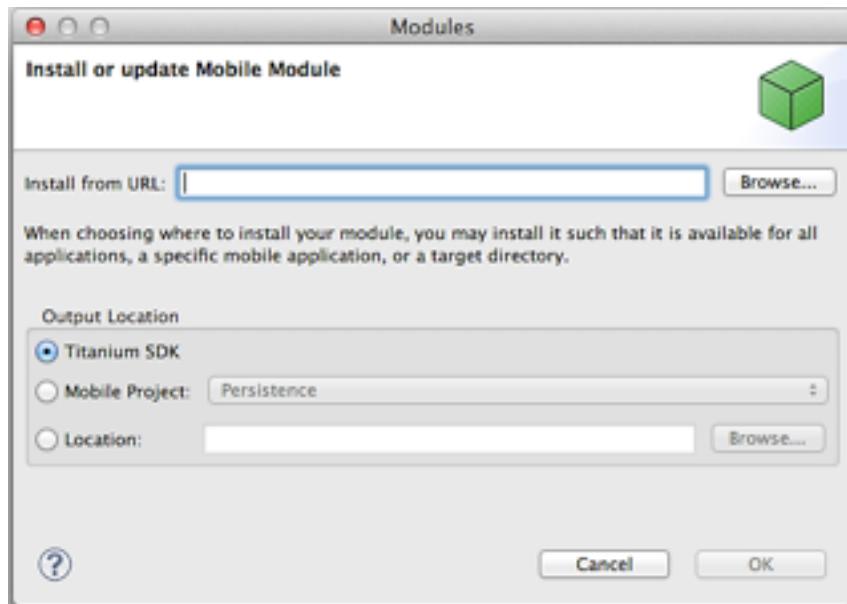
Ben Bahrenburg
benbahrenburg

San Francisco Bay Area
<http://twitter.com/bencoding>
Joined on Oct 27, 2009

101 followers 99 starred 35 following

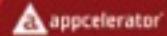
INSTALLING MODULES

INSTALLING A MODULE



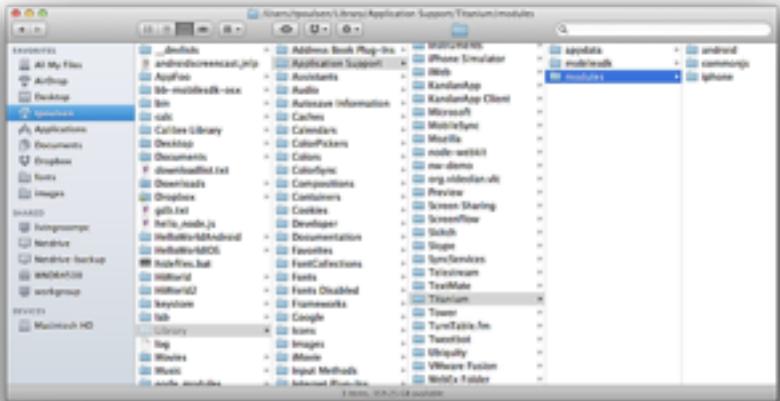
1. Download zip file, or copy URL to zip file
2. Choose Help, Install Mobile Module
3. Paste URL or browse to location
4. Choose Titanium SDK (for use in all future projects) or a specific project

MANUALLY INSTALLING FOR A SINGLE PROJECT



1. Download zip file
2. Place in project's root directory
3. Build once — to unzip and create necessary folders

MANUALLY INSTALLING FOR MULTIPLE PROJECTS

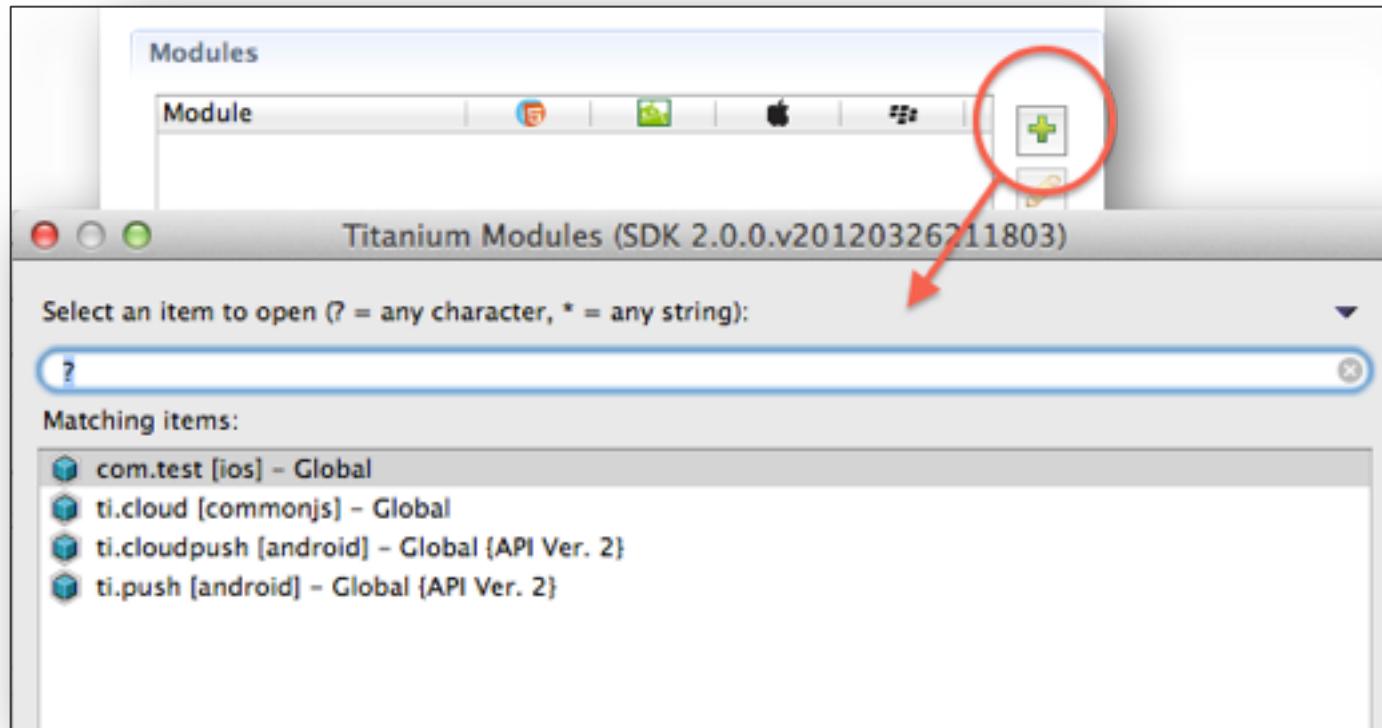


1. Download zip file
2. Unzip to
`%TITANIUM_INSTALL_DIR%/
modules` directory

LOADING AND USING MODULES

LOADING A MODULE

Must declare dependency in tiapp.xml



VIEW-BASED MODULES

If native module exposes a view, you can output it with the <Module> tag

```
13
14      <Module id="map" module="ti.map" method="createView" platform="android"/>
15      <View id="dividerLine" platform="android"/>
16      <Module id="map1" module="ti.map" method="createView" platform="android"/>
17    </Window>
18  </Alloy>
19
```

MODULES IN THE CONTROLLER

```
1: if(OS_ANDROID) {  
2:   // we need a reference to the ti.map module even though we're adding it via the XML  
3:   var MapModule = require('ti.map');  
4:  
5:   // for now, must add annotations in code  
6:   var anno1 = MapModule.createAnnotation({  
7:     title:"Mountain View",  
8:     latitude:37.389549,  
9:     longitude:-122.058212  
0:   });  
1:   $map.addAnnotation(anno1);  
2:   var anno2 = MapModule.createAnnotation({  
3:     title:"Sydney",  
4:     latitude: -33.87365,  
5:     longitude: 151.20689  
6:   });  
7:   $map1.addAnnotation(anno2);  
8:  
9:   // have to set properties that rely on module constants in the controller  
10:  $map1.mapType = MapModule.TERRAIN_TYPE;  
11}  
12
```

- ▶ Must **require()** module in your controller to use its functions
- ▶ (Every marketplace module comes with example and doc)

In this lesson, you:

- ▶ Explored the purpose of modules and identified sources of modules
- ▶ Installed a module for use in a Titanium project

Q&A

- ▶ Download and install a module from the Marketplace
- ▶ Implement it in an app
- ▶ Build for the simulator or emulator

wiki.appcelerator.org/display/td/10+-+Installing+and+Using+a+Titanium+Module

to build a module consider seeing: <http://www.slideshare.net/omorandi/ticonf>

FEEDBACK



[HTTP://FEEDBACK.APPCELERATOR.COM](http://feedback.appcelerator.com)

Books



Thank You

ANDREW MCELROY

@Sophrinix

