

MEAP / Appcelerator Overview

Author: Duy Nguyen

Release: June 2013

Revision: 1.0



Course Objectives

At the end of the course, you will have acquired sufficient knowledge to:

- Understand MEAP in general, and Appcelerator platform particularly.
- Create cross-platform mobile app using Appcelerator Titanium 3.X
- Know how to achieve TCAD certificate.

Course Prerequisite

Trainees should have basic knowledge about following points:

- Object-oriented programming
- Basic level of using JavaScript

Assessment Disciplines

- ❖ Class Participation : Required
- ❖ Assignment Completion : 100%
- ❖ Pass Criteria: $\geq 70\%$

Course Timetable

- ❖ Lecture Duration: 4 hours
- ❖ Hands-on Labs: 8 hours
- ❖ Assignment Duration : One week

Further References

❑ Document References:

- The material of this course
- appcelerator_titanium_up_and_running (pdf)
- appcelerator_titanium_smartphone_app_development_cookbook (pdf)





❑ Online References:

- <http://docs.appcelerator.com/titanium/latest/#!/guide>
- <https://developer.appcelerator.com/questions/newest>
- <https://wiki.appcelerator.org/display/td/TCAD+Course+Labs>









Agenda

MEAP Overview

Appcelerator Platform

-  What is Appcelerator?
-  Appcelerator Platform Features
-  Appcelerator Cloud Service
-  Appcelerator Titanium Overview

Appcelerator Titanium Coding Practice

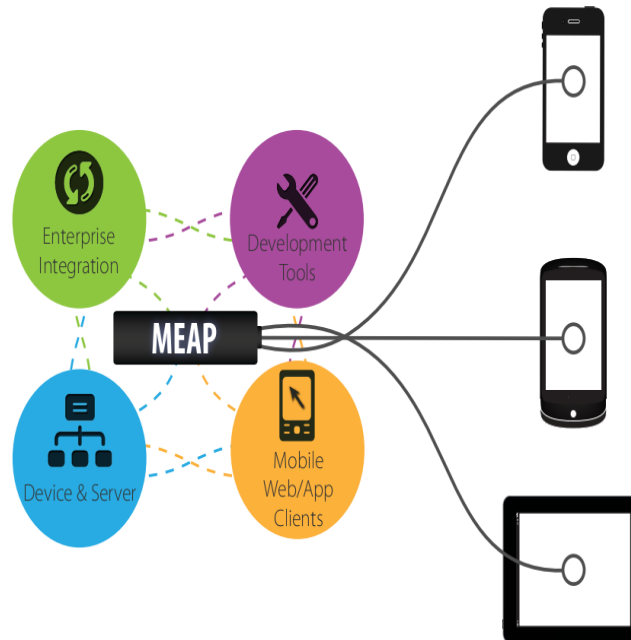
-  User Interface Fundamental
-  Properties & Events
-  Working with Local Data Sources
-  Working with Remote Data Sources
-  Integration with Google Maps & GPS
-  Differences in devices - platforms
-  Deploying and Distributing
-  Working with Cloud Service

MEAP Overview



Definition

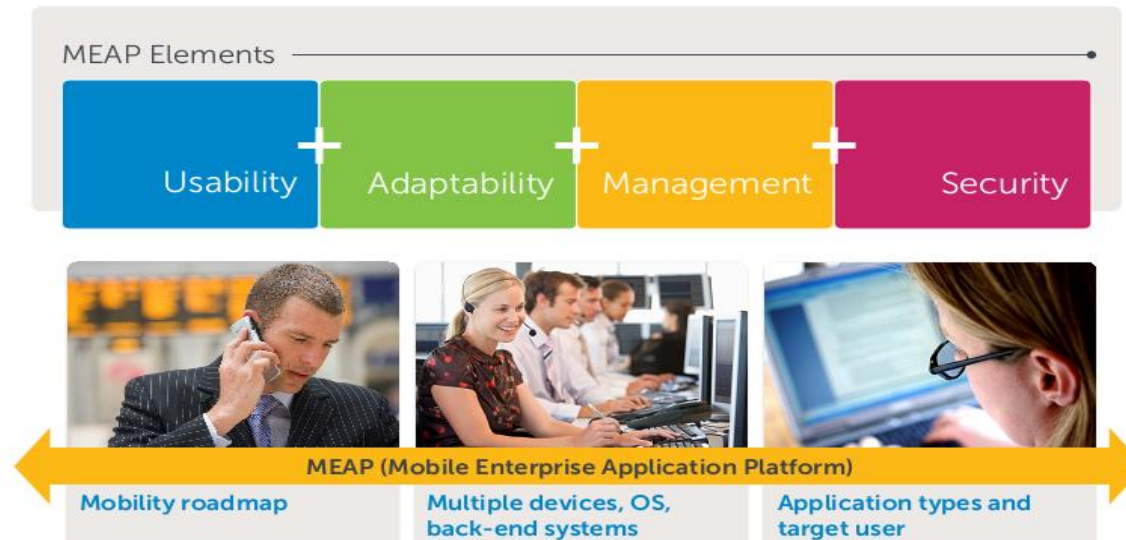
- A **mobile enterprise application platform** (MEAP) is a comprehensive suite of products and services that enable development of mobile applications. (Gartner)
- => Is an integrated development environment that provides tools and client/server middleware for building, deploying and managing mobile applications.



Capacibilities and Support

- ❑ Complete integrated development environment
- ❑ Support for design, develop, test, deploy and manage mobile apps.
- ❑ Graphical WYSIWYG development
- ❑ Run-time middleware server to handle back-end systems of the enterprises
- ❑ Robust security capabilities
- ❑ Local and remote data handling capability
- ❑ Ability to integrate external devices like credit card readers, scanners, printers etc.

Benefits of MEAP



Meap benefits



Faster development and deployment across multiple mobile devices and operating systems



Multiple feature integration



Robust security



Strong back-end connectivity



Easy application and user management

What are MEAPs trying to solve?

» Write app once, run on any device and platform

- Write application in single language and re-compile for native platform

» Take advantage of device hardware

- Provide abstraction layer to take advantage of the device hardware

» Integrate with different data sources

- Set of adapters for XML, databases, web services, SAP, Siebel etc.

» Deployment of applications

- Hosting of application, manage updates and analyze usage

» Management of devices

- Asset management for devices and restrictions for trusted devices

Source-Neudesic

Pros and Cons

Approach	Pros	Cons
MEAP	Highly scalable architecture.	Initial investment is high – requires upfront payment for the solution, maintenance fee and user based fee.
	Pre-integrated to identity management system.	Dedicated administrator is required to maintain the MEAP server.
	Supports multiple platforms – native (iOS, Android, Windows Mobile, BlackBerry), web (XHTML browsers) and hybrid applications.	User interface and usability are sub-optimal.
	Can leverage device features seamlessly.	Initial implementation time is higher.
	Plug-in or connectors are available to communicate with systems such as SAP, Oracle, CRMs, etc.	
	Web service layer runs on DMZ to secure the entire infrastructure.	
	The solution can be deployed on premises or in a cloud.	

Vendors

MEAP Examples



Cross- Platform Development Frameworks Examples



What is Appcelerator



Appcelerator Platform

- Consists of a comprehensive set of integrated products that enable enterprises to create, deliver and analyze their entire mobile application portfolio.



Appcelerator Platform features



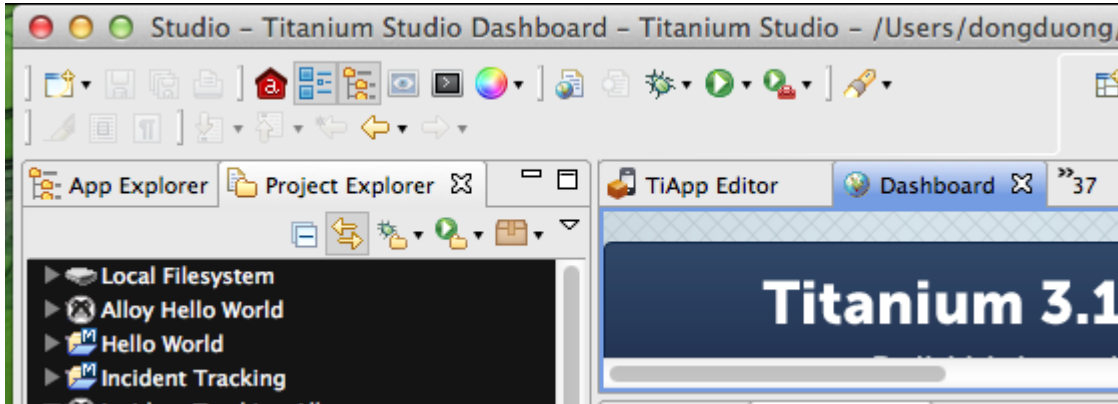
- **Deliver a real-time view of all your mobile apps**
 - The Appcelerator Dashboard displays all relevant information about your portfolio of mobile apps. It enables developers, architects, testers, managers, project teams, and business owners to view, analyze and take action to get their job done.





- **Create amazing native apps**

- **Appcelerator studio:** is a free and open source application development platform that lets you create cross platform native mobile applications using existing Javascript skills

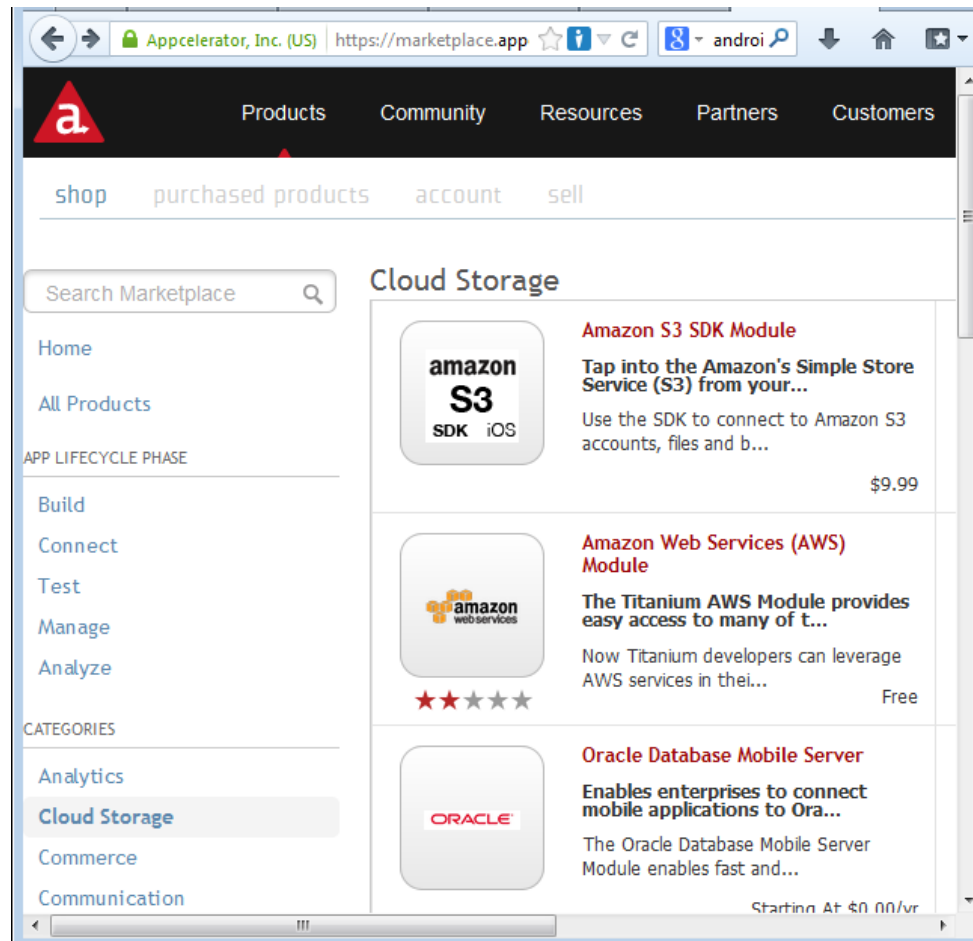


- **Appcelerator Alloy:** Application development framework. It follows a model-view-controller (MVC) architecture



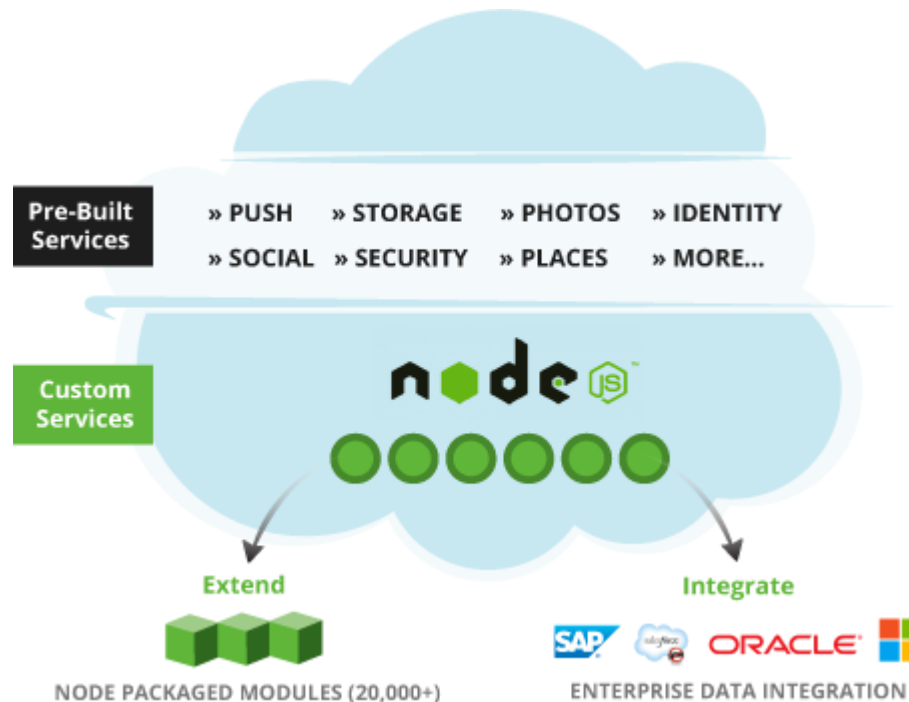
- **Integrate Powerful Extensions:**

- Appcelerator Marketplace is open and integrated with Studio and provides over 370 third party module.
- Modules on the Marketplace are created by independent developers and ISVs alike.



- **Connect to data anywhere**

- Appcelerator Cloud Services is cloud-infrastructure optimized for the delivery of connected mobile apps. Choose from a library of pre-built services such as push notification, photo storage and social integration, or use the custom connectors to mobilize your corporate enterprise data such as SAP, Oracle or Salesforce.





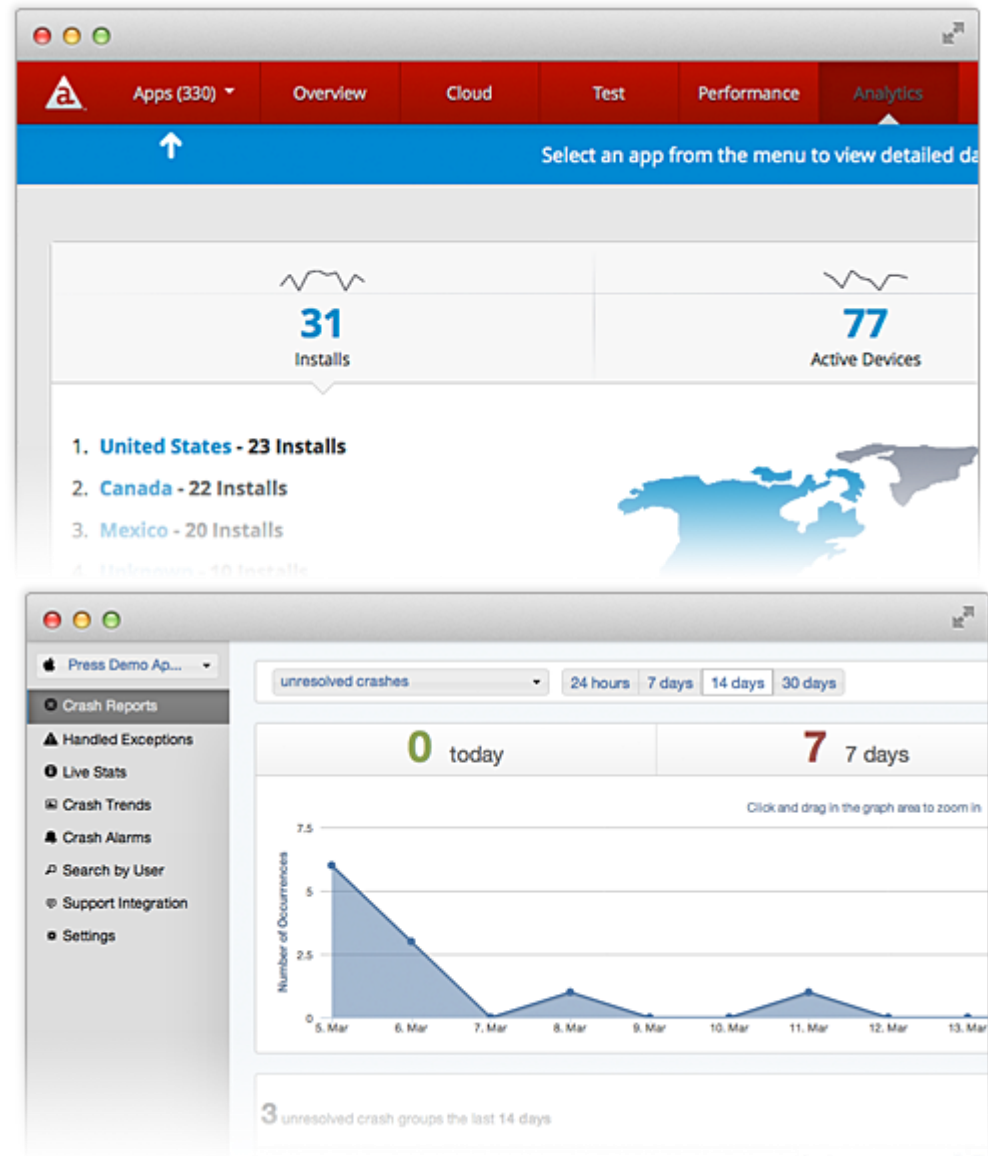
- Release faster with higher quality
 - Developer testing is included within Appcelerator Studio and delivers on-device debugging, 'live' editing of the UI/UX and a profiler to identify memory leaks and performance issues.
 - Appcelerator Test delivers integrated functional test automation to ensure your mobile applications are released with the highest levels of quality and stability. No jail breaking is required.





- **Analyze usage and performance**

- Appcelerator Analytics provides deep insight and visibility into user adoption, engagement and retention for your mobile apps.
- Appcelerator Performance Management Exceptions, app crashes, memory and CPU use at the time of the event.





- Flexible delivery models
- ACS can be consumed in a variety of ways depending on your organization's desired level of security and control.
 - ACS Public Cloud: The back-end is hosted and managed by Appcelerator in the cloud
 - ACS Virtual Private Cloud: The back-end is hosted and managed by Appcelerator in the cloud, but with dedicated infrastructure and VPN access for increased security and control
 - ACS Private Cloud: The back-end is hosted and managed on your premises or data center, for maximum control and access to local data sources



Appcelerator Platform Features

	Titanium (LEARN MORE)	Appcelerator Platform (PUBLIC CLOUD)	Appcelerator Platform (VIRTUAL PRIVATE CLOUD)
	FREE FOR ALL USERS	\$999* PER NAMED USER / MONTH	\$2,667* PER NAMED USER / MONTH
Alloy (MVC Framework)	✓	✓	✓
Marketplace Modules	✓	✓	✓
Custom Cloud Services	✓	✓	✓
Analytics	✗	✓	✓
Enterprise Connectors	✗	✓	✓
Automated Functional Testing	✗	✓	✓
Performance Management	✗	✓	✓
Performance Monitoring	✗	✗	✓
Multi-Region Support	✗	✗	✓
Dedicated Infrastructure	✗	✗	✓
Studio (IDE, SDK)	Community Edition	Enterprise Edition (Code Analyzer, Live view, Profiler, Lifecycle partner extensions)	Enterprise Edition (Code Analyzer, Live view, Profiler, Lifecycle partner extensions)
Pre-Built Cloud Services	Up to 5M push notifications 5M API calls, 20GB of storage, 100,000 emails per month	Up to 10M push notifications 10M API calls, 100GB of storage, 200,000 emails per month	Scales by capacity as required
Support	Community forums	Standard (8x5) / Enhanced (24x7), 45 minute response time	Standard (8x5) / Enhanced (24x7), 15 minute response time

Appcelerator Cloud Service



- Titanium Cloud Services is a Mobile Backend as a Service (MBaaS), offering a fast and easy way to build connected mobile apps.
- Choose from a library of services such as push notification, status updates, photo storage, and social integration, or create your own

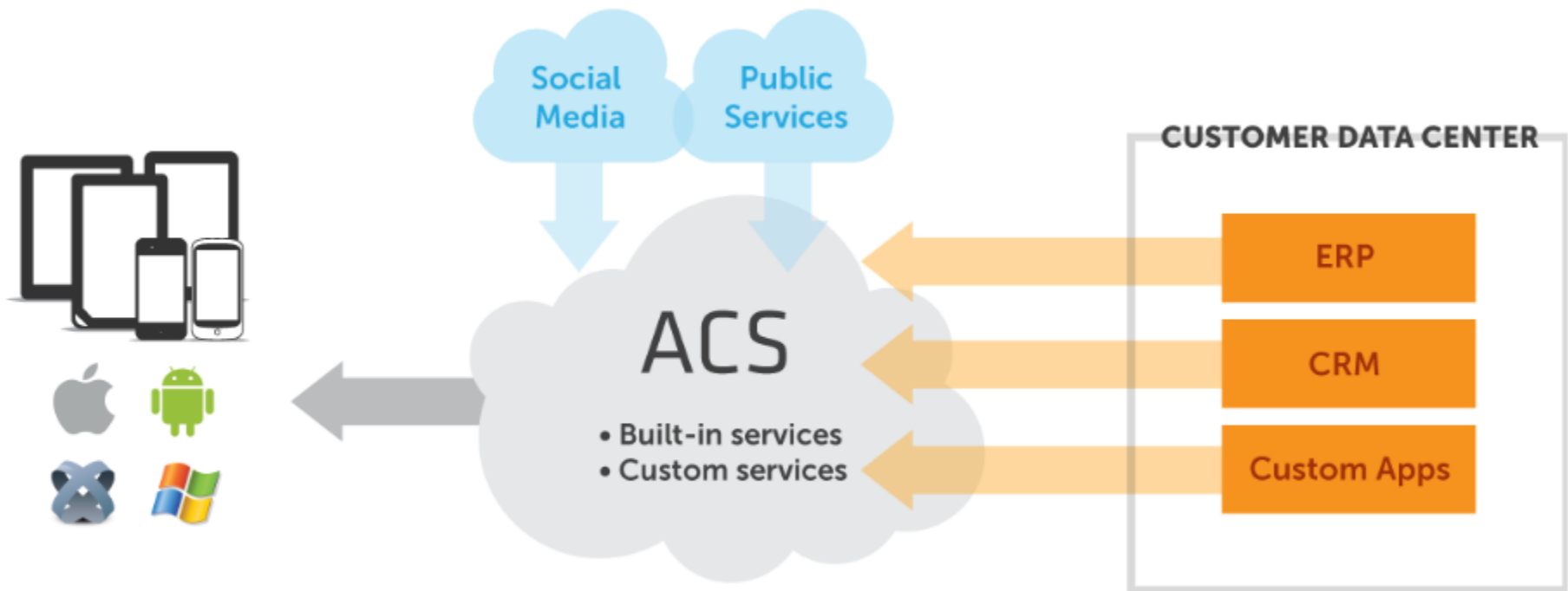


Figure 3: Appcelerator Cloud Services offers a fast, easy way to build connected mobile apps



- Challenges with Internal Server-side Development
 - Development and Infrastructure Costs
 - Time to Market



- MBaaS: A New Approach for Mobile Cloud Services
 - Library of pre-built, commonly used services
 - Client-side APIs for any development platform
 - Extensibility, to build your own custom services for all your mobile apps
 - Elastic scale to support user adoption
 - Flexible delivery models

My Apps / Incident Tracking Alloy-production Production Development

[App Management](#) [Settings](#) [Logs](#) [Push Notifications](#) [Email Templates](#) [Access Control](#) [Monthly Usage](#) [Get Support](#) [Go To Docs](#)

Incident Tracking Alloy-production Details

Status	Running
APP Key	bFmQKtpZj2CxJbCfaDYEIEXHEHYWc5Uf
Show OAuth Credentials	

Export Data[?] [Export Data](#)

Manage Data in Incident Tracking Alloy-production

- [Chats \(0\)](#)
- [Checkins \(0\)](#)
- [Collections \(0\)](#)
- [Events \(0\)](#)
- [Data Files \(0\)](#)
- [Friends](#)

Appcelerator Titanium Overview





Native Apps Using Appcelerator Titanium

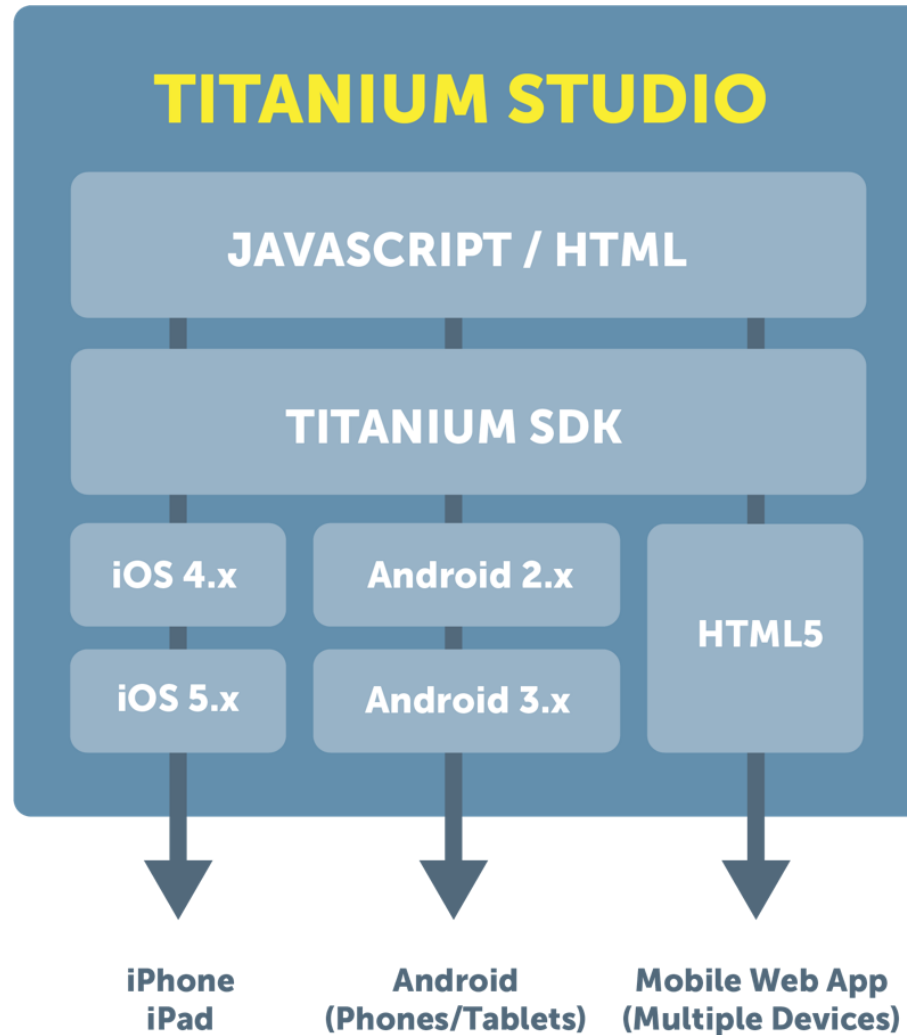


- ☐ Code in Javascript
- ☐ Translate to Native Code
- ☐ Apple iOS
(Mac OSX only)
- ☐ Android
(Mac/Windows/Linux)
- ☐ Blackberry beta
(Windows only)
- ☐ Titanium Studio IDE
(formally Aptana Studio)





Titanium exists as a bridge between the native operating system and your app's code.



Appcelerator Titanium Coding Practice





Objectives



User Interface Fundamental



Properties & Events



Working with Local Data Sources



Working with Remote Data Sources



Integration with Google Maps & GPS



Differences in devices - platforms



Deploying and Distributing



Working with Cloud Service



Using Module



Distributions Methods

User Interface Fundamental





Window and View Object

- Window is most basic object where we can place many UI elements.
- View is a collection of displaying UI elements
- Window can contains views inside, but not child windows.
- We can add more views to a view.
- Layouts: Absolute, vertical, horizontal → Better use of TableView / TableViewRow
- Best practice: Create window → Add views → Open window.

Window and View Object

Syntax:

❑ Create window

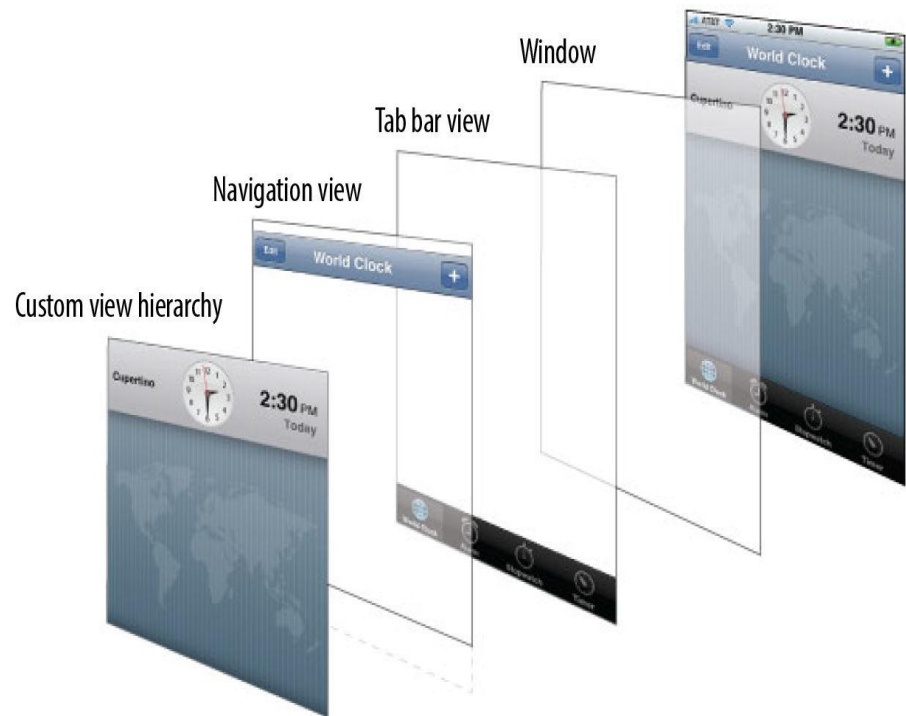
```
w = Ti.UI.createWindow({  
    backgroundColor:"white"  
});
```

❑ Create View

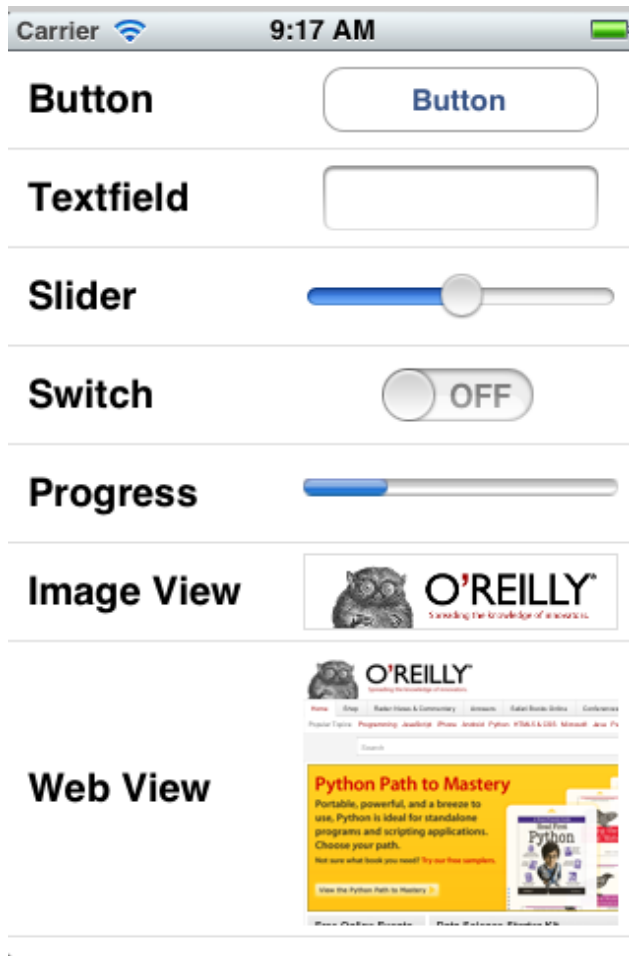
```
v = Ti.UI.createView({  
    //json descriptive  
});
```

❑ Add View & Open window

```
w.add(v);  
w.open();
```

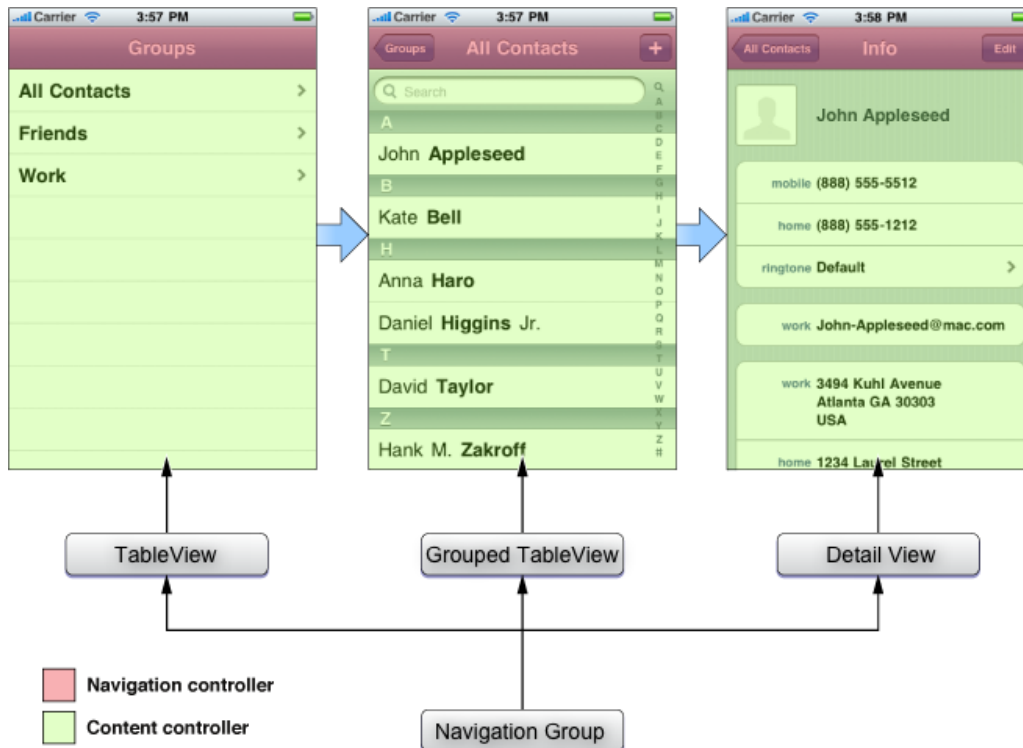


Basic UI Controls

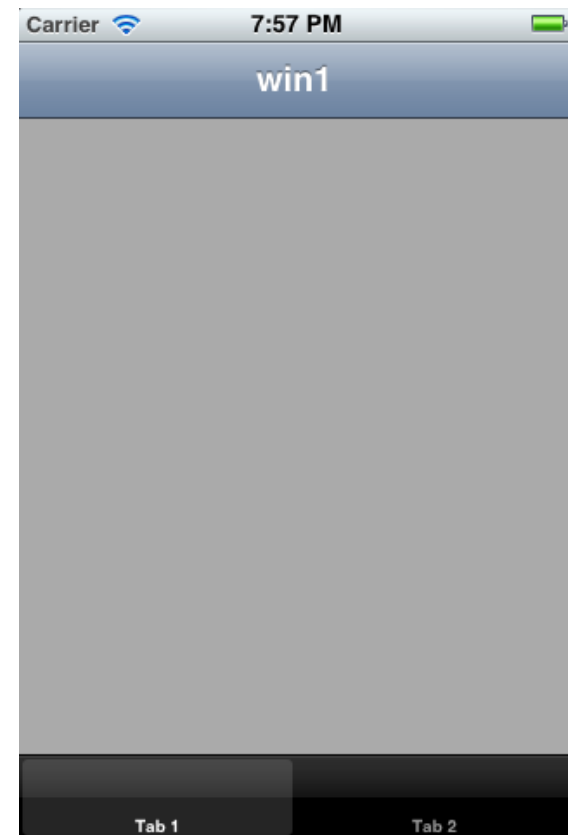


View Controllers

Navigation Group



Tab Group





Practice

Properties & Events



App object and properties

App object

- ❖ App Object represents currently running app that anywhere in the app can access the App.
- ❖ Items stored in App object will persist across application sessions.
- ❖ Data types support: Boolean, Double, Integer, List and String.
- ❖ Best practice: Store simple data (json type) for app preference/settings

App.Properties

- **Store data**

Ti.App.Properties.setX(key, value)

- **Read properties data**

Ti.App.Properties.getX(key), where X: data type

- **Check properties existence**

*Ti.App.Properties.hasProperty(key)
(where X: data type supported)*

Events

- ❖ Most of Titanium is built around concept of event-driven programming
- ❖ Two types of events:
 - Titanium-defined events: like button-click, key-press...
 - Custom events: User self-defines events, subscribes events to controls, fire events and handle the result functions.
- ❖ Titanium almost always passes in an event object as JSON type that gives information about the event.

Syntax

```
addEventListener('event_name', function(e){  
    //event handler function  
});
```

Note:

The parameter e in response function is a JSON object, from which we can get all info about the event.

```
{  
    globalPoint = {  
        x = 172;  
        y = 247;  
    };  
    source = "[object TiUIButton]";  
    type = click;  
    x = 87;  
    y = "19.5";  
}
```

Fire Events

- ❖ You can fire events rather than waiting around for the user of system to initiate them.

```
someButton.fireEvent('click');
```

- ❖ You can pass data (often JSON-serialized data) along with the event as the 2nd parameter of the fireEvent() function.

```
someButton.fireEvent('click', {kitchen: 'sink'});
```

- ❖ Members of passed data can be retrieved via the event object in the listener.

```
someButton.addEventListener('click', function(e){  
    Ti.APP.info('The value of kitchen is '+e.kitchen);  
});
```

Remove Events

```
function doSomething(e) {  
    // do something  
}  
deleteButton.addEventListener('click', doSomething);  
// ... elsewhere in your code ...  
deleteButton.removeEventListener('click', doSomething);  
});
```



Custom Events

- ❖ Custom events promotes loose coupling of our components and makes for more scalable and maintainable Javascript code.
- ❖ Custom events are often used when the database is updated.

Component-Level Event

```
deleteButton.addEventListener('click', function(e){  
    // when something happens in your app  
    database.doDelete(e.whichRecord);  
    // fire an event so components can react  
    theTable.fireEvent('db_updated');  
});  
// ... elsewhere in your code  
theTable.addEventListener('db_updated', function(e){  
    theTable.setData(database.getCurrentRecords());  
});
```

Application-Level Event

```
deleteButton.addEventListener('click', function(e){  
    // when something happens in your app  
    database.doDelete(e.whichRecord);  
    // fire a global event so components can react  
    Ti.App.fireEvent('db_updated');  
});  
// ... elsewhere in your code  
Ti.App.addEventListener('db_updated', function(e){  
    theTable.setData(database.getCurrentRecords());  
    someOtherComponent.doSomethingElse();  
});
```



Practice

Working with Local Data Sources





- ☐ Using SQLite
- ☐ Using Local File System



Using SQLite

- SQLite is light weight relational database mostly wide-used for mobile.
- Use local db to store data when device is offline, or even store data which is required locally.
- Only 5 underlying data types supported: Text, Numeric, Integer, Real, None.
- SQLite supports concurrent read access, but enforces sequential write access.
- Titanium provide access to SQLite via Titanium.Database module.
- Two ways of creating database:
 - Create the javascript function wrapping the SQL that executes the DDL and DML statements.
 - Attach an existing database via Ti.Database.install() method

```
Var db = Titanium.Database.install('data.db', 'packtData');
```




Using SQLite API Calls

- Open(db_name) :

→ create a database if none is present, or simply open the existing db.

```
var db = Titanium.Database.open('mydb');
```

- Execute(sql_statement):

→ run a SQL statement and return the results

```
db.execute('CREATE TABLE IF NOT EXISTS DATABASETEST (ID INTEGER, NAME TEXT)');  
db.execute('DELETE FROM DATABASETEST');  
db.execute('INSERT INTO DATABASETEST (ID, NAME ) VALUES(?,?)',1,'Name 1');
```

- Close(db_name):

→ Close the database and release resource from memory.

- Remove

→ remove the database files for this instance from disk. All data in db will be lost and cannot be reversed.

Using SQLite – Result Sets

- ResultSets is the result of SELECT statement execution.

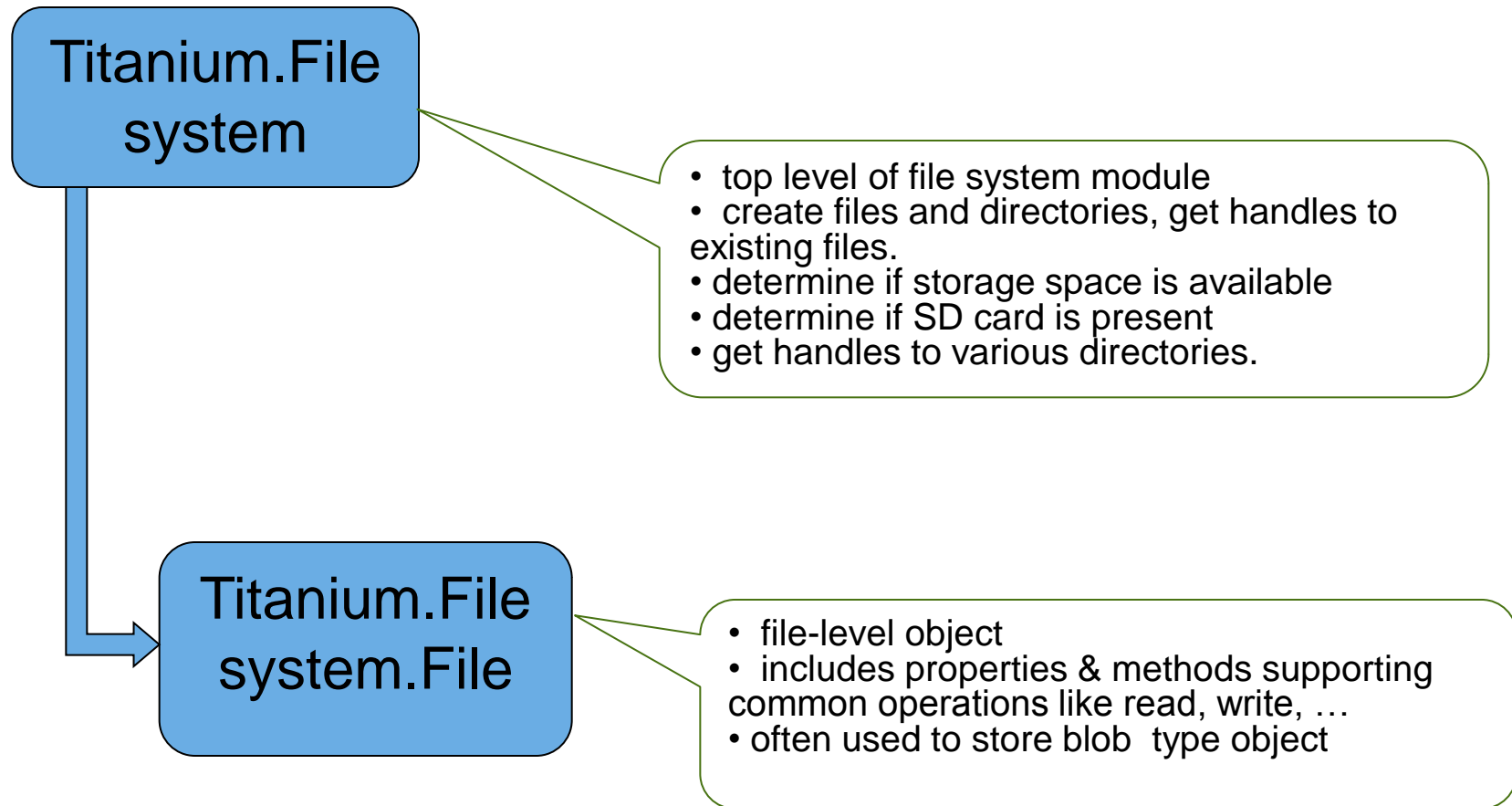
```
var rows = db.execute('SELECT * from table1 order');
```

- With result set, we can loop through each rows to get the information.
 - fieldName:
 - return the value of specified column name in current row.
 - next:
 - return TRUE if there's another row to advance. Return FALSE if end of ResultSet and there are no more rows.

```
var sql = "SELECT * FROM favorites ORDER BY title ASC";
var results = [];
var resultSet = db.execute(sql);
while (resultSet.isValidRow()) {
    results.push({
        id: resultSet.fieldByName('id'),
        title: resultSet.fieldByName('title'),
        description: resultSet.fieldByName('description'),
        link: resultSet.fieldByName('link')
    });

    //iterates to the next record
    resultSet.next();
}
```

Using Local File System



Storage Location (Ti.Filesystem)

applicationDataDirectory

- a read/write directory accessible by your app. The content of this directory persist until you remove the files or uninstall your app.

resourcesDirectory

- A read-only directory where your application resources are located; this directory corresponds to the *project/Resources* directory in Titanium Studio. The contents of this directory persist until the user uninstalls the application.

tempDirectory

- A read-write directory where your application can place temporary files. The contents of this directory persist until your application fully closes, at which time the operating system could delete your files.

externalStorageDirectory

- A read-write directory on the external storage device (SD card) accessible by your app, if such a location exists. Check first with `Ti.Filesystem.isExternalStoragePresent()`



File Operations

- Get file handle

```
var f = Ti.Filesystem.getFile(Ti.Filesystem.applicationDataDirectory, 'yourfile.txt');
```

- Reading

```
var f = Ti.Filesystem.getFile(Ti.Filesystem.applicationDataDirectory, 'yourfile.txt');  
var contents = f.read();  
Ti.API.info('Output as a blob: '+contents); // useful if contents are binary  
Ti.API.info('Output text of the file: '+contents.text);  
Ti.API.info('Output the file\'s MIME type: '+contents.mimeType); // e.g. text/plain
```

- Writing

```
var f = Ti.Filesystem.getFile(Ti.Filesystem.applicationDataDirectory, 'emptyfile.txt');  
f.write('The file is no longer empty!'); // write to the file
```

- Creating

```
var f = Ti.Filesystem.getFile(Ti.Filesystem.applicationDataDirectory, 'nonexistent_file.txt');  
if(f.exists()===false) {  
    // you don't need to do this, but you could...  
    f.createFile();  
}  
f.write('writing to the file would be enough to create it');
```

- Removing

```
var f = Ti.Filesystem.getFile(Ti.Filesystem.applicationDataDirectory, 'delete_me.txt');  
f.write('foo'); // make sure there's content there so we're sure the file exists  
// Before deleting, maybe we could confirm the file exists and is writable  
// but we don't really need to as deleteFile() would just return false if it failed  
if(f.exists() && f.writable) {  
    var success = f.deleteFile();  
    Ti.API.info((success===true) ? 'success' : 'fail'); // outputs 'success'  
}
```

Directory Operations

```
// create our starting file
var f = Ti.Filesystem.getFile(Ti.Filesystem.applicationDataDirectory, 'myfile.txt');
f.write('foo');
// get a handle to the as-yet non-existent directory
var dir = Titanium.Filesystem.getFile(Titanium.Filesystem.applicationDataDirectory, 'mysubdir');
dir.createDirectory(); // this creates the directory
Ti.API.info('Directory list to start: ' + dir.getDirectoryListing()); // it's empty

// let's move myfile.txt to our directory
f.move('mysubdir/myfile.txt');
// output a directory listing
Ti.API.info('Dir list after move: ' + dir.getDirectoryListing());

// delete the directory
if(dir.deleteDirectory()==false) {
    Ti.API.info('You cannot delete a directory containing files');
    dir.deleteDirectory(true); // force a recursive directory, which will delete contents
}
// if we try to list the directory, the output is null because the directory doesn't exist
Ti.API.info('Dir list after deleteDirectory(): ' + dir.getDirectoryListing());
```



Practice

Working with Remote data source



HttpClient and Request Lifecycle

- Titanium can interact with remote servers over Http using Ti.Network.HTTPClient
- It's possible to use HttpClient to interact with many types of web services, especially REST-style ws. Commonly, the mobile app interact data using Http GET or POST.
- Handle response in onload callback function, in which:
 - this.responseText: hold returned data as raw text
 - this.responseXML: hold returned data as XML document instance
 - this.responseData: hold returned data as BLOB (binary data)

```
1 var url = "https://www.appcelerator.com";
2 var xhr = Ti.Network.createHttpClient({
3   onload: function(e) {
4     // this function is called when data is returned from the server and available for use
5     // this.responseText holds the raw text return of the message (used for text/JSON)
6     // this.responseXML holds any returned XML (including SOAP)
7     // this.responseData holds any returned binary data
8     Ti.API.debug(this.responseText);
9     alert('success');
10  },
11  onerror: function(e) {
12    // this function is called when an error occurs, including a timeout
13    Ti.API.debug(e.error);
14    alert('error');
15  },
16  timeout:5000 /* in milliseconds */
17 });
18 xhr.open("GET", url);
19 xhr.send(); // request is actually sent with this statement
20
```

XHR Lifecycle – onreadystatechange() callback

UNSENT
(0):

- The object has been constructed.

OPENED
(1):

- The open() method has been successfully invoked.

HEADERS
RECEIVED
(2):

- All Http headers of final response have been received.

LOADING
(3):

- The response entity body is being received

DONE (4):

- The data transfer has been completed or error happened.

```
xhr.onreadystatechange = function(e) {  
    switch(this.readyState) {  
        case 0:  
            // after XMLHttpRequest declared, prior to open()  
            // though Ti won't actually report on this readyState  
            Ti.API.info('case 0, readyState = ' + this.readyState);  
            break;  
        case 1:  
            // open() has been called, now is the time to set headers  
            Ti.API.info('case 1, readyState = ' + this.readyState);  
            break;  
        case 2:  
            // headers received, xhr.status should be available now  
            Ti.API.info('case 2, readyState = ' + this.readyState);  
            break;  
        case 3:  
            // data is being received,  
            // onreadystatechange being called now  
            Ti.API.info('case 3, readyState = ' + this.readyState);  
            break;  
        case 4:  
            // done, onload or onerror should be called now  
            Ti.API.info('case 4, readyState = ' + this.readyState);  
            break;  
    }  
}
```



Working with JSON data

Receiving & parsing Json data

```
var url = "http://example.com/json.txt";
var json;

var xhr = Ti.Network.createHTTPClient({
  onload: function() {
    // parse the retrieved data, turning it into a JavaScript object
    json = JSON.parse(this.responseText);
    // ...
  }
});
```

Sending Json data

```
var blogPost = {
  title: 'My awesome blog',
  body: 'Today I met Susy at the laundromat. Best day EVAR\!'
};

var xhr = Ti.Network.createHTTPClient({
  onload: function() {
    // handle the response
  }
});

xhr.open('POST', 'http://www.myblog.com/post.php');
// optional:
// blogPost = JSON.stringify(blogPost);
xhr.send(blogPost);
```



Working with XML data

Titanium automatically serialize the responseText into XML for use

```
xhr.onload = function(e) {  
    var doc = this.responseXML.documentElement;  
    //this is the XML document object  
  
    //Use the DOM API to parse the document  
    var elements = doc.getElementsByTagName("someTag");  
};
```

Parsing XML:

- `getElementByTagName`: returns array of node with given name
- `item()`: select specific node in array
- `getAttribute()`: retrieve value of attribute with given name
- `text/nodeValue`: retrieve leaf node values associated with the node

```
// begin looping through blog posts  
var data = [];  
// blog posts are in nodes named "item"  
var items = xml.documentElement.getElementsByTagName("item");  
for (var i=0;i<items.length;i++) {  
    data.push({  
        postTitle: items.item(i).getElementsByTagName("title").item(0).text,  
        postLink: items.item(i).getElementsByTagName("link").item(0).text  
    });  
}
```



Practice

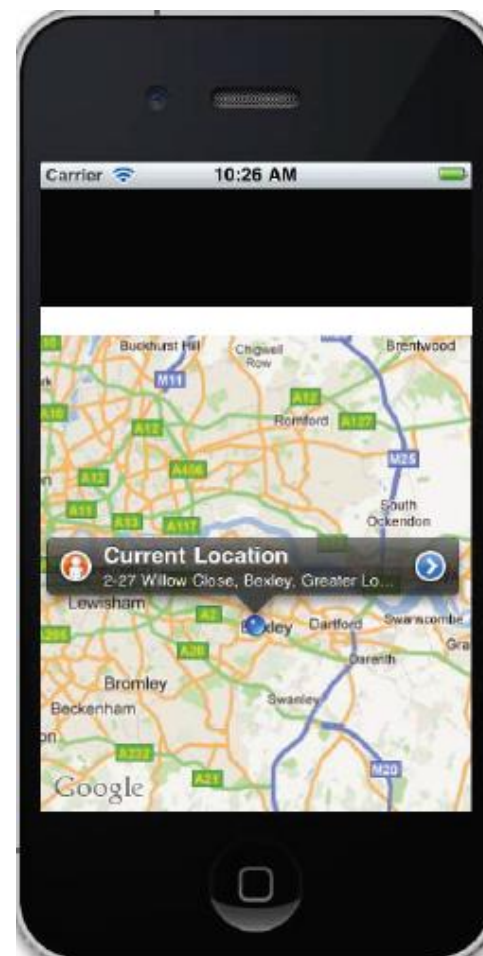
Integration with Google Maps & GPS



Objective

Using Titanium.Map API

- ☐ Adding a MapView to application
- ☐ Working with Geolocation
- ☐ Convert address to latitude & longitude
- ☐ Adding annotations to MapView
- ☐ Drawing route in MapView





Adding Mapview

- MapView is core UI of Ti.Map module, used to display native maps via (lat,lang) coordination.
- mapType:
 - *Ti.Map.STANDARD_TYPE*
 - *Ti.Map.SATELITE_TYPE*
 - *Ti.Map.HYBRID_TYPE*
- Region(Json type): specify the coordination & zoom level.
- Animate(boolean): indicate map loading animation
- regionFit(boolean): fit in visible view
- userLocation(boolean): indicate if the map show a pin as of current user position.

```
var win = Ti.UI.createWindow();
var mapview = Titanium.Map.createView({
  mapType : Titanium.Map.STANDARD_TYPE,
  region : {
    latitude : 37.389569,
    longitude : -122.050212,
    latitudeDelta : 0.1,
    longitudeDelta : 0.1
  },
  animate : true,
  regionFit : true,
  userLocation : false
});
win.add(mapview);
win.open();
```




MapView Events Handling

- Loading: fired when the MapView has started querying for new data & rendering the map.
- Complete: fired when the MapView has completed querying and rendering.
- RegionChanged: fired when the visible region has changed, like change the zoom level, the coordination, scrolling...
- Error: handle the error encountered by native map.

```
mapview.addEventListener('complete', function(e) {  
    Ti.API.info('complete');  
    Ti.API.info(e);  
});  
mapview.addEventListener('error', function(e) {  
    Ti.API.info('error');  
    Ti.API.info(e);  
});  
mapview.addEventListener('loading', function(e) {  
    Ti.API.info('loading');  
    Ti.API.info(e);  
});  
mapview.addEventListener('regionChanged', function(e) {  
    Ti.API.info('regionChanged');  
    Ti.API.info(e);  
});
```



Geolocation

- Using `getCurrentLocation()` method of `Ti.Geolocation` to get information of current position.
- All info is required via the `coords` property of event object (`e`).
- `Geolocation.distanceFilter`
 - determine how accurate (meters) you want the GPS location to be.
- `RegionChanged`: fired when the visible region has changed, like change the zoom level, the coordination, scrolling...
- Note: Geolocation probably not function in emulator.

```
//set the distance filter
Titanium.Geolocation.distanceFilter = 10;
//apple requires this parameter so it can inform the user
//of why you are accessing their location data
Ti.Geolocation.purpose = "For tracking distance .";

Titanium.Geolocation.getCurrentPosition(function(e) {
    if (e.error) {
        //if mapping location doesn't work, show an alert
        alert('Sorry, not available on your device!');
        return;
    }
    //get the properties from Titanium.GeoLocation
    var longitude = e.coords.longitude;
    var latitude = e.coords.latitude;
    var altitude = e.coords.altitude;
    var heading = e.coords.heading;
    var accuracy = e.coords.accuracy;
    var speed = e.coords.speed;
    var timestamp = e.coords.timestamp;
    var altitudeAccuracy = e.coords.altitudeAccuracy;
    //apply the lat and lon properties to our mapview
    mapview.region = {
        latitude : latitude,
        longitude : longitude,
        latitudeDelta : 0.5,
        longitudeDelta : 0.5
    };
});
```

Geolocation

```
Ti.Geolocation.forwardGeocoder('Hanoi', function(e) {  
    Ti.API.info('Hanoi latitude: ' + e.latitude);  
    Ti.API.info('Hanoi longitude: ' + e.longitude);  
});
```

- Using forwardGeocoder() method of Ti.Geolocation to convert a given address to <lat, lang> coordination.
- forwardGeocoder() utilize the Google map Geocoding API.

Adding Annotation

```
var annotations = [Ti.Map.createAnnotation({  
  latitude : 37.389569,  
  longitude : -122.050212,  
  title : 'Appcelerator HQ',  
  subtitle : 'Mountain View, CA',  
  animate : true,  
  pincolor : Ti.Map.ANNOTATION_GREEN,  
  leftButton : 'appcelerator.gif'  
}),  
Ti.Map.createAnnotation({  
  latitude : 37.422502,  
  longitude : -122.0855498,  
  title : 'Google HQ',  
  subtitle : 'Mountain View, CA',  
  animate : true,  
  image : 'google.png'  
})];  
var mapview = Titanium.Map.createView({  
  mapType : Titanium.Map.STANDARD_TYPE,  
  region : {  
    latitude : 37.389569,  
    longitude : -122.050212,  
    latitudeDelta : .05,  
    longitudeDelta : .05  
  },  
  animate : true,  
  regionFit : true,  
  userLocation : false,  
  annotations : annotations  
});
```





Adding Annotation

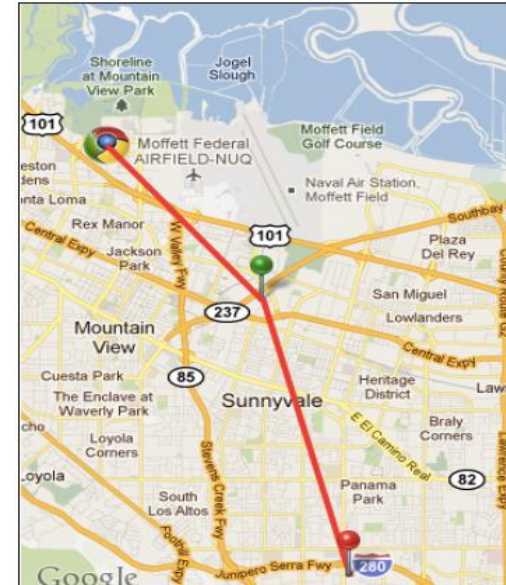
❑ Annotations: contains array of items which properties specify the information of the display markers.

❑ Some of properties:

- ✓ `animate` (boolean): indicate whether an annotation is animated
- ✓ `image`: the image url in place of default pin
- ✓ `leftButton/rightButton`: button appeared on left/right side of annotation when clicked.
- ✓ `leftView, rightView`: the Titanium View object will appear on left/right side of annoation when clicked.
- ✓ `pinColor`: can be `Ti.Map.ANNOTATION_GREEN`, `Ti.Map.ANNOTATION_PURPLE` or `Ti.Map.ANNOTATION_RED`
- ✓ `title`: main text displayed on annotation when clicked.

Adding Routes

- ❑ Routes are an iOS-only feature.
- ❑ Routes can be driving path, bike path, or straight path.
- ❑ Routes are added to MapView via `Ti.Map.MapView.addRoute()` function, where we pass the structured route object.
- ❑ Routes can be removed via `Ti.Map.MapView.removeRoute()`.



```
{
  name: 'route name',
  color: '#f00',
  width: 4,
  points: [
    {latitude:50, longitude:50},
    ...
  ]
}
```

```
mapview.addRoute({
  name : 'myroute',
  width : 4,
  color : '#f00',
  points : [{
    latitude : 37.422502,
    longitude : -122.0855498
  }, {
    latitude : 37.389569,
    longitude : -122.050212
  }, {
    latitude : 37.331689,
    longitude : -122.030731
  }
]
});
```



Practice

Differences In Devices & Platforms



Get Device Information

Make use of Titanium.Platform module to get device-specific information like OS, model, battery...

- ✓ `Ti.Platform.osname` → get OS info
- ✓ `Ti.Platform.batteryLevel` → get battery info
- ✓ `Ti.Platform.availableMemory` → get available memory
- ✓ `Ti.Platform.locale` → get Locality format
- ✓ `Ti.Platform.model` → get Model
- ✓ `Ti.Platform.architecture` → get architecture
- ✓ `Ti.Platform.displayCaps.platformWidth` → get device's window width
- ✓ `Ti.Platform.displayCaps.platformHeight` → get device's window height

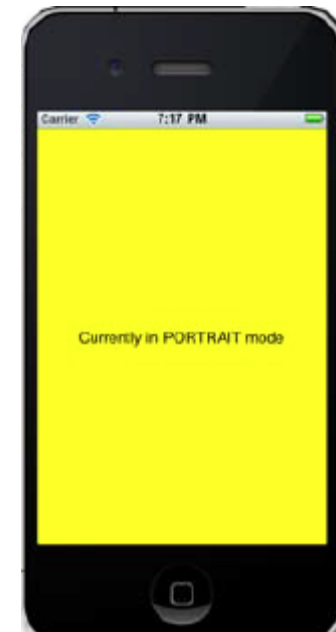


Orientation Mode

Handle responsive layout by attaching 'orientationchange' event into Titanium.Gesture



```
Ti.Gesture.addEventListener('orientationchange', function(e) {  
    //check for landscape modes  
    if (e.orientation == Titanium.UI.LANDSCAPE_LEFT ||  
        e.orientation == Titanium.UI.LANDSCAPE_RIGHT) {  
        view1.width = Titanium.Platform.displayCaps.platformWidth;  
        view1.height = Titanium.Platform.displayCaps.platformHeight;  
        labelOrientation.text = 'Currently in LANDSCAPE mode';  
        view1.backgroundColor = 'Blue';  
    } else {  
        //we must be in portrait mode!  
        view1.width = Titanium.Platform.displayCaps.platformWidth;  
        view1.height = Titanium.Platform.displayCaps.platformHeight;  
        labelOrientation.text = 'Currently in PORTRAIT mode';  
        view1.backgroundColor = 'Yellow';  
    }  
});
```



Deploying & Distributing



Deploying to Android device

Configure an Android device

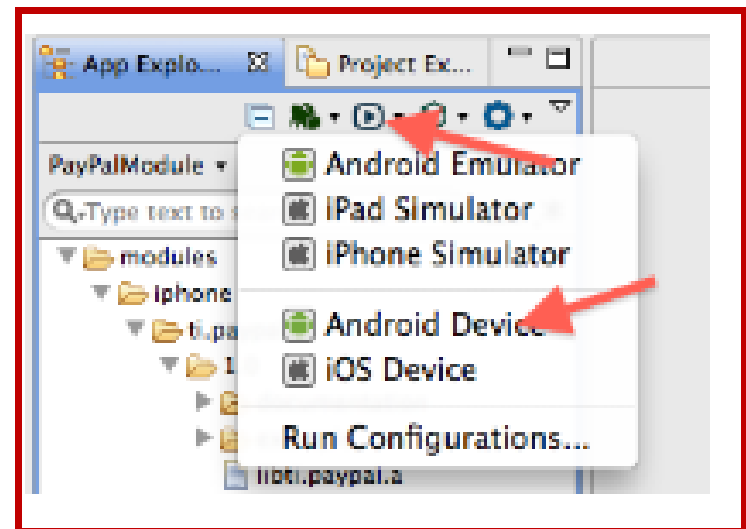


Deploying via adb command

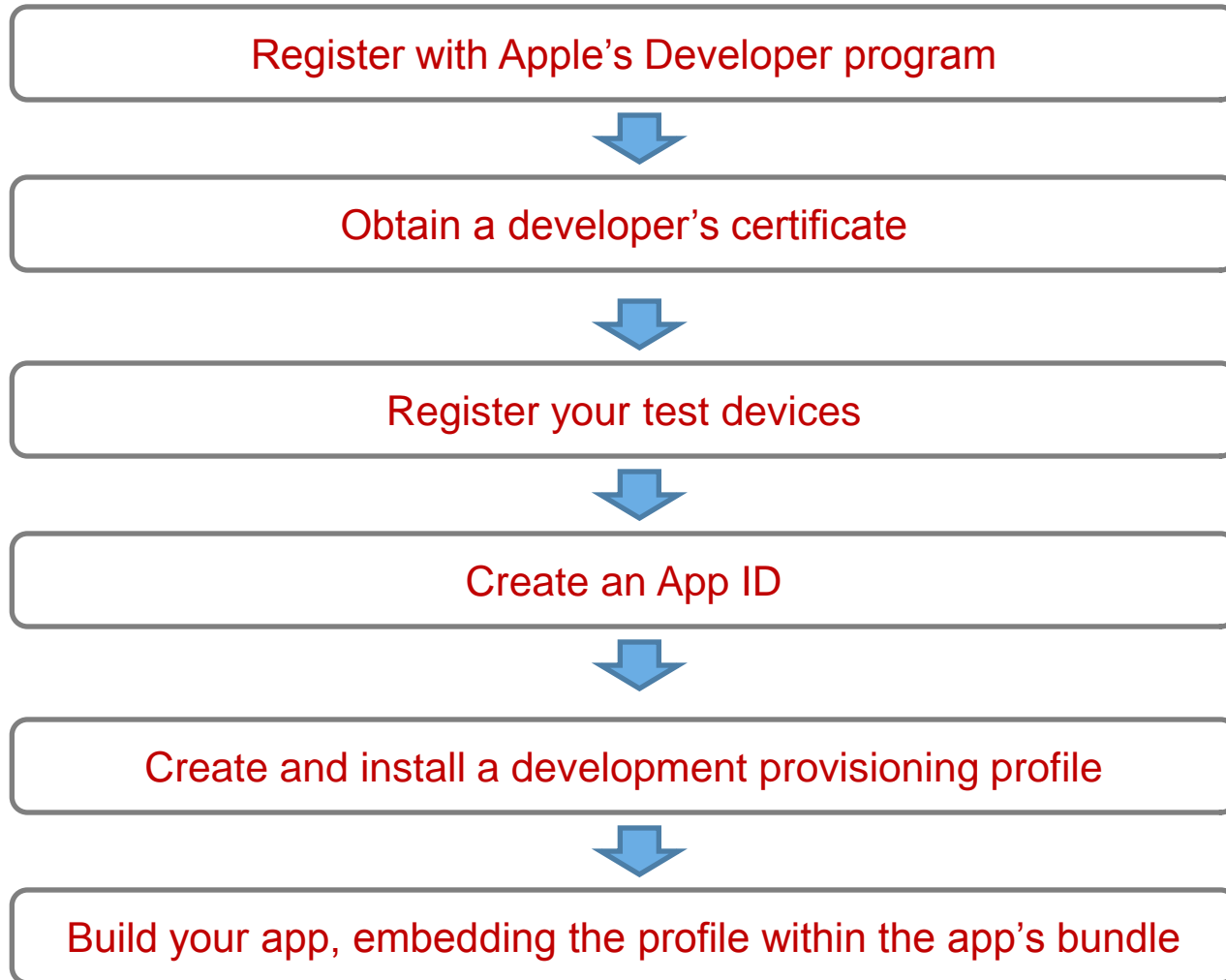
```
adb -d install your_project/build/android/bin/app.apk
```



Deploying via Studio



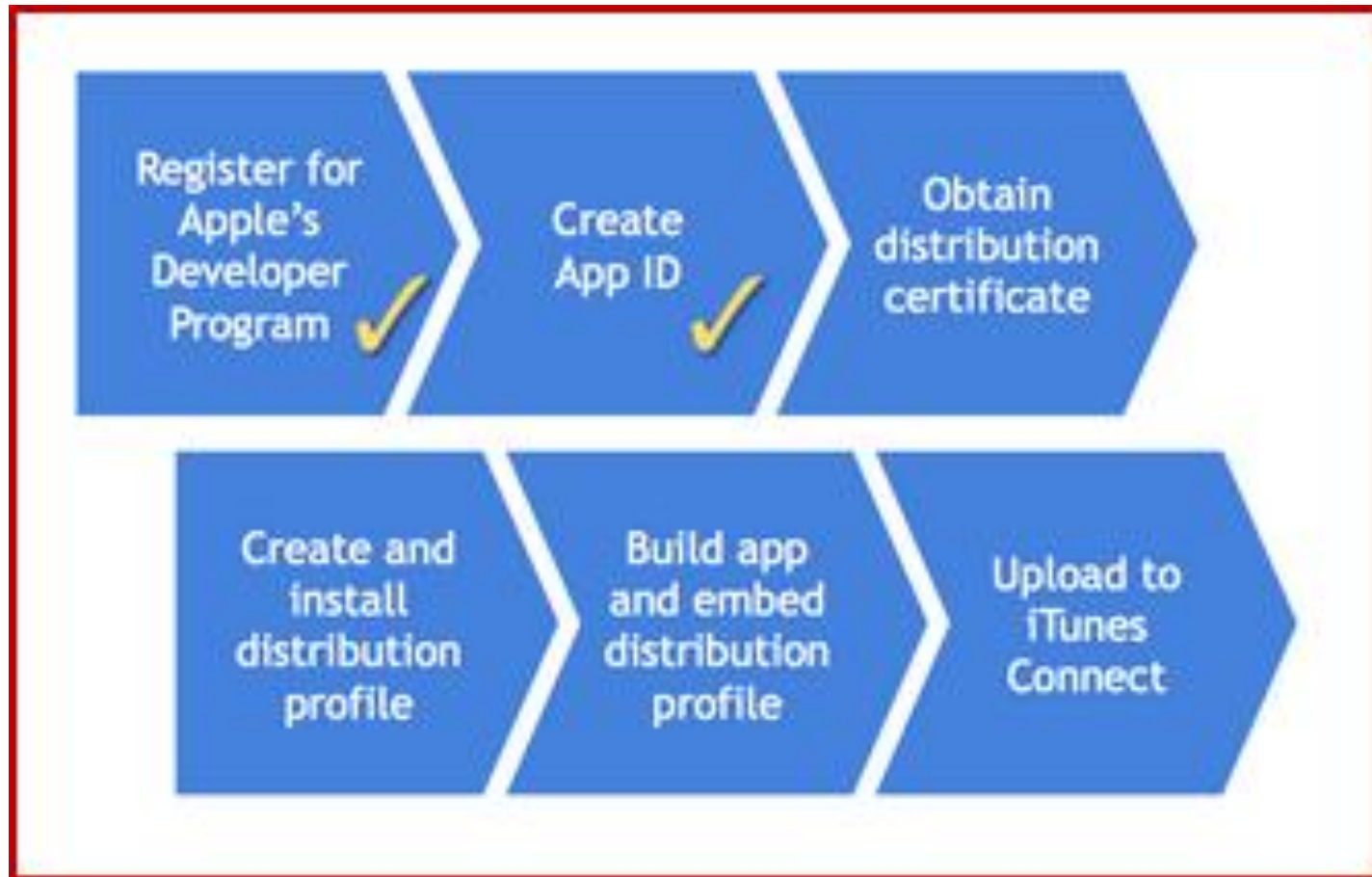
Deploying to iOS device



Distributing an Android application



Distributing an iOS application



Working with Cloud Service





What can we do with ACS?

- Titanium Cloud Services is a Mobile Backend as a Service (MBaaS), offering a fast and easy way to build connected mobile apps.
- Appcelerator provides around 15+ cloud services

Standard:

Users

Files

Photos

Custom Objects

Emails

Push Notification

Advanced:

Chats

Checkins

Events

Friends

Places

Posts

Reviews

Messages

Social integration

ACS Quick Guide: How to create cloud app?

- Create new project with cloud-enabled option

New Mobile Project
Create a new mobile project

Project Template **Project Location**

Project name: cloudTest

☒ Use default location

Location: E:\Mobility\Appcelerator\Workspace\cloudTest [Browse...](#)

Project Settings

App Id: com.duynt.cloudtest

Company/Personal URL: http://

Titanium SDK Version: 3.1.1.GA

Deployment Targets: ☒ Android ☐ BlackBerry ☒ Mobile Web ☐ Tizen

[Set-up/Configure SDKs](#)

Cloud Settings

Titanium platform provides services to cloud-enable this application. This provides a wide array of network features and data objects for your app. [Learn](#)

☒ Cloud-enable this application

? < Back Next > Finish Cancel



ACS Quick Guide: How to create cloud app?

- Two set of cloud API key found:
 - Development keys: used for simulator or device
 - Production keys: used for creating signed package

```
<?xml version="1.0" encoding="UTF-8"?>
<ti:app xmlns:ti="http://ti.appcelerator.org">
  <property name="acs-oauth-secret-production" type="string">qTeELMDXXnFh812os8P9YWr3VnCnAGCE</property>
  <property name="acs-oauth-key-production" type="string">oihXG8KX7sbkLgAxjGvd7mYSnU0SuAzc</property>
  <property name="acs-api-key-production" type="string">pbd8XaQ0XoohW3wfehCuhNxmOYFkiDaH</property>
  <property name="acs-oauth-secret-development" type="string">SFWDnboiyvZReoQFsPVCza7pq0DXdKrq</property>
  <property name="acs-oauth-key-development" type="string">vxpxQwN4U4bp0A09VLD21k1d9WkYodPe</property>
  <property name="acs-api-key-development" type="string">ZmE4QzoE9ozE487WZmkZSIYcMglPiVe0</property>
  <id>com.duynt.cloudtest</id>
  <name>cloudTest</name>
  <version>1.0</version>
```

- Access Appcelerator cloud console with my.appcelerator.com/apps

My Apps / cloudTest

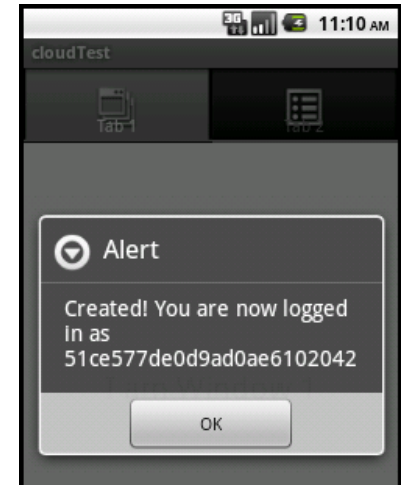
The screenshot shows the Appcelerator cloud console interface for an app named 'cloudTest'. At the top, there is a blue link 'back to my apps'. Below this, there are two main sections: 'Analytics' and 'Cloud'. The 'Analytics' section has a green bar chart icon and the text 'Analytics are not being gathered for this app.' The 'Cloud' section has a blue cloud icon, the word 'Cloud', and a progress bar labeled 'email' with the value '0 of 100,000'.



ACS Quick Guide: Create Cloud User

- Include Ti.Cloud module and use Cloud.Users.create()

```
1 var Cloud = require('ti.cloud');
2   Cloud.Users.create({
3     username: "testuser1",
4     password: "testuser1",
5     password_confirmation: "testuser1",
6     first_name: "test",
7     last_name: "user1"
8   }, function (e) {
9     if (e.success) {
10       var user = e.users[0];
11       alert('Created! You are now logged in as ' + user.id);
12     }
13     else {
14       if (e.error && e.message) {
15         alert('Error : ' + e.message);
16       }
17     }
18   });
19
20
```



- Click Manage → Development → App Management → Users to check for Cloud

My Apps / cloudTest-development Production Development

[Back to App Management](#) [Get Support](#) [Go To Docs](#)

All Users Admin Users

Users in cloudTest-development Set Filter Create a User

✓	Users ID	email	username	Updated At	Edit/Delete
	51ce577de0d9ad0ae6102042		testuser1	2013-06-29T03:41:49+0000	

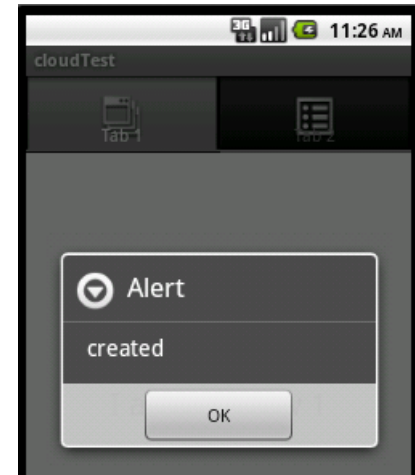
Total 1 result was found. Per page: 10







ACS Quick Guide: Create Cloud Custom Objects

- Use Cloud.Objects.create()

```
1 Cloud.Users.login({
2   login : 'testuser1',
3   password: 'testuser1',
4 }, function(e) {
5   if (e.success) {
6     Cloud.Objects.create({
7       classname : 'books',
8       fields : {
9         book_id : 1,
10        title : 'Javascript',
11        author : 'Peter'
12      },
13      function(e) {
14        if(e.success) {
15          alert("created");
16        } else {
17          alert('Error: ' + ((e.error && e.message) || JSON.stringify(e)));
18        }
19      }
20    });
21  } else {
22    alert('Login Error:' + ((e.error && e.message) || JSON.stringify(e)));
23  }
24 }
25 });
```



- Click Manage → Development → App Management → Custom Objects to check for Cloud Custom objects

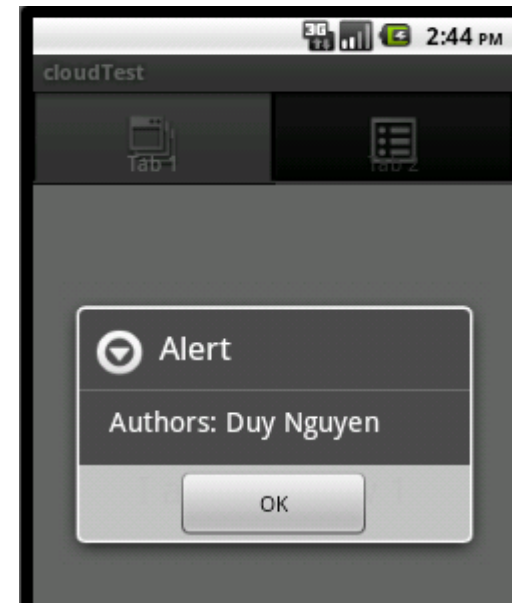
Custom Object in cloudTest-development					Create Custom Object	Set Filter
books (1)	ID	book_id	title	Updated At	Tools	
	51ce6327e0d9ad0adc1046fb	1	Javascript	2013-06-29T04:31:35+0000	  	
Total 1 result was found.					Per page: 10 	



ACS Quick Guide: Show Custom Objects

- Use Cloud.Objects.show()




```
var Cloud = require('ti.cloud');  
  
Cloud.Objects.show({  
  classname : 'books',  
  id : '51ce6327e0d9ad0adc1046fb'  
}, function(e) {  
  if (e.success) {  
    var s = 'Authors: ';  
    for (var i = 0; i < e.books.length; i++) {  
      var b = e.books[i];  
      s += b.author;  
    }  
    alert(s);  
  } else {  
    alert('Error:\n' + ((e.error && e.message) || JSON.stringify(e)));  
  }  
});
```



- Click Manage → Development → App Management → Custom Objects to check for Cloud Custom objects

Custom Object in cloudTest-development

Create Custom Object Set Filter

books (1)	ID	book_id	title	Updated At	Tools
	51ce6327e0d9ad0adc1046fb	1	Javascript	2013-06-29T04:31:35+0000	  

Total 1 result was found.

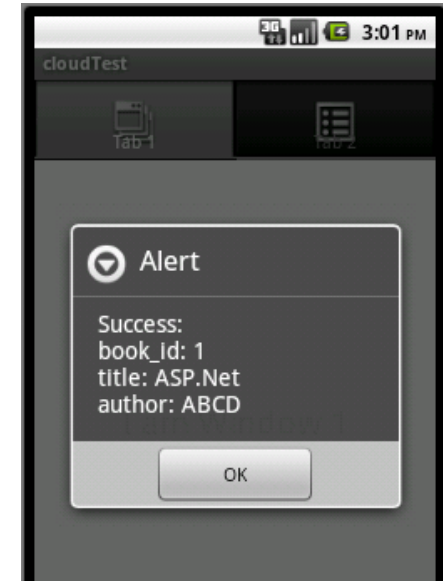
Per page: 10



ACS Quick Guide: Update Custom Objects

- Use Cloud.Objects.update()

```
Cloud.Objects.update({
  classname : 'books',
  id : '51ce6327e0d9ad0adc1046fb',
  fields : {
    title : 'ASP.Net',
    author : 'ABCD'
  }
}, function(e) {
  if (e.success) {
    var book = e.books[0];
    alert('Success:\n' +
      'book_id: ' + book.book_id + '\n' +
      'title: ' + book.title + '\n' +
      'author: ' + book.author);
  } else {
    alert('Error:\n' + ((e.error && e.message) || JSON.stringify(e)));
  }
});
```



- Click Manage → Development → App Management → Custom Objects to check for Cloud Custom objects

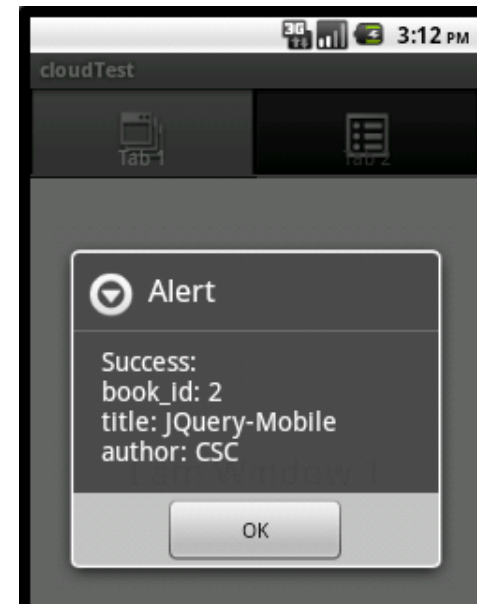
Custom Object in cloudTest-development					Create Custom Object	Set Filter
books (2)	ID	book_id	title	Updated At	Tools	
	51ce6327e0d9ad0adc1046fb	1	ASP.Net	2013-06-29T08:01:35+0000	  	
	book_id	1				
	title	"ASP.Net"				
	author	"ABCD"				
	id	51ce6327e0d9ad0adc1046fb				









ACS Quick Guide: Query Custom Objects

- Use Cloud.Objects.query()

```
Cloud.Objects.query({
  classname : 'books',
  page : 1,
  per_page: 10,
  where: {
    book_id: 2
  }
}, function(e) {
  if (e.success) {
    var book = e.books[0];
    alert('Success:\n' +
      'book_id: ' + book.book_id + '\n' +
      'title: ' + book.title + '\n' +
      'author: ' + book.author);
  } else {
    alert('Error:\n' + ((e.error && e.message) || JSON.stringify(e)));
  }
});
```



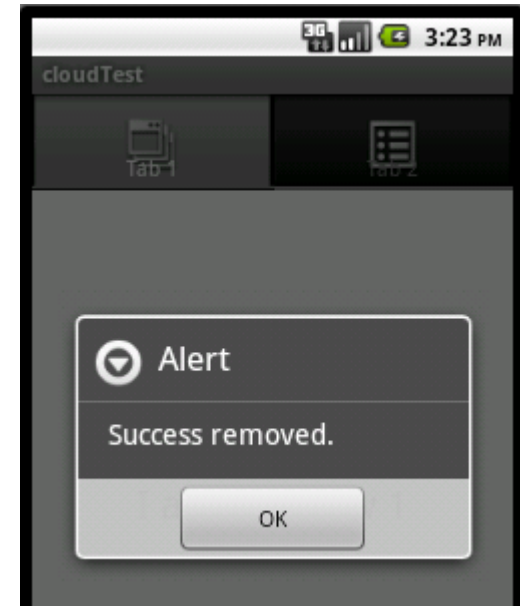
- Click Manage → Development → App Management → Custom Objects to check for Cloud Custom objects

Custom Object in cloudTest-development						Create Custom Object	Set Filter
books (2)	ID	book_id	title	Updated At	Tools		
	51ce6327e0d9ad0adc1046fb	1	ASP.Net	2013-06-29T08:01:35+0000	  		
	51ce840f0cfccd0b660fb7de	2	JQuery-Mobile	2013-06-29T06:51:59+0000	  		
Total 2 results was found.						Per page:	10 ▾




ACS Quick Guide: Remove Custom Objects

- Use `Cloud.Objects.remove()`

```
Cloud.Objects.remove({
  classname: 'books',
  id: '51ce840f0cfccd0b660fb7de'
}, function (e) {
  if (e.success) {
    alert('Success removed.');
```



- Click Manage → Development → App Management → Custom Objects to check for Cloud Custom objects

Custom Object in cloudTest-development						Create Custom Object	Set Filter
books (1)	ID	book_id	title	Updated At	Tools		
	51ce6327e0d9ad0adc1046fb	1	ASP.Net	2013-06-29T08:01:35+0000	  		
Total 1 result was found.						Per page:	10 ▾



Practice

THANK YOU





BUSINESS SOLUTIONS
TECHNOLOGY
OUTSOURCING