Introduction to cross-platform mobile development with Appcelerator Titanium

Clément Guérin

Licence Professionnelle Création Multimédia
March 6, 2012

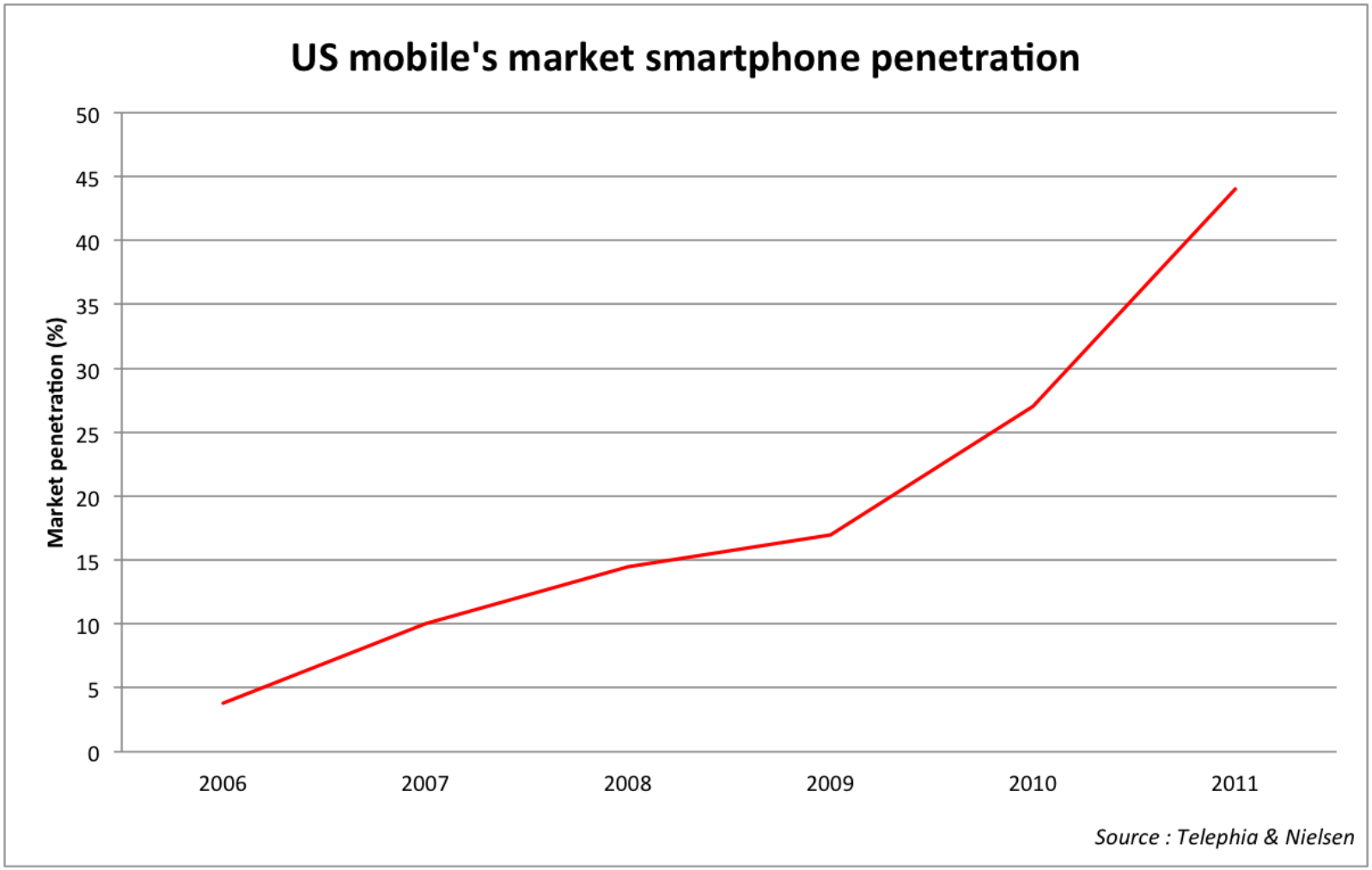
Outline

Timeline

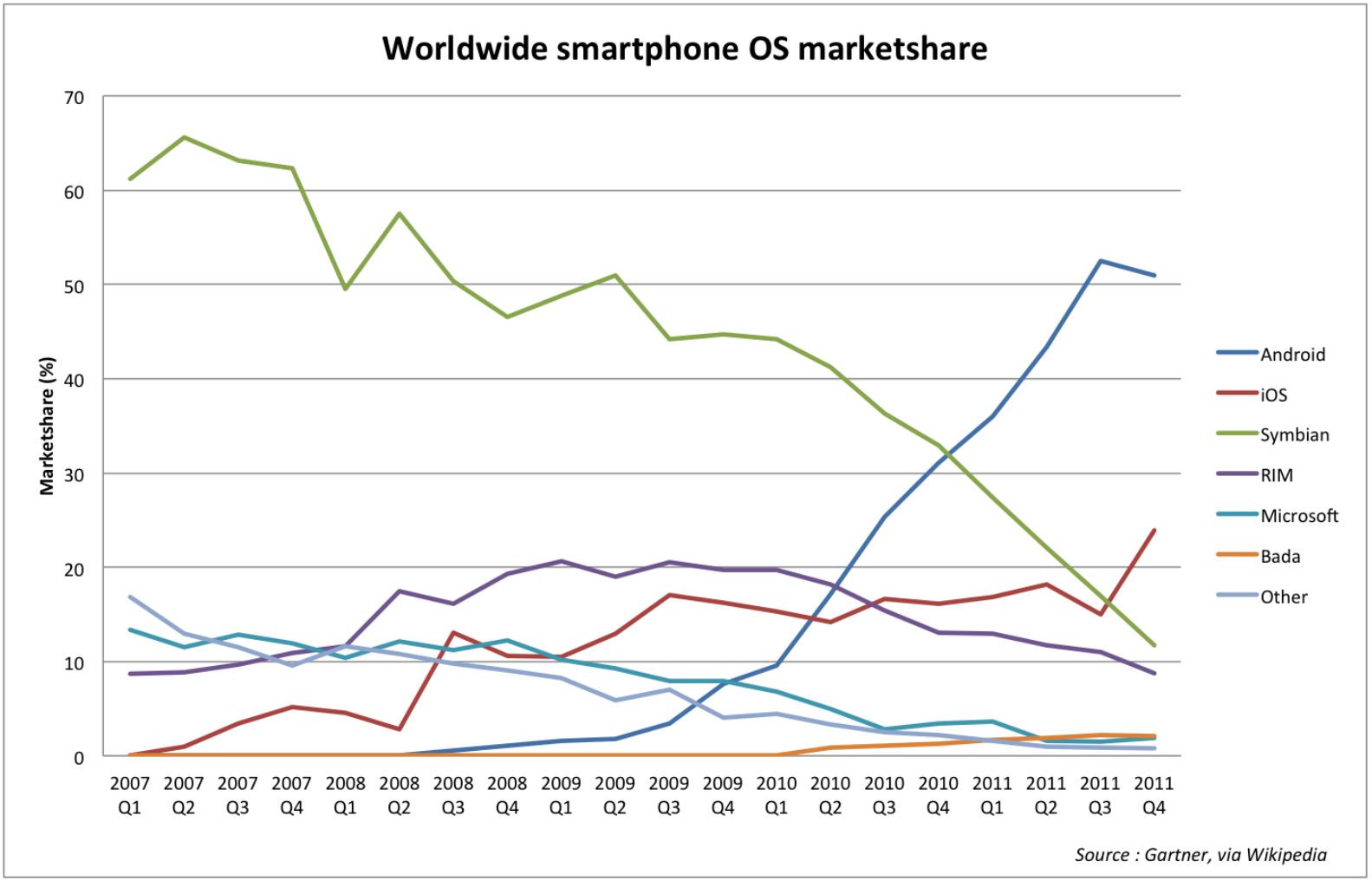
- 1993
  - IBM Simon's release, first smartphone
- 2000
  - Ericsson R380's release running Symbian
- 2002
  - Release of the first mobile devices running Windows CE
  - First BlackBerry release
- 2007
  - iPhone's release, running iOS 1.0
- 2008
  - HTC Dream's release, running Android 1.0
  - App Store and Android Market creation
- 2009
  - Palm Pre's release, running webOS
  - Samsung S8500 's release, running Bada
- 2010
  - First Windows Phones' release
- 2011
  - Over 30 billions apps downloaded

Source : Wikipedia & blog.shoutem.com

Mobile market penetration



Mobile OS marketshare



Some facts

- Android and iOS share more than 75% of the market

Some facts

- Android and iOS share more than 75% of the market
- Smartphones worldwide sales exceeded PCs for the first time in 2011 (490 millions for 415 millions)[1]

[1] *Source: Forbes*

Some facts

- Android and iOS share more than 75% of the market
- Smartphones worldwide sales exceeded PCs for the first time in 2011 (490 millions for 415 millions)[1]
- You can't ignore the smartphone market

[1] *Source: Forbes*

First obvious questions

- Which platform to choose?

First obvious questions

- Which platform to choose?
- I don't know, and don't want to learn, Objective-C, Java, C++, C#...
  Am I doomed to build a web app?

Solutions

- Cross-platform developing tools are a promising solution

	Titanium	PhoneGap	Rho Mobile	Sencha	MoSync
Platforms					
Languages	HTML, CSS, Javascript	HTML, CSS, Javascript	HTML, Ruby	HTML, CSS, Javascript	HTML, C++, C, Javascript
Native apps	Yes	No	Yes	No	No
IDE	Yes	No	Yes	No	Yes
Debugger	Yes	No	Yes	No	Yes

Solutions

- Cross-platform developing tools are a promising solution
- The panel of available frameworks is growing and evolving every day

	Titanium	PhoneGap	Rho Mobile	Sencha	MoSync
Platforms					
Languages	HTML, CSS, Javascript	HTML, CSS, Javascript	HTML, Ruby	HTML, CSS, Javascript	HTML, C++, C, Javascript
Native apps	Yes	No	Yes	No	No
IDE	Yes	No	Yes	No	Yes
Debugger	Yes	No	Yes	No	Yes

Titanium

Basically

Appcelerator Titanium is an open source framework for building native desktop and mobile applications using open web technologies (HTML, CSS and Javascript)

Titanium

Basically

Appcelerator Titanium is an open source framework for building native desktop and mobile applications using open web technologies (HTML, CSS and Javascript)

Platforms

- Desktop: Windows, Mac OS X, Linux
- Mobile: iOS, Android, BlackBerry (beta) and mobile web app

Titanium

Basically

Appcelerator Titanium is an open source framework for building native desktop and mobile applications using open web technologies (HTML, CSS and Javascript)

Platforms

- Desktop: Windows, Mac OS X, Linux
- Mobile: iOS, Android, BlackBerry (beta) and mobile web app

Licence

- Open source (Apache 2.0), free SDK
- Commercial training and support services available

Getting started

Ok, I'm ready, I want to develop an app for iOS and Android. What do I need?

Getting started

Ok, I'm ready, I want to develop an app for iOS and Android. What do I need?

- Titanium SDK

Getting started

Ok, I'm ready, I want to develop an app for iOS and Android. What do I need?

- Titanium SDK
- Android SDK

Getting started

Ok, I'm ready, I want to develop an app for iOS and Android. What do I need?

- Titanium SDK
- Android SDK
- iOS SDK

Getting started

Ok, I'm ready, I want to develop an app for iOS and Android. What do I need?

- Titanium SDK
- Android SDK
- iOS SDK
- A Mac...

Titanium Studio

Web – KitchenSink/Resources/app.js – Titanium Studio – /Users/cguerin/ENSEIGNEMENT/Titanium/workspace

App Explorer | Project Explorer

- Local Filesystem
- Golingo
- HelloWorld
- KitchenSink
  - i18n
  - modules
  - platform
  - plugins
  - Resources
    - android
    - examples
    - images
    - iphone
    - main_windows
    - app.js
    - app.jss
    - bg.js
    - comic_zine_ot.otf
    - common.jss
    - cricket.wav
    - default_app_logo.png

Outline | Samples

- tabGroup
- win1
- tab1
- win2
- tab2
- win3
- tab3
- win4
- tab4
- win5

app.js | 3d_transform.js

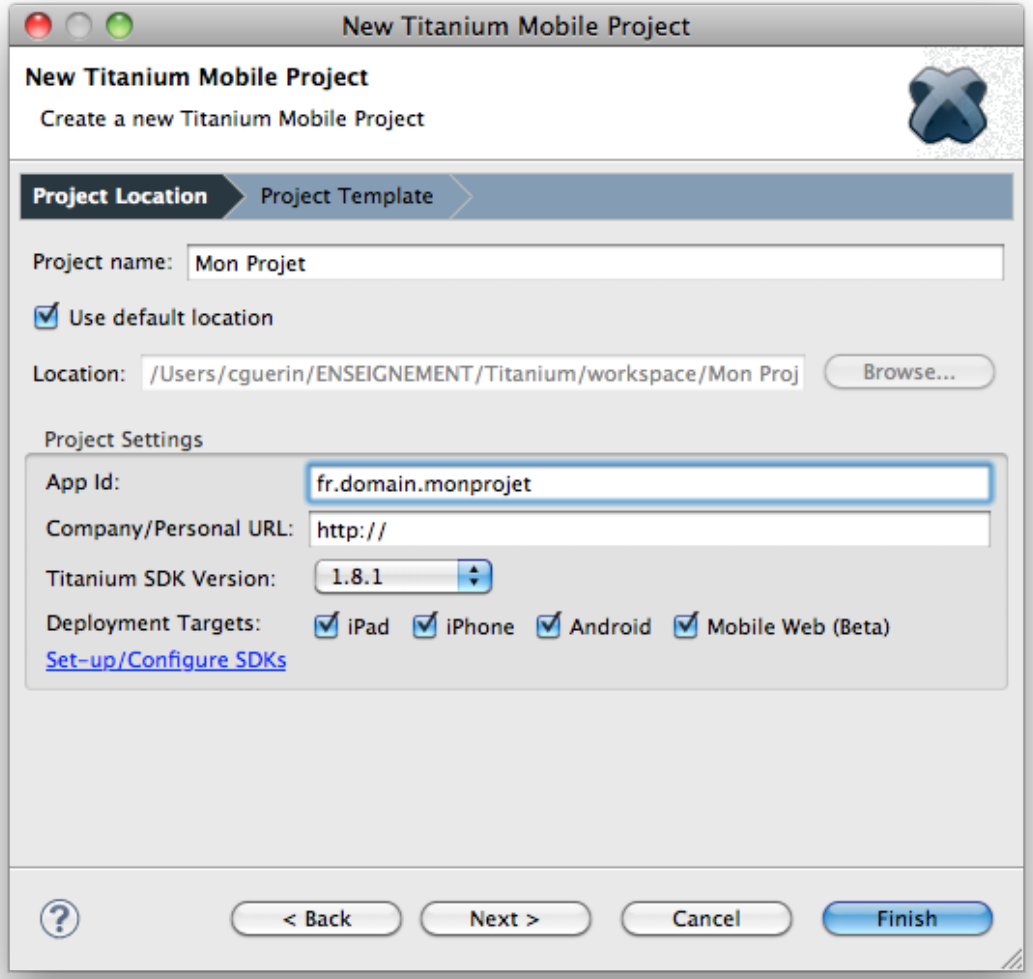
```
// this sets the background color of the master UIView (when there are no windows/tab groups on it)
Titanium.UI.setBackgroundColor('#000');

// create tab group
var tabGroup = Titanium.UI.createTabGroup({id:'tabGroup1'});


//
// create base UI tab and root window
//
var win1 = Titanium.UI.createWindow({className:'win1'});

var tab1 = Titanium.UI.createTab({
    id:'tab1',
    window:win1
});

//
// create controls tab and root window
//
var win2 = Titanium.UI.createWindow({
    url:'main_windows/controls.js',
    titleid:'controls_win_title'
});
var tab2 = Titanium.UI.createTab({
    icon:'images/tabs/KS_nav_ui.png',
    titleid:'controls_win_title',
```

Console | Terminal | Problems | Search

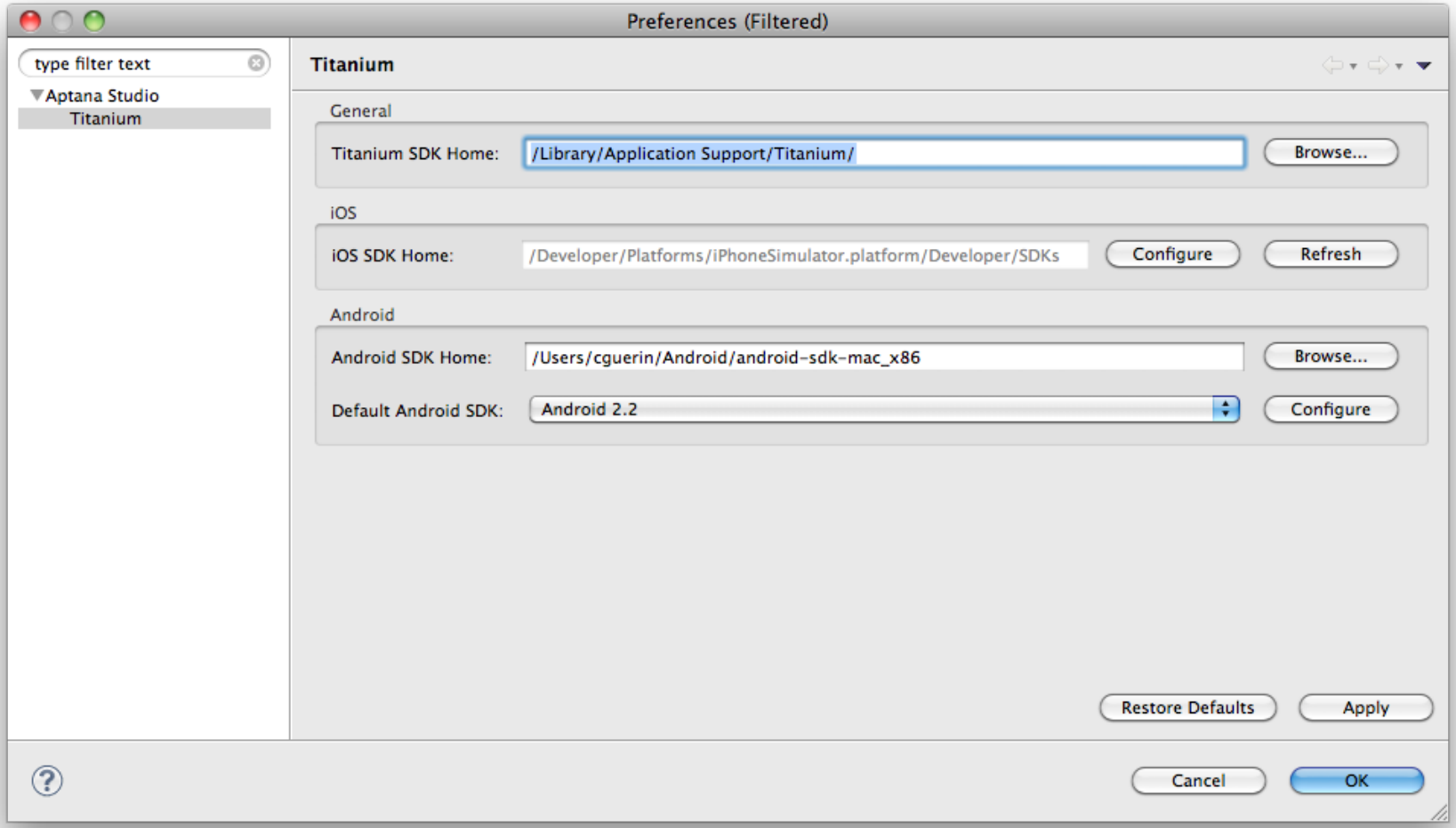
```
<terminated> Titanium iPhone Simulator - KitchenSink [Titanium iOS Simulator] iOS Simulator
[INFO] iPhone Device family: universal
[INFO] iPhone SDK version: 4.2
[INFO] iPhone simulated device: iphone
[INFO] Detected compiler plugin: ti.log/0.1
[INFO] Compiler plugin loaded and working for ios
[INFO] Compiling localization files
```

Clement Guerin

New Project


New Titanium Mobile Project
New Titanium Mobile Project
Create a new Titanium Mobile Project
Project Location
Project Template
Project name:
Mon Projet
Use default location
Location:
/Users/cguerin/ENSEIGNEMENT/Titanium/workspace/Mon Proj
Browse...
Project Settings
App Id:
fr.domain.monprojet
Company/Personal URL:
http://
Titanium SDK Version:
1.8.1
Deployment Targets:
iPad
iPhone
Android
Mobile Web (Beta)
Set-up/Configure SDKs
< Back
Next >
Cancel
Finish

SDK Paths


Preferences (Filtered)
type filter text
Aptana Studio
Titanium
Titanium
General
Titanium SDK Home:
/Library/Application Support/Titanium/
Browse...
iOS
iOS SDK Home:
/Developer/Platforms/iPhoneSimulator.platform/Developer/SDKs
Configure
Refresh
Android
Android SDK Home:
/Users/cguerin/Android/android-sdk-mac_x86
Browse...
Default Android SDK:
Android 2.2
Configure
Restore Defaults
Apply
Cancel
OK

Run

Web – KitchenSink/Resources/app.js – Titanium Studio – /Users/cguerin/ENSEIGNEMENT/Titanium/workspace

App Explorer | Project Explorer

Local Filesystem

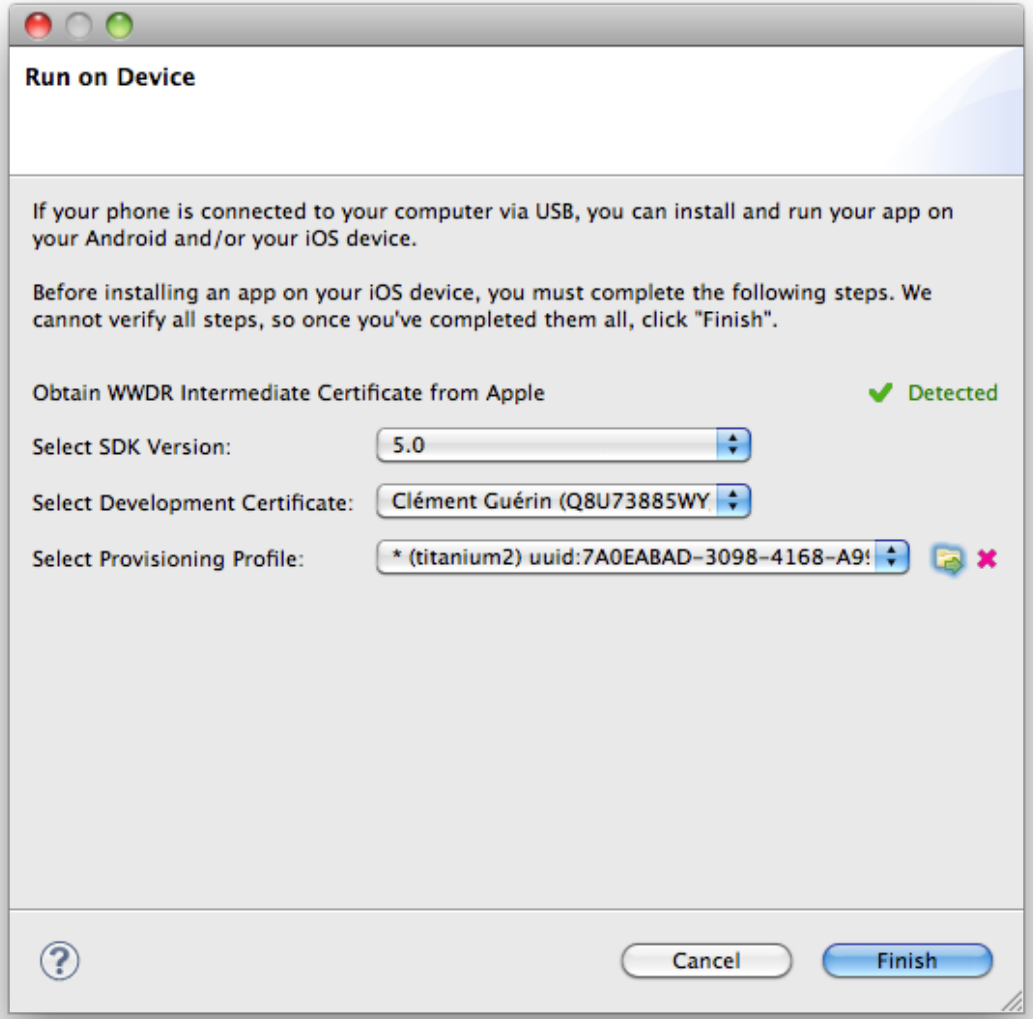
- New
- New From Template
- Go Into
- Deploy
- Indexing
- Show In
- Copy
- Paste
- Delete
- Move...
- Rename...
- Import...
- Export...
- Refresh F5
- Close Project
- Close Unrelated Projects
- Install to iOS Device
- Install to Android Device
- Run As
  - 1 Android Emulator
  - 2 iPad Simulator
  - 3 iPhone Simulator
  - Run Configurations...
- Debug As
- Team
- Compare With
- Restore from Local History...
- PyDev
- Properties ⌘I

app.js | 3d_transform.js

```
// this sets the background color of the master UIView (when there are no windows/tab groups on it)
Titanium.UI.setBackgroundColor('#000');

// create tab group
var tabGroup = Titanium.UI.createTabGroup({id:'tabGroup1'});


//
// create base UI tab and root window
//
var win1 = Titanium.UI.createWindow({className:'win1'});

var tab1 = Titanium.UI.createTab({
    id:'tab1',
    window:win1
});

//
// create controls tab and root window
//
var win2 = Titanium.UI.createWindow({
    url:'main_windows/controls.js',
    titleid:'controls_win_title'
});
var tab2 = Titanium.UI.createTab({
    icon:'images/tabs/KS_nav_ui.png',
    titleid:'controls_win_title',
```

Console | Terminal | Problems | Search

Install to iOS Device



Debug

Debug – KitchenSink/Resources/main_windows/base_ui.js – Titanium Studio – /Users/cguerin/ENSEIGNEMENT/Titanium/workspace

Debug | App Explorer

- Titanium iPhone Simulator – KitchenSink [Titanium iOS Simulator]
  - iOS JS Debugger
    - Thread [kroll$3] (Suspended (exception at line 92 in base_ui.js))
      - [Top-level script] [base_ui.js:92]
    - Thread [kroll$2] (Running)
    - Thread [kroll$1] (Running)
  - iOS Simulator
  - KitchenSink

Variables | Breakpoints | Expressions | Loaded Scripts

Name	Value
isiOS4Plus	[object Function]
view	[object TiUIView]
width	250
backgroundColor	"#000"
opacity	0.7
borderRadius	10
height	30
data	[object Array]

[object TiUIView]

app.js | 3d_transform.js | base_ui.js

```
85
86  // add table view to the window
87  Titanium.UI.currentWindow.add(tableview);
88
89  Titanium.UI.currentWindow.addEventListener('focus', function()
90  {
91      Ti.API.info('FOCUS RECEIVED IN base_ui');
92      Ti.API.info(Ti.dumpCoverage());
93  });
94  //
95  //  ADD EVENT LISTENERS FOR CUSTOM EVENTS
96  //
97  var win = Titanium.UI.createWindow({
98      height:30,
99      width:250,
100     bottom:110,
101     borderRadius:10
102 });
```

Outline

type filter text

- data
- tableview
- win
- view
- label

Console | Tasks | Studio Ajax Monitor

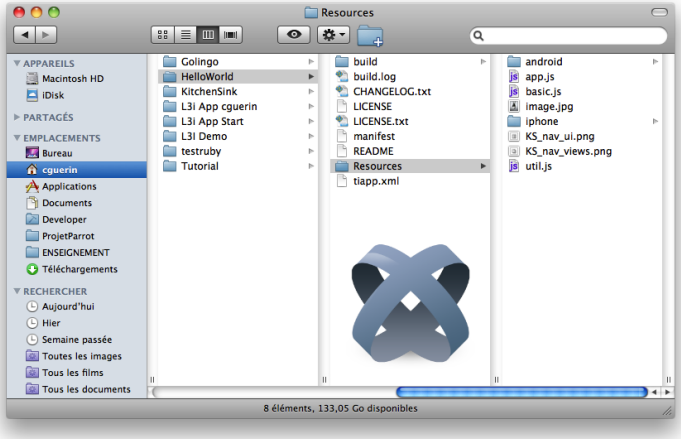
Titanium iPhone Simulator – KitchenSink [Titanium iOS Simulator] KitchenSink

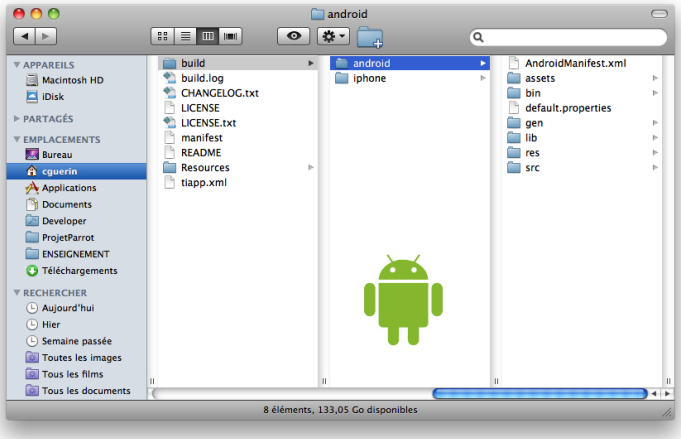
```
[INFO] tab blur - new index 1 old index 1
[INFO] tab Controls prevTab = null
[INFO] tab Controls prevTab = Controls
[INFO] tab Controls prevTab = Controls
[INFO] tab Controls prevTab = null
[INFO] tab blur - new index 0 old index 1
[DEBUG] loading: /Users/cguerin/ENSEIGNEMENT/Titanium/workspace/KitchenSink/Resources/main_windows/base_ui.js, resource: Users/cguerin/ENSEIGNEMENT/Titanium/workspace/KitchenSink/Resources/main_windows/base_u
[DEBUG] include url: file://localhost/Users/cguerin/ENSEIGNEMENT/Titanium/workspace/KitchenSink/Resources/examples/version.js
[DEBUG] loading: /Users/cguerin/ENSEIGNEMENT/Titanium/workspace/KitchenSink/Resources/examples/version.js, resource: Users/cguerin/ENSEIGNEMENT/Titanium/workspace/KitchenSink/Resources/examples/version_js
[INFO] FOCUS RECEIVED IN base_ui
[INFO] tab Base UI prevTab = Controls
```

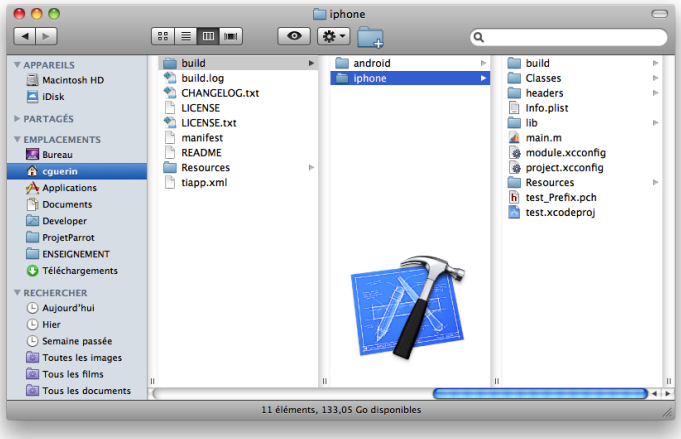
Clement Guerin

App deployment overview

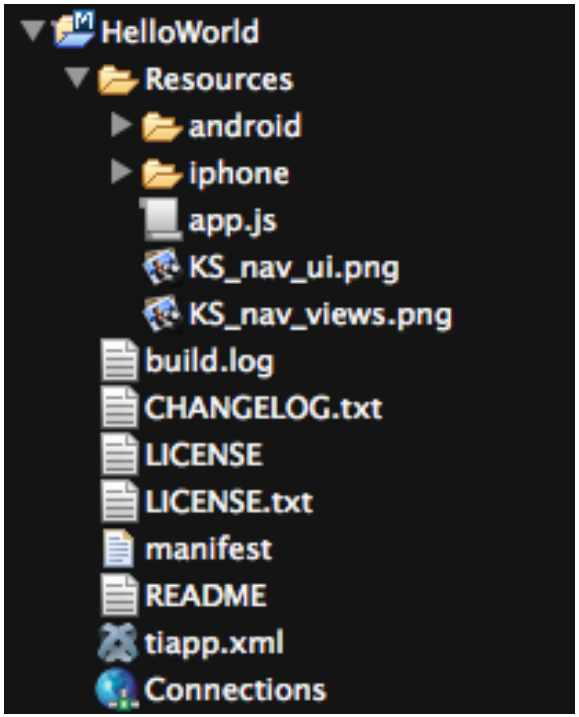
Native projects building

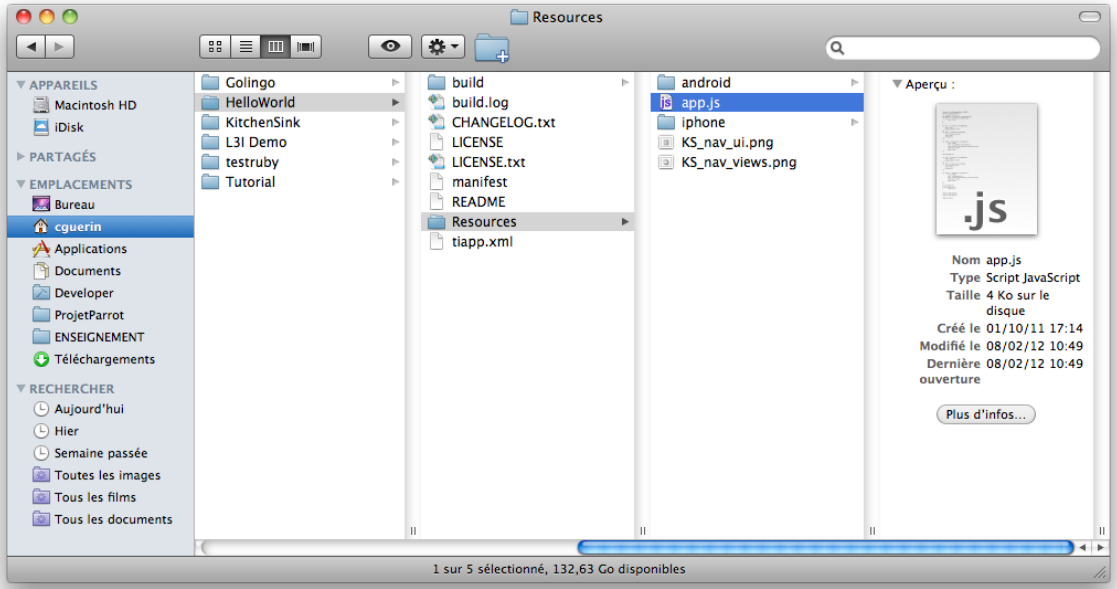




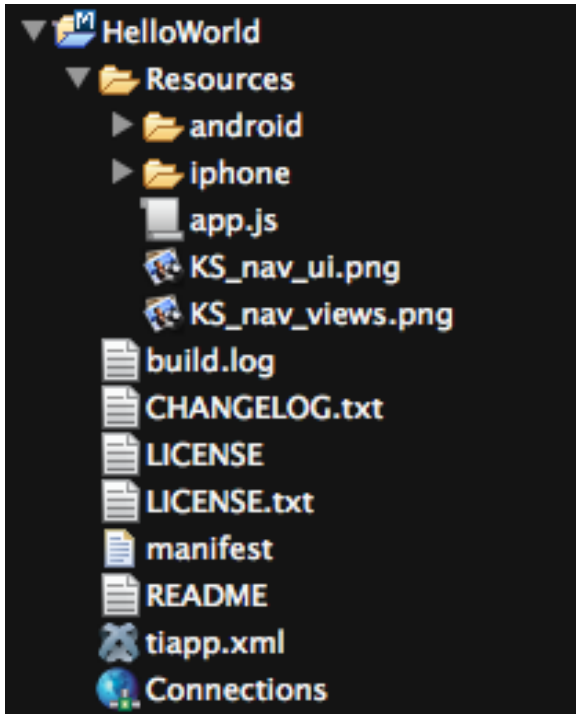


Basic project structure





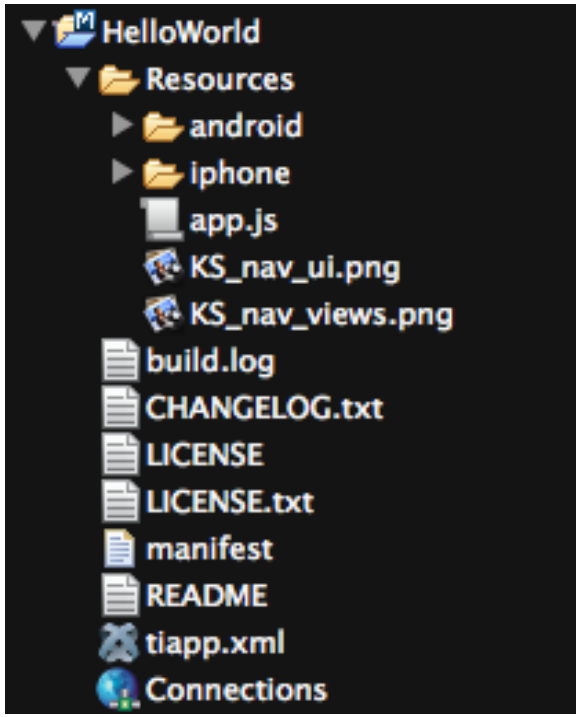
Basic project structure



Resources folder

Put every files and folders related to your project here

Basic project structure



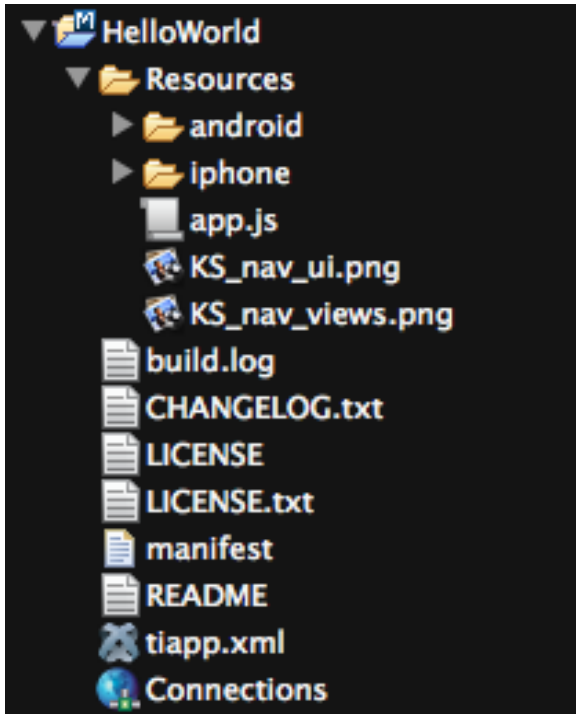
Resources folder

Put every files and folders related to your project here

android and *iphone* folders

Platform specific files folders (start screen, icons, etc.)

Basic project structure



Resources folder

Put every files and folders related to your project here

android and *iphone* folders

Platform specific files folders (start screen, icons, etc.)

app.js

Starting point of your application (main)

iphone and *android* folders

iphone

- appicon.png: 57x57 pixels icon for the application
- Default.png: 320x480 pixels start screen
- Default@2x.png: 640x960 pixels start screen for retina display

iphone and *android* folders

iphone

- appicon.png: 57x57 pixels icon for the application
- Default.png: 320x480 pixels start screen
- Default@2x.png: 640x960 pixels start screen for retina display

android

- appicon.png: icon for the application
- images folders: 5 different resolutions start screen

app.js

- Describe the content of the main window

```
var tabGroup = Titanium.UI.createTabGroup();

var win1 = Titanium.UI.createWindow({
    title:'Win 1',
    backgroundColor:'#fff'
});
var tab1 = Titanium.UI.createTab({
    icon:'KS_nav_views.png',
    title:'Tab 1',
    window:win1
});
var win2 = Titanium.UI.createWindow({
var tab2 = Titanium.UI.createTab({

tabGroup.addTab(tab1);
tabGroup.addTab(tab2);

tabGroup.open();
```

Code segmentation

- You can, and should, split your code into separate files.

Code segmentation

- You can, and should, split your code into separate files.
- You can access variables and functions using:
  - `TI.include`

```
//util.js
function affiche(text) {
    alert(text);
}

var message = "How are you?";
```

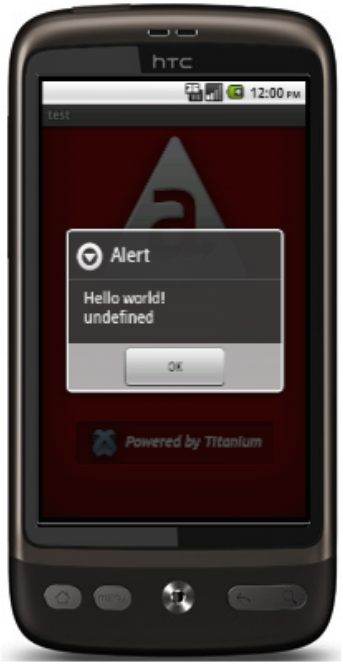
```
//app.js
Ti.include('util.js');

var win1 = Titanium.UI.createWindow({});
win1.open();

affiche("Hello world!\n"+message);
```

Code segmentation

- You can, and should, split your code into separate files.
- You can access variables and functions using:
  - `TI.include`
  - `require` and `exports`

```
//util.js
exports.affiche = function(text) {
    alert(text);
}

var message = "How are you?";
```

```
//app.js
var util = require('util');

var win1 = Titanium.UI.createWindow({});
win1.open();

util.affiche("Hello world!\n"+util.message);
```



Before starting

- This part introduce the different Titanium modules you will have to deal with during the TP

Before starting

- This part introduce the different Titanium modules you will have to deal with during the TP
- There are over 8000 methods and properties implemented into the 35 modules of the Titanium API

Before starting

- This part introduce the different Titanium modules you will have to deal with during the TP
- There are over 8000 methods and properties implemented into the 35 modules of the Titanium API
- The online documentation[2] is quite complete and understandable, don't be afraid to refer to it!

[2] http://developer.appcelerator.com/apidoc/mobile/

Before starting

- This part introduce the different Titanium modules you will have to deal with during the TP
- There are over 8000 methods and properties implemented into the 35 modules of the Titanium API
- The online documentation[2] is quite complete and understandable, don't be afraid to refer to it!
- Appcelerator provides an open source KitchenSink[3] app gathering nearly everything you can do with Titanium. Check it out when you are stuck with something.

[2] http://developer.appcelerator.com/apidoc/mobile/
[3] http://developer.appcelerator.com/doc/kitchensink

First look at the API

- Most of the time, the creation of a component looks like this:

```
13 var myComponent = Titanium.ComponentModule.createComponentName({
14     param1:value1,
15     param2:value2,
16     param3:value3
17 });
```

First look at the API

- Most of the time, the creation of a component looks like this:

```
13 var myComponent = Titanium.ComponentModule.createComponentName({
14     param1:value1,
15     param2:value2,
16     param3:value3
17 });
```

- Example: creation of a label

```
20 var myLabel = Titanium.UI.createLabel({
21     color:'#000',
22     font:{fontSize:20,fontFamily:'Helvetica Neue'},
23     text:'Blah',
24     textAlign:'center',
25     width:'auto',
26     visible:false
27 });
28 myLabel.visible=true;
```

First look at the API

- Most of the time, the creation of a component looks like this:

```
13 var myComponent = Titanium.ComponentModule.createComponentName({
14     param1:value1,
15     param2:value2,
16     param3:value3
17 });
```

- Example: creation of a label

```
20 var myLabel = Titanium.UI.createLabel({
21     color:'#000',
22     font:{fontSize:20,fontFamily:'Helvetica Neue'},
23     text:'Blah',
24     textAlign:'center',
25     width:'auto',
26     visible:false
27 });
28 myLabel.visible=true;
```

- The online documentation gives you access to the whole list of properties

Titanium.UI

- *The UI module is responsible for native user-interface components and interaction inside Titanium.*

Titanium.UI

- *The UI module is responsible for native user-interface components and interaction inside Titanium.*
- It provides a way to create and handle every pieces of user interface you will need for you app

Titanium.UI

- *The UI module is responsible for native user-interface components and interaction inside Titanium.*
- It provides a way to create and handle every pieces of user interface you will need for you app
- There are three kinds of UI components:

Titanium.UI

- *The UI module is responsible for native user-interface components and interaction inside Titanium.*
- It provides a way to create and handle every pieces of user interface you will need for you app
- There are three kinds of UI components:

Windows

Top-level components. Each app will contains, at least, one window.

Titanium.UI

- *The UI module is responsible for native user-interface components and interaction inside Titanium.*
- It provides a way to create and handle every pieces of user interface you will need for you app
- There are three kinds of UI components:

Windows

Top-level components. Each app will contains, at least, one window.

Views

Containers that can host other components. *Ex: TableView, ScrollView*

Titanium.UI

- *The UI module is responsible for native user-interface components and interaction inside Titanium.*
- It provides a way to create and handle every pieces of user interface you will need for you app
- There are three kinds of UI components:

Windows

Top-level components. Each app will contains, at least, one window.

Views

Containers that can host other components. *Ex: TableView, ScrollView*

Controls

Widgets that users can interact with. *Ex: Button, Slider, Tab...*

Titanium.UI – Windows

Titanium.UI.Window

- A window is a top level container which can contain widgets and other views but will unlikely be contained inside other views.

Titanium.UI – Windows

Titanium.UI.Window

- A window is a top level container which can contain widgets and other views but will unlikely be contained inside other views.
- A window can be loaded from, and be described in, a separate JavaScript file by specifying the property `url`. The window is then executed in a separate context.

Titanium.UI – Windows

Titanium.UI.Window

- A window is a top level container which can contain widgets and other views but will unlikely be contained inside other views.
- A window can be loaded from, and be described in, a separate JavaScript file by specifying the property `url`. The window is then executed in a separate context.

```
148 var win = Titanium.UI.createWindow({
149     url:'myWindow.js',
150     title:'My Window',
151     backgroundColor:'#ccc'
152 });
```

Titanium.UI – Windows

Titanium.UI.Window

There are two ways to pass data between contexts:

- Shared references

```
24 //app.js
25 var win = Titanium.UI.createWindow({
26     url:'myWindow.js',
27     title:'My Window',
28     backgroundColor:'#ccc',
29     foo:'bar'
30 });
```

```
24 //myWindow.js
25 alert(Titanium.UI.currentWindow.foo);
```

Titanium.UI – Windows

Titanium.UI.Window

There are two ways to pass data between contexts:

- Shared references
- Firing events

```
24 //app.js
25 var win = Titanium.UI.createWindow({
26     url:'myWindow.js',
27     title:'My Window',
28     backgroundColor:'#ccc'
29 });
30 win.open();
31 win.fireEvent('foo',{text:'bar'});
```

```
24 //myWindow.js
25 Titanium.UI.currentWindow.addEventListener('foo',function(e) {
26     alert(e.text);
27 });
```

Titanium.UI – Windows

Titanium.UI.TabGroup

Windows can't usually be managed by any other component... Well, TabGroup can.[a]

[a] iOS NavigationGroup and SpitWindow too.

Titanium.UI – Windows

Titanium.UI.TabGroup

Windows can't usually be managed by any other component... Well, TabGroup can.[a]

[a] iOS NavigationGroup and SpitWindow too.

Titanium.UI.Tab

iOS: Tabs maintain a stack of windows
Android: calling `open` opens a new heavyweight window

Titanium.UI – Windows

Titanium.UI.TabGroup

Windows can't usually be managed by any other component... Well, TabGroup can.[a]

[a] iOS NavigationGroup and SpitWindow too.

Titanium.UI.Tab

iOS: Tabs maintain a stack of windows
Android: calling `open` opens a new heavyweight window

```
var tabGroup = Titanium.UI.createTabGroup();

var win1 = Titanium.UI.createWindow({
    title:'Win 1',
    backgroundColor:'#fff'
});
var tab1 = Titanium.UI.createTab({
    icon:'KS_nav_views.png',
    title:'Tab 1',
    window:win1
});
var win2 = Titanium.UI.createWindow({ ...
var tab2 = Titanium.UI.createTab({ ...

tabGroup.addTab(tab1);
tabGroup.addTab(tab2);

tabGroup.open();
```

Titanium.UI – Views

Titanium.UI.View

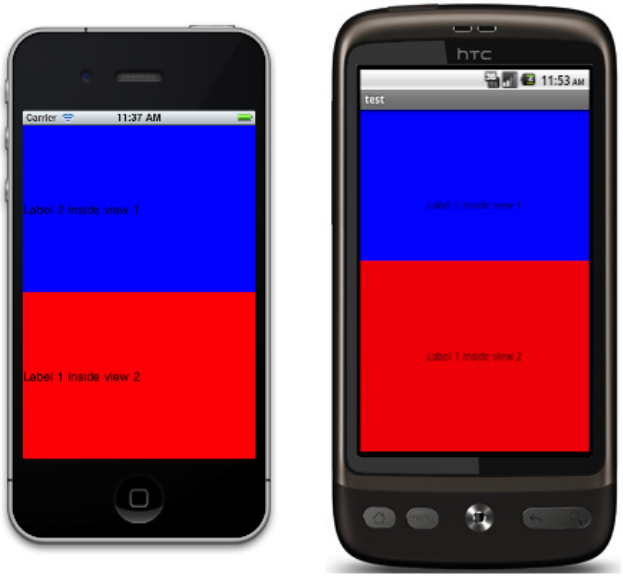
- A View is a container where you can insert other widgets.

Titanium.UI – Views

Titanium.UI.View

- A View is a container where you can insert other widgets.

```
var win = Titanium.UI.createWindow({
    backgroundColor:'white',
});

var view1 = Ti.UI.createView({
    backgroundColor:'blue',
    width:Ti.Platform.displayCaps.platformWidth,
    height:Ti.Platform.displayCaps.platformHeight/2,
    top:0
});
var view2 = Ti.UI.createView({
    backgroundColor:'red',
    width:Ti.Platform.displayCaps.platformWidth,
    height:Ti.Platform.displayCaps.platformHeight/2,
    bottom:0
});

var label1 = Ti.UI.createLabel({
    text:'Label 1 inside view 2'
})
var label2 = Ti.UI.createLabel({
    text:'Label 2 inside view 1'
})
view1.add(label2); view2.add(label1);
win.add(view1); win.add(view2);
win.open();
```



Titanium.UI – Views

Titanium.UI.ScrollView

- Views added to the ScrollView will be scrolled based on the content size of the ScrollView.

Titanium.UI – Views

Titanium.UI.ScrollView

- Views added to the ScrollView will be scrolled based on the content size of the ScrollView.

```
var win = Titanium.UI.createWindow({
    backgroundColor:'white',
});

var scrollView = Titanium.UI.createScrollView({
    contentWidth:'auto',
    contentHeight:'auto',
    showVerticalScrollIndicator:true
});
var view = Ti.UI.createView({
    backgroundColor:'#aaa',
    borderRadius:10,
    width:300,
    height:1000,
    top:10,
    bottom:10
});
var label = Ti.UI.createLabel({
    font:{fontSize:50,fontFamily:'Helvetica Neue'},
    color:'black',
    text:'Some text...'
})
view.add(label);
scrollView.add(view);
win.add(scrollView);
win.open();
```

Titanium.UI – Views

Titanium.UI.TableView

- A TableView is a scrollable list composed of Titanium.UI.TableViewRow components.

Titanium.UI – Views

Titanium.UI.TableView

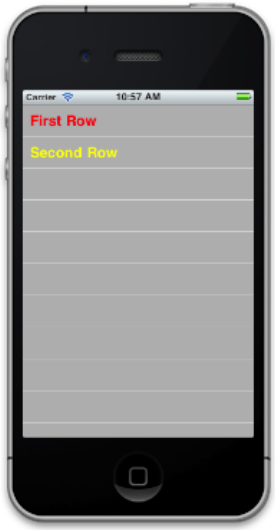
- A TableView is a scrollable list composed of Titanium.UI.TableViewRow components.
- TableViewRows are passed through the `data` property.

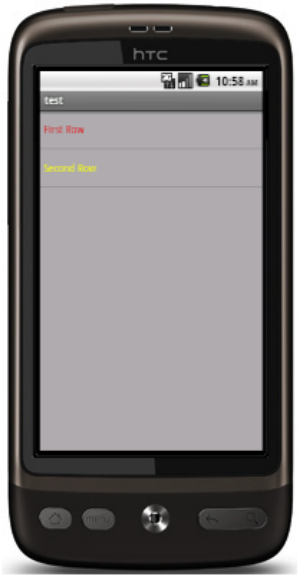
Titanium.UI – Views

Titanium.UI.TableView

- A TableView is a scrollable list composed of Titanium.UI.TableViewRow components.
- TableViewRows are passed through the `data` property.

```
var win = Titanium.UI.createWindow({
});

var rows = [{title:"First Row", color:'red'},
            {title:"Second Row", color:'yellow'}];
var table = Titanium.UI.createTableView({
    backgroundColor:'#bbb',
    data:rows
});

win.add(table);
win.open();
```





Titanium.UI – Views

Titanium.UI.ImageView

- A view to display a single image or series of animated images.

Titanium.UI – Views

Titanium.UI.ImageView

- A view to display a single image or series of animated images.

```
var win = Titanium.UI.createWindow({
    backgroundColor:'white',
});

var imageView = Titanium.UI.createImageView({
    height:300,
    width:'auto',
    image:'/image.jpg',
    top:30
});
win.add(imageView);
win.open();
```

Titanium.UI – Controls

Titanium.UI.Label

- Simple customizable label

Titanium.UI – Controls

Titanium.UI.Label

- Simple customizable label

Titanium.UI.Button

- A button has four states: normal, disabled, focused and selected. You can specify a background image for each state.

Titanium.UI – Controls

Titanium.UI.Label

- Simple customizable label

Titanium.UI.Button

- A button has four states: normal, disabled, focused and selected. You can specify a background image for each state.

```
var win = Titanium.UI.createWindow({
    backgroundColor:'white',
});

var myButton1 = Titanium.UI.createButton({
    backgroundDisabledImage:'KS_nav_ui.png',
    backgroundImage:'KS_nav_views.png',
    enabled:true,
    width:70,
    height:70,
    left:20,
    top:50
});

var myButton2 = Titanium.UI.createButton({
    backgroundDisabledImage:'KS_nav_ui.png',
    backgroundImage:'KS_nav_views.png',
    enabled:false,
    width:70,
    height:70,
    right:20,
    top:50
});

win.add(myButton1);win.add(myButton2);
win.open();
```

Titanium.UI – Controls

See also

- Titanium.UI.Switch
- Titanium.UI.Slider
- Titanium.UI.TextField
- Titanium.UI.TextArea
- Titanium.UI.Picker
- Titanium.UI.ProgressBar

Titanium.UI – Controls

See also

- Titanium.UI.Switch
- Titanium.UI.Slider
- Titanium.UI.TextField
- Titanium.UI.TextArea
- Titanium.UI.Picker
- Titanium.UI.ProgressBar

```
var win = Titanium.UI.createWindow({
    backgroundColor:'white',
});

var mySwitch = Titanium.UI.createSwitch({
    value:false,
    top:50
});

var mySlider = Titanium.UI.createSlider({
    top: 100,
    min: 0,
    max: 100,
    width: '75%',
    value: 50
    });

var myTextField = Titanium.UI.createTextField({
    color:'#bbb',
    height:35,
    top:150,
    width:'50%',
    hintText:'Type something',
    borderStyle:Titanium.UI.INPUT_BORDERSTYLE_BEZEL
});

win.add(mySwitch);win.add(mySlider);win.add(myTextField);
win.open();
```

Events handling

- The behavior of a component responding to an event has to be defined in a callback function.

```
6  var myButton = Titanium.UI.createButton({});
7  myButton.addEventListener('click', function(e){
8      alert(e.x+" "+e.y);
9  });
```

Events handling

- The behavior of a component responding to an event has to be defined in a callback function.

```
6  var myButton = Titanium.UI.createButton({});
7  myButton.addEventListener('click', function(e){
8      alert(e.x+" "+e.y);
9  });
```

- Here is a list of events that can be handled by a Button. Read the documentation of a component to know which event it handles.

- click
- dblclick
- longclick
- longpress
- singletap
- doubletap
- twofingertap
- pinch
- swipe
- touchstart
- touchmove
- touchend

Titanium.Network

Titanium.Network.HTTPClient

- HTTPClient implements the XMLHttpRequest specification.

Titanium.Network

Titanium.Network.HTTPClient

- HTTPClient implements the XMLHttpRequest specification.
- You have to implement the `onload` and `onerror` callback to handle the HTTP response.

Titanium.Network

Titanium.Network.HTTPClient

- HTTPClient implements the XMLHttpRequest specification.
- You have to implement the `onload` and `onerror` callback to handle the HTTP response.
- You can get the response data via `responseText`, `responseData` and `responseXML`.

Titanium.Network

Titanium.Network.HTTPClient

- HTTPClient implements the XMLHttpRequest specification.
- You have to implement the `onload` and `onerror` callback to handle the HTTP response.
- You can get the response data via `responseText`, `responseData` and `responseXML`.

```
16 var xmlHttp = Titanium.Network.createHTTPClient();
17 xmlHttp.onload = function()
18 {
19     alert(this.responseText);
20
21 }
22 xmlHttp.onerror = function()
23 {
24     alert("Web page unreachable")
25 }
26 xmlHttp.open("GET", "http://www.univ-larochelle.fr/");
27 xmlHttp.send();
```

Titanium.Network

See also

- Titanium.Network.Socket
- Titanium.Network.BonjourService
- Titanium.Facebook
- Titanium.Yahoo

Device related modules

Titanium.Platform

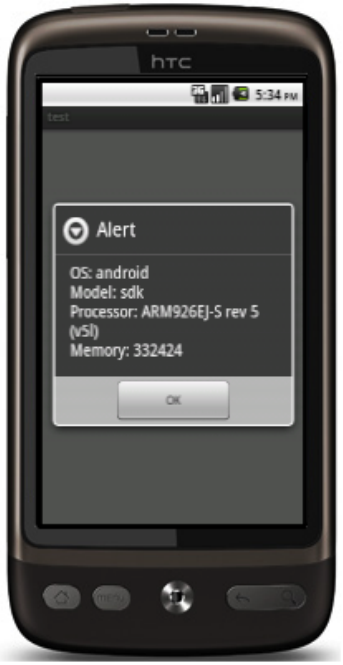
- Platform module grants access to device's platform-related functionalities.

Device related modules

Titanium.Platform

- Platform module grants access to device's platform-related functionalities.

```
var win = Titanium.UI.createWindow({
    backgroundColor:'white',
});

alert("OS: "+Titanium.Platform.osname
      +"\nModel: "+Titanium.Platform.model
      +"\nProcessor: "+Titanium.Platform.architecture
      +"\nMemory: "+Titanium.Platform.availableMemory);

win.open();
```



Device related modules

Titanium.Filesystem

Files management

Device related modules

Titanium.Filesystem

Files management

Titanium.Accelerometer

Accelerometer handling

Device related modules

Titanium.Filesystem

Files management

Titanium.Accelerometer

Accelerometer handling

Titanium.Geolocation

Access location based information

Device related modules

Titanium.Filesystem

Files management

Titanium.Accelerometer

Accelerometer handling

Titanium.Geolocation

Access location based information

```
var win = Titanium.UI.createWindow({
    backgroundColor:'white',
});

var fileLabel = Ti.UI.createLabel({
    top:80,
    color:'black',
    height:'auto',
    text:Titanium.Filesystem.applicationDataDirectory,
    textAlign:'center'
});
win.add(fileLabel);

var accLabel = Ti.UI.createLabel({
    top:150,
    color:'blue',
    height:'auto',
    textAlign:'center'
});
win.add(accLabel);

Titanium.Accelerometer.addEventListener('update',function(e)
{
    accLabel.text="accelerometer - x:"+e.x+",y:"+e.y+",z:"+e.z;
});
```

```
var myMap = Titanium.Map.createView({
    top: 200
});
win.add(myMap);

var mapLabel = Ti.UI.createLabel({
    top:10,
    color:'red',
    height:'auto',
    textAlign:'center'
});
win.add(mapLabel);

function getLocation(){
    Titanium.Geolocation.getCurrentPosition(function(e){
        var region={
            latitude: e.coords.latitude,
            longitude: e.coords.longitude,
            animate:true,
            latitudeDelta:0.01,
            longitudeDelta:0.01
        };
        myMap.setLocation(region);
        mapLabel.text=e.coords.latitude+"\n"+e.coords.longitude;
    });
}

getLocation();
win.open();
```

See also

Titanium.Database

Can be used to create and access a in-application SQLite database.

Titanium.Media

Provides a way to play or record audio and video material.

Titanium.Contacts

Gives you access to the device's address book (currently read-only on Android).

Titanium.XML

Used for parsing and processing XML-based content.

Game development

- Not fully supported yet

Game development

- Not fully supported yet
- An OpenGL plugin (iOS only) is being developped by Logical Labs[4]

[4] http://tzmartin.com/titanium-mobile-opengl-es-docs/2011-09-01

Game development

- Not fully supported yet
- An OpenGL plugin (iOS only) is being developped by Logical Labs[4]
- Box2D module (iOS only) has been released last September[5]

[4] http://tzmartin.com/titanium-mobile-opengl-es-docs/2011-09-01
[5] http://developer.appcelerator.com/blog/2011/09/

Game development

- Not fully supported yet
- An OpenGL plugin (iOS only) is being developped by Logical Labs[4]
- Box2D module (iOS only) has been released last September[5]
- You can manage to build something nice without those though. See Golingo[6][7]

[4] http://tzmartin.com/titanium-mobile-opengl-es-docs/2011-09-01
[5] http://developer.appcelerator.com/blog/2011/09/
[6] http://www.golingoapp.com/
[7] https://github.com/krawaller/Golingo

Game development

- Not fully supported yet
- An OpenGL plugin (iOS only) is being developped by Logical Labs[4]
- Box2D module (iOS only) has been released last September[5]
- You can manage to build something nice without those though. See Golingo[6][7]
- For a cross-platform game development, prefer frameworks like Unity or Corona

[4] http://tzmartin.com/titanium-mobile-opengl-es-docs/2011-09-01
[5] http://developer.appcelerator.com/blog/2011/09/
[6] http://www.golingoapp.com/
[7] https://github.com/krawaller/Golingo

Some problems yet to be fixed

- A significant part of the code still has to be platform specific

Some problems yet to be fixed

- A significant part of the code still has to be platform specific
- Titanium apps are a lot heavier than actual native apps

Some problems yet to be fixed

- A significant part of the code still has to be platform specific
- Titanium apps are a lot heavier than actual native apps
- Memory leaks can be critical in an iOS app and you can't help it

Some problems yet to be fixed

- A significant part of the code still has to be platform specific
- Titanium apps are a lot heavier than actual native apps
- Memory leaks can be critical in an iOS app and you can't help it
- The framework is actively maintained and constantly improved by the growing community

Resources

- Kitchen Sink Application
- Online documentation
- Official forum