Comp 388/422 - Software Development for Wireless and Mobile Devices

Fall Semester 2015 - Week 6

Dr Nick Hayward

Next goals

- DEV week overview...
- DEV week presentation and demo...
- recap of current design
- mobile examples
  - *fix footer on Android*
  - *create shell design for app*
  - *add plugin - media...*
- design examples & considerations
- quiz

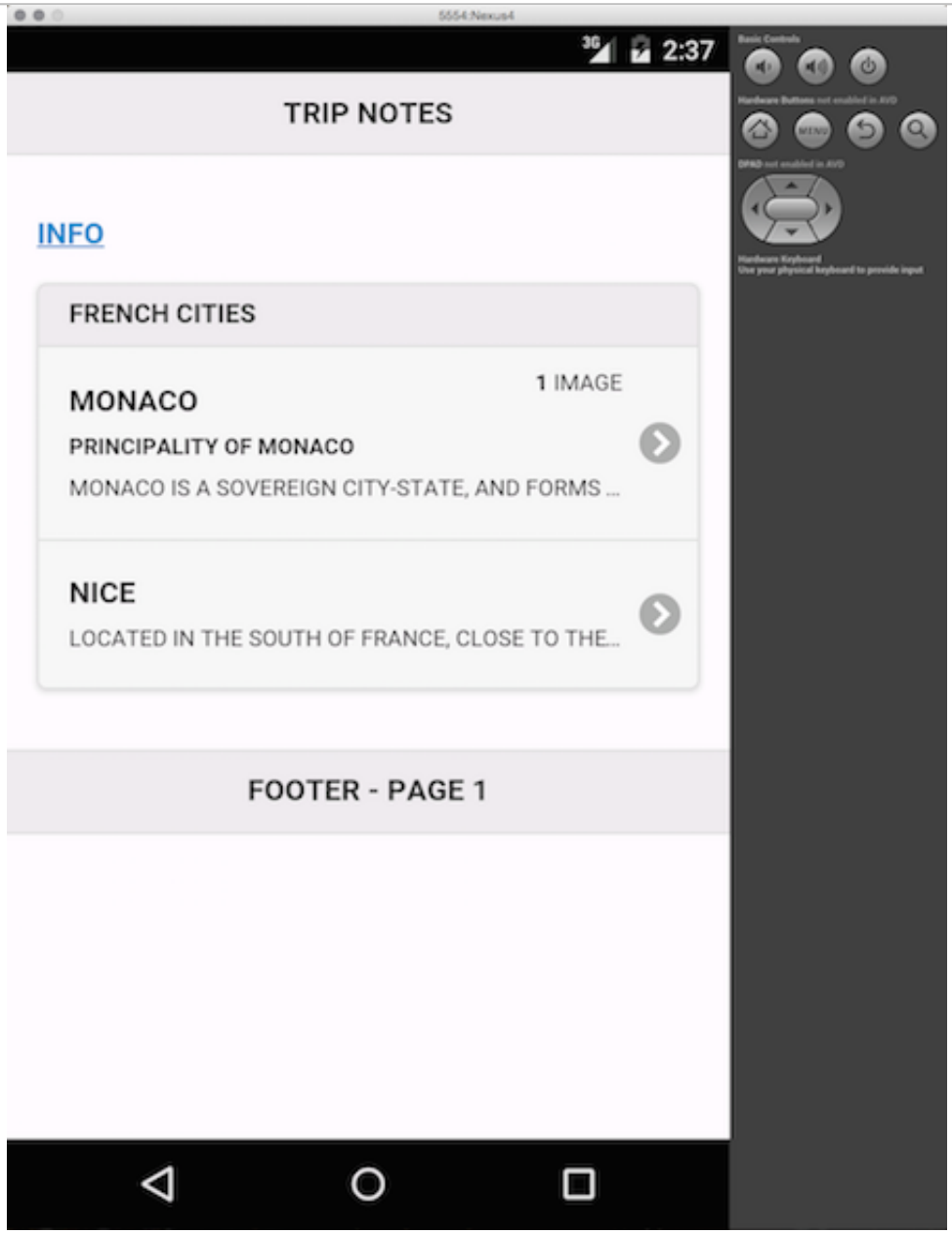
DEV week overview...

- begin development of a mobile application using Apache Cordova
  - *not a responsive website viewed on a mobile device*
- can be platform agnostic (cross-platform) or specific targeted OS
  - *eg: Android, iOS, Windows Phone using Cordova APIs*
  - *consider choice, and explain why?*
- outline concept, research conducted to date
- consider applicable design patterns
- are you using any sensors etc?
  - *how, why?*
- prototyping
  - *demo current prototypes*
  - *any working tests or models etc*

DEV week presentation and demo...

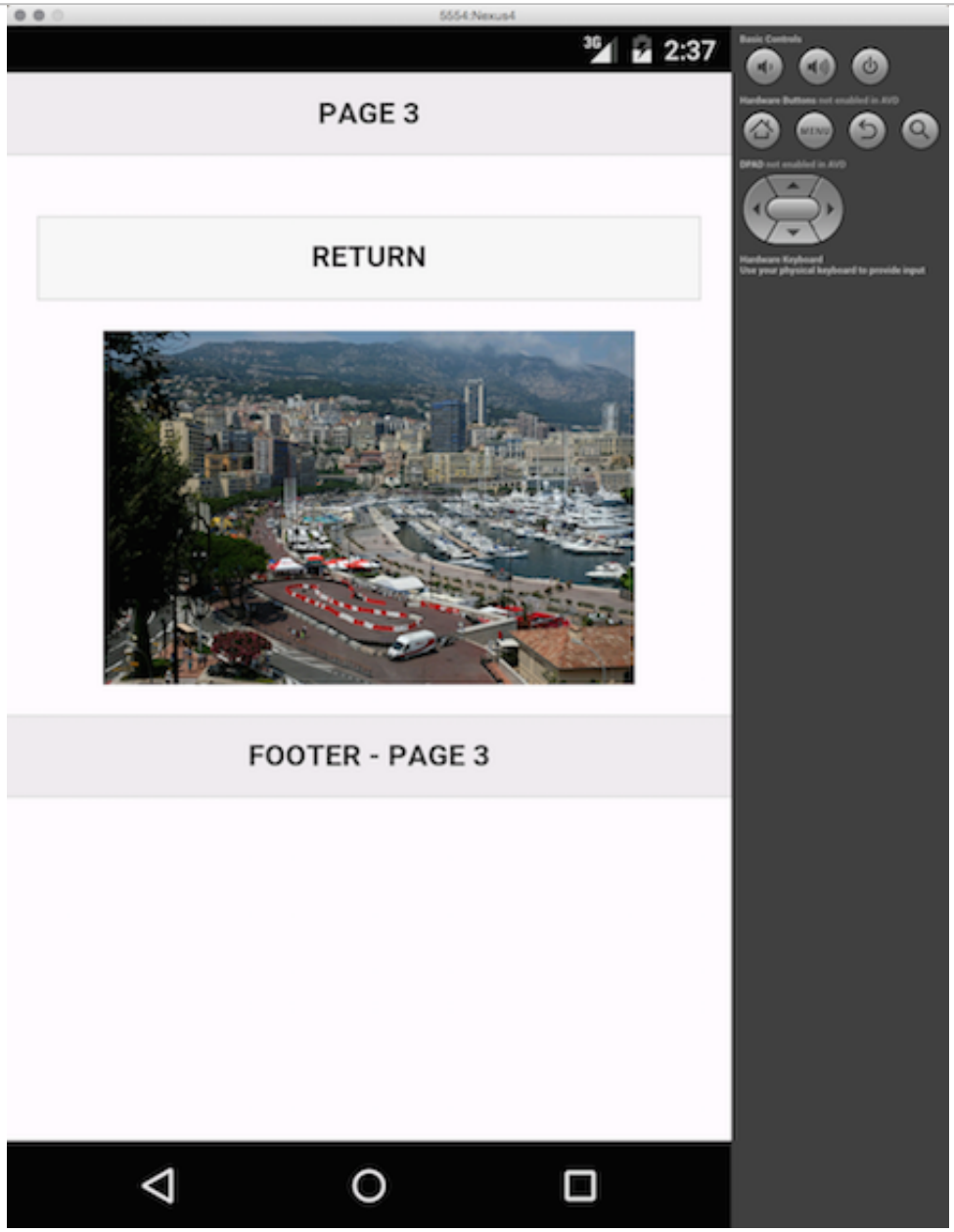
- brief presentation or demonstration of current project work
- ~ 5 to 10 minutes per group
- analysis of work conducted so far
  - *eg: during semester & DEV week*
- presentation, demonstration, or video overview...
  - *outline mobile app*
  - *show prototypes and designs*
  - *explain what works & does not work*
  - *...*

Image - Cordova app - Trip Notes - example 1



Cordova - trip notes - week 5 - home screen

Image - Cordova app - Trip Notes - example 2



Cordova - trip notes - week 5 - page with image

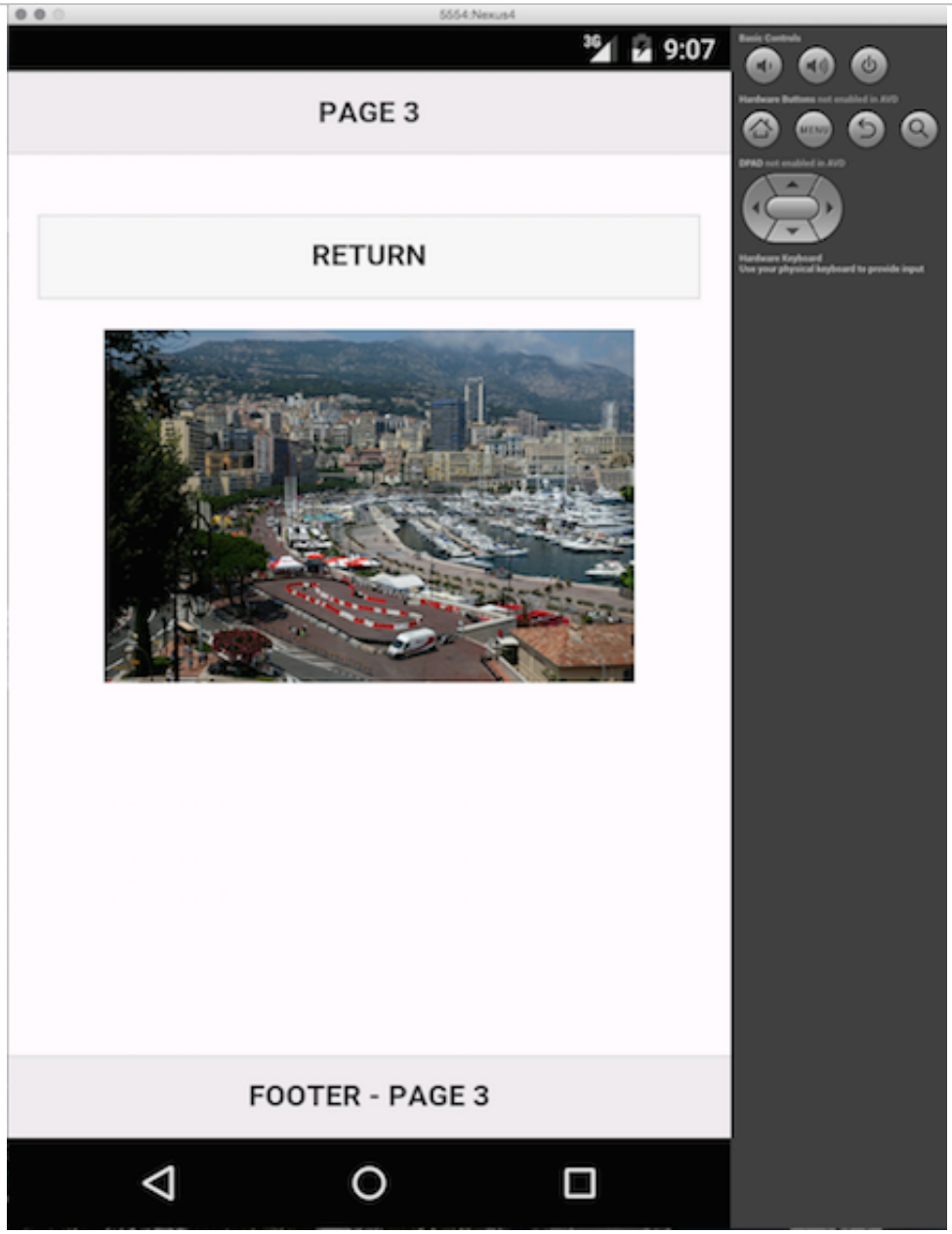
Cordova app - current design

- current design includes
  - *header*
  - *main*
  - *footer*
- fix footer to bottom of a given view using the following attribute,

```
<div data-role="footer" data-position="fixed">
  <h5>footer - page 3</h5>
</div><!-- /footer -->
```

- set this attribute on any of our footer sections

Image - Cordova app - Trip Notes - example 3



Cordova - trip notes - week 5 - fixed footer

Cordova app - create shell app

blueprint

- create a shell app we can use as a template
  - *updating designs*
  - *testing new features*
  - *working with various APIs...*
- updates include
  - *layout of* `index.html`
  - *a few custom styles for* `style.css`
  - *then add an initial splash screen and settings*
  - *then add an initial app icon...*

Cordova app - settings - config.xml

blueprint

- an Apache filled `config.xml` file
  - *we need to slightly modify for our requirements*

```
<name>blueprint</name>
<description>
    blueprint for Apache Cordova frameworks with jQuery Mobile
</description>
<author email="ancientlives@gmail.com" href="http://csteach422.github.io">
    ancientlives
</author>
<content src="index.html" />
```

Cordova app - index.html

blueprint

- good idea to strip out the `index.html` page
  - *create a consistent layout and structure for developing applications*
- start with a simple body
  - *organised with a header and a main content category*

```
<!-- homepage -->
<div data-role="page" id="home">
    <div data-role="header">
    <h3>blueprint</h3>
    </div><!-- /header -->
  <div role="main" class="ui-content">

    </div><!-- /content -->
</div><!-- /home -->
```

- many mobile applications do not include a footer within their content categories
  - *unless specifically required by a given application structure or functionality*
  - *leave footer out of this default blueprint*

Cordova app - style.css

blueprint

- for initial applications use default styling offered by jQuery Mobile
  - *otherwise, we can remove all defaults*
  - *create our own default, basic styles*
- preferred aesthetic scheme and palette
  - *add to* `www/assets/styles/style.css`

Cordova app - working with plugins - part 1

getting started

- start looking at some of the plugins available for Cordova
  - *media playback*
- test our initial Cordova blueprint with jQuery Mobile
  - *add some existing plugins*
  - *see how they fit together to create a coherent, basic application*
- create our new project

```
create plugintest1 com.example.plugintest plugintest1
```

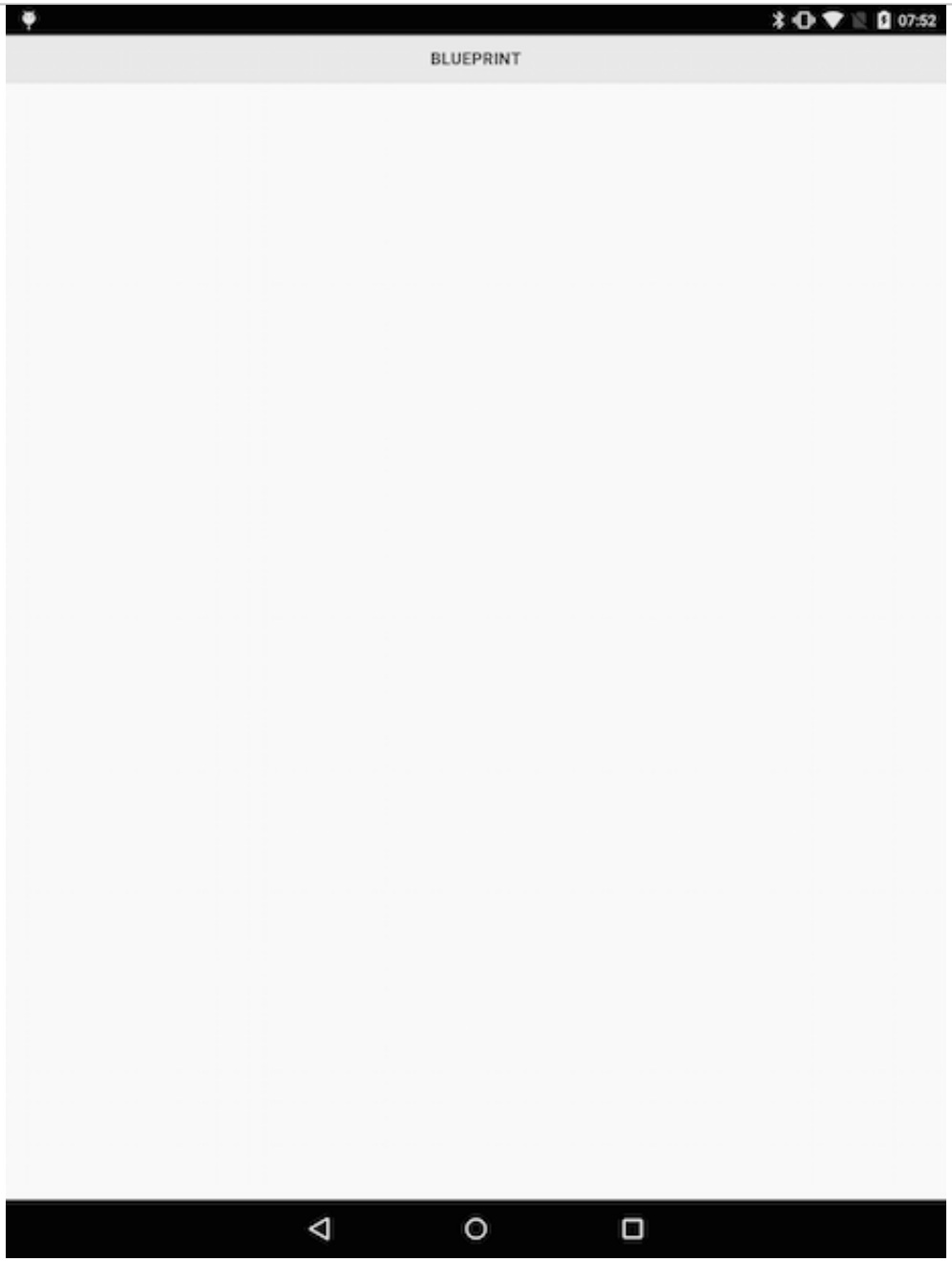
- add support for Android platform

```
cordova platform add android
```

- add support for other platforms, as required, such as iOS, Windows Phone...
- transfer our default `www` directory from the **blueprint**
- start updating some of the settings in the `config.xml` file for the application
  - *metadata for author, description, name...*
- quickly run and test this base for our new application

```
//run in the Android emulator
cordova emulate android
//run on a connected Android device
cordova run android
```

Image - Cordova app - Plugin Test 1 - getting started



Cordova - Plugin Test - getting started

Cordova app - working with plugins - part 2

add plugins

- add our required plugins to the test application
  - *add plugins for **device**, **file**, and **media***
- **device** plugin added to check and read information about current device
  - *in effect our Android phone or tablet*
- **file** plugin is required to access the device's underlying filesystem
- **media** helps us record and playback media files
- add these plugins to our project with the following Cordova commands

```
//add device plugin
cordova plugin add https://git-wip-us.apache.org/repos/asf/cordova-plugin-device.git
//add file plugin
cordova plugin add https://git-wip-us.apache.org/repos/asf/cordova-plugin-file.git
//add media plugin
cordova plugin add https://git-wip-us.apache.org/repos/asf/cordova-plugin-media.git
```

- ensure new plugins are applied to our current project
  - *run the following Cordova command*

```
cordova build
```

Cordova app - working with plugins - part 3

update index.html

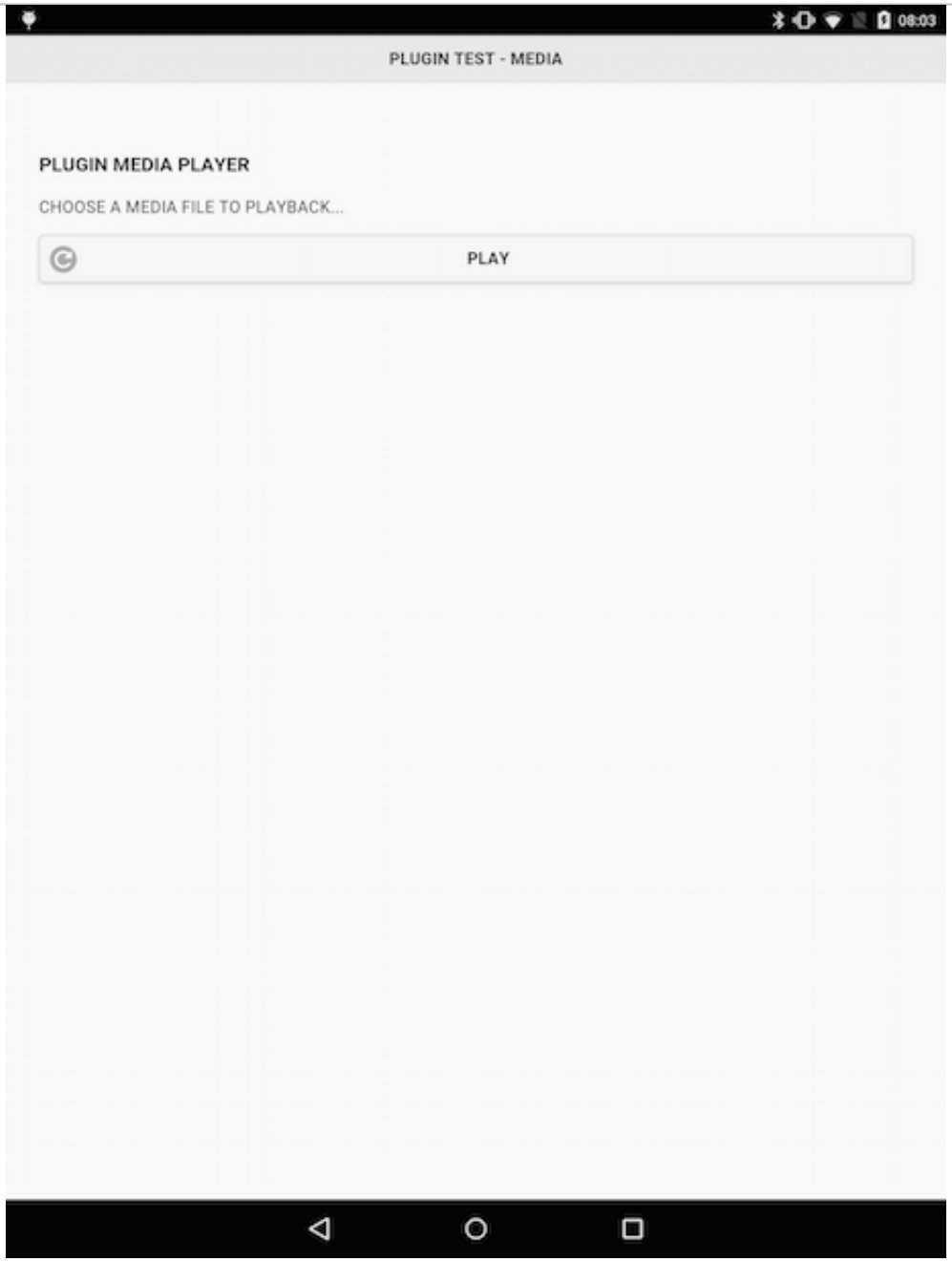
- update our `index.html` page to create the basic layout
  - *allow us to load and use media files*
- use a single page application structure
  - *include our content categories for `header` and `main`*
- add `div` with `data-role` set to `fieldcontain`
  - *signifies that we have a contiguous group of form, input elements*
- use this grouping to add our **play** button
  - *load our sample file using the installed plugins*
- use an `input` element with type set to button
  - *perhaps add an icon*

Cordova app - working with plugins - part 4

update index.html - page structure

```
<!-- homepage -->
<div data-role="page" id="home">
    <div data-role="header">
    <h3>plugin test - media</h3>
    </div><!-- /header -->
  <div role="main" class="ui-content">
    <!-- container for media options... -->
    <div data-role="content">
      <!-- group buttons etc -->
      <div data-role="fieldcontain">
        <h3>Plugin Media Player</h3>
        <p>choose a media file to playback...</p>
        <input type="button" id="playAudio" data-icon="refresh" value="Play" />
      </div>
    </div>
    </div><!-- /content -->
</div><!-- /homepage -->
```

Image - Cordova app - Plugin Test 1 - getting started



Cordova - Plugin Test - index.html

Cordova app - working with plugins - part 5

add some logic

- add some logic to our application
- updates to our JavaScript to allow us to handle events
- add handlers for listeners for each button we add to the application
  - *including the initial **play** button*
- add this code to our application's default JavaScript file
  - *stored in `assets/scripts/plugin.js`*
- setup the application in response to Cordova's `deviceready` event
  - *event informs us that installed plugins are loaded and ready for use*
- add a function for the `deviceready` event
  - *allows us to bind our handler for the tap listener on the **play** button*

```
functon onDeviceReady() {
  $("#playAudio").on("tap", function(e) {
    //add code for action...
  });
}
```

Cordova app - working with plugins - part 6

on deviceready

- add any other required, initial functions later to this same start-up function
- wrap initial function in our main application loader
  - *checks device is ready, and then adds any required handlers*

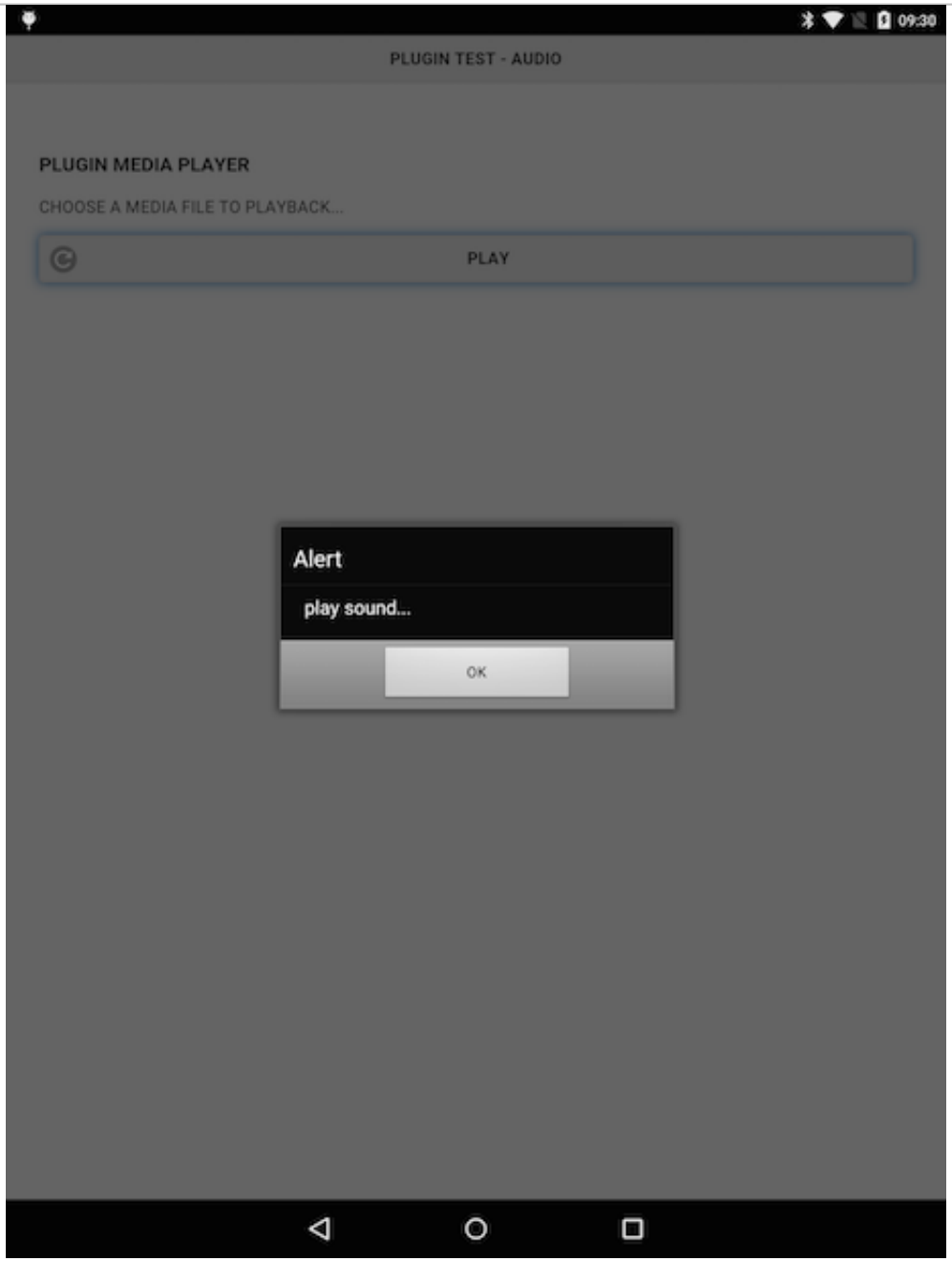
```
(function() {
    //check for page initialisation and #home
    $(document).on("pageinit", "#home", function(e) {
        //prevent any bound defaults
        e.preventDefault();

        //loader function after deviceready event returns
        function onDeviceReady() {
            //play audio
            $("#playAudio").on("tap", function(e) {
                //audio playback logic
                alert("play sound...");
            });

        }
        //as deviceready returns load onDeviceReady()
        $(document).on("deviceready", onDeviceReady);
    });

})();
```

Image - Cordova app - Plugin Test 1 - getting started



Cordova - Plugin Test - audio button

Cordova app - working with plugins - part 7

audio playback logic

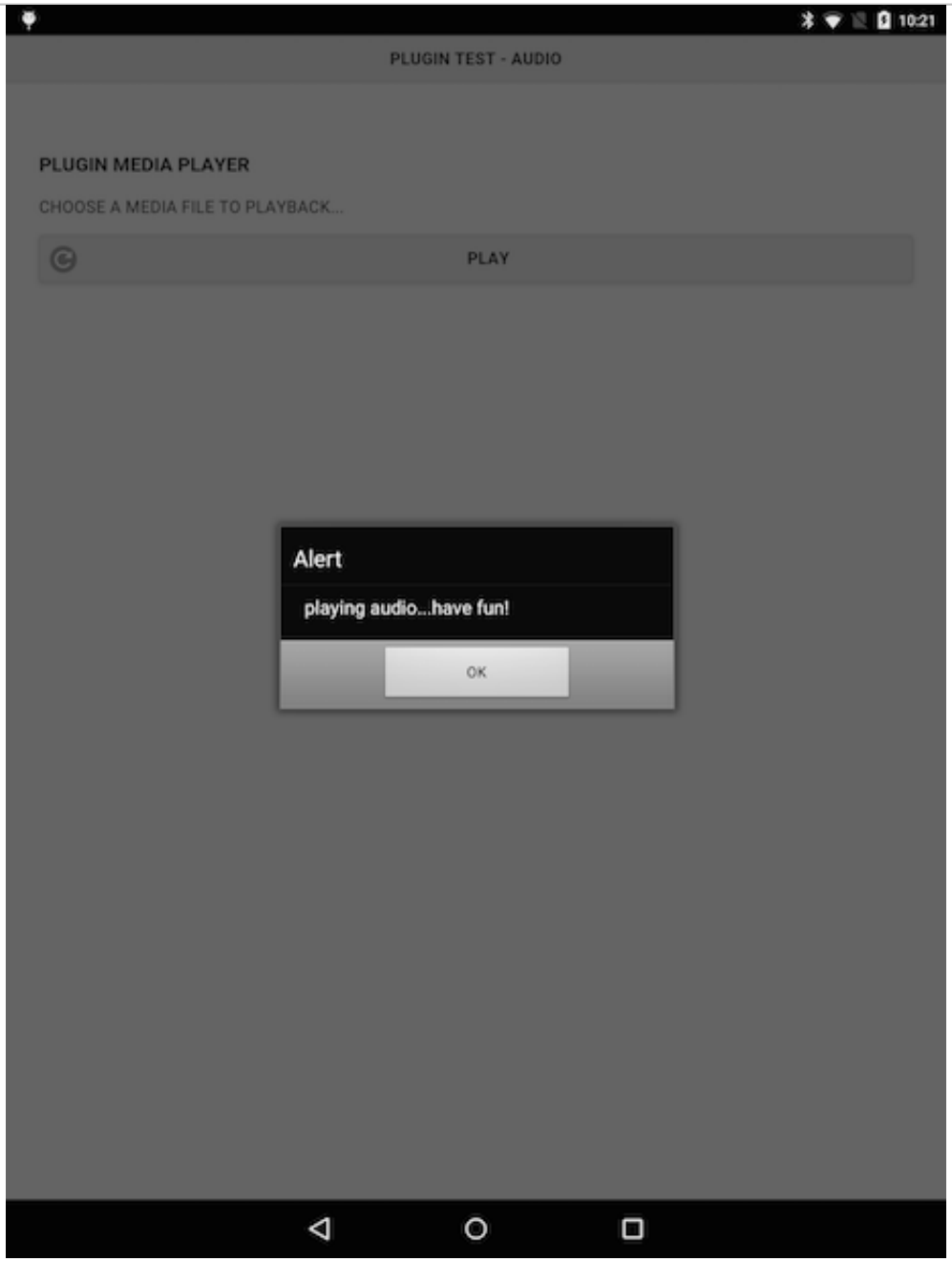
- now setup and tested the basic app logic
  - *added handlers for `deviceready` and clicking the audio playback button*
- update logic for the `#playAudio` button

```
//play audio file
function playAudio() {
  var $audioURL = buildURL("media/audio/egypt.mp3");
  var $audio = new Media($audioURL, null, errorReport);
  $audio.play();
  alert("playing audio...have fun!");
}
```

- add associated media loaders for the audio file
- add basic error checks in case the media file is missing, corrupt...

```
//build url for android
function buildURL(file) {
  if (device.platform.toLowerCase() === "android") {
    var $androidFile = "/android_asset/www/" + file;
    return $androidFile;
  }
}
//return any error message from media playback
function errorReport(error) {
  alert("Error with Audio - " + JSON.stringify(error));
}
```

Image - Cordova app - Plugin Test 1 - getting started



Cordova - Plugin Test - audio playback

Cordova app - working with plugins - part 8

add splashscreen

- add support for splashscreens in Cordova
  - *need to return to our `config.xml` file*
- set different splashscreens for different supported platforms
- specify different images to use for given screen resolutions
- Android example,

```
<platform name="android">
    <!-- you can use any density that exists in the Android project -->
    <splash src="resources/splash/android/splash-land-hdpi.png" density="land-hdpi"/>
    <splash src="resources/splash/android/splash-land-ldpi.png" density="land-ldpi"/>
    <splash src="resources/splash/android/splash-land-mdpi.png" density="land-mdpi"/>
    <splash src="resources/splash/android/splash-land-xhdpi.png" density="land-xhdpi"/>

    <splash src="resources/splash/android/splash-port-hdpi.png" density="port-hdpi"/>
    <splash src="resources/splash/android/splash-port-ldpi.png" density="port-ldpi"/>
    <splash src="resources/splash/android/splash-port-mdpi.png" density="port-mdpi"/>
    <splash src="resources/splash/android/splash-port-xhdpi.png" density="port-xhdpi"/>
</platform>
```

- specifying different images for each screen density, and then also for portrait and landscape formats
- URL for the `src` attribute is relative to the project's root directory
  - *not the customary `www`*

Cordova app - working with plugins - part 9

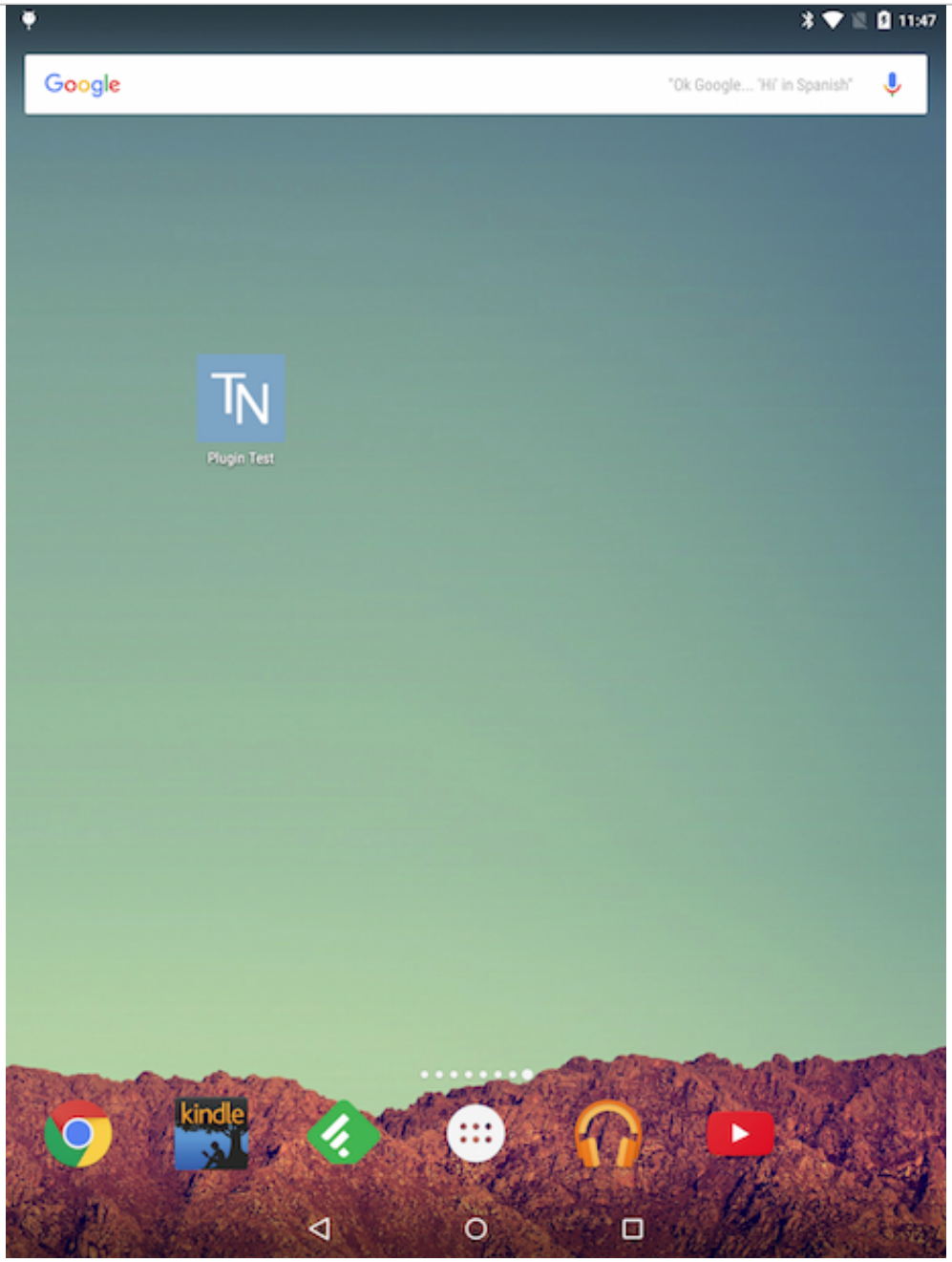
add an icon

- also set our own app's icon
  - *again in the `config.xml` setting for the application*

```
<platform name="android">
  <icon src="resources/icon/android/ldpi.png" density="ldpi" />
  <icon src="resources/icon/android/mdpi.png" density="mdpi" />
  <icon src="resources/icon/android/hdpi.png" density="hdpi" />
  <icon src="resources/icon/android/xhdpi.png" density="xhdpi" />
</platform>
```

- again, we can target specific platforms
  - *useful way to handle different screen resolutions and densities*
- icon's URL is specified relative to the project's root directory

Image - Cordova app - Plugin Test 1 - getting started



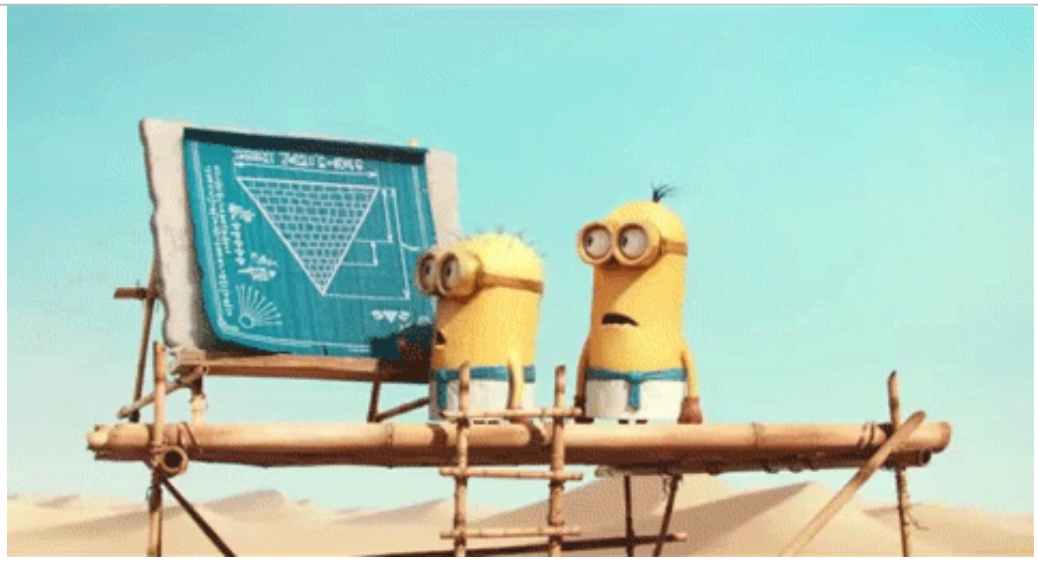
Cordova - Plugin Test - custom icon

Cordova app - working with plugins - part 9

Android icon sizes for launcher

Density	Launcher icon size
ldpi	36 x 36 px
mdpi	48 x 48 px
hdpi	72 x 72 px
xhdpi	96 x 96 px

Image - Designing our app



Designing our app - fundamentals are important

Video - Pyramid builders

Minions (2015) Pyramid

Minions Pyramid Builders - Source: YouTube

Demos

- Cordova - blueprint
- Cordova - plugintest

References

- Cordova API Documentation
  - *config.xml file*
  - *Icons and Splash screens*
  - *Plugin APIs*