PhoneGap & jQuery Mobile

Lesson 5

Thomas Mak

makzan@42games.net

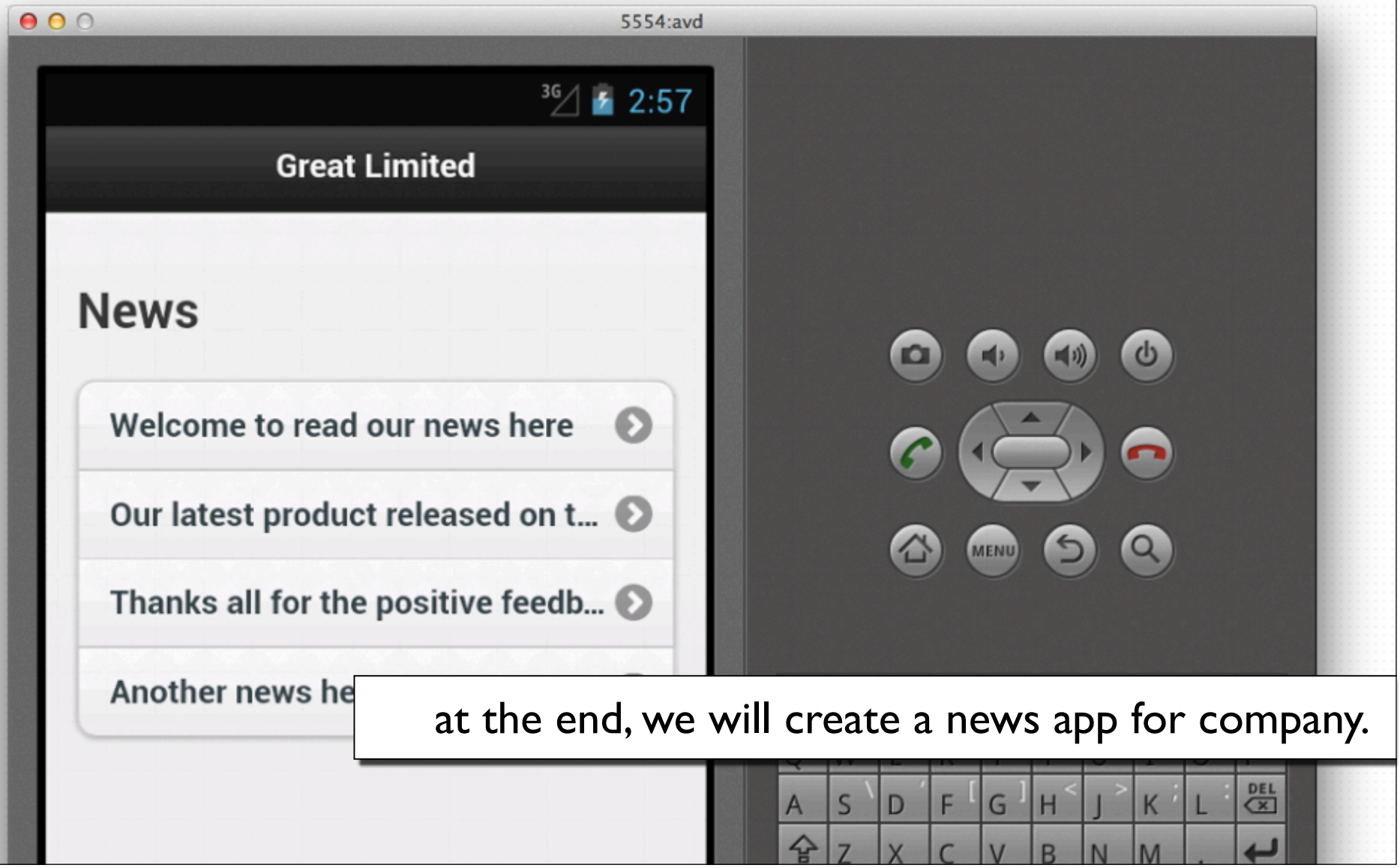
Source Codes

https://github.com/makzan/PhoneGap-Course-Examples

Today

- Load news data from S3 JSON file
- Fetch photos from Facebook Page
- Fetch current location with PhoneGap

Great Limited with News



at the end, we will create a news app for company.

Continue from last lesson

- We will continue the Great Limited project from last lesson.

Great Limited with News

In this section, we will load the JSON file from internet and display it as a list of news.

Great Limited with News

```
callback({
  "news": [
    {
      "date": "2012-08-01 15:14:13",
      "content": "Do you really think that you will read real news here? No kids, this is
just a dummy text.",
      "image": "http://placekitten.com/300/200",
      "title": "Welcome to read our news here"
    },
    {
      "date": "2012-08-10 16:12:13",
      "content": "Welcome to our another news. We are working on a mystery new product.",
      "image": "http://placekitten.com/300/200?image=8",
      "title": "Another news here"
    }
  ]
})
```

first, we will create a news.json file with the dummy news content.

Great Limited with News

http://makzan-temp.s3.amazonaws.com/news.json

we will put it somewhere on the internet so we can fetch it in the app.

Great Limited with News

```
// Internal News Module
(function(){
  var News = (function(){


    return News;
  })();

  // define it in global scope.
  if (!this.greatLtd) this.greatLtd = {}
  this.greatLtd.News = News;
}).call(this);
```

we will need a News class.

Great Limited with News

```
var News = (function(){

  function News() {}

  News.prototype.fetch = function(element) {
    $.ajax({
      url: "http://makzan-temp.s3.amazonaws.com/news.json",
      type: "GET",
      crossDomain: true,
      dataType: "jsonp",
      jsonpCallback: 'callback',
      success: function (data) {
        console.log("ajax success", data);
      },
    });

  }

  return News;
})();
```

implement the basic news fetching.

Great Limited with News

```
var News = (function(){

  function News() {}

  News.prototype.fetch = function(element) {
    $.ajax({
      url: "http://makzan-temp.s3.amazonaws.com/news.json",
      type: "GET",
      crossDomain: true,
      dataType: "jsonp",
      jsonpCallback: 'callback',
      success: function (data) {
        console.log("ajax success", data);
      },
    });

  }

  return News;
})();
```

we'll put more logic here.

Great Limited with News

```
success: (function (data) {
  this.news = data.news;

  $(element).empty();
  for (var i=0, len=data.news.length; i<len; i++)
  {
    var news = data.news[i];
    var listItem = ["<li><a href='#news-detail'>", news.title, "</a></li>"].join('');
    $(element).append(listItem);
  }
  $(element).listview('refresh');
}).bind(this),
```

when fetching success, we put the news into the element

Great Limited with News

```
success: (function (data) {
  this.news = data.news;

  $(element).empty();
  for (var i=0, len=data.news.length; i<len; i++)
  {
    var news = data.news[i];
    var listItem = ["<li><a href='#news-detail'>", news.title, "</a></li>"].join('');
    $(element).append(listItem);
  }
  $(element).listview('refresh');
}).bind(this),
```

bind allows us to pass the outer scope into the event handling method.

Great Limited with News

```
if (!Function.prototype.bind) {
  Function.prototype.bind = function (oThis) {

    var aArgs = Array.prototype.slice.call(arguments, 1),
        fToBind = this,
        fNOP = function () {},
        fBound = function () {
          return fToBind.apply(this instanceof fNOP && oThis
                                 ? this
                                 : oThis,
                               aArgs.concat(Array.prototype.slice.call(arguments)));
        };

    fNOP.prototype = this.prototype;
    fBound.prototype = new fNOP();

    return fBound;
  };
}
```

.bind is new syntax and for those old browser, we can have the above shim from Mozilla.

https://developer.mozilla.org/en-US/docs/JavaScript/Reference/Global_Objects/Function/bind

Great Limited with News

```
<!-- ////////// Home -->
<article data-role='page' id='main'>
  <header data-position='fixed' data-role='header'>
    <h1>Great Limited</h1>

  </header>
  <section data-role='content'>
    <h2>News</h2>
    <ul data-inset='true' data-role='listview' id='news-list'>

    </ul>
  </section>
</article>
```

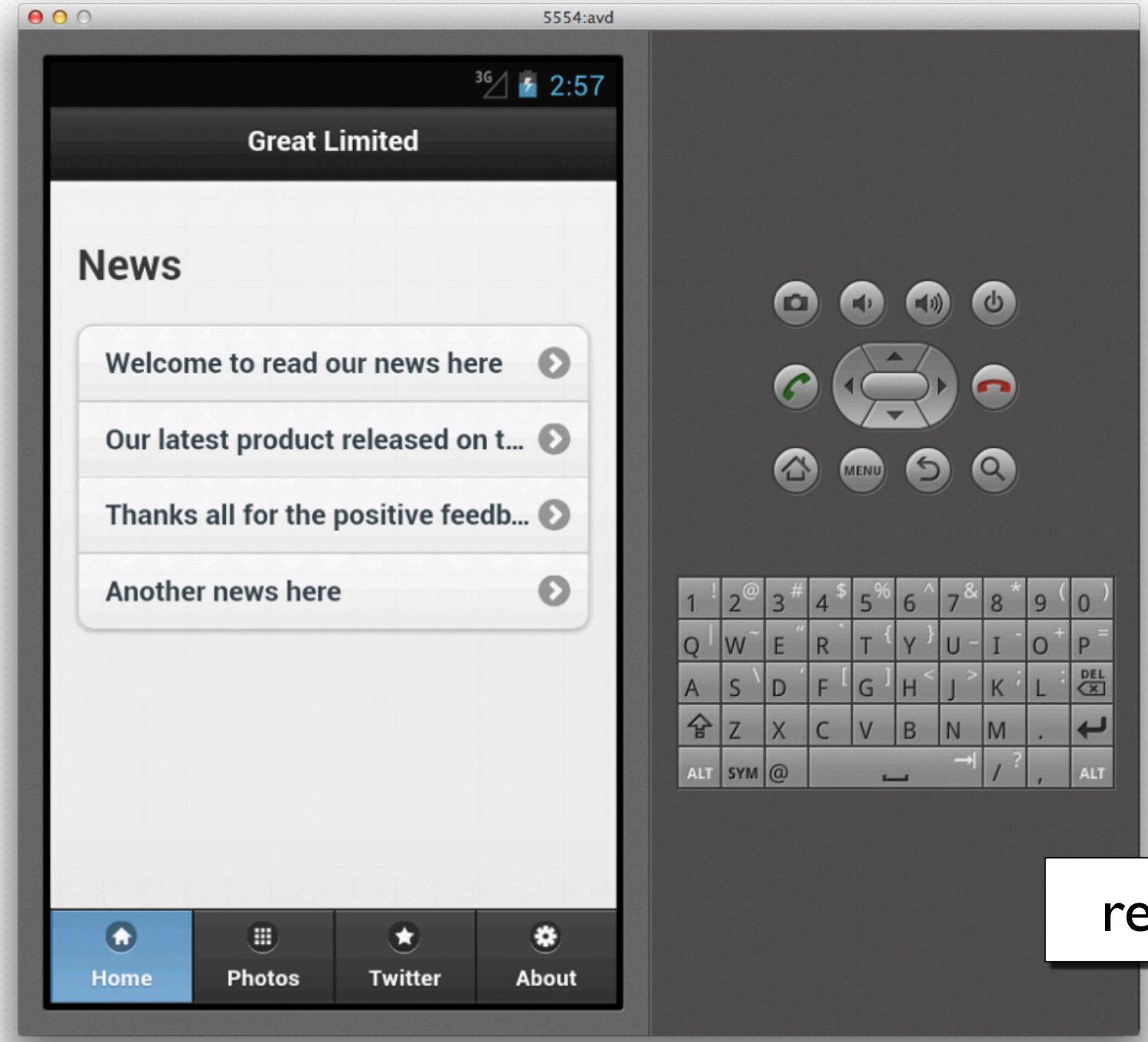
back to the index.html, we empty the #news-list.

Great Limited with News

```
$(document).bind('pagebeforeshow', function(event, ui) {
  switch ($(event.target).attr('id')) {
    case 'main':
      (new greatLtd.News()).fetch($('#news-list'));
      break;
    case 'twitter':
      var twitter = new greatLtd.TwitterQuery("Great Limited");
      twitter.fetch($("#tweet-list"));
      break;
    case 'map-page':
      $('#map').gmap('addMarker', {
        'position': '22.192362,113.54206',
        'bounds': true
      });
      break;
  }

});
```

add ‘main’ section to the pagebeforeshow event handler.

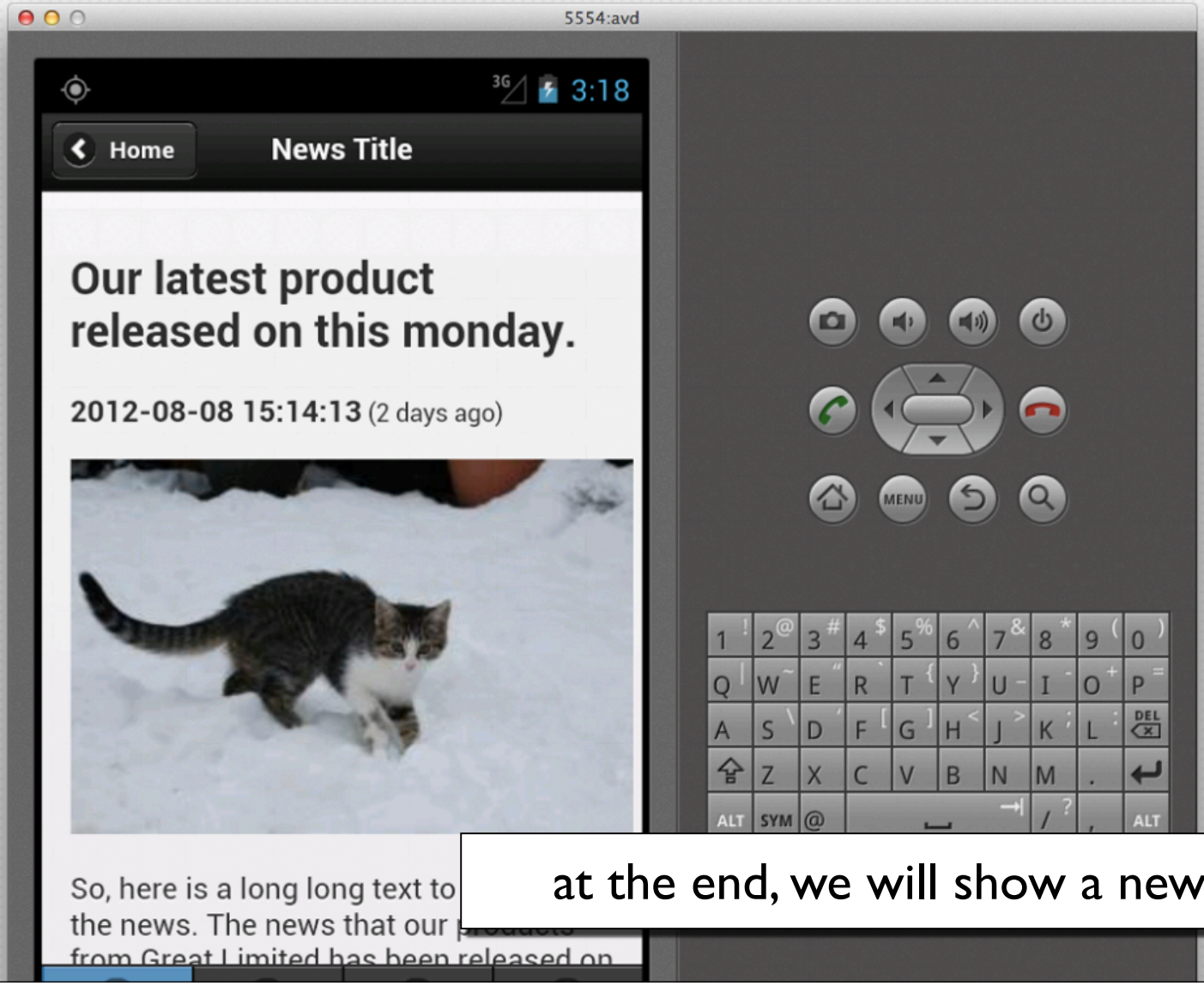

5554:avd
2:57
Great Limited
News
Welcome to read our news here
Our latest product released on t...
Thanks all for the positive feedb...
Another news here
Home
Photos
Twitter
About
MENU

result.

News Detail

In this section, we will display the news content once tapped the news title.

News Detail



at the end, we will show a news content.

News Detail

```
// alias the Classes
var TwitterQuery = greatLtd.TwitterQuery,
    News         = greatLtd.News,
```

so many classes now, we will make the alias of the classes at the beginning of jQuery ready function.

News Detail

```
var news; // store the news instance.
```

we will going to access the same News instance when loading the news detail, so we need to store it inside the jQuery ready function scope.

News Detail

```
case 'main':
  news = new News();
  news.fetch($('#news-list'));
  break;
```

last time we used shorthand to fetch the news, this time we separated it into two statements so we can store the news instance.

News Detail

```
$(element).empty();
for (var i=0, len=data.news.length; i<len; i++)
{
  var news = data.news[i];
  var listItem = ["<li><a data-news-id=", i, " href='#news-detail'>", news.title, "</a></
li>"].join('');
  $(element).append(listItem);
}
$(element).listview('refresh');
```

add data-news-id to store the index of the news inside the DOM directly.

News Detail

```
<div id=‘test’ data-hello-world=‘something’></div>

<script>
$(“#test”).data(“helloWorld”); // get ‘something’
</script>
```

a quick guide on use data-* attribute with jQuery.

News Detail

```
// listen to the a href=#news-detail link to get the news ID.
$("a[href='#news-detail']").live('vclick', function(event){

  var newsId = $(this).data('newsId');

  console.log ("news id:", newsId);

  // load the news.
  // cache the content element.
  var contentElement = $("#news-detail > section[data-role='content']");
  var newsData = news.news[newsId];
  if (newsData === undefined)
  {
    contentElement.html('Error loading news content.');
  }
  else
  {
    var newsText = "<h2>" + newsData.title + "</h2>";
    newsText    += "<strong>" + newsData.date + "</strong>";
    if (newsDat
      newsText
    newsText
    contentEle
  }
});
```

and finally, we change the news detail content according to which news title we clicked.

News Detail

```
// listen to the a href=#news-detail link to get the news ID.
$("a[href='#news-detail']").live('vclick', function(event){

  var newsId = $(this).data('newsId');

  console.log ("news id:", newsId);

  // load the news.
  // cache the content element.
  var contentElement = $("#news-detail > section[data-role='content']");
  var newsData = news.news[newsId];
  if (newsData === undefined)
  {
    contentElement.html('Error loading news content.');
  }
  else
  {
    var newsTe
    newsText
    if (newsDa
      newsText
    newsText
    contentEle
  }
});
```

a[href='#news-detail'] for the news title link.
.live() for the dynamically created list.
vclick from jQuery Mobile for the click event.

News Detail

```
// listen to the a href=#news-detail link to get the news ID.
$("a[href='#news-detail']").live('vclick', function(event){

  var newsId = $(this).data('newsId');

  console.log ("news id:", newsId);

  // load the news.
  // cache the content element.
  var contentElement = $("#news-detail > section[data-role='content']");
  var newsData = news.news[newsId];
  if (newsData === undefined)
  {
    contentElement.html('Error loading news content.');
  }
  else
  {
    var newsText
    newsText    +
    if (newsData.
      newsText  +
    newsText    +
    contentElement.html(newsText);
  }
});
```

the jQuery data() method to get data-* value from DOM.

News Detail

```
// listen to the a href=#news-detail link to get the news ID.
$("a[href='#news-detail']").live('vclick', function(event){

  var newsId = $(this).data('newsId');

  console.log ("news id:", newsId);

  // load the news.
  // cache the content element.
  var contentElement = $("#news-detail > section[data-role='content']");
  var newsData = news.news[newsId];
  if (newsData === undefined)
  {
    contentElement.html('Error loading news content.');
  }
  else
  {
    var newsText = "<h2>" + newsData.title + "</h2>";
    newsText    += "<strong>" + newsData.date + "</strong>";
    if (newsData.image != undefined)
      newsText  += "<p><img src='" + newsData.image + "'></p>";
    newsText    += "<p>" + newsData.content + "</p>";
    contentElement.html(newsText);
  }
});
```

construct the news block.

News Detail

Moment.js | Parse, validate, manipulate, and display dates in javascript.

momentjs.com

Moment.js Documentation Unit Tests Follow @momentjs on twitter Star 2,765

Fork me on GitHub

Moment.js

A lightweight (4.3k) javascript date library for parsing, manipulating, and formatting dates.

Get the current version 1.6.2

```
npm install moment
```

Production (4.3k min & gzip)

Development (full 31.2k source)

Tweet

Formatting dates

```
moment().format('dddd, MMMM Do YYYY, h:mm:ss a');
moment().format('dddd [on the] wo [week of the year]');
moment().format("MMM Do 'YY");
moment().format(); // defaults to ISO-8601
```

Thursday, August 9th 2012, 9:34:58 pm

Thursday on the 32nd week of t

Aug 9th '12

2012-08-09T21:34:58+08:00

Calendar Time

```
moment().subtract('days', 3).calendar();
```

English : last Monday at 9:34 PM

Catalan : el Dilluns passat a les 21:34

Danish : sidste Mandag kl 9:34 PM

Timeago

```
moment("2011-10-31", "YYYY-
```

English : 9 months ago

Catalan : fa 9 mesos

Norwegian : Forrige mandag klokken 21:34

Dutch : afgelopen maandag om 21:34

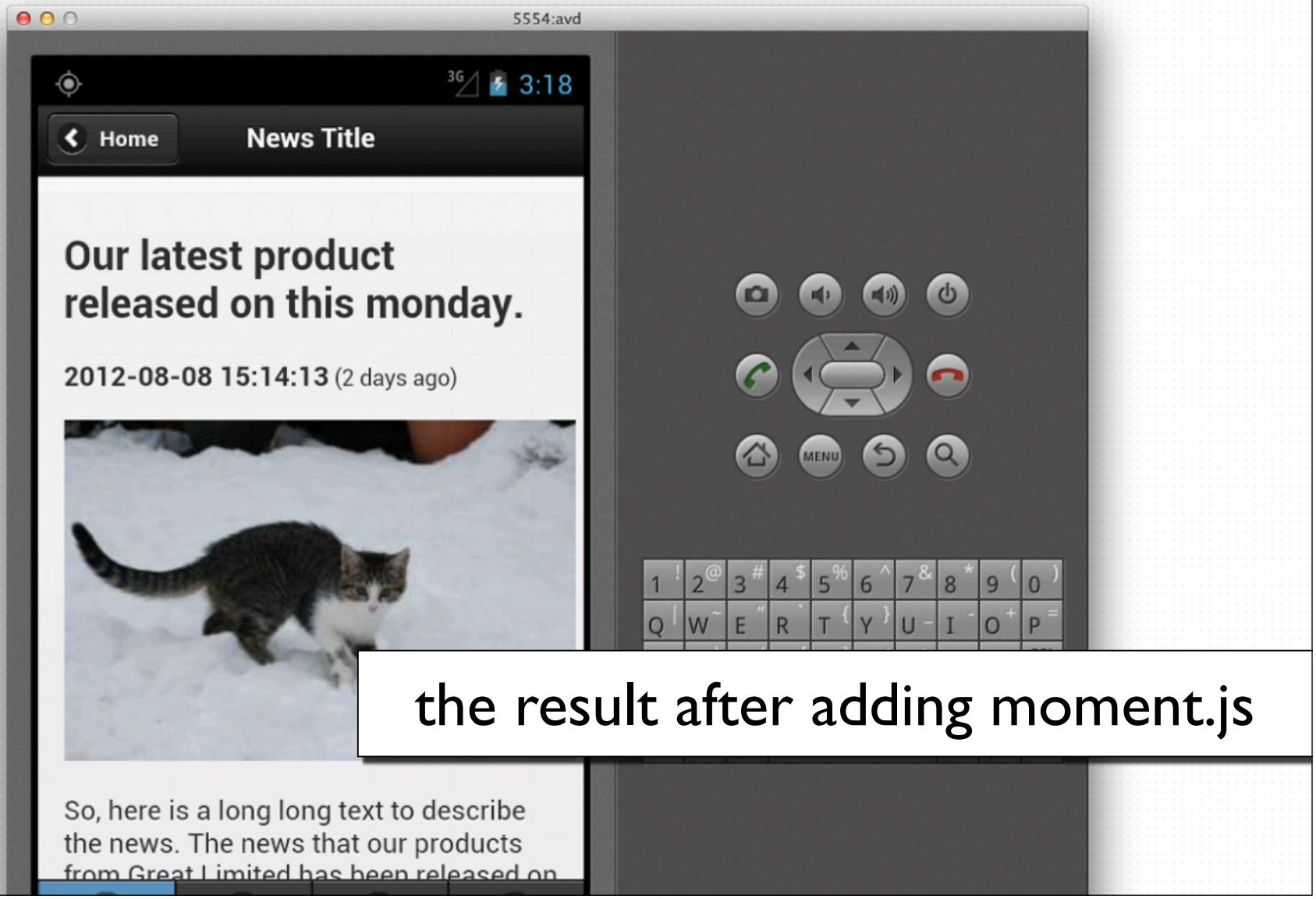
before we move on, let's enhance the news date with moment.js

News Detail

```
var newsText = "<h2>" + newsData.title + "</h2>";
newsText    += "<strong>" + newsData.date + "</strong>";
newsText    += "<small> (" + moment(newsData.date, 'YYYY-MM-DD hh:mm:ss').fromNow() + ") </
small>";
if (newsData.image != undefined)
  newsText  += "<p><img src='" + newsData.image + "'></p>";
newsText    += "<p>" + newsData.content + "</p>";
contentElement.html(newsText);
```

now we use moment() to show how long ago the news is, in human format.

News Detail

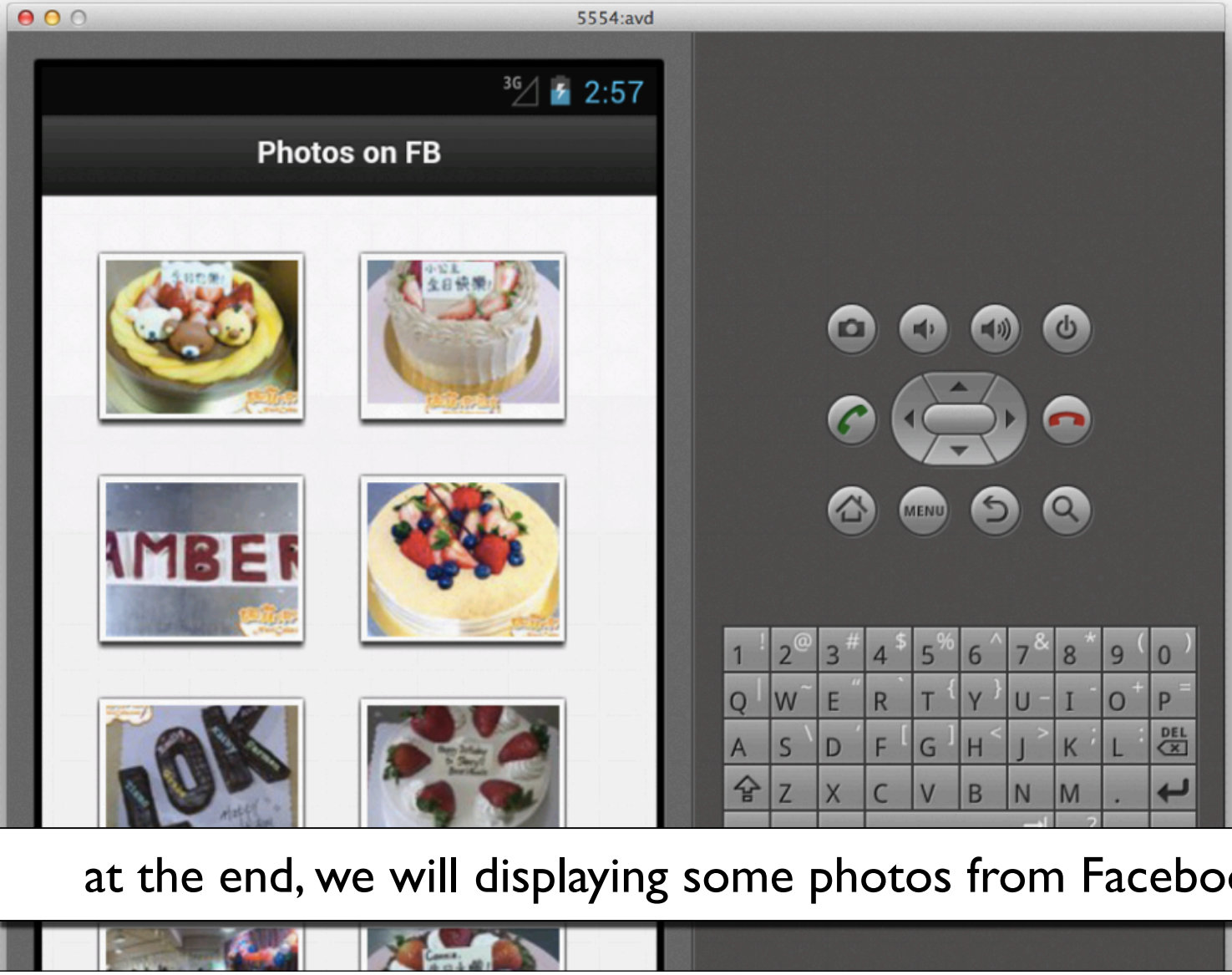


the result after adding moment.js

Fetch photos from Facebook

In this section, we will fetch the public photos of a Facebook page.

Fetch photos from Facebook



at the end, we will displaying some photos from Facebook Page.

Fetch photos from Facebook

```
<!-- /////////// Photo Page -->
<article data-role='page' id='facebook'>
  <header data-position='fixed' data-role='header'>
    <h1>Photos on FB</h1>
  </header>
  <section data-role='content'>
    <div id='photos'>
    </div>
  </section>
</article>
```

let's create one more page for the facebook photo in index.html.

Fetch photos from Facebook

```
<!-- /////////// Templates -->
<div class='footer-template'>
  <footer data-id='footer' data-position='fixed' data-role='footer'>
    <div data-role='navbar'>
      <ul>
        <li class='first-page'>
          <a data-icon='home' data-transition='none' href='#main'>Home</a>
        </li>
        <li class="facebook-page">
          <a data-icon='grid' data-transition='none' href='#facebook'>Photos</a>
        </li>
        <li class='second-page'>
          <a data-icon='star' data-transition='none' href='#twitter'>Twitter</a>
        </li>
        <li class='third-page'>
          <a data-icon='gear' data-transition='none' href='#about'>About</a>
        </li>
      </ul>
    </div>
  </footer>
</div>
```

add one more tab to our footer.

Fetch photos from Facebook

```
// footer active state
var target = 'footer .first-page a';
switch ($(this).attr('id')) {
  case 'twitter'  : target = 'footer .second-page a'; break;
  case 'about'    :
  case 'map-page' : target = 'footer .third-page a'; break;
  case 'facebook' : target = 'footer .facebook-page a'; break;
}
```

let's refactor our footer logic and add the facebook page.

Fetch photos from Facebook

```
The Facebook Class:
// Facebook Page Photo Module
(function() {
  var FacebookPhotos = (function(){
    function FacebookPhotos(){}

    FacebookPhotos.prototype.fetch = function(){
      // some logic here.
    }

    return FacebookPhotos;
  })();

  // export it in global scope.
  if (!this.greatLtd) this.greatLtd = {}
  this.greatLtd.FacebookPhotos = FacebookPhotos;

}).call(this);
```

the skeleton of the class to fetch Facebook photos.

Fetch photos from Facebook

```
FacebookPhotos.prototype.fetch = function(){
  $.ajax({
    url: 'http://graph.facebook.com/159679440710689/photos',
    type: 'GET',
    crossDomain: true,
    jsonp: 'callback',
    jsonpCallback: 'callback',
    dataType: 'jsonp',
    success: function (data) {
      console.log('from fb: ', data);
      for(var i=0, len=data.data.length; i<len; i++)
      {
        var image = data.data[i];
        $('#photos').append(["<div class='photo' style='background-image:url(",
image.picture, ")'></div>"].join(''));
      }
    }
  })
}
```

the core fetching logic.

Fetch photos from Facebook

```
/****** Facebook Photos ******/
.photo {
  background-position: 50% 50%;
  width: 100px;
  height: 80px;

  overflow: hidden;

  float: left;
  margin: 5px;
  border: 3px solid white;
  box-shadow: 0 3px 3px #333, 0 0 3px black;
}
```

add some style to our photo.

Fetch photos from Facebook

```
// alias the Classes
var TwitterQuery   = greatLtd.TwitterQuery,
    News           = greatLtd.News,
    FacebookPhotos = greatLtd.FacebookPhotos;


...


case 'facebook':
  (new FacebookPhotos()).fetch();
  break;
```

and finally, in our jQuery ready function, we fetch the Facebook photos like we fetched the news.

Viewing one photo

Viewing one photo

```
<!-- /////////// View Photo Page -->
<article data-role='page' id='photo-view'>
  <header data-position='fixed' data-role='header'>
    <h1>Photo</h1>
    <a data-icon='arrow-l' data-rel='back'>Photos</a>
  </header>
  <section data-role='content'>
    <div id='photo'></div>
  </section>
</article>
```

let's add one more page for the photo viewing.

Viewing one photo

```
// footer active state
var target = 'footer .first-page a';
switch ($(this).attr('id')) {
  case 'twitter'    : target = 'footer .second-page a'; break;
  case 'about'      :
  case 'map-page'   : target = 'footer .third-page a'; break;
  case 'photo-view' :
  case 'facebook'   : target = 'footer .facebook-page a'; break;
}
```

be sure toe update the footer active state logic.

Viewing one photo

```
// FacebookPhotos $.ajax success callback
success: (function (data) {
  this.data = data.data;
  console.log('from fb: ', data);
  for(var i=0, len=data.data.length; i<len; i++)
  {
    var image = data.data[i];
    $('#photos').append(["<a data-photo-id=", i, " href='#photo-view'><div class='photo'
style='background-image:url(", image.picture, ")'></div></a>"].join(''));
  }
}).bind(this)
```

now we add data-photo-id to our photo link.

Viewing one photo

```
var fbPhotos; // store the FacebookPhotos instsance.
```

as same as news module, we store the FacebookPhotos instance inside the jQuery ready scope.

Viewing one photo

```
case 'facebook':
  fbPhotos = new FacebookPhotos();
  fbPhotos.fetch();
  break;
```

and we separate the FacebookPhotos fetching into two statements.

Viewing one photo

```
// listen to the a href=#photo-view link to get the photo ID.
$("a[href='#photo-view']").live('vclick', function(event){
  var photoId = $(this).data('photoId');

  console.log (photoId);
  console.log (fbPhotos);

  // load the photo data
  var contentElement = $("#photo");
  var photoData = fbPhotos.data[photoId];
  if (photoData === undefined)
  {
    contentElement.html("Error loading the photo.");
  }
  else
  {
    var text = "<img src='" + photoData.images[2].source + "' alt='photo' class='big-
photo'>";
    contentElement.html(text);
  }
});
```

we know it is images[2].source from inspecting the response of Facebook API.

Elements Resources Network Sources Timeline Profiles Audits Console Search Console

```
from fb:  ▼Object                                                                                                                                app.js:61
            ▼data: Array[25]
              ▼0: Object
                created_time: "2012-07-26T04:31:24+0000"
                ▶from: Object
                height: 534
                icon: "http://static.ak.fbcdn.net/rsrc.php/v2/yz/r/StEh3RhPvjk.gif"
                id: "475795205765776"
                ▼images: Array[8]
                  ▶0: Object
                  ▶1: Object
                  ▼2: Object
                    height: 534
                    source: "http://a5.sphotos.ak.fbcdn.net/hphotos-ak-ash4/s720x720/306705_475795205765776_1678716378_n.jpg"
                    width: 720
                    ▶__proto__: Object
                  ▶3: Object
                  ▶4: Object
                  ▶5: Object
                  ▶6: Object
                  ▶7: Object
                  length: 8
                  ▶__proto__: Array[0]
                ▶likes: Object
                link: "https://www.facebook.com/photo.php?fbid=475795205765776&set=a.159679440710689.37198.144090562269577&type=1"
                name: "無糖朱古力蛋糕"
                picture: "http://photos-e.ak.fbcdn.net/hphotos-ak-ash4/306705_475795205765776_1678716378_s.jpg"
                position: 1
                source: "http://a5.sphotos.ak.fbcdn.net/hphotos-ak-ash4/s720x720/306705_475795205765776_1678716378_n.jpg"
                updated_time: "2012-07-26T04:31:25+0000"
                width: 720
                ▶__proto__: Object
              ▶1: Object
              ▶2: Object
              ▶3: Object
              ▶4: Object
              ▶5: Object
              ▶6: Object
              ▶7: Object
              ▶8: Object
              ▶9: Object
              ▶10: Object
              ▶11: Object
              ▶12: Object
              ▶13: Object
```

<top frame> <page context> All Errors Warnings Logs

Viewing one photo

```
#photo {
  text-align: center;
}

.big-photo {
  max-width: 90%;

  border: 10px solid white;
  box-shadow: 0 3px 3px #333, 0 0 3px black;
}
```

several more lines to make it looks better.

Viewing one photo

```
var text = "<div><img src='" + photoData.images[2].source + "' alt='photo' class='big-
photo'></div>";
text    += "<p>" + photoData.likes.data.length + " likes.</p>";
contentElement.html(text);
```

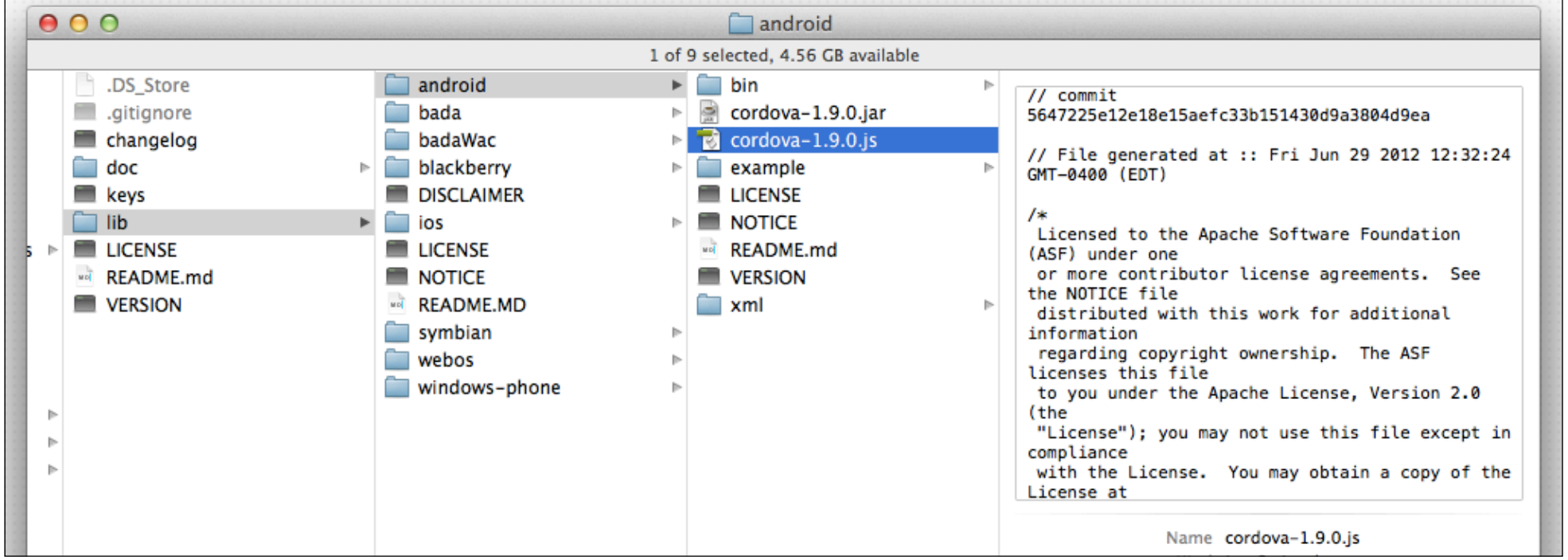
what's more?
we can show how many people like the photo.

PhoneGap Module

In this section, we will starting using the PhoneGap JavaScript.

PhoneGap Module

Cordova-1.9.0.js



PhoneGap Module

```
<script src='cordova-1.9.0.js'></script>
```

first, we need to include cordova.js into index.html

PhoneGap Module

```
// PhoneGap Module
(function(){
  var PhoneGapManager = (function(){
    function PhoneGapManager() {
      this.isReady = false;

      // Note that we need to loop the PhoneGap.available property if we need support on
BlackBerry <5.0
      document.addEventListener("deviceready", (function(){
        this.isReady = true;
        alert('ready');
      }).bind(this), false);
    }

    return PhoneGapManager;

  })();

  // export it in global scope.
  if (!this.greatLtd) this.greatLtd = {}
  this.greatLtd.PhoneGapManager = PhoneGapManager;

}).call(this);
```

one more class, PhoneGapManager.

PhoneGap Module

```
// PhoneGap Module
(function(){
  var PhoneGapManager = (function(){
    function PhoneGapManager() {
      this.isReady = false;

      // Note that we need to loop the PhoneGap.available property if we need support on
BlackBerry <5.0
      document.addEventListener("deviceready", (function(){
        this.isReady = true;
        alert('ready');
      }).bind(this), false);
    }

    return PhoneGapManager;

  })();

  // export it in global scope.
  if (!this.greatLtd
  this.greatLtd.Phon

}).call(this);
```

deviceready is the flag that we can call any PhoneGap API.

PhoneGap Module

```
// alias the Classes
var TwitterQuery    = greatLtd.TwitterQuery,
    News            = greatLtd.News,
    FacebookPhotos  = greatLtd.FacebookPhotos;
    PhoneGapManager = greatLtd.PhoneGapManager;

var news; // store the news instance.
var fbPhotos; // store the FacebookPhotos instsance.
var phoneGapManager; // store the PhoneGapManager instance.
```

and one more class to be aliased and stored.

PhoneGap Module

```
// init the phoneGapManager
phoneGapManager = new PhoneGapManager();
```

inside the jQuery ready function, we start the PhoneGapManager.

Locating Myself

```
PhoneGapManager.prototype.locateMyself = function() {
    var options = {
      maximumAge: 3000,
      timeout: 5000,
      enableHighAccuracy: true
    };
    navigator.geolocation.getCurrentPosition(function(position){
      console.log ('position:'+ position);
      console.log ('position:' + position.coords.latitude);
      console.log ('position:' + position.coords.longitude);
      alert("Location: " + position.coords.latitude + ", "position.coords.longitude)
    },
    function(){
      alert("Location fetching error.")
    }, options);
  }

  return PhoneGapManager;
})();
```

next, we call our PhoneGap API to get the current location and alert the lat/lng.

Locating Myself

```
function PhoneGapManager() {
  this.isReady = false;

  this.location = {
    coords: {
      latitude: 0,
      longitude: 0
    }
  }
  ...
}
```

incase we cannot get the location, we will init the location with 0 value.

Locating Myself

```
navigator.geolocation.getCurrentPosition((function(position){
  this.location = position;
}).bind(this),
```

instead of the alert box, now we store location for later use in map.

Locating Myself

```
case 'map-page':
  $('#map').gmap('addMarker', {
    'position': '22.192362,113.54206',
    'bounds': true
  });
  $('#map').gmap('addMarker', {
    'position': phoneGapManager.location.coords.latitude
+','+phoneGapManager.location.coords.longitude,
    'bounds': true
  });
  break;
```

now when showing the map, we add the current position to the map.

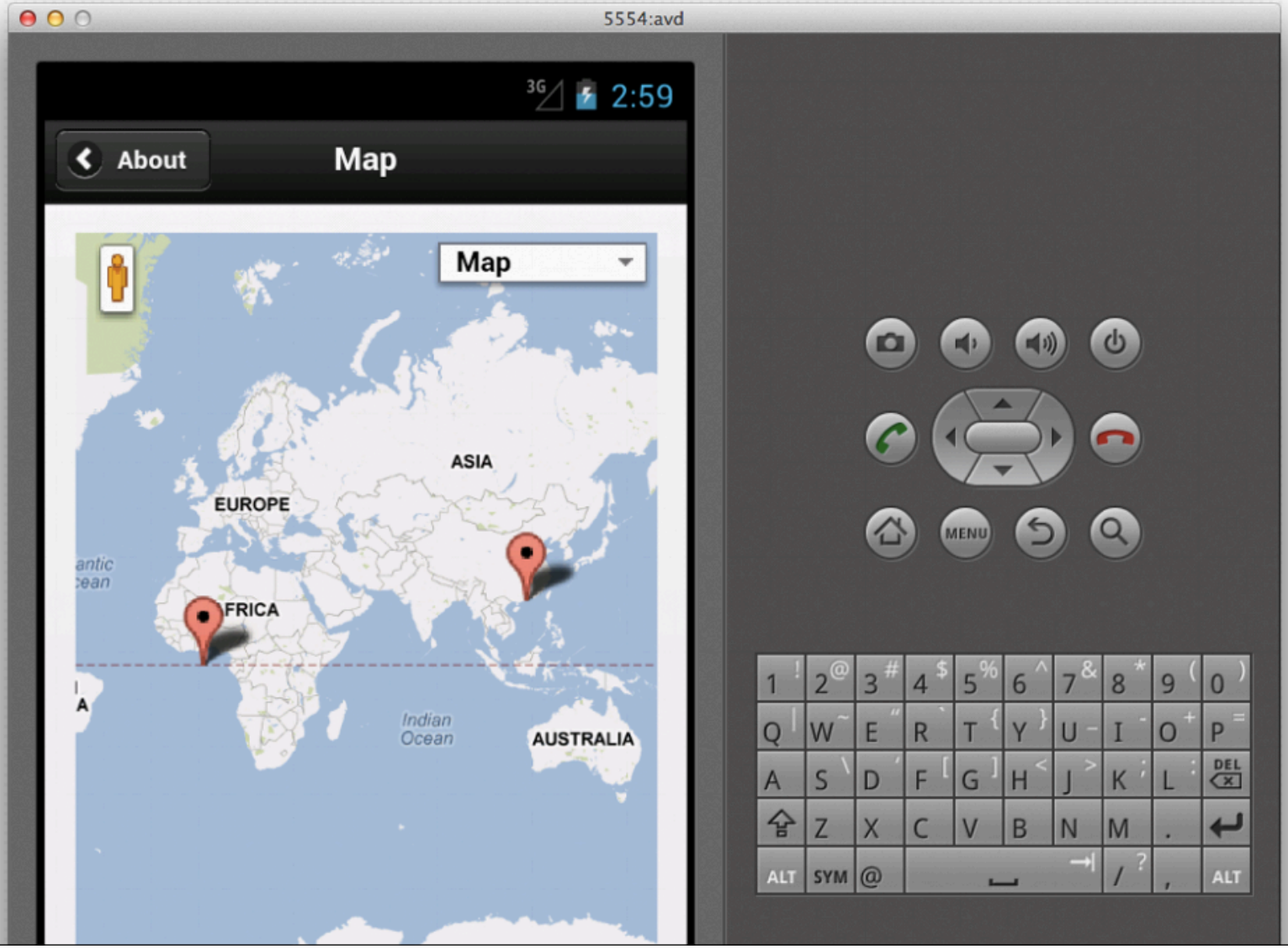
Locating Myself

```
case 'map-page':
  $('#map').gmap('addMarker', {
    'position': '22.192362,113.54206',
    'bounds': true
  });
  $('#map').gmap('addMarker', {
    'position': phoneGapManager.location.coords.latitude
+','+phoneGapManager.location.coords.longitude,
    'title' : 'Current Position',
    'icon' : 'http://makzan-temp.s3.amazonaws.com/male.png',
    'bounds': true
  });
  break;
```

what's more?

we can custom the map marker with a png file.

Locating Myself



Locating Myself

map icons may be found here
http://mapicons.nicolasmollet.com/

Locating Myself

two ways to set the location in Android emulator:

option 1:

1. go to your android/tools directory
2. launch the DDMS tool
3. select the Emulator Control Tab
4. fill in the longitude and latitude values
5. press the Send button

Locating Myself

two ways to set the location in Android emulator:

option 2:

1. open a command shell
2. connect to the emulator with the command:
   telnet localhost <port>
3. to enter a fixed location execute the command:
4. geo fix <longitude> <latitude> [<altitude>]