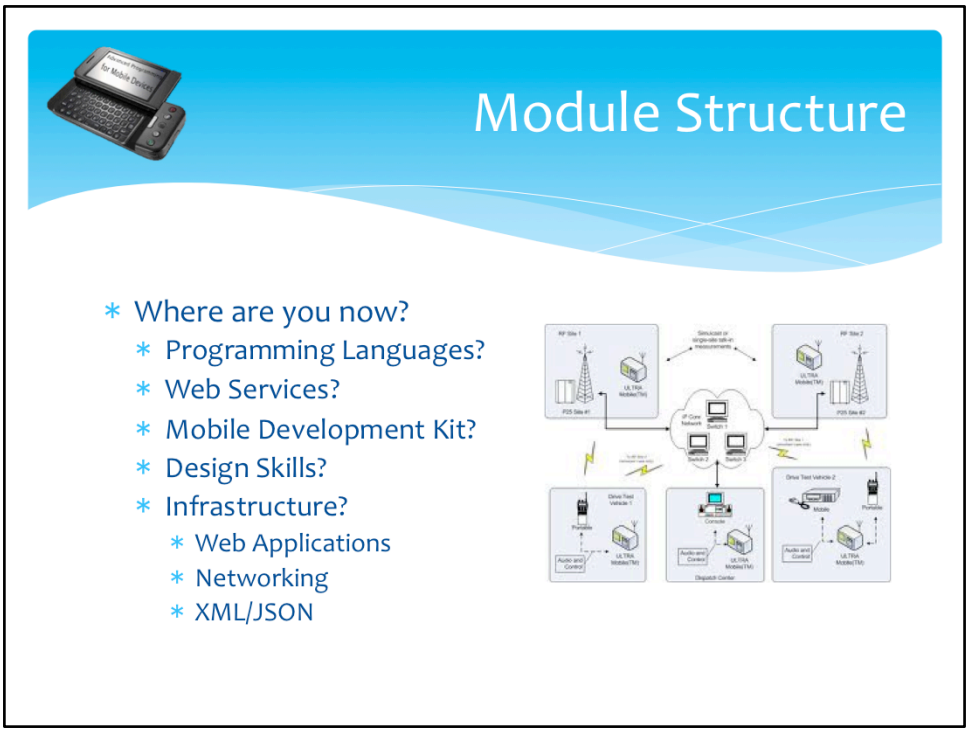
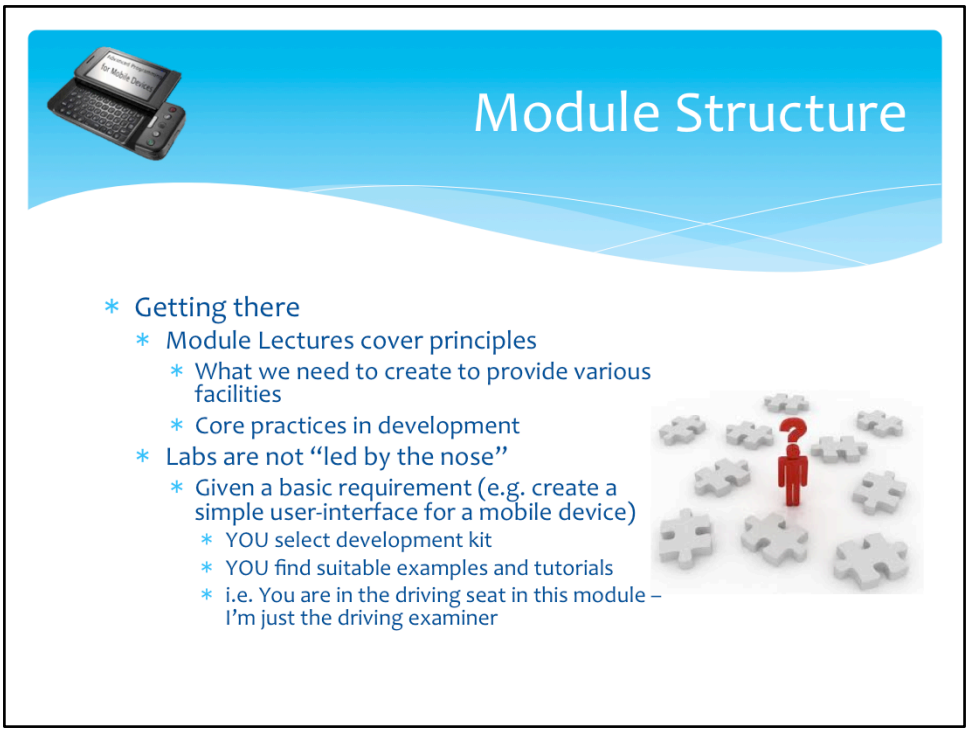
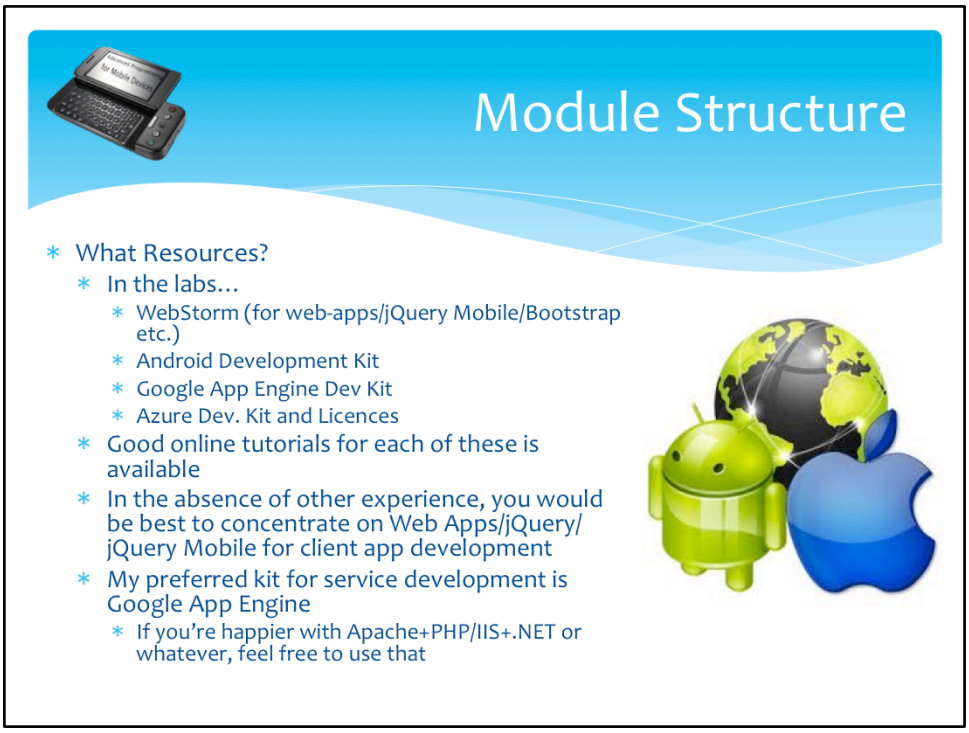


Good to start a route by knowing where you are.

I'm not being prescriptive about what technologies you use in this module. You'll be given goals to meet, you choose the tech that can meet them.



Often this will involve you going through online tutorials for a specific technology (e.g. the jQuery Mobile ones).

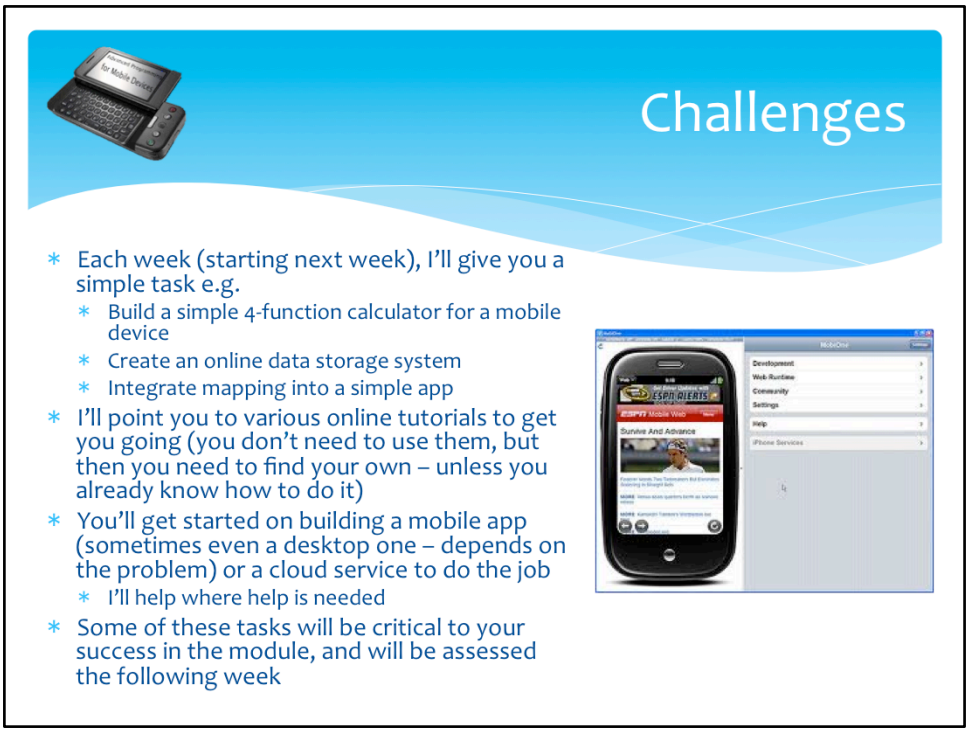


See online sources:

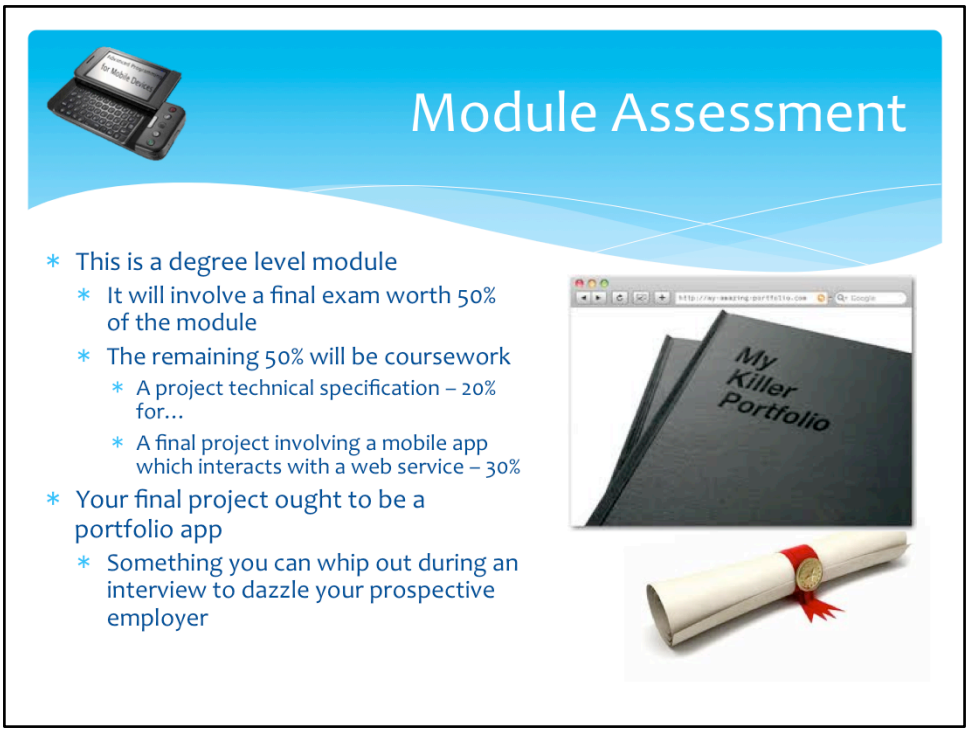
www.jquerymobile.com

www.google.com (app engine)

etc.



Aaarrggghhh..



Even more Aaaarrrrggghhhh!!!!

Questions?

* Any questions about
  * Lectures?
  * Labs?
  * Assessment?

Hopeful parents

Mobile Application Characteristics

NOT – small versions of desktop apps

Even when they are – they generally have different design objectives

Mobility is the key characteristic

Apps respond to geographical location, diary entries, messages,

Assistive technology is common (PDA=Personal Digital "Assistant")

Location awareness is an obvious feature

Information retrieval

Integration of local information (on device) with online data - e.g. where am I? (local) and where is the nearest station? (online)

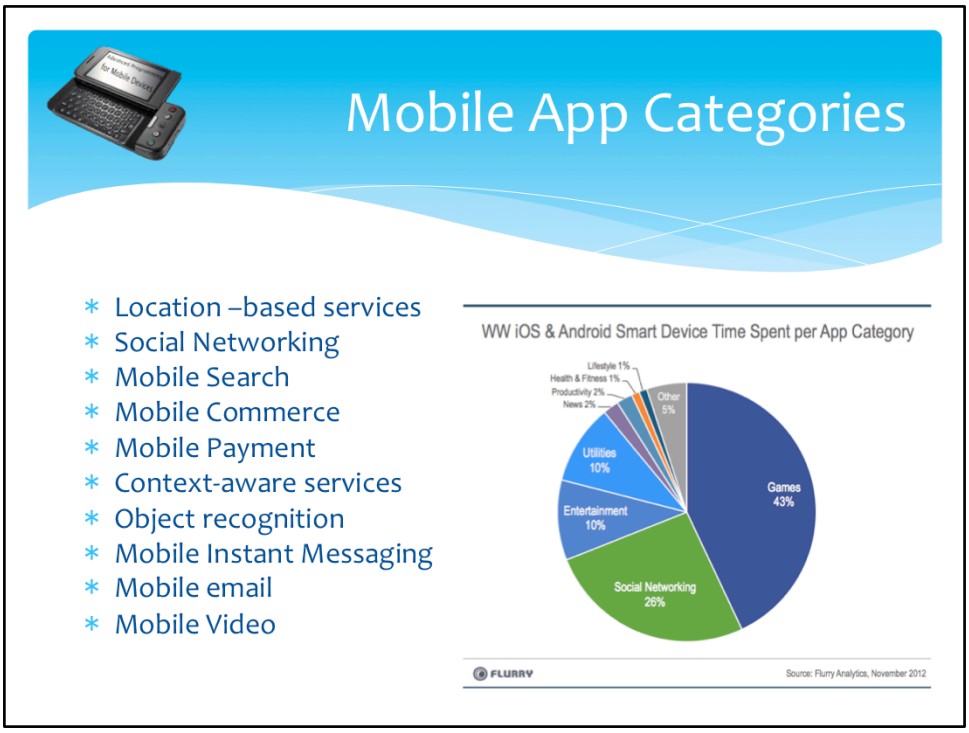
Apps can be always there – a useful feature for many tasks

Business assistance – reminders, info on tap, calculations, push notifications

Medical assistance (e.g. remember to take your pills, check blood pressure, etc.)

Communications – exchange key info with colleagues

Short-term storage – e.g. store information gathered on the road for later upload



Location –based services
- Mapping, Directions
- Use device GPS to direct downloading of map-segments, calculation of routes

Social Networking
- Facebook, Twitter
- Simplified mobile client apps

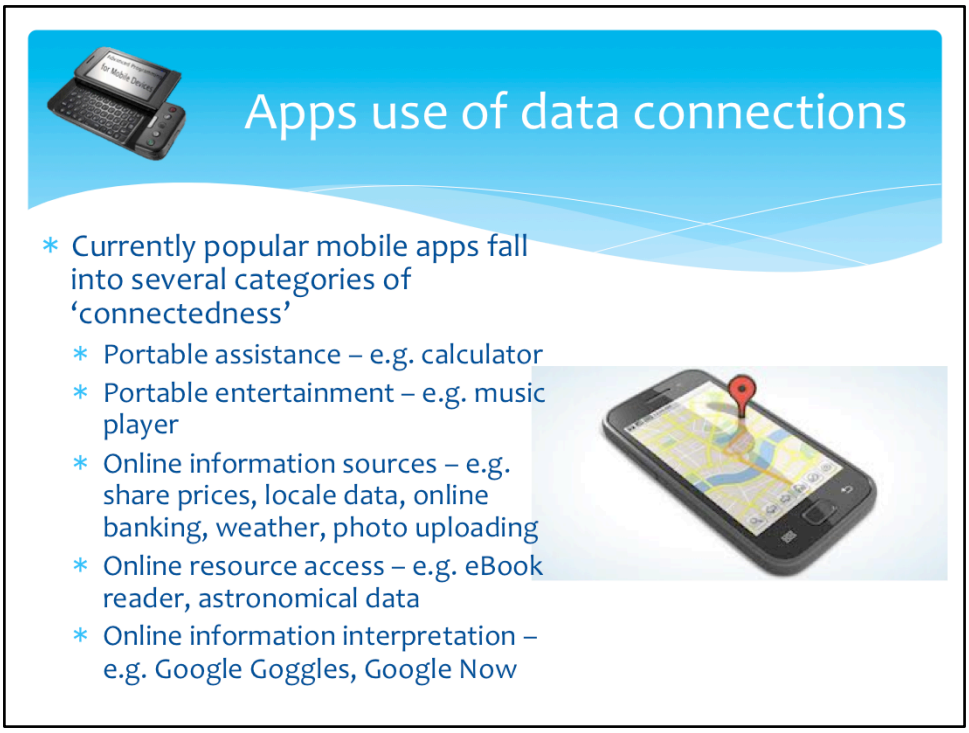
Mobile Search
- Around me
- Use locale information (from GPS) to perform online database search

Mobile Commerce
- Amazon mobile app
- Shopping anywhere

Mobile Payment
- NFC
- Interact with NFC terminals (e.g. fares, small item purchases)

Context-aware services
- Google Now
- Integrate device data (e.g. diary, contacts) with locale and online data (e.g. current location, train timetables) to provide timely information

Object recognition
- Google Goggles, Augmented Reality



Currently popular mobile apps fall into several categories of 'usefulness'

Portable assistance – e.g. calculators (normal or task –specific). e.g. a calculator, a tax calculator, photo editing or management

No (or little) need to access online data

Portable entertainment apps – e.g. music/video player - no need for a permanent connection

Online information sources – e.g. share prices, locale data, online banking, weather, photo uploading (e.g. to Instagram/Flickr)

An internet connection is essential for the app to do its job

The app maintains an online presence on the move

Online resource access – e.g. eBook reader, astronomical data

An internet connection is needed to access information, but this can then be stored locally on the device

The app accesses online data stores, but caches the data for mobile use

Online information interpretation – e.g. Google Goggles, Google Now

The app integrates online data with device information (e.g. locale, PIM etc.) to assist in personal scheduling/information retrieval

Infrastructure is whatever external resource is needed to make a mobile app possible.

At its most minimal, this can be a delivery site (e.g. the Apple App Store or Google Play)

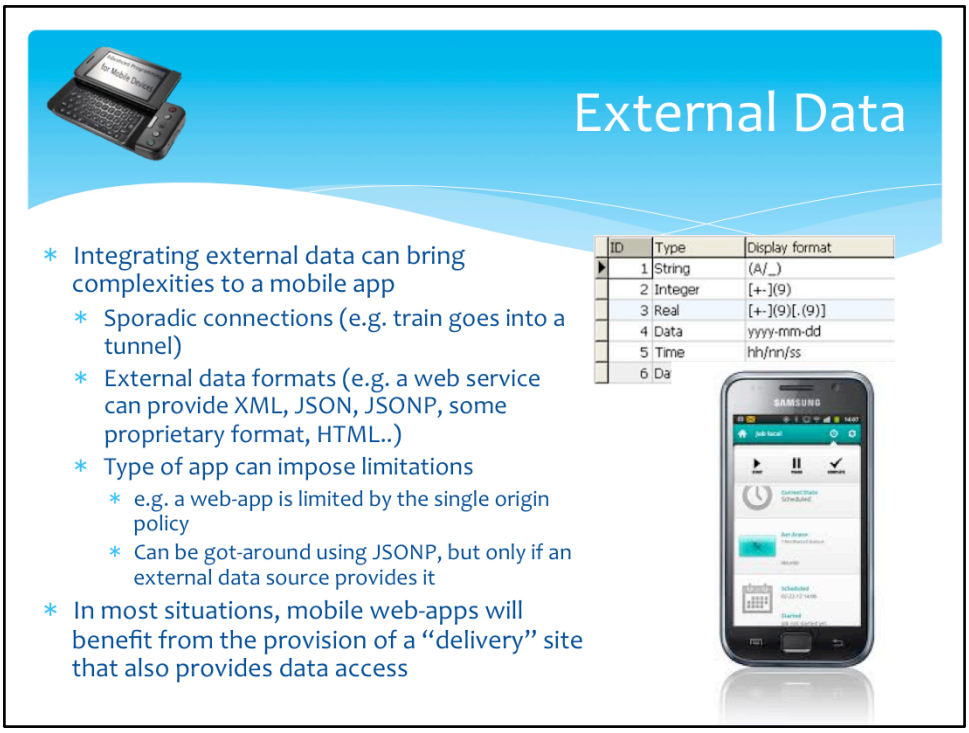
Many apps have needs beyond this – e.g. maps need constant updating as the locale changes

In all apps of this type, there is a need to balance the locus of the processing demand

Device app (i.e. the client) does presentation, but also local processing (e.g. integrating diary or contacts app)

Online processing does things like looking up data-sets, filtering data sent to device (to cut down needed bandwidth)

In most cases, a sweet-spot is where the local and online processing is balanced so that each performs the processing that it is best placed to do.



For a device to use external data, there must be a protocol (or format contract) between the device and its info source.

Although this can be at quite a high level (e.g. Map tiles are complete graphics), often the need is to consider

the data exchanges at a lower level. What is the raw data (at the online end), and how should this be

packaged for effective and efficient transfer to devices?

There are several standard formats (XML, JSON, JSONP, csv, proprietary formats)

There are technical barriers to providing a device with data

Apps are not free to just grab data from a source, even though this data is freely available to a browser

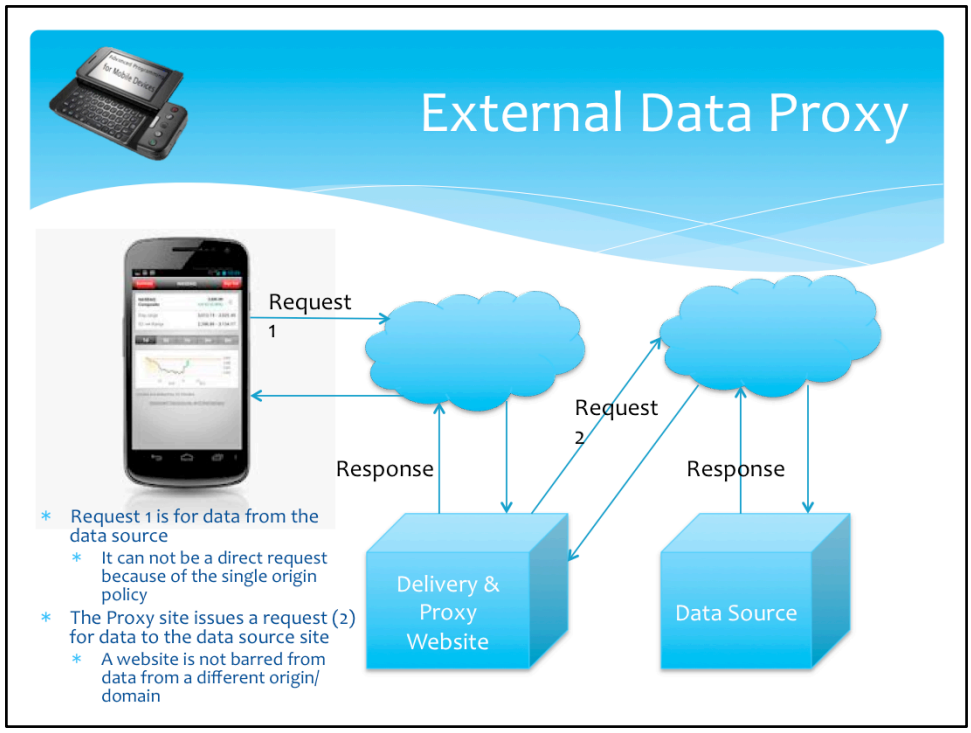
Permission needed (for data access, from the device owner etc.)

The Single Origin Policy is a particular problem – this says that a web application can not simply access data from

other sources – the browser restricts it to access data only where the data origin is the same as the app origin

in terms of Protocol (e.g. http/ftp), Domain (e.g. www.uws.ac.uk) and Port (e.g. 80 for web)

Developers need to work around this (in a variety of creative ways we'll look into later.

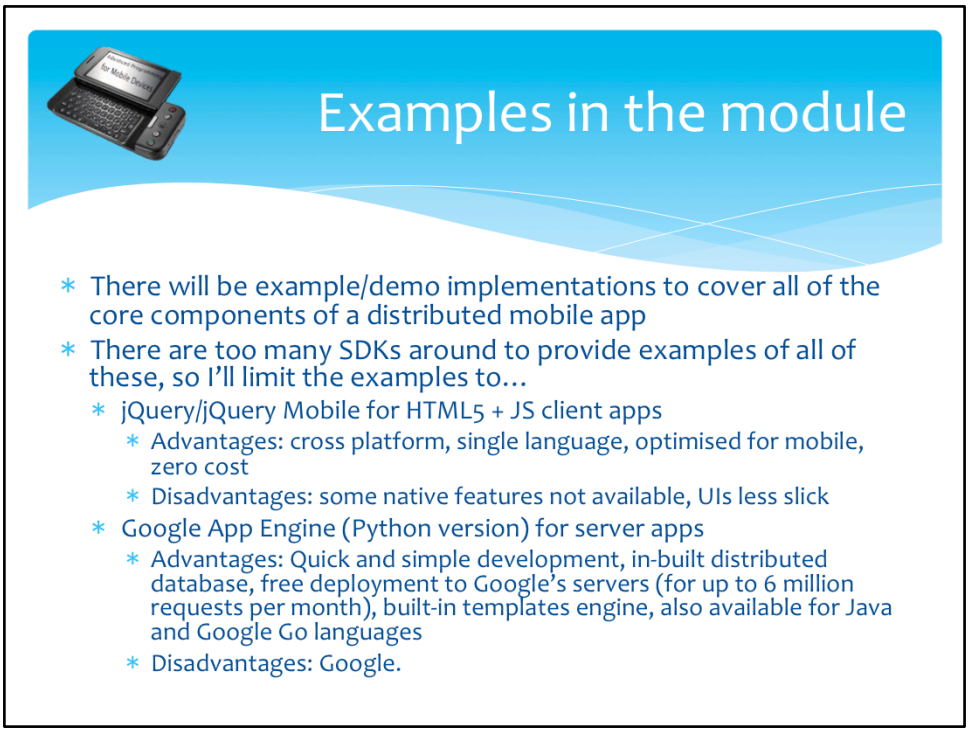


One route around the Single Origin Policy is to create a native app which must request permission either to be hosted (at the Apple app store)
or when it is installed (like Android apps).

Another is to provide a proxy so that a web app can access online data via its original delivery site.

A third is to use a format that is allowed to cross domains – JSON/JSONP does this.

This is a short list of kits. There are many more, usually a bit more obscure or limited.



In some cases, I'll use apps already deployed on App Engine for you to practice

Build a mobile app that access some data fed from a JQM source

Find these in Moodle

Why am I still using jQuery Mobile???

One good reason is that jQM strongly supports the idea of a true Single Page Application (SPA), and uses AJAX to switch between multiple pages in a single HTML file. Other Mobile Web App platforms I've examined tend to promote the idea of converting standard web content into web apps, and typical examples from their websites involve multiple HTML files making up an app. That is an exception in jQM (the recommendation is that all pages are in a single HTML file for normal web-app functionality, with separate HTML files used only for seldom accessed functionality – registration, app configuration etc. The result is that jQM apps tend to work faster once they are installed on a device (provided they have been programmed well). Once Ionic decides that this is a more sensible approach, I'll consider adopting it.