Week 1

26 January 2014 22:37

Mobile Software Development!

How to develop mobile apps! (Android)
Hands on course with loads of programming

LABS MARKED!

Explain the landscape of mobile development
Develop good mobile apps
Program in android
Understand the main android components
Use the android/eclipse environments
Develop your own app <-- This is the assignment

Lecture notes not intended to contain all material covered on course.

Notes on the board/in class etc are the student's responsibility to gather.

50% CA 50% Exam
Same Structure for GUI

Intent class
Instantiate your own object of it
Used to call functions from the system.

.Java files are activity files, stored in src directory

Varients of XML
XHTML
MathsXML
VoiceXML

XML all about formatting data. XML would allow you to use a definition as tags and put the value inside of it. A way of naming data and making it machine understandable.

```
<cat gender="Male" lives="4">
        <name>fluffy</name>
        <age>6</age>
        <type>dunno</type>
        <tail>yes</tail>
</cat>
```

Week 2

05 February 2014 11:04

File -> New Android Application

Custom Launcher icon - icon in the launch tray

Activity created automatically when project is made. It's not a resource, but rather a pulling together of everything in the res directory and compiling it

When you have multiple activities, you have to define which one is shown when the application is launched

AVD Limits
- Mouse control
- Screen sizes
- Resolution
- Performance may be different
- Camera/Accelerometer/Location sensors can't be tested.
- Real world interruptions can't be tested very well

DDMS is your friend for debugging.
DDMS -> Logcat -> Red errors

Applications!
Button on screen, press it and time appears.
Listener on button for when it detects button clicked
set button to system time

Wrap content = wrap around text
Fill parent = fill the parent screen

All widgets are of type view, hence why you have to cast it as the appropriate type when assigning it to an object.

The this parameter sent through refers to either the current class or the interface that is implemented

$\left.\begin{array}{l}\text{Button}\\ \text{Editfield}\end{array}\right]$ View Examples

```
->(    )findViewById(R.id.XXX)

myView.setXXXListener(this)

onXXX(    )
{

}
```

Week 3

12 February 2014 10:56

Containers/Layouts!

- RelativeLayout - used in general Screen layouts
- LinearLayout -> used in row Layouts
- Easier then Java Swing layouts as all XML driven.
- Colors held in XML file in Res/Values
- Colors.xml
- Orientation = Vertical
    - Set orientation of elements in a linear layout
- Fill Model - Wrap_content makes just being enough to enclose, fill_parent spreads it out
- Gravity is alignment
- Padding is white spacing.
- Android:id="***" is needed to link the java with the specific elements
- Relative Layout
    - Say where widgets are relative to one another
    - Great for when doing multiple screen sizes
- TableLayout
    - Bit like HTML tables with rows and columns etc
- ScrollView
    - Automatically scrolls
- Intents
    - Used for specific actions in android such as switching activities, open web browser, specifying first activity
- Two XML Layout files
    - Main.xml and Green.xml
- Two Activity .java Files
- Moving from one to another
    - Uses intents to do this
- App has more than one activity
    - Need AndroidManifest.xml file to know about each activity:
- INTENTS
    - Used to link up components
    - e.g. first activity that opens in an app is declared the "LAUNCHER" in manifest file
    - All other activities are usually reached using intents
- In flashlight app, create intent and use the activities start activity method to execute it

```
Button greenButton = (button)findViewByID(R.id.btn);
greenButton.setOnClickListener(new View.onClickListener(){

        Public void onClick (View v){
                Intent intent = new intent(RedFlashLightActivity.this, GreenFlashLightActivity.class);
                startActivity(intent);
        }
}
```

- Whole set of purposes for Intents.
- Constructor depends on whether you know what you're going to invoke or if you're going to pass a variable
- Used the constructor:

```
Intent(Context packageContext, Class<?> cls)
```

Create an intent for a specific component.

Code:
Intent intent = new Intent(RedFlashlightActivity.this, GreenFlashlightActivity.class);

- Finish() method stops the current activity

```
Public void onClick(View v){
	Finish();
}
```

Intro to Lists

- List != Table
    - List is Dynamic
    - File versus DB
        - Database is a file however it is more structured.
- Making a list from DB
    - Make connection, query DB using SQL, assign results to list.
- List of countries
    - Array
        - Arrays.xml
- Lists are critical on Mobile as they save you from entering data.
    - Text fields on Desktop is fine, Mobile is not.
- Lists used to
    - Get users to select a choice
    - Display a lot of data at one time
- pullExtra() method from intent to bring data from one activity to another

1. Data per row?
2. Array/DB?

- Data mapping is automated in Android.
    - Adapter
- To implement list
    - XML Layout
        - Do you need 2 rows, 100? Etc
    - Need to supply the data to the list
    - Need to know where the current selection is e.g. if user clicks on a name from the list, which name was it?
    - Might need to control page scrolling if a lot of data
        - How many fit on a page? Different device sizes?
    - Might want to customize your row e.g images on each row? Checkboxes? Two rows per item? Etc
- <ListView>
    - Used in XML layout to create a list.
- If using a list you need to supply it with data.
    - e.g. arrays or database queries.
- Android provides various types of data adapters to hand the supply of data

Week 5

27 February 2014 08:53

Persistent Data in Android

convertView is a recycling mechanism

Android can only see data that is within the application aka private data

- Security Reasons
- Operability

Content Provider : Allows you to access data from another application.

Examples of Persistent Data:

- Contacts
- SMS

Several ways:

- Local Database
- Local file
- Shared preferences
- Remote network

Choices:
Database (SQLLite)

- Own private database
- Good for structured data
- Good for regular queries/updates

Files:

- Can store and update information on files
- Can be internal or external
- Faster than database
- Doesn't know about data structure
- To update data, need to read through the whole file

Shared Preferences:

- Ok for simple data across sessions
- Stored in key value pairs
- To implement you need a preference.xml file in your res/xml folder

getPreferences() api calls for different things.

Remote storage on a network:

- Pushing data to or from a database
- API has packages to support this.

Manage DB Structure

- E.G. create table
    - Use SQLLiteOPenHelper

Use DB data in your app

- e.g. select data, insert new data
    - Use SQLLiteDatabase

SQLLiteOpenHelper has methods built to deal with database logic such as onCreate(), onUpdate() and connecting to the database.

Logically, need to be able to code to ensure for if it’s the users first install, if it's not the first install but the database is out of date or if the database is up to date

Database goes into **Model** from **MVC**

Context is the activity it is used in

Cursor is used to store a resultSet - Like in java.
Abstract thing which has a number of rows.

Week 6

05 March 2014 11:04

"*A mobile app to allow surveyors to log water table levels, as they measure them. It also shows a map of water table levels as measured so far by surveyor teams.*"

Data Storage Solution:

- Remote storage - Remote Data
- Local Storage :-
    - Stores own readings
        - -> Database
    - Location
    - Local Table Readings

Tech Arch:
Hardware Components:

- Mobile Device
- Server

Software Components:

- SQLite
- HTTP
- Apache for server
- XML / JSON

Purpose of Assignment is to encapsulate everything we've learned.

Avoid data re-entry for the user. Send data to intents using pushExtra(), retrieve using getExtra() method.

SQLLiteOpenHelper - Sample Code

ContentValues class makes code more robust when doing inserts in code.
The return at the end of a method using ContentValues is the row ID that is inserted, otherwise it will return a -1 if it is not returned

ContentValues can also be used for updates.
To use the database in an app, have database in one class with everything inside it. Then instantiate it via onCreate method in activity

M - Database Class
V - View - The Screen Layout - XML Layouts
C - Everything else

Persistent Data in Android - PT2
Database Operations

- Prone to runtime errors.
- execSQL, getWriteableDatabase() are especially prone
- These throw exceptions which can be caught in the code
- Error handling via Try/Catch Statements

Asynchronized means not waiting for a response. Used with threads so as to not slow down the main thread.
AsyncTask subclass needs to be called

- Used to move anything slowing the GUI down
- Used for database connections, networking, etc.

Queries in android

Use rawQuery() method.
Means pass in the sql statement directly as a string.
Can't have dynamical queries. Won't show errors with it until at runtime.

Other method is simple query(). Builds the SQL statement from a bunch of parameters. 7 Parameters

1. Table name
2. Array of columns to return

3-7 Specific values you want to pass through such as where surname begins with M etc.

Returns Cursor. This is the most common way of building queries.

SQLiteQueryBuilder used for dynamic queries.
Whatever rows are returned comes back in a cursor. To View rows, need to navigate around the cursor. API supplies various methods for moving along the cursor.

e.g.

```
moveToFirst()
moveToNext()
isAfterLast()
```

Can also set a cursor adapter to dump results into a list.

```
Cursor cur = db.rawQuery(SomeSQLStatement, null);
    If(cur.moveToFirst())
    {
        Do
        {
            qName = cur.getString(0);
            telNum = cur.getInt(1);

            //do something with it
            Log.i(TAG, "String" + qName + "Tel" + telNum);
        }
        While(cur.moveToNext())
        {
            //do other stuff such as increment through rows.
        }
    }
```

Displaying database rows on a list

- ListView for xml layout
- Row.xml
- SimpleCursorAdaptor
- Cursor of database rows for feeding into the adaptor.

SQL -> Cursor -> Adaptor -> List

Example

```
Public class TestData extends ListActivity
{
    PersonDBManager db;
    Cursor myCursor;
```

```
        //called when the activity is first created
        @override
        Public void onCreate(Bundle savedInstanceState)
        {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.main);
                db = new PersonDBManager(this);
                db.open();
                addRows();
                getRows();
                String[] columns = new String[] {"Surname", "City"};
                int[] to = new int [] {R.id.surname_entry, R.id.city_entry};
                SimpleCursorAdapter mAdapter = new SimpleCursorAdapter (this, R.layout.row, 
                myCursor, columns, to);
                this.setLIstAdapter(mAdapter);
        }//end onCreate method

        //Green  Highlighted Text indicated the code that creates the adapter and sets it to the list

}//end class
```

To click the list, use onListItemClicked method. To get the position, you need to go back into the cursor and get the index of that value.

```
Public void onListItemClicked(ListView parent, View v, int position, long id)
{
        Super.onListItemClick(l, v, position, id);
        Cursor = (Cursor) mAdapter.getItem(Position);
        //retrieve the 2nd column which contains a name
        String myNaeme = cursor.getString(1);
        //add code to do whatever I want when an item is clicked
}
```

Week 7

12 March 2014 11:12

Last week lab part 3

SimpleCursorAdapter Class

Rows - Contained in cursor class
Layout of rows - Rows.xml
Array of column names

Most people got SimpleCursorAdapter wrong

Part 4:
OnListItemClick()

Can retrieve the data from the cursor using parameter position. Can use position to get the various elements of data that were on the row.

Part 5
Change in the cursor whether you want to or not.

Modify keyboard to enter the types of data entry expected in that field.

Android:inputType="text|Email.."

Autocomplete tag in XML to use it in an EditText

Gestures
Single touch

Listener: OnTouchListener
Callback Method: OnTouch
Event itself: MotionEvent

Week 8

19 March 2014 11:03

Mobile Websites vs Apps

Mobile Websites
-bank - security
-gmail - functionality
-sports - design

Mobile Website	Native App
+ 1 dev. Effort - Test etc. - internet Connectivity - Browser settings may block adds and other settings - browser versions - No access to sensors + shared version, shared info, immediate updates	- Install (+ Memory Storage) - Multiple Platforms - Android - variety of devices + more control over installed apps + Sensor access

Networks: Remote Data Access
Protocol: HTTP - 95% of the time this is used
- GET
- POST

Other Examples - FTP, TCP/IP, Sockets

Have to get a response from server to client

So far we've covered SQLLite database as a form of persistent data. But may be a requirement for remote resources
- Remote dB
- Remote web page
- Etc

Apps often need to connect to remote data or processing

Back end processing often the larger part of 'the system' with app just one (of several?) clients.

Concerns
- Privacy Concerns
- Device performance degradation
- Unwanted network data charges
- Opportunity for app to fail if network access not guaranteed
- Testing: On emulator, computer network connection used - likely to be faster

JSON - Javascript Object Notation
Uses Js syntax
Parsers and libraries.

Example Json object
"Employees":[
{"Key":"value"}
]

Table name = employees
In square brackets is an array

Week 9

26 March 2014 11:17

Activity Lifecycle

- onCreate();

Activities - screens that do something
Services - no UI, runs in background
Content providers - supply data to one or more apps.
Broadcast receivers - Listens out and response to broadcast announcements

- Activities get stopped, killed off all the time.
- Prioritisation

Activity Stack

- Start an activity
- Activity starts another, then the called activity goes to the top
- Everytime a new activity is started, it is pushed on top of the stack
- When activity finishes, it is removed from the stack

Activity States

- Active - Foreground, running, usable
- Paused - lost focus, but visible
- Stopped - not visible, obscured by another activity
- Dead - stopped

Lifecycle methods - moving between the four statements

- When activity is activated, onCreate() method is automatically called
- When activity killed off, onDestroy() automatically called.

Override the activity lifecycle methods in your activity.

onCreate()

- Called when activity is first created
- Bundle supplied in case activity previously frozen

onRestart()

- Called after your activity has been stopped, before it is started again .
- Opening db connection again, starting gps again

onStart()

- Called when the activity is becoming visible to the user

onResume()

- Called when activity starts interacting with the user

onPause()

- ? Save data
- ? Switch or close resource hungry stuff for e.g. GPS

onStop()

- Activity is about to be stopped

onDestroy()
- Activity is being killed off
  - By system
  - By your code

onActivityResult

One Activity starts another activity directly and forces the called activity to return it. Automatically called on return to execute what should happen on return.

Location Based Services
Phone sensors to detect its own location

Once you know where your phone is, you can do a lot of stuff with it such as who's nearby etc.

Concept of a threshold for when you want it to check the frequency of the GPS. Used for apps that might be used for walking or driving etc. Could dynamically change threshold depending on speed of which they go from one point to another

Battery life question - she likes putting this on exam

Week 10

02 April 2014 11:03

Maps in Android

Easy to integrate into app

Google supply google maps api

Zoom, move around, put an overlap of your own stuff, map a typed up street name to a physical map

Be Careful using Google Maps, must use API V2

V1.1 officially deprecated as of Dec 2012, no longer possible to develop a new app using it.

Coding and API key are different between the two generations

Using API V2 use
- Fragment tag with a particular class attribute
- An Activity or FragmentActivity
  - Simple to display the map - extend activity
  - And just render the XML layout
- Bunch of very specific set up steps: API key, Google Play Services, manifest file changes

Set up steps to implement a map

- Install Google Play Services
- Add Google Play Services to your workspace
- Link your android project to google play services lib

Step 2
- Get a google API V2 key
- Retrieve your SHA1 key for your android environment
- On the online google API console, generate an API key for your project
- Add the API key generated to your android manifest file in your project

Step 3
- Set Permissions in the manifest file
- An app that displays maps need to track location, access the internet, write to external storage… and others

Step 4
- Enable your app use to use OPENGL
- Needed to render a map

Map Coding
- Map class which has all the methods is GoogleMap
- Then you have to get hold of the actual map you setup in the xml which is the fragement.

```
myMap = ((MapFragment)getFragmentManager().findFragmentById(r.id.map)).getMap();
```

getFragmentManager is available in your activity

setOnMarkerClick()

Quirks
There is a class called SupportMapFragment for older versions of android

- Anything below API level 12
- Mapfragment requires minimum SDK = 11

Fragments
Fragment is a little bit of a user interface, reusable.

Similar to

- Jpanel
- HTML -> "Include"
- Reusable bits of logic and screen

Origin for why they're created is for the use of adjustable UI over various devices (Phones, Tabets)

Fragments have activity lifecycle including onCreateView()

Two ways of implementing Fragments, Static and Dynamically

Static usage is if Fragment is always going to be used at the one time.

Fragment tag inside main xml file which is a reference to the fragment

You also need another Fragment XML file to define what its layout is.

Class to inflate the fragment, extends fragment. Always need to override onCreateView()

//inflate fragment

Class attribute in XML needs to refer to the java class that controls it.

Static works if your fragment always appears in your main frame. If its conditional you can't just embed the fragment in the XML file

Use a separate landscape folder for defining different layouts.

Check orientation in onclick method of list

getResources().getConfiguration().orientation == Configuration.ORIENTATION_LANDSCAPE

Else

Create intent to switch activity
Intent.putExtra("item", content)
startActivity(intent);

In second activity:

- Check orientation, if landscape donothing
- Setcontentview to fragment
- Bundle extras = getIntent().getExtras();

```
If (extras != null){
        String selectedItem = extras.getString("item");
        TextView textView = (TextView)findViewById(R.id.selectedopt);
        textView.setText("You have selected " +selectedItem);
}
```

Dynamically adding Fragments

FragmentManager

- getFragmentManager
- Depending on conditions
  - Add fragment
  - Remove
  - Replace etc.
  - When you've added/removed fragments, you must use a method called commit();
- If adding a fragment at runtime, you need a placeholder to say if I decide to add fragment, put it in here.
- <FrameLayout
  - Android:id="@+id/place_for_frag"
  - Android:Layout_Height="match_parent">
- </FrameLayout>

Week 12

30 April 2014 11:03

Supporting Multiple Screen Sizes

Problem with android devices is different screen sizes and densities.

Classification of screen size by inches and density

DIAGRAM IN NOTES

Screen size diagram url

Limitation in the terms of repeated downloads, devices that don't download apps etc.

Tactics for different screen sizes and densitites

- Use Screen compatibility mode
- Specify in your app which screen sizes/densitites are supported
- Provide multiple screen layouts
- Provide multiple image densities
- Use best practises
- Use fragments

Generally no way of doing this without testing, each android version brings changes. Biggest drawback of android

Screen Compatibility mode
- Android tries to resize the screen.
- Different things happen depending on API Level
- 1.6 - 3.2 draws application as a postage stamp in the corner with a black border surrounding it.
- 3.2 or greater - system draws application layout and then scales it to fill the screen. Can cause blurring and pixelation

Specific Screen sizes
- Change manifest file to say which screen sizes/density are supported and which are not.

Provide multiple layout files
- Different folders for sizes/orientations with the same xml file name.

Provide multiple density image versions
- Drawable directories

ConvertView in android for recycling lists.

Assignment Ideas

05 March 2014 21:38

Uni Planner Application

Database Structure:
_id autoincrement
Module_Name String
Time_due
Submisssion_ type {Exam, Assignment, Lab}
Description