Part 1 - XML – 15 minutes estimated

Using a simple editor such as wordpad or textpad, type up XML that describes a sample "student". Include 6 things about a student: name, home address, age, course, city of study and institution. Include the following attributes

- Gender for the name;
- FullorParttime for the course
- Attributes on the home address to capture full address details

Validate using the online W3C XML validator at:
http://www.w3schools.com/xml/xml_validator.asp

You will sometimes need this validator when programming in Android, as Eclipse does not pick up all XML grammar errors.

Part 2 – Android Studio – Generate a new project

First, read the notes on the Lab home page about Machines and Android.

Open Android Studio – and Set up a new project. Choose "Blank activity" when you are prompted during the project set up.

When it has set up, investigate the directory structure.

- Find the "java" directory – open up the .java Activity file(s) within the package your created.
  - What is the superclass being used?
  - What is the onCreate method doing?
  - What is setContentView() method doing?

- Find the "res" (resources) directory – Look at the XML layout(s)
- Find the manifest .xml file – and look at how Activities are declared.

Part 3 – Android Studio – Run the project

To see an app on a phone – you can use either the build in Android Virtual Devices – to set up an AVD – Or you can use your own phone OR you can use an online emulator. See notes on the lab home page.

It will be slow running the AVD first time, if you choose that option.

The default project is just an app with one screen, that displays Hello World on it, as shown in class.

Part 4 – Android Studio – Change the project

- Modify the code to change the default “HelloWorld” text to display different text.

- Add a Button to the layout.

- Re run it and test it.

Part 4 – Android Studio – Event handling

- Implement a listener as an interface within the class and add it to the button you have added so that when it is clicked, a popup message (called a “Toast” in Android appears

```
Toast.makeText(getApplicationContext(), "Button clicked",
Toast.LENGTH_LONG).show();
```

Part 5 – Android Studio – Event handling

- Add a second button, and get the listener to “listen” to it too.
- Change your code so that when either button is clicked, the Toast message displays a different message, depending upon which button was clicked. E.g.

```
Toast.makeText(getApplicationContext(), "Top Button clicked",
Toast.LENGTH_LONG).show();

Toast.makeText(getApplicationContext(), "Bottom Button clicked",
Toast.LENGTH_LONG).show();
```