Software Engineering Large Practical (Android version) 2013/2014

Professor Stephen Gilmore
School of Informatics

Issued on: Wednesday 18th September, 2013

About

The Software Engineering Practical is available in two versions: the Android version or the iPhone version. Students should do only one version of the practical, choosing the version which interests them more. This document describes the Android version of the practical: a separate document describes the iPhone version of the practical.

Introduction

The requirement for the Software Engineering Large Practical is to use the Eclipse development environment to create an app implemented in Java and XML for the Android phone. The purpose of the app is to allow School of Informatics students to access the timetable information for their lecture courses more conveniently.

— ◇ —

The School of Informatics publishes a timetable of lectures and tutorials during the academic year. There are nine slots during the day where lectures and tutorials can be timetabled throughout the five days of the working week (although the lunchtime slot is very rarely used). Because the timetable includes information for first-, second-, third-, fourth- and fifth-year students there may be multiple entries for each slot. The timetable for the current academic year includes between zero and five entries in each slot. Each entry specifies the name of the course and the appropriate year group (sometimes one specific year group, sometimes several). Each entry also includes the location of the course lectures. A small number of entries contain a free-text comment such as "Week 1 only" or "Weeks 3 & 6 only".

— ◇ —

In order to fit all this information into the timetable, names of lecture courses are replaced by short letter codes, as are the names of the lecture theatres and course venues. This arrangement has a number of advantages. It is compact, and relatively easy to maintain. Against this, it has a number of disadvantages. The use of codes for course titles and course venues makes the

resulting timetable somewhat cryptic and hard to use for students who do not know the codes. A typical entry reads

CDI1 [5]
DO4 FH

The course venue codes are listed on a separate page and the lecture course codes are listed on yet another page, in the sortable list of courses. To make matters slightly worse, the sortable list of courses is not (at the time of writing) linked from the timetable page so decoding a code such as "CDI1" may require a search to find the sortable list of codes before sorting it in course code order and then scanning down to find the required code. Decoding "DO4" and "FH" requires a visit to the page of venue codes.

— ◇ —

The presentation of the timetable information on the web assumes that the user of the information is viewing the information on a desktop or laptop computer with a full-size screen. However, this is not always where the user is when they most need to access the timetable information. They might instead be on their way from one building to another, at which point if you have forgotten the room and/or the building for your next lecture it would be more convenient to access the timetable information on a mobile phone. The organisation of the timetable as a grid is unhelpful here because if the font shown on the page is large enough to be readable then it is not possible to see the row and column headers giving the information about the day and time for the lecture. Only a single grid view is offered: it is not possible to see the same information in a list view instead.

— ◇ —

Further, the timetable is not personalise-able in any way. Even when you know that you are only interested in third year courses there is no way to filter the timetable to see only the third year courses. Once you have chosen the subset of courses that you are actually taking you cannot then hide the courses which you are not taking in order to be able to see a timetable which is just right for you.

— ◇ —

Added to this, the timetable is not searchable in any convenient way. For example, if you want to know the times for the Computer Communications and Networks course then you have to first visit the sortable list of courses to find the code for this and then return to the timetable page and search for this code. (An attempt to guess the short code for Computer Communications and Networks would fail: it is not "CCN".) Having found the times of Computer Communications and Networks if you then want to read more about the course then you need to return to the list of sortable courses and follow the link there to find the course webpage. Then if you want to get directions to the lecture theatre you need to look at the code on the timetable page then look it up on the page explaining venue codes and follow the link there to find the map to the lecture theatre.

— ◇ —

The purpose of this practical is to develop an app for a small-screen Android phone which will allow Informatics students of the university to find their course lectures more easily and customise their timetable.

The information on courses is obtained by download of three XML documents from a server. One document contains information on lecture times, another on venue codes, and a third contains information on lecture courses.

Frequently asked questions

- *I don't have an Android phone. I've never written an app before. How can I do this practical?*
  - You don't need to have an Android phone to do this practical exercise. The software which you develop will run on an emulator which is freely available for Windows, Mac OS X, and GNU/Linux platforms. There is no expectation that you have written an app before: you will learn how to do this in the course of this practical. You may also need to learn more about Java programming.
- *Can I develop my app on my laptop?*
  - Yes. You are strongly encouraged to do this because it will encourage you to investigate the Android SDK and related libraries. Of course, we recommend taking regular, well-organised backups.
- *What is the point of this app? All of this information is already available on the School of Informatics website. Why would anyone use this?*
  - The data is available on the School website, but it is not organised in the way that we would like. The information is split across three different web pages and uses a system which involves looking up codes to find course and venue names. In addition, the timetable is primarily about helping people get to the right place at the right time. Having GPS available on the phone gives us the option of guiding a student to the right building for their lecture.
- *Can I implement my app in Ruby/Python/Scala/Objective-C instead?*
  - No, not for this practical. We need all students to be working in the same programming language in order to make a fair assessment.

    However, Java is not an arbitrary choice. Java is the most widely used programming language for the Android platform and there are many more Java language resources available online to learn Android development from than for any other language.

    For sound educational reasons, we believe that the choice of Java as the development language should help most students to complete this practical on Android successfully.
- *Can I design my app for an Android tablet instead? I'll still be programming in Java and XML.*
  - No, not for this practical. One of the challenges which we are considering is making the information available on a small-screen device. If designing for a full-size tablet, or even a large-screen phone, then it would be possible to find solutions which work there but which are not workable on a small-screen device.
- *Do I have to develop on Eclipse? I much prefer NetBeans/IntelliJ/Emacs/vi etc.*
  - Your code will be tested by loading it into Eclipse for execution. If your Java and XML code and project files do not constitute an Android project as recognised by Eclipse then we will be unable to run your code and you should expect to lose a lot of marks because of this. We strongly recommend developing on Eclipse for this practical exercise.

Part 1

Software Engineering Large Practical (Android version) 2013/2014

Professor Stephen Gilmore
School of Informatics

Issued on: Wednesday 18th September, 2013

Deadline: Thursday 24th October, 2103 at 16:00

1.1 Introduction

This part of the SELP is zero-weighted: all of the assessment is based on the second part of the practical. Nonetheless, you are strongly encouraged to complete this part. Completion of this part will provide useful feedback on your progress and allow you to work out whether you are on track to complete the second part in time.

Any work which you choose to submit should be your own, although you may make use of any part of any publicly-available Java and Android source code which you find useful. (This means that you can make use of code from tutorials which are available on the Android web site, and others. This is re-use, which is a good thing, not plagiarism, which is a bad thing.)

1.2 Description

Part one of the SELP requires you to submit your Android application as a preview version or an alpha version[1] of the software which is to be delivered for part two of the SELP. It is understood that your software at this stage will necessarily be incomplete, in the sense that necessary functionality will be missing, and that it will not have been comprehensively tested, with the consequence that there may be some errors or problems with the application which are fairly easy for the user to discover. However, this preview or alpha version of your application will prove that you have started work on the practical and that you are getting to grips with the Android technology and learning some of the new concepts which you need for Android development. Further, an alpha version gives a rough impression of what the finished product could be like. For these reasons, you are to submit an alpha version of your Android application.

[1] An "alpha release" is an early version of a software application released in order to get feedback from evaluators. (Later versions are called "beta releases", "release candidates", and then "releases".)

1.3 What to submit

You are to submit the directory which contains your Android project. This should be a working Android application which can be compiled using Eclipse and executed on the Android emulator. You do not need to write any accompanying documentation for this part of the practical. You are not required to submit other documentation for your project, such as design documents, UML documents, or user documentation.

1.4 How to submit

First, get your code on the School of Informatics DiCE computer system, copying it from your own laptop. If your project is in a folder called `TimetableApp` then you should submit it for the `selp` part `1` using the command

```
submit selp 1 TimetableApp
```

Please use the above command in order to ensure that your work is assessed. Work which is not submitted by the `submit` command may not be assessed.

1.5 Deadline

The deadline for this practical exercise is

Thursday 24th October, 2103 at 16:00

1.6 Frequently Asked Questions and Clarification

- *I want to explain which parts of my application are not finished. Can I submit documentation along with Part 1?*
  - Yes, if you wish to then you can do this. If doing so then please supply a printable PDF document.

Part 2

Software Engineering Large Practical (Android version) 2013/2014

Professor Stephen Gilmore
School of Informatics

Issued on: Wednesday 18th September, 2013

Deadline: Thursday 19th December, 2013 at 16:00

2.1 Introduction

This is an assessed practical exercise. It carries 100% of the marks for the SELP. It represents a total of roughly 100 hours of work. The work which you submit for assessment should be your own although you may make use of any part of any publicly-available Java and Android source code which you find useful. (This means that you can make use of code from tutorials which are available on the Android web site, and others. This is re-use, which is a good thing, not plagiarism, which is a bad thing.)

Please refer to the University requirements with regard to all assessed work. Details of these are available at `http://bit.ly/16NkMG1` and `http://bit.ly/1fg2e3d`

2.2 Description

Part two of the SELP requires you to submit the finished version of your Android application. This should be a working Android application which can be compiled using Eclipse and executed on the Android emulator. Your application should assist the user with accessing School of Informatics timetable information and help them be in the right place at the right time for their lectures.

— ◇ —

Key aspects of this problem include the following:

- It should be possible to download the XML files of timetable, venue and course information from the server into your app.
- It should be possible to browse the timetable information and read more information about lectures and courses.

- It should be possible to follow a hyperlink to open the web page for a course in the phone's built-in browser.
- It should be possible to filter courses (for example, to see only third-year courses).
- It should be possible to select courses and see only the information for the courses which you have selected.
- It should be possible to search for courses by name and by course code (e.g. typing "Database" or "DBS" should find information about the Database Systems course).
- It should be possible to see the timetable information for both semesters in your app.
- Your app should display a default semester.
  - From July to December the default semester is Semester 1.
  - From January to June the default semester is Semester 2.
- Your app should cope with missing and redundant information.
  - Missing expansions for room, building or course codes should not cause unexpected behaviour.
  - Unused expansions for room, building or course codes should not cause unexpected behaviour.

2.3 Specification of the phone

Your app must run on a WVGA (Wide VGA) device with a screen size of 480 × 800 running Android OS 2.3.3 (Gingerbread). When specifying an Android device this is known as an hdpi device.

Note that this is not the most recent version of Android, nor the largest screen phone on the market, but students do not always have the latest and most expensive phones. Your app may also run on larger devices, and more recent versions of Android, but it must run on Android devices as specified above.

2.4 Extra credit

The requirements listed in the section above illustrate the core functionality which is required from your application. A well engineered solution which addresses all of the above requirements should expect to attract a very good or excellent mark. Additional credit will be awarded for additional useful features which are *not* on the above list. Thus, if you have time remaining before the submission deadline and you have already met all the requirements listed above then you can attract additional marks by being creative, conceiving of new features which can helpfully be added to the application, and implementing these.

If you have added additional features to your implementation in order to attract extra credit then you should be sure to document these features if they are not immediately evident from normal use of your application. The purpose of this part of the practical exercise is to allow you to exercise your own creativity and deliver a solution which is uniquely your own.

2.5 Early submission credit

In practical software development, timing and delivery of completed applications and projects can be crucial in gaining a commercial or strategic advantage over competitors. Being the first to market means that your product has the opportunity to become known and similar products which come later may struggle to make the same impact, simply because they became available second.

In order to motivate good project management, planning, and efficient software development, the SELP reserves marks above 90% for work which is submitted *early* (specifically, by 16:00 on the Thursday of the week before the deadline for Part 2). To achieve a mark above 90%, a practical submission must be excellent in all technical, functional, and aesthetic aspects *and* in addition to this it must be submitted early.

To be clear, submitting before the early submission deadline does not guarantee that your mark will be over 90%, but *not* submitting before the early submission deadline guarantees that your mark will *not* be over 90%.

2.6 Assessment

Part two of the SELP is worth 100% of the marks for the course. Your mark will be expressed as a percentage given as an integer between 0 and 100. Thus it is not possible to score more than 100% and it is not possible to score less than zero.

2.7 Assessment criteria

This practical exercise will be assessed in terms of the completeness of the solution to the problem, the quality of the Java and XML code produced, and the ingenuity and craftsmanship which have gone into designing a good and robust solution to the problem.

- For example, all other things being equal, an app which allows users to browse the timetable, filter and select courses should expect to receive more marks than an app which does not allow this.
  - A more complete solution will get more marks.
- All else being equal, an app which when loaded into Eclipse reports static analysis errors (called "Java Problems" in Eclipse and displayed in the Problems View) should expect to attract fewer marks than one which does not.
  - Sloppy development style ignoring compiler warnings will lose marks.
- Additionally, all else being equal, an app which contains examples of poor Java programming style (such as empty catch statements) should expect to attract fewer marks than an application in which Java exceptions are handled gracefully.
  - Poor programming style will lose marks.
- Finally, all else being equal, an application which contains logging statements (using the class `android.util.Log` and generating calls to `Log.d` or other methods) would expect to attract more marks than an equivalent application without logging (because the version with logging is more maintainable).
  - Including logging will gain marks.

2.8 What to submit

You are to submit the directory which contains your Android project. Your work will be assessed by compiling and executing your application so you must ensure that all source code and project files needed to compile your application are submitted.

You are to write accompanying documentation for this part of the practical. Please create a folder in your project called `doc` and place your documentation inside this folder. You should use this document to record:

- any special instructions for running your application,
- screenshots showing your app loaded and working in the Android emulator,
- developer notes, UML diagrams or other documentation,
- any parts of your application which are not finished or have non-obvious functionality, and
- any additional features of your application of which you are duly proud.

Please supply your documentation as a printable PDF document, placing this in the `doc` folder within your project. Documents in other formats will be ignored.

2.9 How to submit

First, get your code on the School of Informatics DiCE computer system, copying it from your own laptop as necessary. If your project is in a folder called *TimetableApp* then you should submit it for the `selp` part `2` using the command

```
submit selp 2 TimetableApp
```

Please use the above command in order to ensure that your work is assessed. Work which is not submitted by the `submit` command may not be assessed.

2.10 Deadline

The deadline for this practical exercise is

Thursday 19th December, 2013 at 16:00

2.11 Marking

Your submitted coursework will be marked within four (semester) weeks of submission. (This is the period of time approved by the School of Informatics for large practical classes.) That is, marked work should be returned to you by Thursday of week 4 of the second semester. For 2013/2014 this means Thursday 6th February, 2014.

We will try to meet this deadline but – for good reasons – students sometimes get deadline extensions meaning that their submitted coursework is not available to mark starting at Thursday 19th December, 2013 at 16:00. This can delay the return of marked coursework.

2.12 Frequently Asked Questions and Clarification

- *Should I put comments in my code?*
  - Yes. All else being equal, a submission with comments will attract more marks than one without comments.
- *My code generates warnings. Can I just acknowledge these by putting a comment in the code?*
  - That will let the markers know that you know about the problem which causes the warning but a better fix would be to rewrite the code so that it *doesn't* produce warnings.
- *Are all warnings equally serious? Can't some be ignored?*
  - Some warnings are more serious than others but ignoring the ones which you think are not serious may cause you (or others maintaining your code) to ignore the warnings which actually are serious. Warning messages are the first impression which your code gives to others who look at it.

    Submitting code which generates warnings is like turning up to a job interview in your pyjamas. It doesn't give the impression that you've really made an effort.
- *Can I reuse other apps, source code examples and open-source software in my app?*
  - Yes, you are encouraged to do this. Examples and tutorials are made available on-line in order that people can use them and learn from them.
- *Is there a coding standard or package naming convention which must be adhered to?*
  - No particular coding standard has been identified. However, you should try to be consistent, and to give meaningful and helpful descriptive names to classes and methods.
- *How will this practical be marked?*
  - Submissions for the Software Engineering Large Practical are evaluated using the following process:
    1. The XML files specifying the information about the lecture times, venues and courses are updated to include a previously-unseen course at a previously-unseen venue to be examined during testing.
       - Submissions which have hardcoded in the timetable will lose marks because they will not include the information about the previously-unseen course.
    2. The accompanying documentation is read for instructions on how to use the app.
       - Submissions with insufficient documentation will lose marks here.
    3. The Android project is imported into an instance of the Eclipse platform and inspected for errors or warnings.
       - Submissions with errors or static analysis warnings will lose marks here.
    4. The project is launched as an Android application and run on the emulator
       - Submissions which fail to install or launch will lose marks here.

5. The app is evaluated in user mode by browsing for content.
    - Submissions which have insufficient content will lose marks here.
6. Evaluation continued by filtering and selecting courses.
    - Submissions which fail to filter or select courses will lose marks here.
    - Submissions which crash with a run-time error will lose marks here.
7. Emulation is terminated and then restarted.
    - Submissions which fail to retain selected courses will lose marks here.
8. Other additional features of the application will be explored.
    - Accessing a course web page will be tested here.
    - Submissions with useful additional features will gain marks here.
9. Usability and aesthetics of the application will be explored.
    - Organisation and readability of the application will be evaluated here.
    - Layout, colours, fonts and visual design will be evaluated here.
10. The Java source code will be inspected for good programming style.
    - Submissions with insufficient logging will lose marks here.
    - Submissions with too few comments will lose marks here.
    - Submissions with blocks of commented-out code will lose marks here.