Informatics Large Practical

Stephen Gilmore
School of Informatics

Document version 1.0.0
Issued on: September 22, 2018

About

The Informatics Large Practical is a 20 point Level 9 course which is available for Year 3 undergraduate students on Informatics degrees. It is not available to visiting undergraduate students or students in Year 4 or Year 5 of their undergraduate studies. It is not available to postgraduate students. Year 4, Year 5 and postgraduate students have other practical courses which are provided for them.

Scope

The Informatics Large Practical is an individual practical exercise which consists of one large design and implementation project, with two coursework submissions. Coursework 1 is largely administrative, requiring setting up and populating a source code repository for the practical, and providing a plan of the implementation work to be undertaken. Coursework 2 includes the project report together with the implementation.

Courseworks	Deadline	Out of	Weight
Coursework 1 (Plan)	16:00 on Friday 12th October	25	25%
Coursework 2 (Implementation)	16:00 on Friday 14th December	75	75%

Please note that the two courseworks are not equally weighted. There is no exam paper for the Informatics Large Practical so to calculate your final mark out of 100 for the practical just add together your marks for the two courseworks.

Introduction

The task for the Informatics Large Practical is to use the *Android Studio* development environment to create a mobile phone game implemented for an Android device. Your app should be a prototype of a multi-player collaborative location-based game where players collect and exchange virtual coins which have been scattered at random around the University of Edinburgh's Central Area. Your app should be considered to be a prototype in the sense that you should think that it is being created with the intention of passing it on to a team of developers who will maintain and develop it further. Given that it is based on collecting coins, the game is called *Coinz*, with the unusual spelling with a 'Z' making it easier to find information about the game online.

— ◇ —

In the Coinz game, the locations of the coins are specified on a map. Coins are collected by getting near to their location on the map, by which we mean that the player literally moves to that location with their mobile phone. For the purposes of this game, a player will be judged to be close enough to a coin to be able to collect it if they are within 25 metres of the coin.[1]. A new map is released every day. Each map has fifty coins.

— ◇ —

The coins which are being collected belong to one of four different (fictional) crypto-currencies: Penny, Dollar, Shilling and Quid (these are respectively denoted by the four-letter acronyms PENY, DOLR, SHIL, and QUID). The four different currencies fluctuate in relative value on a daily basis, just as real currencies such as GBP, EUR, and USD do. The relative values of the four currencies are defined in terms of their relation to a fifth currency, GOLD. Maps contain coins from the currencies PENY, DOLR, SHIL and QUID, but no GOLD coins.

— ◇ —

Every coin on a map has a value greater than zero and less than ten of its currency. Because the coins are virtual crypto-coins, not minted physical coins, there is no reason for them to be in a small number of denominations (1p, 2p, 5p, 10p, 20p, 50p etc), and their value can be any positive real value less than ten, e.g. 5.72384765123 DOLR or 2.37846111259 QUID.

— ◇ —

Maps are made available as Geo-JSON documents, in a JSON format which is specialised for describing places and geographical features. Figure 1 shows an extract from a typical map.

[1]GPS-based devices cannot determine your true location perfectly but the Android LocationManager API at least attempts to determine the accuracy of its estimated location.

```
{
  "type": "FeatureCollection",
  "date-generated": "Wed Jul 04 2018",
  "time-generated": "10:53",
  "approximate-time-remaining": "13:06",
  "rates": {
    "SHIL": 24.702404151790425,
    "DOLR": 29.64763865293904,
    "QUID": 6.332861915771016,
    "PENY": 32.393996378130524
  },
  "features": [
    {
      "type": "Feature",
      "properties": {
        "id": "aacc-6b9e-94e0-bdd3-9660-7b78",
        "value": "4.697330935366203",
        "currency": "PENY",
        "marker-symbol": "4",
        "marker-color": "#ff0000"
      },
      "geometry": {
        "type": "Point",
        "coordinates": [
          -3.190193505479799,
          55.94313683057246
        ]
      }
    },
    {
      "type": "Feature",
      "properties": {
        "id": "3982-ccc2-2e46-4c94-033f-8489",
        "value": "3.753936814460823",
        "currency": "DOLR",
        "marker-symbol": "3",
        "marker-color": "#008000"
      },
      "geometry": {
        "type": "Point",
        "coordinates": [
          -3.1907859043303155,
          55.942959264811755
        ]
      }
    },
    ...
    ]
}
```

Figure 1: A map with coins in Geo-JSON format.

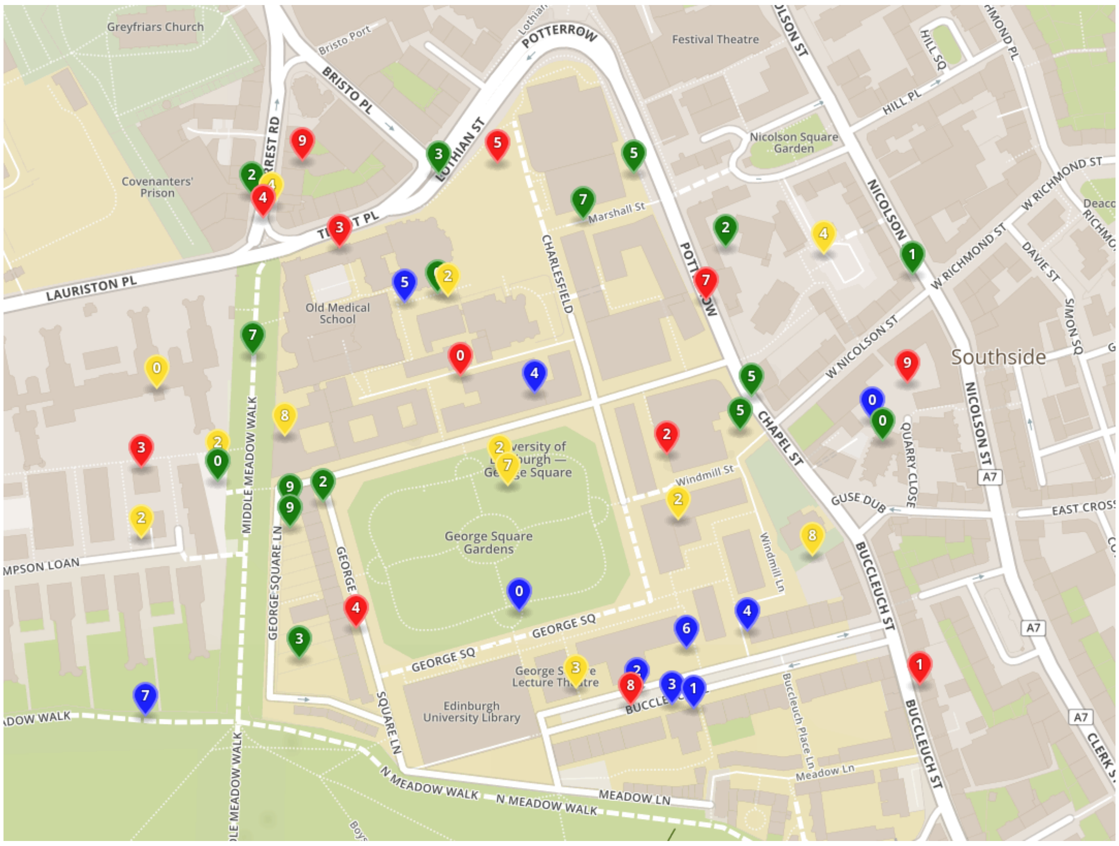


Figure 2: A Geo-JSON map rendered by the website http://geojson.io/

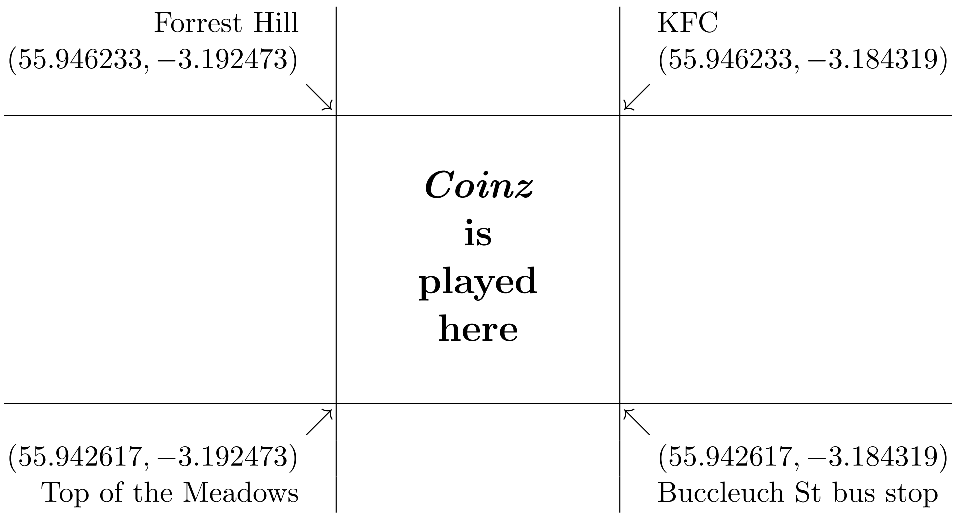
It is difficult for a person to interpret this JSON document directly, but we can render it with some mapping software, such as that provided by the http://geojson.io/ website. Figure 2 shows us what the map from Figure 1 looks like when rendered. This allows us to see where the coins are scattered, with a marker showing us the approximate value of the coin and a colour indicating the currency of the coin. (Red for PENY, Green for DOLR, Yellow for QUID, and Blue for SHIL). Thus the coin in the bottom left corner of the map is approximately 7 SHIL and the coin in the bottom right corner of the map is approximately 1 PENY.

— ◇ —

Every map has the same format; it is a FeatureCollection containing a list of Features. Every Feature has Point geometry with two coordinates, a longitude and a latitude[2]. There will always be 50 Features in the FeatureCollection of each map, each one being a cryptocoin.

[2]In this project, longitudes will always be negative (close to -3) and latitudes will always be positive (close to 56).

All points on every map have a latitude which lies between 55.942617 and 55.946233. All points on every map have a longitude which lies between −3.184319 and −3.192473.



The Features in the map have properties which are metadata which tell us information about the coin, and also markup which suggests a way to render the feature as a placemark on a map. The markup of "marker-symbol" and "marker-color" can be changed, or ignored, but the properties of "id", "value" and "currency" are not to be changed. Every coin has a 24-digit hexadecimal identifier ("id") which uniquely identifies this coin.

— ◇ —

The FeatureCollection contains metadata too, most importantly the exchange rates for each of the four types of currency. Consider the extract below:

```
"rates": {
  "SHIL": 24.702404151790425,
  "DOLR": 29.64763865293904,
  "QUID": 6.332861915771016,
  "PENY": 32.393996378130524
},
```

This tells us that — on the day for which this map applies — PENY was the most valuable currency, with one PENY being worth slightly more than 32 GOLD coins. DOLR was the next most valuable currency with SHIL following that. The QUID was worth a lot less than the other currencies, with one QUID being worth only slightly more than 6 GOLD coins.

During gameplay, players collect coins, storing them in a local wallet, and later pay them into a central bank. However, there is an additional complication, which brings in an element of collaborative play as well as competitive play. Namely, on any given day, you can only pay in directly a maximum of 25 coins[3]. This means that, even if you collect all 50 coins on the map, there will 25 coins which you will not be able to bank. These unbankable coins (the ones which would exceed your 25-coins-a-day banking allowance) are referred to as "spare change".

— ◇ —

You can message your spare change to another player, and they can pay it into the bank even though they have not collected those coins themselves. Coins which you receive from another player can be messaged over to the bank, even in the case where you have collected and paid in 25 coins already. By exchanging your spare change with another player, you can both increase the total that you have in the bank and thereby make indirect use of coins which were of no use to you directly.

— ◇ —

The object of the game is simply to collect as much money as possible.

Underspecification

The above description of the Coinz game provides you with a list of requirements for the game but it does not specify the game design completely.

- For example, it is not specified whether coins are converted to GOLD at the time that they are paid into the bank, or whether they are stored in their respective currencies, and can be converted to GOLD at a later date. (Because the values of the currencies fluctuate every day, these different approaches will produce different results.)
- It is also not specified whether coins expire or not. It might be the case that unused coins held in wallets are discarded at the end of the calendar day, or the end of this session in the game, or it might be the case that they can be held for an indefinite period before being paid into the bank.
- What happens if you receive as spare change a coin which you have already paid into the bank? Should you be allowed to pay it in twice?

Underspecification like this occurs quite often in software projects and it is deliberately included in this practical in an attempt to make the practical more realistic and in order to allow you to be creative in how you design your application. You get to decide questions such as the ones above in a way which makes sense to you. There is not a single "right" answer to design questions such as these: different decisions can produce different, but equally valid, alternatives.

[3]There is no problem with banking fewer than 25 coins, even none.

Bonus features

In addition to the game features described above you should design and implement some *bonus features*, which set your app apart from others. These may be enhancements which are intended to make the game more interesting to play, or more rewarding, causing the user to play more frequently, or for longer sessions. What the bonus features are is up to you but you could consider enhancements in areas such as:

- play modes (easy, moderate, difficult) or attainment levels (beginner, advanced, expert);
- timing-based play (play against the clock);
- recording the distance walked by the user while playing the game;
- user preferences, allowing the user to customise the app in some way; or
- 'background' mode where coins are collected without user intervention while walking, without the user needing to click to confirm collection.

You are not limited to the items above; this list is only to prompt you to think about your own bonus features. You should aim to add *two* additional bonus features of your own invention, but games with more than two additional bonus features are also welcome.

Programming language

Two programming languages are officially supported for app development on Android devices: Java (https://www.java.com) and Kotlin (https://kotlinlang.org).

Java: You are already familiar with Java. The implementation of Java provided by Android Studio supports all Java 7 language features and a subset of Java 8 language features such as lambda expressions, method references, and type annotations (https://developer.android.com/studio/write/java8-support).

Kotlin: Kotlin is a modern, strongly-typed programming language with functional language features which runs on the Java Virtual Machine. For a brief introduction to Kotlin, please visit http://kotlinbyexample.org

In either case, user-interface parts of an Android app are coded in XML. The decision is yours which of these two languages you use to implement your Coinz app. However, please note that you should justify your decision for Java or Kotlin as part of your submission for Coursework 1.

Frequently asked questions

- *I don't have an Android device. I've never written an app before. How can I do this practical?*
  - You don't need to have an Android device 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.
- *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, which is an important service provided by a version control system.
- *Can I develop my app on DICE?*
  - Yes. Android Studio is also available on DICE, so you can develop on DICE if you prefer that.
- *Can I import libraries into my project and use code samples that I find online?*
  - Yes, provided that the software is *free* and you acknowledge your sources in your Coursework 2 submission. You may make use of examples coming from a publicly-available source such as Android documentation, Android tutorial examples, Android sample projects, Mapbox sample projects, Firebase sample projects, answers posted on StackOverflow, or open-source projects hosted on GitHub, GitLab, BitBucket or elsewhere.
- *Can I use paid-for services in my project?*
  - No. For example, you are not allowed to pay to register as a Google Play Developer and then use Google Play Games Services in your app. You cannot use Google Play Games services in your app, even in the case where you already are a registered and fully paid-up Google Play Developer. Similarly, you are not allowed to purchase commercial software licences for libraries or other assets which you then use in your app.
- *Instead of Java or Kotlin, can I implement my app in Dart/Python/Scala/C# etc?*
  - No, not for this practical. These languages are not officially supported for app development on Android so examples and documentation on how to use them are not widely available. We need all students to be working in similar conditions in order to make a fair assessment; using an unsupported language

would unfairly bias the problem against you in a way that we cannot correct for.

- *I already know Java, why on earth would I learn Kotlin to do this practical?*
  - Perhaps think of it as an investment in your skill set. Yes, you already know Java, *but so does everyone else*, so it doesn't make your CV stand out from the rest of the applicants for the job. Kotlin is the latest-and-greatest programming language for the Android platform, and can also be used for web development, and server-side development. It has been warmly embraced by the Android development community and there are many people who believe that it may be the future of Android development, becoming used in more and more projects, perhaps ultimately replacing Java.
- *Do I have to develop in Android Studio? I much prefer Atom/Eclipse/Emacs etc.*
  - You are required to submit an Android Studio project so we strongly recommend developing in Android Studio for this practical exercise. An Android project developed in Eclipse uses a different build system from Android Studio and can require some significant effort to be made to work in Android Studio. Android development in Eclipse has been unsupported since mid-2015 so you will not be able to access the latest libraries and Android services. Android Studio is based on Intellij IDEA and is a full-featured modern professional development environment. Android Studio also contains tools to convert Java code to Kotlin which these other platforms do not.
- *Is there a specified device for this practical or a specified Android version?*
  - No. You can choose an Android device and an Android version. If you have an Android device then you could choose a suitable specification for that device, to allow you to test your app on a real device. If you do not have an Android device then choose the emulator for a relatively recent device and a relatively recent version of the Android platform.
- *Who owns the intellectual property of the code that I submit? Can I make it public or extend if after the practical is over?*
  - You own the code which you produce for this practical; the University has no claim on any of it. After the practical is over you can extend or use your code in any way that you like. You can make it a part of your public portfolio of work and release it open-source or under any other software licence. However, please wait at least *fifteen calendar days* after the submission deadline for Coursework 2 before making your repository public.

Informatics Large Practical Coursework 1

Stephen Gilmore (`Stephen.Gilmore@ed.ac.uk`)
School of Informatics

1.1 Introduction

This coursework and the second coursework of the ILP are for credit; weighted 25%:75% respectively. Please now read Appendix A for information on good scholarly practice and the School's late submission policy.

— ◇ —

In this project you are creating an Android app. The first part of your work on this practical is creating a simple Android project, committing this to a version control system, and giving the course lecturer read access to your repository.

1.2 Using Android Studio

The recommended IDE for Android development is Android Studio, freely available for Mac, Windows and Linux systems. Android Studio is available for download from https://developer.android.com/studio/. If you are working on your laptop, please download and install the version for your architecture and operating system. For this practical we recommend using Android Studio 3.1.4, which is the latest official release at the time of writing, and is also the version which will be installed on DICE this semester.

— ◇ —

Android Studio is available on DICE as `/usr/bin/androidstudio3`. Assuming that `/usr/bin/` is in your PATH then simply type `androidstudio3` to run it. For easier access add the command `alias as3="/usr/bin/androidstudio3"` to your `~/.brc` file. Confirm that the version of Android Studio which is running is 3.1.4.

— ◇ —

Android Studio is a platform, not a single application, so when you begin to use it you will see that it periodically asks to download other components such as Android software libraries, Android device emulators, and the like. It will also periodically suggest that you upgrade components to the latest versions. Please note that you do not have sufficient permissions to upgrade Android Studio if working on DICE.

1.3 Creating a sample project

First, decide whether you are developing in Java or Kotlin. Then, create a new Android project of the appropriate type in your installation of Android Studio following the instructions at https://developer.android.com/studio/projects/create-project.html. Choose the "Phone and Tablet" form factor and API level. Unless you have a compelling reason not to, we recommend choosing Android 5.0 (Lollipop) API level 21 *or higher*. Backwards compatibility has not been identified as an issue for this project; we don't mind if your app doesn't run on old devices. Add an activity to your project such as a Basic Activity. Check whether you can run your project using the Android emulator or an Android device (if you have one). Even if you have an Android device, we recommend using the Android emulator when debugging your app, because it typically provides better diagnostic feedback in the case of errors than a physical device.

1.4 Using version control

In the Informatics Large Practical you will be creating Android project resources such as XML documents and Java or Kotlin files which will form part of your implementation, to be submitted in Coursework 2. These resources should be placed under version control in a source code repository using the Git version control system.

— ◇ —

The hosting service which we would like you to use is *GitHub* (https://github.com/). Although GitHub does not generally provide free private repositories it does have a free service which provides private repositories *for students*. To activate this you just have to provide your university email address at https://education.github.com/.

— ◇ —

Check your current Android project into your GitHub repository. It is not important what your project does at this stage, only that your repository contains a valid Android project.

— ◇ —

Add a standard `README.md` Markdown file to your repository which identifies you by name and university matriculation number.

— ◇ —

Please give GitHub user `sgilmore` read access to your repository. (Settings → Collaborators → "Add collaborator" in GitHub.)

1.5 Your project plan

The second part of this coursework for the ILP involves the creation of a project plan document, recording the decisions that you have made so far in the project, and presenting the plan of the implementation work which you will undertake to carry the project to completion.

— ◇ —

Your plan should record your design decisions for your app including a **list of the bonus features** which you plan to add. Your plans may change as the project progresses, and you may eventually deliver something which is different from the bonus features which you describe here, but you should record your intentions at this stage in any case.

— ◇ —

You should document your **rationale for choosing Java or Kotlin** as the development language of your project. Bear in mind that your Coinz app is being developed in the scenario where you will hand it off to a team of developers who will maintain and develop it further. For this reason, your rationale for choosing Kotlin or Java should be based on a consideration of the relative strengths and weaknesses of the two languages, rather than your personal taste in programming languages.

— ◇ —

Finally, your project plan should include a **week-by-week timetable** which runs from Week 2 to Week 13 showing what you have done so far, and what still remains to be done, in order that you can complete a well-engineered, well-tested, and well-documented project by the end of the semester. Your timetable should include concrete *milestones* which identify specific parts of the project which will be completed in particular weeks, in order to be able to measure your progress against the project plan.

1.6 How to submit

Please submit your ILP plan document from your DICE account using the command:

```
submit ilp cw1 ilp-plan.pdf
```

The expected length of this document is 5 pages, but shorter or longer submissions will also be accepted. The choice of font, font size, and margins is up to you but please consider the readability of your submission, and avoid very small font sizes.

— ◇ —

In order to streamline the processing of your submissions, and help avoid lost submissions, please name your PDF document `ilp-plan.pdf`. The submission format is PDF only; do not submit DOCX files, TXT files, or other formats.

Informatics Large Practical

Coursework 2

Stephen Gilmore (`Stephen.Gilmore@ed.ac.uk`)
School of Informatics

2.1 Introduction

As noted above, Coursework 1 and Coursework 2 of the ILP are for credit; weighted 25%:75% respectively. Information on good scholarly practice and the School's late submission policy is provided in Appendix A.

— ◇ —

The coursework is the report, and the implementation. The code which you submit for assessment should be readable, well-structured, and thoroughly tested. Any automated tests which you have written for your code should also be submitted. For your Android app, these should be in the `test` or `androidTest` folders of your project.

2.2 Constraints

This practical exercise uses *free software and free services.* There are two services that your app will depend on: a map service, and a hosted database service.

Map service: For our map service, we will use *Mapbox*[4]. Instructions for adding Mapbox to your app are available at https://www.mapbox.com/install/.

Database service: For our hosted database service we will use Google's *Firebase* platform (https://firebase.google.com) with the free Spark Plan. We will use *Cloud Firestore* as our database.

[4]It might seem that we are missing the most obvious and appropriate solution here, to use the *Google Maps Platform.* Unfortunately, from June 2018, Google requires developers to enable billing in their apps and to give their credit card details in order to register as a Google Maps developer. For this reason, we are not using Google Maps in this project. Mapbox does not require that developers give any credit card details.

2.3 Report on your implementation

You are to submit a report documenting your project containing the following.

- *A description of the algorithms and data structures used for the core functions of the implementation.* Describe your implementation of features of your app such as:
  - download and parsing of the Geo-JSON Coinz map from the server[5];
  - persistent storage of the user's progress in playing the game; and
  - detection of the coins which are close enough to be collected at the user's current location.
- *Parts of your design which have not been realised in the implementation.* It could be that you have not been able to implement something which you had planned in your design. If so, document that here.
- *Additional features of your implementation which were not described in your design.* It could be that you have implemented something which you had not planned in your design. If so, document that here.
- *Screenshots of your app in use.* Your report should be illustrated with a series of screenshots which have be taken either from the Android emulator, or from a physical Android device. Include as many as possible, in order to show all the features of your app in use. You should aim to show a typical play of the game including aspects such any startup or splash screens, the login dialogue, as the user viewing a map, banking a coin, and sending a coin to another player. Add additional screenshots as necessary to illustrate the bonus features which you have added to your app to make it unique.
- *Acknowledgements:* Include an acknowledgement for any part of your code which comes from a publicly-available source such as Android documentation, Android tutorial examples, Android sample projects, Mapbox sample projects, Firebase sample projects, answers posted on Stack Overflow, or open-source projects hosted on GitHub, GitLab, BitBucket or elsewhere. You do not need to include an acknowledgement for any code which was presented in the course lecture slides.

2.4 Preparing your submission

- We need your project source code (including XML layouts, Gradle build files, image files and any other assets) for assessment but the compiled version of your project can be regenerated so you can make your submitted file smaller in size by removing any precompiled code from your Android Studio project.

[5]Details of the server and the organisation of the maps on the server will be disclosed later.

- First, quit Android Studio, because it automatically rebuilds your project.
- Your project is in your home directory in the folder `AndroidStudioProjects`. Specifically, you can remove the folder `Coinz/app/build`. This folder will typically be several Mb in size and contains only the compiled image of your project which will be regenerated when your project is loaded into Android Studio for testing.
- Having removed the `app/build` folder, make a compressed version of your `Coinz` project folder using ZIP compression.
  - On Linux systems use the command `zip -r Coinz.zip Coinz`.
  - On Windows systems use Send to > Compressed (zipped) folder.
  - On Mac systems use File > Compress "Coinz".

  You should now have a file called `Coinz.zip`.

2.5 How to submit

Please submit your report and your implementation work from your DICE account using these commands:

```
submit ilp cw2 ilp-report.pdf
submit ilp cw2 Coinz.zip
```

In order to streamline the processing of your submissions, and help avoid lost submissions, please name your PDF document `ilp-report.pdf`. The submission format is PDF only; do not submit DOCX files, TXT files, or other formats.

— ◇ —

The expected length of this your project report is 15 pages, but shorter or longer submissions will also be accepted. Of this 15 pages, approximately 5 pages should be screenshots of your app in use. The choice of font, font size, and margins is up to you but please consider the readability of your submission, and avoid very small font sizes.

2.6 Things to consider

- Your submitted code will be read and assessed by a person, not a script. It is not a waste of time to add comments to your code, documenting your intentions. Your submitted code should be readable and clear.
- Logging statements (using `Log.d` and friends) and diagnostic print statements (using `println` and friends) are useful debugging tools for Android apps. You do not need to remove them from your submitted code. It is fine for these to appear in your submission.

- The game should be robust. Failing with a `NullPointerException` or other run-time error will be considered a serious fault.

Appendix A
Coursework Regulations

Good scholarly practice

Please remember the University requirement as regards all assessed work for credit. Details about this can be found at:

http://web.inf.ed.ac.uk/infweb/admin/policies/academic-misconduct

Furthermore, you are required to take reasonable measures to protect your assessed work from unauthorised access. For example, if you put any such work on a public repository then you must set access permissions appropriately (generally permitting access only to yourself, or your group in the case of group practicals).

— ◇ —

The Informatics Large Practical is not a group practical, so all work that you submit for assessment must be your own, or be acknowledged as coming from a publicly-available source such as Android documentation, Android tutorial examples, Android sample projects, Mapbox sample projects, Firebase sample projects, answers posted on Stack-Overflow, or open-source projects hosted on GitHub, GitLab, BitBucket or elsewhere.

Late submission policy

It may be that due to illness or other circumstances beyond your control that you need to submit work late. The School of Informatics late submission policy aligns with the university's Assessment Regulations which applies the following penalty: *5 percentage points will be deducted for every calendar day (or part thereof) it is late, up to a maximum of 7 calendar days.* However, to this University policy, the School of Informatics also adds the constraint that *late submission will only be accepted if no submission in time has been made.* For more information, see

http://web.inf.ed.ac.uk/infweb/student-services/ito/admin/coursework-projects/late-coursework-extension-requests

The staff of the Informatics Teaching Organisation will approve extension requests up to 7 days in accordance with the "good reasons" which are defined in the University's assessment regulations.

— ◇ —

Please note that things that would *not* be considered good reasons include (among other things) last-minute computer problems and loss of work through failing to backup your source code. Frequent commits of your work into a remote source code repository such as GitHub provide an off-site backup of your work which can allow you to continue from where you left off in the case of a disk or other hardware failure, so we recommend committing your work frequently.

Appendix B
The small print

Introduction

This appendix contains some details about the practical which are relevant to know, but are not perhaps the first things that you would think of to ask.

- *Deprecated methods.* Both Java and Kotlin *deprecate* methods, classes, and interfaces if they believe that their use in some way encourages bad programming style, or in the case where there is a newer, better way to implement the same functionality. However, the two languages do not always agree on which methods should be deprecated, and thus some methods are deprecated in one language and not in the other. For this reason, in order not to bias towards Java or towards Kotlin one way or another, no penalty will be applied if you use deprecated methods or classes.

- *Software updates.* Android Studio, the Android emulator, the Kotlin language, Java and the JVM are current, active software projects, and it is to be expected that updates to them will be announced regularly. We anticipate that several updates to Android Studio and the Android tools will be published during this semester, while the project is running. While operating system updates can often be security-critical and it is good practice to install them as they become available, updates to platforms such as Android Studio tend to be less security-critical and if you have a nicely-working installation of Android Studio, please don't feel obligated to update it to the latest version. No penalty will be applied if you are not using the latest version of the software, or the latest version of every library.

Appendix C
Using the Piazza Forum

Details

The Informatics Large Practical has a discussion forum on Piazza, available online at the address https://piazza.com/ed.ac.uk/fall2018/infr09051ilp/home.

Guidelines

Subscribing to the Informatics Large Practical Piazza forum is optional, but strongly encouraged. Questions posted to the forum may be answered by the course lecturer or by another student on the course. Please read the following notes to ensure that you have the best experience with the forum. These guidelines are based on several years of experience with course fora, where issues such as those below have arisen.

- Anonymous and pseudonymous posting on the forum is not allowed; please enrol for the course Piazza forum with your own name. Please be aware that, however they may appear to you, posts on the forum are not anonymous to the course lecturer. The forum is available only to students who are enrolled on the ILP this year. The course lecturer reserves the right to delete the enrolment of anyone who is not (or appears not to be) registered for the ILP.

- Especially when commenting on another student's work, please consider the feelings of the person receiving your message. Please refrain entirely from comments criticising the progress of another student. Each of us works at our own pace and there are many different possible orders in which to tackle the work of the ILP. Perhaps you finished implementing something in Week 4, but that does not mean that everyone did.

- If you find some content on the forum helpful, or think that it is making a useful contribution to the course, please acknowledge this by clicking "Good question" or "Good answer" as appropriate; this encourages continued participation in the forum. The course lecturer will endorse answers which he believes to be helpful.

- Forum postings which intend to correct factual errors or resolve ambiguities in the practical specification are welcome. If necessary, the course lecturer will update this coursework document to correct the error/resolve the ambiguity.

- Asking for help with installing Android Studio or the Android emulator is strongly encouraged. Installing the Android Studio IDE does not form part of the assessment in this practical so please ask as many questions about this as you wish. As usual, please be as specific as possible with any error messages that occur.
- When asking for help with fixing a run-time error, such as an exception, please include what seems to be the most relevant part of the diagnostic error message that you receive, but please include as little of your Java or Kotlin code as possible. The course lecturer may edit or delete your post if you include too much program code.
- The Informatics Large Practical is an individual programming project so you are not allowed to share your code with others. Please bear this in mind when answering questions on the forum; do not post your solution as an example for someone else to borrow from.
- Forum postings which ask for part of the solution to the practical are strongly discouraged. Examples in this category include questions of the form "What is the best way to implement X?" and "Can I have some hints on how to do Y?"
- Questions about the marking scheme for the practical are strongly discouraged. Examples in this category include questions of the form "Which of the following alternatives would get more marks?" and "How much detail is required for Z?"

Appendix D
Document version history

Version 1.0.0 (September 22, 2018): Initial version of this document issued.