ILP 2018/2019: COINZ project plan

Rado M. Kirilchev (s1452923)

1. Introduction

The ILP coursework specification mandates the development of an Android game by the name of *"Coinz"*. Core, required functionality includes the collection of geo-positioned "coins" by walking in close proximity to their locations; the differentiation of these coins into balances between the "bank account" and the "spare change", with a limited daily quota for "depositing" coins from the "spare change" amounts into the "bank account"; and finally, the ability for players to send extra "spare change" coins to other players.

This document presents some basic information about my rendition of the "Coinz" project, its respective Git repository, its bonus features, and the project timetable. I introduce my rendition of the game, henceforth stylised as COINZ (in all caps, standing for *Clandestine Operation Imminent: Nocturnal Ziggurat*), written in Kotlin, with a number of bonus features: multiple currency balances in the bank; the buying and selling of the four collectable currencies for GOLD; a simple narrative and two teams for players to join; a progression system that rewards players with improving team-specific bonuses; the purchasing of an additional "currency" (computing power, also "Compute") and the "decryption" of "secret messages" using it, introducing a global victory condition.

2. Github repository

The Github reposittory for the COINZ project can be found at https://github.com/radoMK/coinz. The repo contains the source files for the project, as well as the source files for the graphical assets (icons and backgrounds). The README file also contains a list of all features of the game – core and bonus features.

3. Kotlin: the language of choice

I elected to use Kotlin as the backbone language for the COINZ project. The reasons for this choice are manifold, and this section shall provide a brief discussion of them.

Kotlin is a relatively new language, developed by JetBrains of IntelliJ IDEA fame (and in turn, Android Studio, to an extent). Since 2017, Kotlin has been openly supported by Google as an alternative to Java for Android development. In comparison with Java, Kotlin adds a number of features absent in Java, such as *(amongst many others)* extension methods, flexible lambdas and lambda expressions, function inlining, explicit usage of nullable and non-nullable types to enforce a more rigorous form of null-safety, "smart casting", delegation of properties and functions, properties as class members, string templatisation, compile-time type inference for variables and properties, anonymous inner classes, a built-in support for singletons, coroutines, operator overloading, and a slightly odd separation of mutable and immutable collections.

A discerning eye may notice that many of these features mirror others present in other languages, such as C#, which is an infinitely more pleasing language to use than Java. In addition, Kotlin addresses a number of "issues" present in Java, such as the aforementioned null-safety problems, a removal of the silly "unchecked exceptions", a revamp of all generics to eliminate raw collection types and wildcards, a built-in support for function types, and many others. *(Information sourced from the Kotlin official site.)*

Kotlin is much easier to use and much more flexible than Java, whilst maintaining perfect interoperability. Coupled with its overwhelming support by Google and JetBrains, Kotlin shines as a wonderful choice. It is all but guaranteed that the language shall continue being supported in the future, and its relative clarity and conciseness should help the "support development team" (that I shall be handing the app to) to maintain it with maximal ease and least cost for the longest possible amount of time.

And last, but not least, I've used Java extensively in the past, and I sincerely dislike it.

4. Planned bonus features

This section will list all planned bonus features for the game, as well as a brief elaboration on the reasons for their presence. **Note** that bonus features have been listed together to reduce the number of subsections present, and to help facilitate a more cogent reading experience for this document. In truth, every comma-separated feature listed in the subsection titles may be considered a feature to some extent.

4.1. Teams, the global win condition, and narrative

Upon registration, every COINZ player must select a team. Teams may not be changed afterwards. The two available teams are:

- The **Eleventh Echelon**: a fictional division of the MI5;
- The **Crimson Dawn**: a fictional group of ruthless, high-tech mercenaries.

The division of teams presents a natural, global win condition for the game. This, in turn, ties in with the narrative, which shall be addressed now.

The "plot" of the game, which can also be considered a bonus feature, is simple, allowing the players to engross themselves in the game without much effort. A Soviet-era nuclear warhead has gone missing, misplaced by the Kyrgyzstani government. The bomb is held by an unknown organisation. The Eleventh Echelon has been dispatched to locate and secure the bomb before anything goes awry; the Crimson Dawn, hired by another unknown party, seeks to also locate the bomb in order to sell it.

Every secret message decrypted by the players (see section 4.3) brings the respective player's team a little bit closer to filling up a global "readiness" meter, representing the respective organisation's knowledge of the location of the bomb. Whichever team reaches 100% readiness first wins the game.

In addition, sending coins to other players is restricted: coins may only be sent to players on the same team. Teams, as well as the global win condition, create a natural sense of competition and tension, as players strive to outperform the opposing team (global "readiness" information is updated in real time within the app) in order to secure the victory for themselves. The "required" feature of the daily deposit quota mandates that players send coins to each other to maximise efficiency; and as the purpose of the coins is to (indirectly) win the game for the team, only sending coins to teammates makes a logical sense and helps to heighten the need for cooperation (whilst also magnifying the competitive effect).

4.2. Currencies, bank balances, "spare change" wallets, exchange rates

As per the requested features, every player has a unique "bank account". In COINZ, this account may hold balances for all five currencies: GOLD, as well as the four collectible currencies, namely DOLR, PENY, SHIL, and QUID.

In COINZ, these currencies are given life with a special icon for each, all of them a symbol inspired by classical real-world currencies: GOLD is represented by a capital "G" with a second horizontal bar under the first; PENY is represented by a capital "P" with a smaller "Y" to the side; DOLR is represented by a capital "D" with two "euro-style" parallel, horizontal bars on the vertical line; SHIL is represented by a capital "S" with two "euro-style" bars at a 45-degree angle in the middle; and QUID is represented by a capital "Q" with two angled, parallel bars acting as a descender. *(The icons for all currencies may be found in the "assets" folder of the Git repository.)* These icons are also used as coin markers on the map, and are given different colours to ensure easier identification and a better visual experience.

"Spare change", one of the features by the specification, is also present, but with a twist. In COINZ, the player may hold up to a given amount of "spare change" of any one currency at any one time. This is referred to as the "wallet size", and it is a variable dependent on the player's team and current level (see section 4.4). The wallet size is per currency, not cumulative, e.g. a level 1 player, member of the Eleventh Echelon, may collect and hold up to 15 coins of each currency as "spare change". Coins from these "spare change" wallets may be deposited into the bank account, incrementing the specific currency balance. This is naturally bounded by the daily deposit quota, which is fixed at 25 *total* units of currency. However, the players may deposit as many coins of each currency as they want (the minimum deposit transaction being 0.01 of any collectible currency).

The bank also provides players with the ability to buy and sell any of the four collectible currencies in exchange for GOLD. The exchange rates are determined by the daily rates sourced from the map's *.geojson* file; however, the bank provides these currencies with a fixed 4.5% commission rate, rendering the buying and selling process less effective than simply sending collected coins to teammates. (The commission rate is reduced for members of the Eleventh Echelon: see section 4.4.)

Coins may also be sent to other players as per the specification document; in COINZ, they may only be sent to teammates. Unique coin identifiers are disregarded completely; players can simply send some balance of any collectible currency, currently found int their "spare change" wallet, to another player of the same team. The balance sent is converted to GOLD (as per the sender's commission rate) and added to the recipient's GOLD balance. The sender is additionally awarded with some experience points for sending the sum (again, see section 4.4).

Since currencies in the bank balances are used to purchase "Compute", which is in turn used to "decrypt secret messages" and improve the team's global "readiness" score to achieve victory (see section 4.3), and since different "compute providers" only take different currencies, each and every currency has a role to play in the game. The ability to buy and sell currencies, coupled with the ability to send coins to teammates, helps players maximise their own efficiency whilst still working as a part of the team.

In addition, given that GOLD is the only currency that is not collected as coins, and is the only currency that all others can be converted to and from, this feature also helps establish GOLD as the de-facto "mediator" currency, and provides players with the ability to hold GOLD as "liquid assets", using it to purchase whatever currency is most optimal given the daily rates, or simply to bolster the player's current balance of another currency before using it to purchase Compute.

4.3. "Compute" and the decryption of "cryptomessages"

Accumulating as much money as possible, the default goal of the game as suggested by the specification document, intrinsically lacks the teamwork aspect that the coin-sending required feature suggests. As a result, the (slightly convoluted) "secret message" system has been introduced in COINZ.

The premise isn't complex: to locate the missing nuclear bomb, players must decrypt "secret messages", also called "cryptomessages". Every cryptomessage decrypted increases the team's global readiness score (victory counter) by a small amount, ranging from 0.003% to 0.040%. (Reminder: once this percentage reaches 100% for a team, the entire team wins the game.) However, decrypting messages requires *computing power*, henceforth referred to as "Compute".

Every cryptomessage has a difficulty rating that determines the readiness bonus it provides, as well as the Compute price for its decryption. These values, as well as the number of cryptomessages available on any one day, are partially randomised, allowing players to pick and choose which messages they'd like to decrypt first. In addition, every decrypted cryptomessage awards players with a modest amount of experience points (see section 4.4).

Compute is purchased using the player's bank balances, giving a natural use for the different currencies, and opening various strategies up to the players. (Members of the Crimson Dawn also receive discounted Compute prices as a team bonus; details listed in section 4.4)

4.4. Experience points, levels, team-specific bonuses, and titles

Progression systems are a widespread game mechanic, giving players something to play for and improving retention rates. In COINZ, the progression is simple: collecting coins, sending coins to teammates, and decrypting messages all give experience points to the player. Accumulating experience points improves the player's level, which, in turn, provides team-specific bonuses that improve as the player's level increases.

Members of the Eleventh Echelon will get lower bank commission rates but smaller wallet sizes, driving them towards a more cooperative playstyle. Their opponents from the Crimson Dawn, however, get larger wallet sizes and slightly discounted Compute prices, allowing them a little more self-sufficiency. Note that members of the Crimson Dawn are still bounded by the daily deposit quota of 25 coins in total, which forces them to also cooperate with each other.

In addition, increasing levels provides a better "title" for the player, e.g. ranging from "Analyst" to "Senior Agent" for the Eleventh Echelon. This feature is completely cosmetic, but helps provide the player with a little extra motivation for completing objectives.

5. Project timetable

This section provides a week-by-week timetable for the project, including brief descriptions for activities planned to be undertaken during the respective weeks.

- **Week 2:** Preliminary work. Repository, graphical assets, Firebase setup, user interface design preparations (establishing the total number of activities required, etc.).
- **Week 3:** Feature implementation. User interface design and functionality. User accounts, database connectivity, rudimentary functionality and feature implementation.
- **Week 4: (Milestone 1:)** Feature completeness. At this point, all features should be feature-complete, and the app should effectively be in an almost-usable state.
- **Week 5:** Manual testing for side effects, unwanted behaviours, and bugs.
- **Week 6:** Additional testing. Resolve any underlying systems issues, such as unreliable mapping behaviour, UI quirks, etc.
- **Week 7:** Add automated tests to validate application stability and lack of side effects.
- **Week 8:** Begin polishing cycle. Minor graphical enhancements, animations, transitions if necessary.
- **Week 9:** Additional polishing, if necessary.
- **Week 10: (Milestone 2:)** Application finished, polished, and ready to ship.
- **Week 11:** Final testing, manual and automated.
- **Week 12:** Final testing, manual and automated.
- **Week 13: (Milestone 3:)** Project fully ready and submitted.

The three distinct milestones specified ought to provide sufficient grounds for progress verification. In addition, the ample time left for application testing ought to ensure optimal final performance and lack of any side effects or issues in the final version of the product.

The descriptions listed for every week reflect the work done over that week, and/or the overall project status at the end of that week.