The Chinese University of Hong Kong

Department of Computer Science and Engineering

CSCI 4140 Project Report

Group 22

Poon Bing Chun 1155063232 (did everything)

Project Title: Tempus E Spatium (Latin for *Space and Time*)

GitHub repository: https://github.com/SoftFeta/tempusespatium

Since I don't have a Google Play dev account, visit this page to download the APK: https://github.com/SoftFeta/tempusespatium/releases

5-minute video: https://www.youtube.com/watch?v=IfJnnvukqHI&feature=youtu.be

1 TABLE OF CONTENTS

2 OVERVIEW

From the proposal…

The downsized project is named 'Tempus E Spatium' (Latin for Time and Space, which is often prefixed by "lost in"). Quiz types include: Fill in the blanks, drag and drop, listen to the national anthem and pin the map, multiple choice, and complete the word. Despite the name of the app, the last type can be of any question type. You can search for any topic as long as it is available in Wikipedia. There are 1000000+ possible questions generated from articles that belongs to 700+ WikiProjects: children's literature, archaeology, Scottish castles, free software, animal anatomy, Canada roads, military history, relativity, constructed languages, Michael Jackson... you name it, you (probably) have it.

The questions are timed 5 seconds to 1 minute each, depending on the question type and difficulty settings. A game consists of 5 rounds. Each round has its own chessboard. A

player advances a variable number of tiles on answering a question correctly, and retreats on answering it incorrectly or failing to answer it on time. The player who reaches the destination first wins the round. The player who wins the most rounds wins the game.

3 IMPACT OF THE APPLICATION / PROBLEMS

From the proposal…

- Not many games on the Play Store are educative, and most edutainment apps are for kids. By contrast, this game is designed to be **hard enough for adults**, including the intellectuals.
- Topic diversity: Any topic, as long as it is available in Wikipedia. 1000000+ possible questions generated from articles that belong to 700+ WikiProjects: children's literature, archaeology, Scottish castles, free software, animal anatomy, Canada roads, military history, relativity, constructed languages, Michael Jackson... you name it, you (probably) have it.
- The questions themselves are multilingual, as their content are taken from French/German/Spanish Wikipedia. This means you can harness the app to **practise foreign languages as well**.

4 CREATIVITY / GOALS / CHALLENGES

From the proposal…

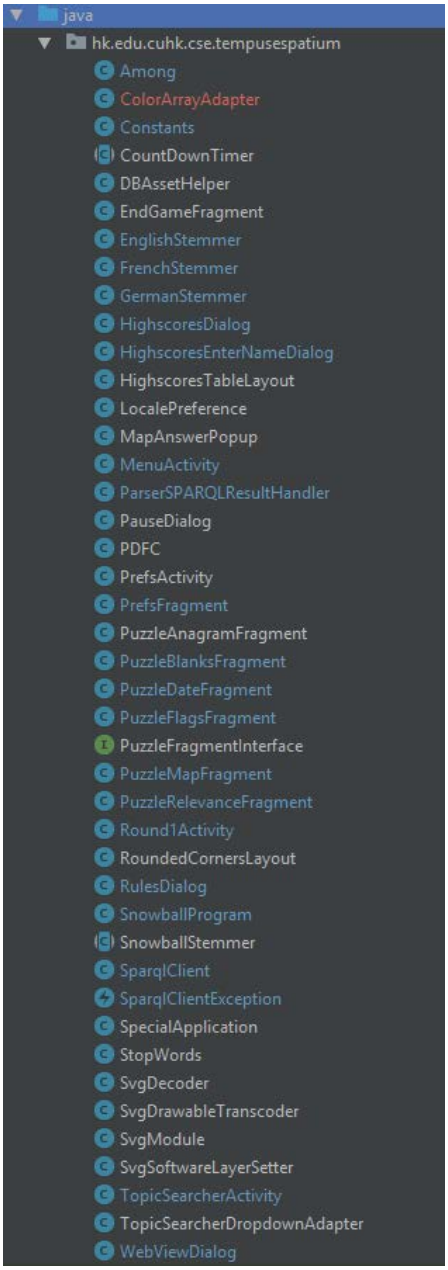
- *Tempus E Spatium* is one of the few apps that takes advantage of **multitouch**, which eliminates network latency in many 2-player games. It is a departure from the traditional 'pass-and-play' model, too.
- The app makes use of **web scraping** fetches and parses Wikipedia articles and turn them into questions, which virtually no games are using this technique at the moment.
- The app belongs to the new field of **'edutainment'**: learning masquerading as playing. As the saying goes, 'all work and no play makes Jack a dull boy.' The app focuses on the holistic development of a person instead of turning him a nerd, by raising cultural awareness and sharing knowledge that is oft-neglected.
- **The soft keyboards are specially designed.** Keyboard layouts include **QWERTY, AZERTY (French) and QWERTZ (German)**, no additional downloads are needed. Note that there are two keyboards on each side, and each of them only enters text into their text fields. I use advanced techniques including Android's @JavascriptInterface to create the **illusion of two cursors**, overcoming two major technical difficulties of having one cursor/focused item, and the lack of direct two-way communication channel between WebView widgets and JavaScript.
- One minigame involves **natural language processing**. You can find details in the Evaluation Plan part.

5 TECHNOLOGIES USED

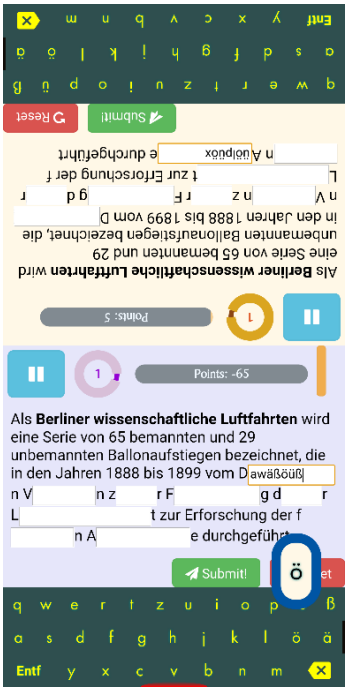
- Semantic web technologies – A web of data (or data web) that can be processed by machines – that is, one in which much of the meaning is machine-readable.
    - Web scraping (traditional) – specify a 'base website' and get its content, then follow its hyperlinks until some useful data is found
        - **XPath** – query language for writing crawlers, like a DOM selector but with predicates, wildcards, math functions and operators. (Supported by *org.w3c.dom* and *javax.xml.xpath.xpath* classes).
        - **Regular expressions** – looks for interesting patterns
    - Semantic queries (newer) – web content interpreted as a knowledge base of named entities and relations.
        - **SPARQL** – query langauge for semantic queries which is very different from traditional SQL. It has the syntax of a logic language like Prolog, which means the main statements of the query are predicates. Specialised in interpret named graphs, linked-data, or triples (which are, subject-predicate-object tuples).
- Stemming
    - Snowball stemmer
        - Converting stemming rules to Java code
- Google Maps API
    - For showing, styling and pinpointing maps
- Google Geolocator API
    - For converting coordinates to country name
- SoundPool and MediaPlayer
    - Playing short sounds (like whoosh), and streaming music from the web respectively
- GeoJSON
    - The Natural Earth Data set is presented as a GeoJSON document
    - For drawing country borders
- SQLite
    - For storing highscores
- AdMob API
    - For showing (sample) ads
- @JavaScriptInterface
    - For interacting with the WebView indirectly (e.g. call Java functions in WebView)
- Asyncronous programming
    - CountdownLatch, SystemClock.sleep, AsyncTask, Runnable…
- SVG
    - ImageView does not support SVG (not to be confused with vector drawable). One has to convert it to a Drawable in complex procedures.
- Various libraries
    - OKHttp
        - For sending HTTP requests (e.g. getting webpages)
    - Glide
        - For loading and caching images from the web

- Android-Bootstrap
  - Interactive elements (like buttons and drop-down menus) in Bootstrap style
- Drag & drop
  - For the relevance game described below
- Android Design Library
  - Snackbar (like Toast)
- Etc.

6 PROJECT STRUCTURE



40 classes, 2 abstract classes and 1 interface. See the code for details.



7 Features

The app includes 5 minigames:

- Fill in the blanks (from sections of Wikipedia articles)
  - Thousands of topics in five languages, which correspond to millions of articles
- Match three category to the most relevant words
  - Thousands of topics in five languages (the other two categories are randomly chosen)
- Choose the correct arms/flag
  - Customisable: country name, capital, population, …
- Hear the song and pin the map
  - Zooms to the geological region
- Double rollers, scroll the correct date
  - Customisable: user may enter the Wikidata identifier (e.g. Q7944 is earthquake, Q168983 is conflagration) Notice that keyword searching is not implemented because ‘siege’ will probably return ten entities with the same name, in which only one is the real entity. Theoretically thousands of topics.

The first two minigames scrapes from Wikipedia of the specified language, while the last three uses semantic queries from Wikidata. There is a ‘Learn’ feature where the is supposed to read the Wikipedia articles before he plays the minigames involving them.

Settings:

- Changing UI locale (separated from question locale)
- Two difficulty levels: hard and InSaNe (shortens the response time of timers)
- Changing themes for both players

Others:

- Pausing the game
- Highscores
- Easter egg: endgame animation

8 Further development

- There are currently five minigames in my application. I would like to add more minigames to it.
  - Maybe a game where pictured meteorites are falling down from the top, and one has to type the name of the picture before the meteorite crashes the Earth…
- Use NDK and OpenGL for better graphics