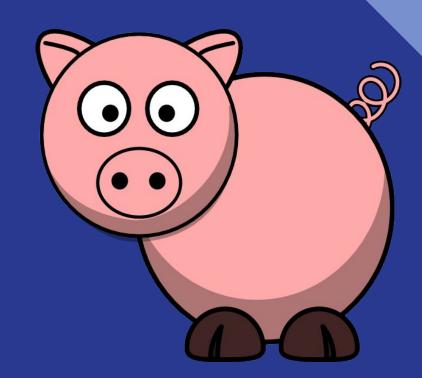
Project QuiqStiq

Team Pig



SRS

Project Requirements

Project Requirements

- 1. Application supports any chunk of text.
- 2. Improves retention of the text in the user's long-term memory.
- 3. Multiple user activities.
- 4. The software doesn't have the user type out full words.
- 5. The software shows proficiency level on lines and sets.
 - a. Proficiency increments with success.
 - b. Proficiency decrements with failure.

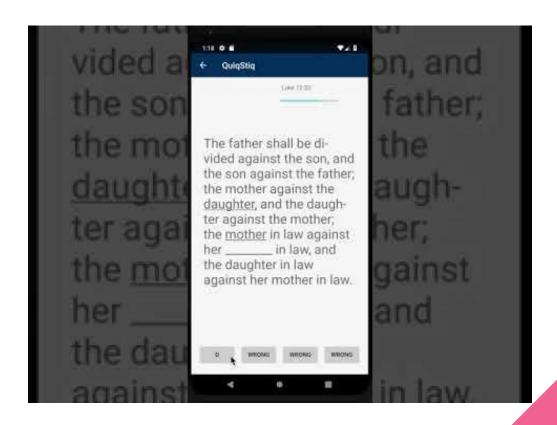
Project Requirements

- 1. Has an interactive UX.
- 2. The software supports an android version.
- 3. Supports one user per phone

Demo

Working Software

Demo



Non-goals

Next steps

- Backend Server
- Complete Texts
- Gamification
- User Settings

Backend Server

Multiple Users

- Multiple users will be allowed for any device.

Recommended Activity

- The app will be able to take a users data and recommend an activity to help their memorization based on their last performance.

Memorization Research

- Extended research will be available on which activities and intervals are most efficient for memorization.

Complete Texts

Bible

- Various complete versions of the Bible will be available, as well as different languages and a speech-to-text options.

Other Texts

 Other books and documents will be available for memorization, such as the "Declaration of Independence."

Gamification

Badges

- Assigned based off of streaks, verses, chapters, packages, and books.

Leaderboard

- A global leaderboard will be in place, showing the number of badges achieved.

Experience

Levels of experience will be awarded based off badges:
 Beginners, intermediate, skilled, advanced, and expert.

User Settings

User Profile

- The user will be able to create a username and password, as well as customizing their profile, such as Bible version and language.

Difficulty Level

- The user will be able to select their difficulty for the activities.

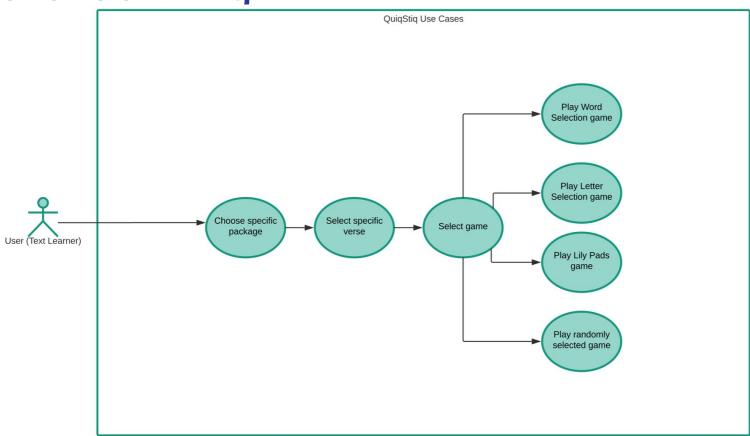
Custom and Group Packages

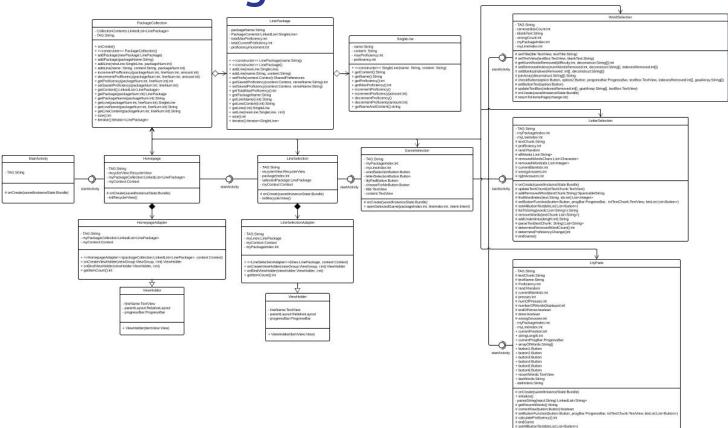
- The user will be able to connect with others and share custom made packages.

Diagrams

- UML Class
- Use Case

Use Case Diagram





PackageCollection inkedList<LinePackage

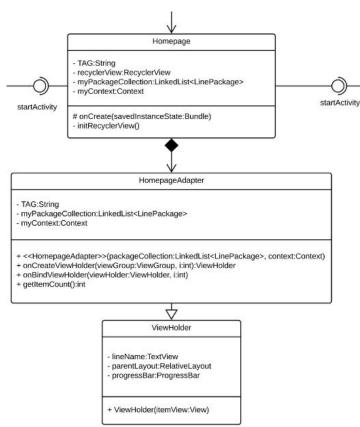
- CollectionContents:LinkedList<LinePackage>
- TAG:String
- + onCreate()
- + <<constructer>> PackageCollection()
- + addPackage(newPackage:LinePackage)
- + addPackage(packageName:String)
- + addLine(newLine:SingleLine, packageNum:int)
- + addLine(name: String, content:String, packageNum:int)
- + incrementProfficeincy(packageNum:int, lineNum:int, amount:int)+ decrementProfficeincy(packageNum:int, lineNum:int, amount:int)
- + getProficiency(packageNum:int, lineNum:int):int
- + setSavedProficiency(packageNum:int, lineNum:int)
- + getContent():LinkedList<LinePackage>
- + getPackage(packageNum:int):LinePackage
- + getPackageName(packageNum:int):String
- + getLine(packageNum:int, lineNum:int):SingleLine
- + getLineName(packageNum:int, lineNum:int):String
- + getLineContent(packageNum:int, lineNum:int):String
- + size():int
- + iterator():Iterator<LinePackage>

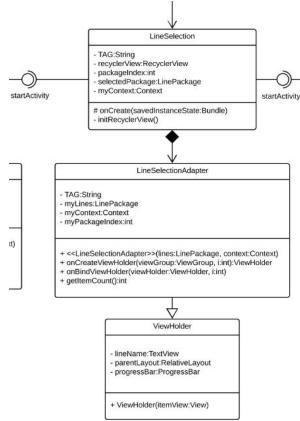
LinePackage

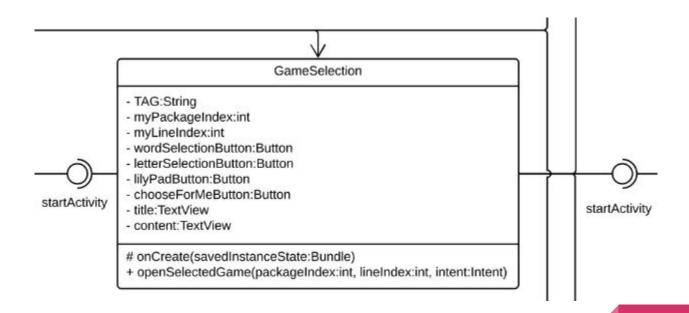
- packageName:String
- PackageContents:LinkedList<SingleLine>
- totalMaxProficiency:int
- totalCurrentProficiency:int
- proficiencyIncrement:int
- + <<constructer>> LinePackage(name:String)
- + <<constructer>> LinePackage()
- + addLine(newLine:SingleLine)
- + addLine(name:String, content:String)
- + setPrefs(context:Context):SharedPreferences
- + getSavedProficiency(context:Context, verseName:String):int
- + setSavedProficiency(context:Context, verseName:String).ii
 + setSavedProficiency(context:Context, verseName:String)
- + getTotalMaxProficiency():int
- + getPackageName:String
- + getLineName(i:int):String
- + getLineContent(i:int):String
- + getLine(i:int):SingleLine
- + setLine(newLine:SingleLine, i:int)
- + size():int
- + iterator():Iterator<SingleLine>

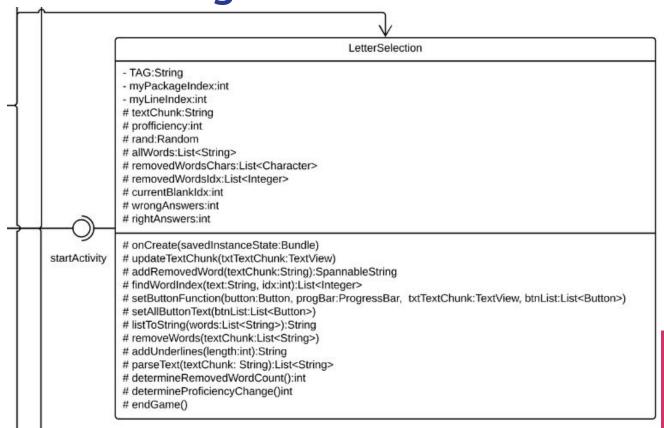
SingleLine

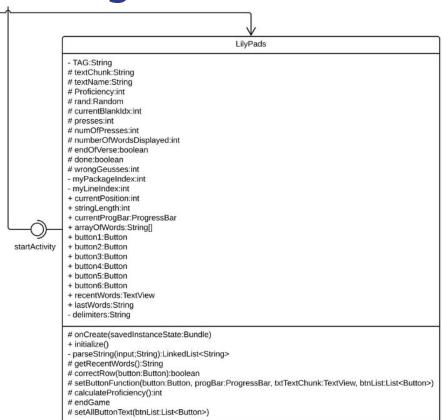
- name:String
- content: String
- maxProficiency:int
- proficiency:int
- + <<constructer>> SingleLine(name: String, content: String)
- + getContent():String
- + getName():String
- + getProficiency():int
- + getMaxProficiency():int
- + incrementProficiency()
- + incrementProficiency(amount:int)
- + decrementProficiency()
- + decrementProficienty(amount:int)
- + getNameAndContent():string



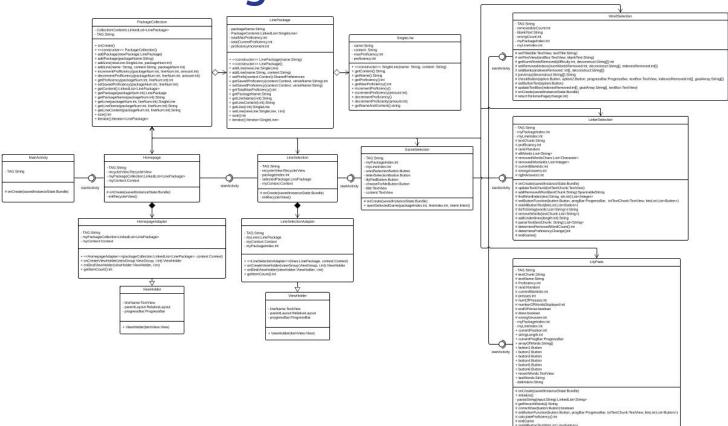








WordSelection - TAG:String removedIdxCount:int - blankText:String - wrongCount:int - myPackageIndex:int - myLineIndex:int # setTitle(title:TextView, textTitle:String) # setTextView(textBox:TextView, blankText:String) # getNumWordsRemoved(difficulty:int, deconstruct:String[]):int # setRemovedIndexs(numWordsRemoved:int, deconstruct:String[], indexesRemoved:int[]) startActivity # setBlanks(indexesRemoved: int[], deconstruct:String[]) # joinArray(deconstruct:String[]):String # checkButton(option:Button, option2:Button, progressBar;ProgressBar, textBox;TextView, indexesRemoved:int[], goalArray;String[]) # setButtonText(option:Button) # updateTextBox(indexesRemoved:int[], goalArray:String[], textBox:TextView) # onCreate(savedInstanceState:Bundle) # returnToHomePage(change:int)



- Android Studio
- Git process
- UML
- Iterative process
- Working in the large
- Working in a group

- Save data between runs
- Realistic expectations of time requirements
- Structure of an android app
- Serializable functionality in Java

- Not to use .docx
- Bible verses
- Engineering isn't just for 2nd North Euler
- Python doesn't have ++

Questions?



Arduino Studio