MultiQuiz App (Part 3 of 3)

In this third and final part of the MultiQuiz project, we will update our app to display summary counts at the end of the quiz. Once the user reviews the summary, the Back button (or gesture) can be used to return to the quiz again. While viewing the summary activity, the user may optionally reset the quiz before clicking Back to completely start everything again. We will also incorporate several test cases into our project.

Getting Started

First, please make a copy of your P1B project by exporting it via File | Export | Export to Zip File and select a safe name/location for the backup file that isn't within the project folder.

Next, you must get some new updates from our remote Git repository. Select Git | Update Project and Merge the incoming changes.

These updates provide the following changes specific to Project 1C:

* **build.gradle.kts** (Module :app)
  + This configuration file has been updated to enable testing of intents, and to disable animations during instrumented testing
* **MultiQuizTest.kt**
  + This file contains 24 instrumented tests that your implementation should pass. Please review the test code carefully, as you need to understand what the test cases are expecting. Further, in future projects, you may need to write test cases of your own.

After updating, please don't forget to use File | Sync Project with Gradle Files to ensure the build.gradle updates are incorporated. The IDE should prompt you to do this.

**Important note:** You must use these files as provided. During the development process, you may comment-out any test cases that don't compile, but you must re-enable them before submission. Your testing score may be zero if the test file – as provided – doesn't compile with your implementation as submitted. Beware that refactoring identifiers in the layout might unexpectedly alter the test file. Right-click the test file and select Git | Rollback if you need to return to the original contents.

Summary Activity

When creating the new SummaryActivity, please select "Empty Views Activity" (rather than "Empty Activity" as shown in BNRG).

Layout

Please use nested LinearLayouts, along with TextView, Button, and Space views to create the layout.

A screenshot of a cell phone

Description automatically generated

[[enlarge image](https://canvas.vt.edu/courses/185409/files/31461567?wrap=1)]

The layout for the new activity must include a "Quiz Summary" heading, labels for the summary counts, values for the summary counts, and a Reset All button. There should be equal vertical spacing above the heading, above the stats, above the button, and below the button. The heading and Reset All button should be horizontally centered. Each label-view combination of stats should be horizontally centered. The IDs for certain views must be as follows:

* correct\_answers\_count
* hints\_used\_count
* reset\_all\_button

The heading and labels don't need any ID values. A padding of 8dp should be applied to each of the count views. The text size of the heading should be 24sp, and the font should be bold.

Strings

The following should be added to the strings.xml file:

|  |  |
| --- | --- |
| **String name** | **String value** |
| summary\_heading | "Quiz Summary" |
| correct\_answers\_label | "Correct Answers:" |
| hints\_used\_label | "Hints Used:" |
| reset\_all\_button | "Reset All" |

No user-facing strings should be hard-coded within the app, with the exception of the string representations of the Int values of the counts.

New Intent

A newIntent function must be implemented, within the companion object of the SummaryActivity, using the technique shown in Listing 7.8. The function signature must be:

fun newIntent(  
    packageContext: Context,  
    correctAnswers: Int,  
    hintsUsed: Int  
): Intent { ... }

This function will be used during testing, so it's important to use this exact signature.

Behavior

The summary activity will be reached by clicking the Submit button from the final question. The two count values must be passed as two extra Int values, from the main activity to the summary activity, using the techniques demonstrated in BNRG Chapter 7.

The Reset All button behavior is as follows:

* Before the Reset All button is clicked:
  + Nothing changes in the summary activity
  + Upon returning to the main activity via Back:
    - The first question and its answers will be displayed
    - Every button state must be as it was before reaching the summary screen (for all questions)
* After the Reset All button is clicked (if it is clicked at all):
  + Both summary counts should display as zero
  + The Reset All button becomes disabled
  + Upon returning to the main activity via Back:
    - The first question and its answers will be displayed
    - Everything should revert to the state when the app has just been launched for the first time:
      * All summary counts should be reset to zero
      * Every answer button will be enabled and deselected (for all questions)
      * The Hint button will be enabled, and the submit button will be disabled (for all questions)

Rotation must be supported at any time, from either activity, without losing any application state. Previously you should have already handled rotation of the MainActivity, so now this must be done with SummaryActivity (which is essentially the same as the first challenge of BNRG Chapter 7).

However, please note that saving data across process death is **not** required for any projects this semester. Specifically, pleas **don't** use any SavedStateHandle as in Listing 7.15.

The user will remain in the summary screen until the device's **Back** button/gesture is used to return to the main screen. No code is needed to handle the Back navigation, since this is handled entirely by Android. If the Reset All button was clicked, then the summary activity must return a single Boolean value to the main activity as an indicator.

Functional Programming

While developing the app, be sure to use Kotlin's functional programming features, rather than an imperative programming style. In this part of the project, the apply and flatMap functions may be useful (although neither is strictly required).

Intent Naming Requirements

The extra values passed from MainActivity to SummaryActivity must have the following key names and types:

|  |  |
| --- | --- |
| Key name | Value type |
| edu.vt.cs5254.multiquiz.correct\_answers | Int |
| edu.vt.cs5254.multiquiz.hints\_used | Int |

The extra value returned from SummaryActivity back to MainActivity, to indicate whether the Reset All button was clicked, must have the following key name and type:

|  |  |
| --- | --- |
| Key name | Value type |
| edu.vt.cs5254.multiquiz.reset\_all | Boolean |

You may use any names for the above constants within your code.