Quiz App

Design and implement a short quiz app about some topic you are familiar with.

Design and implement a short quiz app about some topic you are familiar with.

Before You Submit

Be sure you follow the steps to prepare your project for submission, clean your build, review the project rubric, and zip your project files.

Submitting Your Project

To submit your project, you will upload your zip file.

In order to help project reviewers test and troubleshoot the project, please provide the devices (emulator and physical) and API level on which you tested your work. Note, providing this information does not guarantee the review will be done on the exact same device type(s); however, it will assist in providing a more efficient review experience for you as a student.

You can also provide any other additional notes to the reviewer in that section.

What to Expect After Submitting

It can take us up to a week to grade the project, but in most cases it is much faster. You will get an email when your submission has been reviewed.

Your reviewer will provide a code review with helpful comments and review your project against the project rubric.

# Quiz App

Layout

| **Criteria** | **Meet Specification** |
| --- | --- |
| Overall Layout | App contains 4 - 10 questions, including at least one check box, one radio button, and one text entry. |
| Question types | Questions are in a variety of formats such as free text response, checkboxes, and radio buttons.  Checkboxes are only used for questions with multiple right answers. Radio buttons are only used for questions with a single right answer. |
| Submit button | App includes a button for the user to submit their answers and receive a score. |
| Layout best practices | The code adheres to all of the following best practices:   * Text sizes are defined in sp * Lengths are defined in dp * Padding and margin is used appropriately, such that the views are not crammed up against each other. |
| View variety | The app includes at least four of the following Views: TextView, ImageView, Button, Checkbox, EditText, LinearLayout, RelativeLayout, ScrollView, RadioButton, RadioGroup.  If applicable, the app uses nested ViewGroups to reduce the complexity of the layout. |
| Rotation | The app gracefully handles displaying all the content on screen when rotated. Either by updating the layout, adding a scrollable feature or some other mechanism that adheres to Android development guidelines. |

Functionality

| **Criteria** | **Meet Specification** |
| --- | --- |
| Runtime Errors | The code runs without errors. |
| Question Answers | Each question has a correct answer. |
| Radio Button Implementation | Any question which uses radio buttons allows only one to be checked at once. |
| Control Statements | The app contains at least one if/else statement |
| Grading Button Function | The grading button displays a toast which accurately displays the results of the quiz. |
| Grading Logic | The grading logic checks each answer correctly. The app accurately calculates the number of correct answers and does not include incorrect answers in the count.  Note: When applicable, in the grading logic remember to check that the correct answers are checked AND the incorrect answers are not checked. |

Code Readability

| **Criteria** | **Meet Specification** |
| --- | --- |
| Naming Conventions | All variables, methods, and resource IDs are descriptively named such that another developer reading the code can easily understand their function. |
| Format | The code is properly formatted i.e. there are no unnecessary blank lines; there are no unused variables or methods; there is no commented out code. The code also has proper indentation when defining variables and methods. |