Lab 4: Your First Android App, the Model-View-Controller and the Activity Lifecycle

In this assignment you will create an app called GeoQuiz. GeoQuiz tests the user's knowledge of geography. The user will press TRUE or FALSE to answer a question and GeoQuiz will provide instant feedback. You will allow for more than one question and provide the user with a next button to navigate through the questions. You will also create a landscape layout and experiment with the various lifecycle events of an Activity.

Part A

1. Create a new Android Studio project called "GeoQuiz", create a GitHub repository and link it to your Android Studio project, then perform an initial commit and push.

   IMPORTANT: be sure to pick Java as your language when creating a project.

2. Edit the string resource file and add strings for:
   a) The text of your question (e.g. "Canberra is the capital of Australia.")
   b) The label text of your true button
   c) The label text for your false button

3. Open the layout file and add the following using LinearLayouts:
   a) A TextView, use your question text string resource as it's text
   b) Two buttons (one for true and one for false), use your button label text string resources as their text (put these buttons in their own LinearLayout)

4. Add ids to your buttons within the layout

5. Hook up your buttons using the findViewById method:

```
public class MainActivity extends AppCompatActivity {
    private Button mTrueButton;
    private Button mFalseButton;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);

      mTrueButton = findViewById(R.id.true_button);
      mFalseButton = findViewById(R.id.false_button);
    }
}
```

6. Set an event listener for your true button:

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    mTrueButton = (Button) findViewById(R.id.true_button);
    mFalseButton = (Button) findViewById(R.id.false_button);

    mTrueButton.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View v) {
      }
    });
  }
```

7. Set an event listener for your false button:

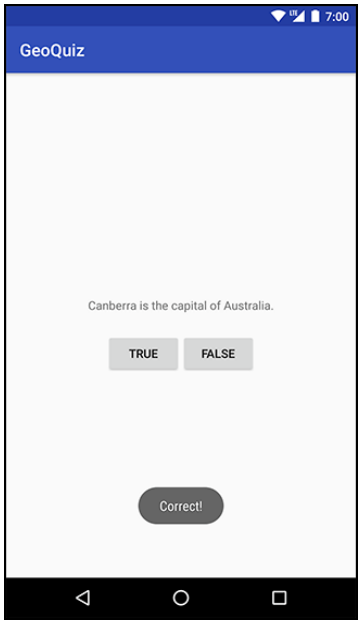
```
protected void onCreate(Bundle savedInstanceState) {
  …

  mTrueButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
    }
  });

  mFalseButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
     }
  });
}
```

8. Add strings to your strings file for Toast messages (add a string for if they get the answer correct and add a string for if they get the answer wrong).

9. When the user presses the buttons, make a Toast message that will tell the user if they are correct or not:

```
protected void onCreate(Bundle savedInstanceState) {
  …
 mTrueButton.setOnClickListener(new View.OnClickListener() {
     @Override
     public void onClick(View v) {
       Toast.makeText(MainActivity.this, R.string.correct_toast, Toast.LENGTH_SHORT).show();
     }
  });

 mFalseButton.setOnClickListener(new View.OnClickListener() {
   @Override
   public void onClick(View v) {
     Toast.makeText(MainActivity.this, R.string.incorrect_toast, Toast.LENGTH_SHORT) .show();
   }
 });
}
```

10. Run your app using the emulator and ensure it works properly



11. Add a new class called Question to your project. Add a new Java class to your project, make sure you create a new Java class and name it Question.

12. This class will have two class variables, one will hold the question string resource id and the other will hold the answer. We will also include a constructor that will take an int and boolean parameter and set the values of your class variables:

```
public class Question {
   int textResId;
   boolean answer;

   Question(int textResId, boolean answer) {
      this.textResId = textResId;
      this.answer = answer;
   }
}
```

13. Next, generate getters and setters in your Question class (right click on your class and go to Generate, then Getters and Setters).

14. In your layout file, add the id "question_text_view" to your TextView and add a new button to your layout that says "NEXT" (you will need to add a string resource for the next button text).

15. Edit your string resource file and add 6 different question strings:

```
<string name="question_australia">Canberra is the capital of Australia.</string>
<string name="question_oceans">The Pacific Ocean is larger than
 the Atlantic Ocean.</string>
<string name="question_mideast">The Suez Canal connects the Red Sea
 and the Indian Ocean.</string>
<string name="question_africa">The source of the Nile River is in Egypt.</string>
<string name="question_americas">The Amazon River is the longest river
 in the Americas.</string>
<string name="question_asia">Lake Baikal is the world\'s oldest and deepest
 freshwater lake.</string>
```

16. Add a class variable in your MainActivity class that will hold an array of questions. Also, add a class variable in your MainActivity class that will hold the current question's array index:

```
public class MainActivity extends AppCompatActivity {
   private Button mTrueButton;
   private Button mFalseButton;
   private Question[] mQuestionBank = {
        new Question(R.string.question_australia, true),
        new Question(R.string.question_oceans, true),
        new Question(R.string.question_mideast, false),
        new Question(R.string.question_africa, false),
        new Question(R.string.question_americas, true),
        new Question(R.string.question_asia, true)
   };
   private int mCurrentIndex = 0;
   …
}
```

17. Add a class variable in your MainActivity class that will hold your TextView and next Button and in your OnCreate method, hook these variable up (e.g. using findViewByID). Also, add the code needed to set the TextView to show the first question:

```
public class MainActivity extends AppCompatActivity {
   private Button mTrueButton;
   private Button mFalseButton;
   private Button mNextButton;
   private TextView mQuestionTextView;

   …

   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);

      mTrueButton = findViewById(R.id.true_button);
      mFalseButton = findViewById(R.id.false_button);
      mNextButton = findViewById(R.id.next_button);
      mQuestionTextView = findViewById(R.id.question_text_view);

      mTrueButton.setOnClickListener(new View.OnClickListener() {
         @Override
         public void onClick(View v) {
               …
         }
      });

      mFalseButton.setOnClickListener(new View.OnClickListener() {
         @Override
         public void onClick(View v) {
            …
         }
      });

      int questionTextResId= mQuestionBank[mCurrentIndex].getTextResId();
      mQuestionTextView.setText(questionTextResId);
   }
```

18. Run GeoQuiz, you should see the first question in the array appear in the TextView.

19. Now let's make the NEXT button functional. Set a View.OnClickListener on it. This listener will increment the index and update the TextView's text.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    …

    mNextButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            mCurrentIndex = (mCurrentIndex + 1) % mQuestionBank.length;
            int questionTextResId = mQuestionBank[mCurrentIndex].getTextResId();
            mQuestionTextView.setText(questionTextResId);
        }
    });

    int questionTextResId = mQuestionBank[mCurrentIndex].getTextResId();
    mQuestionTextView.setText(questionTextResId);
}
```

20. You now have the same code in two separate places that updates the text displayed in questionTextView. Take a moment to put this code into a function instead:

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    …
    mNextButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            mCurrentIndex = (mCurrentIndex + 1) % mQuestionBank.length;
            int questionTextResId = mQuestionBank[mCurrentIndex].getTextResId();
            mQuestionTextView.setText(questionTextResId);
            updateQuestion();
        }
    });

    int questionTextResId = mQuestionBank[mCurrentIndex].getTextResId();
    mQuestionTextView.setText(questionTextResId);
    updateQuestion();
}

private void updateQuestion() {
    int questionTextResId = mQuestionBank[mCurrentIndex].getTextResId();
    mQuestionTextView.setText(questionTextResId);
}
}
```

21. Run GeoQuiz and test your new NEXT button.

22. Write a method called checkAnswer that will accept a boolean (this will be the answer the user pressed, e.g. true of false). In this method write code that will check the answer with the current questions answer and show a Toast message if they were correct or not:

```
…
   private void updateQuestion() {
      int questionTextResId = mQuestionBank[mCurrentIndex].getTextResId();
      mQuestionTextView.setText(questionTextResId);
   }

   private void checkAnswer(boolean answer) {
      boolean correctAnswer = mQuestionBank[mCurrentIndex].isAnswer();

      if (answer == correctAnswer) {
         Toast.makeText(this, R.string.correct_toast, Toast.LENGTH_SHORT).show();
      } else {
         Toast.makeText(this, R.string.incorrect_toast, Toast.LENGTH_SHORT).show();
      }
   }
}
```

23. Update your true and false button click listener's on click methods to use the checkAnswer function instead of the code from earlier:

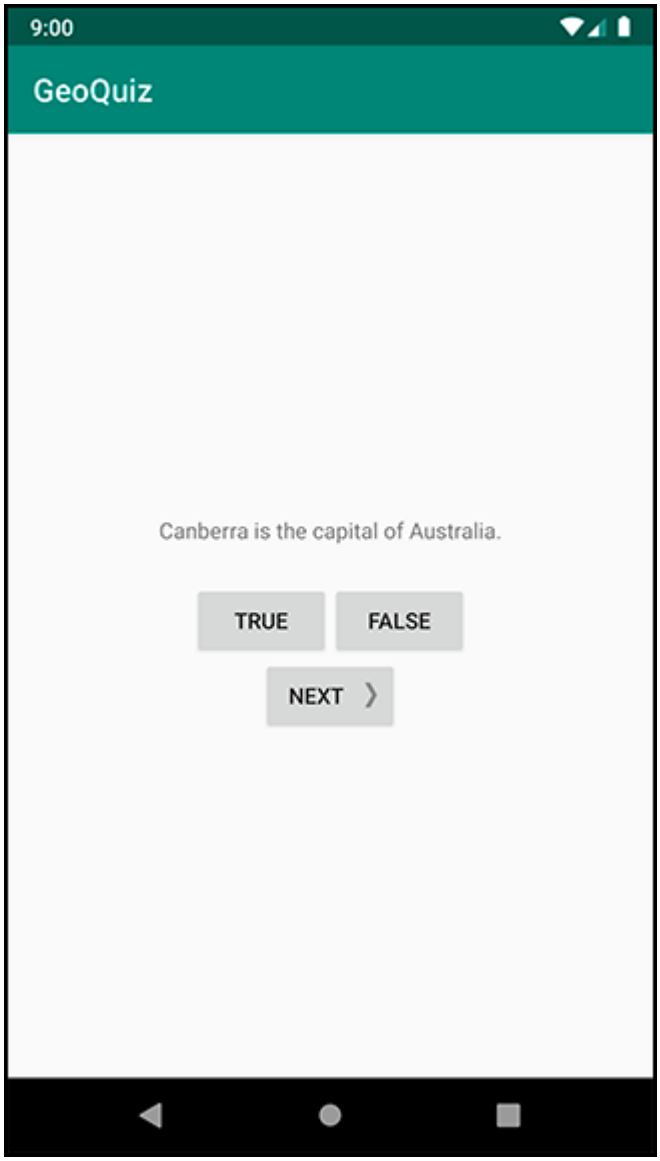
```
mTrueButton.setOnClickListener(new View.OnClickListener() {
  @Override
  public void onClick(View v) {
     Toast.makeText(MainActivity.this, R.string.correct_toast, Toast.LENGTH_SHORT).show();
     checkAnswer(true);
  }
});

mFalseButton.setOnClickListener(new View.OnClickListener() {
   @Override
   public void onClick(View v) {
     Toast.makeText(MainActivity.this, R.string.incorrect_toast, Toast.LENGTH_SHORT) .show();
     checkAnswer(false);
   }
 });
```

24. Run GeoQuiz. Verify that the toasts display the right message based on the answer to the current question and the button you press.

25. To make your app look better, put an image of an arrow on your next button. First download the images from the LMS, then add them to your resources (you have to unzip them to a folder, then copy them, and go into Android Studio, right click the “res” folder, and selected paste). Next set the drawableEnd attribute of your button in your layout file.

26. Run GeoQuiz and admire your button’s new appearance. Then test it to make sure it still works as before.

Final Part A App



Part B

1. We will be continuing to work on our GeoQuiz app, therefore open it in Android Studio.

2. Let's setup a variable we can use as the tag when logging messages to the Logcat. Open your MainActivity.java file and add a TAG variable:

```
import …

public class MainActivity extends AppCompatActivity {
   private final String TAG = "MainActivity";
   …
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      …
   }
   …
}
```

3. Add a log message in the onCreate() method so we can see when that method triggers:

```
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      Log.d(TAG, "onCreate(Bundle) called");
      setContentView(R.layout.activity_main);

      …
   }
```

4. Override the activity lifecycle methods (onStart, onResume, onPause, onStop, and onDestroy), ensure you call the super's method for each method you override:

```
public class MainActivity extends AppCompatActivity {
   …
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      …
   }

   @Override
   public void onStart() {
      super.onStart();
      Log.e(TAG, "onStart called.");
   }

   @Override
   public void onResume() {
      super.onResume();
      Log.d(TAG, "onResume() called.");
   }

   @Override
   public void onPause() {
      super.onPause();
      Log.d(TAG, "onPause() called.");
   }

   @Override
   public void onStop() {
      super.onStop();
      Log.d(TAG, "onStop() called.");
   }

   @Override
   public void onDestroy() {
      super.onDestroy();
      Log.d(TAG, "onDestroy() called.");
   }
}
```

5. Run GeoQuiz and view the messages in the Logcat.

6. Create a filter in LogCat to show only the log items for your TAG.

7. Run your app and open the LogCat, rotate the device in the emulator and use the back button and home buttons, observe when the lifecycle events executed by reviewing your messages in the LogCat.

8. Create a landscape layout for your app by right clicking the res folder and creating a new resource (ensure you select layout as the resource type, and Orientation and Landscape in the options).

9. Use a FrameLayout to build a different layout for your landscape layout.

10. Run your app and rotate the device to see your new layout.

Final Part B App