Maydm

The Quizzler: Updating View & Controller Layers

Maydm

Update the View Layer

We need to update the view layer since we're adding multiple questions. Remember, the view layer is everything the user sees on their screen.

We will add a "Next" button to enable users to cycle through questions. Here are the steps we'll take to update the view layer:

- Remove android:text attribute from the TextView in activity_main.xml
- Create an android:id attribute in TextView in activity_main.xml
    - Give widget a resource ID so text can be established in MainActivity
- Add a new Button widget as a child of the root ConstraintLayout.

Maydm

Edit activity_main.xml: Edit the TextView Widget

Remove the android:text attribute and edit the android:id attribute.

The android:text value may show @string/question_text OR the actual question inside of quotation marks.

```
<TextView
   android:id="@+id/question_text_view"
   ….
```

Maydm

Edit activity_main.xml: Add a New Button Widget

Add a new button widget with an android:id attribute called next_button then add constraints using the preview screen.

```
<Button
    android:id="@+id/next_button"
    android:text="@string/next_button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />
```

Maydm

Edit the String Resources in strings.xml

Click to strings.xml and rename `` `question_text` `` to `` `question_madison` `` and add a string for the new Next Button Widget.

```
<resources>
   <string name="app_name">TheQuizzler</string>
   <string name="question_madison">Madison is the capital of Wisconsin.</string>
   <string name="true_button">True</string>
   <string name="false_button">False</string>
   <string name="next_button">Next</string>
   <string name="correct_toast">Correct!</string>
   <string name="incorrect_toast">Incorrect</string>
</resources>
```

Maydm

Let's add additional questions in strings.xml

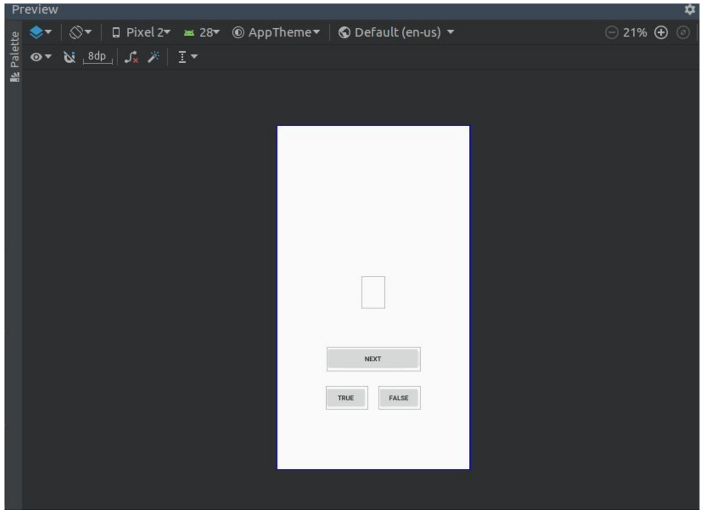
Click to strings.xml and rename `question_text` to `question_madison` and add a string for the new Next Button Widget.

```
<resources>
  <string name="app_name">TheQuizzler</string>
  <string name="question_madison">Madison is the capital of Wisconsin.</string>
  <string name="question_milwaukee">Milwaukee is smaller in population than Madison.</string>
  <string name="question_chicago">Chicago is larger than Milwaukee and Madison combined.</string>
  <string name="question_bird">The robin is the state bird of Wisconsin</string>
  <string name="question_tree">The birch is the state tree of Wisconsin</string>
  <string name="question_midwest">Wisconsin is one of seven states classified as being in the Midwest.</string>
  ….
```

Maydm

Preview the App's Layout in activity_main.xml

Inspect the graphical layout preview tool to be sure your widgets are placed how you like them.



Maydm

Update the Controller Layer

We will need to update the controller to facilitate communication between the view and model layers as The Quizzler becomes more complex.

To accomplish this, we're going to add two variables for TextView and the new Button widget. We'll also add an array of Question objects and an array index.

Maydm

Edit MainActivity.java; add array and variables

```
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

  private Button mTrueButton;
  private Button mFalseButton;
  private Button mNextButton;
  private TextView mQuestionTextView;

  private Question[] mQuestionBank = new Question[] {
        new Question(R.string.question_madison, true),
        new Question(R.string.question_milwaukee, false),
        new Question(R.string.question_chicago, true),
        new Question(R.string.question_bird, true),
        new Question(R.string.question_tree, false),
        new Question(R.string.question_midwest, false),
  };

  private int mCurrentIndex = 0;
```

Be sure to add the TextView widget. (Android Studio will prompt you to do this after creating the variable for TextView.

Private variables so they can only be accessed with getters and setters

New Question array which constructs new questions for the Questions Bank. (For now, we'll store the model in the controller, but later we'll learn a better way to store the model layer).

Which variables do you think we'll use to display questions in the view?

Maydm

Update TextView in MainActivity.java

We'll apply a reference for TextView and tell Android to show the text to the question at the current index.

We're going to add 3 lines of code after

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

Update TextView in MainActivity.java

```
setContentView(R.layout.activity_main);

mQuestionTextView = (TextView) findViewById(R.id.question_text_view);
int question = mQuestionBank[mCurrentIndex].getTextResId();
mQuestionTextView.setText(question);

mTrueButton = (Button) findViewById(R.id.true_button);
```

New snippet here

Maydm

Open the Emulator

Save the changes and now we can run the emulator to see the question!
Note that the questions will not appear in activity_main.xml.

Remember, to run the emulator select Run -> Run 'app' in the menu.

If you receive permission denied errors then try this for linux machines:

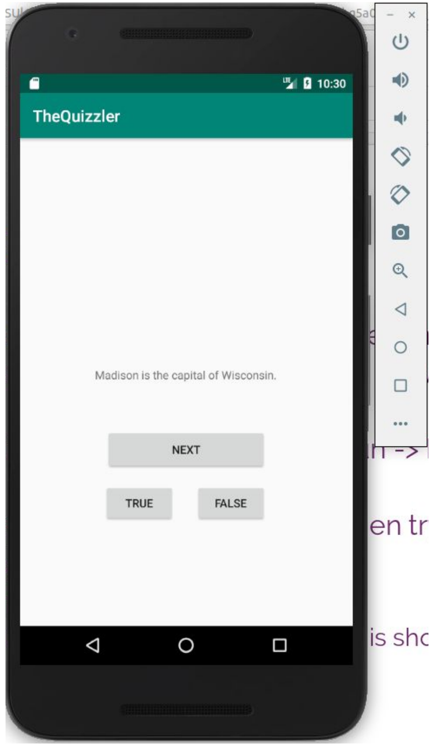
```
sudo apt install qemu-kvm
sudo adduser $USER kvm
```

($USER is the username used for your login, this should be in the terminal prompt)

Open the Emulator

If you see the first question in your emulator then congratulations! Next we'll wire the NEXT button to display the next question in the Question array.

Remember, to save time, leave the emulator running in the background while we make changes to the code.

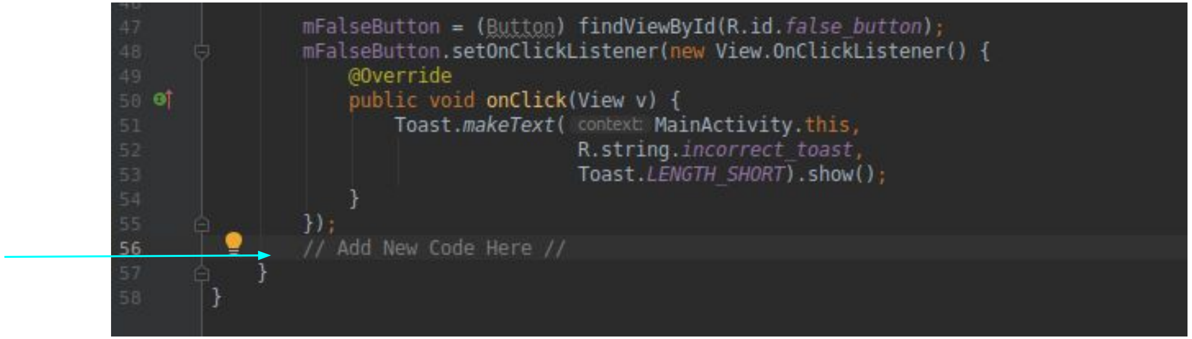


Madison is the capital of Wisconsin.

Wire the Next Button in MainActivity.java

To make the Next Button behave as we wish we'll first need to give Android a reference to the button and then set an event listener to wait for a user action. Our listener is going to increment through the question array to update the View Layer with a new question.

We'll add this snippet just before the two closing "}" curly brackets.



```
        mFalseButton = (Button) findViewById(R.id.false_button);
        mFalseButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Toast.makeText( context: MainActivity.this,
                        R.string.incorrect_toast,
                        Toast.LENGTH_SHORT).show();
            }
        });
        // Add New Code Here //
    }
}
```

Maydm

Create a New Function So You DRY

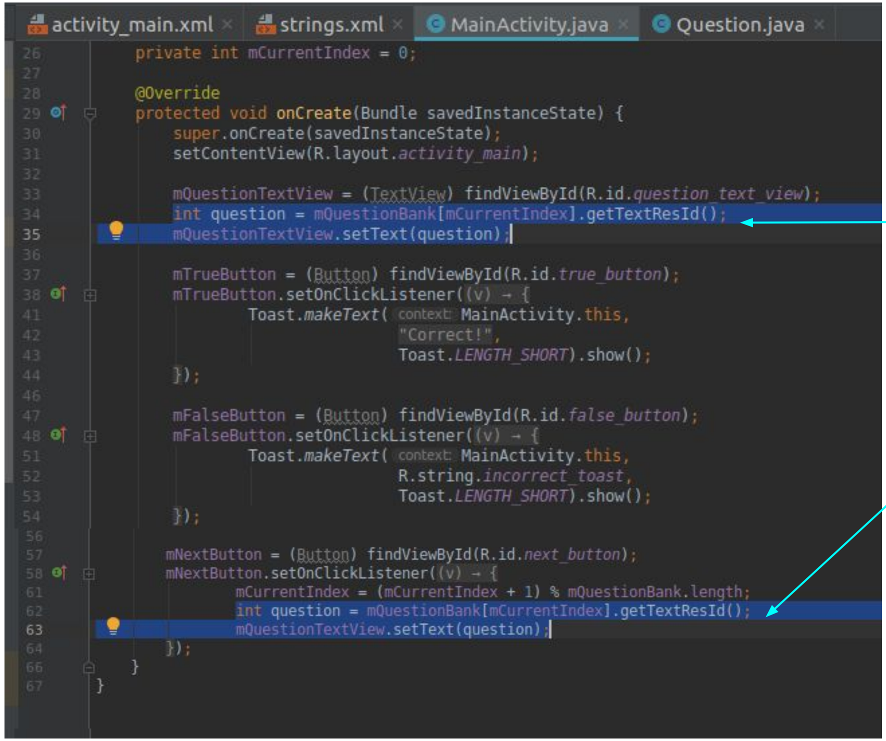
DRY is a concept in coding that helps us avoid becoming saturated in unnecessary code. DRY stands for: Don’t Repeat Yourself

When you see code repeating itself that’s likely a perfect opportunity to create a function or a loop.

In this case, we have code that updates `mQuestionTextView` in two different places. We’ll create a private function and call the new method in the `mNextButton` listener.

Maydm

Create a New Function So You DRY

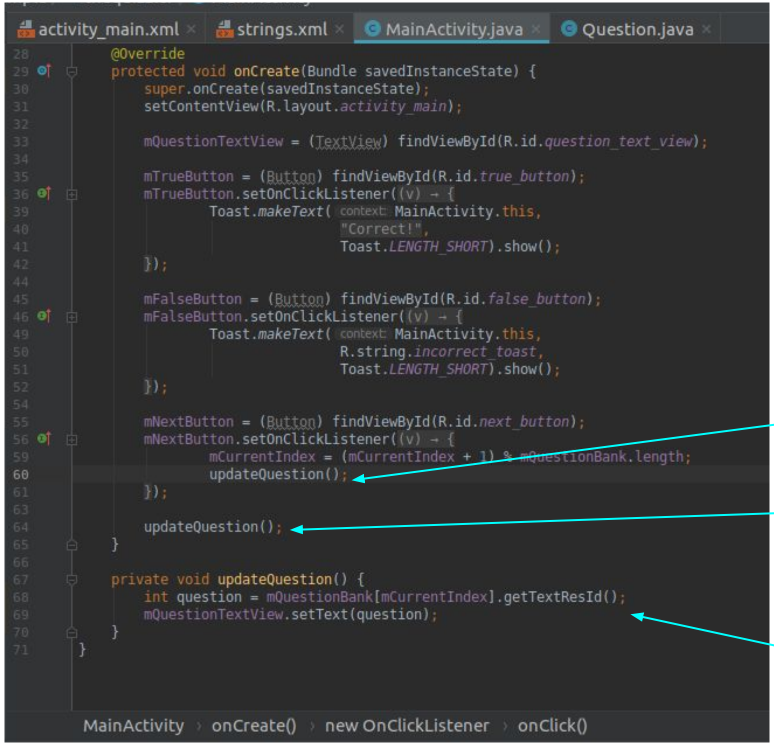


```
    private int mCurrentIndex = 0;

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

        mQuestionTextView = (TextView) findViewById(R.id.question_text_view);
        int question = mQuestionBank[mCurrentIndex].getTextResId();
        mQuestionTextView.setText(question);

        mTrueButton = (Button) findViewById(R.id.true_button);
        mTrueButton.setOnClickListener((v) → {
                Toast.makeText( context: MainActivity.this,
                        "Correct!",
                        Toast.LENGTH_SHORT).show();
        });

        mFalseButton = (Button) findViewById(R.id.false_button);
        mFalseButton.setOnClickListener((v) → {
                Toast.makeText( context: MainActivity.this,
                        R.string.incorrect_toast,
                        Toast.LENGTH_SHORT).show();
        });

        mNextButton = (Button) findViewById(R.id.next_button);
        mNextButton.setOnClickListener((v) → {
            mCurrentIndex = (mCurrentIndex + 1) % mQuestionBank.length;
            int question = mQuestionBank[mCurrentIndex].getTextResId();
            mQuestionTextView.setText(question);
        });
    }
}
```

Copy and remove the repeated code

Maydm

Create a New Function So You DRY



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

        mQuestionTextView = (TextView) findViewById(R.id.question_text_view);

        mTrueButton = (Button) findViewById(R.id.true_button);
        mTrueButton.setOnClickListener((v) → {
                Toast.makeText( context: MainActivity.this,
                        "Correct!",
                        Toast.LENGTH_SHORT).show();
        });

        mFalseButton = (Button) findViewById(R.id.false_button);
        mFalseButton.setOnClickListener((v) → {
                Toast.makeText( context: MainActivity.this,
                        R.string.incorrect_toast,
                        Toast.LENGTH_SHORT).show();
        });

        mNextButton = (Button) findViewById(R.id.next_button);
        mNextButton.setOnClickListener((v) → {
                mCurrentIndex = (mCurrentIndex + 1) % mQuestionBank.length;
                updateQuestion();
        });

        updateQuestion();
    }

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

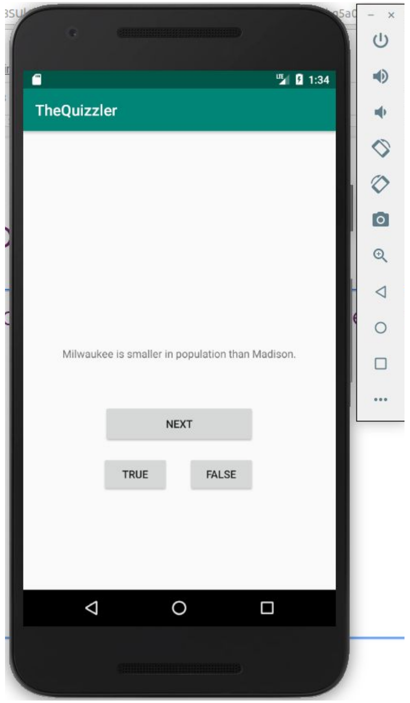
Call the new updateQuestion() method in the nextButton event listener and at the end of OnCreate()

Create a new private function called updateQuestion after the onCreate function.

Maydm

Test the App

Save your work and run the emulator to be sure your code works as expected.



Milwaukee is smaller in population than Madison.

Notice Something Odd?

Did you notice that Android thinks that every question should be TRUE to be correct?

We know that's not correct. (More proof that computers are not as smart as many people think).

Let's help this program out so it show the correct answer responses.

Code for private void checkAnswer

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

    private void checkAnswer(boolean userPressedTrue) {
        boolean answerIsTrue = mQuestionBank[mCurrentIndex].isAnswerTrue();

        int messageResId = 0;

        if (userPressedTrue == answerIsTrue) {
            messageResId = R.string.correct_toast;
        } else {
            messageResId = R.string.incorrect_toast;
        }

        Toast.makeText(this, messageResId, Toast.LENGTH_SHORT).show();
    }
}
```

Accepts a boolean to ID if the user pressed True or False.

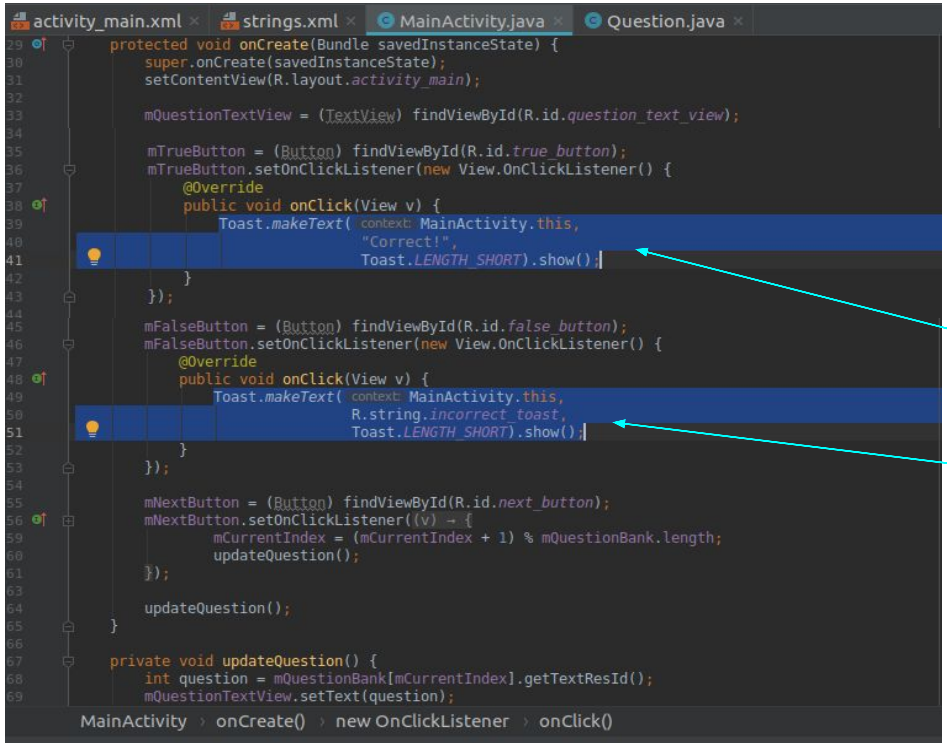
Checks if the answer is saved as being true or false

Determines if user is correct or not and stores the correct or incorrect toast

Instructs the Toast to display on the View Layer

Maydm

Call the checkAnswer() Method in the button listeners

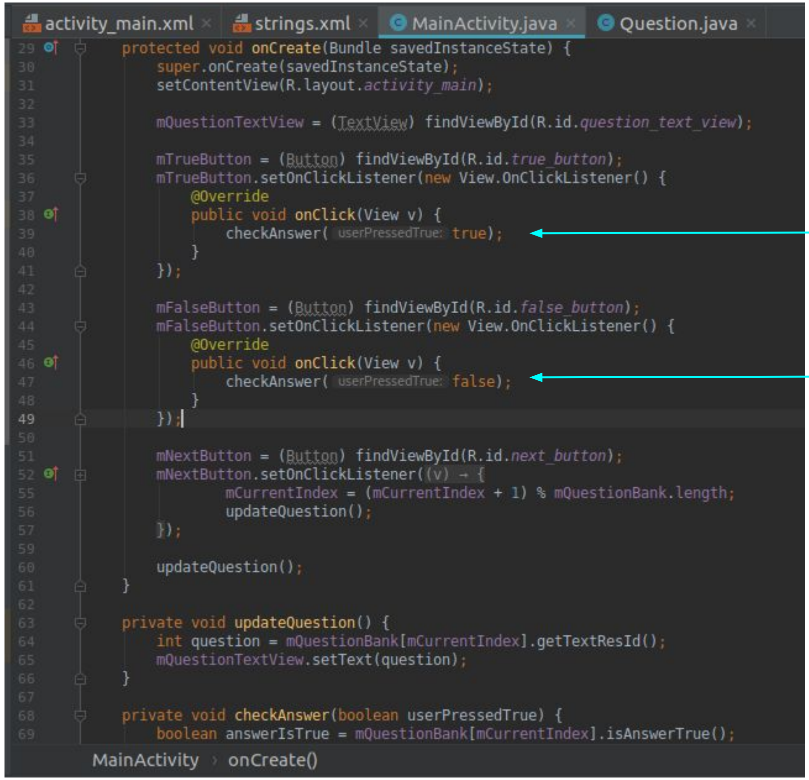


```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    mQuestionTextView = (TextView) findViewById(R.id.question_text_view);

    mTrueButton = (Button) findViewById(R.id.true_button);
    mTrueButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Toast.makeText( context: MainActivity.this,
                    "Correct!",
                    Toast.LENGTH_SHORT).show();
        }
    });

    mFalseButton = (Button) findViewById(R.id.false_button);
    mFalseButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Toast.makeText( context: MainActivity.this,
                    R.string.incorrect_toast,
                    Toast.LENGTH_SHORT).show();
        }
    });

    mNextButton = (Button) findViewById(R.id.next_button);
    mNextButton.setOnClickListener((v) → {
            mCurrentIndex = (mCurrentIndex + 1) % mQuestionBank.length;
            updateQuestion();
    });

    updateQuestion();
}

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

This code violates DRY and is what we created in the new checkAnswer() function. Remove this and call the checkAnswer(true) and checkAnswer(false) methods, respectively.

Maydm

MainActivity with checkAnswer() Methods



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

        mQuestionTextView = (TextView) findViewById(R.id.question_text_view);

        mTrueButton = (Button) findViewById(R.id.true_button);
        mTrueButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                checkAnswer( userPressedTrue: true);
            }
        });

        mFalseButton = (Button) findViewById(R.id.false_button);
        mFalseButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                checkAnswer( userPressedTrue: false);
            }
        });

        mNextButton = (Button) findViewById(R.id.next_button);
        mNextButton.setOnClickListener((v) → {
                mCurrentIndex = (mCurrentIndex + 1) % mQuestionBank.length;
                updateQuestion();
        });

        updateQuestion();
    }

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

    private void checkAnswer(boolean userPressedTrue) {
        boolean answerIsTrue = mQuestionBank[mCurrentIndex].isAnswerTrue();
```

Great work!

Run the app in the emulator.

Next we'll run the app on a real device! If you don't have an Android phone then pair up with a friend that does.

Maydm