Github

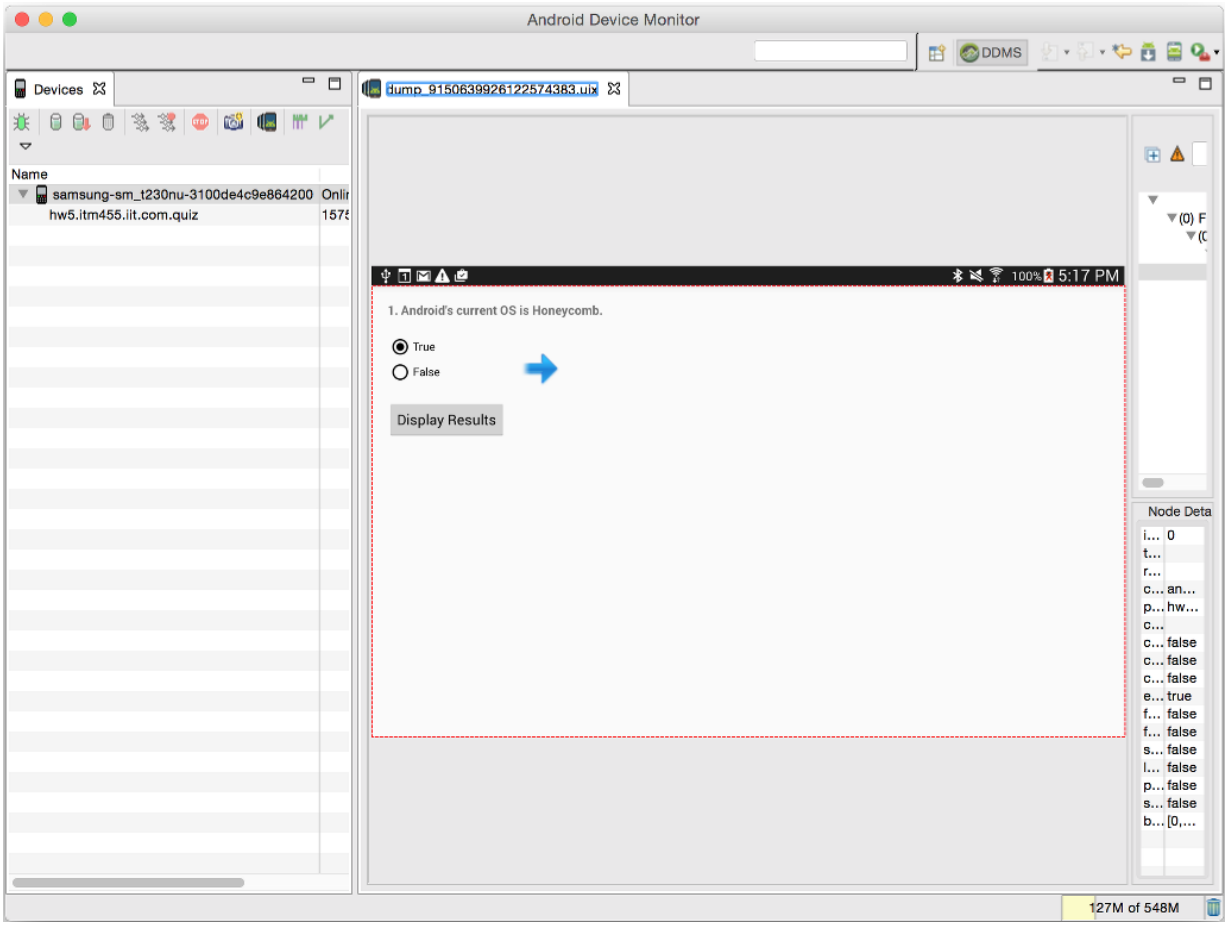
The source code for this assignment, which discussed in this document, is published online @ https://github.com/bradyhouse/ITM455/tree/master/HW5.

Screenshots

The application was tested using a Galaxy Tab 4 tablet connected via USB. The Galaxy Tab 4 tablet, comes loaded with Android 4.4.2. Each screenshot was then taken using Android Device Monitor.

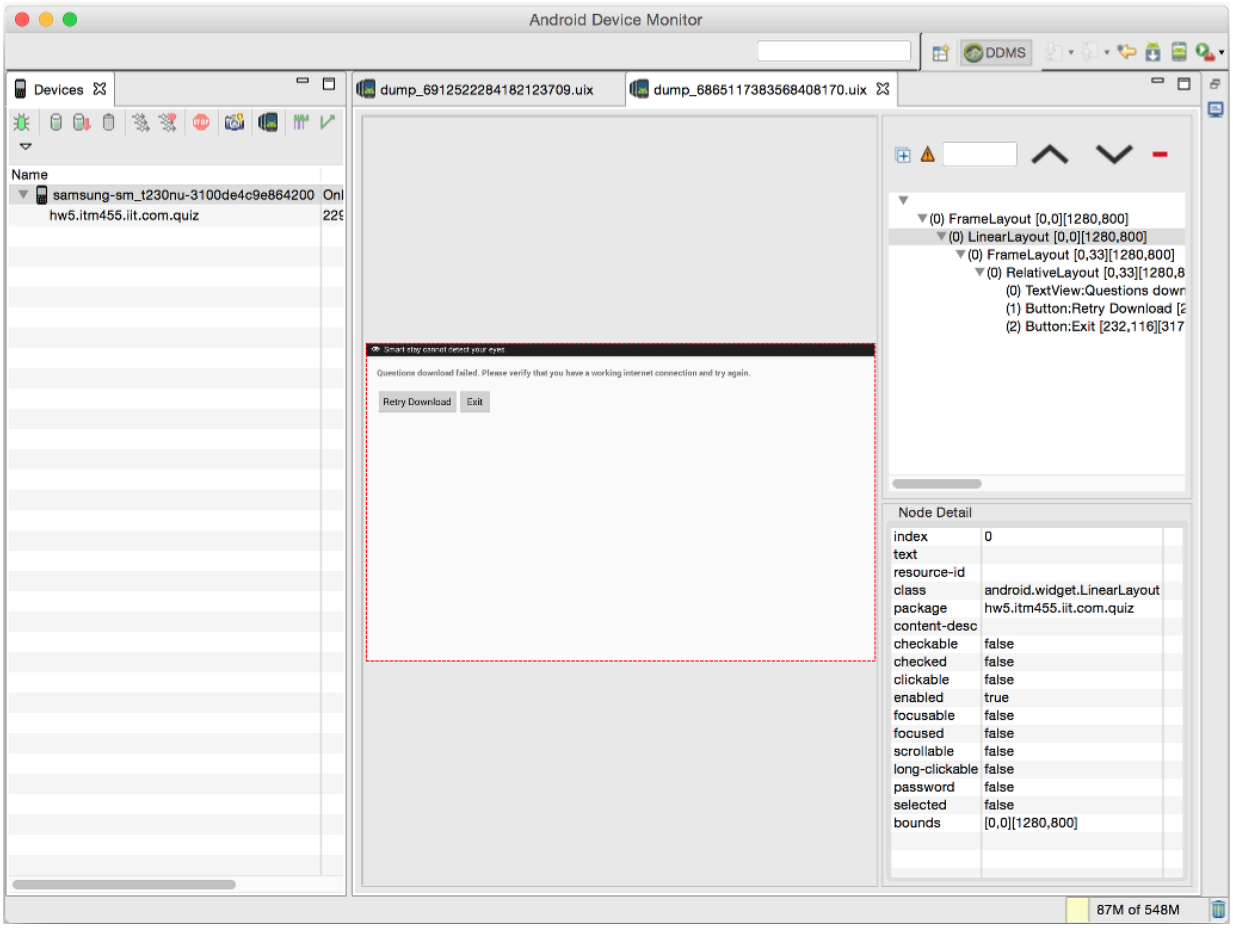
Startup (with Internet)

When the application is started on a device with a valid internet connection, it will download the question list, populate the application store and then display the first question.



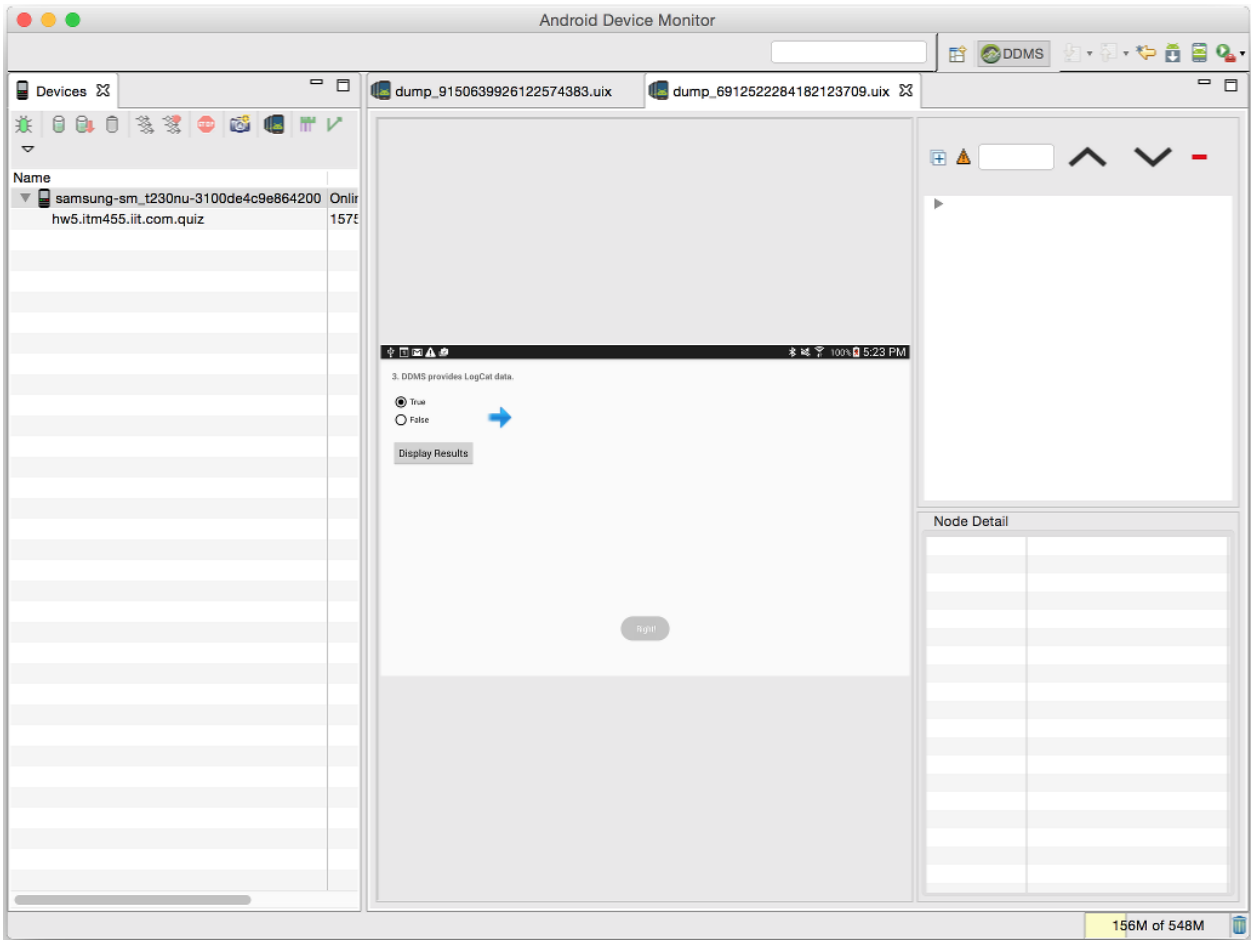
Startup (without Internet)

If the application started on a device without an internet connection, on startup will display an error message along with a "retry download" and an "exit" button. The "retry download" button can be used to initiate the download of the question list. The "exit" button can be used to simply hide the application.



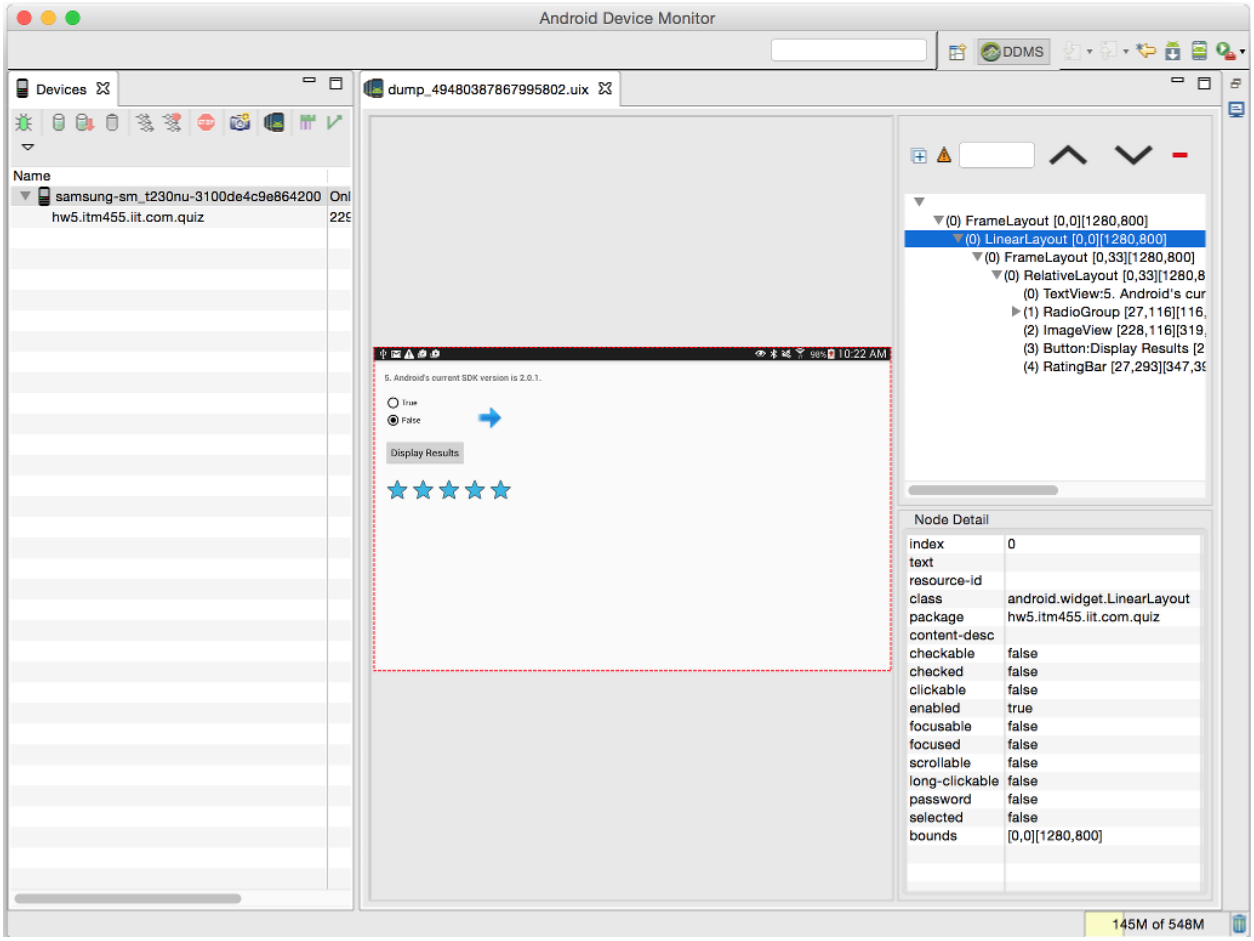
Question 3 (Toast Result)

When the user clicks the "display results" button, toast is used to display a pop-up message indicating whether the user's input is correct.



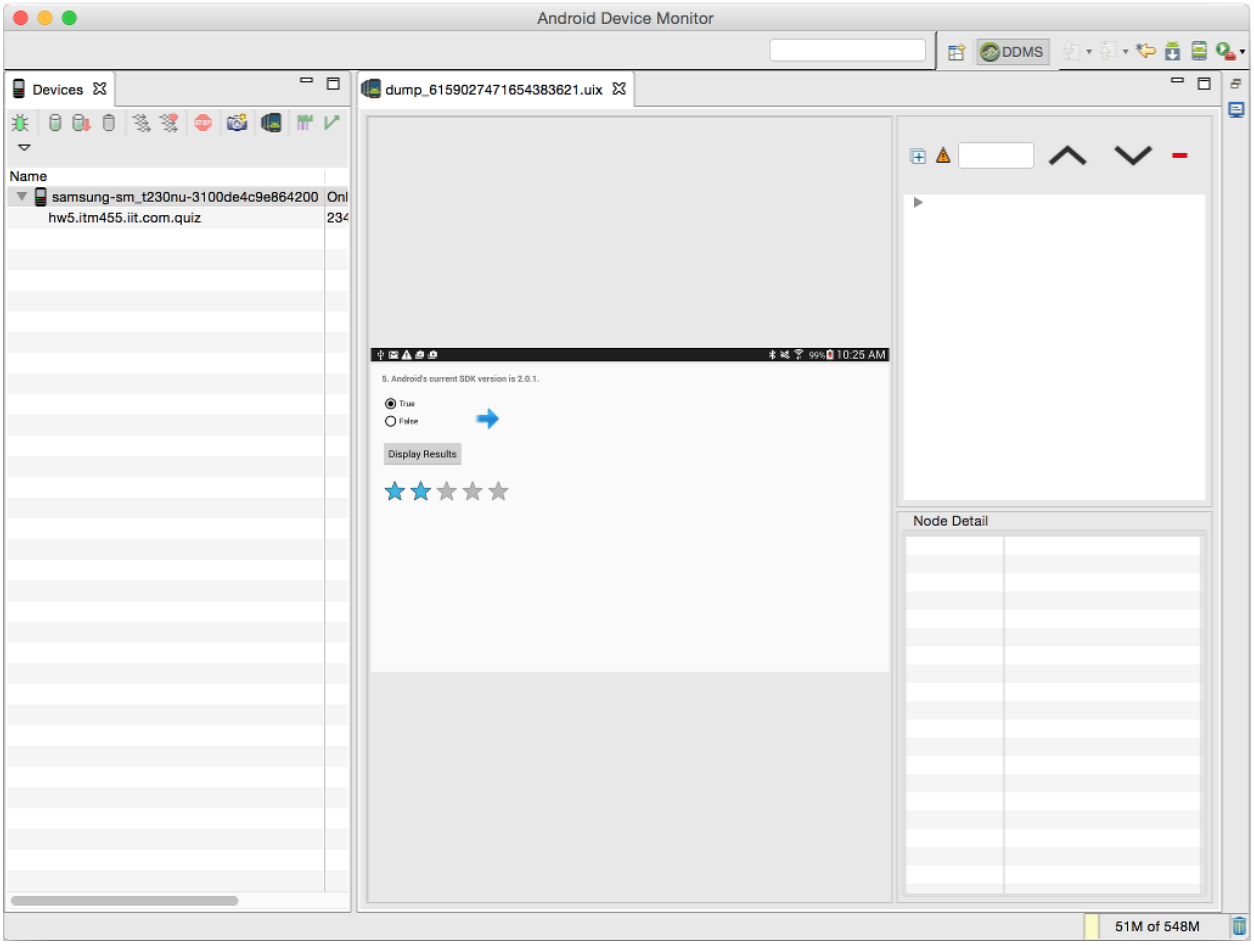
Ratings Bar (All Correct)

On the fifth question, when the user clicks the "display results" button, the rating bar becomes visible. If the user has entered a correct response for all 5 questions, all 5 stars are highlighted.



Rating Bar (2 Correct)

On the fifth question, when the user clicks the "display results" button, the rating bar becomes visible. If the user has entered only 2 correct responses then only 2 stars will be highlighted.



Source Code

The application code breaks down into the following classes and packages.

Package	Class	Type (Extends)	Description
(root)	MainActivity	Activity	Application entry point and main consumer. All other classes are consumed by this class.
model	Question	Java Class	Model class corresponding to a single question. Includes properties for capturing and users "actual" answer.
store	Questions	Java Class	Data store class. It is a wrapper class for an array list of Question model instances. Additionally, it provides method for evaluating the questions in aggregate.
util	DownloadTask	AsyncTask Class	Class used to create an asynchronous event and secondary thread in order to safely, "attempt" a download of the question list.
util	IDownloadTaskComplete	Interface Class	Simple interface used to define a delegate method that can be passed to the DownloadTask constructor. The class calls the delegate once the download completes.

MainActivity.java

```
package hw5.itm455.iit.com.quiz;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.TextView;
import android.widget.Toast;
import android.widget.RatingBar;

import java.util.ArrayList;

import hw5.itm455.iit.com.quiz.model.Question;
import hw5.itm455.iit.com.quiz.store.Questions;
import hw5.itm455.iit.com.quiz.util.DownloadTask;
import hw5.itm455.iit.com.quiz.util.IDownloadTaskComplete;

public class MainActivity extends Activity {

    final String textSource = "http://www.papademas.net/sample.txt";
    static int questionNum = -1;


    private Questions questions;
    private Boolean actual;
    private TextView textView;
    private RadioGroup radioQuestions;
    private RadioButton radioButton;
    private Button btnDisplay;
    private Button btnRetry;
    private Button btnExit;
    private ImageView image;
    private RatingBar ratingBar;

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

    public void downloadListener() {
        textView = (TextView) findViewById(R.id.textMessage);
        btnRetry = (Button) findViewById(R.id.btnRetry);
        textView.setText(R.string.text_msg_downloading);
        btnRetry.setVisibility(View.INVISIBLE);
        DownloadTask downloadQuestions = new DownloadTask(new IDownloadTaskComplete() {
            @Override
            public void onDownloadComplete(ArrayList<Question> list) {
                questions = new Questions(list);
                startQuiz();
            }
        }, textSource);
        downloadQuestions.execute();
    }

    public void startQuiz() {
        stateListener();
        buttonListener();
        radioListener();
        imageListener();
    }

    public void stateListener() {
        textView = (TextView) findViewById(R.id.textMessage);
        radioQuestions = (RadioGroup) findViewById(R.id.radioQuestions);
        btnDisplay = (Button) findViewById(R.id.btnDisplay);
        image = (ImageView) findViewById(R.id.imageNext);
        btnRetry = (Button) findViewById(R.id.btnRetry);
        btnExit = (Button) findViewById(R.id.btnExit);
        Question question;
        if (!this.questions.isEmpty()) {
```

```
            question = questions.getList().get(++questionNum);
            textView.setText(Integer.toString(question.getId()) + ". " + question.getQuestion());
            btnRetry.setVisibility(View.INVISIBLE);
            btnExit.setVisibility(View.INVISIBLE);
            radioQuestions.setVisibility(View.VISIBLE);
            btnDisplay.setVisibility(View.VISIBLE);
            image.setVisibility(View.VISIBLE);
        } else {
            textView.setText(R.string.text_msg_download_err);
            btnRetry.setVisibility(View.VISIBLE);
            btnExit.setVisibility(View.VISIBLE);
        }
    }

    public void buttonListener() {
        if (!this.questions.isEmpty()) {
            radioQuestions = (RadioGroup) findViewById(R.id.radioQuestions);
            btnDisplay = (Button) findViewById(R.id.btnDisplay);
            btnDisplay.setOnClickListener(new OnClickListener() {
                @Override
                public void onClick(View v) {
                    int selectedId = radioQuestions.getCheckedRadioButtonId();
                    radioButton = (RadioButton) findViewById(selectedId);
                    Question question = questions.getList().get(questionNum);
                    if (question.getId() == 5) {
                        displayRating();
                    } else {
                        displayResult(questions.getList().get(questionNum).isCorrect());
                    }
                }
            });
        } else {
            btnRetry = (Button) findViewById(R.id.btnRetry);
            btnExit = (Button) findViewById(R.id.btnExit);
            btnRetry.setOnClickListener(new OnClickListener() {
                @Override
                public void onClick(View v) {
                    btnRetry.setVisibility(View.INVISIBLE);
                    btnExit.setVisibility(View.INVISIBLE);
                    downloadListener();
                }
            });
            btnExit.setOnClickListener(new OnClickListener() {
                @Override
                public void onClick(View v) {
                    Intent intent = new Intent(Intent.ACTION_MAIN);
                    intent.addCategory(Intent.CATEGORY_HOME);
                    intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
                    startActivity(intent);
                }
            });
        }
    }

    public void radioListener() {
        if (!questions.isEmpty()) {
            radioQuestions = (RadioGroup) findViewById(R.id.radioQuestions);
            radioQuestions.setOnCheckedChangeListener(new RadioGroup.OnCheckedChangeListener() {
                @Override
                public void onCheckedChanged(RadioGroup group, int checkedId) {
                    int selectedId = radioQuestions.getCheckedRadioButtonId();
                    radioButton = (RadioButton) findViewById(selectedId);

                    if (radioButton.getText().equals("True")) {
                        actual = true;
                    } else {
                        actual = false;
                    }
                    questions.getList().get(questionNum).setActual(actual);
                }
            });
        }
    }

    public void imageListener() {
        if (!questions.isEmpty()) {
            textView = (TextView) findViewById(R.id.textMessage);
            image = (ImageView) findViewById(R.id.imageNext);
            image.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View view) {
```

```
                    Question question = questions.getList().get(questionNum);
                    hideRating();
                    if (question.getId() == 5) {
                        questionNum = -1;
                    }
                    question = questions.getList().get(++questionNum);
                    textView.setText(Integer.toString(question.getId()) + ". " + question.getQuestion());
                    if (question.getActual()) {
                        radioQuestions.check(R.id.radioTrue);
                    } else {
                        radioQuestions.check(R.id.radioFalse);
                    }
                }
            });
        }
    }

    public void displayResult(boolean correct) {
        if (correct) {
            Toast.makeText(MainActivity.this,
                    " Right!", Toast.LENGTH_SHORT).show();
        } else {
            Toast.makeText(MainActivity.this,
                    " Wrong!", Toast.LENGTH_SHORT).show();
        }
    }

    public void displayRating() {
        ratingBar = (RatingBar) findViewById(R.id.ratingBar);
        ratingBar.setRating((float) questions.getTotalCorrect());
        ratingBar.setVisibility(RatingBar.VISIBLE);

    }

    public void hideRating() {
        ratingBar = (RatingBar) findViewById(R.id.ratingBar);
        ratingBar.setVisibility(RatingBar.INVISIBLE);
    }

}
```

Model > Question.java

```
package hw5.itm455.iit.com.quiz.model;


public class Question {

    private Integer id;
    private String question;
    private Boolean expected;
    private Boolean actual;

    public Question(Integer id, String question) {
        this.id = id;
        this.question = question;
        this.expected = this.hardCodeExpected(id);
        this.actual = true;
    }

    public Integer getId() {
        return id;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public String getQuestion() {
        return question;
    }

    public void setQuestion(String question) {
        this.question = question;
    }

    public Boolean getExpected() {
        return expected;
    }

    public void setExpected(Boolean expected) {
        this.expected = expected;
    }

    public Boolean getActual() {
        return actual;
    }

    public void setActual(Boolean actual) {
        this.actual = actual;
    }

    public Boolean isCorrect() {
        return this.expected == this.actual;
    }

    private Boolean hardCodeExpected(Integer id) {
        switch (id) {
            case 1:
                return true;
            case 2:
                return false;
            case 3:
                return true;
            case 4:
                return false;
            case 5:
                return false;
            default:
                return false;
        }
    }

}
```

Store > Questions.java

```
package hw5.itm455.iit.com.quiz.store;


import java.util.ArrayList;


import hw5.itm455.iit.com.quiz.model.Question;

public class Questions {

    private ArrayList<Question> _list = new ArrayList<Question>();

    public Questions(ArrayList<Question> questions) {
        this._list.addAll(questions);
    }

    public ArrayList<Question> getList() {
        return _list;
    }

    public Integer getTotalCorrect() {
        ArrayList<Question> correctAnswers = new ArrayList<Question>();
        for (Question q : this._list) {
            if (q.isCorrect()) {
                correctAnswers.add(q);
            }
        }
        return correctAnswers.size();
    }

    public boolean isEmpty() {
        return this._list.size() == 0;
    }

}
```

Util > DownloadTask.java

```
package hw5.itm455.iit.com.quiz.util;

import android.os.AsyncTask;
import android.util.Log;

import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.net.URL;
import java.util.ArrayList;

import hw5.itm455.iit.com.quiz.model.Question;

public class DownloadTask extends AsyncTask<String, String, String> {

    private String _url = "";
    private ArrayList<Question> _list = new ArrayList<>();
    private IDownloadTaskComplete _delegate = null;


    public DownloadTask(IDownloadTaskComplete callback, String url) {
        this._url = url;
        this._delegate = callback;
    }


    @Override
    protected String doInBackground(String... f_url) {

        URL _textUrl;
        Integer _questionId = 1;
        try {
            _textUrl = new URL(this._url);
            BufferedReader bufferReader = new BufferedReader(
                    new InputStreamReader(_textUrl.openStream()));
            String questionText;
            while ((questionText = bufferReader.readLine()) != null) {
                this._list.add(new Question(_questionId, questionText));
                ++_questionId;
            }
            bufferReader.close();

        } catch (Exception e) {
            Log.i("DownloadTask", e.getMessage());
        }
        return null;
    }

    @Override
    protected void onPostExecute(String file_url) {
        this._delegate.onDownloadComplete(this._list);
    }

}
```

Util > IDownloadTaskComplete.java

```
package hw5.itm455.iit.com.quiz.util;

import java.util.ArrayList;

import hw5.itm455.iit.com.quiz.model.Question;

public interface IDownloadTaskComplete {
    void onDownloadComplete(ArrayList<Question> list);
}
```

Layout

The application includes one layout xml file, which includes the definition/configuration of all input visible on screen. At startup, the majority of these controls are invisible. Depending on whether the questions list can be downloaded, the controls then become visible.

activity_quiz.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    android:padding="20sp">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="16sp"
        android:id="@+id/textMessage"
        android:textStyle="bold"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:text="@string/text_msg_downloading"
        android:paddingBottom="20sp" />

    <RadioGroup
        android:id="@+id/radioQuestions"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/textMessage"
        android:visibility="invisible"
        android:layout_alignLeft="@+id/textMessage"
        android:paddingBottom="20sp">

        <RadioButton
            android:id="@+id/radioTrue"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/radio_true"
            android:checked="true" />

        <RadioButton
            android:id="@+id/radioFalse"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/radio_false" />
    </RadioGroup>

    <Button
        android:id="@+id/btnDisplay"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/btn_display"
        android:visibility="invisible"
        android:layout_below="@id/radioQuestions"
        android:layout_alignLeft="@id/textMessage" />

    <ImageView
        android:id="@+id/imageNext"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@drawable/next"
        android:clickable="true"
        android:paddingTop="20sp"
        android:paddingLeft="20sp"
        android:visibility="invisible"
```

```
        android:layout_toRightOf="@id/btnDisplay"
        android:layout_below="@id/textMessage" />

    <Button
        android:id="@+id/btnRetry"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/btn_retry"
        android:visibility="invisible"
        android:layout_below="@id/textMessage"
        android:layout_alignLeft="@id/textMessage" />

    <Button
        android:id="@+id/btnExit"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_toEndOf="@+id/btnRetry"
        android:text="@string/btn_exit"
        android:visibility="invisible"
        android:layout_below="@id/textMessage"
        android:layout_toRightOf="@id/btnRetry"></Button>


    <RatingBar
        android:id="@+id/ratingBar"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:numStars="5"
        android:stepSize="1.0"
        android:rating="2.0"
        android:isIndicator="true"
        android:visibility="invisible"
        android:paddingTop="20sp"
        android:layout_below="@id/btnDisplay"
        android:layout_alignLeft="@id/btnDisplay" />
</RelativeLayout>
```

Manifest

In order for the application to download the questions list, internet permissions must be granted for the application. This done via the manifest.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="hw5.itm455.iit.com.quiz">

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-sdk
        android:targetSdkVersion="19"
        android:minSdkVersion="8" />
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme">
        <activity
            android:name=".MainActivity"
            android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```