Uddhav Prasad Gautam <upgautam@ualr.edu> Recruiter: Arnie D. Singh

Rock - Paper - Scissors Review

Enhancements Approach

1) Created Android Studio project and detail Code Review using Android Studio
2) Created AndroidManifest.xml. Added below two lines:

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />.
```

Added <activity> section in the <application> as below.

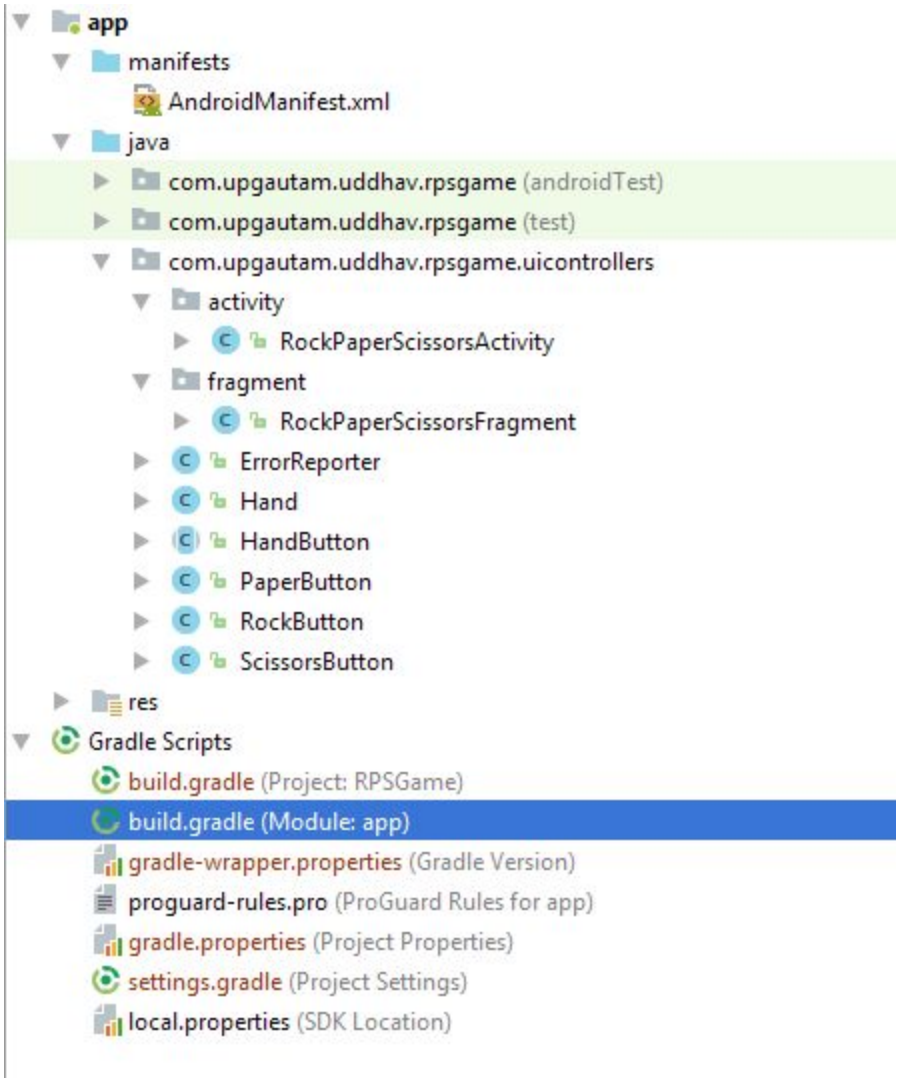
```
<activity
   android:name=".uicontrollers.activity.RockPaperScissorsActivity"
   android:label="@string/title_activity_navigation_drawer"
   android:theme="@style/AppTheme">
   <intent-filter>
       <action android:name="android.intent.action.MAIN" />

       <category android:name="android.intent.category.LAUNCHER" />
   </intent-filter>
</activity>
```

3) In project level build.gradle, added Retrofit library as below. Retrofit is way better than HttpUrlConnection.

```
//retrofit
compile 'com.squareup.retrofit2:retrofit:2.3.0'
```

4) Structured the Project Directory structure as below:



5) I didn't modify `RockPaperScissorsActivity` activity because we are directly putting `RockPaperScissorsFragment` in Activity's Layout.

6) In fragment `RockPaperScissorsFragment` , I created my own `getOpponentHand()` method as below.

```
        // avoid creating several instances, should be singleton OkHttpClient
        OkHttpClient client = new OkHttpClient();


        public void getOpponentHand() {

    //default method is GET
    Request request = new Request.Builder()
            .url("http://example.com/randhand")
            .build();

    client.newCall(request).enqueue(new Callback() {
```

```
        @Override
        public void onFailure(Call call, IOException e) {
            e.printStackTrace();
        }

        @Override
        public void onResponse(Call call, final Response response) throws
IOException {
            if (!response.isSuccessful()) {
                //connection failed
                //provide random response {0, 1, 2} yourself using random
function
                Random r = new Random();
                int Low = 0;
                int High = 3;
                int value = r.nextInt(High - Low) + Low; //0 inclusive to 3
exclusive
                System.out.println("Uddhav: " + value);
                opponentHand = Hand.fromInt(value);

        //do the error reporting
        ErrorReporter.report(new IOException("Connection Error!"));


            } else {

                // do something wih the result
                BufferedReader reader = new BufferedReader(new
InputStreamReader(
                        new BufferedInputStream(response.body().byteStream())));
                String str = reader.readLine();

                Log.i(Thread.currentThread().getName(), str);

                while (!TextUtils.isEmpty(str)) {
                    opponentHand = Hand.fromInt(Integer.valueOf(str.trim()));
                    str = reader.readLine();
                }

            }
        }
    });
}
```

On Connection Fail, we must implement our logic to response the button clicks, which I did above using below code snippet.

```
 if (!response.isSuccessful()) {
                //connection failed
                //provide random response {0, 1, 2} yourself using random
function
```

```
                Random r = new Random();
                int Low = 0;
                int High = 3;
                int value = r.nextInt(High - Low) + Low; //0 inclusive, 3
exclusive
                System.out.println("Uddhav: " + value);
                opponentHand = Hand.fromInt(value);

        //do the error reporting
        ErrorReporter.report(new IOException("Connection Error!"));
            }
```

ConnectionError is reported on unsuccessful connection.

I also removed unnecessary typecast world. For instance,

```
View fragmentView = (View)
inflater.inflate(R.layout.rockpaperscissors_fragment, container, false);
```

These types of all lines are converted by

```
View fragmentView = inflater.inflate(R.layout.rockpaperscissors_fragment,
container, false);
```

7) Inside the `PrsButtonClickListener` method, I approached this solution as below.

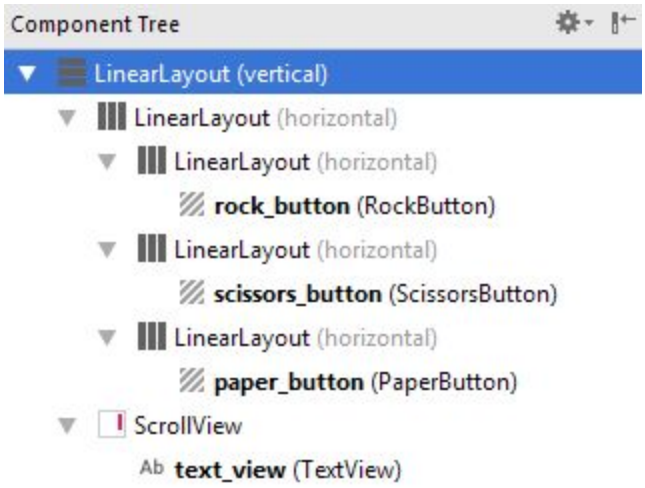
```
        CharSequence mText = "";
        if (result < 0) {

           mText = resultTextView.getText() + "LOSE"
                   + "(You: " + button.getHand().toString() + ", Opponent: "
                   + opponentHand.toString() + ")\n";

           resultTextView.setText(mText);

        }
```

8) On `rockpaperscissors_fragment` xml layout file. I modified it as below:



Surrounded each Button with horizontal LinearLayout, removed all nested layout_weight properties, provided margin, and padding, defined dimensions etc.

9) In class, `ErrorReporter,` I didn't change anything.
10) In `Hand` class, I didn't change anything.
11) In `HandButton, PaperButton, ScissorsButton, RockButton,` I didn't change anything.
12) Finally, I put the whole project in https://github.com/uddhavgautam/RPSGame