Tutorial Week 2 – 31285 MAD
4%

Part A: (aim to complete in class)

Objectives:

- Nested layouts
- Build two Activities
- Respond to user events.
- Pass data between Activities
- Use online resources (Google/Stack Overflow)
- Comment your code

1. (5%) Create a new Android Studio project with the following characteristics:
    - Application name: Exercise2
    - Package name: com.mad.exercise2
    - Min SDK: API15
    - Empty Activity
    - Activity name: MainActivity

2. (27%) Create a layout that looks exactly like fig 1. You must use nested layouts in order to achieve the desired result. All layouts must be LinearLayout. Also change the root layout from Android Studio's default ConstraintLayout to LinearLayout (for now; we wil

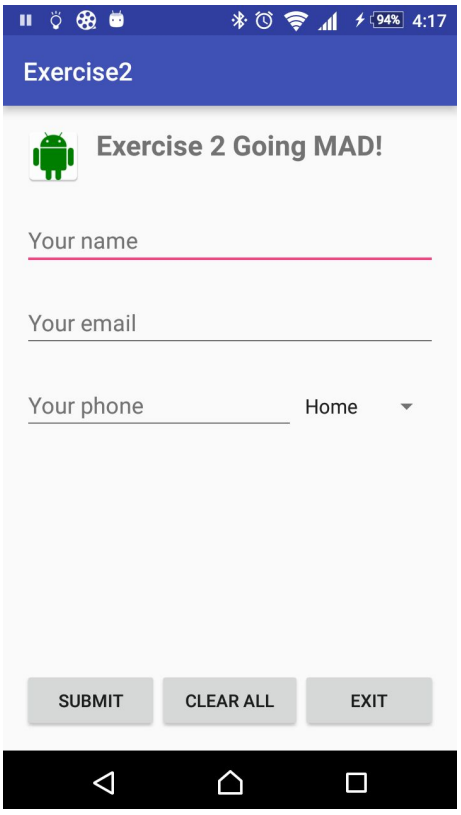

Fig. 1

Requirements:

- **Header:** Image on the left, Title, **bold** and fontsize 20sp.
- **Inputs:** 3 EditText widgets. A spinner widget with the items “Home, Work, Mobile”. The “your phone” edit text and spinner need to be on the same line. All lines stretch the width of the screen.
- **Footer:** 3 buttons that need to be evenly spaced horizontally. The buttons must always be at the bottom of the screen.

Hint 1: edit your activity_main.xml file by hand (i.e. Text tab) and do not use the graphical interface (i.e. Design tab). Try to use code completion. E.g. TextView<Tab><Tab><Tab>
Hint 2: You will need to create a string array in strings.xml for the spinner widget (use Google to find a snippet of code to define a Spinner using a string array in XML).

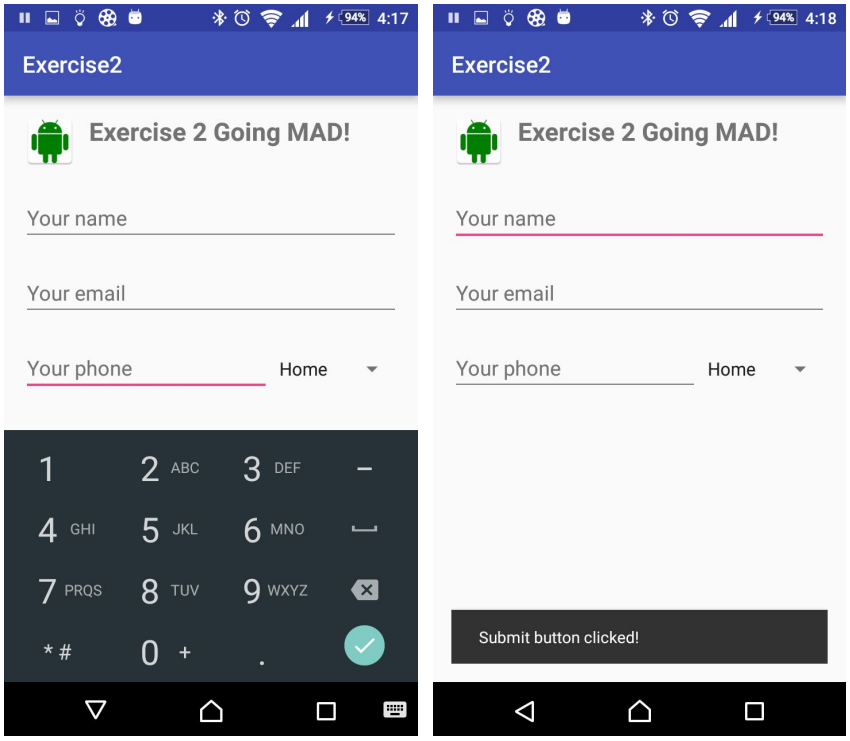

Fig2: MainActivity

3. (3%) When you click “Your phone” edit text your softkeyboard **must** show digits like fig 2.

4. Create a click listener for the “Submit” button widget in MainActivity using the anonymous inner class construction:

```
Button submitBtn = (Button)findViewById(R.id.main_activity_submit_btn);
submitBtn.setOnClickListener(new OnClickListener()
{   @Override
        public void onClick(View v)
        {     // button 1 was clicked!
        }
});
```

5. (5%) Show a snackbar message “Submit button clicked!” when you click the Submit button. See Fig 2 right. See link below for an example on how to use snackbar: http://www.androidhive.info/2015/09/android-material-design-snackbar-example

    - You need to add a CoordinatorLayout to your main_activity.xml file as the root container. Use the one from the design library. i.e. <android.support.design.widget.CoordinatorLayout>
    - Refer to the template-generated :Exercise1 code from last week for an example of both CoordinatorLayout and Snackbar.

6. (10%) Create click listeners for the “Clear All” and “Exit” buttons. When the “Clear All” button is pressed the three text fields should be cleared and the spinner should be set back to “Home”. When the “Exit” button is pressed the application should close gracefully (i.e. by going through the complete activity lifecycle: onPause, onStop, onDestroy). Ask your tutor if you are not sure what this means. Do NOT use System.exit()!

PART B (complete at home)

Watch Videos in Section 5 “Android App Development Essential Training” on LYNDA.com

7. (5%) Create another activity called ActivityTwo with 5 textviews, a checkbox and a Button at the bottom. (see fig 3 left). When you click the Submit button, start ActivityTwo.

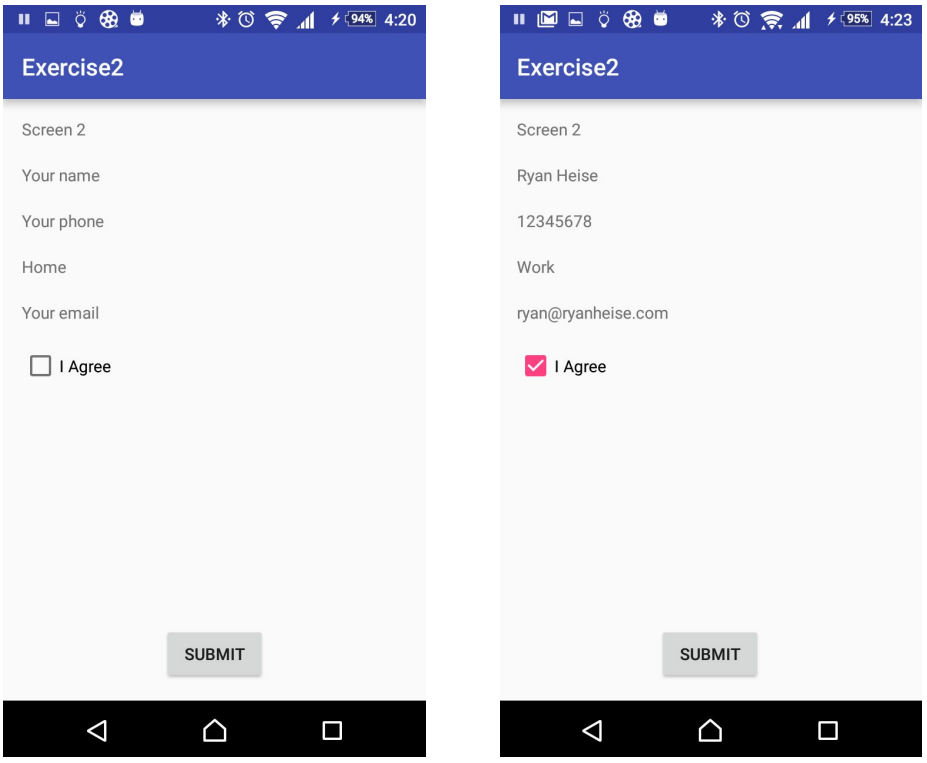

Fig 3: ActivityTwo

8. Adjust the code so that you **pass via an intent** the following data from MainActivity to ActivityTwo when you click “Submit” button in MainActivity:
    - “name”, “phone number”, “phone type”, and “email”.

9. (8%) Display the data entered in MainActivity in ActivityTwo (Fig 3. right).

10. Adjust the code in MainActivity so that you start ActivityTwo using this construct: **startActivityForResult(....);**

11. (2%) Implement the click listener for the ActivityTwo “Submit” button using this construct: public class ActivityTwo extends ActionBarActivity **implements View.OnClickListener**

12. When you click the submit button in ActivityTwo you collect the value of the “I agree” checkbox and you send a result back to MainActivity. You need to show a snackbar with the result, see Fig 4. The result is a **String** with the following text:
    - “I Agree” if checkbox checked.
    - “I Disagree” if checkbox not checked.

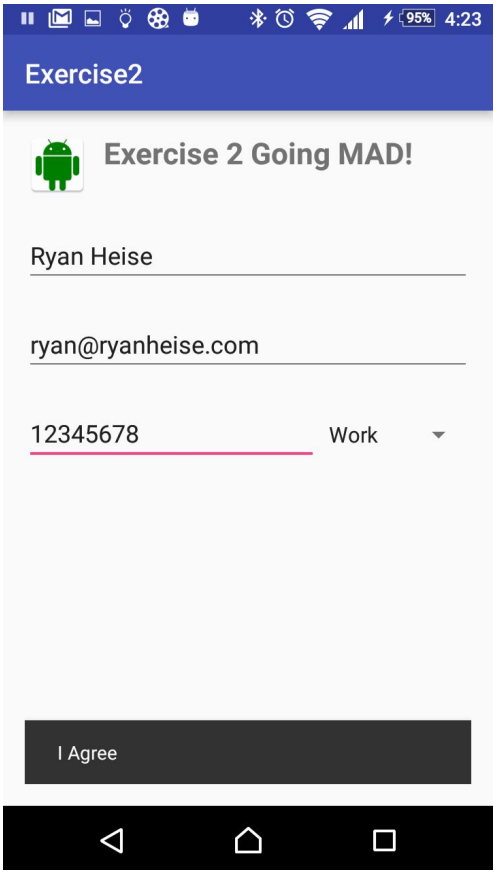

Fig4: Show snackbar with the result (I Agree/I Disagree)

13. In MainActivity you need to implement this method:
    public void onActivityResult(int requestCode, int resultCode, Intent data)

    This handles the result coming from ActivityTwo. You need to check that the requestCode is the same as you used in **startActivityForResult(...);**

14. (10%) Retrieve the result from the Intent in the onActivityResult(..) method and display a Snackbar showing the result (i.e “I Agree” or “I Disagree”).

15. (5%) Rework all your code so that you do not have hardcoded strings in your code. For example:

WRONG: nameTextView.setText(“replace me”);
CORRECT: nameTextView.setText(getString(R.string.replace_me));

16. (5%) Replace hardcoded Key strings with constants. For example:
    WRONG: getIntent().getExtras().getString(“key”);
    CORRECT: getIntent().getExtras().getString(EXTRA_KEY);
    where:
    public static final String EXTRA_KEY = “key”;
    Define a constant **once** and reuse it.

17. (2%) Rework all your code so that you use easy to understand naming for your variables. For example in your activity_one.xml file:
    WRONG: android:id="@+id/textview1"
    CORRECT: android:id="@+id/activity_one_title_textview"

18. (3%) Add all the Activity LifeCycle methods to MainActivity and add a Log.d(...) statement to each lifecycle method. Use the debugging mode in Android Studio and add a breakpoint on each Log.d(..) statement. Start your app in debug mode and step through it. Observe what is happening.

19. (10%) Comment all your code using Java coding standard. See https://source.android.com/source/code-style.html