TASK GUIDE (B3.06)

A. Objectives.

Student will start programing for MainActivity. This task will introduce how to handle ListView, open image selector intent, show a dialog, and go to another intent.

B. Requirements.

Hardware:

- 2 GB RAM minimum, 8 GB RAM recommended
- 2 GB of available disk space minimum, 4 GB Recommended (500 MB for IDE + 1.5 GB for Android SDK and emulator system image)
- 1280 x 800 minimum screen resolution
- Intel processor with support for Intel VT-x, Intel EM64T (Intel 64), and Execute Disable (XD) Bit functionality

Software:

- Microsoft Windows 7/8/10 (32-bit or 64-bit)
- JDK 8
- Android Studio IDE (Minimum 3.2) with AndroidX library.

C. Resources.

Documents:

- Guide

Test code:

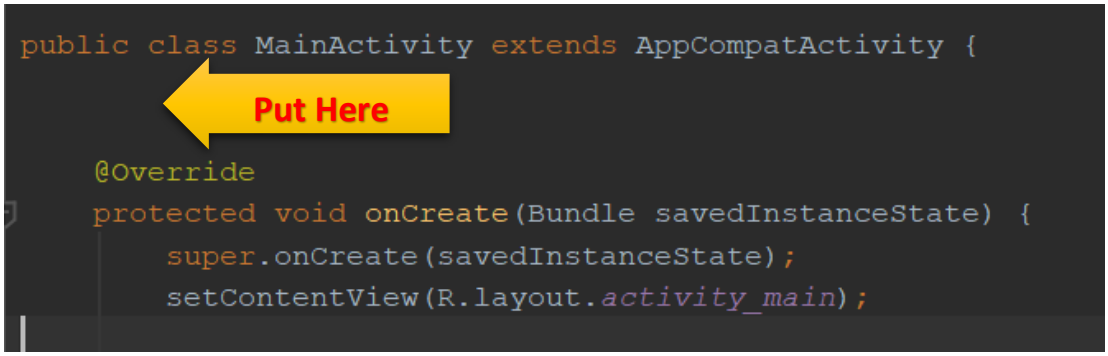
- TestB3MultiActivities061.java

D. Task Description.

Student start programming with MainActivity.

E. Specification.

1. Open “MainActivity.java” in java folder folder.
2. Declare all variables that represents all widgets in activity_main.xml.



Using this template:

```
private <Widget_Type> <variable_name>;
```

All the widgets are:

Widget Type	Widget Id
ImageButton	‘homeImage’, ‘awayImage’, ‘addHomePlayer’, ‘addAwayPlayer’
Button	‘startBtn’
EditText	‘homeTeam’, ‘awayTeam’
TextView	‘homePlayerNumber’, ‘awayPlayerNumber’
ListView	‘homePlayer’, ‘awayPlayer’

3. Also define some variables like below:

```
private final int RESULT_HOME_IMG = 1;
private final int RESULT_AWAY_IMG = 2;

private String homeImgPath = "";
private String awayImgPath = "";

private ArrayList<String> homePlayer = new ArrayList<>();
private ArrayList<String> awayPlayer = new ArrayList<>();

private AlertDialog homeDialog, awayDialog;
private final int EDITTEXT_ID = 900;
```

4. In the onCreate method, define all widgets variables, which has been declared in point 1, to the related widget id using this template:

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

    <variable_name> = (<Widget_type>)findViewById(R.id.<widget_id>);
    .
    .
}
```

5. Then, to show a dialog to input player name, we must create a new method to activate a AlertDialog, like below:

```
private AlertDialog getInputPlayerDialog(ArrayList<String> list,
          ListView listView, TextView playerNumber) {

    AlertDialog.Builder builder =
        new AlertDialog.Builder(new ContextThemeWrapper(this,
        R.style.AlertDialogCustom));
    builder.setMessage("Input Player Name:");
    LayoutInflater inflater = getLayoutInflater();
    View layout = inflater.inflate(R.layout.layout_dialog, null);
    builder.setView(layout);
    EditText playerName = (EditText)
        layout.findViewById(R.id.playerName);

    builder.setPositiveButton("OK", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
            final String name = playerName.getText().toString();
            list.add(name);
            loadListView(list,listView);
            playerNumber.setText(list.size()+" player(s)");
            playerName.setText("");
        }
    });
    builder.setNegativeButton("Cancel",null);
    return builder.create();
}
```

6. Then, in the 'onCreate' method, assign 'homeDialog' and 'awayDialog' button with the init method, like below:

```
homeDialog = getInputPlayerDialog
    (homePlayer,homePlayerList,homePlayerNumber);
awayDialog = getInputPlayerDialog
    (awayPlayer,awayPlayerList,awayPlayerNumber);
```

Then put onClickListener still in the 'onCreate' method for each addPlayer buttons to show the AlertDialog when it clicked.

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

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

7. Create a new method to load a ListView content from an ArrayList of String.
   - First we must declare an ArrayAdapter of String variable and contruct the object with context, a layout resource, a widget inside layout resource, and an ArrayList of String.
   - Then we set the adapter of ListView with the ArrayAdapter variable.

```
private void loadListView(ArrayList<String> list, ListView listView) {
   ArrayAdapter<String> arrayAdapter =
      new ArrayAdapter<String>(this, R.layout.layout_list, R.id.listItem, list);
   listView.setAdapter(arrayAdapter);
}
```

8. In the onCreate method, create a temporary variable with data type ArrayList<String>. Put a string "add 11 players" in that variable with this code:

```
<var_name>.add("add 11 players");
```

   Then use the method to update the 'homePlayerList' and 'awayPlayerList' ListView content with the temporary ArrayList variable.

```
loadListView(<var_name>,<homePlayerList_variable>);
loadListView(<var_name>,<awayPlayerList_variable>);
```

9. Now we create a method to process selected image when the Image Gallery opened. We must create an override method " onActivityResult" like below:

```
@Override
protected void onActivityResult(int reqCode, int resultCode, Intent data) {
      super.onActivityResult(reqCode, resultCode, data);

      ...
      ...
}
```

Code

In the code, we have to handle the 2 possibility if the user select an image or not. Use this code like below:

```
if (resultCode == RESULT_OK) {
   try {
      final Uri imageUri = data.getData();
      final InputStream imageStream =
          getContentResolver().openInputStream(imageUri);
      final Bitmap selectedImage =
          BitmapFactory.decodeStream(imageStream);

      if (reqCode==RESULT_HOME_IMG) {
         imgHome.setImageBitmap(selectedImage);
         homeImgPath = imageUri.toString();
      } else if (reqCode==RESULT_AWAY_IMG) {
         imgAway.setImageBitmap(selectedImage);
         awayImgPath = imageUri.toString();
      }

   } catch (FileNotFoundException e) {
      e.printStackTrace();
      Toast.makeText(MainActivity.this, "Something went wrong",
                  Toast.LENGTH_LONG).show();
   }
} else {
   Toast.makeText(MainActivity.this, "You haven't picked Image",
               Toast.LENGTH_LONG).show();
}
```

Select Code

Unselect Code

Please imagine what will happen to the app.

10. In the onCreate method, create an OnClickListener for ImageButton 'homeImage' and 'awayImage' with call method 'onActivityResult' like below:

```
imgHome.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
       Intent photoPickerIntent = new Intent(Intent.ACTION_PICK);
       photoPickerIntent.setType("image/*");
       startActivityForResult(photoPickerIntent, RESULT_HOME_IMG);
    }
});

imgAway.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
       Intent photoPickerIntent = new Intent(Intent.ACTION_PICK);
       photoPickerIntent.setType("image/*");
       startActivityForResult(photoPickerIntent, RESULT_AWAY_IMG);
    }
});
```

11. Create a method to check the completeness of main form to activate the start button, like below:

```
private boolean isFormComplete() {
   if (homeTeam.getText().toString().isEmpty()) {
      Toast.makeText(MainActivity.this,
          "Home team name is still empty", Toast.LENGTH_LONG).show();
      return false;
   }
   if (awayTeam.getText().toString().isEmpty()) {
      Toast.makeText(MainActivity.this,
          "Away team name is still empty", Toast.LENGTH_LONG).show();
      return false;
   }
   if (homeImgPath.isEmpty()) {
      Toast.makeText(MainActivity.this,
          "Home team logo is still empty", Toast.LENGTH_LONG).show();
      return false;
   }
   if (awayImgPath.isEmpty()) {
      Toast.makeText(MainActivity.this,
          "Away team logo is still empty", Toast.LENGTH_LONG).show();
      return false;
   }
   if (homePlayer.size()<11) {
      Toast.makeText(MainActivity.this,
          "Home player list is not complete", Toast.LENGTH_LONG).show();
      return false;
   }
   if (awayPlayer.size()<11) {
      Toast.makeText(MainActivity.this,
          "Away player list is not complete", Toast.LENGTH_LONG).show();
      return false;
   }
   return true;
}
```

12. Then, create a new method 'openPlayActivity' again to activate an Intent of 'PlayActivity' with passing some necessary variables:

```
private void openPlayActivity() {
    Intent play = new Intent(getApplicationContext(),PlayActivity.class);
    play.putExtra("HOME_TEAM_NAME", homeTeam.getText().toString());
    play.putExtra("AWAY_TEAM_NAME", awayTeam.getText().toString());
    play.putExtra("HOME_IMG_URI", homeImgPath);
    play.putExtra("AWAY_IMG_URI", awayImgPath);
    play.putStringArrayListExtra("HOME_TEAM_PLAYER",homePlayer);
    play.putStringArrayListExtra("AWAY_TEAM_PLAYER",awayPlayer);
    startActivity(play);
}
```

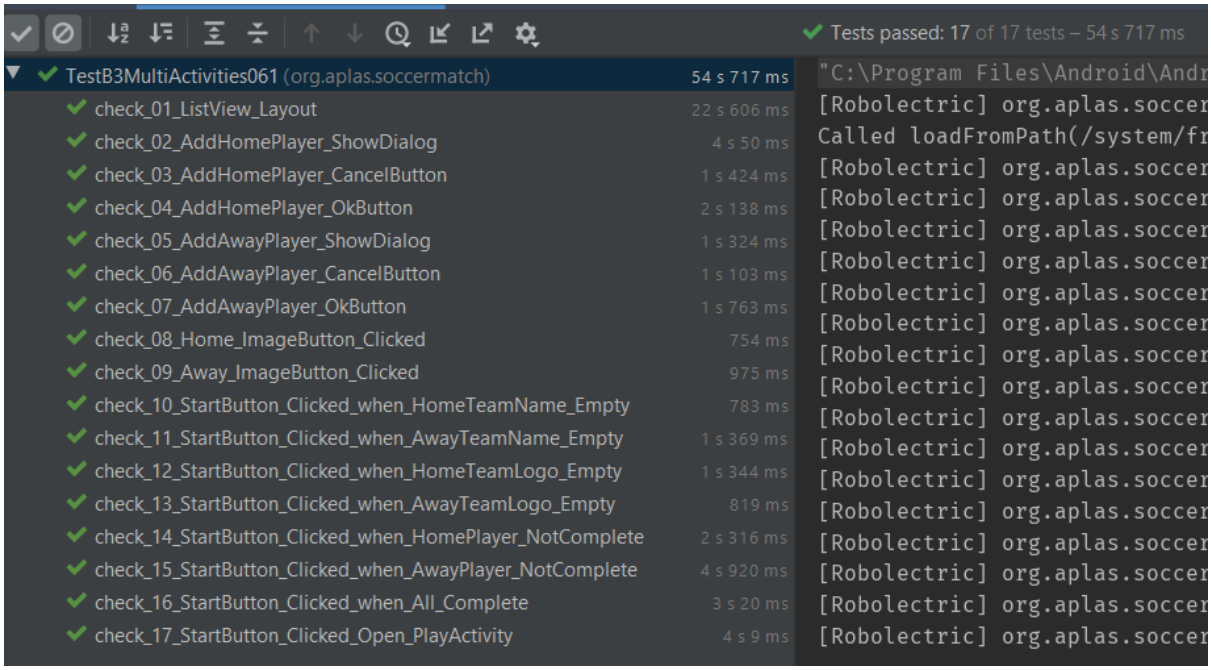
In this point, you have to understand how to put some types of variables and how to start an Activity as Intent.

13. In the onCreate method, create an OnClickListener for Button 'startBtn' with calling method 'openPlayActivity' when the form is complete:

```
startBtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (isFormComplete()) {
            openPlayActivity();
        }
    }
});
```

F. Testing.

1. Copy "TestB3MultiActivities061.java" file to "org.aplas.soccermatch (test)" folder.
2. Right click on the "TestB3MultiActivities061.java" file then choose Run. It may take long time to execute.
3. Get the result of your task. If passed you will get green check like below. If the test failed, you will get orange check get the messages and you must check your work again.



You have to try until get all green checkes and continue to the next task.

G. Note.

You can modify the style and the layout arrangement of the Play activity by yourself to improve the UI design.