Department of Information Systems and Technologies

# CTIS 487 - Mobile Application Development

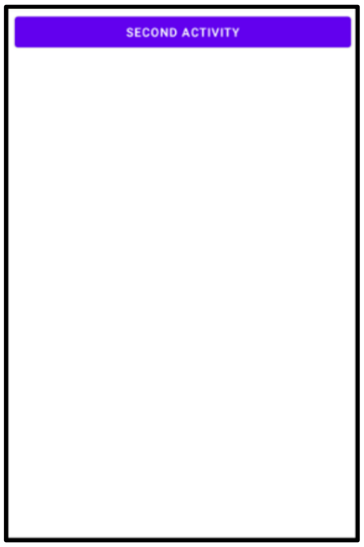
Spring 2022 - 2023

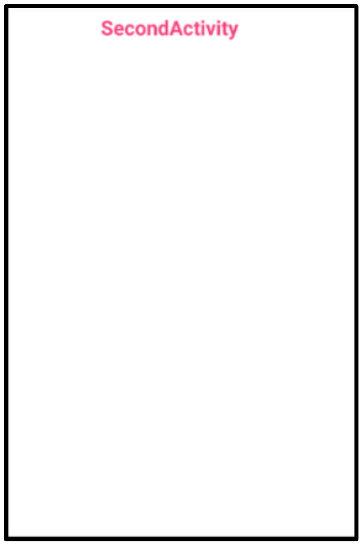
**Lab Guide # 6**

| | |
|---|---|
| **OBJECTIVES :** Intent + Constraint Layout | |
| **Instructor** : Neşe ŞAHİN ÖZÇELİK<br>**Assistant** : Sena Altun | |

**STEP 1. Open another activity.**

- Create two activities (MainActivity, SecondActivity). Over app folder right click and select **New**, select **Activity,** select **Empty Activity** and then give the second activity class name and xml file name.





- If user clicks on the **SECOND ACTIVITY** button, second activity will be opened.

```
public class MainActivity extends AppCompatActivity implements View.OnClickListener {
  …
  @Override
  protected void onCreate(Bundle savedInstanceState) {
          …
          btnSecond.setOnClickListener(this);
  }
}
```

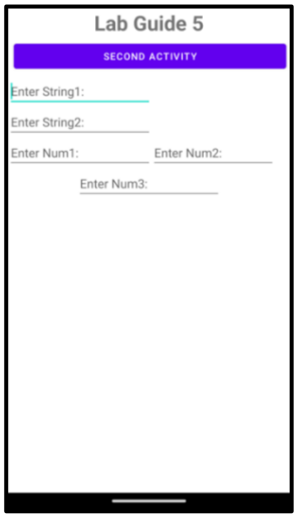
**In MainActivity**, implement the onClick(…) method and direct the execution to the SecondActivty.

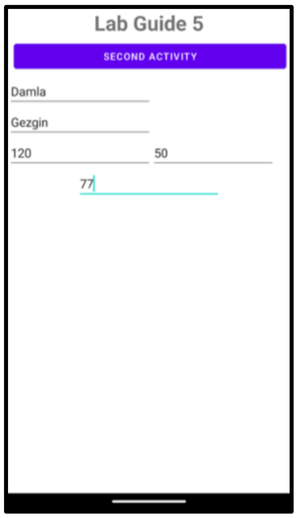
```
@Override
public void onClick(View view) {
  Intent intent = null;
   if (view.getId() == R.id.btnSecond) {
    intent = new Intent(this, SecondActivity.class);
    startActivity(intent);
  }
}
```

- Run your application and see that second activity can be opened.

**STEP 3. Send data to another activity**

- Add Five EditText on MainActivity.
- When SECOND ACTIVITY button is clicked, edit text values will send to SecondActivity.
- Form the Second Activity, get these values, concatenate first two string and find the average of three integers.



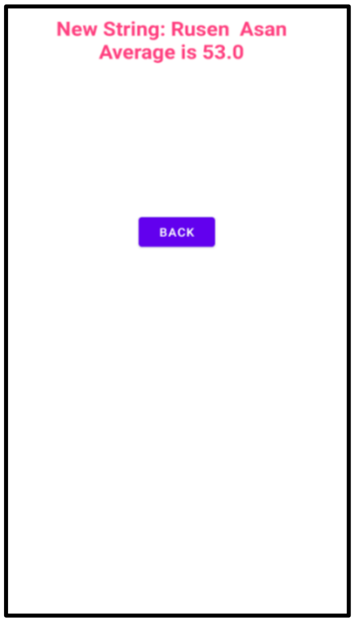


- Modify the **onClick** method of **MainActivity** to send data. You may use **direct** or **bundle** method.

```
@Override
public void onClick(View view) {

Intent intent = null;
  if (view.getId() == R.id.btnSecond) {
    intent = new Intent(this, SecondActivity.class);
    int num1 = Integer.parseInt(editNum1.getText().toString());
    int num2 = Integer.parseInt(editNum2.getText().toString());
    int num3 = Integer.parseInt(editNum3.getText().toString());
    String str1 = editSTR1.getText().toString();
    String str2 = editSTR2.getText().toString();
    Bundle b = new Bundle();
    b.putInt("num1", num1);
    b.putInt("num2", num2);
    b.putInt("num3", num3);
    b.putString("str1", str1);
    b.putString("str2", str2);
    intent.putExtras(b);

    startActivity(intent);
  }

}
```

- Modify the **onCreate** method of **SecondActivity** to **get and display the data which is send from Main Activity**. If you use direct method in MainActivity, to get the data use direct method. If Bundle method is used in MainActivity, to get data use bundle method.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    // Hiding title bar using code
    getSupportActionBar().hide();
 getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
WindowManager.LayoutParams.FLAG_FULLSCREEN);

    setContentView(R.layout.activity_second);

     intent = getIntent();

    Bundle b = intent.getExtras();
    int num1 = b.getInt("num1");
    int num2 = b.getInt("num2");
    int num3 = b.getInt("num3");
    String str1 = b.getString("str1");
    String str2 = b.getString("str2");
    double res = (num1+num2+num3) / 3;
    String str = str1.concat(" ").concat(str2);

    txtRes.setText("New String: "+str);
    txtRes.append("\nAverage is " + res  );
    msg = "New String "+str+" and Average: "+ res;
}
```


New String: Rusen  Asan
Average is 53.0

**STEP 4. When SecondActivity is closed, return to the MainActivity.**

- Put the button **BACK** in SecondActivity. When the user click on the button, execution will returned to the MainActivity.
- Write onClick method for the BACK button of the SecondActivity.

```
@Override
public void onClick(View view) { // add the back button to listener
     finish();
}
```

- Main activity will get data from the SecondActivity whenever SecondActivity is finished/closed. Modify the **onClick** method of the **MainActivity to receive data from SecondActivity.** Instead of `startActivity(intent)` use **`ActivityResultLauncher<Intent>`** object and call its **`launch(intent)`** method.

```
public class MainActivity extends AppCompatActivity implements View.OnClickListener {
    … … …
    ActivityResultLauncher<Intent> secondActivityIntentLauncher;
}
```

- Modify the **onCreate** method of the MainActivity to create the secondActivityIntentLauncher object.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
            …
            …
     // Initialize Activity Result Launcher
     secondActivityIntentLauncher = registerForActivityResult(
          new ActivityResultContracts.StartActivityForResult(),
          new ActivityResultCallback<ActivityResult>() {
              @Override
              public void onActivityResult(ActivityResult result) {
                  //When SecondActivity is closed, onActivityResult method will be 
executed.
              }
    });
  }
}
```

- Modify the **onClick** method of the MainActivity to call the launch(intent) method to open the SecondActivity by informing operating system a data will be taken from SecondActivity whenever it is.

```
@Override
public void onClick(View view) {

    Intent intent = null;
    if (view.getId() == R.id.btnSecond) {
        intent = new Intent(this, SecondActivity.class);

        int num1 = Integer.parseInt(editNum1.getText().toString());
        int num2 = Integer.parseInt(editNum2.getText().toString());
        int num3 = Integer.parseInt(editNum3.getText().toString());
        String str1 = editSTR1.getText().toString();
        String str2 = editSTR2.getText().toString();

        Bundle b = new Bundle();
        b.putInt("num1", num1);
        b.putInt("num2", num2);
        b.putInt("num3", num3);
        b.putString("str1", str1);
        b.putString("str2", str2);

        intent.putExtras(b);

        secondActivityIntentLauncher.launch(intent);
    }
}
```

**STEP 5. When SecondActivity is closed, execution will returned to the MainActivity and send data/result to the MainActivity.**

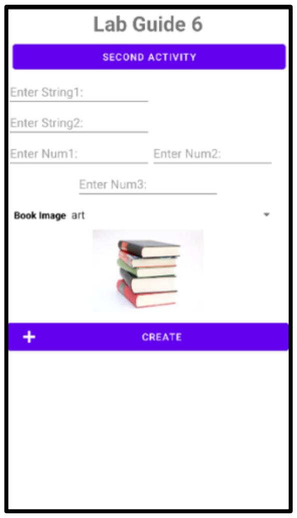
- Modify the **onClick** method of the **SecondActivity;**

```
@Override
public void onClick(View v) {
    Intent intent = new Intent();
    intent.putExtra("res",msg);

    setResult(RESULT_OK, intent);
    finish();
}
```

- Modify the **onActivityResult()** method of the MainActivity, to get data/results from SecondActivity.

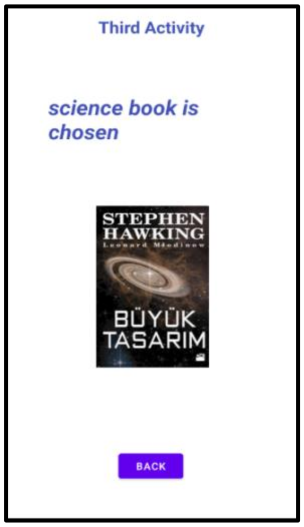
```
secondActivityIntentLauncher = registerForActivityResult(
        new ActivityResultContracts.StartActivityForResult(),
        new ActivityResultCallback<ActivityResult>() {
        @Override
        public void onActivityResult(ActivityResult result) {
              if (result.getResultCode() == Activity.RESULT_OK) {
                    Intent data = result.getData();
                    String msg = data.getStringExtra("res");
                    Toast.makeText(MainActivity.this, msg, Toast.LENGTH_LONG).show();
              }
        }
});
```

**STEP 6. (Send a message to another activity)**



- Put on the MainActivity a TextView to display title with a color animation, an ImageView to display selected book type image, spinner to display book types with **art**, **science** and **music** items and a button CREATE to open ThirdActivity.
- When an item is selected from the spinner, image with the selected book type name is displayed on the ImageView.
- Create a ThirdActivity and put an ImageView and a BACK button on it.
- When **CREATE** button is clicked, open the ThirdActivity and send selected book type item (art, science or music) to the ThirdActivity. On ThirdActivity get book type and display the book image with the given type name as art.jpg, science.jpg or musics.jpg).
- Put an icon to the CREATE button.
    - Insert the given attribute to the xml file into the CREATE button;

      **`app:icon="@android:drawable/ic_input_add"`**



science book is chosen

- When **BACK** button on the ThirdActivity is clicked, ThirdActivity will be closed and ThirdActivity will return a string message like “art book is chosen” to MainActivity and execution will directed to the MainActivity. Whenever MainActivity gets data from ThirdActivity, an AlertDialog will be displayed on it.

**For step 6 implement the following statements:**

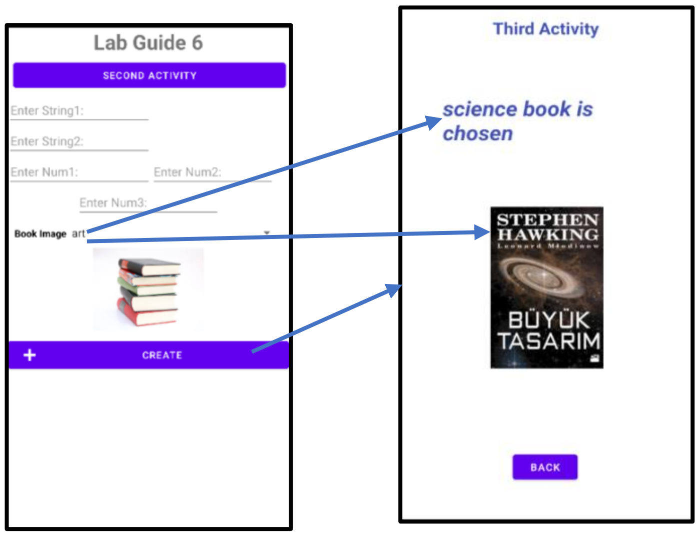
- Modify the onCreate method of the MainActivity to add the spinner to its event handler so that when an item is selected, on the imageView book image from the selected book type will be displayed.

```
int[] imgIds = {R.drawable.art, R.drawable.science, R.drawable.music};
spImage.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
   @Override
   public void onItemSelected(AdapterView<?> parent, View view, int i, long id) {
      imageSP.setImageResource(imgIds[i]);
   }
   @Override
   public void onNothingSelected(AdapterView<?> parent) {}
});
```

- Assign **color animation** the title text.

```
private ValueAnimator colorAnim;
...

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    … …. …
    colorAnim = ObjectAnimator.ofInt(tvTitle, "textColor", Color.RED,Color.BLUE );
    colorAnim.setDuration(3000);
    colorAnim.setEvaluator(new ArgbEvaluator());
    colorAnim.setRepeatCount(ValueAnimator.INFINITE);
    colorAnim.setRepeatMode(ValueAnimator.REVERSE);
    colorAnim.start();
}
```

- When **CREATE** button on the MainActivity is clicked, send selected book type to the ThirdActivity and display book message and set the resource of the image view according to the book name on ThirdActivity.



- Modify the **onClick** method to open the ThirdActivity when Create button is clicked and send the selected book type to the ThirdActivty.

```
public class MainActivity extends AppCompatActivity implements View.OnClickListener {
    ActivityResultLauncher<Intent> secondActivityIntentLauncher;
    ActivityResultLauncher<Intent> createActivityIntentLauncher;
    … … … …
    btnCreate.setOnClickListener(this);
    … … … …
    @Override
    public void onClick(View view) {
        Intent intent = null;
        if (view.getId() == R.id.btnSecond) {
            intent = new Intent(this, SecondActivity.class);

            … … … …

            secondActivityIntentLauncher.launch(intent);

        } else if (view.getId() == R.id.btnCreate) {
            intent = new Intent(this, ThirdActivity.class);
            Bundle b = new Bundle();
            b.putString("imgName", spImage.getSelectedItem().toString());
            intent.putExtras(b);
            createActivityIntentLauncher.launch(intent);
        }
    }
```

- Modify the **onCreate** method of the **ThirdActivity** to get book type.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    // Hiding title bar using code
    getSupportActionBar().hide();
    getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
WindowManager.LayoutParams.FLAG_FULLSCREEN);

    setContentView(R.layout.activity_third);
    intent = getIntent();
    Bundle b = intent.getExtras();
    String imageName = b.getString("imgName");
    txtImageInfo.setText(imageName+" book is chosen");
    int resImageID = getResources().getIdentifier( imageName, "drawable", getPackageName());
    imageView.setImageResource(resImageID);
}
```

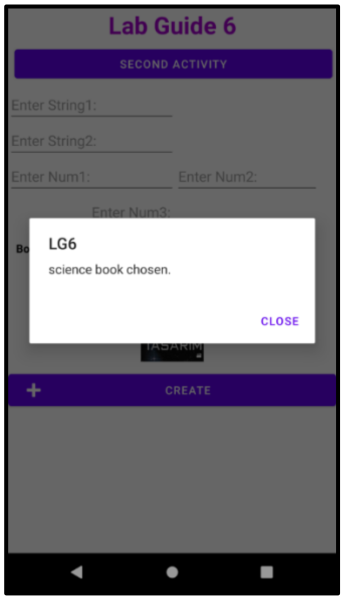
- When **ThirdActivity is closed, execution will returned to the MainActivity and send data/result(s) to the MainActivity.**

```
@Override
public void onClick(View view) {
     Intent intent = new Intent();
     intent.putExtra("res", msg);
     setResult(RESULT_OK, intent);
     finish();
}
```

- From the MainActivity, to get data/result(s) from ThirdActivity, **modify** the **onActivityResult**() method in the MainActivity thirdActivityIntentLauncher.

```
thirdActivityIntentLauncher = registerForActivityResult(
      new ActivityResultContracts.StartActivityForResult(),
      new ActivityResultCallback<ActivityResult>() {
      @Override
       public void onActivityResult(ActivityResult result) {
         if (result.getResultCode() == Activity.RESULT_OK) {
             Intent receivedIntent = result.getData();
             String msg = receivedIntent.getStringExtra("res");
         }
      }
});
```

- When the ThirdActivity is closed, execution returns to the MainActivity and alert dialog is displayed on the MainActivity with the message taken from the ThirdActivity. So that implement the makeAndShowDialog () method and modify the onActivityResult to call that method.



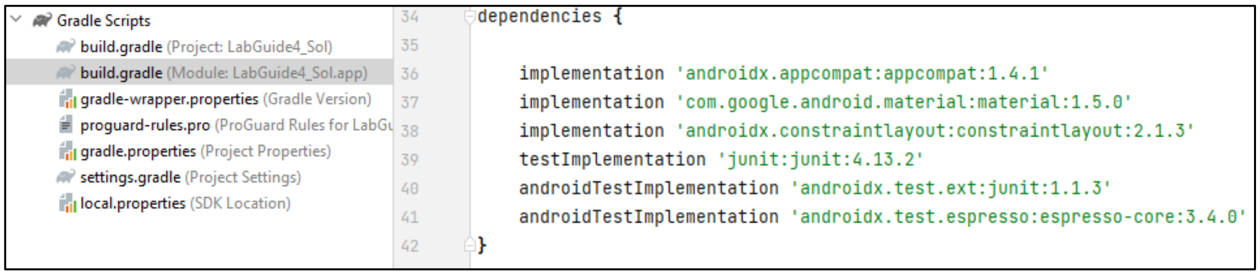
LG6
science book chosen.

```
thirdActivityIntentLauncher = registerForActivityResult(
   new ActivityResultContracts.StartActivityForResult(),
   new ActivityResultCallback<ActivityResult>() {
   @Override
   public void onActivityResult(ActivityResult result) {
      if (result.getResultCode() == Activity.RESULT_OK) {
          Intent receivedIntent = result.getData();
          String msg = receivedIntent.getStringExtra("res");
          makeAndShowDialog(msg + " is chosen.");
         }}
});
private void makeAndShowDialog(String message) {
    AlertDialog.Builder box = new AlertDialog.Builder(this);
    box.setTitle("LG6");
    box.setMessage(message);
    box.setPositiveButton("Close", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
        }
    });
    box.create();
    box.show();
}
```

**You can watch the videos about constraint layout from the moodle part 1 to part 5 before starting the guide. The contrasts between constraint and relative layout are further discussed in the sources provided below. PERFORMANCE is the most significant distinction between them.**

- https://stackoverflow.com/questions/37321448/differences-between-constraintlayout-and-relativelayout.
- https://www.quora.com/What-is-the-difference-between-constraint-layout-and-relative-layout

The following screen shows the constraint layout usage, to see the necessary implementation open the build app **(Module file)** part, it is already implemented.


Gradle Scripts
build.gradle (Project: LabGuide4_Sol)
build.gradle (Module: LabGuide4_Sol.app)
gradle-wrapper.properties (Gradle Version)
proguard-rules.pro (ProGuard Rules for LabGu
gradle.properties (Project Properties)
settings.gradle (Project Settings)
local.properties (SDK Location)

```
dependencies {

    implementation 'androidx.appcompat:appcompat:1.4.1'
    implementation 'com.google.android.material:material:1.5.0'
    implementation 'androidx.constraintlayout:constraintlayout:2.1.3'
    testImplementation 'junit:junit:4.13.2'
    androidTestImplementation 'androidx.test.ext:junit:1.1.3'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.4.0'
}
```