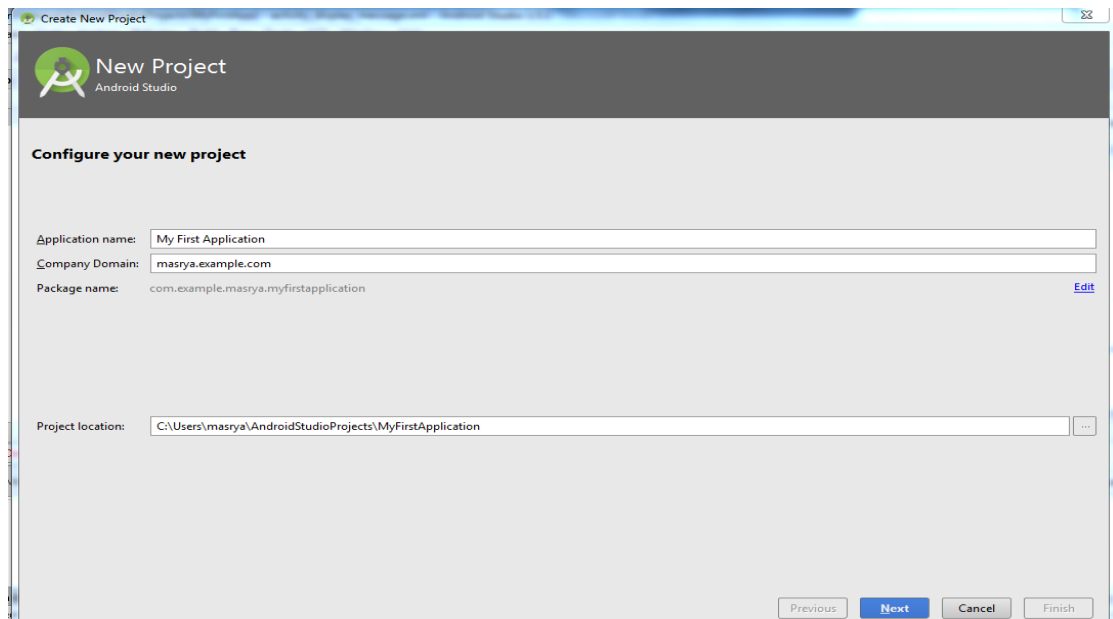


In this lab, you will learn how to:

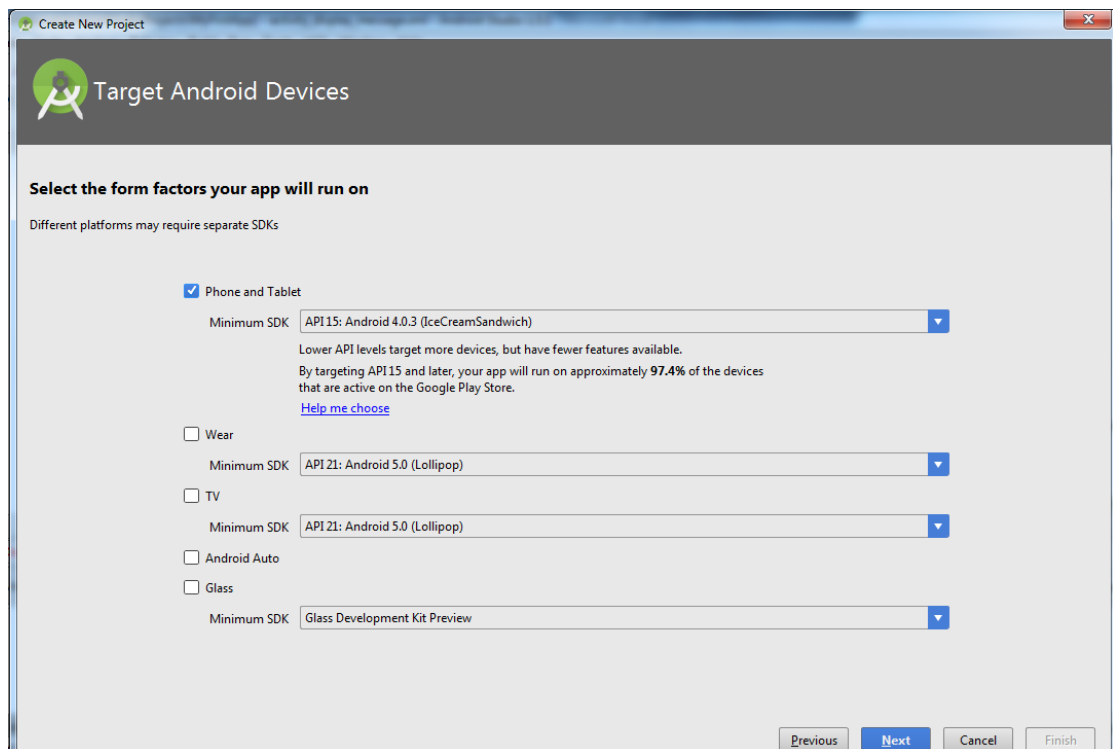
- I. Create a new project
- II. Run a project
- III. Create a simple UI
- IV. Starting another activity

I. Create an Android Project

From file menu: New > Android Application Project



The screenshot shows the 'Create New Project' dialog in Android Studio. The title bar says 'Create New Project'. The main heading is 'New Project' with the Android Studio logo. Below it, the subtitle is 'Configure your new project'. There are three input fields: 'Application name' with the value 'My First Application', 'Company Domain' with the value 'masrya.example.com', and 'Package name' with the value 'com.example.masrya.myfirstapplication'. There is an 'Edit' link next to the package name. Below these fields is the 'Project location' field with the value 'C:\Users\masrya\AndroidStudioProjects\MyFirstApplication' and a browse button (...). At the bottom, there are four buttons: 'Previous', 'Next', 'Cancel', and 'Finish'.



The screenshot shows the 'Target Android Devices' dialog in Android Studio. The title bar says 'Create New Project'. The main heading is 'Target Android Devices' with the Android Studio logo. Below it, the subtitle is 'Select the form factors your app will run on'. There is a note: 'Different platforms may require separate SDKs'. There are five options with checkboxes and dropdown menus for 'Minimum SDK':

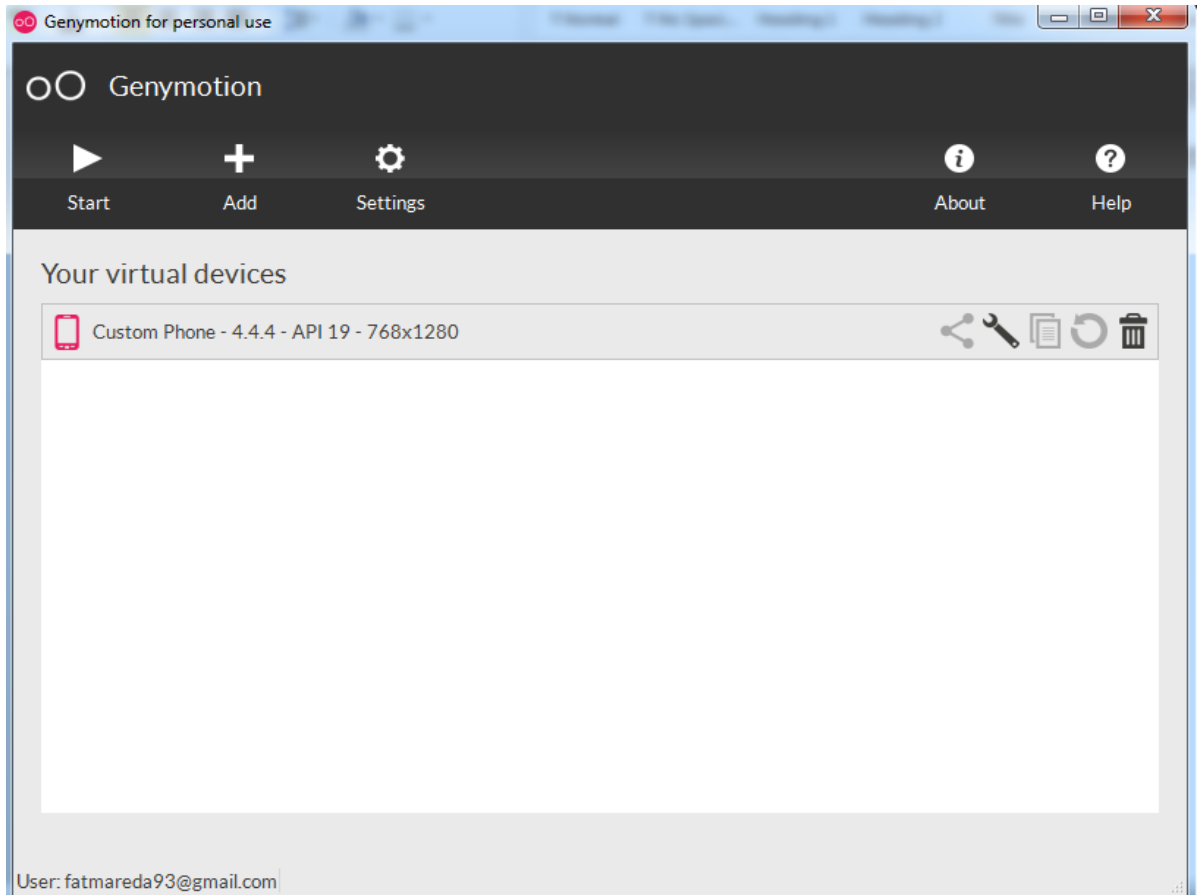
- ☒ Phone and Tablet: Minimum SDK is 'API 15: Android 4.0.3 (IceCreamSandwich)'. Below it, text says: 'Lower API levels target more devices, but have fewer features available. By targeting API 15 and later, your app will run on approximately 97.4% of the devices that are active on the Google Play Store.' There is a link 'Help me choose'.
- ☐ Wear: Minimum SDK is 'API 21: Android 5.0 (Lollipop)'.
- ☐ TV: Minimum SDK is 'API 21: Android 5.0 (Lollipop)'.
- ☐ Android Auto: Minimum SDK is 'API 21: Android 5.0 (Lollipop)'.
- ☐ Glass: Minimum SDK is 'Glass Development Kit Preview'.

At the bottom, there are four buttons: 'Previous', 'Next', 'Cancel', and 'Finish'.

Leave all the details for the activity in their default state and click Finish.

II. Running on the emulator

- Go to genymotion website <https://www.genymotion.com/>
- Create account and download it
- Install it , and then sign in
- Create a New virtual device
- Start it



You can also run on your own android device

III. Creating a Simple UI

1. Adding a text field

- Open the activity_main.xml file from the App/src/main/res/layout/activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal"
    >
    <EditText
        android:id="@+id/edit_message"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:hint="@string/edit_message"
    />
</LinearLayout>
```

- Then add a string resource for the EditText, open App/src/main/res/values/strings.xml

```
<resources>
    <string name="app_name">My First App</string>
    <string name="edit_message">Enter message</string>
</resources>
```

2. Adding a button

- Open the activity_main.xml file from the App/src/main/res/layout

```
<Button
    android:id="@+id/send_button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/send_button"
/>
```

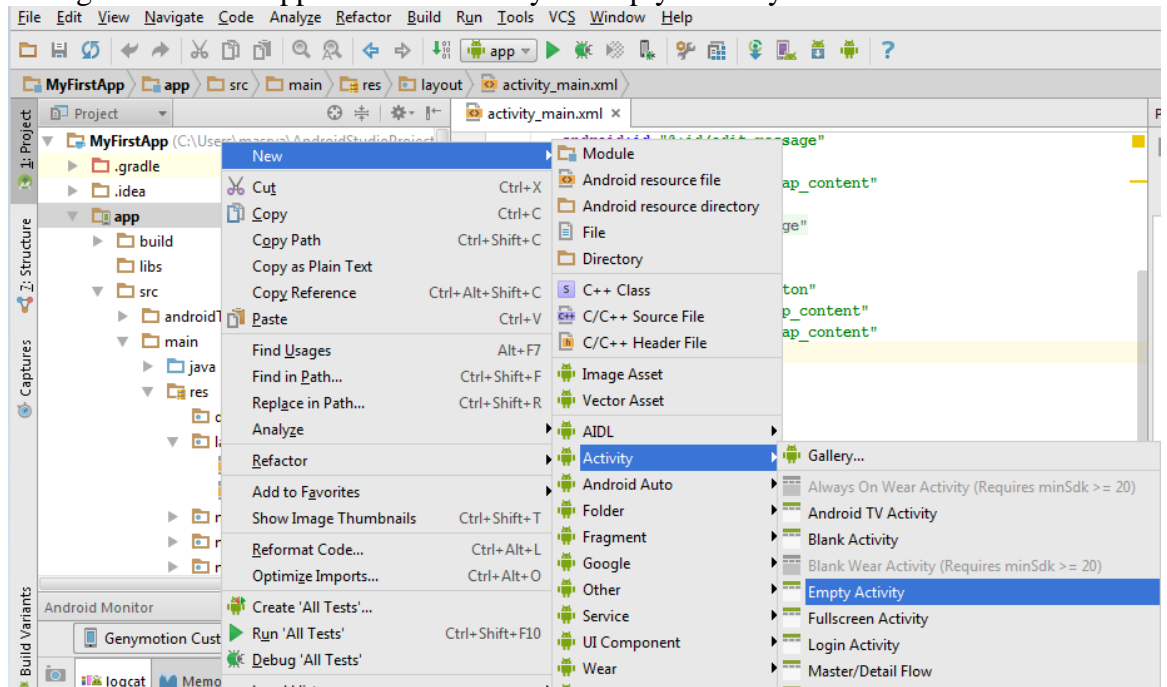
- Then add a string resource for the EditText, open App/src/main/res/values/strings.xml

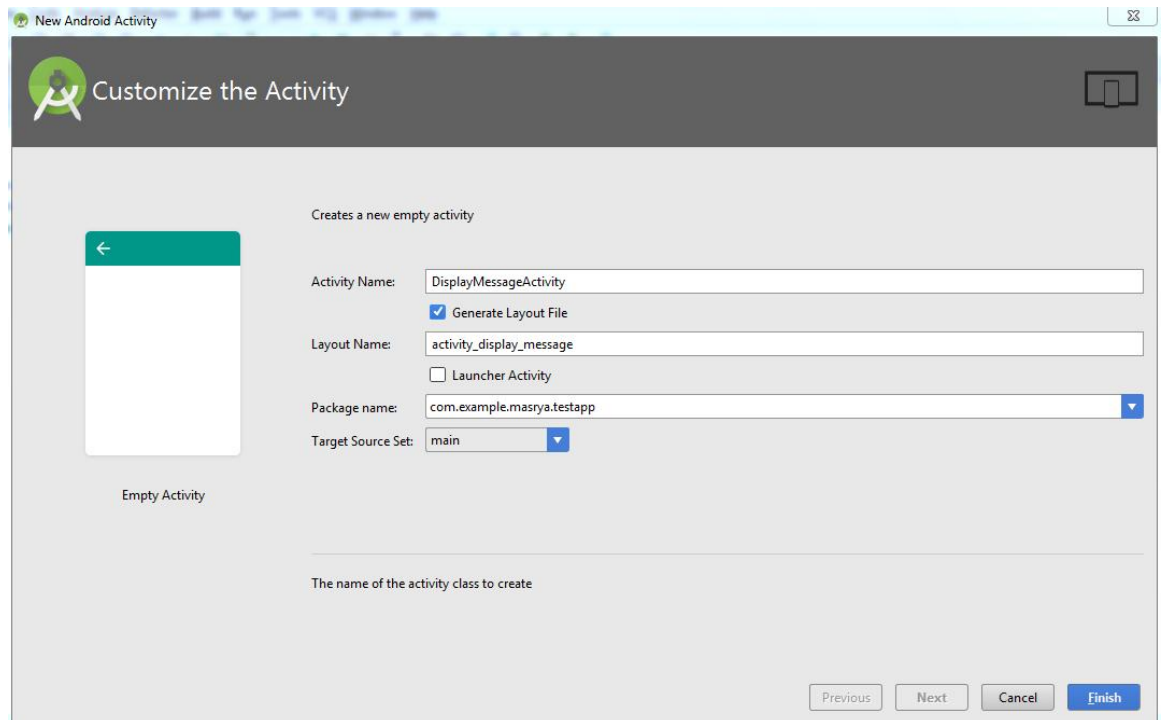
```
<resources>
    <string name="app_name">My First App</string>
    <string name="edit_message">Enter message</string>
    <string name="send_button">Send</string>
</resources>
```

IV. Starting another activity

1. Create a new activity

- Right mouse on App > New > Activity > Empty Activity





2. Add a text view

- Open the activity_main.xml file from the App/src/main/res/layout/activity_display_message.xml

```
<TextView
    android:id="@+id/display_text"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textSize="80dp"
/>
```

- Then add a string resource for the EditText, open App/src/main/res/values/strings.xml

```
<string name="display_text"></string>
```

3. Create an Intent

- In MainActivity we will read the text and then send it to the author activity by click to the button

```
import android.content.Intent;
import android.widget.Button;
import android.widget.EditText;
import android.view.View;

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

```

public void sendMessage() {
    Button send = (Button) findViewById(R.id.send_button);
    send.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            EditText text = (EditText) findViewById(R.id.edit_message);
            String message = text.getText().toString();
            Intent intent = new
            Intent(MainActivity.this, DisplayMessageActivity.class);
            intent.putExtra("Message", message);
            startActivity(intent);
        }
    });
}
}

```

4. Receive the Intent

```

import android.content.Intent;
import android.widget.TextView;

protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_display_message);
    displayMessage();
}

public void displayMessage() {
    TextView display_text = (TextView) findViewById(R.id.display_text);
    Intent intent = getIntent();
    String message = intent.getStringExtra("Message");
    display_text.setText(message);
}
}

```

▪ Run the application.

