Swami Keshvanand Institute of Technology Management &

Gramothan, Jaipur

Lab Manual

(6CS4-24) Mobile Application Development Lab Semester: VI

Branch: Computer Science & Engineering

Session 2021-22

Faculty:

Garima Gupta (Assistant Professor) Pratipal Singh (Assistant Professor) Dr. Rajat Goel (Associate Professor)

Pawan Kumar Patidar (Assistant Professor)

Department of Computer Science & Engineering

Swami Keshvanand Institute of Technology Management & Gramothan,

Ramnagaria, Jaipur-302017

COURSE FILE

(6CS4-24) Mobile Application Development Lab

VERSION 1.0

	AUTHOR/ OWNER	REVIEWED BY	APPROVED BY
NAME	Garima Gupta Pratipal Singh Dr. Rajat Goel Pawan Kumar Patidar		
DESIGNATION	Assistant Professor Assistant Professor Associate professor Assistant Professor		

SIGNATURE

LAB ETHICS

DO's

1. Please switch off the Mobile/Cell phone before entering Lab.
2. Enter the Lab with complete source code and data.
3. Check whether all peripheral are available at your desktop before proceeding for program.
4. Intimate the lab In charge whenever you are incompatible in using the system or in case software get corrupted/ infected by virus.
5. Arrange all the peripheral and seats before leaving the lab.
6. Properly shutdown the system before leaving the lab.
7. Keep the bag outside in the racks.
8. Enter the lab on time and leave at proper time.
9. Maintain the decorum of the lab.
10. Utilize lab hours in the corresponding experiment.
11. Get your Cd / Pendrive checked by lab In charge before using it in the lab.

Don'ts

1. No one is allowed to bring storage devices like Pan Drive /Floppy etc. in the lab.
2. Don't mishandle the system.
3. Don't leave the system on standing for long
4. Don't bring any external material in the lab.
5. Don't make noise in the lab.
6. Don't bring the mobile in the lab. If extremely necessary then keep ringers off.
7. Don't enter in the lab without permission of lab Incharge.
8. Don't litter in the lab.
9. Don't delete or make any modification in system files.
10. Don't carry any lab equipments outside the lab

INSTRUCTIONS

BEFORE ENTERING IN THE LAB

-
    - All the students are supposed to prepare the theory regarding the next program.
    - Students are supposed to bring the practical file and the lab copy.
    - Previous programs should be written in the practical file.
    - Algorithm of the current program should be written in the lab copy.
    - Any student not following these instructions will be denied entry in the lab.

WHILE WORKING IN THE LAB

-
    - Adhere to experimental schedule as instructed by the lab incharge.
    - Get the previously executed program signed by the instructor.
    - Get the output of the current program checked by the instructor in the lab copy.
    - Each student should work on his/her assigned computer at each turn of the lab.
    - Take responsibility of valuable accessories.
    - Concentrate on the assigned practical and do not play games.
    - If anyone caught red handed carrying any equipment of the lab, then he will have to face serious consequences.

Marking Scheme

Total Marks- 75

Internal Assessment Marks Distribution

Attendance	File Work	Performance	Viva	Total
5	10	20	10	45

External Assessment Marks Distribution(Depends on Examiner

File Work	Performance	Viva	Total
5	15	10	30

List of Experiments

S.No.	Name of Experiment
1	To develop a Simple Android Application that uses GUI components, Font and Colors
2	To develop a Simple Android Application that uses Layout Managers and Event Listeners.
3	Design simple GUI application with activity and intents e.g. calculator
4	Develop an application that makes use of RSS Feed.
5	Write an application that draws basic graphical primitives on the screen
6	Create an android app for database creation using SQlite Database
7	Develop a native application that uses GPS location information
8	Implement an application that writes data on the SD card.
9	Design a StopWatch application
10	Create an application to play a video in android

Introduction to Android

Android Studio is a new and fully integrated development environment, which has been recently launched by Google for the Android operating system. It has been designed to provide new tools for app development and to provide an alternative to Eclipse, currently the most widely used IDE. When you begin a new project in Android studio, the project's structure will appear with almost all the files held within the SDK directory, this switch to a Gradle based management system offers an even greater flexibility to the build process.

Android Studio allows you to see any visual changes you make to your app in real-time, and you can also see how it will look on a number of different Android devices, each with different configurations and resolutions, simultaneously. Another feature in Android Studio are the new tools for the packing and labelling of code. These let you keep on top of your project when dealing with large amounts of code. The programme also uses a drag & drop system to move the components throughout the user interface.

The programme will also help you to localize your apps, giving you a visual way to keep programming while controlling the flow of the application.

What else does Android Studio offer?

- A robust and straight forward development environment.
- An easy way to test performance on other types of device.
- Wizards and templates for common elements found in all Android programming.
- A full-featured editor with lots of extra tools to speed up the development of your applications.

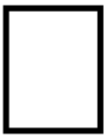
Android UI Controls: There are number of UI controls provided by Android that allow you to build the graphical user interface for your app.

- **TextView::** This control is used to display text to the user.
- **EditText::** EditText is a predefined subclass of TextView that includes rich editing capabilities.
- **AutoCompleteTextView**:: The AutoCompleteTextView is a view that is similar to EditText, except that it shows a list of completion suggestions automatically while the user is typing.
- **Button::** A push-button that can be pressed, or clicked, by the user to perform an action.
- **ImageButton::** AbsoluteLayout enables you to specify the exact location of its children.
- **CheckBox::** An on/off switch that can be toggled by the user. You should use checkboxes when presenting users with a group of selectable options that are not mutually exclusive.
- **ToggleButton::** An on/off button with a light indicator.
- **RadioButton::** The RadioButton has two states: either checked or unchecked.
- **Spinner**::A drop-down list that allows users to select one value from a set.

Android Event Handling : Events are a useful way to collect data about a user's interaction with interactive components of your app, like button presses or screen touch etc. The Android framework maintains an event queue into which events are placed as they occur and then each event is removed from the queue on a first-in, first-out (FIFO) basis. You can capture these events in your program and take appropriate action as per requirements. There are following three concepts related to Android Event Management:

Event Listeners: The View class is mainly involved in building up a Android GUI, same View class provides a number of Event Listeners. The Event Listener is the object that receives notification when an event happens.

Event Listeners Registration: Event Registration is the process by which an Event Handler gets registered with an Event Listener so that the handler is called when the Event Listener fires the event.



Event Handlers: When an event happens and we have registered and event listener for the event, the event listener calls the Event Handlers, which is the method that actually handles the event.

Event Listeners & Event Handlers

Event Handler	Event Listener & Description
onClick()	onClickListener() This is called when the user either clicks or touches or focuses upon any widget like button, text, image etc. You will use onClick() event handler to handle such event.
onLongClick()	onLongClickListener() This is called when the user either clicks or touches or focuses upon any widget like button, text, image etc. for one or more seconds. You will use onLongClick() event handler to handle such event.
onFocusChange()	onFocusChangeListener() This is called when the widget loses its focus i.e. user go away from the view item. You will use onFocusChange() event handler to handle such event.
onKey()	This is called when the user is focused on the item and presses or releases a hardware key on the device. You will use onKey() event handler to handle such event.

onTouch()	onTouchListener() This is called when the user presses the key, releases the key, or any movement gesture on the screen. You will use onTouch() event handler to handle such event.
onMenuItemClick()	onMenuItemClickListener() This is called when the user selects a menu item. You will use onMenuItemClick() event handler to handle such event.

Download Android Studio

Google provides Android Studio for the Windows, Mac OS X, and Linux platforms. You can download Android Studio from the Android Studio homepage, where you'll also find the traditional SDKs with Android Studio's command-line tools. Before downloading Android Studio, make sure your platform meets the following requirements:

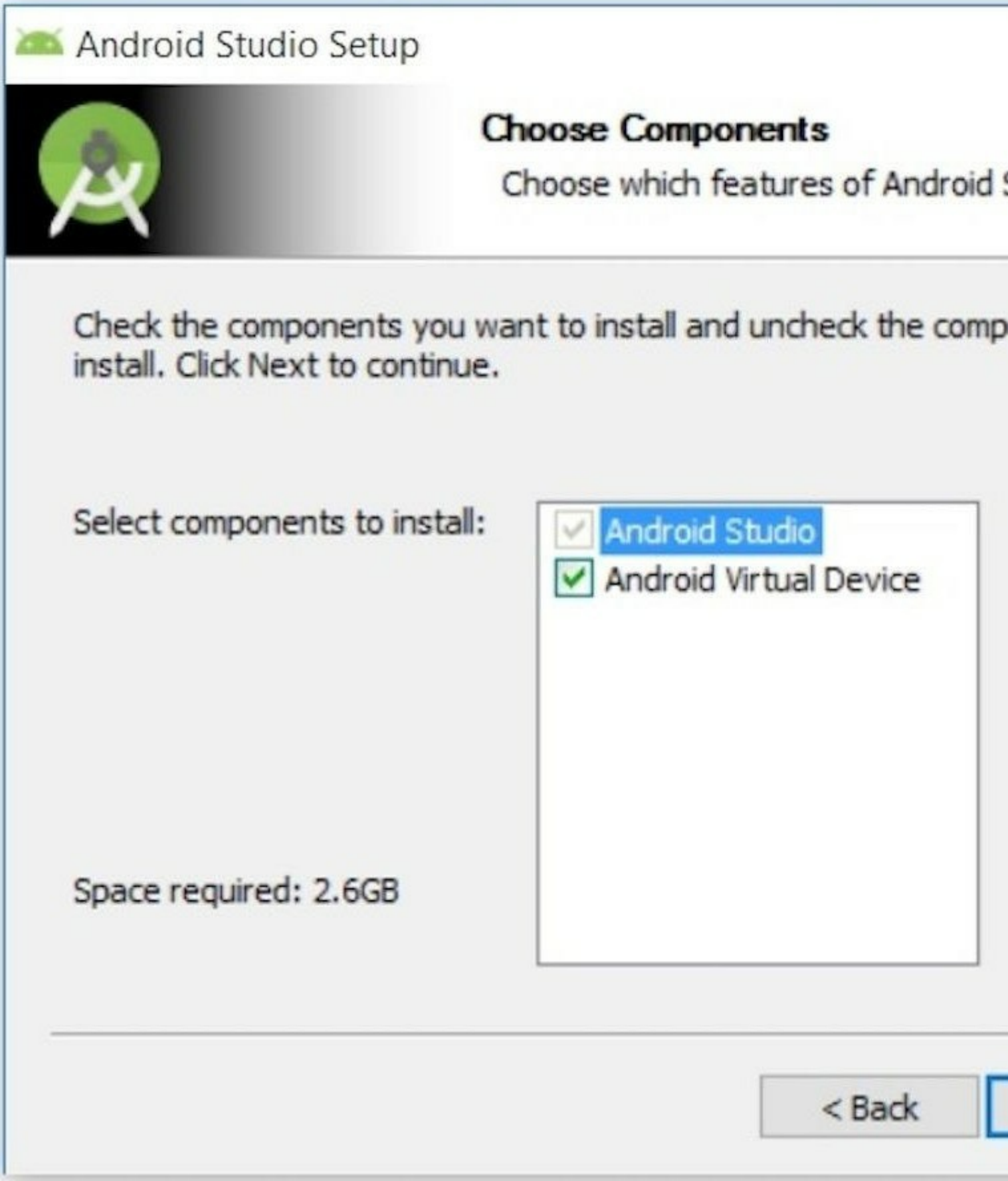
Windows requirements

-
    - Microsoft Windows 7/8/10 (32-bit or 64-bit)
    - 3 GB RAM minimum, 8 GB RAM recommended (plus 1 GB for the Android Emulator)
    - 2 GB of available disk space minimum, 4 GB recommended (500 MB for IDE plus 1.5 GB for Android SDK and emulator system image)
    - 1280 x 800 minimum screen resolution

Installing Android Studio on 64-bit Windows 10

Launch android-studio-ide-181.5056338-windows.exe to start the installation process.

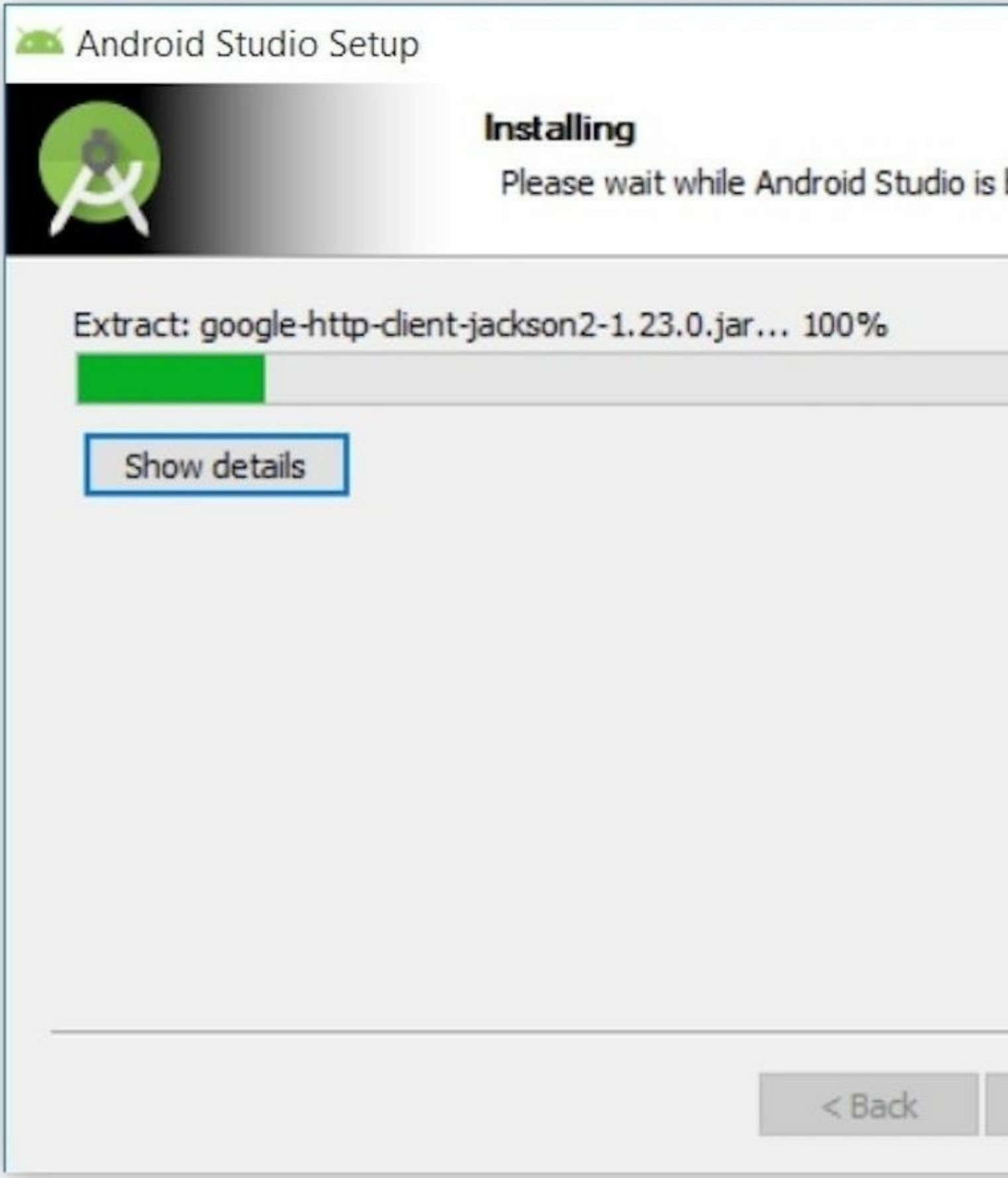
Clicking Next took to the following panel, which provides the option to decline installing an Android Virtual Device (AVD).



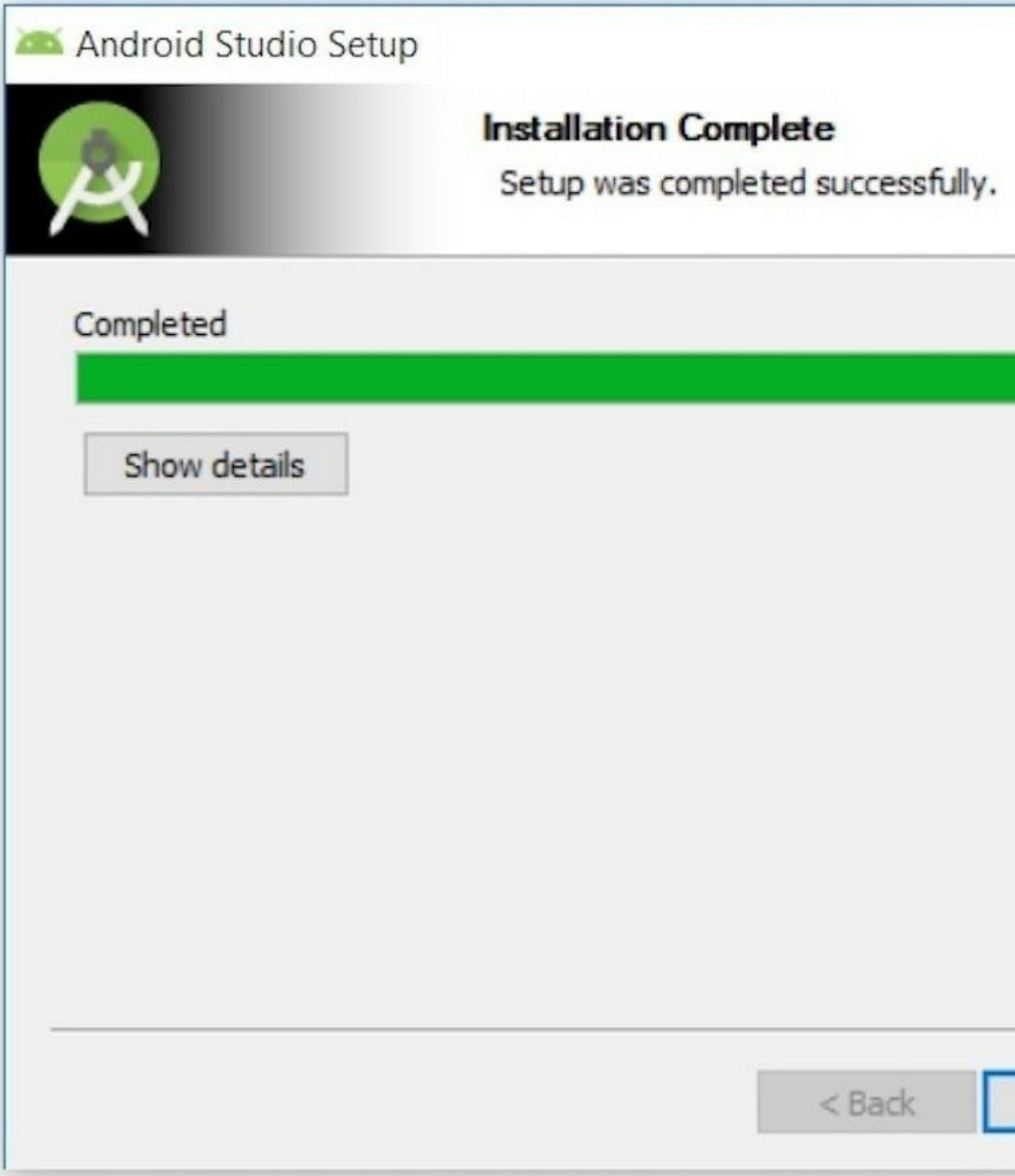
Keep the default settings. After clicking Next, Configuration Settings panel will be open, where you have to choose where to install Android Studio.

Keep the default installation location and click Next, this will open Choose Start Menu Folder panel.

Keep the default setting and click Install. The following Installing panel appeared:



Clicking Show details causes the names of files being installed and other activities to be displayed. When installation finished, the Installation Complete panel appeared.

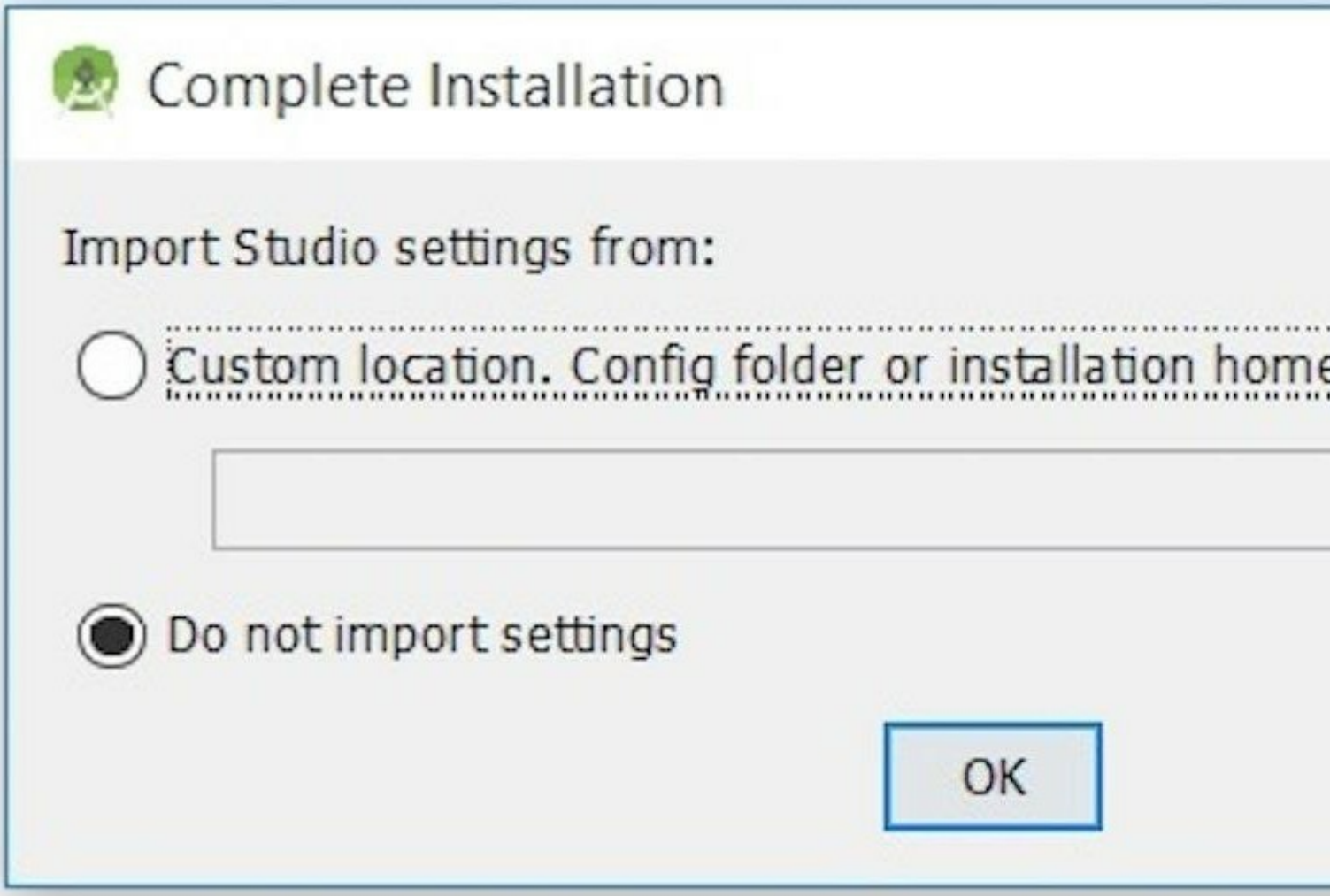


After clicking Next, the installer presented the Completing Android Studio Setup panel.

To complete the installation, I left the Start Android Studio box checked and clicked Finish.

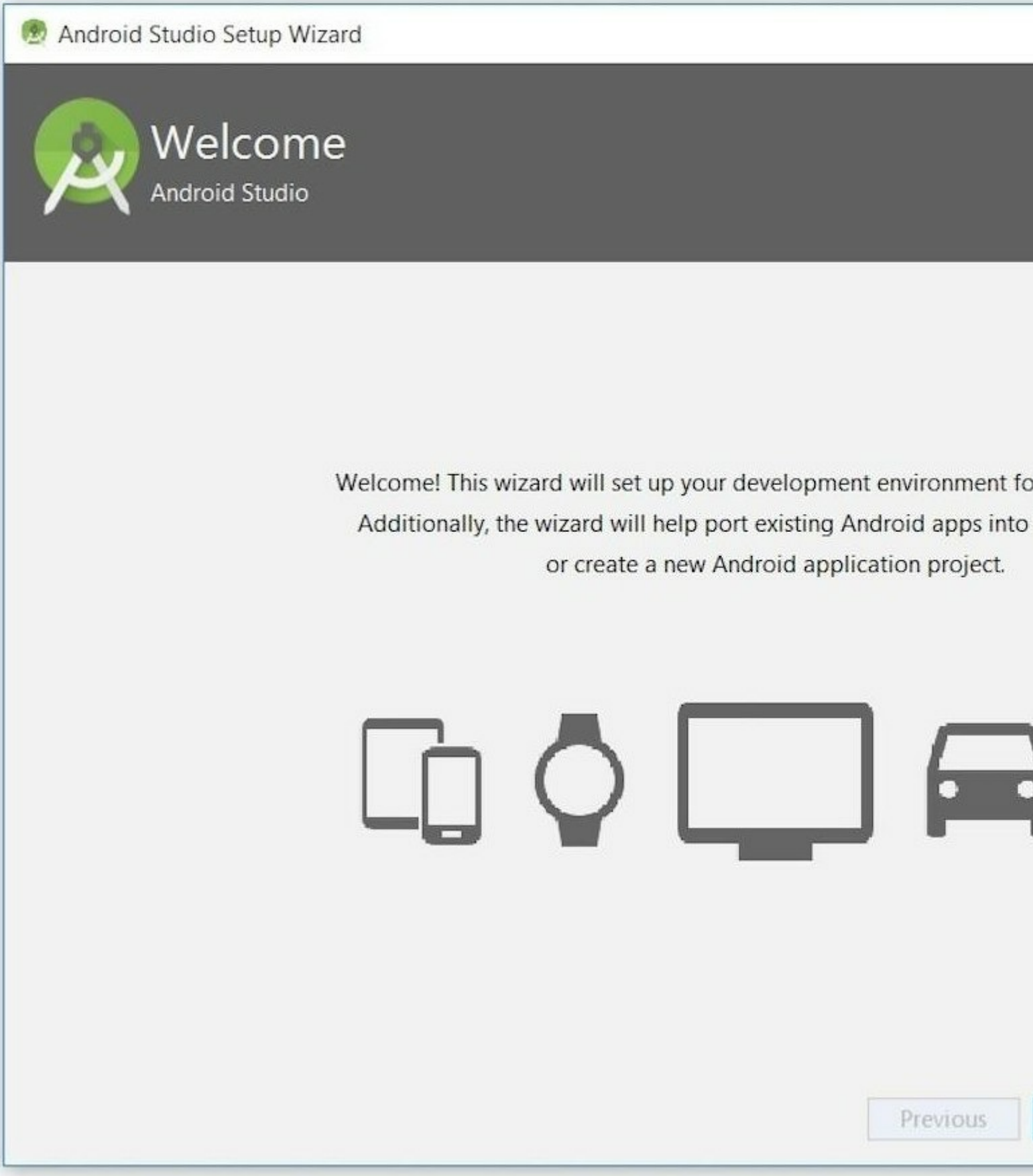
Running Android Studio

The first time Android Studio runs, it presents a Complete Installation dialog box that offers the option of importing settings from a previous installation.

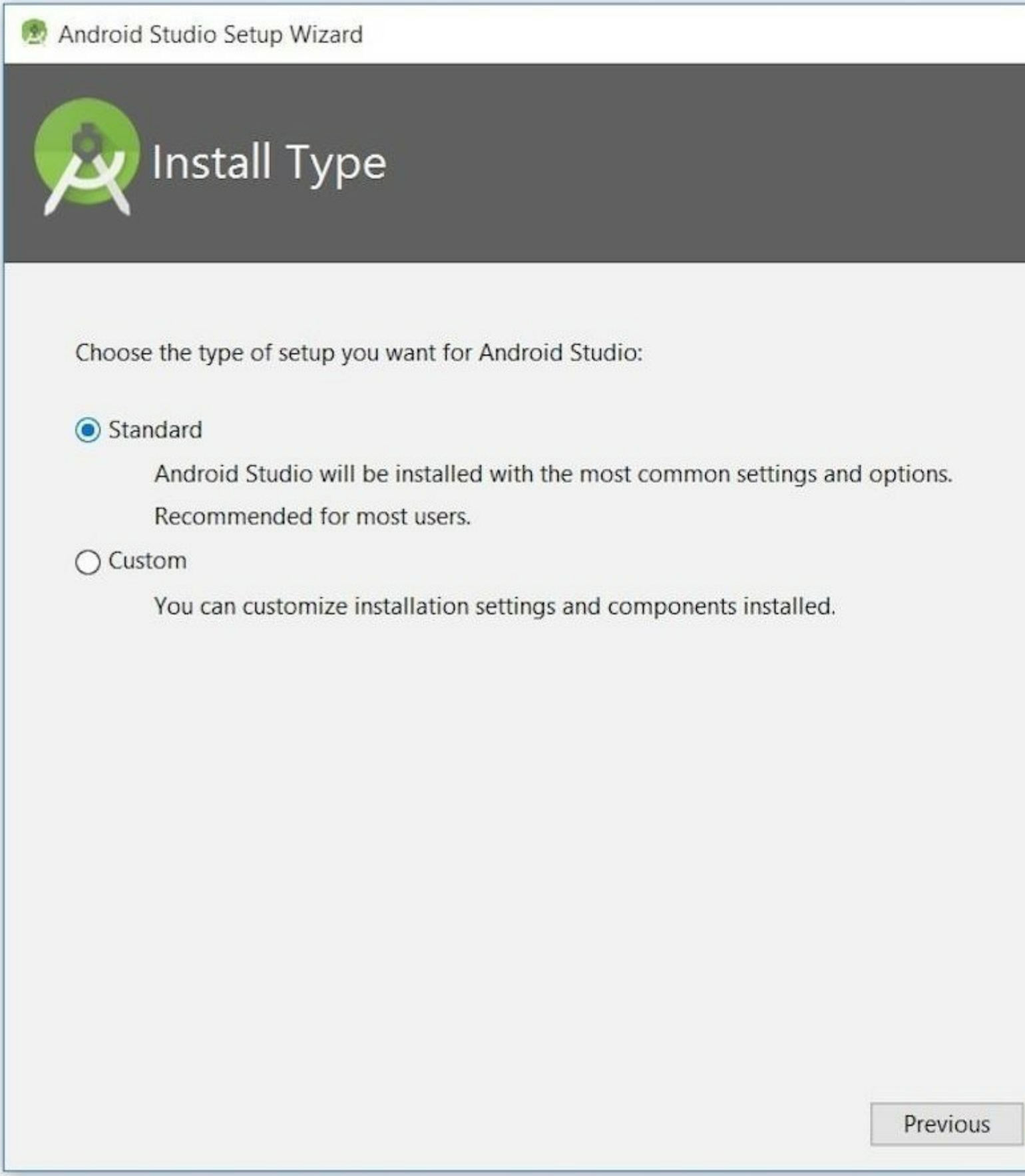


Choose not to import settings (the default selection) and clickOK, and the following splash screen will be displayed:

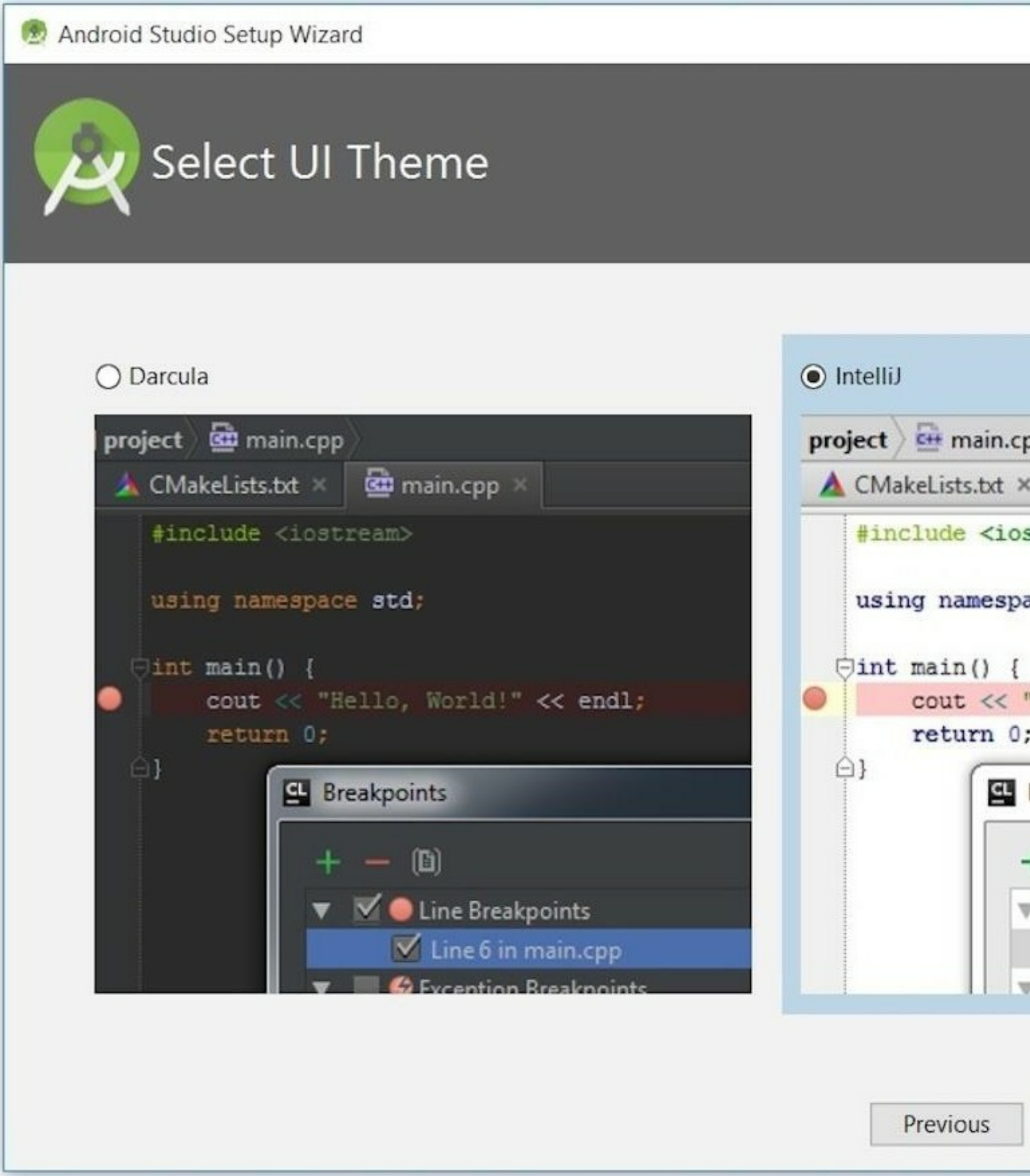
At this point, Android Studio presented the following Android Studio Setup Wizard dialog box:



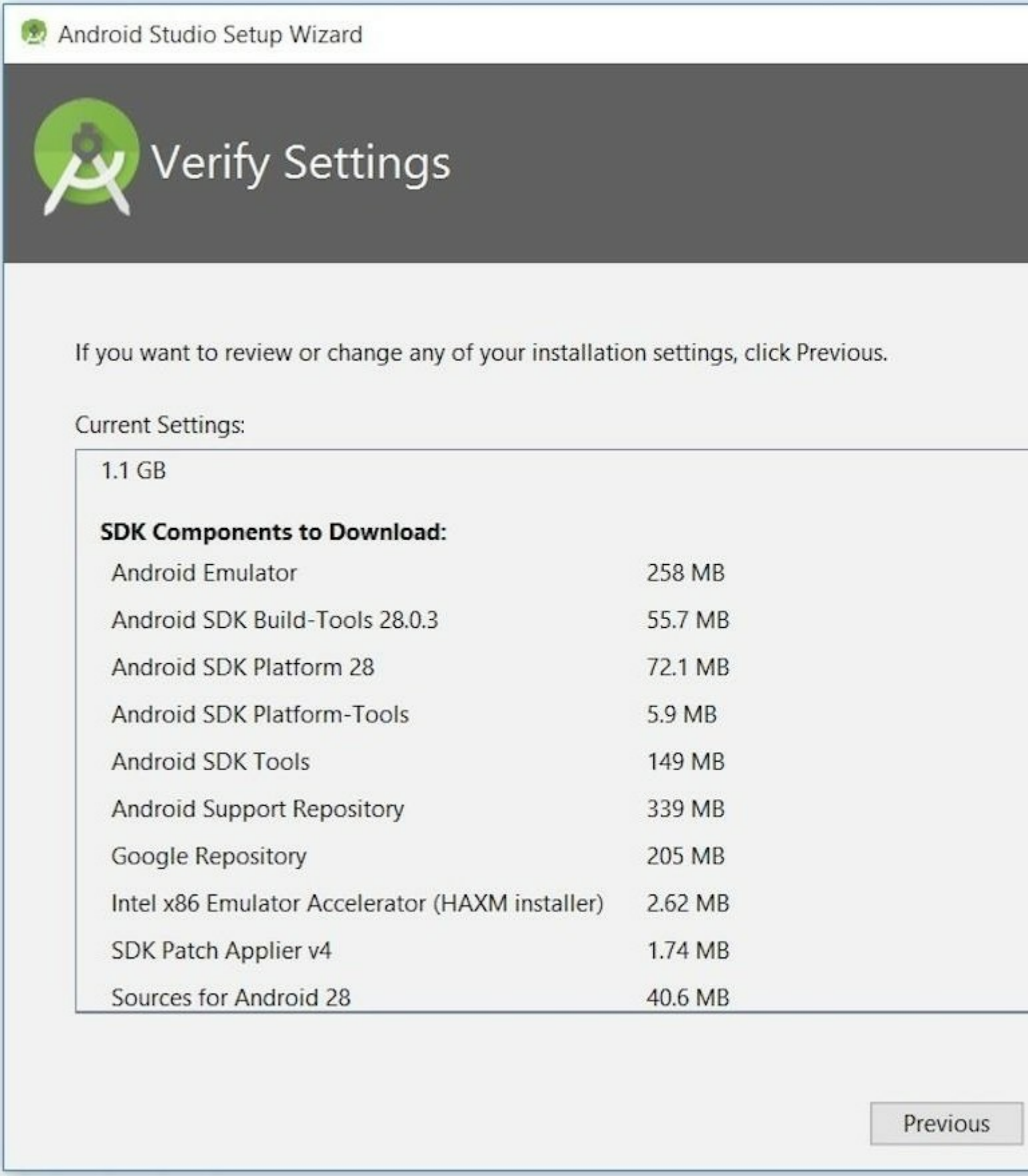
Click Next, and the wizard invited you to select an installation type. Keep the default standard setting.



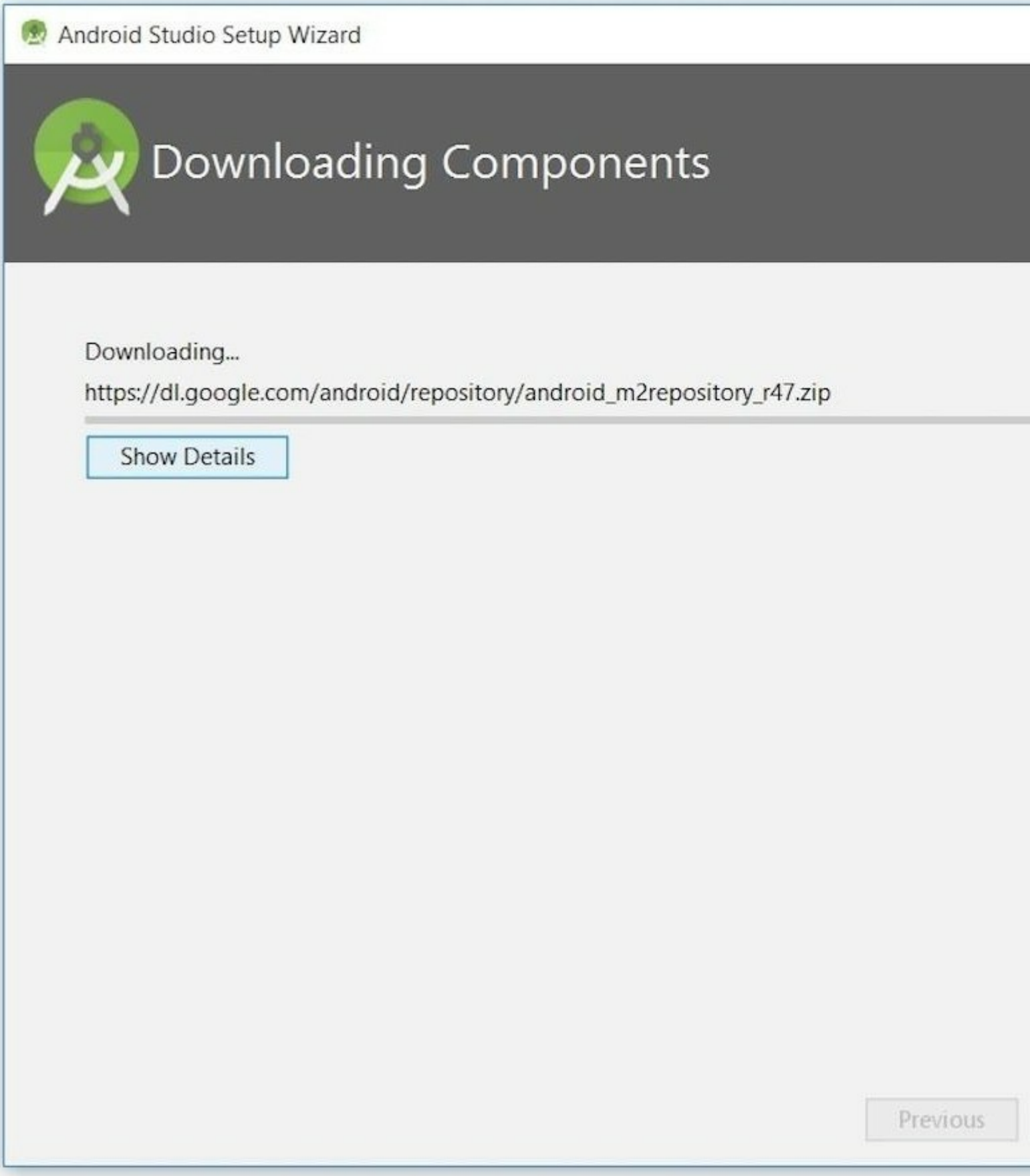
You will be directed to the page to choose a user interface theme.



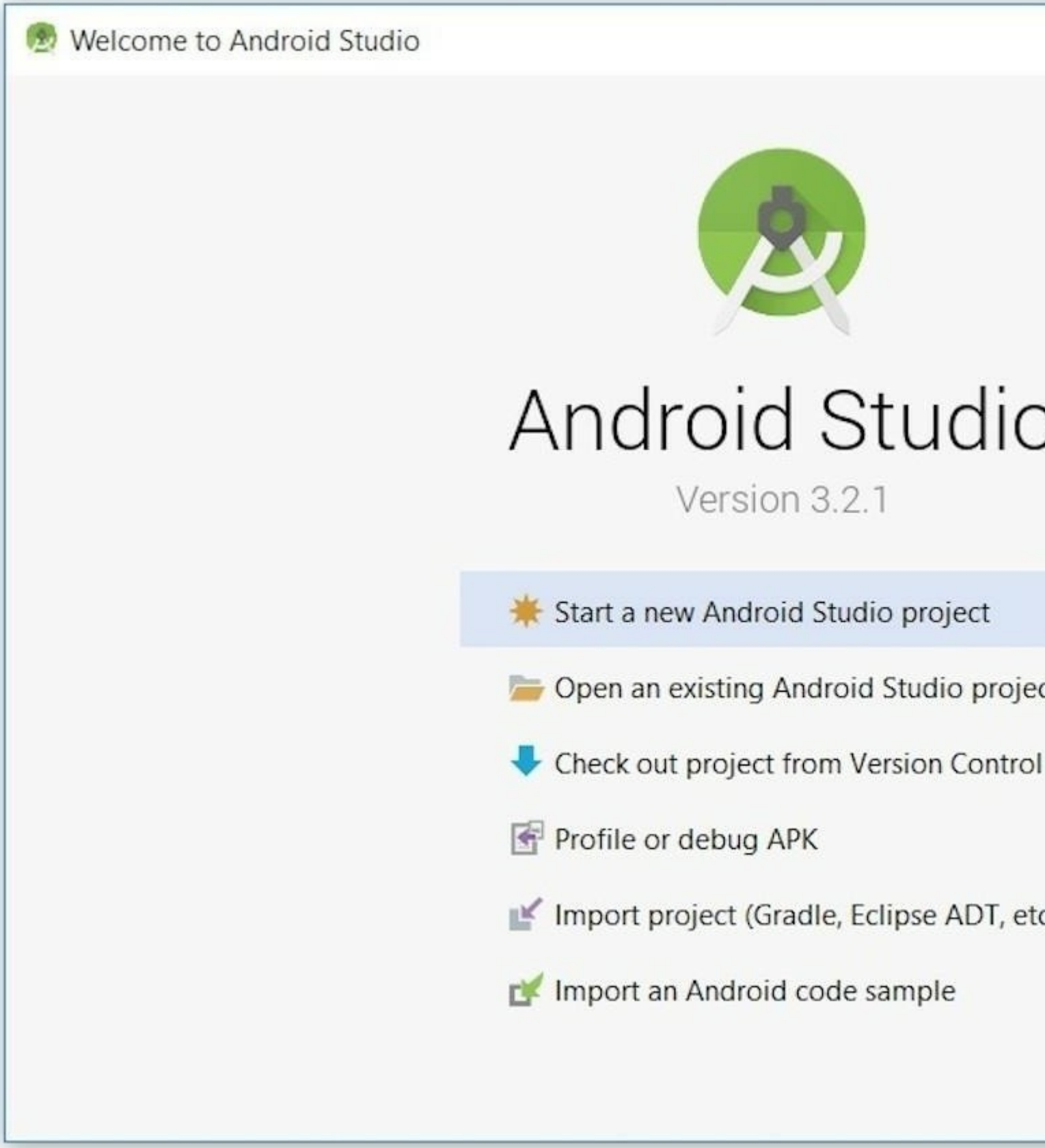
Keep the default IntelliJ setting and click Next. Android Studio next provided the opportunity to verify settings.



Click Finish and Android Studio began the process of downloading SDK components.

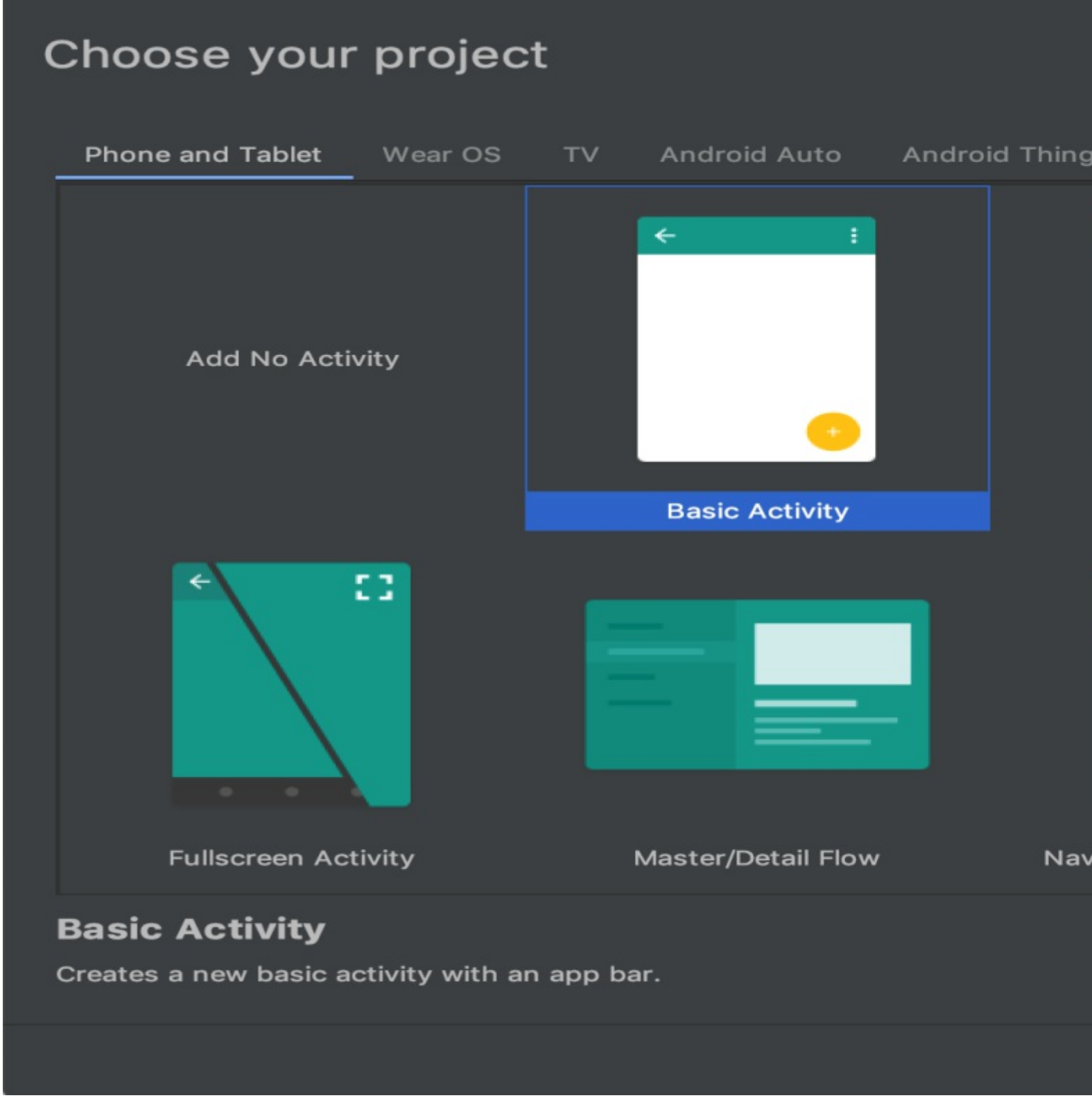


Finally, Click Finish to complete the wizard. The Welcome to Android Studio dialog box appeared.

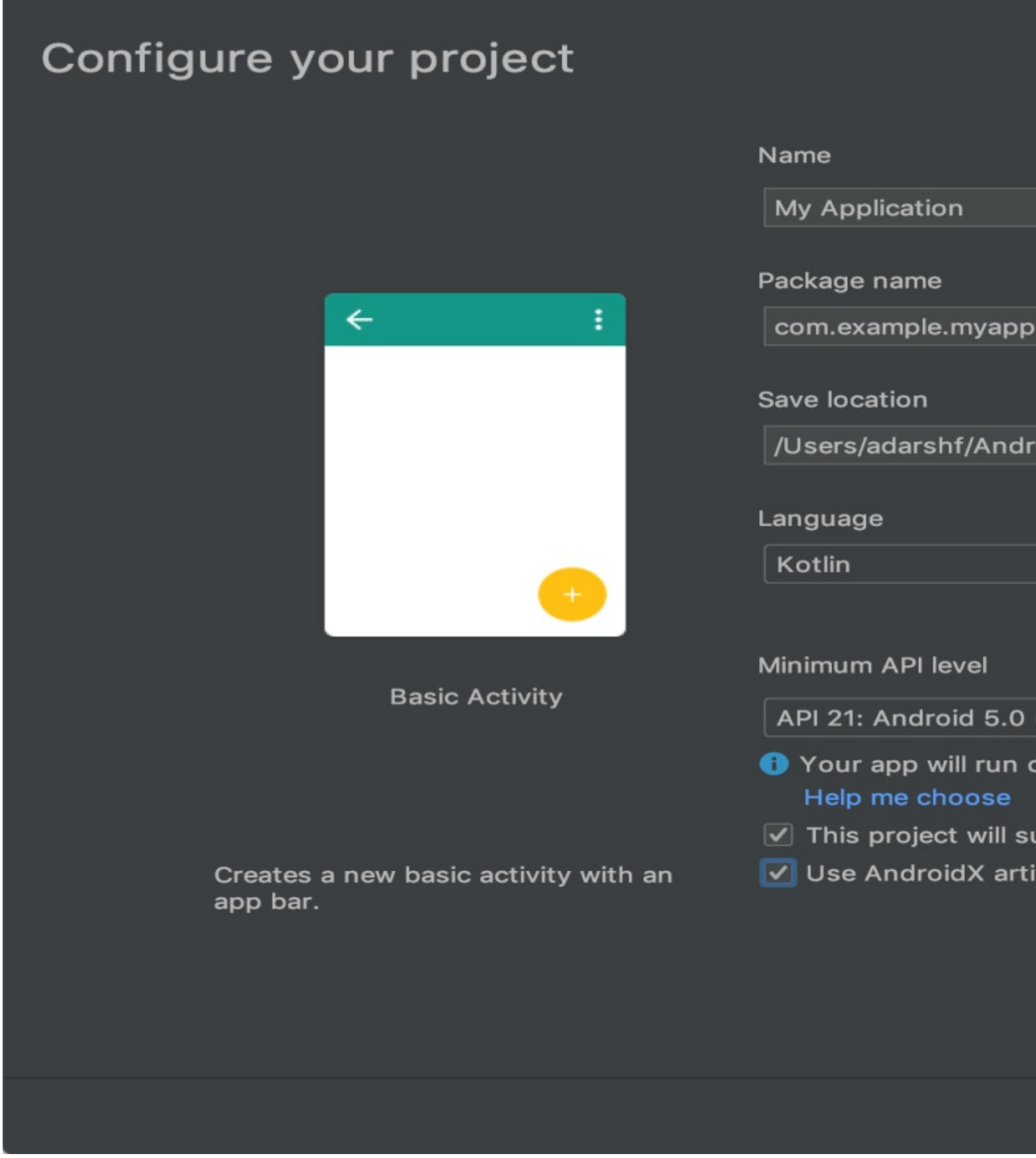


Starting a new project

click Start a new Android Studio project. Android Studio will respond with the Create New Project dialog box shown in Figure



After you make a selection, click Next.



Select language as Java, name your activity and package and click on finish. This will launch the basic activity project.

RUN Application

To run your application in android studio you have two options.

1. either use a real device and connect it via USB cable .and run the application on it
2. or create a AVD in android(Tools->AVD manager-> create virtual device) and run application on this emulator.

Experiment-1

Aim: To develop a Simple Android Application that uses GUI components, Font and Colors

Procedure:
Steps to Create the Application

1. Set Up Android Studio

Open Android Studio and create a new project.

Select Empty Activity.

Name the project, e.g., SimpleGUIApp, and choose the language as Java or Kotlin.

Set the Minimum SDK to API 21 (Android 5.0 Lollipop) or above.

2. Modify the Layout (XML)

In the res/layout/activity_main.xml file, define your GUI components, font styles, and colors.

Code for activity_main.xml:

```
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:orientation="vertical"
android:padding="16dp"
android:background="#E3F2FD">
<!-- Title TextView -->
<TextView
android:id="@+id/titleText"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="Welcome to Simple GUI App"
android:textSize="24sp"
android:textColor="#0D47A1"
android:gravity="center"
android:padding="8dp"
android:fontFamily="@font/poppins_bold" />
<!-- EditText for User Input -->
<EditText
```

```
android:id="@+id/userInput"

android:layout_width="match_parent"

android:layout_height="wrap_content"

android:hint="Enter your name"

android:textSize="18sp"

android:padding="10dp"

android:layout_marginTop="16dp"

android:background="#FFFFFF" />

<!-- Button -->

<Button

android:id="@+id/submitButton"

android:layout_width="match_parent"

android:layout_height="wrap_content"

android:text="Submit"

android:textSize="18sp"

android:layout_marginTop="16dp"

android:background="#1565C0"

android:textColor="#FFFFFF" />

<!-- Result TextView -->

<TextView

android:id="@+id/resultText"

android:layout_width="match_parent"

android:layout_height="wrap_content"

android:text=""

android:textSize="20sp"

android:textColor="#4CAF50"

android:gravity="center"

android:layout_marginTop="16dp" />

</LinearLayout>
```

3. Add Custom Font

Create a font directory in the res folder:

Right-click on res > New > Android Resource Directory.

Select Resource Type as font, and click OK.

Download a font (e.g., Poppins) and add the .ttf or .otf font file into the res/font directory.

Example: Place a file named poppins_bold.ttf in the res/font directory.

4. Define Colors

Modify res/values/colors.xml to define custom colors:

<resources>

<color name="primary_color">#1565C0</color>

<color name="background_color">#E3F2FD</color>

<color name="text_color">#0D47A1</color>

<color name="success_color">#4CAF50</color>

</resources>

5. Implement Logic in MainActivity

Write the Java or Kotlin code to handle the button click and update the TextView with the user's input.

Code for MainActivity.java:

package com.example.simpleguiapp;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity_main);

// Initialize GUI components

EditText userInput = findViewById(R.id.userInput);

Button submitButton = findViewById(R.id.submitButton);

```
TextView resultText = findViewById(R.id.resultText);

// Set button click listener

submitButton.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

String name = userInput.getText().toString().trim();

if (!name.isEmpty()) {

resultText.setText("Hello, " + name + "!");

} else {

resultText.setText("Please enter your name.");

}

}

});

}

}
```

6. Run the Application

Connect an Android device or use an emulator.

Click on the Run button in Android Studio.

The app should display a text field, a button, and a result area. Enter your name and click the button to see the greeting.

7. Output

Background: Light blue.

Title: Displays "Welcome to Simple GUI App" in custom font.

EditText: Input field with a white background.

Button: Blue with white text.

Result: Displays a personalized message (e.g., "Hello, [Name]!").

Features Demonstrated

1. GUI components: TextView, EditText, Button.
2. Fonts: Custom font (Poppins).
3. Colors: Custom colors defined in colors.xml.
4. Interactivity: Handling button clicks to display a message.