



---

[CODE](#) > [ANDROID SDK](#)

# What Are Android Intents?

---

by [Chinedu Izuchukwu](#) 24 Aug 2017

Difficulty: Beginner Length: Medium Languages: English ▼

Android SDK



---

Intents are a fundamental topic for Android developers. It is impossible to build Android applications without coming in contact with intents. In this tutorial, I'll teach you about intents from A to Z.

## What Are Intents?

In a football match, teammates pass the ball around the field with the aim of sending it into the goal of their opponent. The ball is passed from the team's goalkeeper to their defenders. Next, it finds its way to the midfielders, and if things work out as planned, one of the strikers sends it into the net of the opponent. That's assuming the goalkeeper of the other side was not able to keep it away!

In Android, the ability to send messages around is made possible by the `Intent` object. With the help of intents, Android components can request functionality from other Android components. When you open up the Instagram app on your phone and use it to take a picture, you just made use of an intent. Intents also help communicate between parts of an app; the movement from one screen (activity) to another is made possible by intents.

Look at it this way: all components (applications and screens) of the Android device are isolated. The only way they communicate with each other is through intents.

## Starting Activities With Intents

As mentioned earlier, you can use intents to start different components: activities, services, and broadcast receivers.

To start an activity, you will make use of the method `startActivity(intent)`.

Here is a code snippet that demonstrates how to start another activity from an intent.

```
1 Intent numbersIntent = new Intent(MainActivity.this, NumbersActivity.class);
2
3 startActivity(numbersIntent);
```

First, we create a new `Intent` object and pass it the `NumbersActivity` class. Then we start a new activity using that intent.

## Types of Intents

Android supports two types of intents: explicit and implicit. When an application defines its target component in an intent, that it is an explicit intent. When the application does not name a target component, that it is an implicit intent.

### Explicit Intent Example

The code snippet of code above is an example of explicit intent. Have a look at it again.

```
1 Intent numbersIntent = new Intent(MainActivity.this, NumbersActivity.class);
2
3 startActivity(numbersIntent);
```

Here, `NumbersActivity` is the target component from our `MainActivity`. This means that `NumbersActivity` is the defined component that will be called by the Android system. It is important to note (as in the example above), that explicit intents are typically used within an application, because that gives the developer the most control over which class will be launched.

### Implicit Intent Example

Here's an implicit intent:

```
1 Intent intent = new Intent(Intent.ACTION_VIEW, Uri.parse("http://www.tutsplus.com"));
2 startActivity(intent);
```

If you have the above code in your codebase, your application can start a browser component for a certain URL via an intent. But how does the Android system identify the components which can react to a certain intent?

A component can be registered via an *intent filter* for a specific action. Intent filters can be registered for components statically in the **AndroidManifest.xml**. Here is an example that registers a component as a web viewer:

```
1 <activity android:name=".BrowserActivity"
2         android:label="@string/app_name">
3     <intent-filter>
4         <action android:name="android.intent.action.VIEW" />
5         <category android:name="android.intent.category.DEFAULT" />
6         <data android:scheme="http" />
7     </intent-filter>
8 </activity>
```

## Using Intents in an App

Let's write some code to see how it works out. In this section, you'll build a tiny app to try out both styles of intent. The app will have a little form to enter a first name and last name. When the **Submit** button is clicked, both of the entered values will be passed to another activity. There will also be a button to launch a browser of your choice. The chosen browser will open up <http://code.tutsplus.com>.

Open up Android Studio and generate your `MainActivity`. You can set the name of the package to **com.tutsplus.code.android.droidintent**.

Your `MainActivity` will start with some imports and the class declaration:

```
01 package com.tutsplus.code.android.droidintent;
02
03 import android.content.Intent;
04 import android.net.Uri;
05 import android.support.v7.app.AppCompatActivity;
06 import android.os.Bundle;
07 import android.view.View;
08 import android.widget.Button;
09 import android.widget.EditText;
10
11 public class MainActivity extends AppCompatActivity {
12
13
14 }
```

Then you'll override the `onCreate()` method to initialize the activity with any saved state and the activity layout (we'll create this later).

```
1 @Override
2 protected void onCreate(Bundle savedInstanceState) {
3     super.onCreate(savedInstanceState);
4     setContentView(R.layout.activity_main);
5
6     // button handlers go here
7
8 }
```

Next you'll get references to each of the buttons defined in the layout and attach a click listener to each of them.

```
01 final Button submitButton = (Button) findViewById(R.id.submit_button);
02
03 submitButton.setOnClickListener(new View.OnClickListener() {
04     @Override
05     public void onClick(View v) {
06         EditText firstName = (EditText) findViewById(R.id.first_name_text);
07         EditText lastName = (EditText) findViewById(R.id.last_name_text);
08
09         String firstNameString = firstName.getText().toString();
10         String lastNameString = lastName.getText().toString();
11
12         Intent submitIntent = new Intent(MainActivity.this, ShowActivity.class);
13         submitIntent.putExtra("firstNameString", firstNameString);
14         submitIntent.putExtra("lastNameString", lastNameString);
15         startActivity(submitIntent);
16     }
17 });
```

For the **Submit** button, you set an `OnClickListener` to trigger an action whenever the button is clicked. When a click occurs, we grab the first and last name from the view, and send them to the next activity: `ShowActivity`. The target component is explicitly defined in the intent, making this an example of explicit intent.

```

01 final Button browserButton = (Button) findViewById(R.id.browser_button);
02
03 browserButton.setOnClickListener(new View.OnClickListener() {
04     @Override
05     public void onClick(View v) {
06
07         Intent intent = new Intent(Intent.ACTION_VIEW, Uri.parse("https://code.tutsplus.com"));
08         startActivity(intent);
09     }
10 });

```

For the browser button, the `OnClickListener` will create a new intent to launch any application that matches the filter: an `ACTION_VIEW` which should handle a web URL. In other words, it launches a web browser. If you have more than one browser application installed on your device, you will be asked to select one to perform the action of opening the web site. This is an example of an implicit intent.

## MainActivity Layout

The layout for `MainActivity` will be very simple for the purpose of this tutorial.

```

01 <?xml version="1.0" encoding="utf-8"?>
02 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
03     xmlns:app="http://schemas.android.com/apk/res-auto"
04     xmlns:tools="http://schemas.android.com/tools"
05     android:layout_width="match_parent"
06     android:orientation="vertical"
07     android:layout_margin="16dp"
08     android:layout_height="match_parent"
09     tools:context="com.tutsplus.code.android.droidintent.MainActivity">
10
11     <TextView
12         android:id="@+id/first_name_view"
13         android:layout_width="wrap_content"
14         android:layout_height="wrap_content"
15         android:text="Enter first name" />
16
17     <EditText
18         android:id="@+id/first_name_text"
19         android:padding="8dp"
20         android:layout_width="200dp"
21         android:layout_height="wrap_content" />
22
23     <TextView
24         android:id="@+id/last_name_view"
25         android:layout_width="wrap_content"
26         android:layout_height="wrap_content"
27         android:text="Enter last name" />
28
29     <EditText
30         android:id="@+id/last_name_text"
31         android:layout_height="wrap_content"
32         android:layout_width="200dp"
33         android:padding="8dp" />
34
35     <Button
36         android:id="@+id/submit_button"
37         android:layout_width="wrap_content"
38         android:layout_height="wrap_content"
39         android:text="Submit" />
40
41     <Button

```

```

42         android:id="@+id/browser_button"
43         android:layout_width="wrap_content"
44         android:layout_height="wrap_content"
45         android:layout_marginTop="50dp"
46         android:layout_marginLeft="100dp"
47         android:text="Open Browser"/>
48
49     </LinearLayout>

```

Here you have two `TextView` and two `EditText` indicating First Name and Last Name respectively. There is also a button to submit the names, and another to launch your browser.

## Create the `ShowActivity`

To complete our app, we need to create an activity to handle the explicit intent defined above. Create a new activity called `ShowActivity`. This is the activity where the result of entering the first name and last name will be shown. Here is what it should look like:

```

01 package com.tutspus.code.android.droidintent;
02
03 import android.support.v7.app.AppCompatActivity;
04 import android.os.Bundle;
05 import android.widget.TextView;
06
07 public class ShowActivity extends AppCompatActivity {
08
09     @Override
10     protected void onCreate(Bundle savedInstanceState) {
11         super.onCreate(savedInstanceState);
12         setContentView(R.layout.activity_show);
13
14         Bundle extras = getIntent().getExtras();
15         String inputFirstName = extras.getString("firstNameString");
16         String inputLastName = extras.getString("lastNameString");
17
18         TextView showFirstName = (TextView) findViewById(R.id.show_first_name);
19         TextView showLastName = (TextView) findViewById(R.id.show_last_name);
20
21         showFirstName.setText(inputFirstName);
22         showLastName.setText(inputLastName);
23     }
24 }

```

In this activity, you start by getting the strings passed from the `MainActivity`. You can get a reference to the intent that triggered the launch of this activity with the `getIntent()` function. Then you can access the strings that were passed to it using `getExtras().getString()`. Finally, after getting the `TextView` instances from the layout, you display the values you obtained.

## ShowActivity Layout

The layout for this activity is simple:

```

01 <?xml version="1.0" encoding="utf-8"?>
02 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
03     xmlns:app="http://schemas.android.com/apk/res-auto"
04     xmlns:tools="http://schemas.android.com/tools"
05     android:layout_width="match_parent"
06     android:layout_height="match_parent"
07     android:layout_margin="16dp"
08     android:orientation="vertical"
09     tools:context="com.tutspus.code.android.droidintent.ShowActivity">
10
11     <TextView

```

```
12         android:layout_width="wrap_content"
13         android:layout_height="wrap_content"
14         android:text="First Name:"/>
15
16     <TextView
17         android:id="@+id/show_first_name"
18         android:layout_width="wrap_content"
19         android:layout_height="wrap_content" />
20
21     <TextView
22         android:layout_width="wrap_content"
23         android:layout_height="wrap_content"
24         android:text="Last Name:"/>
25
26     <TextView
27         android:id="@+id/show_last_name"
28         android:layout_width="wrap_content"
29         android:layout_height="wrap_content" />
30
31 </LinearLayout>
```

## Testing the App

Now you can build your app and try it out on your Android device!

## Passing Data Using Bundles

You can also make use of Bundles when passing data via intents.

The `Bundle` class allows you store complex data and supports data types such as strings, chars, boolean, integer and so on. Here is an example of how part of **MainActivity.java** would look if you used `Bundle`.

```
01 // Creating Bundle object
02 Bundle b = new Bundle();
03
04 // Storing data into bundle
05 b.putString("firstNameString", firstNameString);
06 b.putString("lastNameString", lastNameString);
07
08 // Creating Intent object
09 Intent submitIntent = new Intent(MainActivity.this, ShowActivity.class);
10
11 // Storing bundle object into intent
12 submitIntent.putExtra(b);
13 startActivity(submitIntent);
```

## Conclusion

In this tutorial, we got a brief introduction to using intents to create activities in Android. We looked at the difference between explicit and implicit intents, and coded a simple example that used each type.

You can read more about intents in the [Android documentation](#). Understanding how they work is very important. As you build more apps, you will encounter lots of different kinds of Intents and ways of using them.

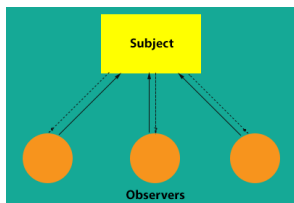
And in the meantime, check out some of our other posts on Android app development!



ANDROID SDK

## Introduction to Android Architecture Components

Tin Megali



JAVA

## Android Design Patterns: The Observer Pattern

Chike Mgbemena



ANDROID

## Android From Scratch: Building Your First Android Application

Gianluca Segato

Advertisement



Chinedu Izuchukwu  
Lagos, Nigeria

I enjoy writing code and passing knowledge. When not doing either of them, I watch movies.

## Weekly email summary

Subscribe below and we'll send you a weekly email summary of all new Code tutorials. Never miss out on learning about the next big thing.

Update me weekly

Advertisement

[View on GitHub](#)

### Translations

Envato Tuts+ tutorials are translated into other languages by our community members—you can be involved too!

[Translate this post](#)

Powered by  native



0 Comments


Tuts+ Hub

1 Login

Recommend 2

Share

Sort by Best



Start the discussion...

LOG IN WITH

OR SIGN UP WITH DISQUS ?

Name

Be the first to comment.

Subscribe


Add Disqus to your siteAdd DisqusAdd

Privacy

Advertisement

QUICK LINKS - Explore popular categories

ENVATO TUTS+	+
JOIN OUR COMMUNITY	+
HELP	+



tuts+

25,695

Tutorials

1,117

Courses

22,209

Translations

