



3.2 Android Intents and Intent Filters

[TOC](#)[3.1 Android applications](#)[3.3 Android Activities](#)

We already know what Android intent is and the types of intents as well. Android Intent is used for the following activities:

1. Activity is started.
2. Service is started.
3. Broadcast is delivered.

Android Intent basically facilitates the most important phenomenon of Object oriented programming paradigm called **Late Binding**. If you are not aware of what these two terms are then don't worry. **Object oriented programming** is a type of programming where basic point of doing anything starts from object. **Object** is anything which has a value and can be referenced by an identifier in memory of computer. Thus object has properties. It can be anything like might be a variable or a function. Anyways, the object is not our primary concern now. Object Oriented Programming (OOP) works with objects which have attributes and related procedures. For example, if you are a student then for a student object we will have attributes as name, age, address, etc, whereas procedures might include reading, writing, programming, etc. To make it simple let us check out the representation of object diagrammatically.



Figure Student Object

Object oriented programming is simply an approach of designing functions around data so that data can be secured from unauthorized access and hence elevates and eases programming freak points. Late Binding is one of the characteristics of object oriented programming. **Late Binding** is a process which implements polymorphism. When you execute your program, the declarations are resolved at runtime. They are found by their names. They correspond to a specific type. There is no compile time checking in this case.

Intent facilitates this binding in case of android apps which are again nothing but a few lines of code intelligently coded to get customer satisfaction. Intent can be thought of as gum between different activities. In an application we can have one or more activities. In the later case, these intents stick different activities together. It launches one activity from other activity. They act like data structure. They contain a summary of actions which are to be performed.

3.2.1 Structure of Android Intent

Android Intent generally consists of two different portions of information. They have primary and secondary attributes. Attributes are as follow:

Table Attributes of Intent

Serial Number	Attribute Name	Attribute Type	Description
1	Action	Primary	Operation is done on this data. It is expressed as Uri
2	Data	Primary	It contains information about the usual action it has to perform.
3	Type	Secondary	If set then we force the intent to be an explicit type. Type of intent is deduced from data.
4	Category	Secondary	Additional information about the basic action to be informed is given.
5	Component	Secondary	Name of a component class is specified. If set, other attributes are not evaluated and component matching this is explicitly used. In short other attributes becomes additional then.
6	Extras	Secondary	It provides extended information to the component.

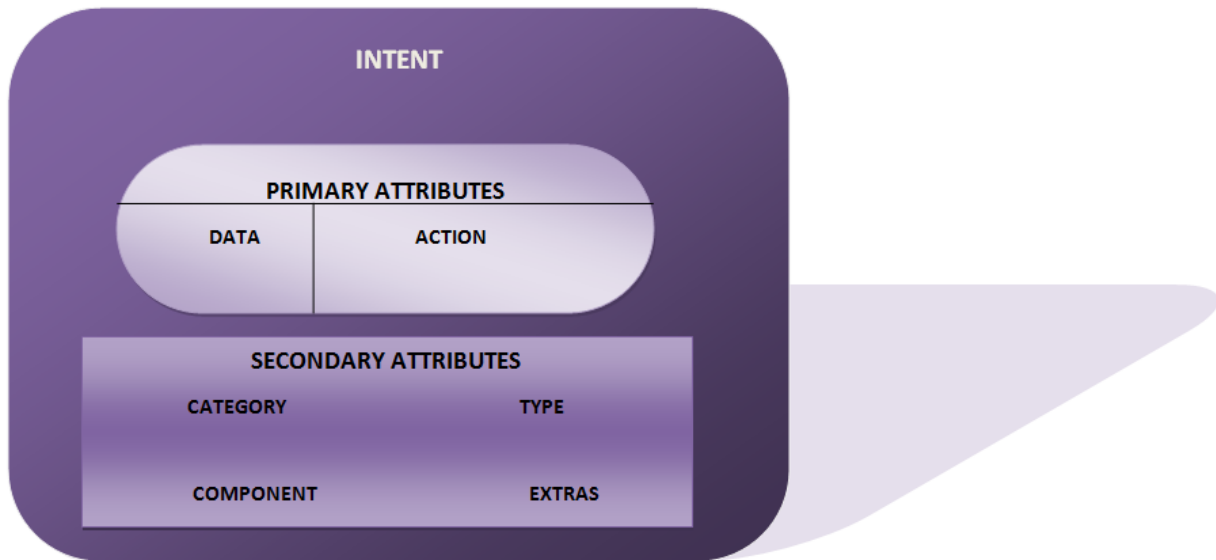


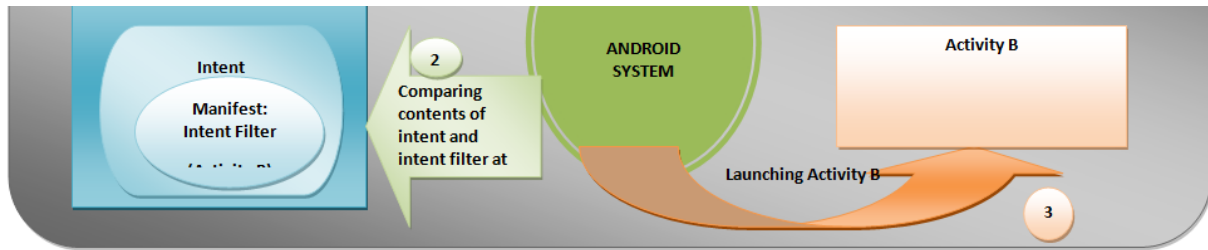
Figure Structure of an intent

3.2.2 Android Intent Filter

An Android **intent filter** is an expression. It is expressed in the Android application's manifest file. It defines the type of intent the component needs to receive. For an implicit intent, intent filter is required. If there is no intent filter for an activity, then only an explicit intent can initiate it.

For security purposes, the intent filter should not be used for service. Service may start without notifying the user and runs in the background.

Diagrammatically we shall understand how it works. Android architecture in your device will find the component declared in intent filter and search for it in entire application. When found, it starts respective activity and performs the appropriate action.



3.2.3 Work of Intent

We already know about implicit intent and explicit intent respectively. With an explicit intent, the app is started directly. With implicit intents, intent filter is used to filter out the component to start. Intents are used to perform three things mainly and they are as follows:

1. **Activity is started:** There is a method called **startActivity()**. When we pass intent to this method a new instance of that activity is started. Intent contains important information about the data and gives details about that activity which is to be started. Not only this, we can have result from other activity with the help of intent and methods like **startActivityForResult()** and **onActivityResult()** respectively. But that part we are going to discuss in the next section which is dedicated for activities..
2. **Service Activation:** Similarly, there is a method called **startService()**. When we pass intent to this method service is started. This intent gives details about the service which is to be started. If we want to get a result or want to have a server-client set up, then we can do it with a method called **bindService()**. This will be discussed in the section dedicated for services.
3. **Sending Broadcast:** There are three methods named **sendBroadcast()**, **sendOrderedBroadcast()** and **sendStickyBroadcast()**. Any broadcast can be delivered to apps by passing intent to one of these methods.

3.2.4 Example of Intent Filter in Android HelloWorld Application

We already have a running app called HelloWorld, let's find out the intent filter from HelloWorld's manifest file.

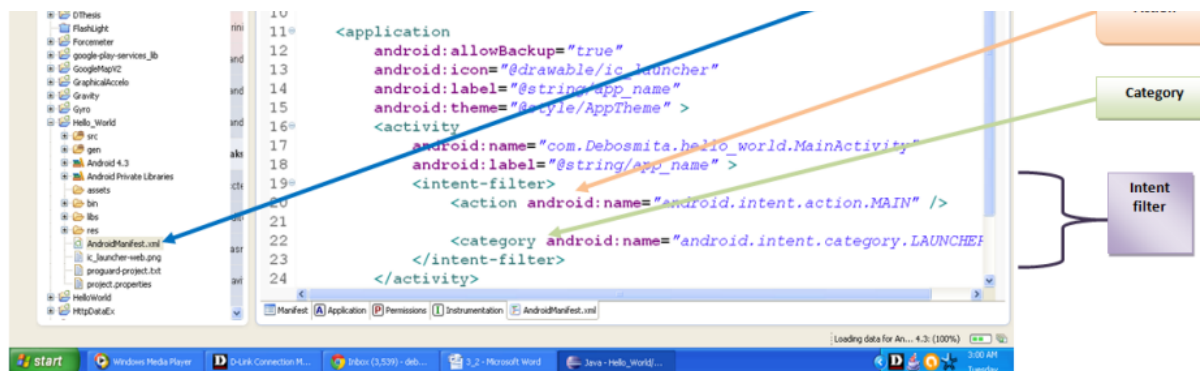


Figure Example of intent filter from HelloWorld App.

The above example shows a category and action. Action says standard MAIN action. This action is the entry point of the application. LAUCHER category says that in application launcher this entry point should be listed. It is very important to expertise in intents and passing intents. From the next topic, we are going to use intents.

3.2.5 Constructing Android Intent

As we know that android system identifies the component from intent. Thus intent object contains information such as name of the component or the category of the component. It also contains information for the recipient. It determines which action it has to perform when the intent is received.

- **Name of the component:** it is optional. It contains the exact name of the component which it has to initiate. It is a very strong requirement for the explicit intent as only that component will be initiated whose name is explicitly mentioned. For example, you want to send a courier then you have to mention the name of your best friend as writing only "To my best friend is not going to work". So here, Courier is the thing that you want your best friend (component) to read or do some action. Courier service is your explicit intent. Component name should be specified with a fully qualified name. The package name is to be mentioned. In the above figure, package name is indicated.

If name is not mentioned then intent becomes implicit.

- **Action of Intent:** It is a set of characters which specifies the general action of intent. Actions can be specified, i.e., you can create your own intent action.

```
static final String ACTION_DOCOUNT="com.Debosmita.action.DOCOUNT";
```

Figure Example of declaring intent



above statement, `android.intent.action.MAIN` is the name of the action.

- **Data:** A URI object is required. It is used to reference the data which is to be acted upon. It may reference the MIME type of data. Intent's action dictates the type of available data. If action is `ACTION_SEND` then the user must have some data to share. We already know what is meant by URI. Let us find out who is this MIME? Another friend which has killer impressive characteristics. Let's talk technically; the internet media type identifiers are called **MIME types**.

The Internet media type is an identifier which indicates the type of data in file on the internet. They are also called **Content-types**. Internet media types are also used as a set of rules to provide communication between different applications. Say, type audio has MIME type: `audio/mp4` for MP4 audio, `audio/mpeg` for MP3 or other MPEG audio, etc.

Now for intents, it is very important to keep clear of all the issues between Uri format and MIME type, especially in the case of methods. `setData()` method is used to set data for Uri and `setType()` to set type of MIME. Now both of them are like positive and negative forces which cancel out each other. If you want to use both of them, then we have another method called `setDataAndType()` to set data for Uri and type for MIME respectively. Great isn't it!!!

- **Category:** Most intents don't require a category. Additional information is given about the component. This component, which is going to handle the intent. There can be multiple categories for single intent. We have already discussed about them.
- **Extras:** These are the key-value pairs. They contain extra information for the fulfillment of the action requested in intent. `putExtra()` method is used to add extra data. A **Bundle** object can be created with additional data and introduced to intent with `putExtras()` method.
- **Flags:** They act as data about data, i.e., a metadata for intent. It may instruct the system about the nature of treatment to be given to an activity which is launched by this intent.

3.2.6 Resolution of an intent

There are three tests carried out to resolve intent. This is done on the basis of the three parameters of intent filters. They are compared with respect to the declaration in the app's manifest file. So the intent in component and intent filter of manifest files is examined and the basis of examination is action, category and data. We have following tests:

1. Intent type
2. Intent Category
3. Intent action.

Briefly let us check out these tests.

- **Action Test:** Zero or more elements are declared in an intent filter within `<action />` tag. The accepted intent actions are specified. To pass this test there should be a corresponding




Figure Example of action in intent filter of HelloWorld app

- **Category Test:** There can be zero or more `<category />` elements. For success every category specified in intent should match with the specification in filter. Let us consider our HelloWorld app again. One important thing to notice is that there can be categories in intent-filter which are not present in corresponding intent. It's perfectly alright. But the reverse is not true. There has to be a one to one matching for each category in intent and in intent filter respectively.

```
<intent-filter>
  <category
    android:name="android.intent.category.LAUNCHER" />
  ...
</intent-filter>
```

Figure Example of category in intent filter of HelloWorld

- **Data Test:** There can be zero or more elements regarding the specification of intent's data and its type. The `<data />` element can specify a URI format and a data type, i.e., MIME media type respectively. Intent is successful in the test which doesn't contain any data and there are no specifications for them in intent filter. Again, if there is a specification for MIME type and nothing for URI format, then intent should have corresponding MIME type and nothing for URI format respectively to pass in the test. the same rule applies if only URI format is specified and MIME isn't. An intent that contains both a URI and a MIME type passes the MIME type part of the test only if that type matches a type listed in the filter. It passes the URI part of the test either if its URI matches a URI in the filter. There is one more possibility of passing if it has content: or file: URI and the filter do not specify a URI.



Figure Resolution of intent

3.2.7 Pending Intent

PendingIntent is an object which acts like a housecoat over an intent object. It grants permission to the foreign application to use the covered intent as if it is executed from a process homed in the same app. For example, you might want to declare an intent which will be executed in the future at a specified time.

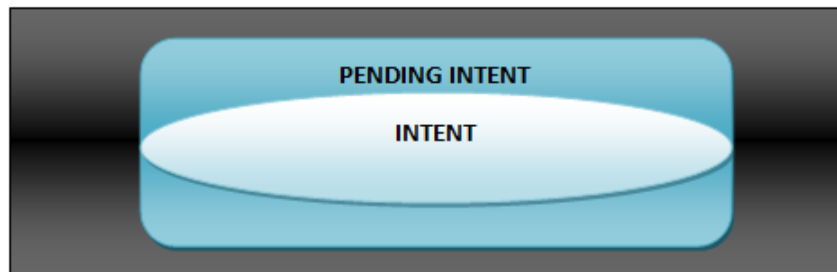


Figure Illustration of Pending Intent

[Technology](#)[Android Programming and Development Tutorial](#)[CHAPTER THREE: Application Components](#)[3.2 Android Intents and Intent Filters](#)[!\[\]\(d66ff64371a51729ac8c1cdaa685ba6f_img.jpg\)](#) [!\[\]\(0f31ebba7abcd47777e178db26f29705_img.jpg\)](#) [!\[\]\(63ea948177b1bcc486b2b76d20d5fb69_img.jpg\)](#) [!\[\]\(886f7dced1265a6d438eca0881817b40_img.jpg\) Login](#)

C
H
A
P
T
E
R
S

O
T
H
E
R

S
U
B
J
E
C
T
S