Intents, Intent filters, Broadcast

http://javat.in

Intent

- ❑ Three of the core components of an application — activities, services, and broadcast receivers — are activated through messages, called *intents*.
- ❑ An Intent object is passed to Context.startActivity() or Activity.startActivityForResult() to launch an activity or get an existing activity to do something new.
- ❑ An Intent object is passed to Context.startService() to initiate a service or deliver new instructions to an ongoing service.
- ❑ Intent objects passed to any of the broadcast are delivered to all interested broadcast receivers.

Intent Objects

- An Intent object is a bundle of information.
  It contains information of interest to the component that receives the intent.
- An Intent can contain following:
- **Component name:**
  - The name of the component that should handle the intent.
  - The component name is optional. If it is set, the Intent object is delivered to an instance of the designated class. If it is not set, Android uses other information in the Intent object to locate a suitable target.
  - The component name is set by setComponent(), setClass(), or setClassName() and read by getComponent().

Action

- A string naming the action to be performed — or, in the case of broadcast intents, the action that took place and is being reported.

Constant	Target Component	Action
ACTION_CALL	activity	Initiate a phone call.
ACTION_EDIT	activity	Display data for the user to edit.
ACTION_MAIN	activity	Start up as the initial activity of a task, with no data input and no returned output.
ACTION_BATTERY_LOW	broadcast receiver	A warning that the battery is low.

- The action in an Intent object is set by the setAction() method and read by getAction().

Data:

- The URI of the data to be acted on and the MIME type of that data.
- Different actions are paired with different kinds of data specifications.
  For example, if the action field is ACTION_EDIT, the data field would contain the URI of the document to be displayed for editing. If the action is ACTION_CALL, the data field would be a tel: URI with the number to call.

- The setData() method specifies data only as a URI, setType() specifies it only as a MIME type, and setDataAndType() specifies it as both a URI and a MIME type. The URI is read by getData() and the type by getType().

❑ **Category**

- A string containing additional information about the kind of component that should handle the intent.
- Any number of category descriptions can be placed in an Intent object.

Constant	Meaning
CATEGORY_BROWSABLE	The target activity can be safely invoked by the browser to display data referenced by a link
CATEGORY_GADGET	The activity can be embedded inside of another activity that hosts gadgets.
CATEGORY_HOME	The activity displays the home screen
CATEGORY_LAUNCHER	The activity can be the initial activity of a task and is listed in the top-level application launcher.

- ❑ **Extra**
  - Key-value pairs for additional information that should be delivered to the component handling the intent.
  - For example, an ACTION_TIMEZONE_CHANGED intent has a "time-zone" extra that identifies the new time zone, and ACTION_HEADSET_PLUG has a "state" extra indicating whether the headset is now plugged in or unplugged.
  - The Intent object has a series of put...() methods for inserting various types of extra data and a similar set of get...() methods for reading the data.

Intent Resolution

- Intents can be divided into two groups:
  - ***Explicit intents*** designate the target component by its name.
  - ***Implicit intents*** do not name a target. Implicit intents are often used to activate components in other applications.
- To resolve an Explicit Intent Android uses only the name of the Component.
- Android resolves Implicit Intent by comparing the contents of the Intent object to *intent filters*, structures associated with components that can potentially receive intents.
- Filters advertise the capabilities of a component and delimit the intents it can handle.

Intent Resolution

- A component with filters can receive both explicit and implicit intents.
- Only three aspects of an Intent object are consulted when the object is tested against an intent filter:
  - action
  - data (both URI and data type)
  - category
- The extras and flags play no part in resolving which component receives an intent.

Intent filters

- To inform the system which implicit intents they can handle, activities, services, and broadcast receivers can have one or more intent filters.
- An explicit intent is always delivered to its target, no matter what it contains; the filter is not consulted.
- But an implicit intent is delivered to a component only if it can pass through one of the component's filters.
- An intent filter is an instance of the IntentFilter class.
- A filter has fields that parallel the action, data, and category fields of an Intent object.
- An implicit intent is tested against the filter in all three areas.
- To be delivered to the component that owns the filter, it must pass all three tests. If it fails even one of them, the Android system won't deliver it to the component

Intent filters

Action test

- While an Intent object names just a single action, a filter may list more than one.
- The list cannot be empty; a filter must contain at least one <action> element, or it will block all intents.
- To pass this test, the action specified in the Intent object must match one of the actions listed in the filter.

```
<intent-filter . . . >
    <action android:name="com.example.project.SHOW_CURRENT" />
    <action android:name="com.example.project.SHOW_RECENT" />
    <action android:name="com.example.project.SHOW_PENDING" />
    . . .
</intent-filter>
```

❑ Category test

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

❑ Data test

```
<intent-filter . . . >
   <data android:mimeType="video/mpeg" android:scheme="http" . . . />
   <data android:mimeType="audio/mpeg" android:scheme="http" . . . />
   . . .
</intent-filter>
```

Broadcasts with Intents

- ❑ Broadcast can also be used to broadcast messages anonymously *between components with the sendBroadcast method.*
- ❑ You can implement Broadcast Receivers to listen for, and respond to, these broadcast Intents within your applications.
- ❑ Broadcast Intents are used to notify listeners of system or application events.
- ❑ Android uses Broadcast Intents extensively to broadcast system events like battery-charging levels, network connections, and incoming calls.

Broadcasting Events with Intents

- Within your application component, construct the Intent you want to broadcast, and use the sendBroadcast method to send it.

```
Intent intent = new Intent(NEW_LIFEFORM_DETECTED);
intent.putExtra(“lifeformName”, lifeformType);
intent.putExtra(“longitude”, currentLongitude);
intent.putExtra(“latitude”, currentLatitude);
sendBroadcast(intent);
```

Listening for Broadcasts with Broadcast Receivers

- ❑ Broadcast Receivers are used to listen for Broadcast Intents.
- ❑ To enable a Broadcast Receiver, it needs to be registered, either in code or within the application manifest.
- ❑ When registering a Broadcast Receiver, you must use an Intent Filter to specify which Intents it is listening for.
- ❑ To create a new Broadcast Receiver, extend the BroadcastReceiver class and override the onReceive event handler as shown in the skeleton code below:

```
public class MyBroadcastReceiver extends BroadcastReceiver {
@Override
public void onReceive(Context context, Intent intent) {
//TODO: React to the Intent received.
}
}
```

Listening for Broadcasts with Broadcast Receivers

- Applications with registered Broadcast Receivers do not have to be running when the Intent is broadcast for the receivers to execute.

Registering Broadcast Receivers

- Through XML

```
<receiver android:name=”.LifeformDetectedBroadcastReceiver”>
<intent-filter>
<action android:name=”com.paad.action.NEW_LIFEFORM”/>
</intent-filter>
</receiver>
```

- Through Code

```
IntentFilter filter = new IntentFilter(NEW_LIFEFORM_DETECTED);
LifeformDetectedBroadcastReceiver r = new
LifeformDetectedBroadcastReceiver();
registerReceiver(r, filter);
```

- Unregister

```
unregisterReceiver(r);
```

Native Android Broadcast Actions

- ❑ Android broadcasts Intents for many of the system Services.
- ❑ Following are the list of actions

Constant in Intent class	Description
ACTION_BOOT_COMPLETED	Fired once when the device has completed its start-up sequence. Requires the RECEIVE_BOOT_COMPLETED permission.
ACTION_CAMERA_BUTTON	Fired when the Camera button is clicked.
ACTION_DATE_CHANGED and ACTION_TIME_CHANGED	These actions are broadcast if the date or time on the device is manually changed
ACTION_MEDIA_BUTTON	Fired when the Media button is clicked.
ACTION_SCREEN_OFF and ACTION_SCREEN_ON	Broadcasts when the screen turns off or on.