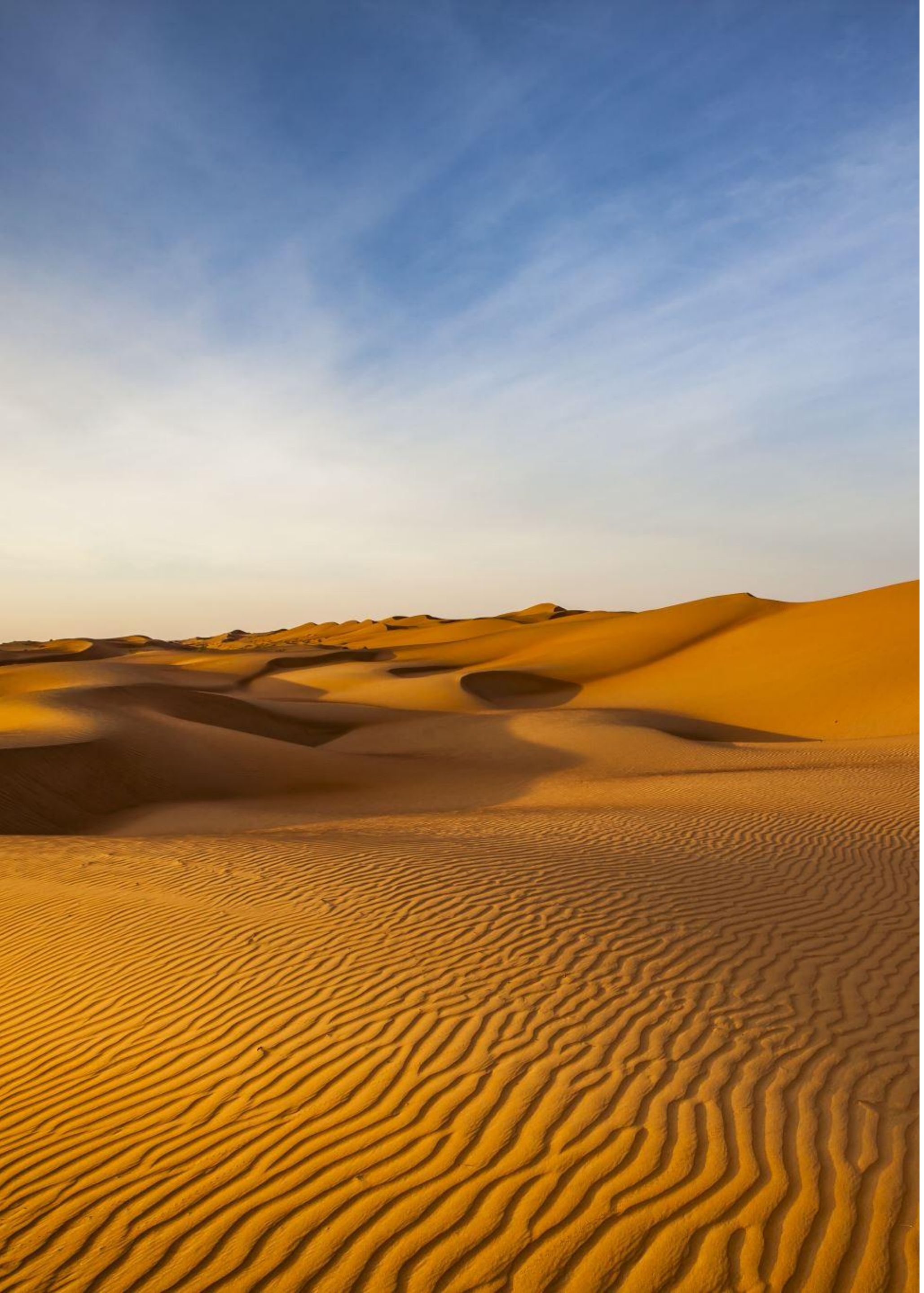


INTENTS

Mobile Application Development

By

Michael Sadgun Rao Kona,

Assistant Professor,

CSE, PVPSIT

INTENTS - INTRODUCTION

- An `Intent` is a **messaging object** which can be used to request an action from another app component.
- Intents facilitate **communication between components** in several ways
  - three fundamental use cases:
    - **Starting an activity**
    - **Starting a service**
    - **Delivering a broadcast**

INTENT TYPES

- There are two types of intents:
  - **Explicit intents**
  - **Implicit intents**

EXPLICIT INTENTS

- These **intents** specify which application (target) will satisfy the intent.
- **Target** app's package name or a fully-qualified component class name **must be specified**.
- Note:-
  - Use an explicit intent to start a component in your own app, because you know the class name of the activity or service you want to start

IMPLICIT INTENTS

Implicit intents do not name a specific component.

Declares a general action to perform

Generally, a component from another app will handle it.

For example,

- if you want to show the user a location on a map, you can use an implicit intent to request that another capable app show a specified location on a map.

Note:-

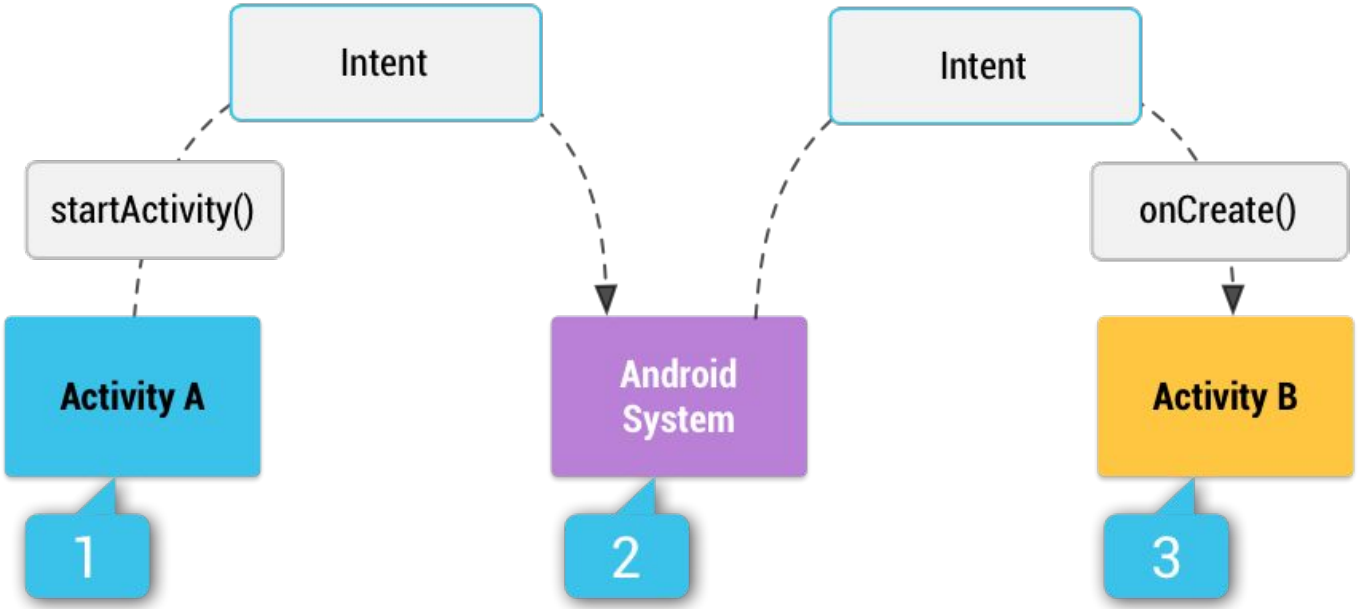
- If multiple targets are compatible, the system displays a dialog so the user can pick which app to use.

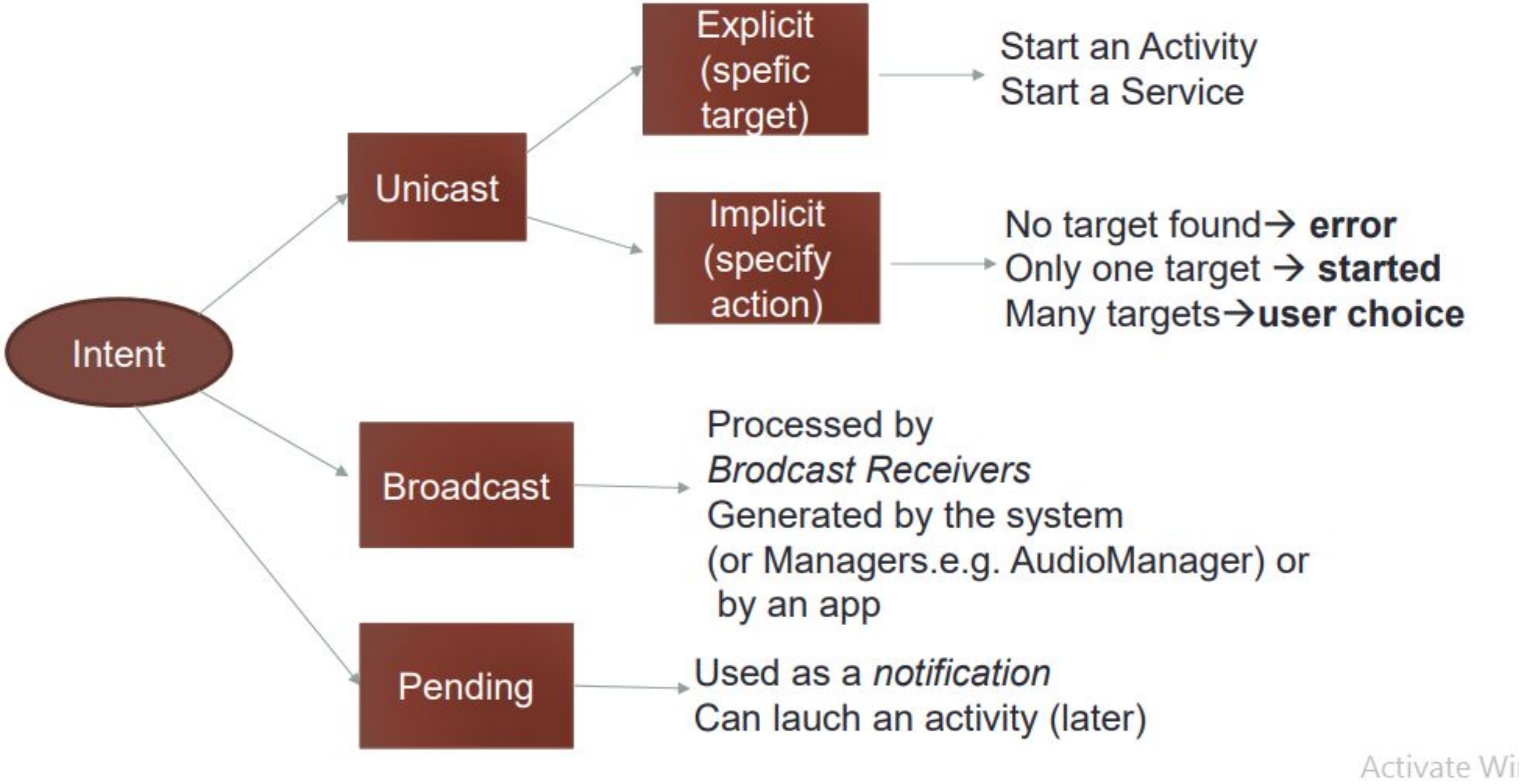
HOW AN IMPLICIT INTENT IS DELIVERED THROUGH THE SYSTEM TO START ANOTHER ACTIVITY:

[1] *Activity A* creates an Intent with an action description and passes it to startActivity().

[2] The Android System searches all apps (manifest file) for an intent filter that matches the intent. When a match is found,

[3] the system starts the matching activity (*Activity B*) by invoking its onCreate() method and passing it the Intent.





Intent

Unicast

Explicit (spefic target)

Start an Activity
Start a Service

Implicit (specify action)

No target found→ **error**
Only one target → **started**
Many targets→**user choice**

Broadcast

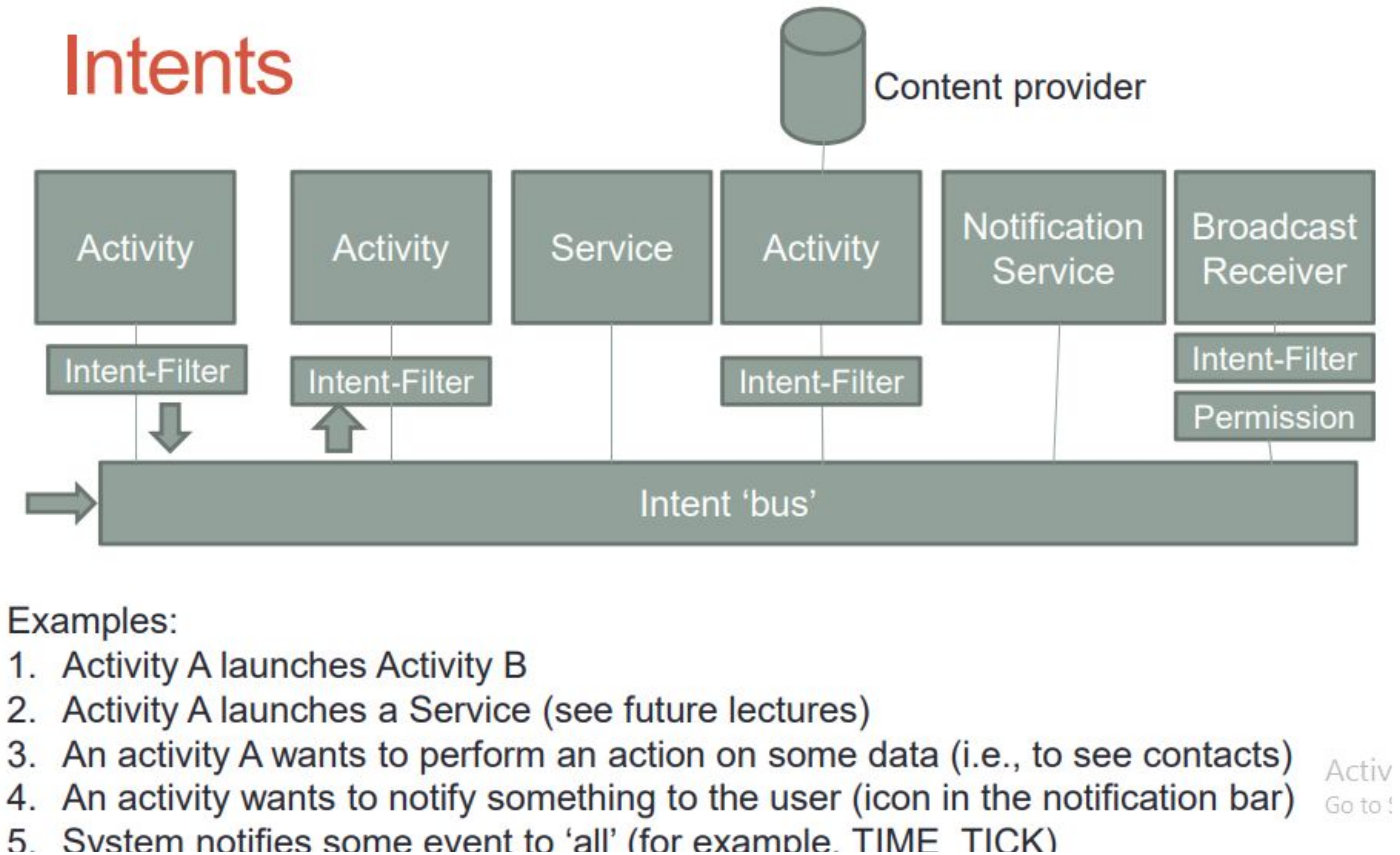
Processed by
Brodcast Receivers
Generated by the system
(or Managers.e.g. AudioManager) or
by an app

Pending

Used as a *notification*
Can lauch an activity (later)

DIFFERENT WAYS OF USING INTENT

Source: - http://www.dis.uniroma1.it/~beraldi/MACC_16/slides/05.pdf



Intents

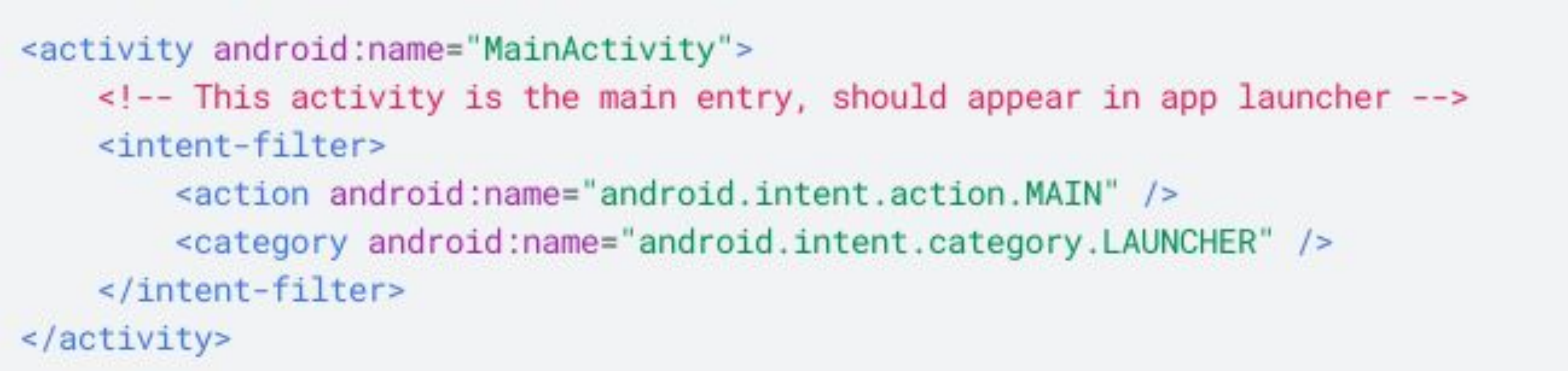
Examples:

1. Activity A launches Activity B
2. Activity A launches a Service (see future lectures)
3. An activity A wants to perform an action on some data (i.e., to see contacts)
4. An activity wants to notify something to the user (icon in the notification bar)
5. System notifies some event to ‘all’ (for example, TIME_TICK)

Source: - http://www.dis.uniroma1.it/~beraldi/MACC_16/slides/05.pdf

INTENT FILTER

- An intent filter is an expression in an app's manifest file that specifies the **type of intents** that the component would like to receive.
- Likewise, if you do *not* declare any intent filters for an activity, then it can be started only with an explicit intent.



```
<activity android:name="MainActivity">
    <!-- This activity is the main entry, should appear in app launcher -->
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

INTENT FIELDS

- An `Intent` object carries
  - information about which component to start
    - exact **component name** that should receive the intent

      or
    - **component category** that should receive the intent
  - information that the recipient component uses, such as
    - the **action** to take

      and
    - the **data** to act upon

INTENT FIELDS

- The primary information contained in an `Intent` is the following:
  - **Component name**
  - **Action**
  - **Data**
  - **Category**
  - **Extras**
  - **Flags**

INTENT FIELDS - **COMPONENT NAME**

- The name of the component to start.
- This is optional,
  - If **specified then intent is *explicit*,**
  - If **not specified, the intent is *implicit***
- **Note:** When starting a `Service`, ***always specify the component name***.

INTENT FIELDS - **ACTION**

- A string that specifies the generic action to perform (such as *view* or *pick*).
- In broadcast intent -> action says about action took place and is being reported.
  - specify **your own actions** for use by intents within your app
  - specify **action constants defined by the `Intent` class** or other framework classes
  - specify the action for an intent with `setAction()` or with an `Intent` constructor.
- `ACTION_VIEW` - uses with `startActivity()` when you have some information that an activity can show to the user, such as a **photo to view in a gallery app**, or an address to view in a map app.
- `ACTION_SEND` - use this in an intent with `startActivity()` when you have some data that the user can **share** through another app, such as an email app or social sharing app.

INTENT FIELDS - **DATA**

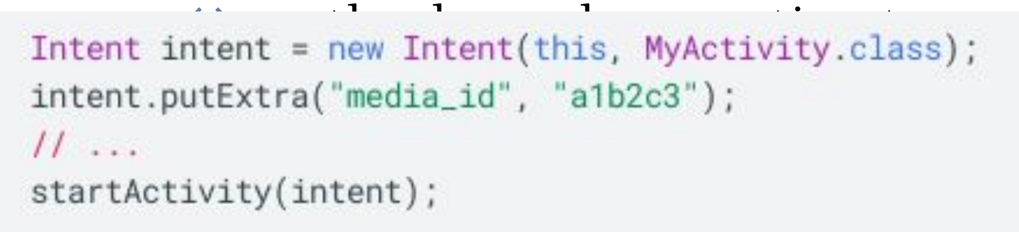
- The URI (a `Uri` object) that references the data to be acted on
- The type of data supplied is generally dictated by the intent's action.
- For example, if the action is `ACTION_EDIT`, the data should contain the URI of the document to edit.
- To set only the data URI, call `setData()`.

INTENT FIELDS
- **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, but most intents do not require a category.
- Example:
  - `CATEGORY_BROWSABLE` - The target activity allows itself to be started by a web browser to display data referenced by a link, such as an image or an e-mail message.
  - `CATEGORY_LAUNCHER` - The activity is the initial activity of a task and is listed in the system's application launcher.
- You can specify a category with `addCategory()`.

INTENT FIELDS - **EXTRAS (PASSING DATA TO INTENTS)**

- Key-value pairs that carry additional information required to accomplish the requested action.
- Add extra data with various pu arameters:
  - the key name and the value.

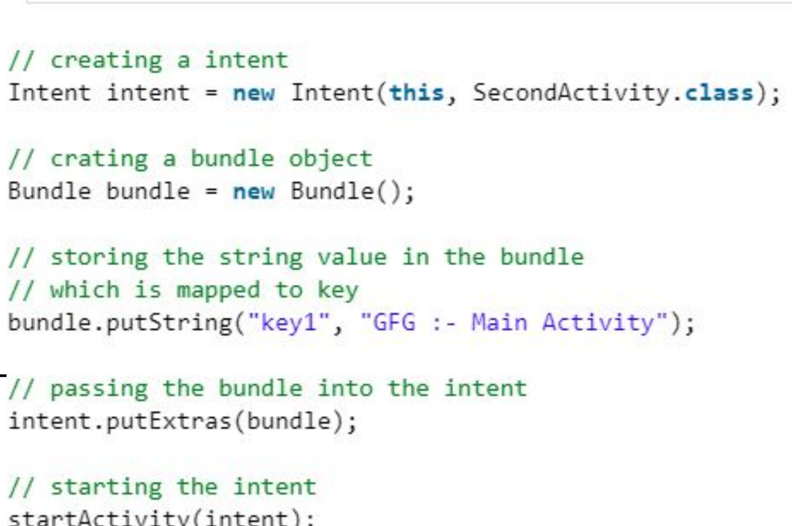


```
Intent intent = new Intent(this, MyActivity.class);
intent.putExtra("media_id", "a1b2c3");
// ...
startActivity(intent);
```

- Create a Bundle object with all the extra data,

then insert the Bundle in the Intent

with putExtras().

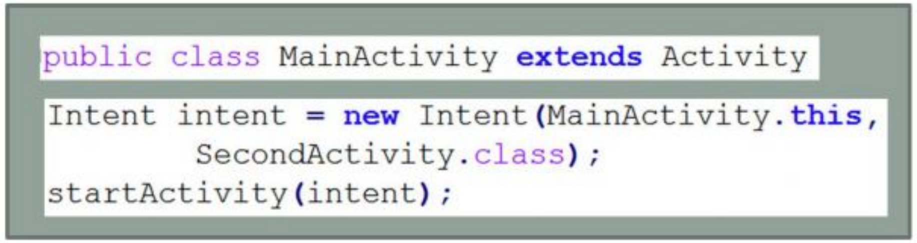


```
// creating a intent
Intent intent = new Intent(this, SecondActivity.class);

// crating a bundle object
Bundle bundle = new Bundle();

// storing the string value in the bundle
// which is mapped to key
bundle.putString("key1", "GFG :- Main Activity");

// passing the bundle into the intent
intent.putExtras(bundle);

// starting the intent
startActivity(intent);
```

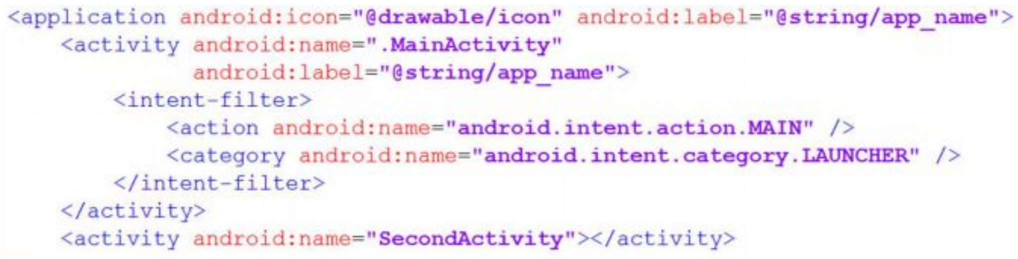
INTENT FIELDS - **FLAGS**

- Flags are defined in the `Intent` class that function as metadata for the intent.
- The flags may instruct the Android system how to launch an activity
- `setFlags()` method.

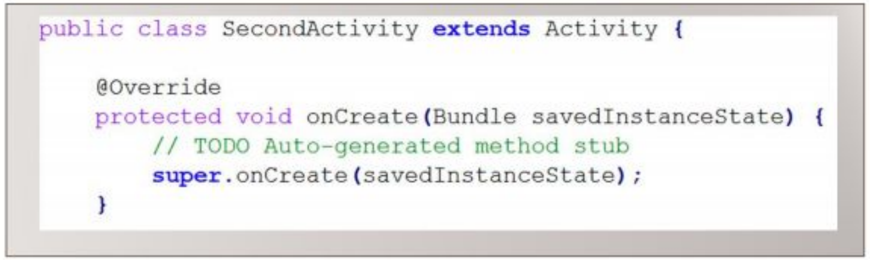
EXPLICITLY STARTING NEW ACTIVITY



```
public class MainActivity extends Activity

Intent intent = new Intent(MainActivity.this,
        SecondActivity.class);
startActivity(intent);
```



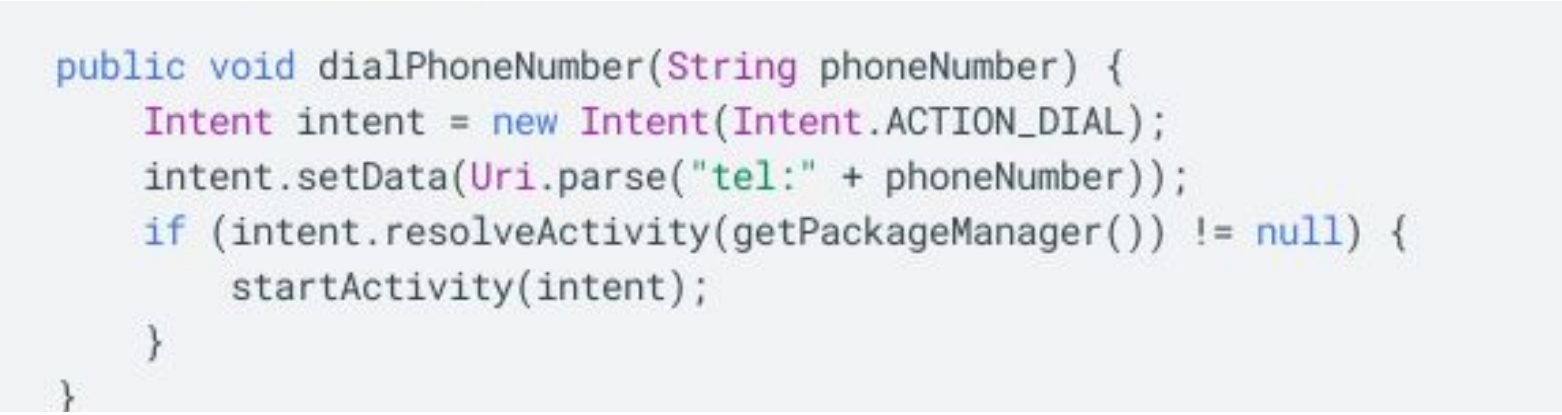
```
<application android:icon="@drawable/icon" android:label="@string/app_name">
    <activity android:name=".MainActivity"
              android:label="@string/app_name">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
    <activity android:name="SecondActivity"></activity>
```



```
public class SecondActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        // TODO Auto-generated method stub
        super.onCreate(savedInstanceState);
    }
```

INITIATE A PHONE CALL

- **Action**
  - ACTION_DIAL - Opens the dialer or phone app.
  - ACTION_CALL - Places a phone call (requires the `CALL_PHONE` permission)
    - `<uses-permission android:name="android.permission.CALL_PHONE" />`
- **Data URI Scheme**
  - `tel:<phone-number>`
  - `voicemail:<phone-number>`



```
public void dialPhoneNumber(String phoneNumber) {
    Intent intent = new Intent(Intent.ACTION_DIAL);
    intent.setData(Uri.parse("tel:" + phoneNumber));
    if (intent.resolveActivity(getPackageManager()) != null) {
        startActivity(intent);
    }
}
```

COMPOSE AN SMS/MMS MESSAGE WITH ATTACHMENT

- https://developer.android.com/guide/components/intents-common

USING INTENT FILTERS TO SERVICE IMPLICIT INTENTS

- Intent Filters are used to **register** Activities, Services, and Broadcast Receivers as being capable of performing an action on a particular kind of data.
- Using Intent Filters, application components tell Android that they can service action requests from others, including components in the same, native, or third-party applications.
- To register an application component as an Intent handler, use the intent-filter tag within the component's manifest node

TAGS WITHIN THE INTENT FILTER NODE

- action
  - **android:name** attribute - to specify the name of the action being serviced.
  - Actions should be <u>unique strings</u>.
- category
  - **android:category** attribute - to specify under which circumstances the action should be serviced.
  - Each Intent Filter tag can include multiple category tags.

END OF INTENTS

- NEXT topic
  - Broadcast Receivers