



Programming with Android: **Activities and Intents**



Luca Bedogni

Marco Di Felice

Dipartimento di Informatica – Scienza e Ingegneria
Università di Bologna



Outline

What is an **intent**?

Intent description

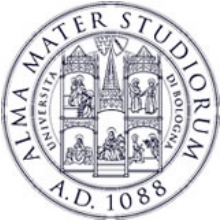
Handling **Explicit Intents**

Handling **implicit Intents**

Intent-Resolution process

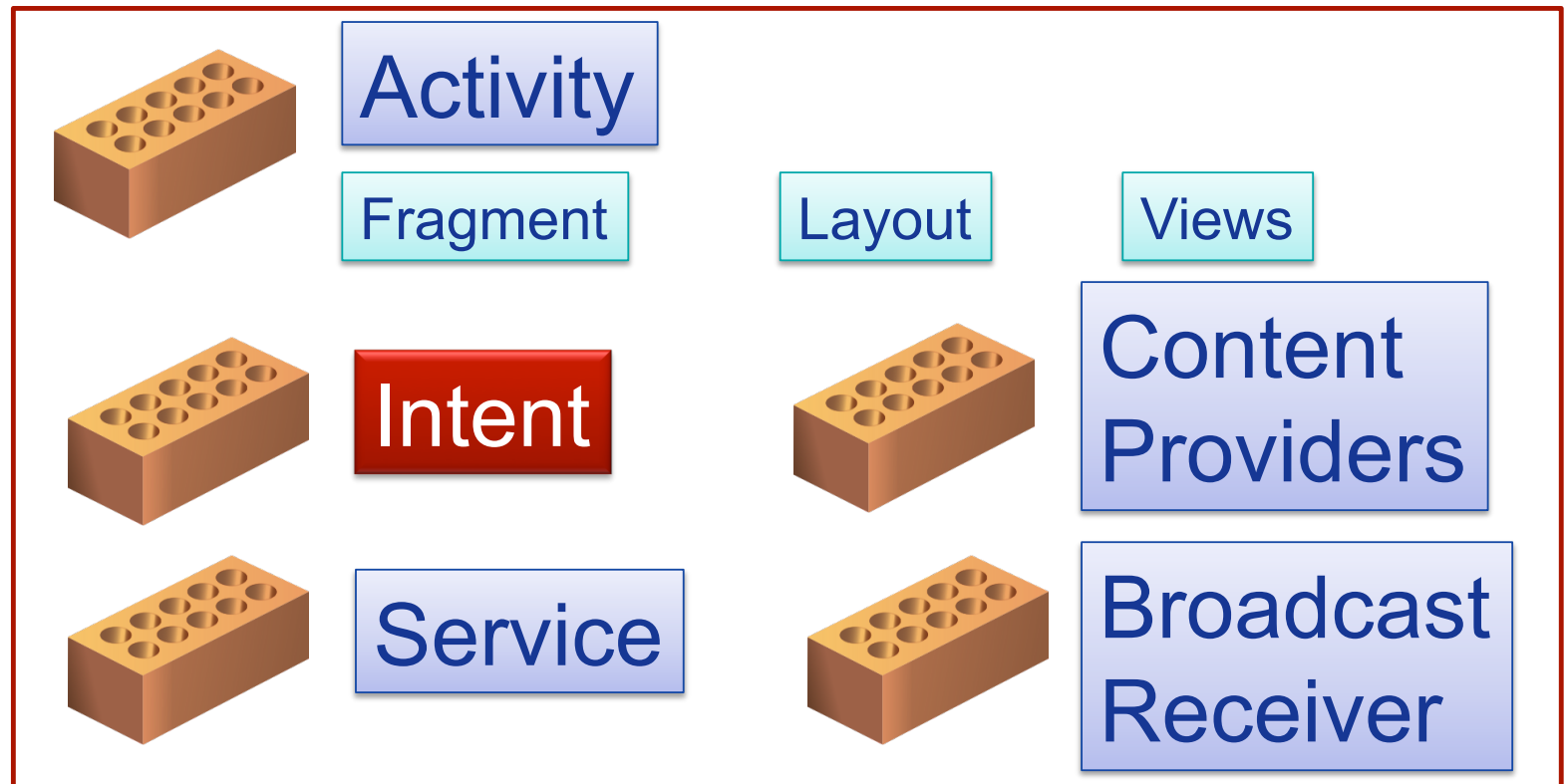
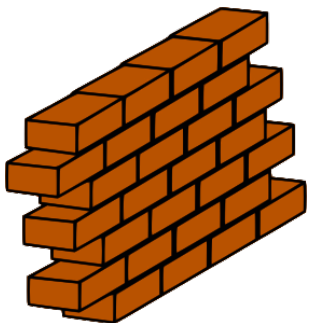
Intent with results: Sender side

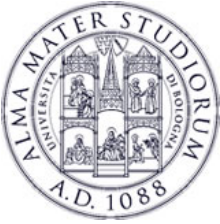
Intent with results: Receiver side



Android Applications **Design**

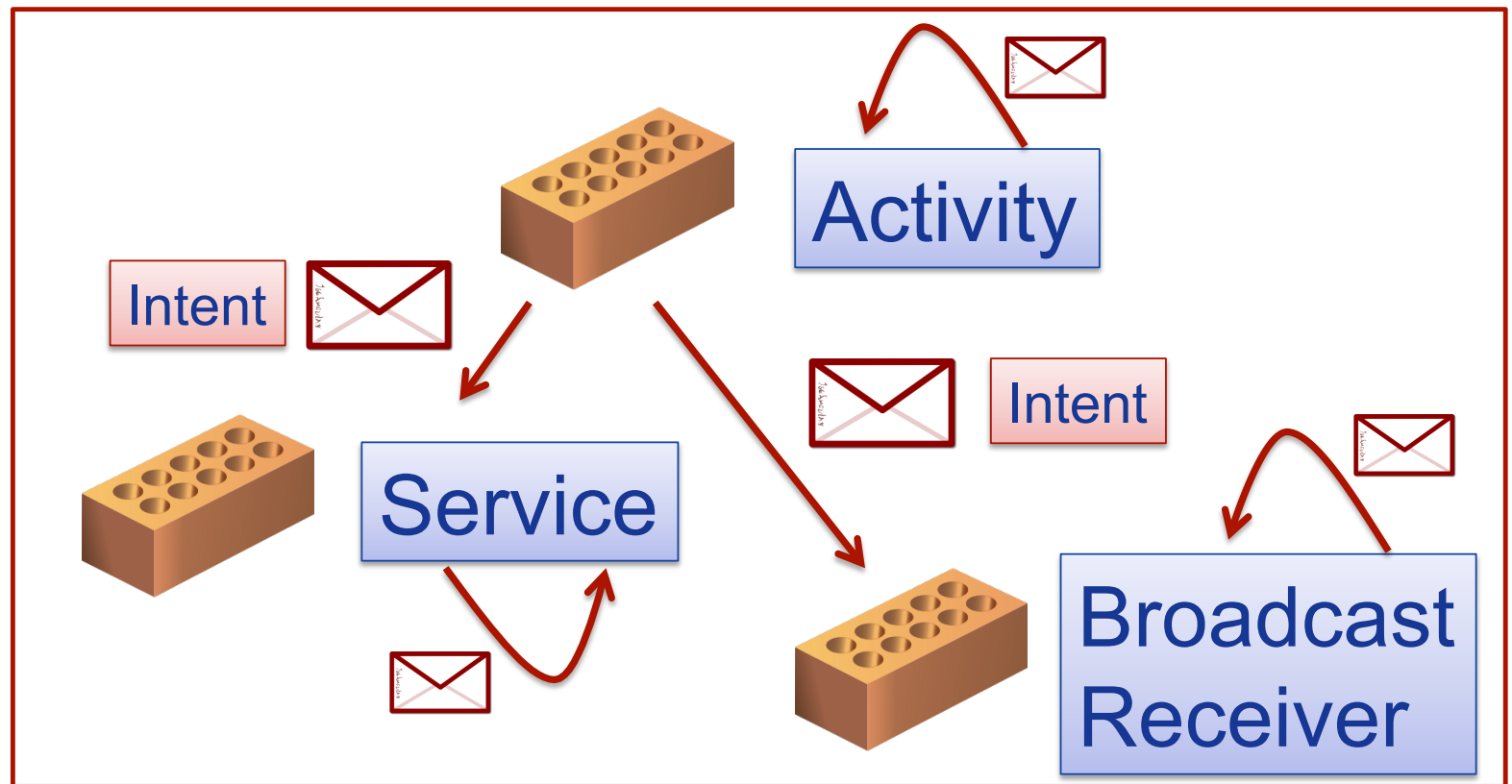
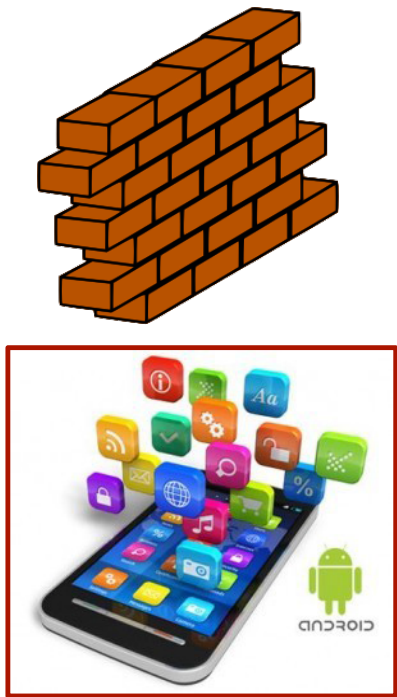
Developing an Android Application means using in a proper way the **Android basic components** ...

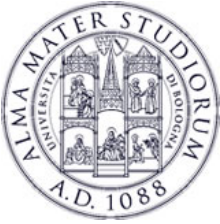




Android Applications **Design**

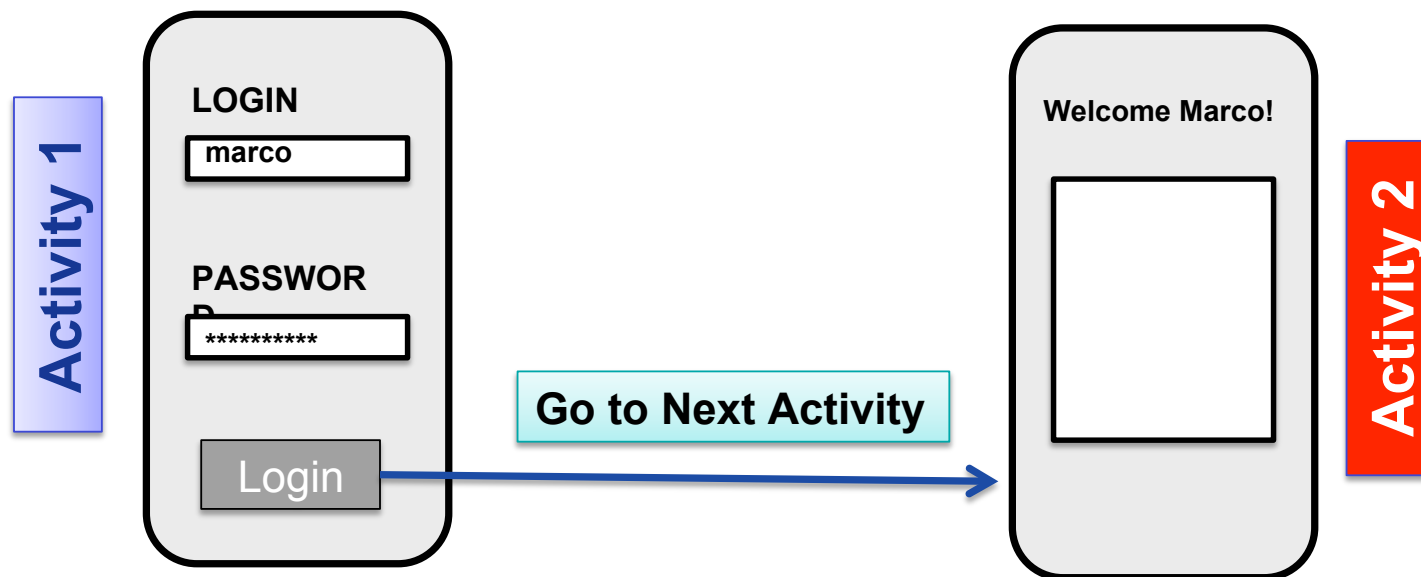
Developing an Android Application means using in a proper way the **Android basic components** ...





More on Activities: Activity **states**

In most cases, an Android Application is composed of **multiple Activities, not just one ...**





More on Activities: **Saving resources**

Each Activity has its own:

- ✧ **Java** implementation $\longrightarrow$ `MyActivity.java`
- ✧ **XML Layout** file $\longrightarrow$ `activity_main.xml`
- ✧ **Lifecycle** with different states **ACTIVE** **PAUSED** **STOPPED**
- ✧ **XML Tag** in `AndroidManifest.xml`

```
<application>
  <activity android:name=".MyActivity" />
</application>
```



Activities and **AndroidManifest.xml**

✧ Each activity has its **Java** class and **layout** file.

```
public class FirstActivity extends Activity {  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_first);  
    }  
}
```

```
public class SecondActivity extends Activity {  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_two);  
    }  
}
```



Intent Definition

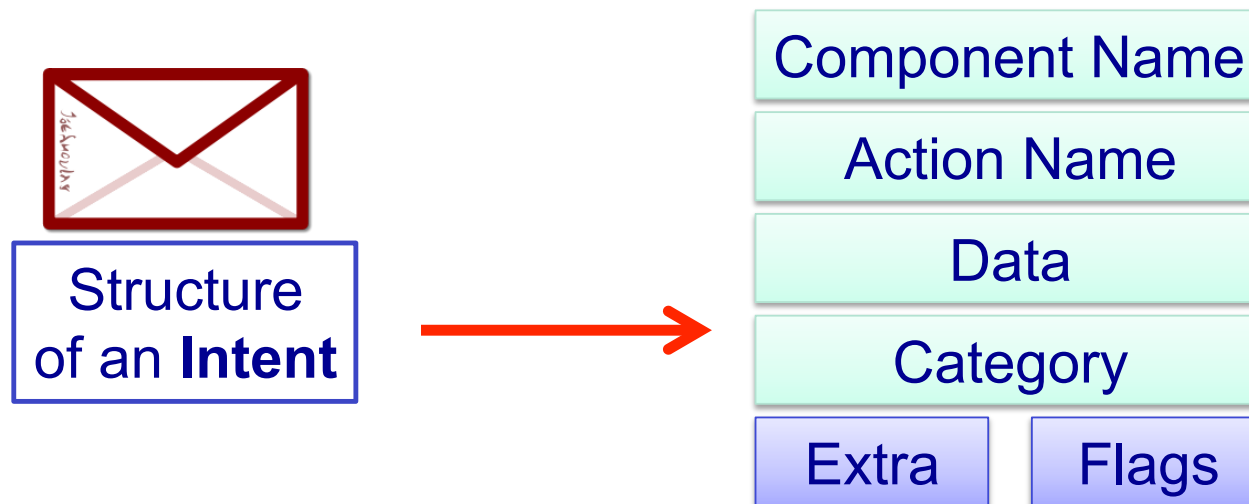
Intent: facility for late **run-time binding** between components of the same or different applications.

- ✧ **Call** a component from another component
- ✧ Possible to **pass data** between components
- ✧ Components: **Activities**, *Services*, *Broadcast receivers* ...
- ✧ Something like:
 - ✧ “Android, please do that with these data”
- ✧ **Reuse** already installed applications and components

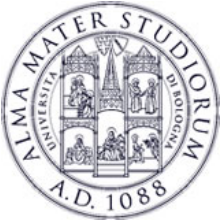


Intent Definition

We can think to an **Intent** object as a **message** containing a bundle of information.



✧ Intent is sent from current Activity to a receiver Activity which is then **activated** and **executed**.



Intent **types**



INTENT TYPES

EXPLICIT



The target receiver is specified through the **Component Name**

Used to launch specific Activities

IMPLICIT



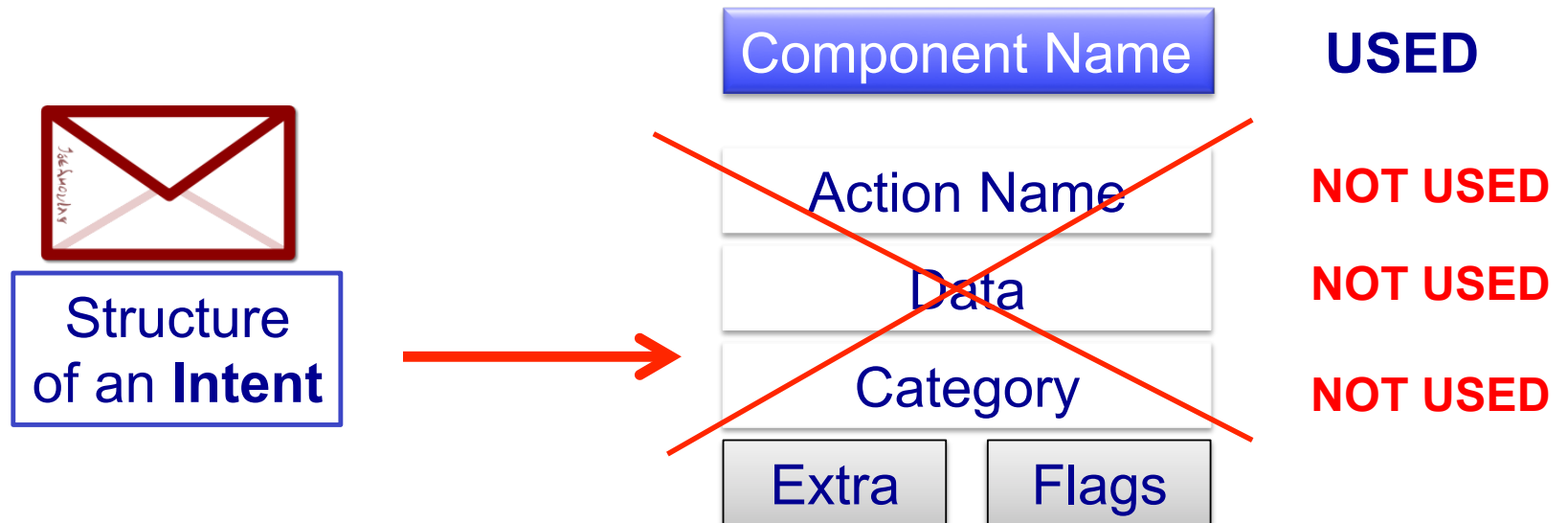
The target receiver is specified through the actions to be executed.

The system chooses the receiver that matches the request.

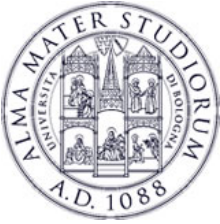


Intent **types**: Explicit Intents

Explicit Intent → Specify the **name** of the Activity that will handle the intent.



- ✧ Used to control the execution flow **between Activities belonging to the same Android application.**

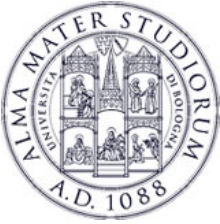


Intent **types**: Explicit Intents

1. Build a new Intent message
2. Specify the Activity who will **receive** the Intent
3. Fire the Intent through the startActivity()

NAME OF THE ACTIVITY TO START

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

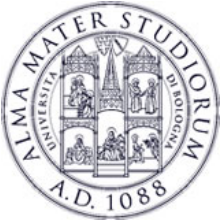


Intent **types**: Explicit Intents

1. Build a new Intent message
2. Specify the Activity who will **receive** the Intent
3. Fire the Intent through the `startActivity()`

```
Intent intent=new Intent();  
ComponentName component=new  
ComponentName(this,SecondActivity.class);  
intent.setComponent(component);  
startActivity(intent);
```

ALTERNATIVE code



Intent **types**: Explicit Intents

(OPTIONAL) Insert parameters to be sent to the called Activity in the **Extra** field of the Intent.

```
intent.putExtra("KEY", VALUE);
```

Set an argument named "MyValue" and equal to 5.

```
Intent intent=new Intent(this, SecondActivity.class);  
intent.putExtra("myValue",5);  
startActivity(intent);
```



Intent **types**: Explicit Intents

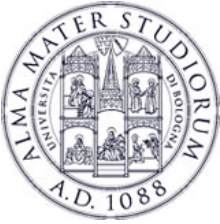
(OPTIONAL) From the called Activity, **retrieve the parameters** inserted from the calling Activity

```
intent.getExtras().getTYPE("KEY");
```

Get an argument of type int with key equal to "myValue"



```
intent.getExtras().getInt("myValue");  
intent.getExtras().getString("myString");  
intent.getExtras().getBoolean("myBoolean");
```



Intent **types**: Explicit Intents

Some Activities can return **results** to the caller Activity

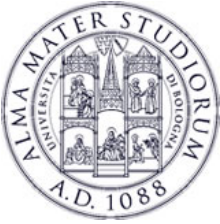
SENDER SIDE

1. **IMPLEMENT** method `onActivityResult(int requestCode, int resultCode, Intent data)`
2. **INVOKE** method `startActivityForResult(Intent intent, int requestCode)`

```
Intent intent = new Intent(this, SecondActivity.class);  
startActivityForResult(intent, CHOOSE_ACTIVITY_CODE);
```

...

```
public void onActivityResult(int requestCode, int  
resultCode, Intent data) {  
    // Invoked when SecondActivity completes its operations
```



Intent **types**: Explicit Intents

Some Activities can return **results** to the caller Activity

RECEIVER SIDE

1. **IMPLEMENT** method `setResult(int resultCode, Intent data)`
2. **SET results** on the Extra field of the Intent

```
Intent intent=new Intent();  
setResult(RESULT_OK, intent);  
intent.putExtra("result", resultValue);  
finish();
```

The result is **delivered** only after invoking the **finish()** method!



Intent **types**



INTENT TYPES

EXPLICIT



The target receiver is specified through the **component Name**

Used to launch specific Activities

IMPLICIT



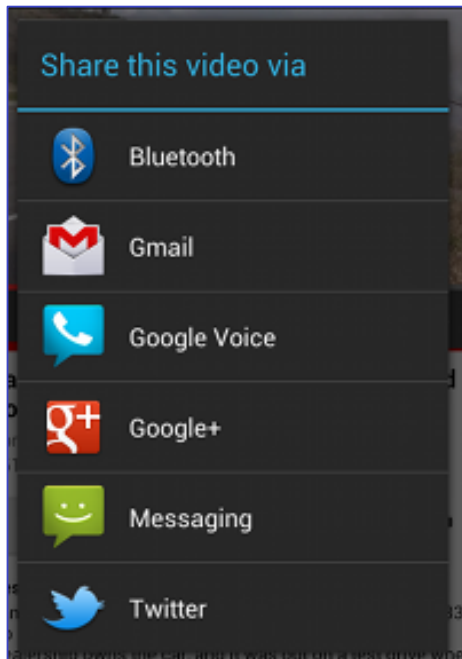
The target receiver is specified through the actions to be executed.

The system chooses the receiver that matches the request.



Intent **types**: Implicit Intents

Implicit Intents → do not name a target (component name is left blank) but an **intention** of what to do ...

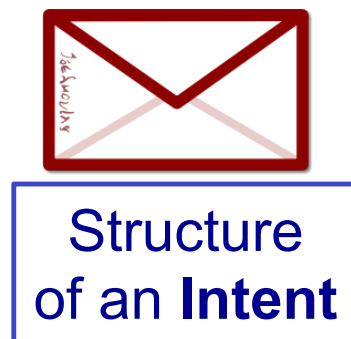


- ✧ When an Intent is launched, **Android** checks out which Activities might answer to the Intent ...
- ✧ If at least one is found, then that Activity is started
- ✧ Binding does not occur at compile time, nor at install time, but at run-time ...
(*late run-time binding*)



Intent **types**: Explicit Intents

Implicit Intents → do not name a target (component name is left blank) but an **intention** of what to do ...



~~Component Name~~

NOT USED

Action Name

USED

Data

USED

Category

USED

Extra

Flags

- ✧ Used to control the execution flow **between Activities belonging to DIFFERENT Android applications.**



Intent **types**: Explicit Intents

Implicit Intents → do not name a target (component name is left blank) but an **intention** of what to do ...

- ✧ A string naming the **action** to be performed.
- ✧ **Pre-defined**, or can be **specified** by the programmer.
- ✧ void **setAction**(String)

~~Component Name~~

NOT USED

Action Name

USED

Data

USED

Category

USED

Extra

Flags



Intent **types**: Explicit Intents

Special actions (<http://developer.android.com/reference/android/content/Intent.html>)

Action Name	Description
ACTION_IMAGE_CAPTURE	Open the camera and receive a photo
ACTION_VIDEO_CAPTURE	Open the camera and receive a video
ACTION_DIAL	Open the phone app and dial a phone number
ACTION_SENDTO	Send an email (email data contained in the extra)
ACTION_SETTINGS	Open the system setting
ACTION_WIRELESS_SETTINGS	Open the system setting of the wireless interfaces
ACTION_DISPLAY_SETTINGS	Open the system setting of the display



Intent **types**: Explicit Intents

Generic actions (<http://developer.android.com/reference/android/content/Intent.html>)

Action Name	Description
ACTION_EDIT	Display data to edit
ACTION_MAIN	Start as a main entry point, does not expect to receive data.
ACTION_PICK	Pick an item from the data, returning what was selected.
ACTION_VIEW	Display the data to the user
ACTION_SEARCH	Perform a search

✧ **Action Defined by the programmer**

it.example.projectpackage.FILL_DATA (package prefix + name action)

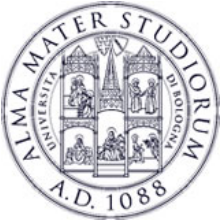


Intent **types**: Explicit Intents

1. Build a new Intent message
2. Specify only the **Action** you want to perform
3. Fire the Intent through the startActivity()

ACTION NAME

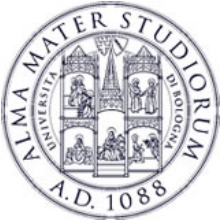
```
Intent intent=new Intent(MediaStore.ACTION_IMAGE_CAPTURE);  
  
startActivity(intent);
```



Intent **types**: Explicit Intents

1. Build a new Intent message
2. Specify only the **Action** you want to perform
3. Fire the Intent through the `startActivity()`
4. **Verify whether the Intent can be handled**

```
Intent intent=new Intent(MediaStore.ACTION_IMAGE_CAPTURE);  
if (intent.resolveActivity(getPackageManager()) !=  
null) {  
    startActivity(intent);  
}
```



Intent **types**: Explicit Intents

Implicit Intents → do not name a target (component name is left blank) but an **intention** of what to do ...

✧ Data passed from the caller to the called Component.

✧ Def. of the data (**URI**) and Type of the data (**MIME** type)

1. void **setData**(Uri)
2. void **setType**(String)

~~Component Name~~

NOT USED

Action Name

USED

Data

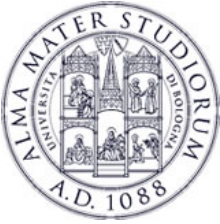
USED

Category

USED

Extra

Flags



Intent **types**: Explicit Intents

In an Intent, the **Data** is specified by a **name** and a **type**

NAME: Uniform Resource Identifier (URI)

scheme://host:port/path

tel:003-232-134-126

content://contacts/people/1

http://www.cs.unibo.it

EXAMPLEs



Intent **types**: Explicit Intents

Some actions require **data in input** to be executed.



Use method `setData(Uri)` to define the data input of an Implicit Intent

```
Intent intent=new Intent(Intent.ACTION_DIAL);  
intent.setData(Uri.parse("tel:0123456789"));  
startActivity(intent);
```



Intent **types**: Explicit Intents

Some actions require **data in input** to be executed.



Use method **setData(Uri)** to define the data input of an Implicit Intent

```
Intent intent=new Intent(Intent.ACTION_VIEW);  
intent.setData(Uri.parse("content://contacts/people/1"));  
startActivity(intent);
```



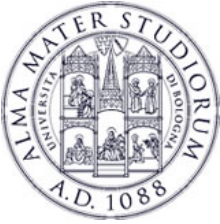
Intent **types**: Explicit Intents

Some actions require **data in input** to be executed.



Use method **setData(Uri)** to define the data input of an Implicit Intent

```
Intent intent=new Intent(Intent.ACTION_VIEW);  
intent.setData(Uri.parse("http://www.cs.unibo.it"));  
startActivity(intent);
```



Intent **types**: Explicit Intents

In an Intent, the **Data** is specified by a **name** and a **type**

TYPE: Multipurpose Internet Mail Extensions (**MIME**)

type/subtype

EXAMPLES

<code>image/gif</code>	<code>image/jpeg</code>	<code>image/png</code>
<code>text/html</code>	<code>text/plain</code>	<code>text/css</code>
<code>video/mpeg4</code>	<code>...</code>	<code>...</code>



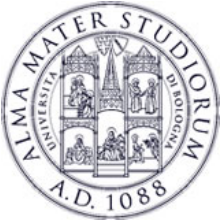
Intent **types**: Explicit Intents

Some actions require **data in input** to be executed.



Use method **setType(MIME)** to define the data input of an Implicit Intent

```
Intent sendIntent = new Intent();  
sendIntent.setAction(Intent.ACTION_SEND);  
sendIntent.putExtra(Intent.EXTRA_TEXT, textMessage);  
sendIntent.setType(HTTP.PLAIN_TEXT_TYPE);  
startActivity(sendIntent)
```



Intent **types**: Explicit Intents

Implicit Intents → do not name a target (component name is left blank) but an **intention** of what to do ...

- ✧ A string containing information about the **kind of component** that should handle the Intent.
- ✧ More than one can be specified for an Intent
- ✧ `void addCategory(String)`

~~Component Name~~

NOT USED

Action Name

USED

Data

USED

Category

USED

Extra

Flags



Intent Components

Category → String describing the kind of component (Activity) that should handle the Intent.

Category Name	Description
CATEGORY_HOME	The activity displays the HOME screen.
CATEGORY_LAUNCHER	The activity is listed in the top-level application launcher, and can be displayed.
CATEGORY_PREFERENCE	The activity is a preference panel.
CATEGORY_BROWSABLE	The activity can be invoked by the browser to display data referenced by a link.



Intent Components

1. Build a new Intent message
2. Specify only the **Category** of the receiver Activity
3. Fire the Intent through the startActivity()

```
Intent intent=new Intent();  
Intent.setAction(Intent.ACTION_MAIN);  
intent.addCategory(Intent.CATEGORY_HOME);  
  
startActivity(intent);
```



Intent **types**: Intent Resolution

QUESTION: How can Android know which application to call after an Implicit Intent is fired?

ANSWER: Each application declares the **Intent** is able to handle in the `AndroidManifest.xml` file

If an Intent with Action name `ACTION_ECHO` is invoked, the Activity is launched

```
<intent-filter>  
    <action android:name="ACTION_ECHO" />  
</intent-filter>
```



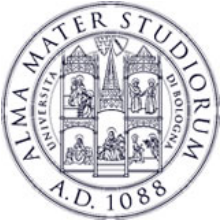
Intent **types**: Intent Resolution

The **Intent resolution process** resolves the Intent-Filter that can handle a given Intent (e.g. ACTION_ECHO).

Three **tests** must be passed:

1. **Action** field test
2. **Category** field test
3. **Data** field test

If the Intent-filter of Activity A **passes all the three test**, then it is selected to handle the Intent.



Intent **types**: Intent Resolution

(ACTION Test): The action specified in the Intent must match one of the actions listed in the filter.

- ✧ If the filter does not specify any action → **FAIL**
- ✧ An Intent that does not specify an action → **SUCCESS** as long as the filter contains at least one action.

```
<intent-filter ... >  
    <action android:name="com.example.it.ECHO"/>  
</intent-filter>
```



Intent **types**: Intent Resolution

(CATEGORY Test): Every category in the Intent must match a category of the Intent Filter.

- ✧ If the category is **not specified** in the Intent → Android assumes it is **CATEGORY_DEFAULT**, thus the filter must include this category to handle the Intent.

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



Intent **types**: Intent Resolution

(DATA Test): The URI of the intent is compared with the parts of the URI mentioned in the filter (this part might be incompleted).

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

✧ Both URI and MIME-types are compared (4 different sub-cases ...)