Master in Computer Engineering

Information Management Mobile Devices

Intents

Javier Abad (abad@decsai.ugr.es)

Dept. Of Computer Science and Artificial Intelligence

http://decsai.ugr.es

Let's remember...

- An activity can programmatically start another activity
  - We create an object Intent
  - Intent to pass that startActivity () or startActivityForResult ()
- **contents:**
  - The Intent class
  - Initiation of activities
    - explicit activation
    - Implicit activation by resolving intents

The Intent class

- Intent is a data structure that can represent:
  - An operation to be performed
  - An event that has occurred in the system
- Let's focus on the use of Intents to specify operations to be performed, not for event notification.
- We will see the use of Intents for event notification when talking receptor ads (Broadcast Receivers).

Using Intents to specify operations

- The Intents provide a flexible and convenient language for specifying operations that we want to perform.
  - Select a contact
  - Take a picture
  - Dial a phone number
  - Show map
  - ...
- The intent builds an activity that wants a particular task is made
- Android starts another activity to perform the desired task. This activity receives the Intent.

Fields of Intent

- Action
- Data
- Category
- Type
- Component
- Additional features
- Flags

Action

- Or name string representing the operation to perform

- Examples:
  - ACTION_DIAL: dial a phone number
  - ACTION_EDIT: display data editing
  - ACTION_SYNC: synchronize data with a server device

  - ACTION_MAIN: initiating activity as the initial application activity

Action

- Predefined actions to launch activities:

ACTION_MAIN	ACTION_FACTORY_TEST
ACTION_EDIT	ACTION_WEB_SEARCH
ACTION_VIEW	ACTION_SEARCH
ACTION_ATTACH_DATA	ACTION_PICK_ACTIVITY ACTION_PICK
	ACTION_SYNC
ACTION_CHOOSER	ACTION_RUN
ACTION_GET_CONTENT	ACTION_DELETE ACTION_DIAL
	ACTION_INSERT
ACTION_CALL	ACTION_ANSWER
ACTION_SEND	ACTION_SENDTO

Setting the Action of Intent

- Action spend builder

```
Intent nuevo_intent = new Intent (Intent.ACTION_DIAL);
```

- Intent create the vacuum and using the method setAction

```
Intent nuevo_intent = new Intent (); nuevo_intent.setAction
(Intent.ACTION_DIAL);
```

Data

- Data associated with the Intent
  - Formatted as a URI (*Uniform Resource Identifier:* Uniform Resource Identifier)
- Examples:
  - Data for a map

    Uri.parse ("geo: q = 0.0 + 50 + Recogidas Granada?")
  - Number to dial on your phone

    Uri.parse ("tel: +34900123456")

Uri.parse creates a Uri object from the chain that is passed

Setting the Data of Intent

- Data pass to the constructor

```
Intent nuevo_intent = new Intent (Intent.ACTION_DIAL,
                                        Uri.parse ( "tel: +34900123456"));
```

- Intent create and use the setData method

```
Intent nuevo_intent = new Intent (Intent.ACTION_DIAL); nuevo_intent.setData
(Uri.parse ( "tel: +34900123456"));
```

Category

- Additional information about which components can handle the Intent.
- Examples:
  - CATEGORY_BROWSABLE: It may be invoked from a browser through a link type URI (URL).
  - CATEGORY_LAUNCHER: It may be the initial activity of a task and appears in the application launcher (Launcher).

Setting the category of Intent

Intent.addCategory (String category);

Category

- Predefined categories:

CATEGORY_DEFAULT	CATEGORY_PREFERENCE
CATEGORY_BROWSABLE	CATEGORY_TEST
CATEGORY_TAB	CATEGORY_CAR_DOCK
CATEGORY_ALTERNATIVE	CATEGORY_LE_DESK_DOCK
CATEGORY_SELECTED_ALTERNATIVE	CATEGORY_HE_DESK_DOCK
CATEGORY_LAUNCHER	CATEGORY_DESK_DOCK
CATEGORY_INFO	CATEGORY_CAR_MODE
CATEGORY_HOME	CATEGORY_APP_MARKET

Type

- Specifies the MIME type of the data (Data) of Intent.
- Examples:
  - image / *, image / png, image / jpg
  - text / html, text / plain
- If you do not specify the MIME type of the data, Android will try to infer.

How to set the kind of Intent

Intent.setType (String type)

Intent.setDataAndType (data Uri, String type)

Eye!: If we want to specify Data (URI) and Type (MIME),
we should not call setData Y setType,
because a method overrides another. In this case we use setDataAndTyp

Component

- The component that should receive the Intent.
- It used when we know that there is indeed an activity which should receive the Intent.

Setting the component of Intent

```
Nuevo_intent intent = new Intent (Context packageContext,
                                                    Class class) <?>;
```

O well:

```
Intent nuevo_intent = new Intent ();
```

and using one of the methods:

setComponent ()

setClass ()

setClassName ()

Additional features

- Additional information associated with the Intent
- They are treated as a map of key-value pairs
- Examples:
  - Intent.EXTRA_EMAIL: mail recipients

```
Intent nuevo_intent = new Intent (Intent.ACTION_SEND); nuevo_intent.putExtra
(android.content.Intent.EXTRA_EMAIL,
                                        new String [] {
                                                          "Abad@decsai.ugr.es" "tu@tudominio.com"
                                                          "alcalde@granada.org",
                                                          "presidente@juntadeandalucia.es"}

                                      );
```

Extra setting the attribute of Intent

- The putExtra () method is overloaded for a variety of types.

  Intent.putExtra (String name, String value); Intent.putExtra (String name, String [] value); Intent.putExtra (String name, float value);

  . . .

Additional features

- Extras predefined:

EXTRA_ALARM_COUNT	EXTRA_ORIGINATING_URI
EXTRA_BCC	EXTRA_PHONE_NUMBER
EXTRA_CC	EXTRA_REFERRER
EXTRA_CHANGED_COMPONENT_NAME	EXTRA_REMOTE_INTENT_TOKEN
EXTRA_DATA_REMOVED	EXTRA_REPLACING
EXTRA_DOCK_STATE	EXTRA_SHORTCUT_ICON
EXTRA_DOCK_STATE_HE_DESK	EXTRA_SHORTCUT_ICON_RESOURCE
EXTRA_DOCK_STATE_LE_DESK	EXTRA_SHORTCUT_INTENT
EXTRA_DOCK_STATE_CAR	EXTRA_STREAM
EXTRA_DOCK_STATE_DESK	EXTRA_SHORTCUT_NAME
EXTRA_DOCK_STATE_UNDOCKED	EXTRA_SUBJECT
EXTRA_DONT_KILL_APP	EXTRA_TEMPLATE
EXTRA_EMAIL	EXTRA_TEXT
EXTRA_INITIAL_INTENTS	EXTRA_TITLE
EXTRA_INTENT	EXTRA_UID
EXTRA_KEY_EVENT	

Flags

- They provide information on how it should handle the Intent.
- Examples:
  - FLAG_ACTIVITY_NO_HISTORY: not to include this activity in the history stack.
  - FLAG_DEBUG_LOG_RESOLUTION: Show additional logging when the Intent is processed.

How to Set Flags of Intent

```
Intent nuevo_intent = new Intent (Intent.ACTION_SEND); nuevo_intent.setFlags
(Intent.FLAG_ACTIVITY_NO_HISTORY);
```

Category

- Flags predefined (for startActivity):

FLAG_ACTIVITY_BROUGHT_TO_FRONT	FLAG_ACTIVITY_NEW_TASK
FLAG_ACTIVITY_CLEAR_TASK	FLAG_ACTIVITY_NO_ANIMATION
FLAG_ACTIVITY_CLEAR_TOP	FLAG_ACTIVITY_NO_HISTORY
FLAG_ACTIVITY_CLEAR_WHEN_TASK_RESET	FLAG_ACTIVITY_NO_USER_ACTION
FLAG_ACTIVITY_EXCLUDE_FROM_RECENTS	FLAG_ACTIVITY_PREVIOUS_IS_TOP
FLAG_ACTIVITY_FORWARD_RESULT	FLAG_ACTIVITY_RESET_TASK_IF_NEEDED
FLAG_ACTIVITY_LAUNCHED_FROM_HISTORY	FLAG_ACTIVITY_REORDER_TO_FRONT
FLAG_ACTIVITY_MULTIPLE_TASK	FLAG_ACTIVITY_SINGLE_TOP
FLAG_ACTIVITY_NEW_DOCUMENT	FLAG_ACTIVITY_TASK_ON_HOME

Initiation of activities with Intents

- Programmatically initiating activities:
  - startActivity (Intent intent, ...)
  - startActivityForResult (Intent intent, ...)

- Resolution Intents
  - Explicitly, establishing the component of Intent
  - Implicitly, from data Intent and the characteristics of the activities installed

Before you continue ...

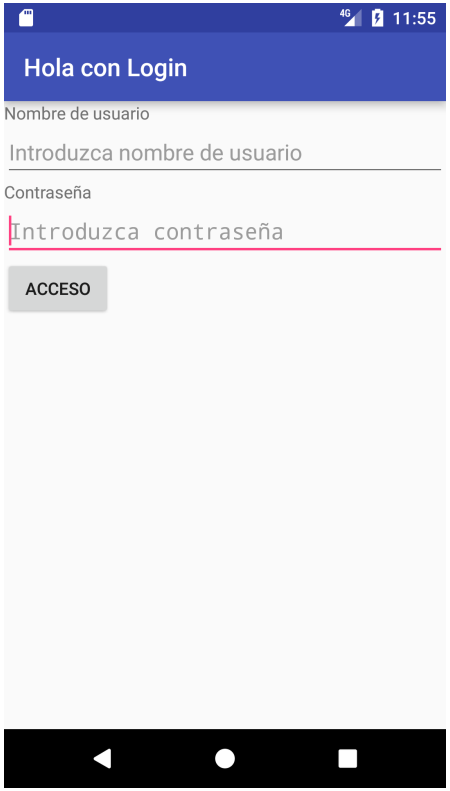
- It is possible that the device does not have any application to meet the intent we pass to startActivity. Then the call will fail and the application will abort execution (crash).

- To verify that there is an activity that will serve the intent, we can call resolveActivity ():

```
if (mi_intent.resolveActivity (getPackageManager ())! = null) {
        startActivity (sendIntent); }
```

explicit activation

- Hello with Login: two activities
  - LoginActivity: check username and password and initiates HolaAndroid
  - HolaAndroid displays a message on screen



11:55

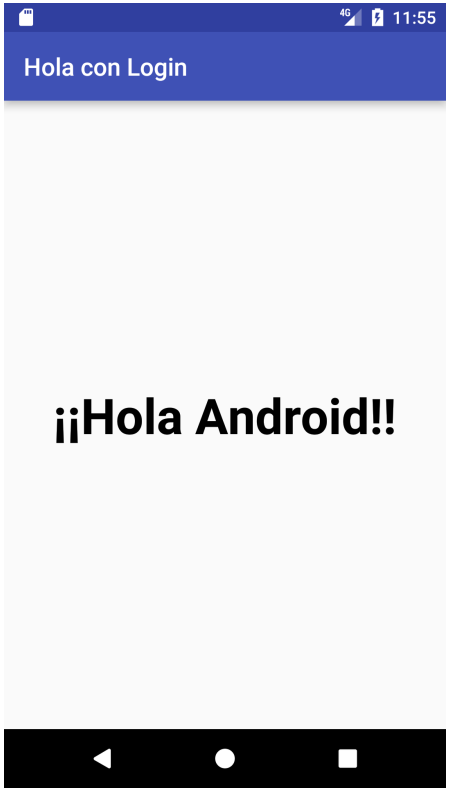
Hola con Login

Nombre de usuario

Introduzca nombre de usuario

Contraseña

Introduzca contraseña



11:55

Hola con Login

¡¡Hola Android!!

HolaConLogin

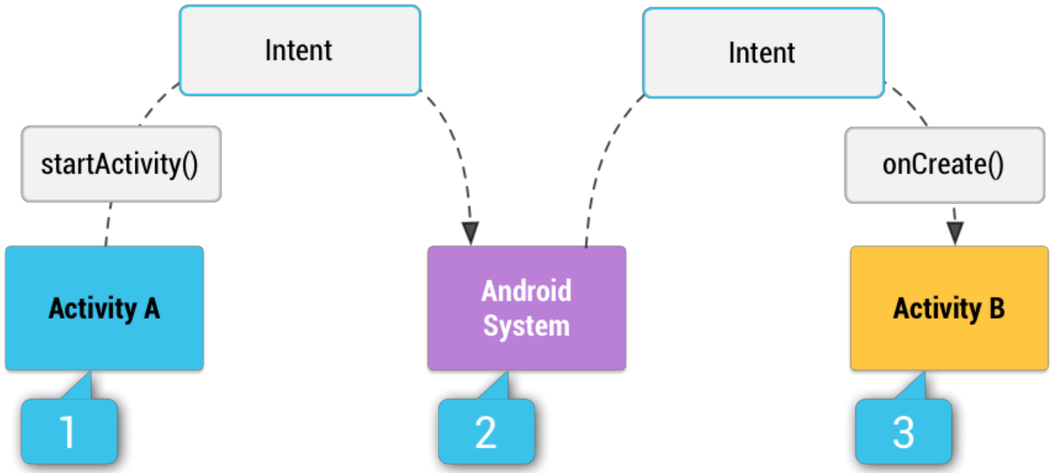
```
public class LoginActivity extends AppCompatActivity {
    @Override
    protected void onCreate (Bundle savedInstanceState) {
        super.onCreate (savedInstanceState); setContentView
        (R.layout.activity_login);

        EditText end user = (EditText) findViewById (R.id.usuario_edittext); end EditText password = (EditText)
        findViewById (R.id.password_edittext); end BUTTONs = (Button) findViewById (R.id.login_button);


        boton.setOnClickListener (new View.OnClickListener () {
            public void onClick (View v) {
                if (compruebaPassword (usuario.getText (), password.getText ())) {
                    Intent holaAndroidIntent = new Intent (LoginActivity.this, HolaAndroid.class); startActivity (holaAndroidIntent);
                  } Else {

                        usuario.setText ( "");
               password.setText ( ""); }}});




    }
. . .
}
```

implicit activation

- When the activity you want to activate is not mentioned explicitly, Android needs to find which activities can attend the device Intent.

```
Intent geoIntent = new Intent (android.content.Intent.ACTION_VIEW,
                 Uri.parse ( "geo: 0.0 q =" + address)); startActivity
(geoIntent);
```

Resolution Intents



Resolution Intents

- Uses two types of information:
  - Intent that describes the desired operation
  - Intents filters that describe what operations can manage an activity.
    - Are normally specified in AndroidManifest.xml (can also be done programmatically)

- Data used in the resolution of Intents:
  - Action
  - Data (URI and Type)
  - Category

Intents filter specification

```
<Activity ...>
        <Intent-filter ...>
        . . .
        <Action android: name = "android.intent.action.DIAL" />
        . . .
        </ Intent-filter>
        . . .
</ Activity>
```

Intents filter specification

- In fact, all applications have an intent- filter:

```
<Activity android: name = ". LoginActivity">
    <Intent-filter>
        <Action android: name = "android.intent.action.MAIN" />

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

to be launched from the launcher Android Studio includes default AndroidManifest.xml

More information on the intent-filter

```
<Intent-filter ...>
        . . .
        <data
          android: mimeType = "string" android:
         scheme = "string" android: host = "string"
         android: port = "string" android: path =
         "string" android: pathPattern = "string"
         android: Pathprefix = "string" / >



        . . .
</ Intent-filter>
```

http://developer.android.com/intl/es/guide/components/intents-filters.html

Example

Filter for an activity that can display maps:

```
<Intent-filter ...>
        . . .
        <Data android: scheme = "geo" />
        . . .
</ Intent-filter>
```

Specifying a category filter

```
<Intent-filter ...>

        . . .

        <Category android: name = "string" />

        . . .

</ Intent-filter>
```

Google Maps filter

```
<Intent-filter ...>
    <Action android: name = "android.intent.action.VIEW" /> <category android: name =
  "android.intent.category.DEFAULT" /> <category android: name =
  "android.intent.category.BROWSABLE" / > <data android: scheme = "geo" /> </
  intent-filter>
```

Note: To receive implicit intents, the activity must include an intent-filter to category android.intent.category.DEFAULT
(So that it can be chosen as default application). All implicit intents are considered DEFAULT category.

Priority

```
<Intent-filter android: icon = "drawable resource"
                    android: label = "string resource" android:
                    priority = "integer">
     .   .   .
</ Intent-filter>
```

- **android: priority** Android helps to choose applications that can manage an Intent
- Between -1000 and 1000
- A higher value indicates a higher priority

Display a Chooser App

- If we want to give the user the option to choose which application to use to handle an intent, we must display a Chooser Dialog.

```
Intent sendIntent = new Intent (Intent.ACTION_SEND);
. . .
String title = getResources () getString (R.string.chooser_title).; // Create tried to show the chooser
dialog
Intent chooser = Intent.createChooser (sendIntent, title);

if (sendIntent.resolveActivity (getPackageManager ())! = null) {
        startActivity (chooser); }
```