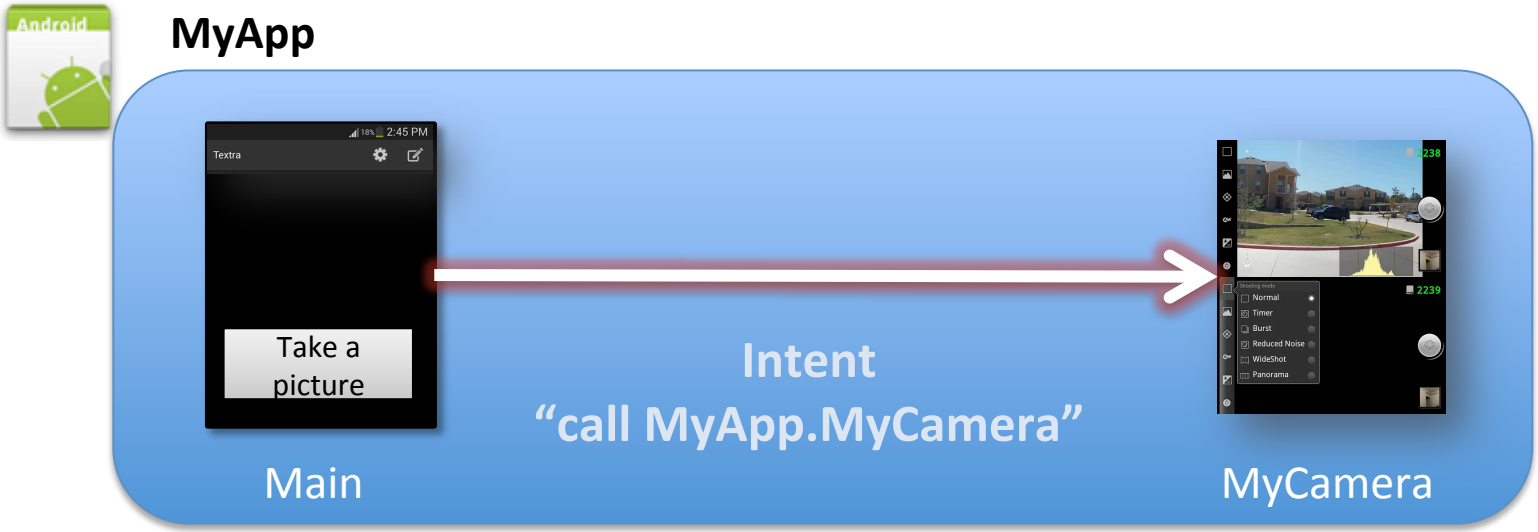
Android Intents

Intents

- Recap
- The class Intent
- Demos
  - Implicit Intents
  - Explicit Intents
  - Intents with or without result
- Intent filters

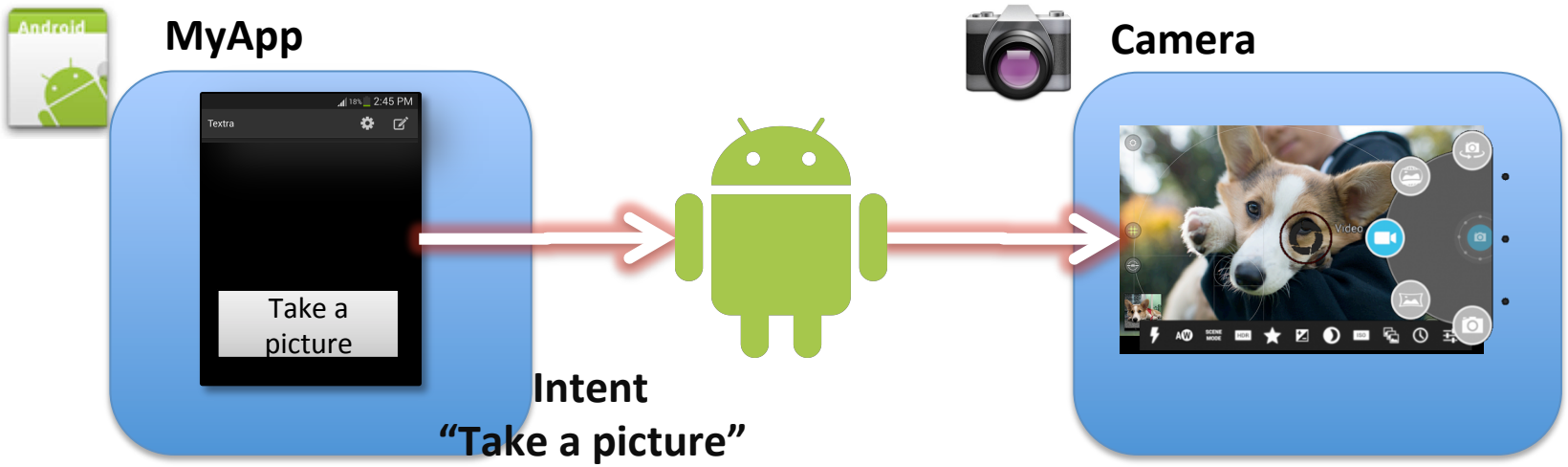
Intents – Recap

- Asynchronous messages which allow application components to **request functionality** from other Android components
- **Explicit Intent** when calling a specific Activity
  - i.e. calling another Activity of the same app



Intents – Recap

- Asynchronous messages which allow application components to **request functionality** from other Android components
- **Explicit Intent** when calling a specific Activity
- **Implicit Intent** when requesting a general action to perform
  - Call a component of another application



The intents

- Intents provide a flexible language for specifying operations to be performed
  - e.g., “Pick a contact”, “take a photo”, “dial a phone number”
- Intent is constructed by one Activity that wants some work done
- Received by one activity that can perform that work

- Intent are created as objects of the class `Intent`

http://developer.android.com/reference/android/content/Intent.html

Intents

- Recap
- **The class Intent**
- Demos
  - Implicit Intents
  - Explicit Intents
  - Intents with or without result
- Intent filters

Intent - Fields

Action

Data

Category

Type

Component

Extra

Flags

- A String representing desired operation

String	Action
ACTION_DIAL	Dial a number
ACTION_PICK	Pick an item from the data
ACTION_VIEW	Display data
ACTION_SEND	"share" data with other app
ACTION_MAIN	Start as initial activity of app

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

//Or

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


http://developer.android.com/reference/android/content/Intent.html

Intent - Fields

Action

Data

Category

Type

Component

Extra

Flags

- The data associated with the intent
- Formatted as a Uniform Resource Identifier URI
- Example of URI:

```
//For geolocation data
Uri.parse("geo:0,0?q=4+Rue+Camichel+Toulouse”);

//For a telephone number
Uri.parse("tel:+0534322177”);
```

- Set the data for the intent with `setData()`

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

newInt.setData(Uri.parse("tel:+0534322177"));
```


http://developer.android.com/reference/android/content/Intent.html

Intent - Fields

Action

Data

Category

Type

Component

Extra

Flags

- Additional information about the components that can handle the intent

- CATEGORY_BROWSABLE
  - It can be invoked by a browser to display data ref's by a URI
- CATEGORY_LAUNCHER
  - Can be the initial activity of a task and it is listed in system application launcher

- Set with `addCategory()`
- Most of the times it is not required

http://developer.android.com/reference/android/content/Intent.html

Intent - Fields

Action
Data
Category
Type
Component
Extra
Flags

- Specifies the MIME type of the Intent data
  - `"image/*"`,
  - `"image/png", .`
  - `"image/jpg"`
  - `"text/plain"`,
  - `"text/html"` …
- MIME = two-part String identifier for format contents, `type/subtype`

- `setType(String)` if no data is passed
- `setDataAndType(Uri, String)` if data is passed too

http://developer.android.com/reference/android/content/Intent.html

Intent - Fields

Action
Data
Category
Type
Component
Extra
Flags

- EXPLICIT Intents only
- The component that should receive this intent

```
Intent newInt =
    Intent(Context packageContext,
           Class<?> cls);
```

```
Intent newInt = new Intent(this, MyCamera.class);

//or

Intent newInt = new Intent();
newInt.setClass(this, MyCamera.class);
```


http://developer.android.com/reference/android/content/Intent.html

Intent - Fields

Action
Data
Category
Type
Component
Extra
Flags

- Key-value pairs with additional information required to accomplish the requested action.
- various `putExtra( key, value )`

```
Intent email = new Intent();
email.setAction( Intent.ACTION_SEND );
email.setData( Uri.parse("mailto:you@google.com") );
email.putExtra( Intent.EXTRA_EMAIL, "me@google.com" );
email.putExtra( Intent.EXTRA_SUBJECT, "Hello!");
email.putExtra( Intent.EXTRA_TEXT, "Dear Alice...");
```


http://developer.android.com/reference/android/content/Intent.html

Intent - Fields

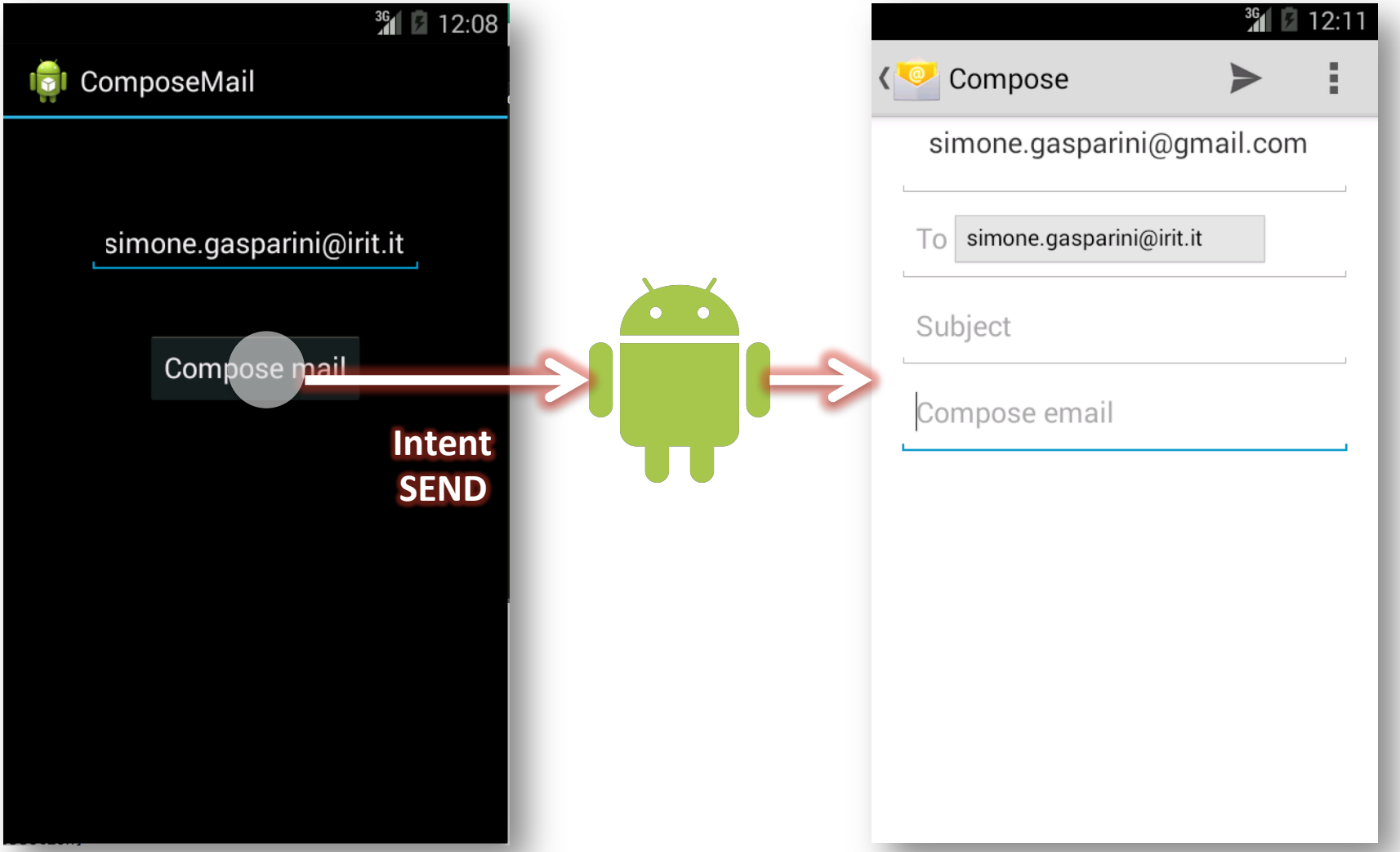
Action
Data
Category
Type
Component
Extra
Flags

- Specify how Intent should be handled
- Ex.
  - `FLAG_ACTIVITY_NO_HISTORY`
    Don't put this Activity in the History stack
  - `FLAG_DEBUG_LOG_RESOLUTION`
    Print extra logging information when this Intent is processed

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

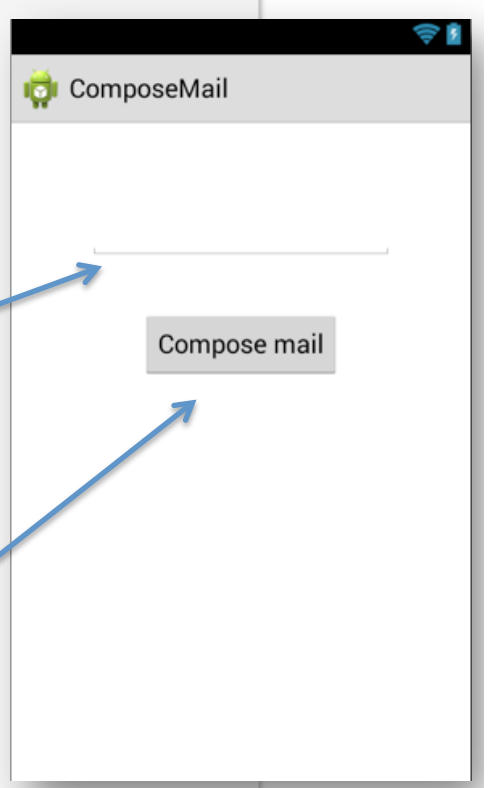

http://developer.android.com/reference/android/content/Intent.html

A simple example with SEND Intent



Layout file

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="${packageName}.${activityClass}" >

    <EditText
        android:id="@+id/address"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="59dp"
        android:ems="10"
        android:inputType="textEmailAddress" />

    <Button
        android:id="@+id/buttonCompose"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/address"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="35dp"
        android:text="@string/buttoncompose" />

</RelativeLayout>
```



activity_main.xml

The main activity

```
public class MainActivity extends Activity
{

    @Override
    protected void onCreate( Bundle savedInstanceState )
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // get the reference to the "Compose" button
        Button composeButton = (Button) findViewById( R.id.buttonCompose );

        // set a callback for the button
        composeButton.setOnClickListener( new View.OnClickListener()
        {
            public void onClick( View view )
            {
                // function that launch the intent
                sendEmail();
            }
        });
     }
    ...
```

The main activity

```
    protected void sendEmail()
    {
        Log.i("Send email", "");

        // get the reference to the edit text
        EditText address = (EditText) findViewById( R.id.address );

        // prepare the intent
        Intent emailIntent = new Intent( Intent.ACTION_SEND );
        // set data
        emailIntent.setData( Uri.parse("mailto:") );
        // set type
        emailIntent.setType( "text/plain" );

        String[] recipients = new String[]{ address.getText().toString() };

        emailIntent.putExtra( Intent.EXTRA_EMAIL, recipients );

        // start the activity
        startActivity( emailIntent );
        finish();
    }
}
```

Getting the address for the EditText widget

```
String[] recipients = new String[]{ address.getText().toString() };

emailIntent.putExtra( Intent.EXTRA_EMAIL, recipients );
```

public static final String EXTRA_EMAIL

A String[] holding e-mail addresses that should be delivered to.

Constant Value: "android.intent.extra.EMAIL"

public Intent putExtra (String name, String[] value) Added in API level 1

Add extended data to the intent. The name must include a package prefix, for example the app com.android.contacts would use names like "com.android.contacts.ShowAll".

Parameters

name	The name of the extra data, with package prefix.
value	The String array data value.

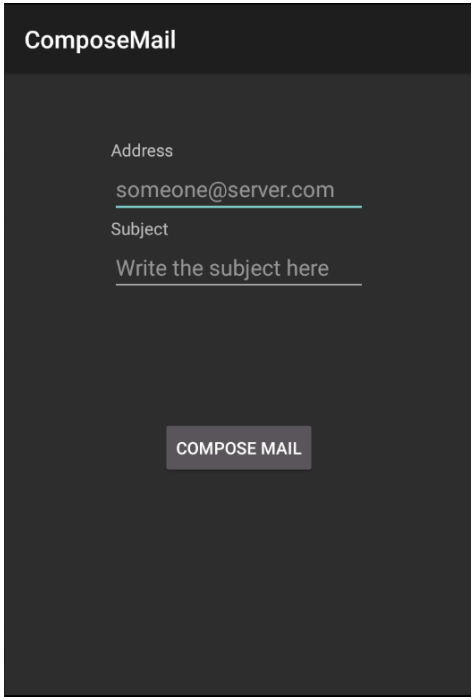
Returns

Returns the same Intent object, for chaining multiple calls into a single statement.

Adding the Subject - EXTRA_SUBJECT

- To add the subject field as an extra to the intent

```
// get the widget containing the subject
EditText subject = (EditText) findViewById(R.id.editTextSubject);
.
.
.
// add the subject field as extra
emailIntent.putExtra(Intent.EXTRA_SUBJECT, subject.getText().toString());
```



Starting activities with intents

- There are 2 ways to start an activity:

- `startActivity(Intent intent, ...)`
  - No result is expected from the called activity
  - Eg. Show a place on GMaps, compose/send a mail

- `startActivityForResult(Intent intent, ...)`
  - If the called activity has to return a result
  - Eg. Select an image, a video, get a contact information…
  - The result is received as a separate intent object
  - Implement the Activity method `onActivityResult()`

Intents

- Recap
- The class Intent
- **Demos**
  - Implicit Intents
  - Explicit Intents
  - Intents with or without result
- Intent filters

Learning by example

	Implicit Intent	Explicit Intent
Start Activity	Simple Camera Intent	Intent Demo
Start Activity with result	Camera Intent with Result	Intent Demo

Check Moodle for the source code.

(`New->Other->Android project from existing code` to import them to Android Studio)

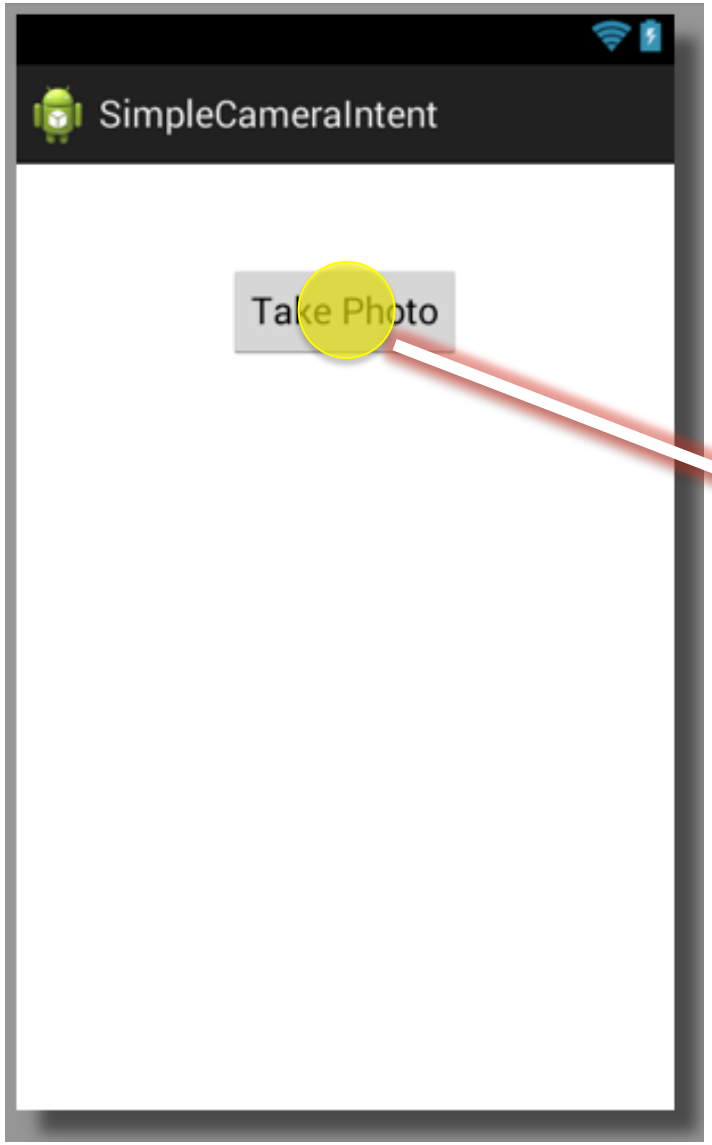
Learning by example

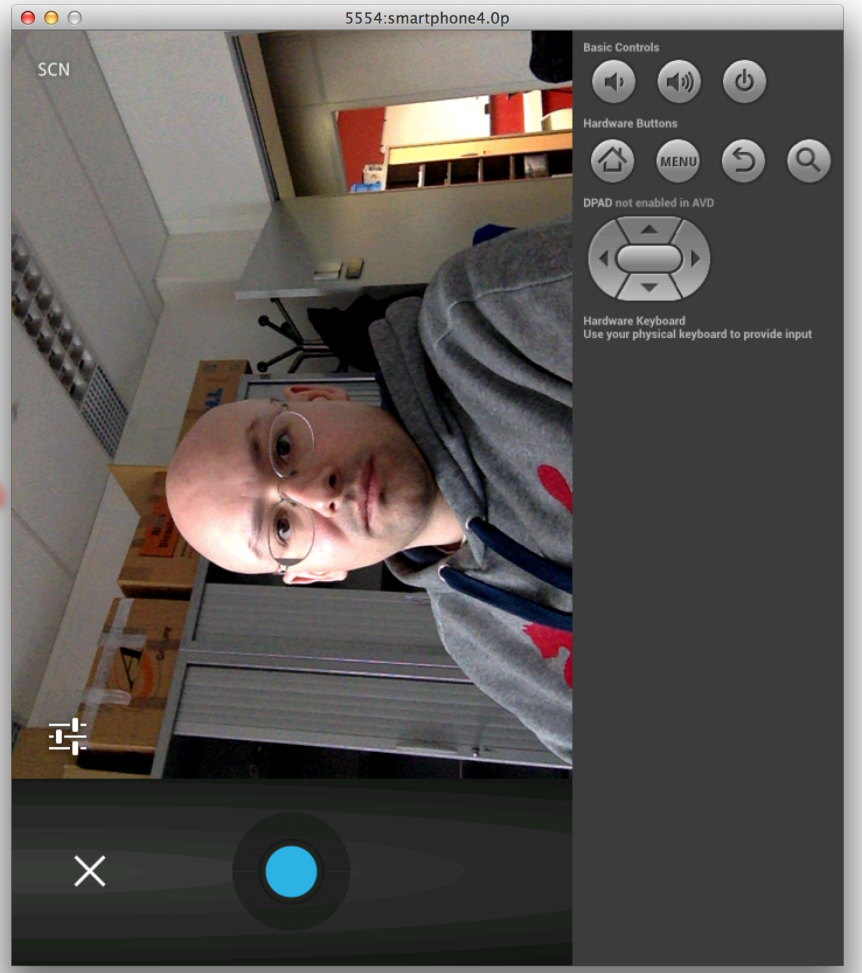
	Implicit Intent	Explicit Intent
Start Activity	Simple Camera Intent	Intent Demo
Start Activity with result	Camera Intent with Result	Intent Demo

Check Moodle for the source code.

(`New->Other->Android project from existing code` to import them to Eclipse)

Simple Camera Intent





Android Camera app

The layout file

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".MainActivity" >

    <Button
        android:id="@+id/shotbutton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="32dp"
        android:text="@string/takephoto" />

</RelativeLayout>
```

activity_main.xml

The code

```
public class MainActivity extends Activity implements OnClickListener
{

     private Button mShotButton;

     @Override
     protected void onCreate( Bundle savedInstanceState )
     {
          super.onCreate( savedInstanceState );
          setContentView( R.layout.activity_main );
          // get the reference to the button from the resources
          mShotButton = (Button) findViewById( R.id.shotbutton );
          // set the call back listener
          mShotButton.setOnClickListener( this );

     }

     @Override
     public void onClick( View v )
     {
          // new intent to capture a photo
          Intent newInt = new Intent(android.provider.MediaStore.ACTION_IMAGE_CAPTURE);
          startActivity( newInt );
     }

}
```

Implicit activation

```
@Override
public void onClick( View v )
{
    // new intent to capture a photo
    Intent newInt = new Intent(android.provider.MediaStore.ACTION_IMAGE_CAPTURE);
    startActivity( newInt );
}
```

- When the Activity to be activated is not explicitly named, Android tries to find Activities that match the Intent
- This process is called **intent resolution**

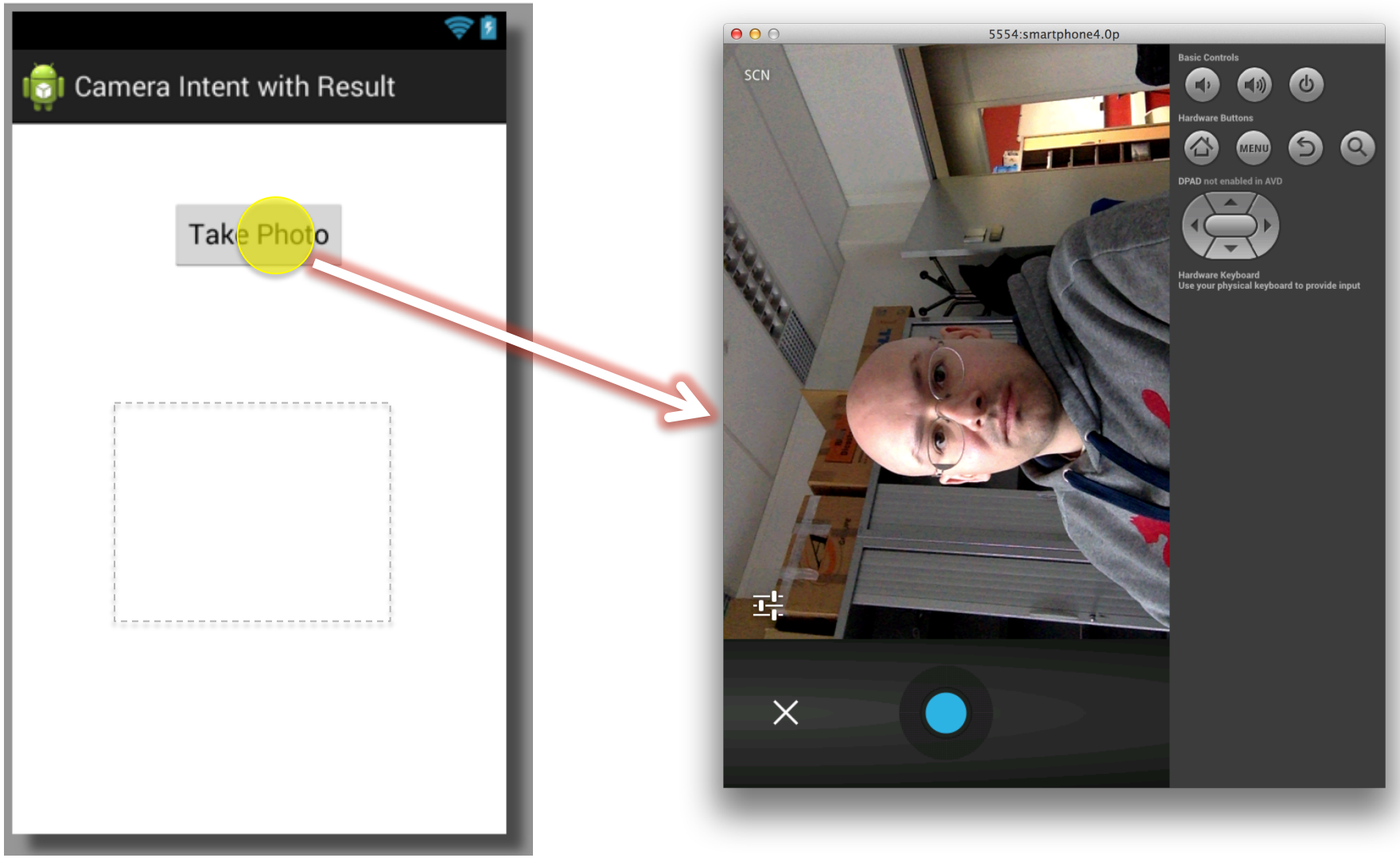
Learning by example

	Implicit Intent	Explicit Intent
Start Activity	Simple Camera Intent	Intent Demo
Start Activity with result	Camera Intent with Result	Intent Demo

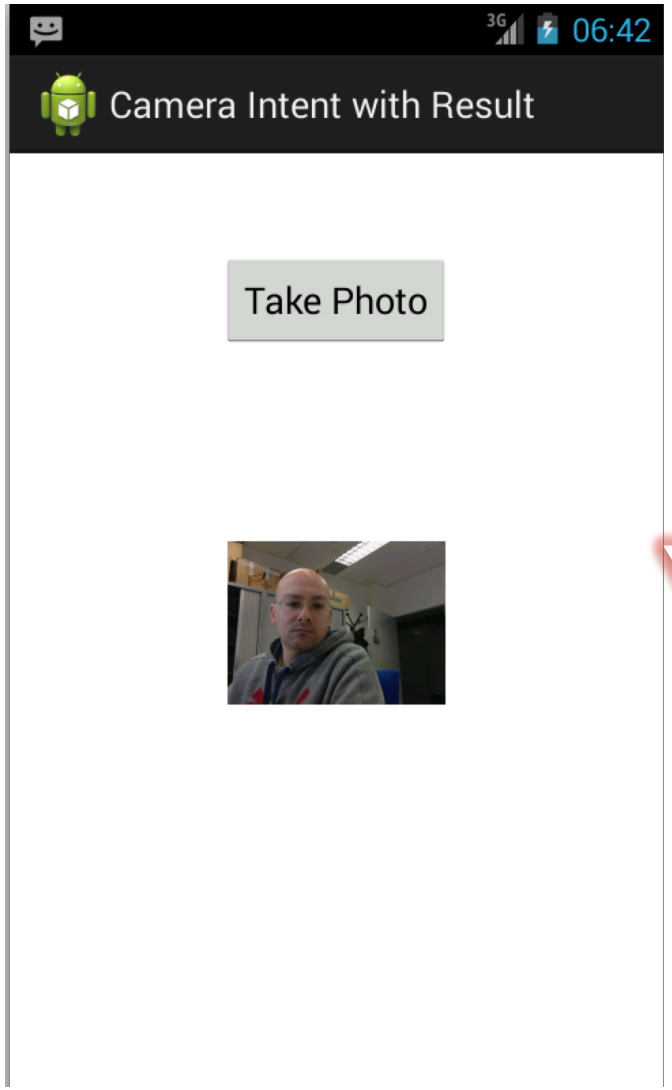
Check Moodle for the source code.

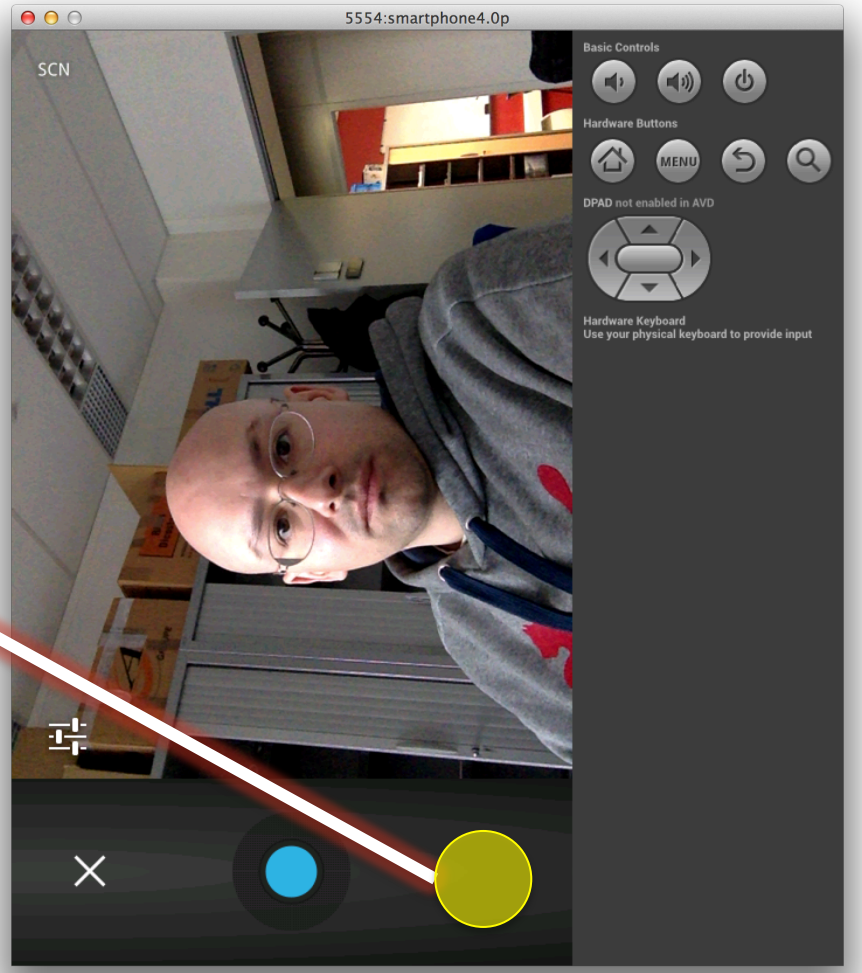
(`New->Other->Android project from existing code` to import them to Eclipse)

Camera Intent with Result



Camera Intent with Result





The layout file

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".MainActivity" >

    <Button
        android:id="@+id/shotbutton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="32dp"
        android:text="@string/takephoto" />

    <ImageView
        android:id="@+id/ReturnedImageView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true" />

</RelativeLayout>
```

activity_main.xml

The code

```
 public class MainActivity extends Activity implements OnClickListener
{

        private ImageView imv;

        private Button mShotButton;

        final static int CAMERA_RESULT = 99;

        @Override
        protected void onCreate(Bundle savedInstanceState)
        {
                super.onCreate( savedInstanceState );
                setContentView( R.layout.activity_main );

                mShotButton = (Button) findViewById( R.id.shotbutton );

                mShotButton.setOnClickListener( this );
        }

        @Override
        public void onClick( View v )
        {
                Intent newInt = new Intent(android.provider.MediaStore.ACTION_IMAGE_CAPTURE);
                startActivityForResult(newInt, CAMERA_RESULT);
        }
        @Override
        protected void onActivityResult(int requestCode, int resultCode, Intent intent)
        {
                super.onActivityResult( requestCode, resultCode, intent );

                if ( resultCode == RESULT_OK && requestCode == CAMERA_RESULT )
                {
                        Bundle extras = intent.getExtras();
                        Bitmap bmp = (Bitmap) extras.get( "data" );

                        imv = (ImageView) findViewById( R.id.ReturnedImageView );
                        imv.setImageBitmap( bmp );
                }
        }

}
```

Starting the activity for result

```
final static int CAMERA_RESULT = 99;

...

@Override
public void onClick( View v )
{
    Intent newInt = new Intent(android.provider.MediaStore.ACTION_IMAGE_CAPTURE);

    // start the activity using the intent and a request code
    startActivityForResult( newInt, CAMERA_RESULT );
}
```

Getting the result

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent intent)
{
    super.onActivityResult( requestCode, resultCode, intent );

    // if the request code is the same as the one used to start the activity
    // and the user has actually taken a picture
    if ( resultCode == RESULT_OK && requestCode == CAMERA_RESULT )
    {

        // get the image from the "extras" of the result intent
        Bundle extras = intent.getExtras();
        Bitmap bmp = (Bitmap) extras.get( "data" );

        // set the result image to the ImageView widget
        imv = (ImageView) findViewById( R.id.ReturnedImageView );
        imv.setImageBitmap( bmp );
    }
}
```

Bundle is a mapping between a `String` (the key) and any value (`int, float etc`)

Learning by example

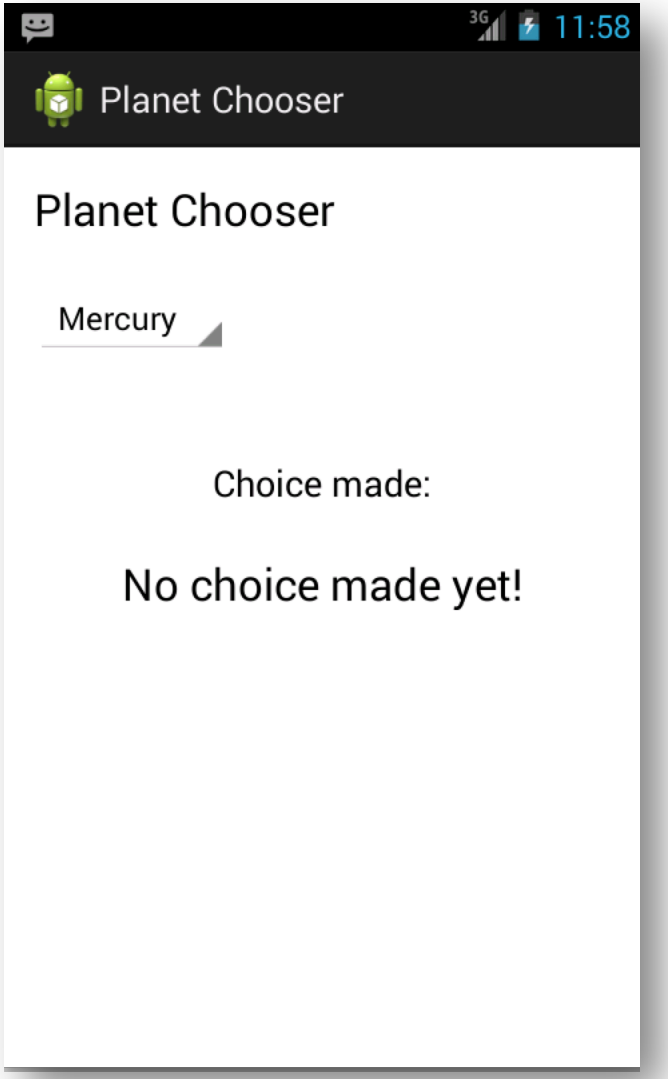
	Implicit Intent	Explicit Intent
Start Activity	Simple Camera Intent	Intent Demo
Start Activity with result	Camera Intent with Result	Intent Demo

Check Moodle for the source code.

(`New->Other->Android project from existing code` to import them to Eclipse)

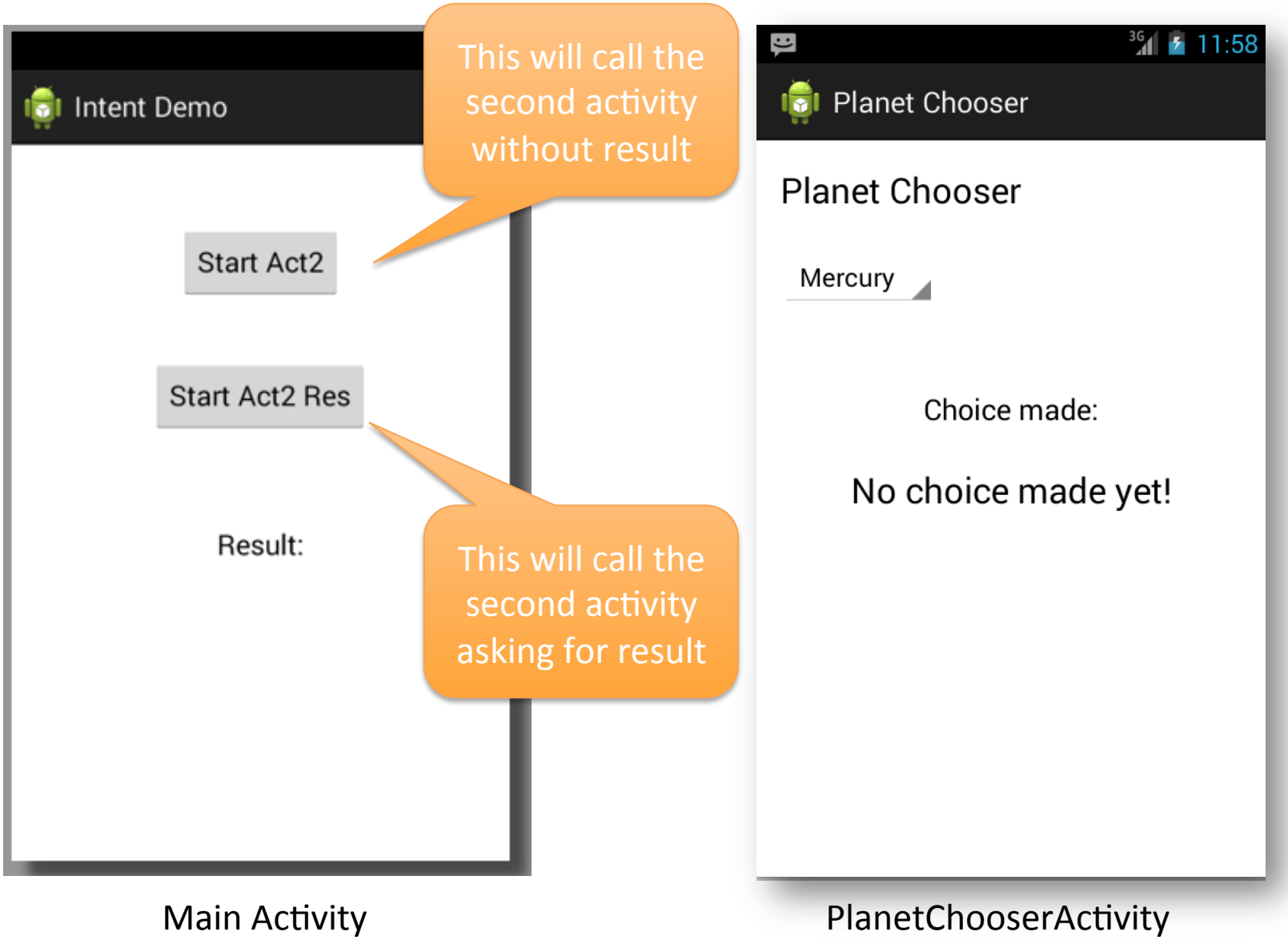
A demo application with 2 activities

Main Activity



PlanetChooserActivity

A demo

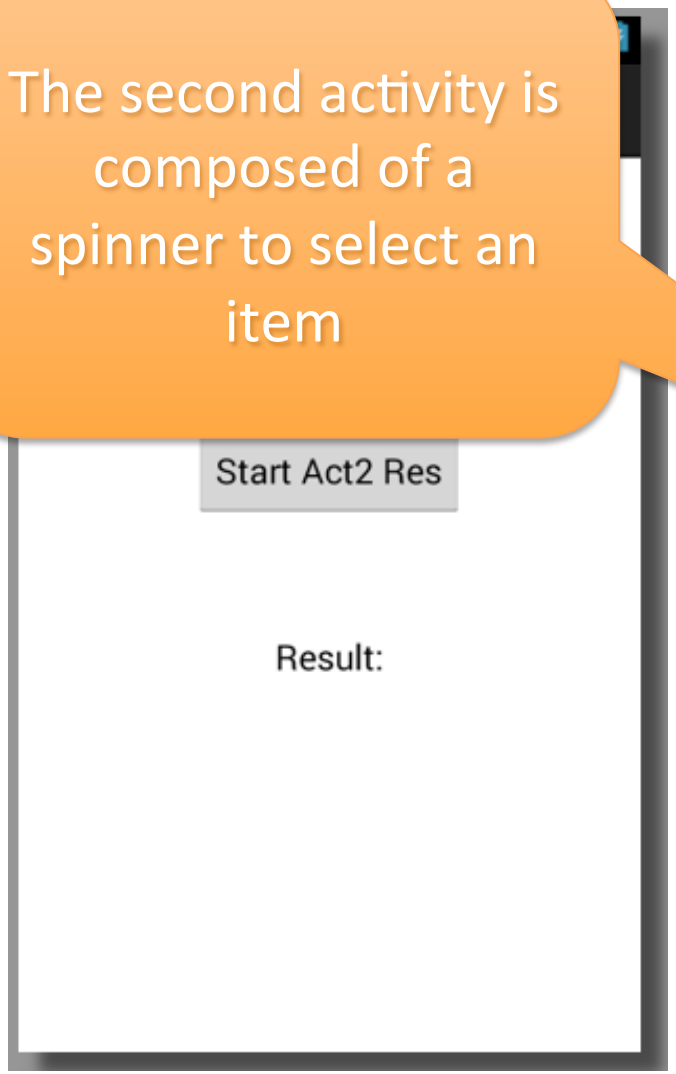


Main Activity

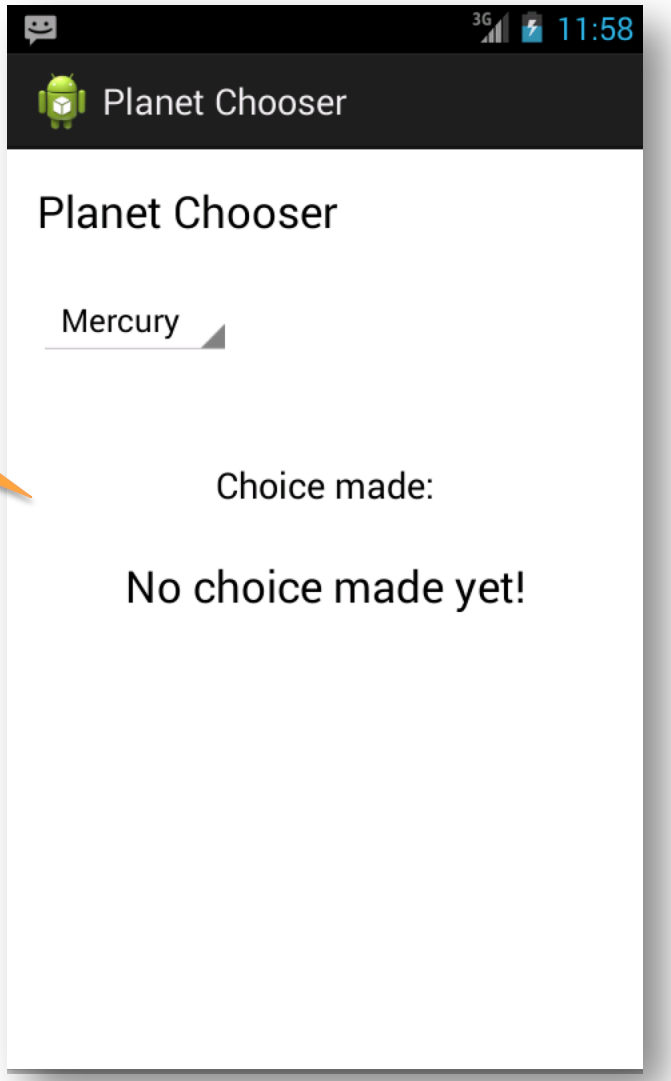
PlanetChooserActivity

A demo

The second activity is composed of a spinner to select an item



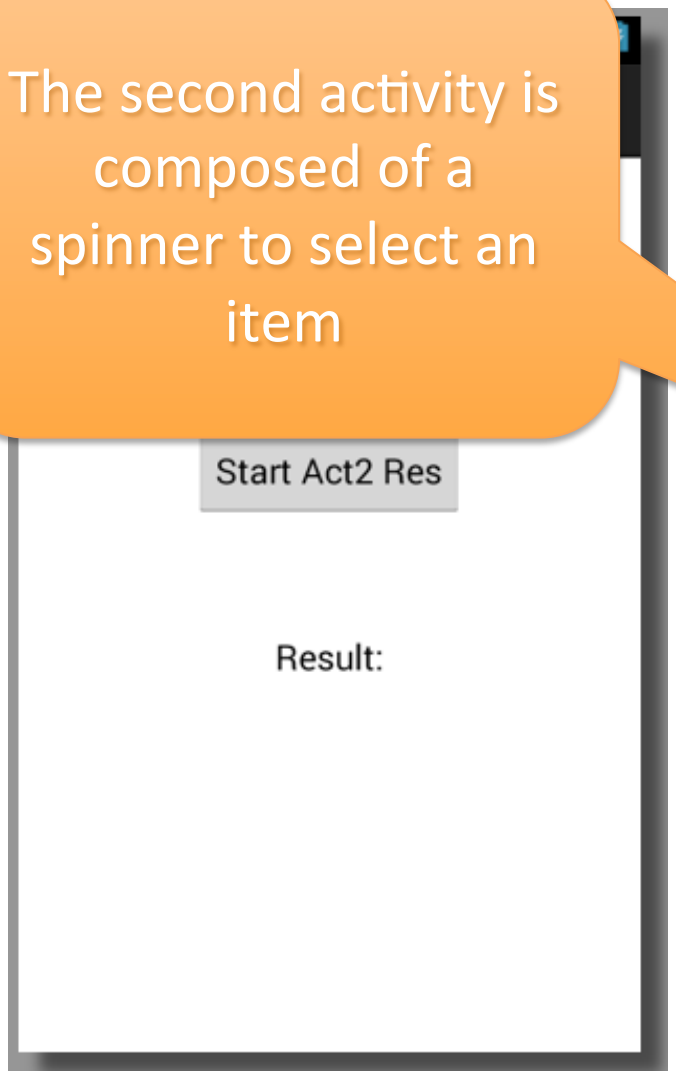
Main Activity



PlanetChooserActivity

A demo

The second activity is composed of a spinner to select an item

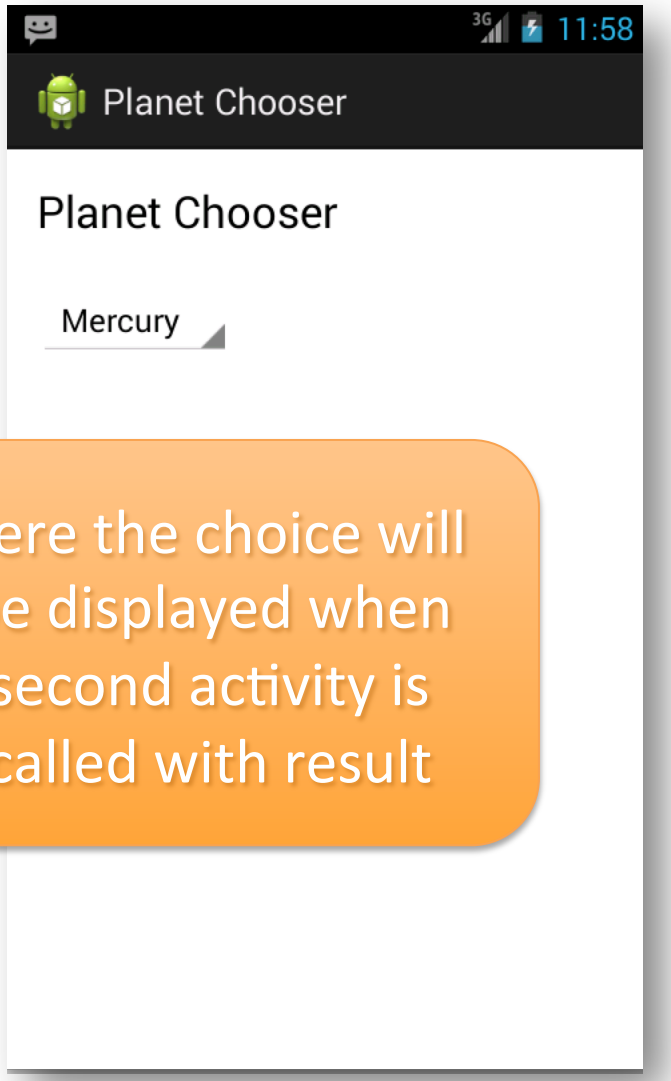


Main Activity

PlanetChooserActivity

A demo

Main Activity



PlanetChooserActivity

Here the choice will be displayed when second activity is called with result

Explicit Intent without result

MainActivity.java

```
@Override
protected void onCreate(Bundle savedInstanceState)
{
    super.onCreate( savedInstanceState );
    setContentView( R.layout.activity_main );

    // get the reference of all the graphical widget
    mSimplyStartButton = (Button) findViewById( R.id.simplybutton );
    mResultStartButton = (Button) findViewById( R.id.resultbutton );

    mResultView = (TextView) findViewById( R.id.resultFromActivityView );

    // set the listener for the two buttons
    // we use the same listener callback for both buttons, we will check inside
    // which one has been pressed
    mSimplyStartButton.setOnClickListener( this );
    mResultStartButton.setOnClickListener( this );

}
```

Explicit Intent without result

MainActivity.java

```
@Override
public void onClick( View v )
{
    // if the button to launch the activity has been pressed
    if ( v.getId() == mSimplyStartButton.getId() )
    {
        Log.i( TAG, "Button Simply pressed" );
        // set the explicit intent and launch the activity
        Intent newInt = new Intent( this, PlanetChooserActivity.class );
        startActivity( newInt );
    }
```

```
Intent( Context packageContext, Class<?> cls )
```

- The Context is an Interface used to access global application information
- The class activity is a subclass of Context hence we can pass "**this**"

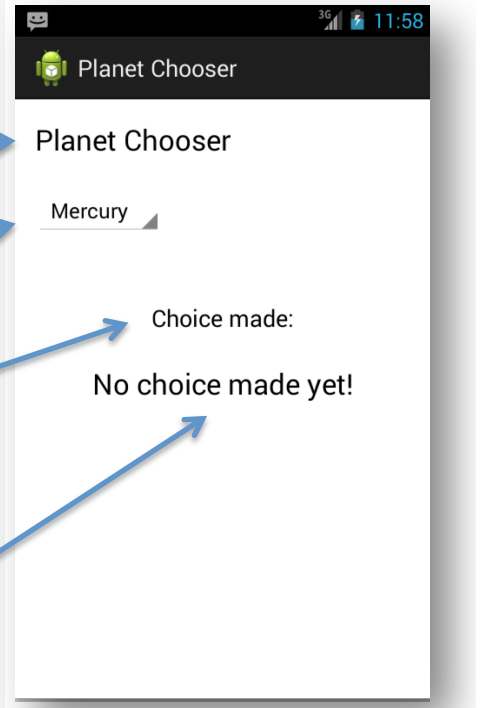
https://developer.android.com/reference/android/content/Intent.html

Explicit intent, both versions

```
 @Override
public void onClick(View v)
{
    // if the button to launch the activity has been pressed
    if ( v.getId() == mSimplyStartButton.getId() )
    {
        Log.i( TAG, "Button Simply pressed" );
        // set the explicit intent and launch the activity
        Intent newInt = new Intent( this, PlanetChooserActivity.class );
        startActivity( newInt );
    }
    // if the button to launch the activity with result has been pressed
    else if ( v.getId() == mResultStartButton.getId() )
    {
        Log.i( TAG, "Button Result pressed" );
        // set the explicit intent and launch the activity
        Intent newInt = new Intent( this, PlanetChooserActivity.class );
        newInt.setAction( Intent.ACTION_PICK );
        startActivityForResult( newInt, GET_PLANET );
    }

}
```

The PlanetChooserActivity

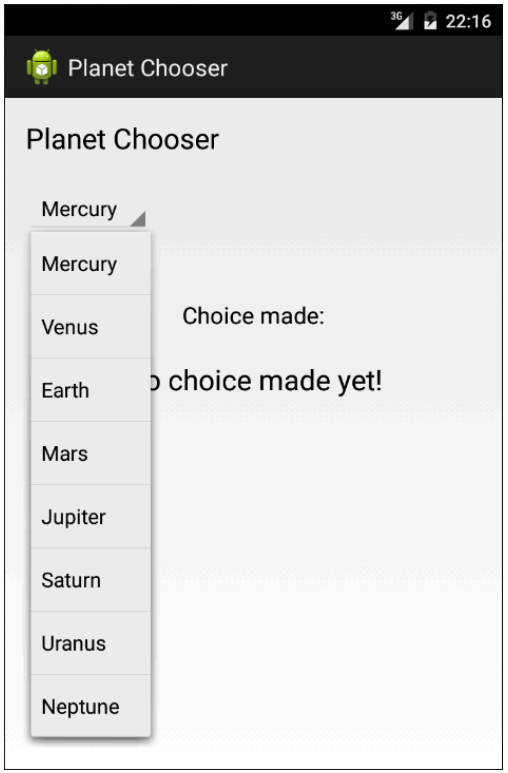
```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools=http://schemas.android.com/tools
    tools:context=".PlanetChooserActivity" >

    <TextView
        android:id="@+id/textViewResult”
        android:text="@string/title_activity_second"
        android:textAppearance="?android:attr/textAppearanceLarge" />

    <Spinner
        android:id="@+id/spinner1”
        android:entries="@array/planets_array"
        android:textAlignment="center" />

    <TextView
        android:id="@+id/textView2”
         android:text="@string/choicetext"
        android:textAppearance="?android:attr/textAppearanceMedium" />

    <TextView
        android:id="@+id/resultText”
        android:text="@string/choicemadedef”
        android:textAppearance="?android:attr/textAppearanceLarge" />


</RelativeLayout>
```



The PlanetChooserActivity

```
<Spinner
    android:id="@+id/spinner1"
    android:entries="@array/planets_array"
    android:textAlignment="center" />
```

```
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <string name="app_name">Intent Demo</string>
...
    <string name="title_activity_second">Planet Chooser</string>
    <string name="choicetext">Choice made:</string>
    <string name="choicemadedef">No choice made yet!</string>

    <string-array name="planets_array">
        <item>Mercury</item>
        <item>Venus</item>
        <item>Earth</item>
        <item>Mars</item>
        <item>Jupiter</item>
        <item>Saturn</item>
        <item>Uranus</item>
        <item>Neptune</item>
    </string-array>

</resources>
```



The PlanetChooserActivity

```
public class PlanetChooserActivity extends Activity
                                    implements OnItemSelectedListener
{
    private final static String TAG = "SecondActivity";
    private Spinner mSpinner;
    private TextView mResultText;
    String[] mPlanets;

    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate( savedInstanceState );
        setContentView( R.layout.activity_second );

        // get the reference of all the graphical widget
        mSpinner = (Spinner) findViewById( R.id.spinner1 );
        mResultText = (TextView) findViewById( R.id.resultText );
        // get the list of the planet from the string array in resources
        mPlanets = getResources().getStringArray(R.array.planets_array);

        // set the callback for the spinner
        mSpinner.setOnItemSelectedListener( this );
    }
```

The spinner listener

```
@Override
public void onItemSelected(AdapterView<?> parent, View arg1, int pos, long arg3)
{
    // check if we are running because of an intent
    if( getIntent().getAction() == Intent.ACTION_PICK )
    {
        Log.i( TAG, "Activity called for result with ACTION_PICK" );

        // create the result intent
        Intent resultIntent = new Intent();
        // add the result as an extra
        resultIntent.putExtra( "ChoosenPlanet", mPlanets[pos] );
        // set the result as positive
        setResult(Activity.RESULT_OK, resultIntent);

        finish();
    }
    else
        Log.i( TAG, "Activity called with "+getIntent().getAction() );

    mResultText.setText( mPlanets[pos] );

}
```

Back to the main activity

MainActivity.java

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data)
{
    super.onActivityResult( requestCode, resultCode, data );

    // if the request code is the same we used to start the activity and there is
    // a positive result
    if ( requestCode == GET_PLANET && resultCode == RESULT_OK )
    {
        mResultView.setText( data.getExtras().getString( "ChoosenPlanet" ) );
    }

    if(resultCode != RESULT_OK)
    {
        Log.i( TAG, "Returned without ok result" );
    }
}
```

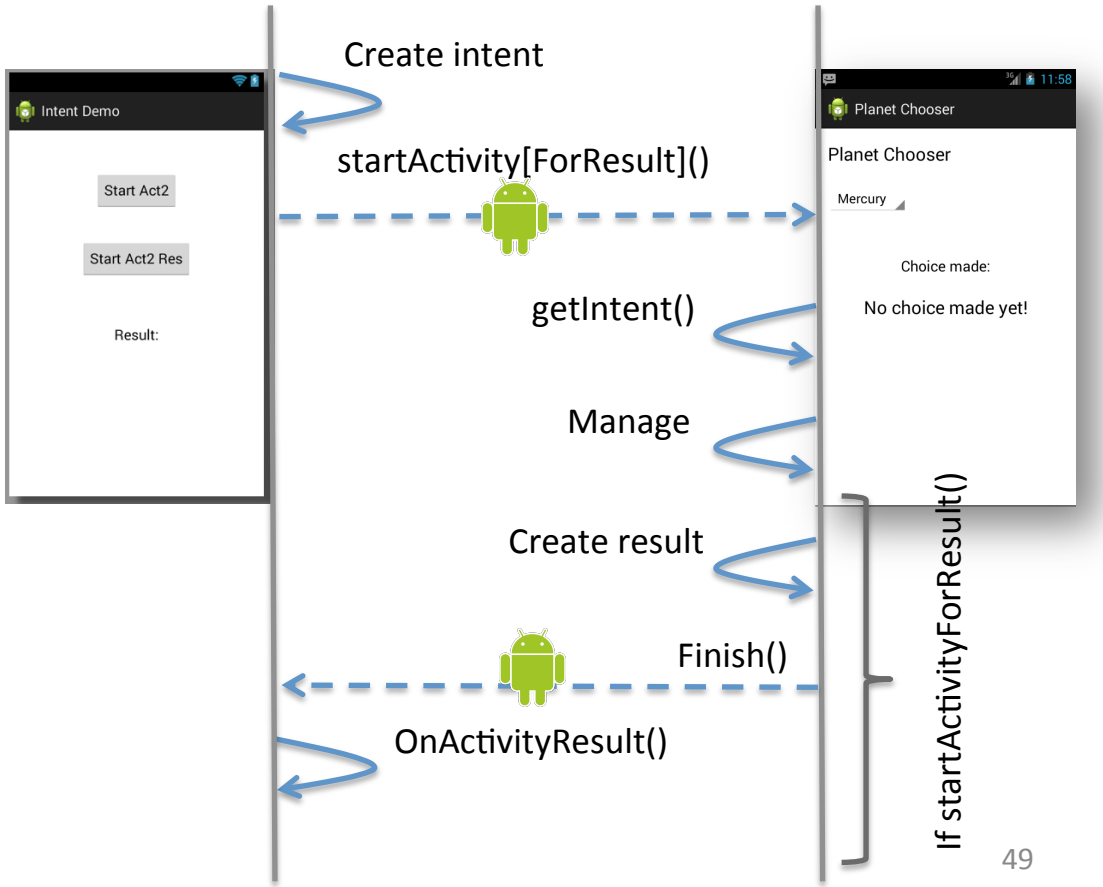
To recap

On the caller

- Create the Intent
  - Set the Action, Data (and Component if explicit)
- Start activity
  - startActivity()
  - startActivityForResult()
- Get the result
  - OnActivityResult()

On the callee

- Get the intent
  - getIntent()
- Manage
- Create result
  - Intent, setResult()



Intents

- Recap
- The class Intent
- Demos
  - Implicit Intents
  - Explicit Intents
  - Intents with or without result
- **Intent filters**

Intent filters

- To advertise which implicit intents your app can receive, declare one or more **intent filters** in the manifest file.
- Each intent filter specifies the type of intents in terms of
  - action,
  - data,
  - category.
- The intent is delivered to the app only if the intent pass through one of the filters.
- Explicit intent is always delivered, regardless the filters

Filter example

AndroidManifest.xml

```
...
<activity android:name="ShareActivity">
    <intent-filter>
        <action android:name="android.intent.action.SEND"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:mimeType="text/plain"/>
    </intent-filter>
</activity>
```

- `<action>` contains the action the activity can respond to
- `<category>` has to be DEFAULT to receive implicit intents
- `<data>` contains the information about the MIME and URI

More filters

```
<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>

<activity android:name="ShareActivity">
    <!-- This activity handles "SEND" actions with text data -->
    <intent-filter>
        <action android:name="android.intent.action.SEND"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:mimeType="text/plain"/>
    </intent-filter>
    <!-- This activity also handles "SEND" and "SEND_MULTIPLE" with media data -->
    <intent-filter>
        <action android:name="android.intent.action.SEND"/>
        <action android:name="android.intent.action.SEND_MULTIPLE"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:mimeType="application/vnd.google.panorama360+jpg"/>
        <data android:mimeType="image/*"/>
        <data android:mimeType="video/*"/>
    </intent-filter>
</activity>
```

More filters

```
<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>

<acti

</activity>
```

- *MainActivity*, is the app's main entry point
  - the activity that opens when the user initially launches the app with the launcher icon:
- *ACTION_MAIN* indicates this is the main entry point and does not expect any intent data.
- The *CATEGORY_LAUNCHER* category indicates that this activity's icon should be placed in the system's app launcher

More filters

- *SharedActivity* allows to share media and text content
- It can be called both from main activity and another app

```
<a
</activity>

<activity android:name="ShareActivity">
    <!-- This activity handles "SEND" actions with text data -->
    <intent-filter>
        <action android:name="android.intent.action.SEND"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:mimeType="text/plain"/>
    </intent-filter>
    <!-- This activity also handles "SEND" and "SEND_MULTIPLE" with media data -->
    <intent-filter>
        <action android:name="android.intent.action.SEND"/>
        <action android:name="android.intent.action.SEND_MULTIPLE"/>
        <category android:name="android.intent.category.DEFAULT"/>
        <data android:mimeType="application/vnd.google.panorama360+jpg"/>
        <data android:mimeType="image/*"/>
        <data android:mimeType="video/*"/>
    </intent-filter>
</activity>
```

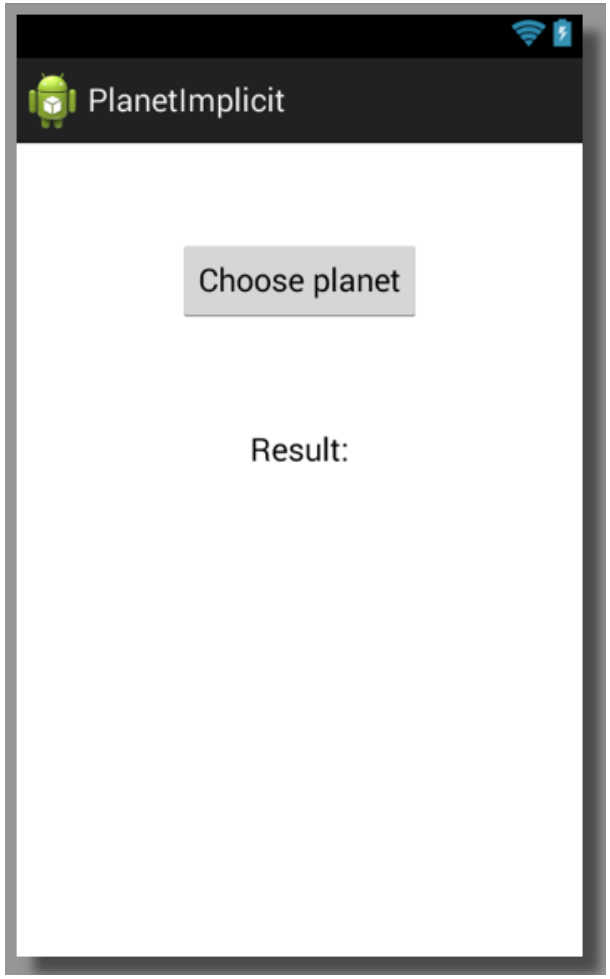
Back to our demo…

- If we want to make the second Activity available for other applications…
- Just declare the action and the data in the manifest file:

```
<application
     <activity
     android:name="fr.enseeiht.gasparini.intentdemo.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="fr.enseeiht.gasparini.intentdemo.PlanetChooserActivity"
      android:label="@string/title_activity_second" >
      <intent-filter>
          <action android:name="android.intent.action.PICK"/>
          <category android:name="android.intent.category.DEFAULT"/>
          <data android:mimeType="text/plain"/>
      </intent-filter>
     </activity>
</application>
```

A new app…

- We can write **a new app** that creates an intent to choose a planet



The code

```
@Override
public void onClick( View v )
{
    if ( v.getId() == mResultStartButton.getId() )
    {
        Log.i( TAG, "Button Result pressed" );
        // set the implicit intent and launch the activity
        Intent newInt = new Intent();
        newInt.setAction( Intent.ACTION_PICK );
        newInt.setType( "text/plain" );
        startActivityForResult( newInt, GET_PLANET );
    }

}
```

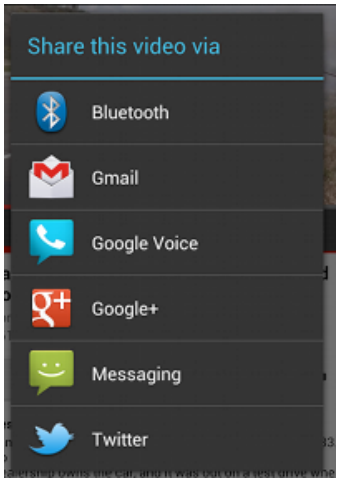
A more elegant way

```
@Override
public void onClick( View v )
{
    Log.i( TAG, "Button Result pressed" );
    // set the implicit intent and launch the activity
    Intent newInt = new Intent();
    newInt.setAction( Intent.ACTION_PICK );
    newInt.setType( "text/plain" );

    Intent chooser = Intent.createChooser(newInt,
                "App for choosing a planet");

    // Verify the intent will resolve to at least one activity
    if (chooser.resolveActivity(getPackageManager()) != null) {
            startActivityForResult( chooser, GET_PLANET );
    }
}
```

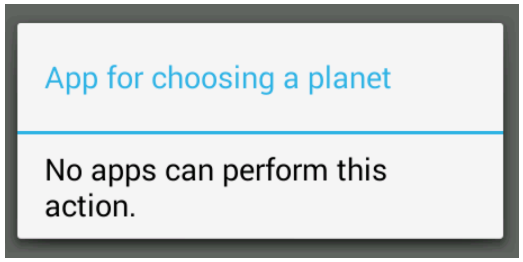
A more elegant way

```
Intent chooser = Intent.createChooser(newInt,
                 "App for choosing a planet");

// Verify the intent will resolve to at least one activity
if (chooser.resolveActivity(getPackageManager()) != null)
{
        startActivityForResult( chooser, GET_PLANET );
```

This verifies that there is at least an application that can handle the action



A "chooser"



No application can handle the intent