Android Developer Fundamentals

Activities and Intents

Lesson 2

Contents

Introduction

- ❑ What is an Activity ?
- ❑ What is an Intent ?

What is an Activity?

- An `Activity` is an application component
- Represents one window, one hierarchy of views
- Typically fills the screen, but can be embedded in other activity or a appear as floating window
- Java class, typically one activity in one file

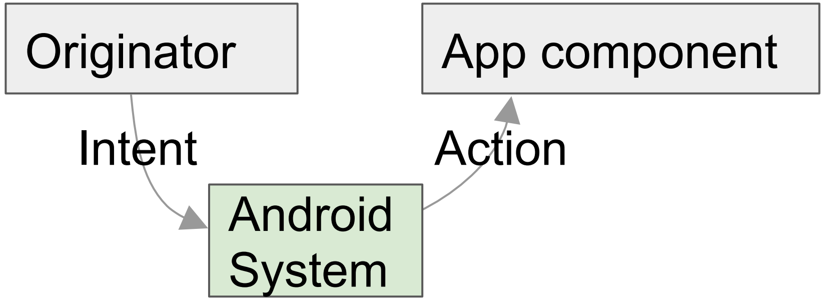
Implement new activities

1. Define layout in XML
2. Define Activity Java class
    - extends AppCompatActivity
3. Connect Activity with Layout
    - Set content view in onCreate()
4. Declare Activity in the Android manifest

What is an intent?

An intent is a description of an operation to be performed.

An Intent is an object used to request an action from another app component via the Android system.



What can intents do?

- Start activities
  - A button click starts a new activity for text entry
  - Clicking Share opens an app that allows you to post a photo
- Start services
  - Initiate downloading a file in the background
- Deliver broadcasts
  - The system informs everybody that the phone is now charging

Services and Broadcasts are covered in later lessons

Explicit and implicit intents

Explicit Intent

- Starts a specific activity
    - Request tea with milk delivered by Nikita
    - Main activity starts the ViewShoppingCart activity

Implicit Intent

- Asks system to find an activity that can handle this request
    - Find an open store that sells green tea
    - Clicking Share opens a chooser with a list of apps

Explicit Intents

- ❑ Start an Activity with an explicit intent.
- ❑ Sending & receiving data
- ❑ Activity Stack
- ❑ Returning data

Start an Activity with an explicit intent

To start a specific activity, use an explicit intent

1. Create an intent
   - `Intent intent = new Intent(this, ActivityName.class);`
2. Use the intent to start the activity
   - `startActivity(intent);`

Sending and retrieving data

In the first (sending) activity:

1. Create the Intent object
2. Put data or extras into that intent
3. Start the new activity with `startActivity()`

In the second (receiving) activity,:

1. Get the intent object the activity was started with
2. Retrieve the **data** or **extras** from the Intent object

Two types of sending data with intents

- Data—one piece of information whose data location can be represented by an URI
- Extras—one or more pieces of information as a collection of key-value pairs in a Bundle

Putting a URI as intent data

```
// A web page URL
intent.setData(
    Uri.parse("http://www.google.com"));

// a Sample file URI
intent.setData(
     Uri.fromFile(new File("/sdcard/sample.jpg")));
```

Put information into intent extras

- `putExtra(String name, int value)`
  ⇒ `intent.putExtra("level", 406);`
- `putExtra(String name, String[] value)`
  ⇒ `String[] foodList = {"Rice", "Beans", "Fruit"};`
  `intent.putExtra("food", foodList);`
- `putExtras(bundle);`
  ⇒ if lots of data, first create a bundle and pass the bundle.
- See documentation for all

Sending data to an activity with extras

```
public static final String DATA_KEY = "MESSAGE";

Intent intent = new Intent(this, SecondActivity.class);
String value = "Hello Activity!";
intent.putExtra(DATA_KEY, value);

startActivity(intent);
```

Get data from intents

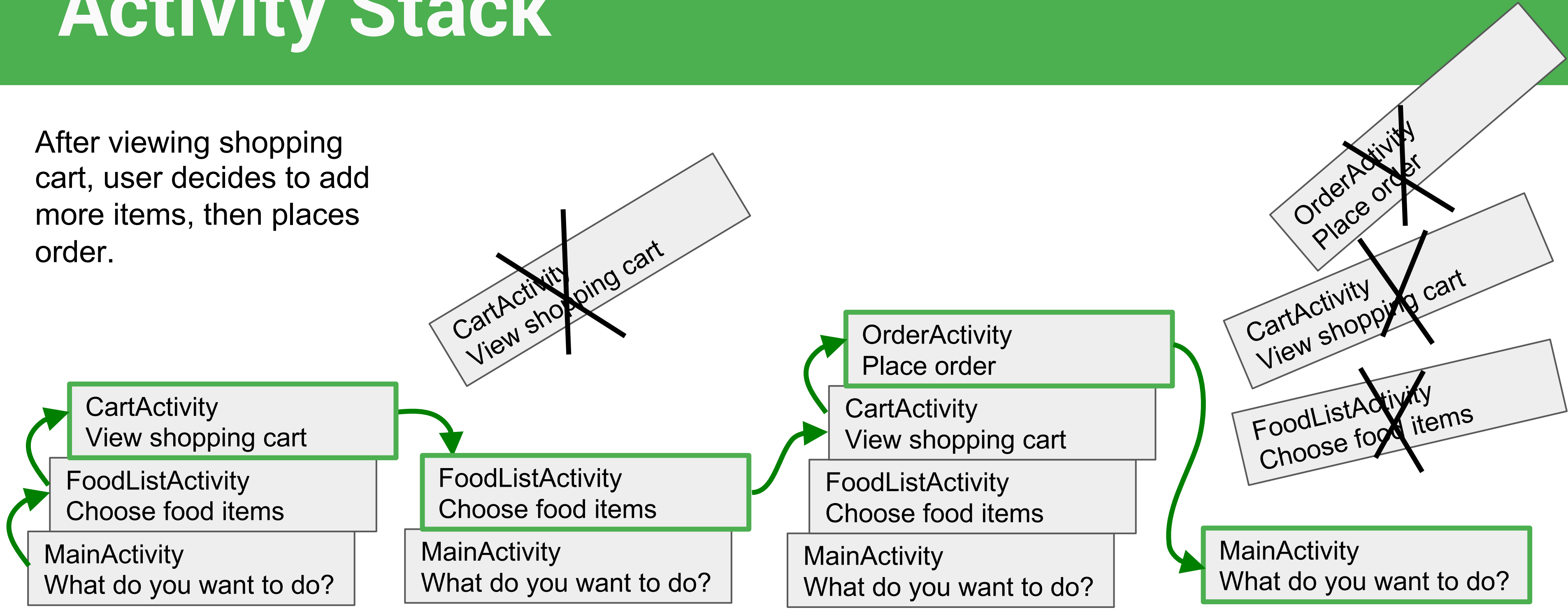
- `getData();`
  ⇒ `Uri locationUri = intent.getData();`
- `int getIntExtra (String name, int defaultValue)`
  ⇒ `int level = intent.getIntExtra("level", 0);`
- `Bundle bundle = intent.getExtras();`
  ⇒ Get all the data at once as a bundle.
- See documentation for all

Activity stack

- When a new activity is started, the previous activity is stopped and pushed on the activity back stack
- Last-in-first-out-stack—when the current activity ends, or the user presses the B◁k button, it is popped from the stack and the previous activity resumes

Activity Stack

After viewing shopping cart, user decides to add more items, then places order.



Returning data

In first activity

1. Use `startActivityForResult()` to start the second activity
2. Implement `onActivityResult()` to receive the returned data from second activity

startActivityForResult()

`startActivityForResult(intent, requestCode);`

- Starts activity (`intent`), assigns it identifier (`requestCode`)
- Returns data via intent extras
- When done, pop stack, return to previous activity, and execute `onActivityResult()` callback to process returned data
- Use `requestCode` to identify which activity has "returned"

startActivityForResult() Example

```
public static final int CHOOSE_FOOD_REQUEST = 1;


Intent intent = new Intent(this, ChooseFoodItemsActivity.class);
startActivityForResult(intent, CHOOSE_FOOD_REQUEST);
```

Implement onActivityResult()

```
public void onActivityResult(int requestCode,
                             int resultCode, Intent data) {
  super.onActivityResult(requestCode, resultCode, data);
  if (requestCode == CHOOSE_FOOD_REQUEST) { // Identify activity
    if (resultCode == RESULT_OK) { // Activity succeeded
      // … do something with the data
 }}}
```

Returning data

In second activity

1. To return data from the second Activity:
   - Create a ***new*** Intent
   - Put the response data in the Intent using `putExtra()`
   - Set the result to `Activity.RESULT_OK` or `RESULT_CANCELED`, if the user cancelled out

2. Call `finish()` to close second activity ➔ invoke onActivityResult() of the first activity.

Return data and finish second activity

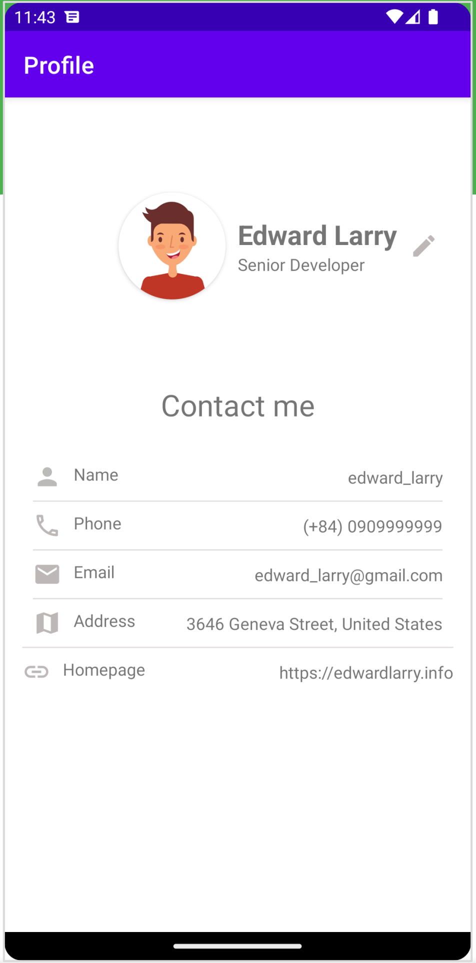
```
 // Create an intent
Intent replyIntent = new Intent();

// Put the data to return into the extra
replyIntent.putExtra(EXTRA_REPLY, reply);

// Set the activity's result to RESULT_OK
setResult(RESULT_OK, replyIntent);

// Finish the current activity
finish();
```

Example 01

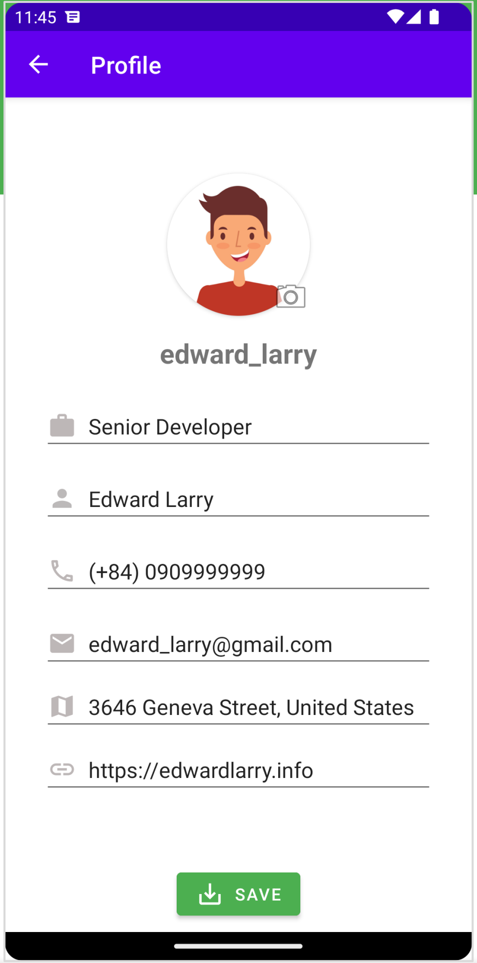

11:43

Profile

Edward Larry
Senior Developer

Contact me

Name	edward_larry
Phone	(+84) 0909999999
Email	edward_larry@gmail.com
Address	3646 Geneva Street, United States
Homepage	https://edwardlarry.info


11:45

Profile

edward_larry

Senior Developer

Edward Larry

(+84) 0909999999

edward_larry@gmail.com

3646 Geneva Street, United States

https://edwardlarry.info

Implicit Intents

- ❑ Start an Activity with an implicit intent.
- ❑ Sending data with URI or Extras
- ❑ Common actions
- ❑ Receiving Implicit Intents

Start an Activity with implicit intent

To ask Android to find an Activity to handle your request, use an implicit intent

1. Create an intent
   - `Intent intent = new Intent(action, uri);`
2. Use the intent to start the activity
   - `startActivity(intent);`

Implicit intents

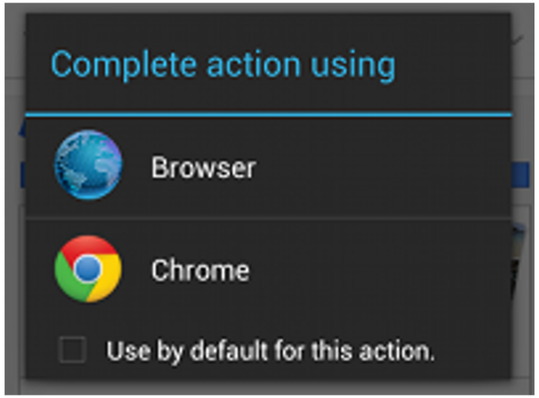
- An implicit intent allows you to start an activity in another app by describing an action you intend to perform, such as "share an article", "view a map", or "take a picture"
- An implicit intent specifies an action and may provide data with which to perform the action

Implicit intents

- Implicit intents do not specify the target activity class, just the intended action
- Android runtime matches the implicit intent request with registered intent handlers
- If there are multiple matches, an App Chooser will open to let the user decide

App Chooser

When the Android runtime finds multiple registered activities that can handle an implicit intent, it displays an App Chooser to allow the user to select the handler



How do implicit intents work?

1. The Android Runtime keeps a list of registered Apps
2. Apps have to register via the Android Manifest
3. Runtime receives the request and looks for matches
4. Android runtime uses intent filters for matching
5. If more than one match, shows a list of possible matches and let the user choose one
6. Android runtime starts the requested activity

Implicit Intents - Examples

Show a web page

```
Uri uri = Uri.parse("http://www.google.com");
Intent it = new Intent(Intent.ACTION_VIEW,uri);
startActivity(it);
```

Dial a phone number

```
Uri uri = Uri.parse("tel:8005551234");
Intent it = new Intent(Intent.ACTION_DIAL, uri);
startActivity(it);
```

Avoid exceptions and crashes

Before starting an implicit activity, use the package manager to check that there is a package with an activity that matches the given criteria.

```
Intent myIntent = new Intent(Intent.ACTION_CALL_BUTTON);

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

Sending an implicit intent with data URI

1. Create an intent for action

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

1. Provide data as a URI

```
intent.setData(Uri.parse("tel:8005551234"));
```

1. Start the activity

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

Providing the data as URI

Create an URI from a string using `Uri.parse(String uri)`

- `Uri.parse("tel:8005551234")`
- `Uri.parse("geo:0,0?q=brooklyn%20bridge%2C%20brooklyn%2C%20ny")`
- `Uri.parse("http://www.android.com");`

Uri documentation

Sending an implicit intent with extras

1. Create an intent for an action

```
Intent intent = new Intent(Intent.ACTION_WEB_SEARCH);
```

1. Put extras

```
String query = edittext.getText().toString();
intent.putExtra(SearchManager.QUERY, query));
```

1. Start the activity

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

Category

Additional information about the kind of component to handle the intent.

- `CATEGORY_OPENABLE`
  Only allow URIs of files that are openable
- `CATEGORY_BROWSABLE`
  Only activities that can start a web browser to display data referenced by the URI

Sending an implicit intent with type and category

1. Create an intent for an action

```
Intent intent = new Intent(Intent.ACTION_CREATE_DOCUMENT);
```

2. Set mime type and category for additional information

```
intent.setType("application/pdf"); // set MIME type
intent.addCategory(Intent.CATEGORY_OPENABLE);
```

continued on next slide...

Sending an implicit intent with type and category

3. Start the activity

```
if (intent.resolveActivity(getPackageManager()) != null) {
  startActivityForResult(myIntent,ACTIVITY_REQUEST_CREATE_FILE);
}
```

4. Process returned content URI in `onActivityResult()`

Common actions for implicit intents

Common actions for implicit intents include:

- ACTION_SET_ALARM
- ACTION_IMAGE_CAPTURE
- ACTION_CREATE_DOCUMENT
- ACTION_SENDTO
- and many more

Apps that handle common actions

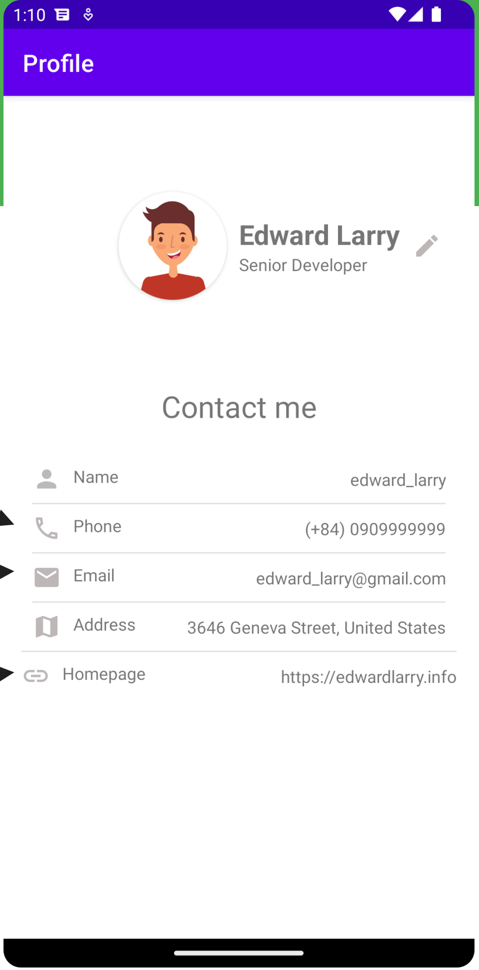
Common actions are usually handled by installed apps, both system apps and other apps, such as:

- Alarm Clock, Calendar, Camera, Contacts
- Email, File Storage, Maps, Music/Video
- Notes, Phone, Search, Settings
- Text Messaging and Web Browsing

→ List of **common** actions for implicit intents

→ List of all available actions

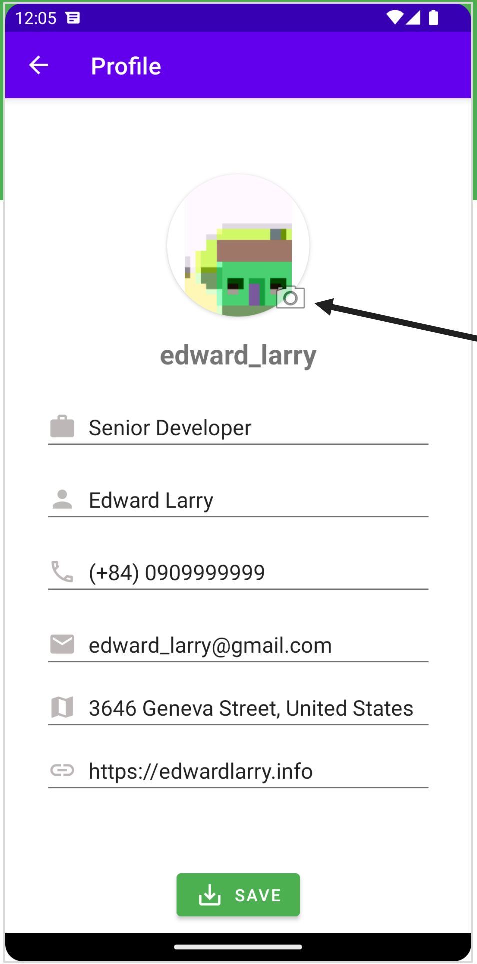
Example 02


Profile

Edward Larry
Senior Developer

Contact me

Name	edward_larry
Phone	(+84) 0909999999
Email	edward_larry@gmail.com
Address	3646 Geneva Street, United States
Homepage	https://edwardlarry.info


Profile

edward_larry

Senior Developer

Edward Larry

(+84) 0909999999

edward_larry@gmail.com

3646 Geneva Street, United States

https://edwardlarry.info

SAVE

Make a phone call

Compose email

Open web browser

Capture a picture

Receiving Implicit Intents

Register your app to receive intents

- Declare one or more intent filters for the activity in the Android manifest
- Filter announces activity's ability to accept implicit intents
- Filter puts conditions on the intents that the activity accepts

Intent filter in the Android Manifest

```
<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>
```

Intent filters: action and category

- action — Match one or more action constants
  - android.intent.action.VIEW — matches all intents with ACTION_VIEW
  - android.intent.action.SEND — matches all intents with ACTION_SEND

- category — additional information (list of categories)
  - android.intent.category.BROWSABLE—can be started by web browser
  - android.intent.category.LAUNCHER—Show activity as launcher icon

Intent filters: data

- `data` — Filter on data URIs, MIME type
  - `android:scheme="https"`—require URIs to be https protocol
  - `android:host="developer.android.com"`—only accept intents from specified hosts
  - `android:mimeType="text/plain"`—limit the acceptable types of documents

An activity can have multiple filters

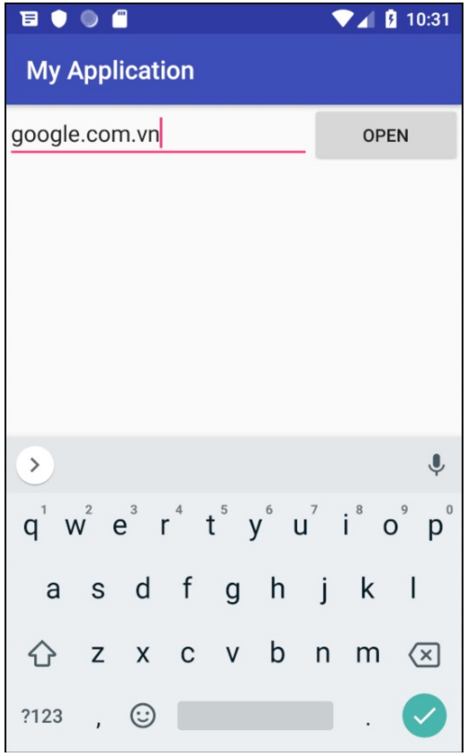
```
<activity android:name="ShareActivity">
  <intent-filter>
    <action android:name="android.intent.action.SEND"/>
    ...
  </intent-filter>
  <intent-filter>
    <action android:name="android.intent.action.SEND_MULTIPLE"/>
    ...
  </intent-filter>
</activity>
```

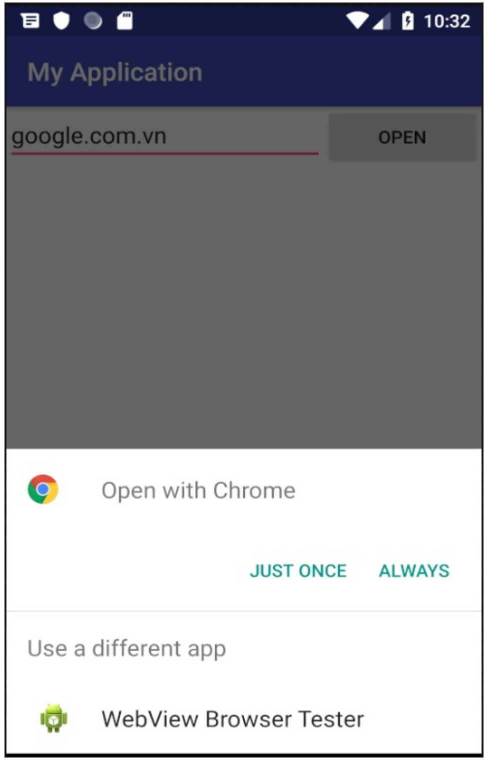
An Activity can have several filters

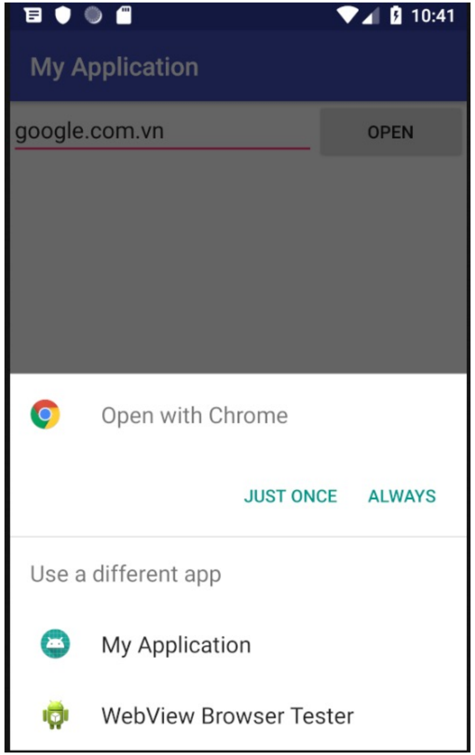
A filter can have multiple actions & data

```
<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="image/*"/>
  <data android:mimeType="video/*"/>
</intent-filter>
```

Example 03


10:31
My Application
google.com.vn
OPEN


10:32
My Application
google.com.vn
OPEN
Open with Chrome
JUST ONCE
ALWAYS
Use a different app
WebView Browser Tester


10:41
My Application
google.com.vn
OPEN
Open with Chrome
JUST ONCE
ALWAYS
Use a different app
My Application
WebView Browser Tester

Learn more

Learn more

- Android Application Fundamentals
- Starting Another Activity
- Activity (API Guide)
- Activity (API Reference)
- Intents and Intent Filters (API Guide)
- Intent (API Reference)
- Navigation

What's Next?

- Concept Chapter: 2.1 C Understanding Activities and Intents
- Practical: 2.1 P Create and Start Activities

END