ANDROID DEVELOPMENT

ACTIVITIES & INTENTS

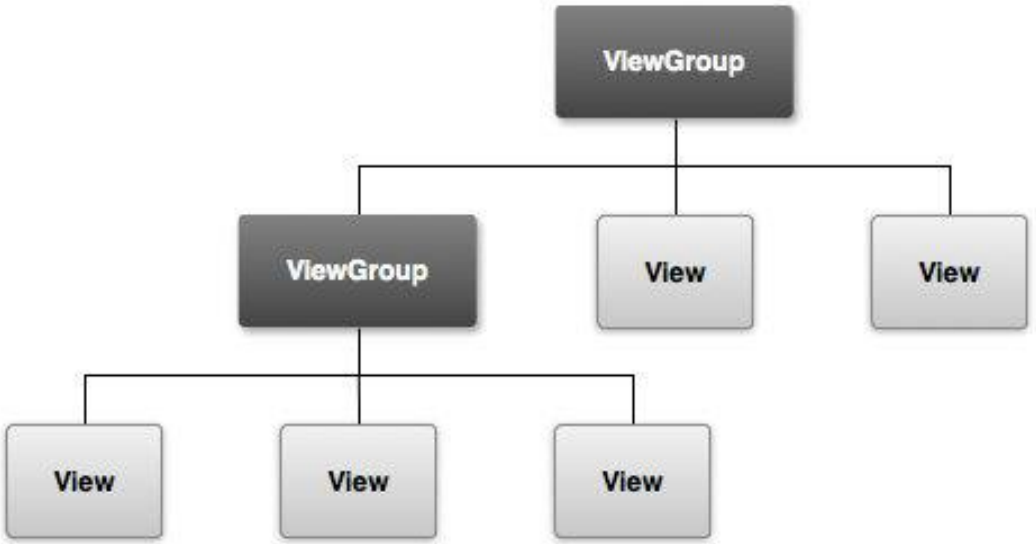
Tan Cher Wah (isstcw@nus.edu.sg)

Agenda

- Activity
- Intent
  - Explicit
  - Implicit
- Pass data to an activity
- Get results from an activity
- Intent Filters

Interaction in Activity

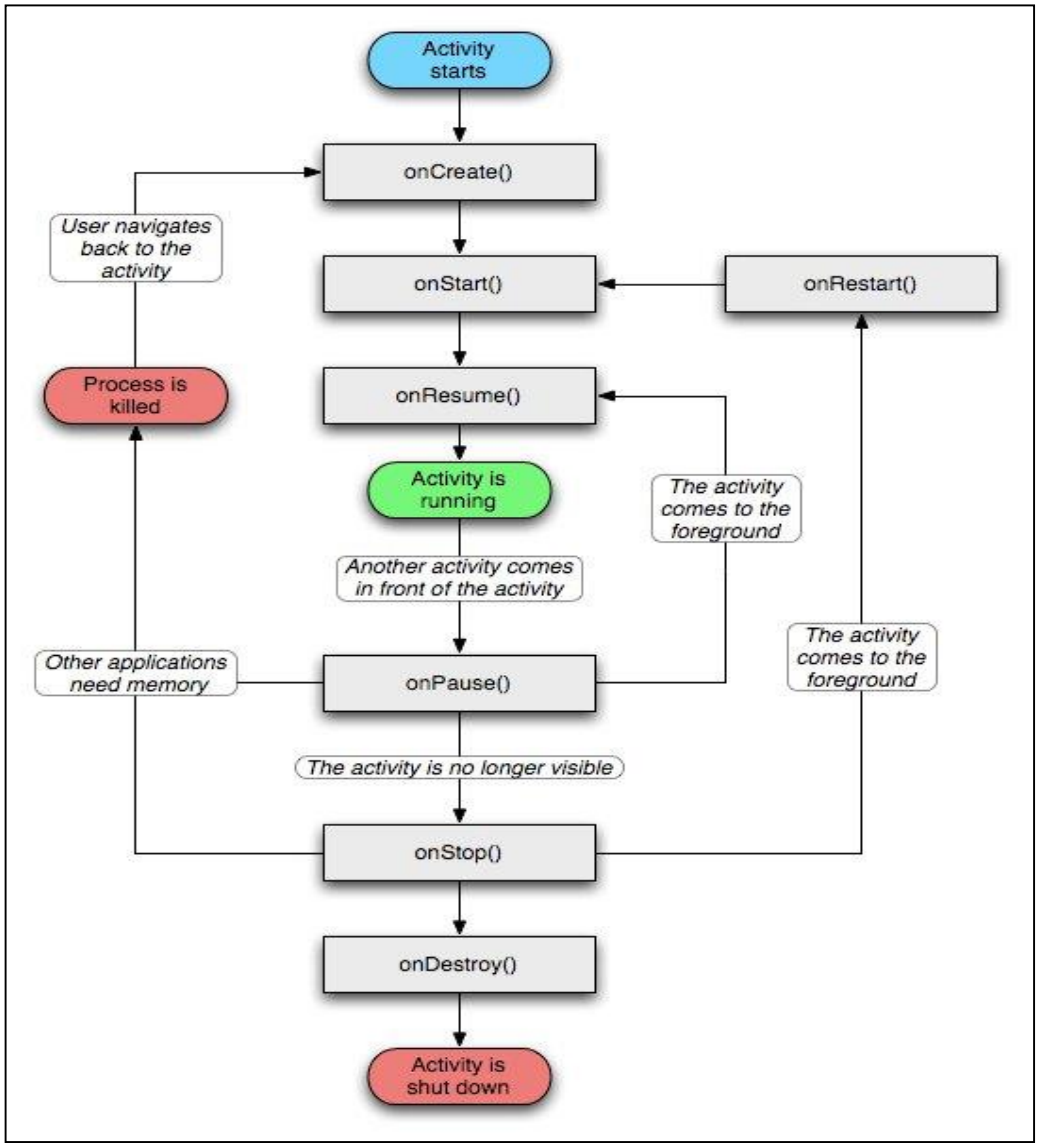
- An Android app allows user interaction via a set of activity screens
- Each Activity screen incorporates UI views for interaction (and associated event handlers)
- Views specified via static **XML** or **dynamically inserted via code**
- Positioned by Layout Managers



Activity

- An Activity may start another Activity via an **Intent**
- When the former Activity stops; it is preserved in Android's "back stack"
- An Activity can be notified via **callback methods** within its life-cycle
- An Activity is paused/stopped when
  - Home screen accessed via *home* button
  - Task selection with *task* button
  - Previous activity resumed via *back* button

Activity Lifecycle



Activity Events

- Override and handle the various lifecycle events

```
@Override
protected void onStart() {
    super.onStart();
    // The activity is about to become visible.
}
@Override
protected void onResume() {
    super.onResume();
    // The activity has become visible (it is now "resumed").
}
@Override
protected void onPause() {
    super.onPause();
    // Another activity is taking focus (about to be "paused").
}
@Override
protected void onStop() {
    super.onStop();
    // The activity is no longer visible (it is now "stopped")
}
```

Intents

- Use an **Intent** to request an **action** from another Android component
  - **Explicit Intent** – specific component (i.e. class name) is specified
  - **Implicit Intent** – component is inferred from action
- Encourages the **reuse** of application components

Explicit Intent

- Explicit **Intent** allows an activity to start another **Activity** implemented by Java class, such as **UserSettingActivity**

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

Implicit Intent

- Implicit **Intent** allows an activity to start another via action and data
  - **Intent.ACTION_VIEW** with **http** URI invokes the browser
  - Constructor of class **Intent** is overloaded

```
Uri uri = Uri.parse("https://www.iss.nus.edu.sg");
Intent i = new Intent(Intent.ACTION_VIEW, uri);
startActivity(i);
```

Intents Delivery

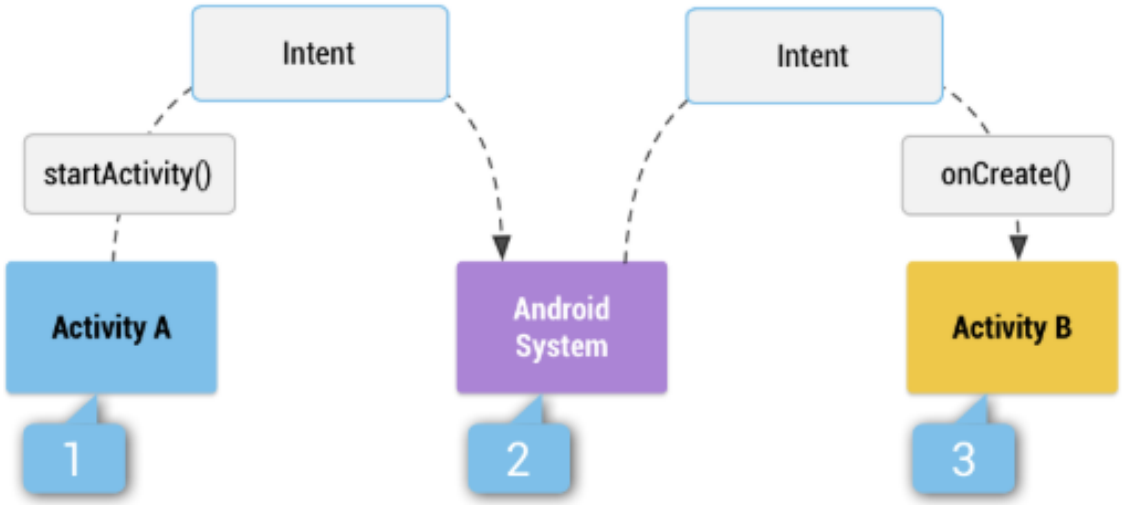


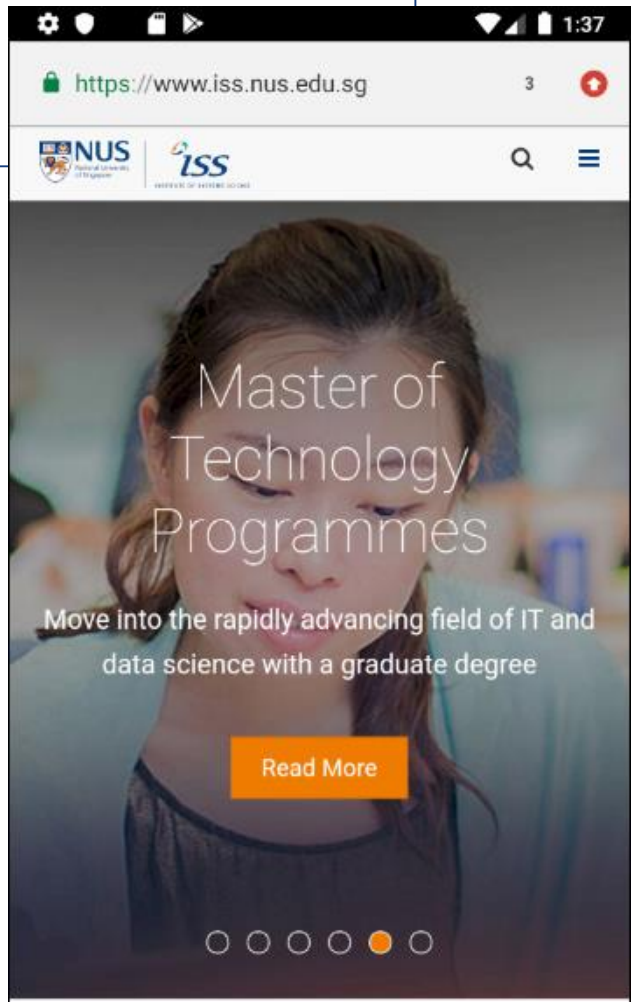
Figure 1. How an implicit intent is delivered through the system to start another activity: **[1]** *Activity A* creates an `Intent` with an action description and passes it to `startActivity()`. **[2]** The Android System searches all apps for an intent filter that matches the intent. When a match is found, **[3]** the system starts the matching activity (*Activity B*) by invoking its `onCreate()` method and passing it the `Intent`.

Sample actions / URI

Intent.ACTION_VIEW “http://www.google.com/"	Browse
Intent.ACTION_CALL, "tel:(+65)12345678"	Call
Intent.ACTION_DIAL, "tel:(+65)12345678"	Dialer
Intent.ACTION_VIEW, "geo:1.292248,103.7722283"	Map
"android.media.action.IMAGE_CAPTURE"	Camera
Intent.ACTION_VIEW, "content://contacts/people/"	Contacts
Intent.ACTION_SENDTO, “smsto:12345678”	SMS
Intent.ACTION_SENDTO, “mailto:someone@somewhere.com”	Email

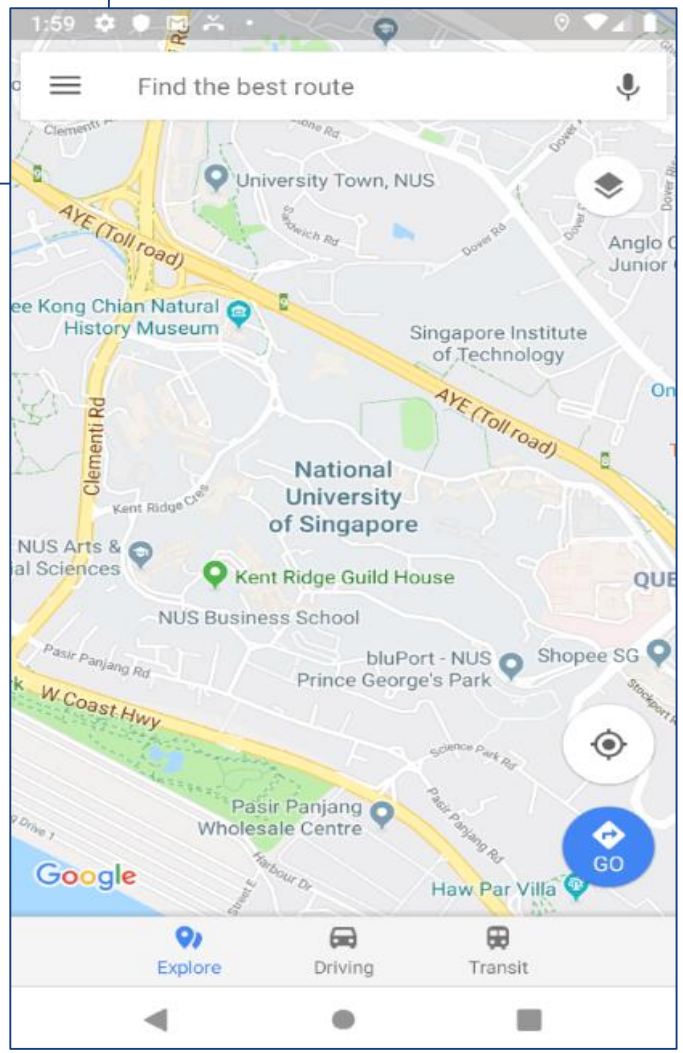
Browse

```
Uri uri = Uri.parse("https://www.iss.nus.edu.sg");

Intent intent = new Intent(Intent.ACTION_VIEW, uri);

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



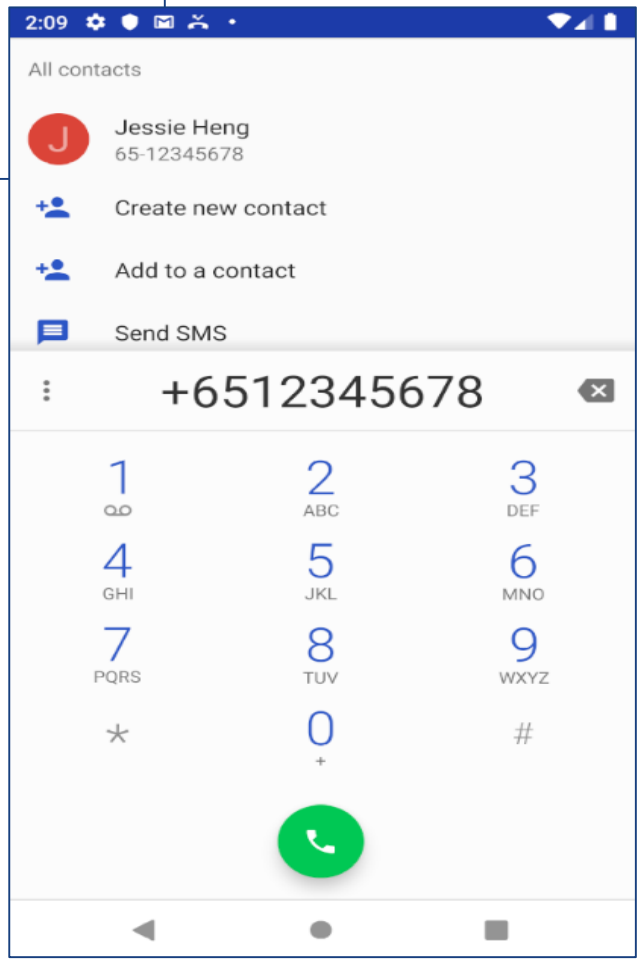
Geo-Location

```
Uri uri = Uri.parse("geo:1.296643,103.776398");

Intent intent = new Intent(Intent.ACTION_VIEW, uri);

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



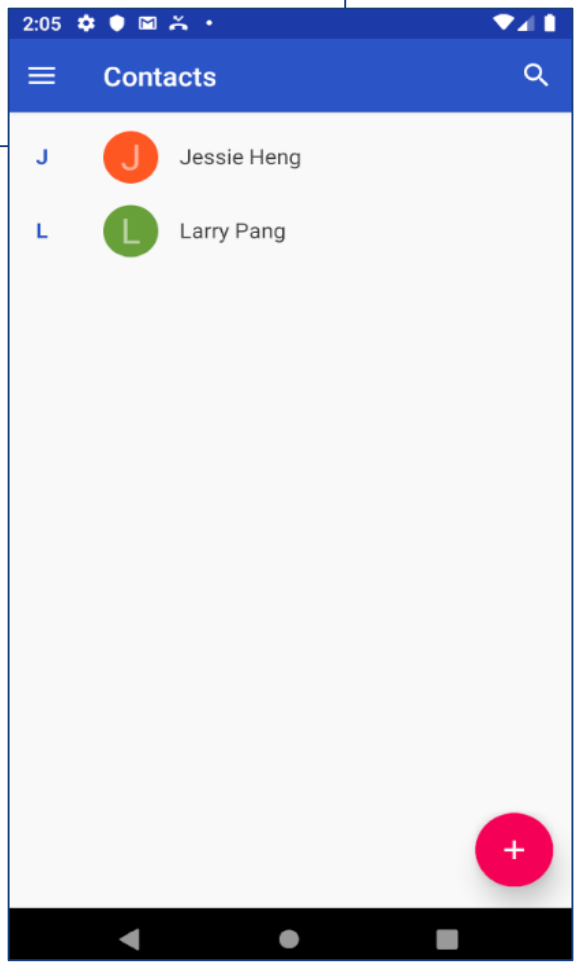
Dial

```
Uri uri = Uri.parse("tel:(+65)12345678");

Intent intent = new Intent(Intent.ACTION_DIAL, uri);

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



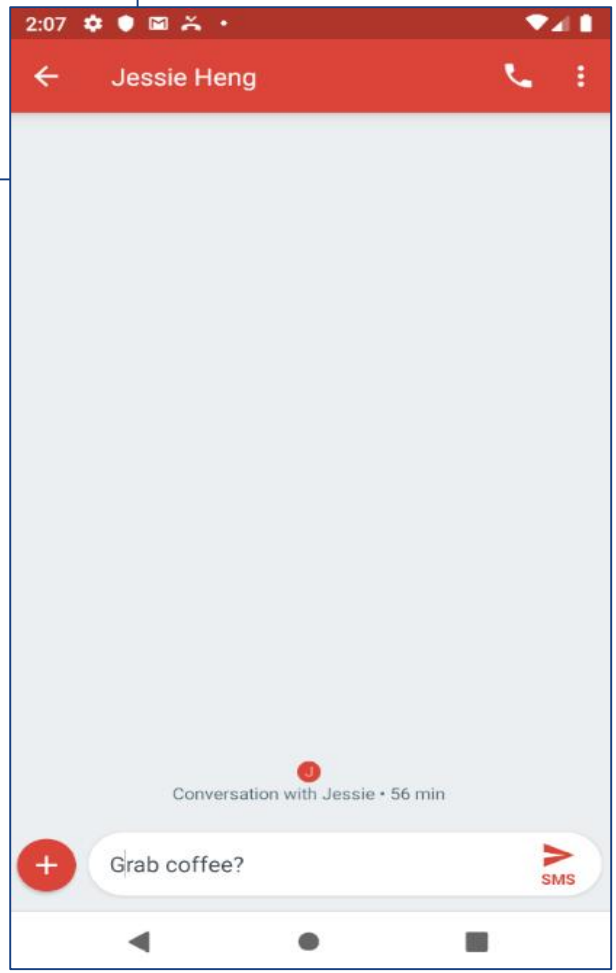
Contacts

```
Uri uri = Uri.parse("content://contacts/people");

Intent intent = new Intent(Intent.ACTION_VIEW, uri);

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



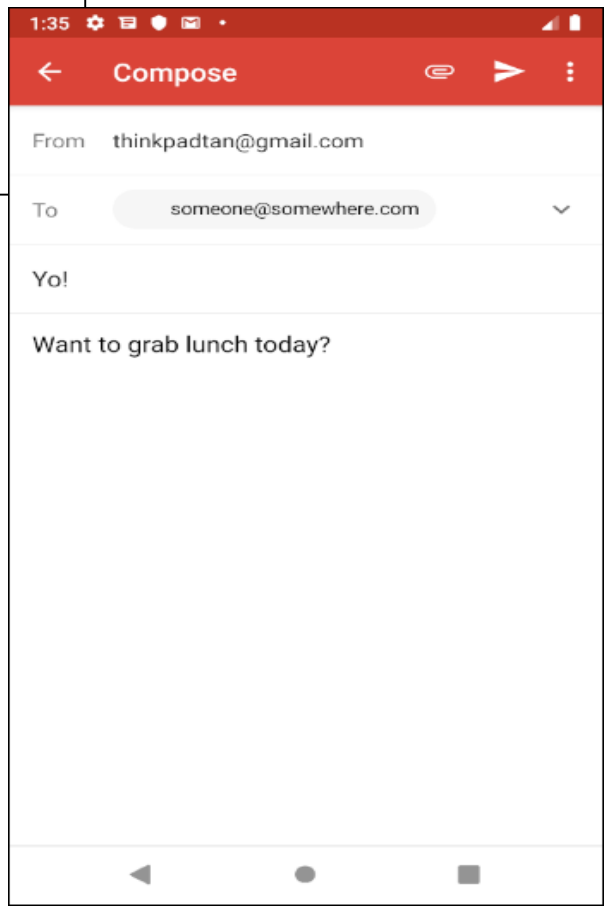
SMS

```
Uri uri = Uri.parse("smsto:12345678");

Intent intent = new Intent(Intent.ACTION_SENDTO, uri);
intent.putExtra("sms_body", "Grab coffee?");

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



Email

```
Uri uri = Uri.parse("mailto:someone@somewhere.com");

Intent intent = new Intent(Intent.ACTION_SENDTO, uri);
intent.putExtra(Intent.EXTRA_SUBJECT, "Yo!");
intent.putExtra(Intent.EXTRA_TEXT, "Want to grab lunch today?");

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



Intent with data

- **Intent** may pass data through key/value associations
- Anticipate results via unique **REQUEST_CODE**

```
int REQUEST_CODE = 1;

Intent i = new Intent(this, UserSettingActivity.class);
i.putExtra("key1", "value1");
i.putExtra("key2", "value2");

startActivityForResult(i, REQUEST_CODE);
```

Invoked activity receiving data

- Invoked activity may receive data via **getExtras()** and **getString()** in Activity initialisation

```
Bundle extras = getIntent().getExtras();
if (extras == null)
    return;

String value1 = extras.getString("key1");
String value2 = extras.getString("key2");
```

Returning data

- Ends an activity by calling the **finish()** method
- Set results to be returned

```
@Override
public void finish()
{
    Intent data = new Intent();

    data.putExtra("Key1", "Value1");
    data.putExtra("Key2", "Value2");

    setResult(RESULT_OK, data);

    super.finish();
}
```

Extracting returned data

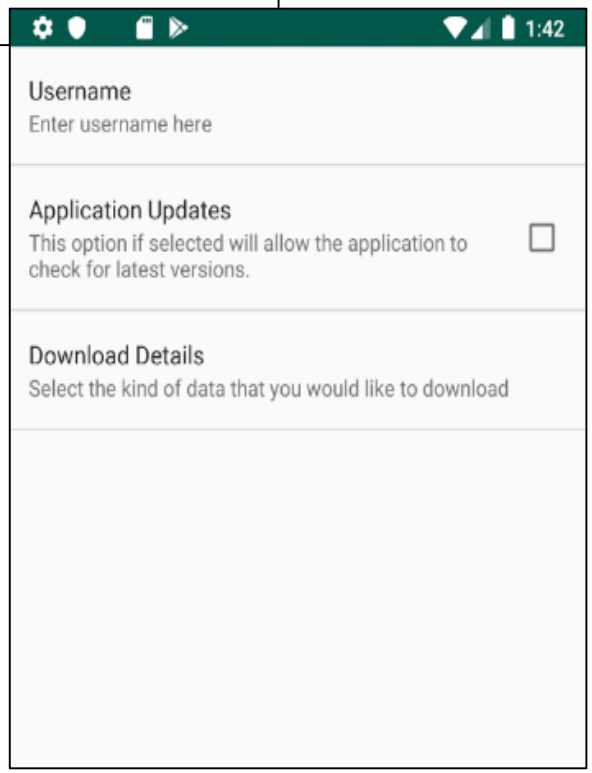
- Calling activity is notified of completion via **onActivityResult()** method

```
protected void onActivityResult(int requestCode, int resultCode, Intent data)
{
   String res = null;
   int REQUEST_CODE = 1;

   if (resultCode==RESULT_OK && requestCode == REQUEST_CODE)
   {
      if (data.hasExtra("Key1"))
      {
         Bundle bundle = data.getExtras();
         if (bundle != null)
            res = bundle.getString("Key1");
      }
   }
}
```

Setting explicit Intent

- Activity invoked explicitly by providing the class of the target activity

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

startActivity(intent);
```



Setting explicit Intent

- Activity invoked explicitly with data

```
Intent intent = new Intent(this, MyBrowser.class);
intent.setData(Uri.parse("http://www.google.com/"));

startActivity(intent);
```

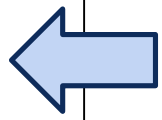
OR

```
Intent intent = new Intent(this, MyBrowser.class);
intent.putExtra("home", "http://www.google.com/");

startActivity(intent);
```

New Activity

- Create new class inheriting from **android.app.AppCompatActivity**
- Setup a corresponding layout specification for activity
- Include activity in **AndroidManifest.xml**

```
<application
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:theme="@style/AppTheme">
    <activity android:name=".P1Activity" />
    <activity android:name=".UserSettingActivity" />
    <activity
        android:name=".MainActivity"
        android:label="@string/app_name"
```



- Create **Intent** and start activity

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

Intent Filters

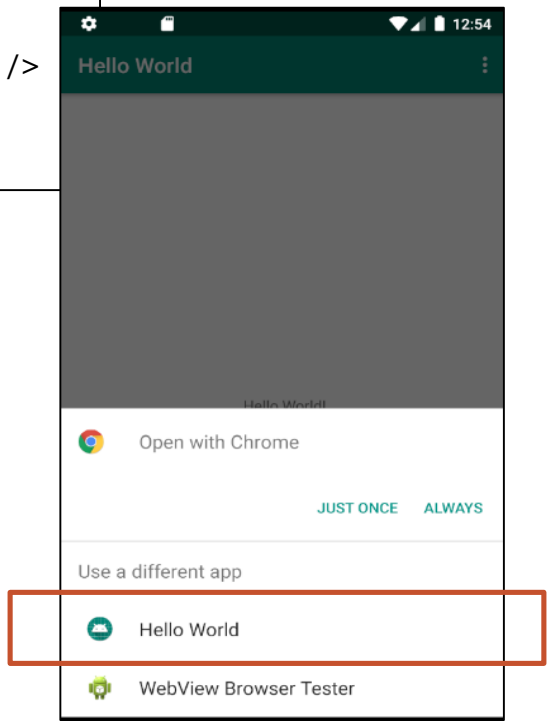
- To specify the type of Intents that an Android component wishes to **receive**
- Specify via
  - Manifest file (AndroidManifest.xml)
  - Programmatically (via new IntentFilter())

```
public void startReceiver() {
    bcReceiver = new BroadcastReceiver() {
        @Override
        public void onReceive(Context context, Intent intent) {
            stopReceiver();
        }
    };

    IntentFilter filter = new IntentFilter();
    filter.addAction(MainActivity.STOP_INTENTSERVICE);
    registerReceiver(bcReceiver, filter);
}

public void stopReceiver() {
    if (bcReceiver != null) {
        unregisterReceiver(bcReceiver);
        bcReceiver = null;
    }
    carryOn = false;
}
```

Intent Filters

- Our Activity becomes a user-selectable web browsing option by registering an **implicit intent** for accessing web pages (via HTTP protocol)

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



References

- Intents & Intent Filters - https://developer.android.com/guide/components/intents-filters

Active Recall