

# Recording in Progress

This class is being recorded

Please turn off your video and/or video if you do not wish to be recorded

# **CMSC436: Programming Handheld Systems**

# The Intent Class

# Today's Topics

The Intent Class

Starting Activities with Intents

- Explicit Activation

- Implicit Activation via Intent resolution

# The Intent Class

A data structure that represents

An operation to be performed, or

An event that has occurred

# Today's Focus

Using Intents for operations to be performed

i.e., using Intents to start a single Activity

We'll cover using Intents for event notification  
when we talk about BroadcastReceivers

# Intents Identify a Desired Operation

Intents provide a flexible “language” for specifying operations to be performed

e.g., I want to pick a contact, take a photo, dial a phone number, etc.

# Intents Identify a Desired Operation

An Intent is constructed by one component that wants some work done

It is delivered to another component that offers to perform that work



# Intent Fields

Action

Component

Data

Extras

Category

Flags

Type

# Action

String representing the desired operation

# Platform-Defined Examples

`ACTION_DIAL` – Dial a number

`ACTION_EDIT` – Display data to edit

`ACTION_SYNC` – Synchronize device data with a server

`ACTION_MAIN` – Start as initial activity of app

## Setting the Intent Action

```
val newIntent = Intent(Intent.ACTION_DIAL)
```

Or

```
val newIntent = Intent()  
newIntent.action = Intent.ACTION_DIAL
```

# Data

Data associated with the Intent

Formatted as a Uniform Resource Identifier (URI)

# Examples

Data to view on a map

```
Uri.parse("geo:0,0?q=1600+Pennsylvania  
+Ave+Washington+DC")
```

Number to dial in the phone dialer

```
Uri.parse("tel:+15555555555")
```

## Setting Intent Data

```
val intent= Intent (Intent.ACTION_DIAL,  
                    Uri.parse("tel:+15555555555"))
```

Or

```
val intent = Intent(Intent.ACTION_DIAL)  
intent.data = Uri.parse("tel:+15555555555")
```

# Category

Additional information about the components that are allowed to handle the Intent



# Examples

CATEGORY\_BROWSABLE – Activity can be invoked to display data referenced by a URI

CATEGORY\_LAUNCHER – can be the initial Activity of a task and is listed in top-level app launcher

# Type

Specifies an explicit MIME type of the Intent data

## Examples

image/\*, image/png, image/jpeg

text/html, text/plain

If unspecified, Android will infer the type

# Component

The component that should receive this Intent

Use this when there's exactly one named component that should receive the intent

# Setting the component

```
val intent = Intent(packageContext: Context!,  
                    cls: Class<*>!)
```

# Setting the component

Or

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

and one of:

```
setComponent(), setClass(), or setClassName()
```

# Extra

Additional information associated with Intent

Treated as a map (key-value pairs)

## Intent.EXTRA\_EMAIL: Email Recipient List

```
val intent = Intent(Intent.ACTION_SEND)
intent.putExtra(Intent.EXTRA_EMAIL,
    arrayOf("aporter@cs.umd.edu",
            "ceo@microsoft.com",
            "potus@whitehouse.gov",
            "mozart@musician.org"))
```

# Setting the Extra Attribute

Several forms depending on data type

```
putExtra(name: String!, value: String?);
```

```
putExtra(name: String!, value: FloatArray?);
```

...



# Flags

Specify additional information on how Intent should be handled

# Examples

## FLAG\_ACTIVITY\_NO\_HISTORY

Don't put this Activity in the Task backstack

## FLAG\_DEBUG\_LOG\_RESOLUTION

Print extra logging information when this Intent is processed

# Setting Flags

```
val intent = Intent(Intent.ACTION_SEND)
intent.flags =
    Intent.FLAG_ACTIVITY_NO_HISTORY
```

# Starting Activities with Intents

```
fun startActivity(intent: Intent!): Unit
```

```
fun startActivityForResult(intent: Intent!,  
                           requestCode: Int): Unit
```

# The Target Activity

Can be named ***explicitly*** by setting the Intent's component

Otherwise, it is determined ***implicitly***

# Explicit Activation

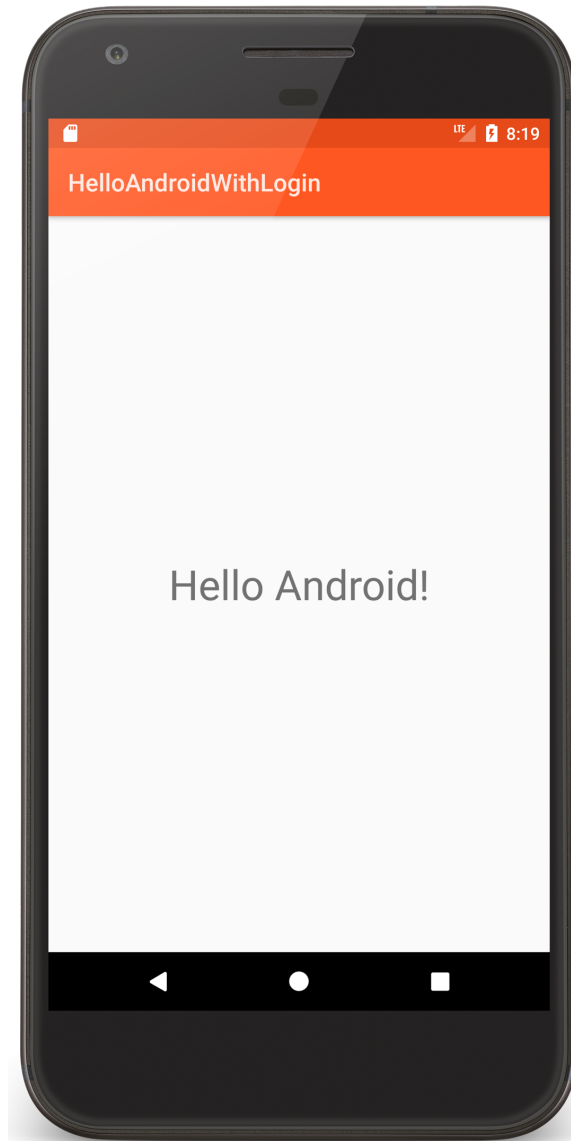
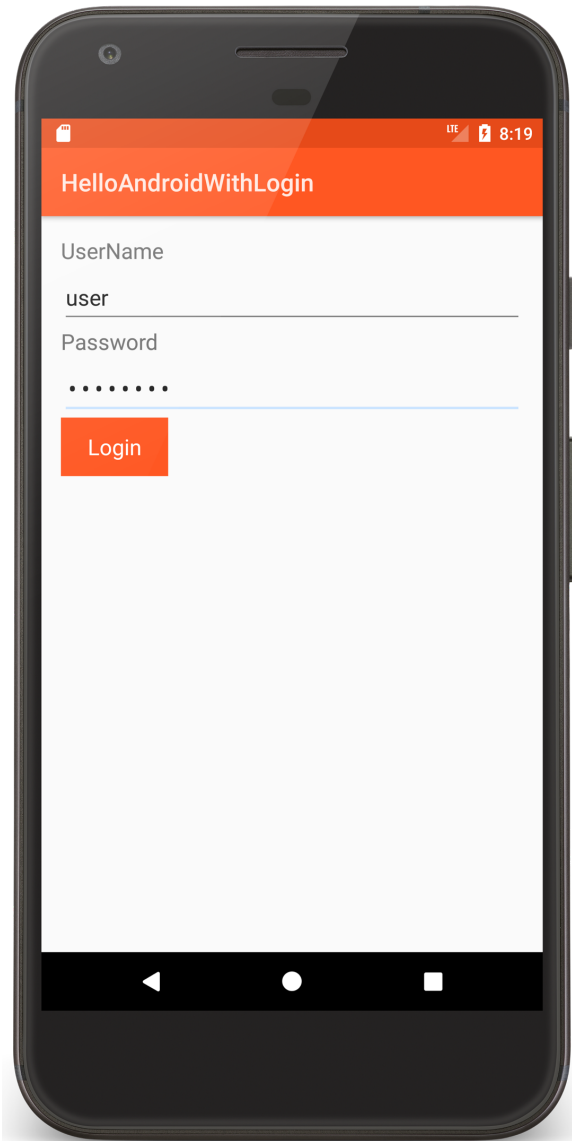
HelloWorldWithLogin

Two Activities

LoginActivity checks username & password and then starts HelloAndroidActivity

HelloAndroidActivity shows “Hello Android!” message

# HelloAndroid WithLogin



# LoginScreen.kt

```
fun onClick(v: View?) {
    if (/* authorized */) {
        // Create an explicit Intent for starting the
        // HelloAndroid Activity
        val helloAndroidIntent = Intent(
            this@loginScreen,
            HelloAndroid::class.java)

        // Use the Intent to start the HelloAndroid Activity
        startActivity(helloAndroidIntent)
    }
    ...
}
```



# Implicit Activation

When the Activity to be started is not explicitly named, Android tries to find Activities that match information contained in the Intent

This process is called Intent Resolution

# Intent Resolution Process

Intents describe desired operations

IntentFilters describe which operations a given Activity can handle

IntentFilters specified in AndroidManifest.xml or programmatically

# Intent Resolution Data

Action

Type (includes data)

Categories

# Specifying IntentFilters

```
<activity ...>
```

```
...
```

```
<intent-filter ...>
```

```
...
```

```
<action android:name="actionName" />
```

```
...
```

```
</intent-filter>
```

```
...
```

```
</activity>
```

# Handling Intent.ACTION\_DIAL

```
<activity ...>
```

```
...
```

```
<intent-filter ...>
```

```
...
```

```
<action android:name="android.intent.action.DIAL" />
```

```
...
```

```
</intent-filter>
```

```
...
```

```
</activity>
```

# Adding Data to IntentFilter

```
<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>
```

# Handling geo: Scheme Intents

```
<intent-filter ...>
```

```
...
```

```
<data android:scheme="geo" />
```

```
...
```

```
</intent-filter>
```

# Adding a Category to an IntentFilter

```
<intent-filter ...>
```

```
...
```

```
  <category android:name="string" />
```

```
...
```

```
</intent-filter>
```



# Example: Maps Application

```
<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>
```

# Receiving Implicit Intents

Note: to receive implicit intents an Activity should specify an IntentFilter with the category

`"android.intent.category.DEFAULT"`

# Priority

android:priority – Priority given to the parent component when handling matching Intents

Causes Android to prefer one activity over another

$-1000 \leq \text{priority} \leq 1000$

Higher values represent higher priorities

# Using Implicit Intents

The MapLocation app created an implicit Intent and then used it in a call to `startActivity()`

Should start a Maps app

What if the user has uninstalled all Maps apps?

Your code should always check before attempting to start an Activity with an implicit Intent

# MapLocation.kt

```
private fun processClick() {
    try {
        ...
        // Create Intent object for starting Google Maps application
        val geoIntent = Intent(
            Intent.ACTION_VIEW, Uri
                .parse("geo:0,0?q=$address"))

        geoIntent.resolveActivity(packageManager)?.let {
            startActivity(geoIntent)
        }
    }
    ...
}
```

# Using Implicit Intents

Implicit Intents can pose a security hazard

Prefer explicit Intents within your own app

Set the `android:exported` attribute to `false` in `AndroidManifest.xml`, if you don't want other apps to start a given component in your app

# Investigate Intent Filters

```
% adb shell dumpsys package
```

```
1761a23 com.google.android.gm/.Gmail2PreferenceActivity
comgooglewallet:
  551fb20 com.google.android.gms/.tapandpay.tokenization.AddNewCardThroughBrowserActivity
:
  4b70c8a com.google.android.apps.photos/.pager.HostPhotoPagerActivity
  b0349a9 com.google.android.calendar/.ICalLauncher (4 filters)
geo:
  b1dd765 com.google.android.apps.maps/com.google.android.maps.MapsActivity
mms:
  92bdcd9 com.google.android.talk/com.google.android.apps.hangouts.phone.BabelHomeActivity
  d06357f com.example.android.apis/.os.MmsMessagingDemo
  dcd569e com.google.android.apps.messaging/.ui.conversation.LaunchConversationActivity
sip:
  12d683 com.android.phone/.PrivilegedOutgoingCallBroadcaster
  1b37000 com.android.server.telecom/.components.UserCallActivity
  586e039 com.android.server.telecom/.PrivilegedCallActivity
  647ad3d com.android.phone/.OutgoingCallBroadcaster
  7d5067e com.android.server.telecom/.EmergencyCallActivity
  d7b8932 com.android.phone/.EmergencyOutgoingCallBroadcaster
sms:
  73ac3a com.android.fallback/.Fallback
  92bdcd9 com.google.android.talk/com.google.android.apps.hangouts.phone.BabelHomeActivity
  dcd569e com.google.android.apps.messaging/.ui.conversation.LaunchConversationActivity
  f2ba94c com.example.android.apis/.os.SmsMessagingDemo
tel:
  12d683 com.android.phone/.PrivilegedOutgoingCallBroadcaster
  1b37000 com.android.server.telecom/.components.UserCallActivity
```

```
-uu-:---F1 dumpsys.out.txt 4% L592 (Text Isearch)-----
```

```
I-search: geo
```



**Next**

Permissions

# Example Applications

HelloAndroidWithLogin