ITW202: Mobile Application

Unit IV: Developing for Android

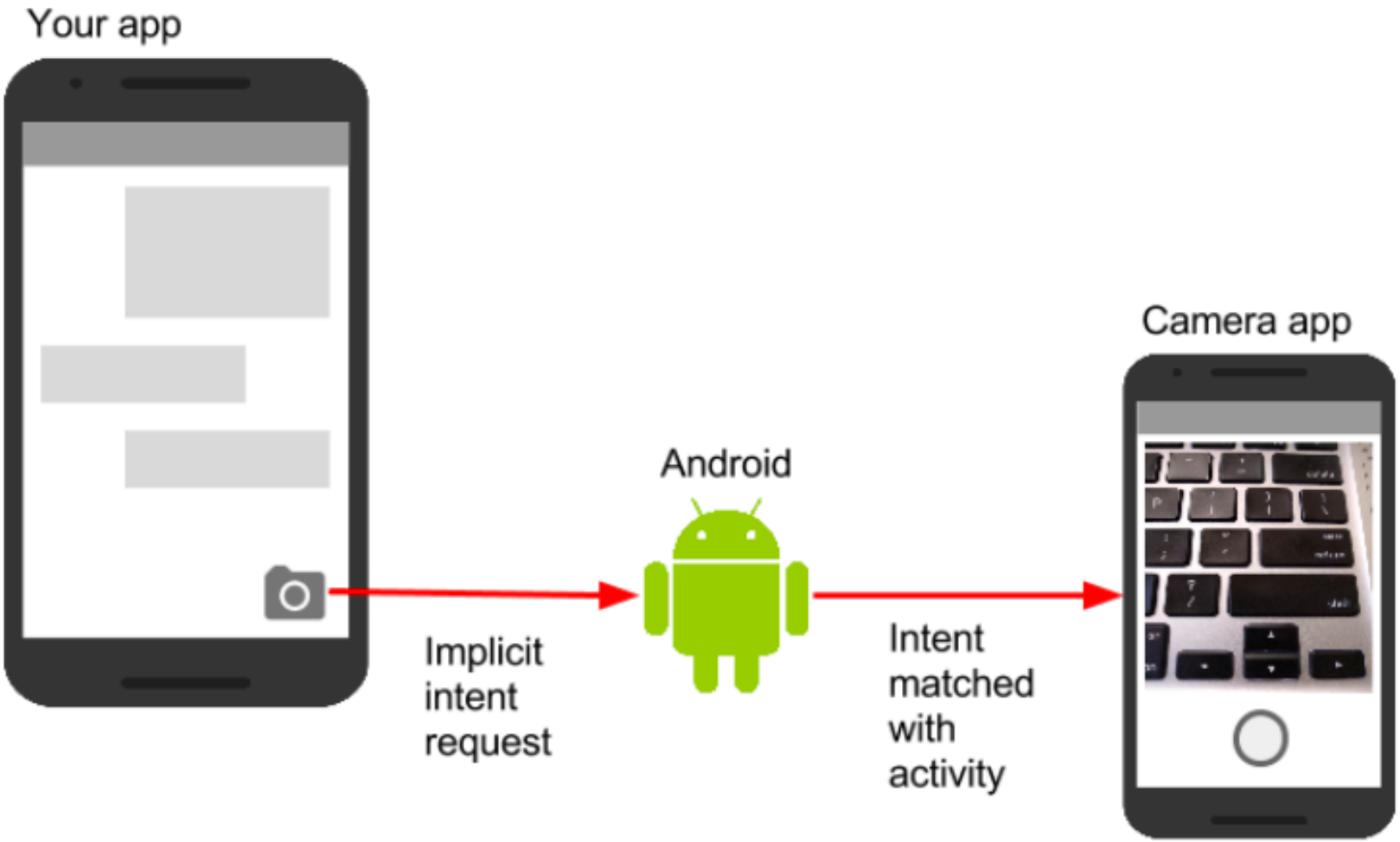
Ms. Sonam Wangmo

Gyalpozhing College of Information Technology
Royal University of Bhutan

April 20, 2021

Implicit Intents

Implicit Intents



What you do with an implicit Intent

- 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"
- Specify an action and optionally provide data with which to perform the action
- Don't specify the target Activity class, just the intended action

What system does with implicit Intent

- 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

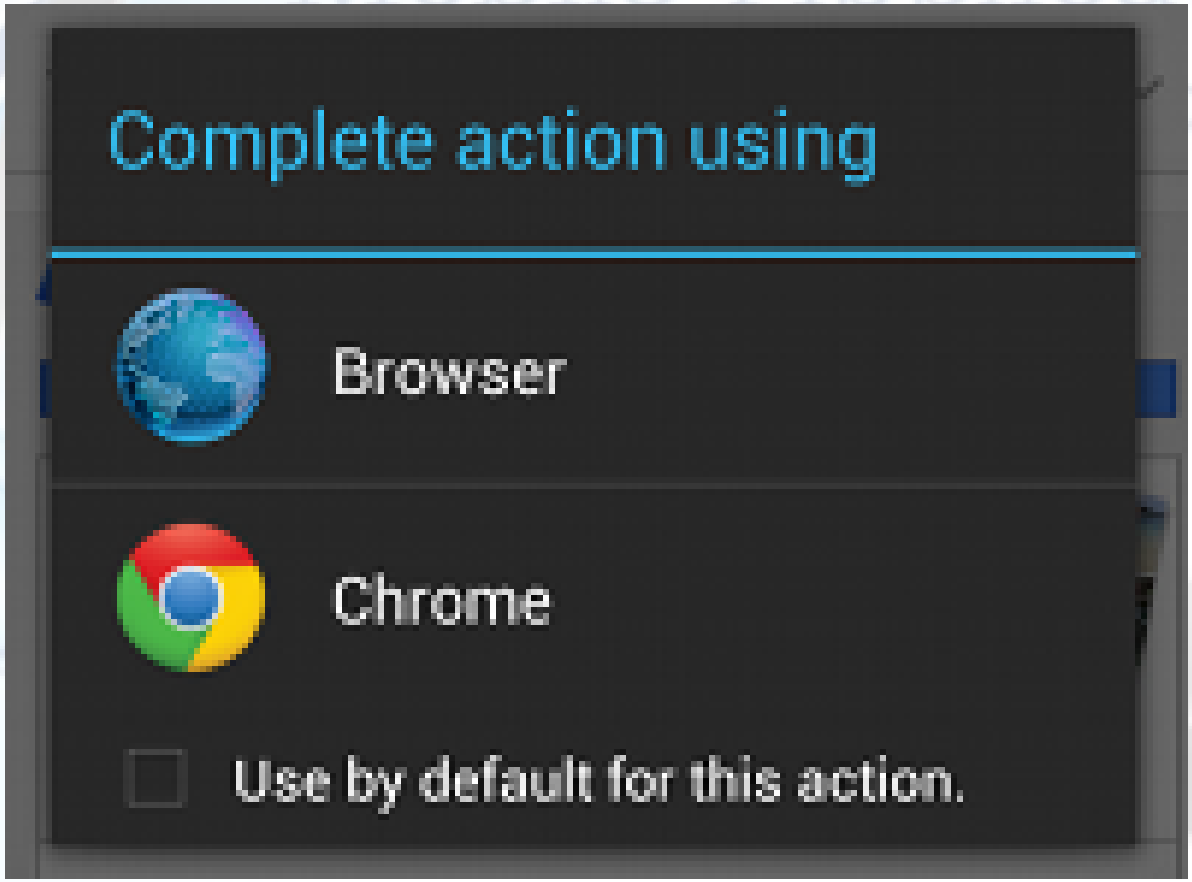
How does implicit Intent work?

1. The Android Runtime keeps a list of registered Apps
2. Apps have to register via AndroidManifest.xml
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 lets the user choose one
6. Android runtime starts the requested activity

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.

App Chooser


Complete action using
Browser
Chrome
Use by default for this action.

Intent actions, categories, and data

Fields used by an implicit Intent are:

- The Intent action: Which is the generic action the receiving Activity should perform. A common Intent action is ACTION_VIEW. The available Intent actions are defined as constants in the Intent class and begin with the word ACTION_.

Intent actions, categories, and data

- An Intent category: Which provides additional information about the category of component that should handle the Intent. Intent categories are also defined as constants in the Intent class and begin with the word CATEGORY_.

Intent actions, categories, and data

- The data type: Which indicates the MIME type of data the Activity should operate on.

Sending an implicit Intent

1. In the sending Activity, create a new Intent object.
2. Add information about the request to the Intent object, such as data or extras.
3. Send the Intent with startActivity() (to just start the Activity) or startActivityforResult() (to start the Activity and expect a result back).

Create implicit Intent objects

```
Intent sendIntent = new Intent();
```

You can also create the Intent object with a specific action:

```
Intent sendIntent = new
Intent(Intent.ACTION_VIEW);
```

Create implicit Intent objects

```
Intent sendIntent = new Intent();
sendIntent.setAction(Intent.ACTION_SEND);
sendIntent.putExtra(Intent.EXTRA_TEXT,
textMessage);
sendIntent.setType("text/plain");
```

Resolve the Activity before starting it

To verify that an Activity or other component is available to receive your Intent, use the resolveActivity() method with the system package manager like this:

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

Implicit Intent 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);
```

Receiving an implicit Intent

- If you want an Activity in your app to respond to an implicit Intent (from your own app or other apps), declare one or more Intent filters in the AndroidManifest.xml file.

Receiving an implicit Intent

- Each Intent filter specifies the type of Intent it accepts based on the action, data, and category for the Intent.
- The system will deliver an implicit Intent to your app component only if that Intent can pass through one of your Intent filters.

Receiving an implicit Intent

Note: An explicit Intent is always delivered to its target, regardless of any Intent filters the component declares. Conversely, if an Activity does not include Intent filters, it can only be launched with an explicit Intent.

Intent filters

- Define Intent filters with one or more <intent-filter> elements in the AndroidManifest.xml file, nested in the corresponding <activity> element.
- Inside <intent-filter>, specify the type of intent your activity can handle.

Intent filters

- The Android system matches an implicit intent with an activity or other app component only if the fields in the Intent object match the Intent filters for that component.

Intent filters

An Intent filter may contain the following elements:

- <action>: The Intent action that the activity accepts.
- <data>: The type of data accepted, including the MIME type or other attributes of the data URI (such as scheme, host, port, and path).
- <category>: The Intent category.

Intent filters

- An Intent filter can declare zero or more <action> elements for the Intent action.
- The action is defined in the name attribute, and consists of the string "android.intent.action."

```
<intent-filter>
    <action android:name="android.intent.action.EDIT" />
    <action android:name="android.intent.action.VIEW" />
</intent-filter>
```

Intent filters

- An Intent filter can declare zero or more <category> elements for Intent categories.
- The category is defined in the name attribute, and consists of the string "android.intent.category."

```
<intent-filter>
    <category android:name="android.intent.category.DEFAULT" />
    <category android:name="android.intent.category.BROWSABLE" />
</intent-filter>
```

Intent filters

Note that any Activity that you want to accept an implicit Intent must include the android.intent.category.DEFAULT Intent filter. This category is applied to all implicit Intent objects by the Android system.

Intent filters

- An Intent filter can declare zero or more <data> elements for the URI contained in the Intent data.
- As the Intent data consists of a URI and (optionally) a MIME type, you can create an Intent filter for various aspects of that data, including:
  - URI Scheme
  - URI Host
  - URI Path
  - Mime type

Intent filters

```
<intent-filter>
    <data android:mimeType="video/mpeg" android:scheme="http" />
    <data android:mimeType="audio/mpeg" android:scheme="http" />
</intent-filter>
```

Intent filters

```
<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` – Match one or more action constants
  - `android.intent.action.VIEW` – matches any Intent with `ACTION_VIEW`
  - `android.intent.action.SEND` – matches any Intent 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` – Filter on data URIs, MIME type
  - `android:scheme="https"`—require URIs to be https protocol
  - `android:host="developer.android.com"`—only accept an Intent 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 and 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>
```

Sharing data using ShareCompat.IntentBuilder

- Share actions are an easy way for users to share items in your app with social networks and other apps.
- Although you can build a share action in your own app using an implicit Intent with the ACTION_SEND action, Android provides the ShareCompat.IntentBuilder helper class to easily implement sharing in your app.

Sharing data using ShareCompat.IntentBuilder

```
String txt = mEdtTxtShare.getText().toString();
String mimeType = "text/plain";
ShareCompat.IntentBuilder
        .from(this)
        .setType(mimeType)
        .setChooserTitle("Share this text with: ")
        .setText(txt)
        .startChooser();
```


Mobile Application
Development
THANK YOU