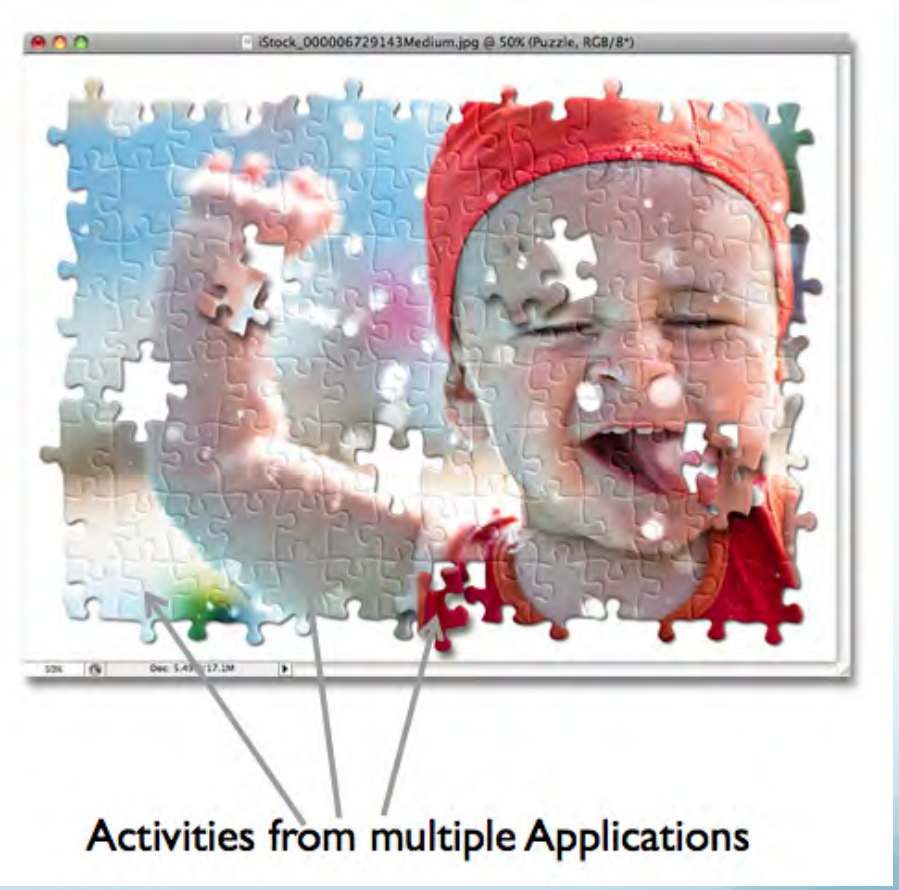
Pengembangan Aplikasi Perangkat Bergerak: Event Driven Development

Aryo Pinandito, ST, M.MT

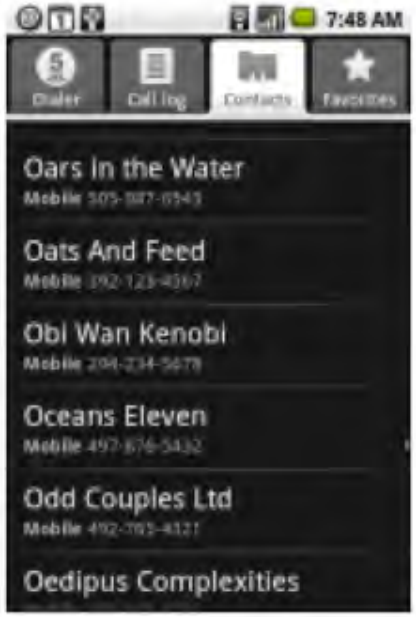
Android Component Model

- Android uses a component model based on Activities, Services, and ContentProviders
- Each application runs in a separate process and is composed of one or more of these components
- These components can be reused outside of the application that provides them
- This reuse approach allows developers to quickly build apps from pieces of other apps

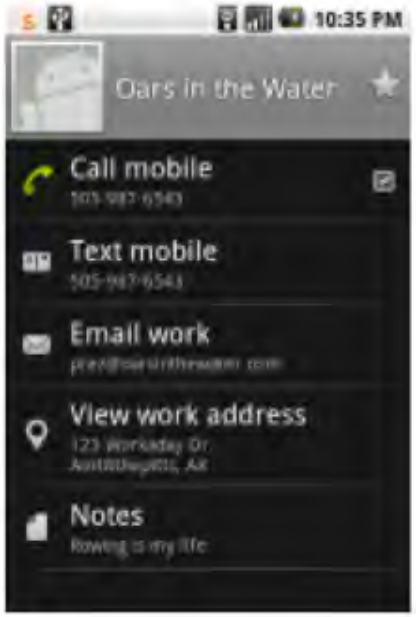


Screens are Managed by Activity

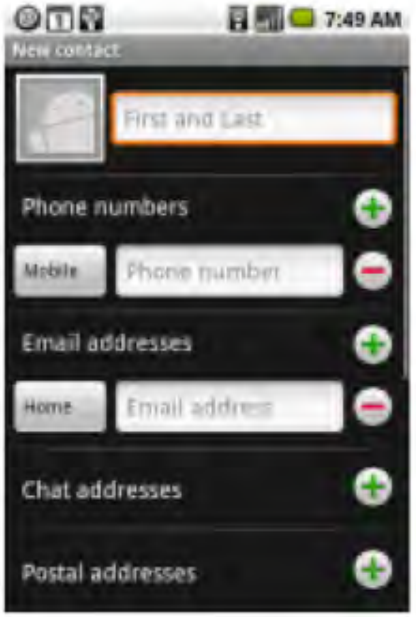
Dialer



Contacts



View Contact



New Contact

Activities are Event Driven

- Your Activities provide event handling code to process events
- Your primary work is to create one or more Activities
- Your activities DO NOT control the flow of program execution but instead respond to key events
- Each Activity sets up a GUI in its onCreate() method (e.g. the “you are being created” event)
- Event listeners are added to the GUI elements so that your code can respond to user input
- Sensor listeners are registered in onCreate()
- Subsequent code in your Activity is triggered by your event listeners
- Lifecycle methods register/unregister event listeners

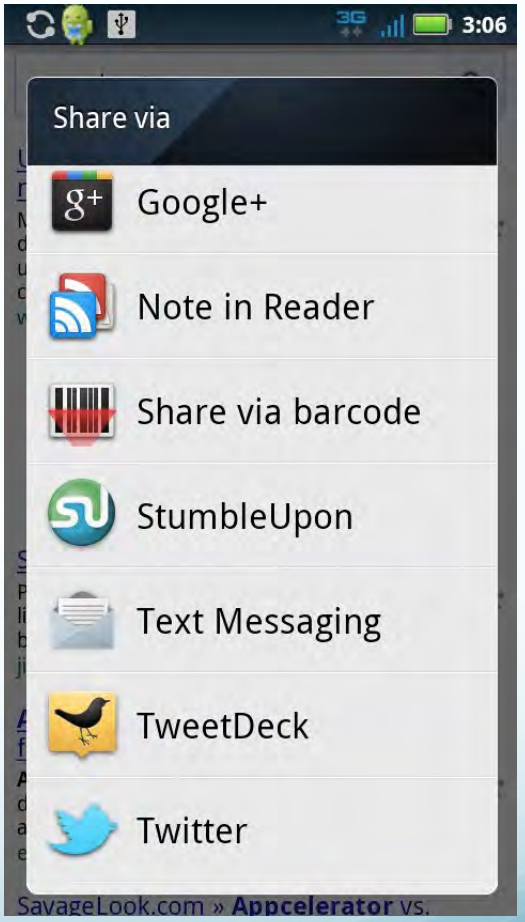
Event Handlers (Listeners)

```
public class LoginActivity extends Activity {

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

  Button signInButton = (Button) findViewById(R.id.sign_in_button);
  final TextView label = (TextView) findViewById(R.id.label_text);

  signInButton.setOnClickListener(
    new View.OnClickListener() {
      public void onClick(View view) {
         label.setText('loading...');
         attemptLogin();
      }
    });
```

Android Intent

- All Activites are launched via the sending of an event data structure called an *Intent*
- An application can leverage existing Activities by sending Intents
- An Intent is a message object that is sent to the Android platform to tell it that you want to complete a specific action

- Support interaction between any application components available on an Android device
  - start a new Activity
  - broadcast messages (broadcast intents)



Intents & Intents Filter

- Intents: request for an action to be performed (usually on a set of data)
- Intent Filters : register Activities, Services, and Broadcast Receivers (as being capable of performing an action on a set of data)
- Broadcast Receivers : listens to intent
- For example, you can send an Intent to Android to tell it that you want to send an email
- When you send Android an Intent, it figures out which Activity needs to be run in order to complete the action described by the Intent

Intents

Starting New Activity

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

Intents

- Dial a phone number

```
Intent intent = new Intent (Intent.ACTION_DIAL,
    Uri.parse(“tel:93675359”));
startActivity(intent);
```

- Open a website URL to a web browser

```
Intent intent = new Intent (Intent.ACTION_VIEW,
    Uri.parse(“http://codeandroid.org”));
startActivity(intent);
```

Intent: Launching an Activity

- Launching an activity

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

- Launching an activity with extras (data)

```
Intent intent = new Intent (this, HelloWorld.class);
intent.putExtra("title", "Hello World App Title");
startActivity(intent);
```

Intent Filters

- Required for Intent resolution to match Intents to Activities, Services, or BroadcastReceivers
- Most Intent Filters are declared in AndroidManifest.xml of an application

Intent Filters

- AndroidManifest.xml

```
<activity android:name=".HelloWorld"
    android:label="@string/app_name">
    <intent-filter>
        <action
android:name="org.codeandroid.intentstest.HelloWorld"/>
        <category
android:name='android.intent.category.DEFAULT"/>
    </intent-filter>
</activity>
```

- Launch Hello World

```
Intent intent =
    New Intent("org.codeandroid.intentstest.HelloWorld");
startActivity(intent);
```

Intent Filters

- AndroidManifest.xml

```
<activity android:name=”.HelloWorld”
    android:label=”@string/app_name”>
<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=”http” android:host=”androidium.org”/>
</intent-filter>
</activity>
```

- Launch Hello World

```
Intent intent =
    New Intent(Intent.ACTION_VIEW, Uri.parse(“http://androidium.org”));
startActivity(intent);
```

Intent Filters: Custom Scheme

- AndroidManifest.xml

```
<activity android:name=”.HelloWorld”
    android:label=”@string/app_name”>
<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=”helloandroid”/>
</intent-filter>
</activity>
```

- Launch Hello World

```
Intent intent =
    New Intent(Intent.ACTION_VIEW, Uri.parse("helloandroid://"));
startActivity(intent);
```

Intent Filters: Launching Market URI

```
Uri marketUri =
    Uri.parse(“http://market.android.com/search?
q=pname:com.buuuk.buUuk”)
Intent intent = new Intent (Intent.ACTION_VIEW, marketUri);
startActivity(intent);
```

or

```
Uri marketUri =
    Uri.parse(“market://search?q=pname:com.buuuk.buUuk”)
Intent intent = new Intent (Intent.ACTION_VIEW, marketUri);
startActivity(intent);
```

Broadcast Intents

- broadcast messages between components with the sendBroadcast method
- makes an application more open, by broadcasting to current and other applications

```
Intent intent = new
Intent("org.codeandroid.intentstest.TestBroadcastReceiver");
sendBroadcast(intent);
```

Broadcast Receivers

- Listen to Broadcast Intents
- Must be registered (either in code or within the app manifest)
- Use Intent Filter to specify which Intents it is listening for

Broadcast Receivers: Registered Inside Code

```
IntentFilter filter = new
IntentFilter("org.codeandroid.intentstest.TestBroadcastReceiver");
TestBroadcastReceiver receiver = new TestBroadcastReceiver();
registerReceiver(receiver, filter);

public class TestBroadcastReceiver extends BroadcastReceiver
{
  @Override
  public void onReceive(Context context, Intent intent) {

    // do something with broadcast intent...

  }
}
```

Broadcast Receivers: Registered in App Manifest

```
<receiver android:name=”CameraPressedReceiver”>
  <intent-filter>
    <action android:name=”android.intent.action.CAMERA_BUTTON” />
  </intent-filter>
</receiver>

public class CameraPressed extends BroadcastReceiver {

  @Override
  public void onReceive(Context context, Intent intent) {

  // do something ...

  }
}
```

Any Questions?

Grazias

Kiitos

감사합니다

Gratias

Danke

شكراً

Terima Kasih

谢谢

Merci

धन्यवाद

Thank You

ありがとうございます