
# Entwicklung von Android Anwendungen für Java Entwickler

Jochen Hiller, Deutsche Telekom AG

# Android Development for Java Developers



Source: http://duke.kenai.com/misc/DukePhoning.png

# About Me

- Jochen Hiller
  - jo.hiller@googlemail.com
- Developer Evangelist,
  Deutsche Telekom AG
  - www.developergarden.com


ANDROID

# Introduction

# What is Android?

# Software for making phone calls

# A network stack and Internet client

# A platform for running code

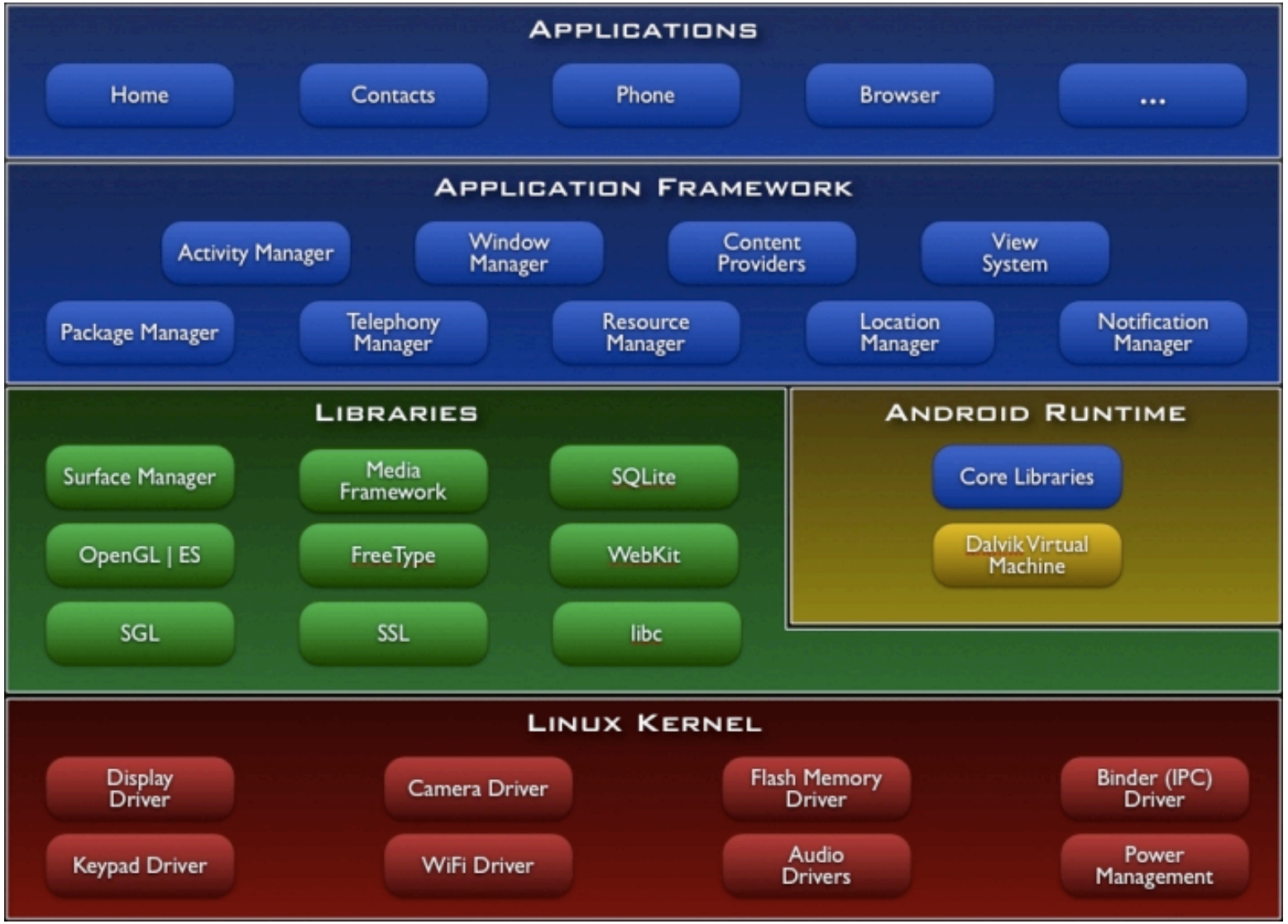
# What else is Android?

open handset alliance

The first *complete*, *open* and *free* mobile platform.

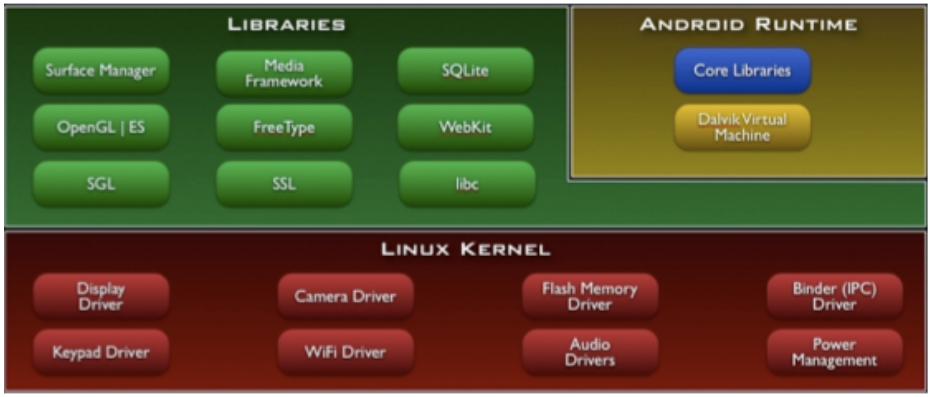
- *Open*: Android source code available
- *Free*: Licensed under Apache 2.0
- Hardware vendor independent
- Initiated by Google

# The Android Architecture



# Some characteristics

- based on Linux 2.6.x kernel
- Mobile Hardware support (GSM, WiFi, GPS, Camera, Bluetooth, USB, ...)

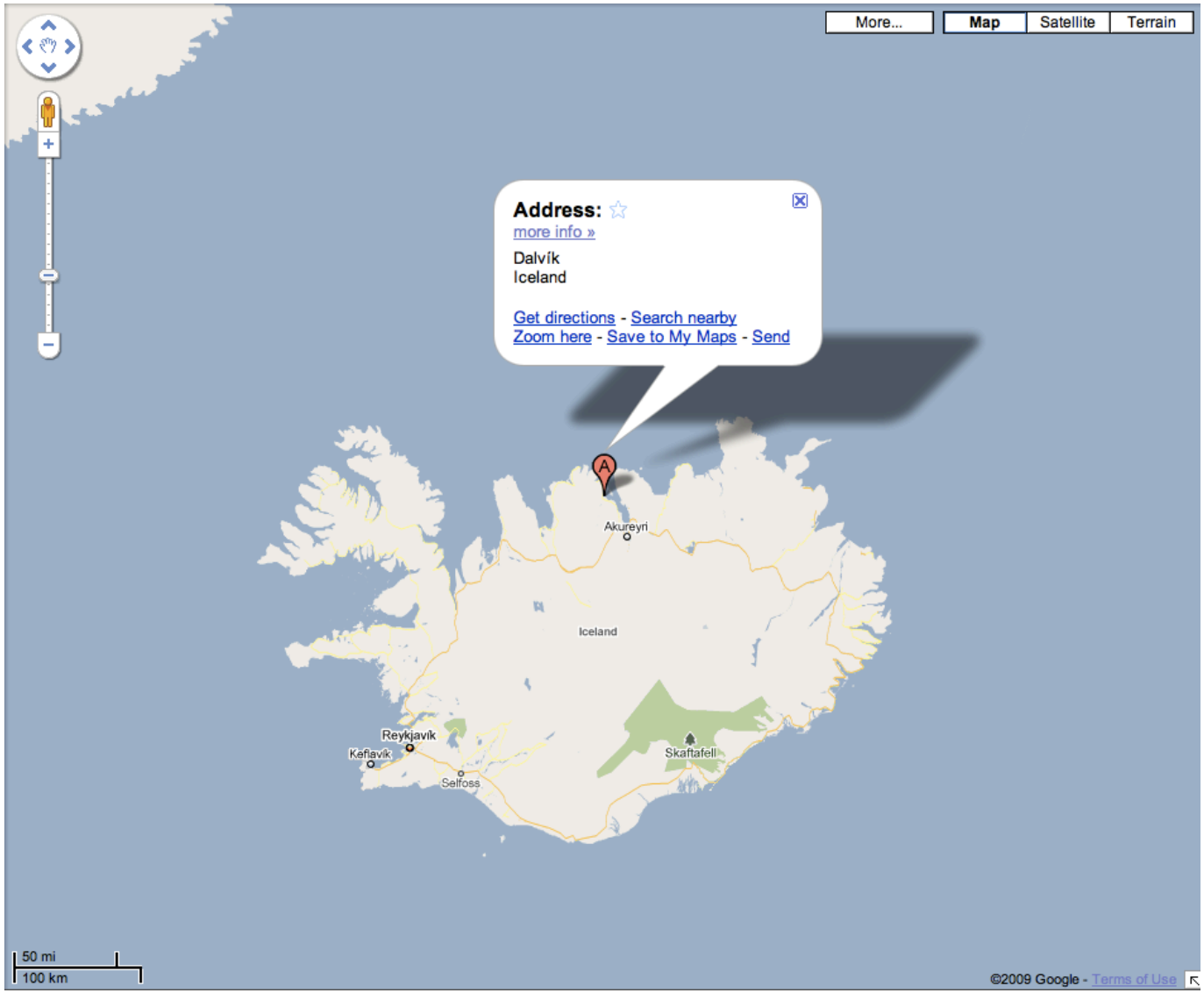



- Integrated Browser (WebKit Engine)
- Graphics (OpenGL/ES), DB, Media, ... Support
- Dalvik Virtual Machine

# What is Dalvik?

# ~~What~~ Where is Dalvik?

# ~~What~~ Where is Dalvik?

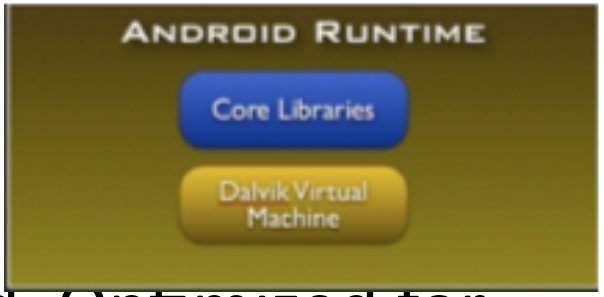


Dalvik was written by Dan Bornstein, who named it after the fishing village of Dalvík in Eyjafjörður, Iceland, where some of his ancestors lived.



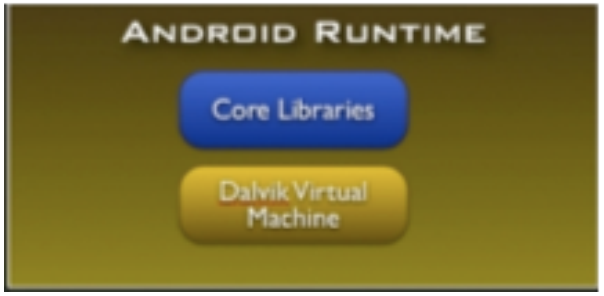
Source: Wikipedia, http://en.wikipedia.org/wiki/Dalvik_virtual_machine

# Dalvik – An Overview




- Dalvik is a Virtual Machine, like JVM, .NET CLR
- Register-based VM, not stack based. Optimized for embedded environments
- Memory-protection, memory-optimized
- Garbage-collection supported
- Lifecycle management of applications
- Translator (dx) from Java bytecode to Dalvik bytecode
- Since 2.2: JIT Compiler available

# Runtime – Core Libraries




- Java required core packages (`java.*`)
- Android specific libraries in `android.*` namespace
- Java libraries based on Apache Harmony and other Open Source implementations
- Most Java 1.5 language features supported
- Java based application framework

# Application Framework


- *Views, Layout Manager* – UI elements
- *Activities* – screen of an Android application
- *Intents* – provide / requests services from other applications
- *Resource Manager* – handles all text and graphical resources in an optimized way
- *Services* – background activities
- *Content Providers* – provides data to applications

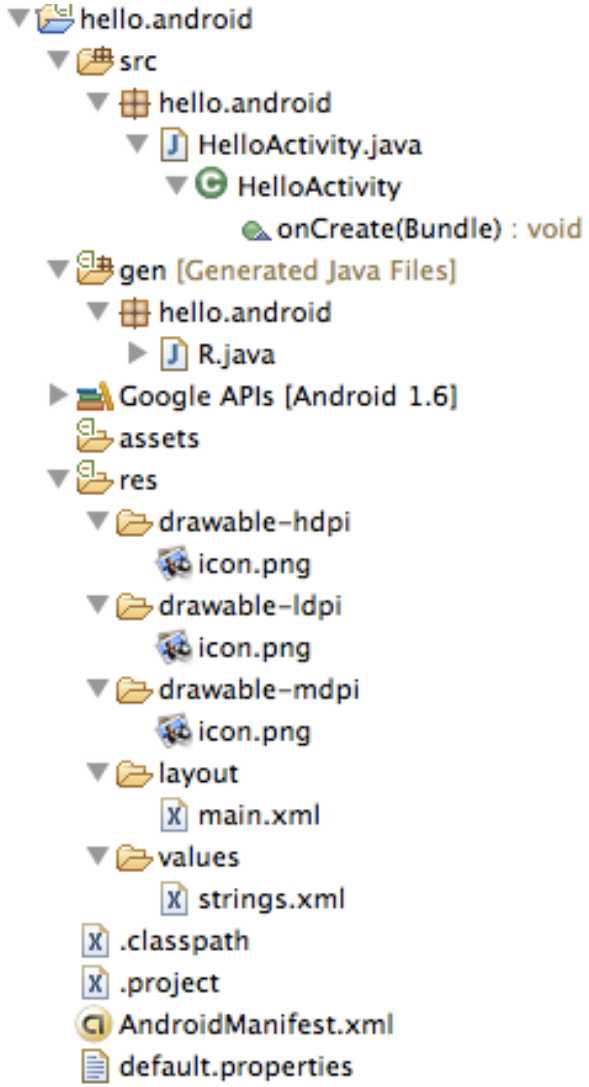
# Tool chain

# Tool chain

- Android SDK (SDK r6)
  - Complete tool chain (emulator, shell, deployment tools, …)
  - Supported for Win, Mac, Linux
- Eclipse IDE (>=3.4)
- Android Development Toolkit (ADT) (0.9.7)
  - Wizards, debugging, logging, emulator control
  - Thread and Memory analysis

# Project layout

- R.java generated from res folder, compiled code references to resources
- Layout, values, resources may be density / locale dependent
- Resources may be localized
- Assets for all static resources (media, html, JavaScript, other web resources, …)



# AndroidManifest.mf

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android=http://schemas.android.com/apk/res/android
    package="hello.android“
    android:versionCode="1“
    android:versionName="1.0">
  <application
      android:icon="@drawable/icon" android:label="@string/app_name">
    <activity
        android:name=".HelloActivity" android:label="@string/app_name">
      <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
    </activity>
  </application>
  <uses-sdk android:minSdkVersion="4" />
</manifest>
```

# AndroidManifest.mf

Reference to drawable resources

Reference to external text resources

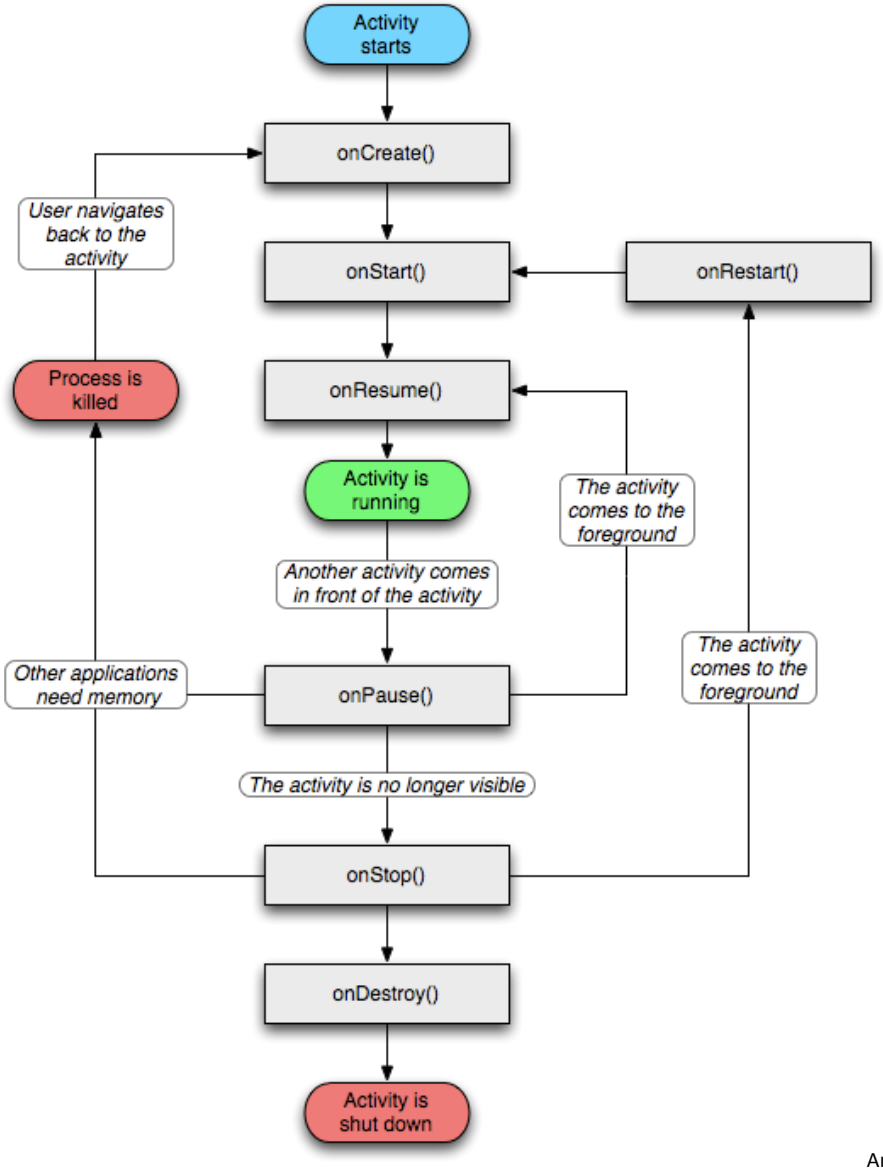
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android=http://schemas.android.com/apk/res/android
    package="hello.android“
    android:versionCode="1“
    android:versionName="1.0">
  <application
      android:icon="@drawable/icon" android:label="@string/app_name">
    <activity
        android:name=".HelloActivity" android:label="@string/app_name">
      <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
    </activity>
  </application>
  <uses-sdk android:minSdkVersion="4" />
</manifest>
```

“Main” activity to be launched when starting app

API compatibility

Activity may start app. It will be present in app launcher

# Activity Lifecycle




- on<Method> Callbacks
- Android may kill process due to lacking resources
- onPause/onResume: persist and restore state
- onStart/onStop: activity is visible / hidden to the user

# HelloActivity.java

```
package hello.android;

import android.app.Activity;
import android.os.Bundle;

public class HelloActivity extends Activity {

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        Button aboutButton = (Button) findViewById(R.id.about);
        aboutButton.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                Log.e("hello", "about button clicked");
                showDialog(DIALOG_ABOUT);
            }
        });
    }
}
```

# HelloActivity.java

onCreate: set content view

Reference to an view id

```
package hello.android;

import android.app.Activity;
import android.os.Bundle;

public class HelloActivity extends Activity {

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        Button aboutButton = (Button) findViewById(R.id.about);
        aboutButton.setOnClickListener(new View.OnClickListener() {
        public void onClick(View v) {
            Log.e("hello", "about button clicked");
            showDialog(DIALOG_ABOUT);
            }
        });
    }
}
```

Logging with category, message

# UI: main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android=
    "http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    >
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/hello"
    />
 <Button
    android:id="@+id/about“
    android:text="@string/about.button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />
</LinearLayout>
```

# UI: main.xml

Reference to an view id

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android=
    "http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    >
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/hello"
    />
 <Button
    android:id="@+id/about“
    android:text="@string/about.button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />
</LinearLayout>
```

Reference to external text resources

# Resources: res/values/strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello World, HelloAndroidActivity!</string>
    <string name="app_name">HelloAndroid</string>
    <string name="about.button">About</string>
    <string name="about.dialog.title">About</string>
    <string name="about.dialog.message">
        Hello Android sample for JFS 2010
    </string>
    <string name="about.dialog.ok_button">OK</string>
</resources>
```

# Resources: res/values/strings.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello World, HelloAndroidActivity!</string>
    <string name="app_name">HelloAndroid</string>
    <string name="about.button">About</string>
    <string name="about.dialog.title">About</string>
    <string name="about.dialog.message">
        Hello Android sample for JFS 2010
    </string>
    <string name="about.dialog.ok_button">OK</string>
</resources>
```

May be localized
(e.g. res/values-de/strings.xml)

Resources (like properties)

# Selected APIs

# Intents

- Intents to start Activities, Services, Broadcasts
  - Inter-App communication, loosely coupled
- Intents are described by:
  - ComponentName, Action, Data (Uri, Mime-Types), Category
- Intents resolved by Application Framework
- Get Results from a started Intent
- Intent-Filtering: hints for Intent resolver
- OpenIntents: “Marketplace” for Intents

# Intent – Start Activity

```
// start activity to pick up a contact
addContactButton.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {

        // start activity to pick up a contact
        Intent intent = new Intent(Intent.ACTION_PICK,
            Contacts.CONTENT_URI);
        startActivityForResult(intent, PICK_CONTACT_REQUEST);

    }
});
```

# Intent – Start Activity

```
// start activity to pick up a contact
addContactButton.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {

        // start activity to pick up a contact
        Intent intent = new Intent(Intent.ACTION_PICK,
            Contacts.CONTENT_URI);
        startActivityForResult(intent, PICK_CONTACT_REQUEST);

    }
});
```

Uri will be matched to registered Intents

Actions, may use Android predefined Actions

# Intent – Get Result

```
protected void onActivityResult(int requestCode,
                                int resultCode, Intent data) {
    if (requestCode == PICK_CONTACT_REQUEST && resultCode == RESULT_OK) {
        Uri contactUri = data.getData();
        // Load the display name for the specified person
        Cursor cursor = getContentResolver().query(contactUri,
            new String[] { Contacts._ID, Contacts.DISPLAY_NAME }, null,
            null, null);
        try {
            if (cursor.moveToFirst()) {
                long contactId = cursor.getLong(0);
                String name = cursor.getString(1);
            }
        } finally {
            cursor.close();
        }
        // Load more...
    }
}
```

# Intent – Get Result

```
protected void onActivityResult(int requestCode,
                                int resultCode, Intent data) {
    if (requestCode == PICK_CONTACT_REQUEST && resultCode == RESULT_OK) {
        Uri contactUri = data.getData();
        // Load the display name for the specified person
        Cursor cursor = getContentResolver().query(contactUri,
            new String[] { Contacts._ID, Contacts.DISPLAY_NAME }, null,
            null, null);
        try {
            if (cursor.moveToFirst()) {
                long contactId = cursor.getLong(0);
                String name = cursor.getString(1);
            }
        } finally {
            cursor.close();
        }
        // Load more...
    }
}
```

Callback method

Result from Intent: Contact as URI

Query a content resolver

# Location

- LocationManager as a system service
- Multiple LocationProvider (device dependent)
  - Network: based on WiFi hotspots triangulation
  - GPS: based on satellites
  - Passive (since 2.2)
- Choose best provider for criteria:
  - Accuracy, power consumption
- LocationListener: track changes of location/providers
  - For energy efficiency: longer intervals for location update
  - Minimal location change for notification

# Location, Location

- Application must express need for location access
- User must give consent during installation process (only once!)
- Extend AndroidManifest.mf for required permissions

```
<uses-permission
    android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission
    android:name="android.permission.ACCESS_FINE_LOCATION" />
```

Different accuracy may be requested

# Location, Location, Location

```
// get location manager
LocationManager locManager = (LocationManager)
                                getSystemService(LOCATION_SERVICE);

// there are more providers: network, gps
List<String> providers = locManager.getAllProviders();

// choose best provider for accurancy, power consumption
Criteria criteria = new Criteria();
criteria.setPowerRequirement(Criteria.POWER_LOW);
String bestProvider = locManager.getBestProvider(criteria, true);

// last location will be cached
Location location = locManager.getLastKnownLocation(bestProvider);

// register and unregister a location listener
// Start updates (doc recommends delay >= 60000 ms)
// 1 means notify when location change more than 1 meter
locManager.requestLocationUpdates(
    bestLocationProvider, 15000, 1, aListener);

// Stop updates to save power while app paused
locManager.removeUpdates(aListener);
```

# SQLite

- Android provides SQLite database
- Security sandbox:
  Databases are visible to application only
  - `/data/data/<appname>/databases`
- SQLite Helper classes implementation
- Supports cursor for incremental data access
- No JDBC layer! As simple as possible!

# SQLite – Sample code

```
public class DBRoute extends SQLiteOpenHelper {

  public void onCreate(SQLiteDatabase db) {
    db.execSQL("CREATE TABLE " + TABLE_NAME + " (" + _ID
    + " INTEGER PRIMARY KEY AUTOINCREMENT, " + COLUMN_TIME
    + " INTEGER," + COLUMN_LONGITUDE + " NUMERIC, "
    + COLUMN_LATITUDE + " NUMERIC, " + COLUMN_ALTITUDE
    + " NUMERIC, " + COLUMN_DESCRIPTION + " TEXT);");
  }
  public void addLocation(Location loc, String desc) {
    // Insert a new record into the route data source.
    // You would do something similar for delete and update.
    SQLiteDatabase db = getWritableDatabase();
    ContentValues values = new ContentValues();
    values.put(COLUMN_TIME, System.currentTimeMillis());
    values.put(COLUMN_LONGITUDE, loc.getLongitude());
    values.put(COLUMN_LATITUDE, loc.getLatitude());
    values.put(COLUMN_ALTITUDE, loc.getAltitude());
    values.put(COLUMN_DESCRIPTION, desc);
    db.insertOrThrow(TABLE_NAME, null, values);
  }
}
```

# Google Maps integration

- Google Maps supported at Android
- Note: Not part of Android, but of Google APIs!
  - Requires correct Virtual Device configuration
- Very easy integration approach
- Google Maps API key required
- Good documentation, easy going!

# Google Maps – Enable it

- Google Maps is an additional Library
- Addition library support is required for hardware vendors to plugin their libraries
- Extend AndroidManifest.mf permissions and libraries

```
<uses-permission
    android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission
    android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission
     android:name="android.permission.INTERNET" />
<application
    android:icon="@drawable/icon" android:label="@string/app_name">
    ...
    <uses-library android:name="com.google.android.maps" />
</application>
```

Libraries for applications

# Google Maps – Sample code (1)

```
public class MyLocationActivity extends MapActivity {

  private MapView map;
  private MapController controller;

  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);
    initMapView();
    initMyLocation();
  }

  /** Find and initialize the map view. */
  private void initMapView() {
    map = (MapView) findViewById(R.id.map);
    controller = map.getController();
    map.setSatellite(true);
    map.setBuiltInZoomControls(true);
  }
```

Predefined MapActivity

Map view modes

# Google Maps – Sample code (2)

```
  /** Start tracking the position on the map. */
  private void initMyLocation() {
    final MyLocationOverlay overlay = new MyLocationOverlay(this, map);
    overlay.enableMyLocation();

    overlay.runOnFirstFix(new Runnable() {
      public void run() {
        // Zoom in to current location
        controller.setZoom(8);
        controller.animateTo(overlay.getMyLocation());
      }
    });
    map.getOverlays().add(overlay);
  }

  protected boolean isRouteDisplayed() {
    // Required by MapActivity
    return false;
  }
}
```

Get my current location

Move map to my location

# Google Maps – API key

- Google Maps API key required
  - For development: based on generated debug certificate
  - For Android Market: based on your application signing certificate

```
<com.google.android.maps.MapView
      android:id="@+id/map"
      android:apiKey="1Ss8D-asdfghjklqwerertzuiapWLrs1Q"
      android:layout_width="fill_parent"
      android:layout_height="fill_parent"
      android:clickable="true" />
```

Maps API key

# Network access

- Android provides Apache HttpClient (4.x)
- SSL support included (BouncyCastle implementation)
- SOAP Stacks not (really) supported
- Preferred:
  - HTTP / REST / JSON, XML
- XML Parser, JSON Support included
- Google GData libraries available for Android to access Google services

# Make a REST call – Sample code

```
DefaultHttpClient client = new DefaultHttpClient();
String restURL = "https://gateway.developer.telekom.com/" +
                      "p3gw-mod-odg-sms/rest/production/sms";
HttpPost httppost = new HttpPost(restURL);
String authHeader = "TAuth realm=\"https://odg.t-online.de\",tauth_token"
                         + "=\"67GHD5...6432lkgdfs=="+ "\"";
httppost.setHeader("Authorization", authHeader);

List<NameValuePair> formparams = new ArrayList<NameValuePair>();
formparams.add(new BasicNameValuePair("number", number));
formparams.add(new BasicNameValuePair("message", message));
if (flash) {
    formparams.add(new BasicNameValuePair("flash", "true"));
}
httppost.setEntity(new UrlEncodedFormEntity(formparams, "UTF-8"));
HttpResponse response = client.execute(httppost);
String body = EntityUtils.toString(response.getEntity());
JSONObject obj = new JSONObject(body);
JSONObject status = obj.getJSONObject("status");
String statusCode = (String) status.get("statusCode");
String statusMessage = (String) status.get("statusMessage");
```

# Make a REST call – Sample code

```
DefaultHttpClient client = new DefaultHttpClient();
String restURL = "https://gateway.developer.telekom.com/" +
                 "p3gw-mod-odg-sms/rest/production/sms";
HttpPost httppost = new HttpPost(restURL);
String authHeader = "TAuth realm=\"https://odg.t-online.de\",tauth_token"
                    + "=\"67GHD5...6432lkgdfs=="+ "\"";
httppost.setHeader("Authorization", authHeader);

List<NameValuePair> formparams = new ArrayList<NameValuePair>();
formparams.add(new BasicNameValuePair("number", number));
formparams.add(new BasicNameValuePair("message", message));
if (flash) {
    formparams.add(new BasicNameValuePair("flash", "true"));
}
httppost.setEntity(new UrlEncodedFormEntity(formparams, "UTF-8"));
HttpResponse response = client.execute(httppost);
String body = EntityUtils.toString(response.getEntity());
JSONObject obj = new JSONObject(body);
JSONObject status = obj.getJSONObject("status");
String statusCode = (String) status.get("statusCode");
String statusMessage = (String) status.get("statusMessage");
```

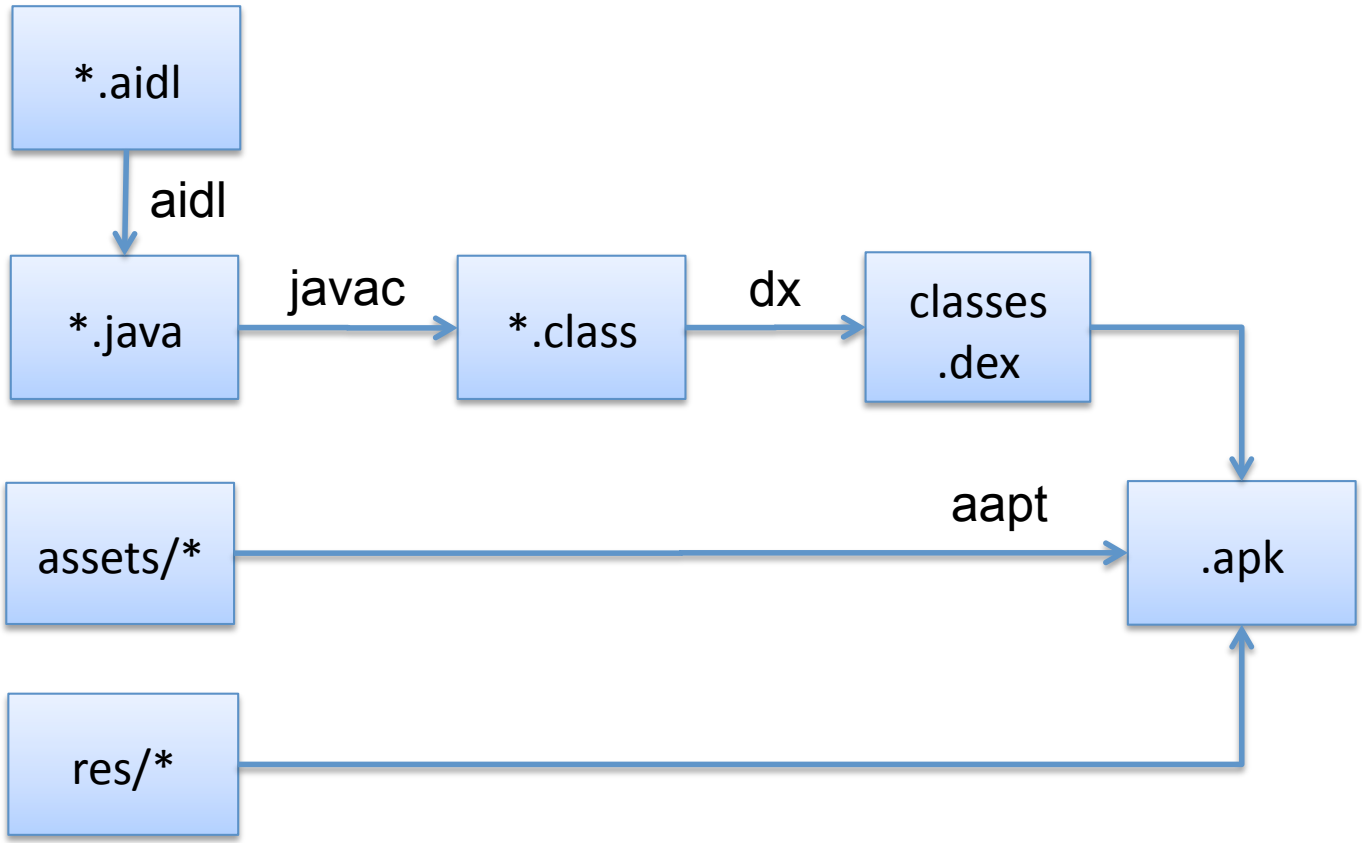
Apache HttpClient

HTML Form

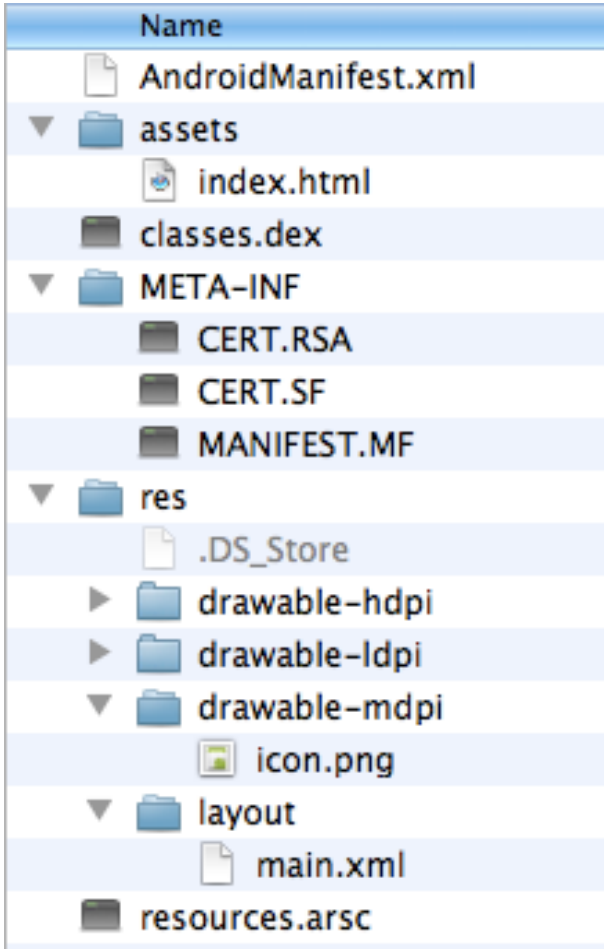
Make a POST call

org.json included

# Android packaging (1)



# Android packaging (2)




- Manifest describes application
- Assets are packed together
- Bytecode goes into classes.dex
- META-INF contains package signing
- Resources may be device density specific
- All text resources will be size optimized
- resources.arsc index's all resource files
- Build process allows individual apk based on density (optimization)

# We did not talk about…

- Services, Notifications, Handler
- Advanced UI concepts (adapters, themes, …)
- Advanced Security concepts
- Content providers, Filesystems
- Ajax, JavaScript application (Java/JS bridge)
- Bluetooth, more sensors
- Media (Audio, Video)
- Graphics 2D, OpenGL ES
- …

# Google DevPhone 2

- Available in Android Marketplace http://android.brightstarcorp.com
- 1 per developer (about 400$)
- Unlocked (SIM, Firmware)
- HTC provides firmware updates
- Shipped with Android 1.6

# References

- Android:
  - Developer: http://developer.android.com
  - Source code: http://source.android.com
- Google I/O 2008, 2009
  - http://code.google.com/events/io
- Ed Burnette: Hello Android 2nd Edition (incl. Android 2.2)
- Becker/Pant: Android: Grundlagen und Programmierung
  - v1 available as free eBook, v2 available soon

# Android Development for Java Developers

Q & A



Source: http://duke.kenai.com/misc/DukePhoning.png