LFE Medieninformatik • Andreas Butz (Dozent), Florence Balagtas-Fernandez, Gregor Broll, Alexander De Luca

Praktikum Entwicklung von Mediensystemen mit Android

Introduction to Android

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Outline

- Schedule
- Organizational Stuff
- Introduction to Android
- Exercise 1

Schedule

- Two phases: individual and team phase
- Phase 1 – Individual Phase:
  - Introduction to basics about Android
  - Exercises 1 to 4
  - Each student works on exercises himself/herself
  - Weekly meetings
- Phase 2 – Project Phase:
  - Concept and implementation of an Android application
  - Topic: mobile healthcare
  - Students work in teams
  - Regular milestone meetings

Timeline

Date	Topic / Activity
17.04.2008	Introduction and Overview of the Android Platform
24.04.2008	Implementing a User Interface
08.05.2008	Storing, Retrieving and Exposing Data
15.05.2008	Optional APIs, Google APIs and Services in Android
22.05.2008	Brainstorming, Application Design
29.05.2008	Project Phase Starts

Organizational Stuff I

- 4 SWS
- Weekly meetings
  - Thursday 10:00 s.t. – 12:00
  - Room 107, Amalienstraße 17
- Room for the practical parts:
  - Medienlabor 103, Amalienstraße 17
  - Special accounts required
  - Open during normal working times (8:00 – 17:00)
  - 1 key for each group
- Homepage:
  - http://www.medien.ifi.lmu.de/lehre/ss08/pem

Organizational Stuff II

- Students work in teams
- SVN accounts for each team
  - svn://murx.medien.ifi.lmu.de/ss08/pem/team[number] (e.g. svn://murx.medien.ifi.lmu.de/ss08/pem/team1)
- Students check their exercises in with their group‘s SVN repository
- Needed Accounts
  - SVN username
  - Medienlabor-Kennung
  - Belegungsplan Medienlabor

Teams

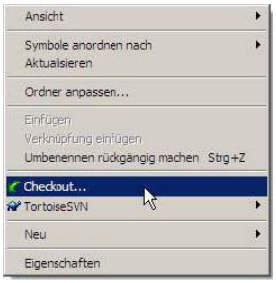
- Team 1
  - Lodde, Löhmann, Keck, Frauendienst
- Team 2
  - Anastasiu, Sauernheimer, Wallner, Tsiouprou
- Team 3
  - Kellerer, Schulz, Huber, Zehic
- Team 4
  - Alle anderen

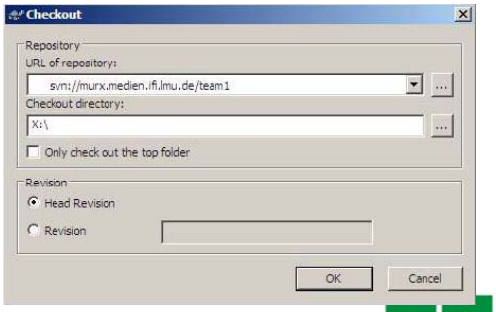
Technology – SVN I

- SVN - General
  - Version control system
  - Enables collective editing of shared source code
  - Data stored in a „Repository“ which is accessed over the network
  - Editing on local copies of the files
  - Old version available on the server
  - When possible, files will be merged automatically when edited by multiple users at the same time
  - Similar to CVS

Technology – SVN II

- SVN – First Steps (using Tortoise SVN)
  1. Download a SVN Client like Tortoise SVN for Windows
     http://tortoisesvn.net/
  2. Checkout your team repository (creates a local copy of the repository)
     Create an empty folder, open it, right-click and choose „Checkout“.





Technology – SVN III

- SVN – First Steps (using Tortoise SVN)
  3. Each time you start working perform the “Update“ command.
  4. Each time you‘re done working perform a “Commit”. Both commands are located in the right-click menu.
  5. Further functionalities are available in the right-click menu like “delete“, “rename“ and more.
     Attention: Do not use the OS-functionalities for this functions.

For further Information read the German SVN introduction by Richard Atterer, which can be found here:
http://www.medien.ifi.lmu.de/fileadmin/mimuc/mmp_ss04/Projektaufgabe/mmp-subversion.pdf

An Introduction to Android - Outline

- What is Android?
- Installation
- Getting Started
- Anatomy of an Android Application
- Life Cycle of an Android Application

What is Android?

- Released in Nov. 2007 – rumored to be some kind of GPhone
- Open, free mobile platform with a complete software stack
  - Operating system
  - Middleware
  - Key mobile applications
- Developed by the Open Handset Alliance
- Built on the open Linux kernel
- Custom Dalvik virtual machine for mobile environments
- Applications written in Java
- Open source; Apache v2 open source license
- Applications can access all core functionalities of a mobile device
- No differentiation between core and 3rd party applications
- Can be extended to incorporate new technologies

Open Handset Alliance

- Group of more than 30 technology and mobile companies led by Google
  - Mobile Operators, e.g. China Mobile, KDDI, NTT DoCoMo, T-Mobile, Sprint Nextelk, Telefonica
  - Semiconductor Companies, e.g. Broadcom, Intel, Nvidia, Qualcomm, SiRF, Texas Instruments
  - Handset Manufactureres, e.g. HTC, LG, Motorola, Samsung
  - Software Companies, e.g. eBay, Google,
- Goal: „*to accelerate innovation in mobile and offer consumers a richer, less expensive, and better mobile experience* "
- Android as the first project towards an open and free mobile experience, but also commercial deployment
- URL: www.openhandsetalliance.com/index.html

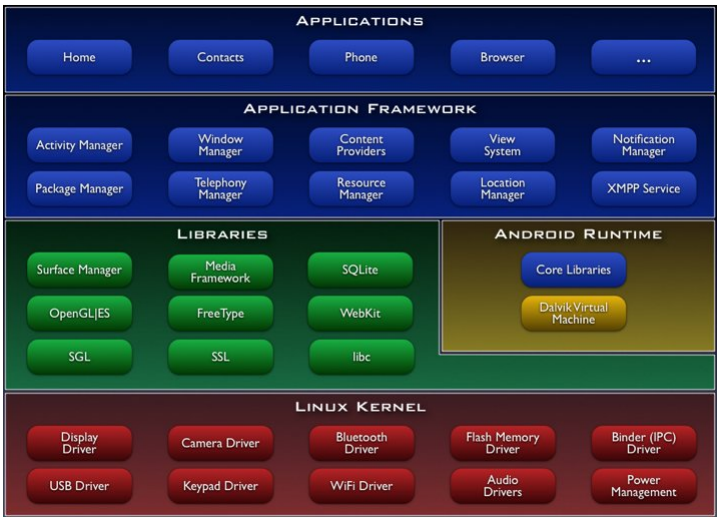
Source: www.openhandsetalliance.com/

Android Features

- **Application framework** enabling reuse and replacement of components
- **Dalvik virtual machine** optimized for mobile devices
- **Integrated browser** based on the open source WebKit engine
- **Optimized graphics** powered by a custom 2D graphics library; 3D graphics based on the OpenGL ES 1.0 specification (hardware acceleration optional)
- **SQLite** for structured data storage
- **Media support** for common audio, video, and still image formats (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF)
- **GSM Telephony** (hardware dependent)
- **Bluetooth, EDGE, 3G, and WiFi** (hardware dependent)
- **Camera, GPS, compass, and accelerometer** (hardware dependent)
- **Rich development environment** including a device emulator, tools for debugging, memory and performance profiling, and a plugin for the Eclipse IDE

Source: http://code.google.com/android/index.html

Android Architecture



Source: http://code.google.com/android/index.html

Linux Kernel

- Linux kernel version 2.6
- Abstraction layer between hardware and the software stack
- Core services
  - Security
  - Memory management
  - Process management
  - Network stack
  - Driver model

Source: http://code.google.com/android/index.html

Libraries

- C/C++ libraries used by various Android components
- Developers can use their capabilities through the application framework
- Includes:
  - Media Libraries: includes MPEG4, H.264, MP3, JPG, PNG, ...
  - WebKit/LibWebCore: web browser engine
  - SQLite: relational database engine
  - Libraries/engines for 2D and 3D graphics

Source: http://code.google.com/android/index.html

Android Runtime

- Core libraries provide Java functionalities
- Dalvik virtual machine relies on Linux kernel for e.g. threading or low-level memory management
- Devices can run multiple Dalvik VMs, every Android application runs with its own instance of Dalvik VM
- VM executes optimized Dalvik Executable files (.dex)
- Dx-tool transforms compiled Java-files into dex-files

Source: http://code.google.com/android/index.html

Applications / Application Framework

- Core applications, e.g. contacts, mail, phone, browser, calender, maps, ...
- Full access to all framework APIs for core applications
- Simplified reuse of components
- Applications written in Java

Source: http://code.google.com/android/index.html

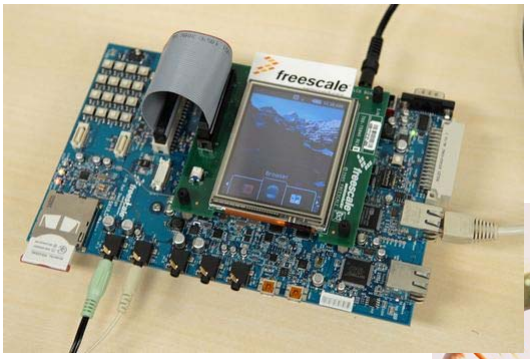
Core Android Packages

- android.util
  - contains various low-level utility classes, such as specialized container classes, XML utilities, etc.
- android.os
  - provides basic operating system services, message passing, and inter-process communication.
- android.graphics
  - is the core rendering package.
- android.text, android.text.method, android.text.style, and android.text.util
  - supply a rich set of text processing tools, supporting rich text, input methods, etc.
- android.database
  - contains low-level APIs for working with databases.
- android.content
  - provides various services for accessing data on the device: applications installed on the device and their associated resources, and content providers for persistent dynamic data.
- android.view
  - is the core user-interface framework.
- android.widget
  - supplies standard user interface elements (lists, buttons, layout managers, etc) built from the view package.
- android.app
  - provides the high-level application model, implemented using Activities.

Installing and Using the Android SDK

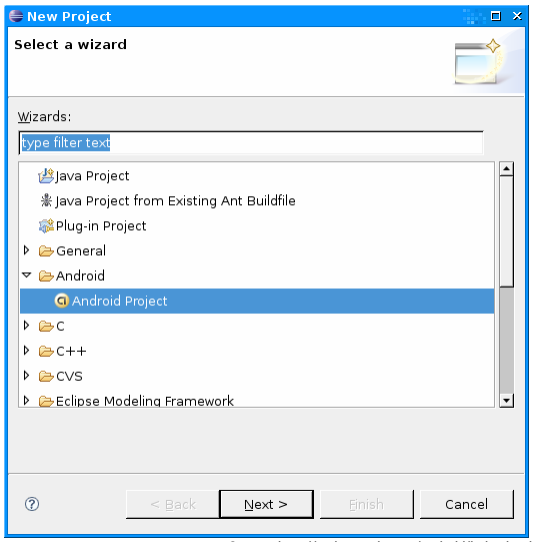
- Please follow instructions from the Android doc
- Download and install the Android SDK
- SDK includes documentation, tools and examples
- Set up your IDE; Eclipse (Java EE) recommended
- Install Eclipse Android Development Tools (ADT) plugin, connect it with the Android SDK
- Create an Android project
  - Standard Eclipse procedure
- Set up a launch configuration
  - Run application from menu or
  - Define settings for run configuration (project, activity, emulator options, ...) from Run > Open Run Dialog >
- Run Android application in emulator

No Real Android-Phones yet



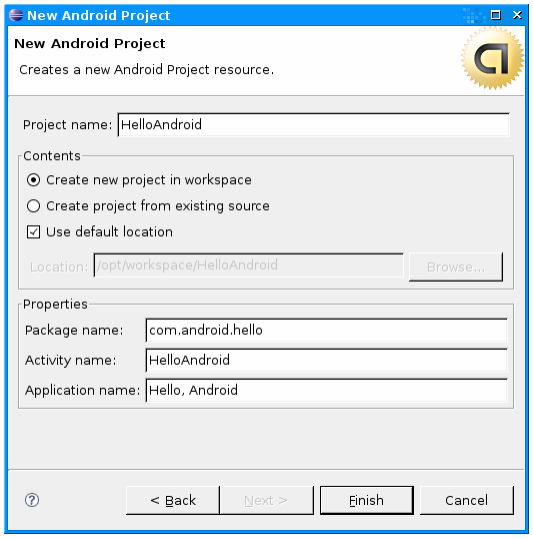
Sources: http://asia.cnet.com, www.google-phone.com

Hello Android I



Source: http://code.google.com/android/index.html

Hello Android II



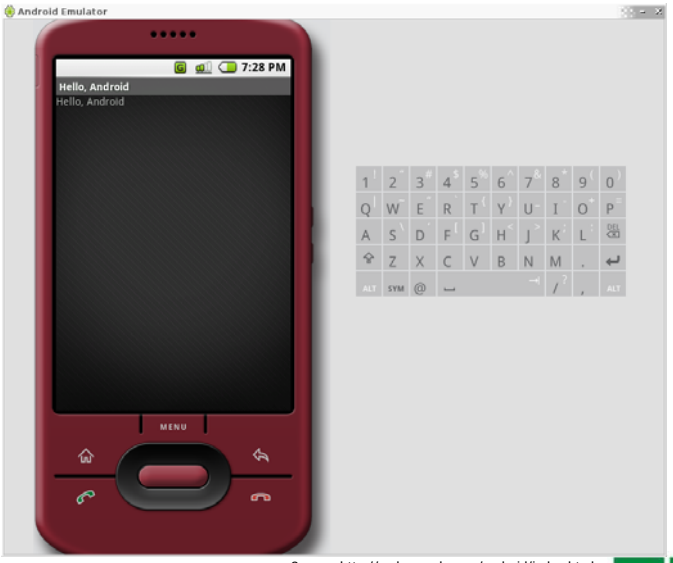
Source: http://code.google.com/android/index.html

Hello Android III

```
package com.android.hello;

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

public class HelloAndroid extends Activity {
   /** Called when the activity is first created. */
   @Override
   public void onCreate(Bundle icicle) {
       super.onCreate(icicle);
       TextView tv = new TextView(this);
       tv.setText("Hello, Android");
       setContentView(tv);
   }
}
```

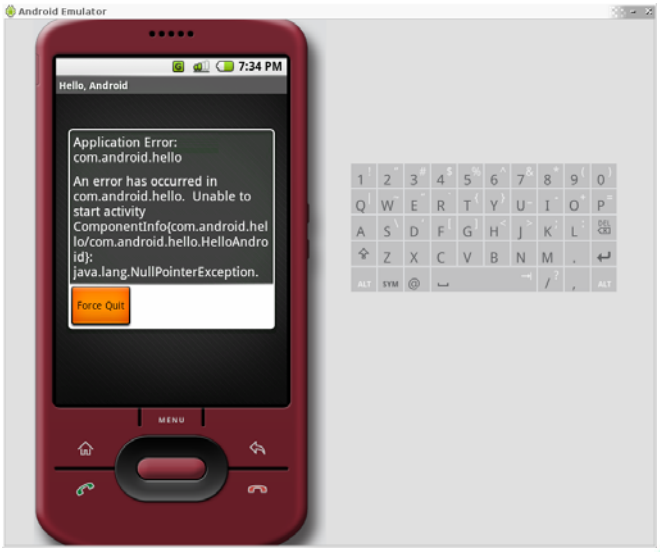
Source: http://code.google.com/android/index.html

Hello Android IV



Source: http://code.google.com/android/index.html

Hello Android V



Source: http://code.google.com/android/index.html

Anatomy of an Android Application

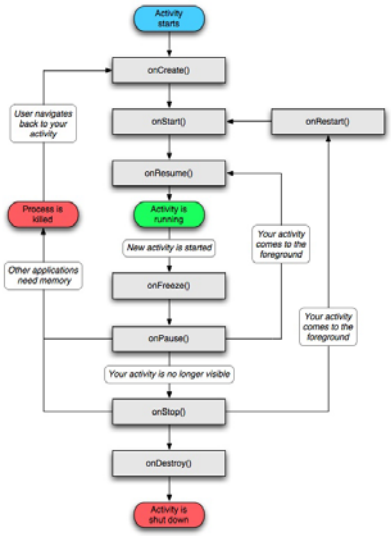
- 4 main building blocks for Android applications
  - Activity
  - Intent Receiver
  - Service
  - Content Provider
- AndroidManifest.xml lists all components of an application, their capabilities and requirements

```
<?xml version="1.0" encoding="utf-8"?>

<manifest xmlns:android="http://schemas.android.com/apk/res/android"
        package="com.my_domain.app.helloactivity">

    <application android:label="@string/app_name">

        <activity android:name=".HelloActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>
                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>

    </application>

</manifest>
```

Source: http://code.google.com/android/index.html

Activity

- Single, focused thing or task
- Extends the Activity base class
- Refers to a single screen in a (multi-screen) application
- Displays a UI, interacts with user, responds to events
- 2 main methods:
  - onCreate(Bundle): initialization of activity, set UI, ...
  - onPause(): leaving an activity
- Moving through screens by starting other activities
- Activities managed by activity stack
- New activity put on top of the stack
- 4 states: active/running, paused, stopped, killed/shut down



Source: http://code.google.com/android/index.html

Intents and Intent Filters

- Intent
  - Abstract description of an operation/action to be performed
  - Mostly used for launching activities; "glue between activities"
  - Action: general action to be performed, e.g. VIEW_ACTION, EDIT_ACTION, MAIN_ACTION, ...
  - Data: data to operate on, expressed as a URI
  - Example: **VIEW_ACTION** ***content://contacts/1***

- Intent Filter
  - Describes what Intents an activity can handle
  - Activities publish Intent Filters describing their capabilities/ how they can handle certain Intents and their actions
  - Navigating between screens is accomplished by resolving Intents => system matches Intents and Intent Filters
  - Activity calls method startActivity(myIntent)

Intent Receiver, Service, Content Provider

- Intent Receiver
  - Used to execute code upon an external event, e.g. phone rings
  - Usually no UI; may use the NotificationManager
- Service
  - Application component running in the background
  - Runs indefinitely, no UI, no interaction with user
  - E.g. media player
- Content Provider
  - Used to share data with other applications

Life Cycle of an Android Application

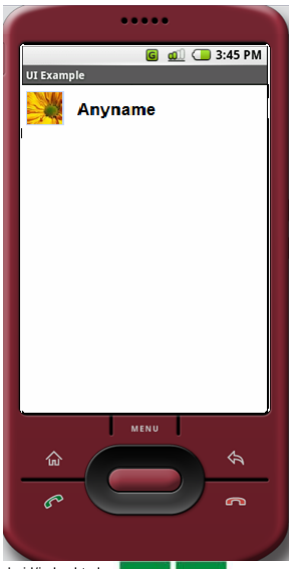
- Each Android application runs in its own Linux process
- Process's lifetime not directly controlled by application
- Determined by the system, depending on running applications, their importance, available memory
- Components (Activity, Service, Intent Receiver) impact the lifetime of the application's process
- Importance hierarchy for killing processes based on
  - Components running in them
  - The state of these components

Android's Importance Hierarchy

1. Foreground Process
   - Required for current user activities
   - E.g. running an Activity at the top of the screen
2. Visible Process
   - Activity is visible but not in the foreground (onPause())
   - E.g. previous activity displayed behind a foreground dialog
3. Service Process
   - Holds a Service, not directly visibleE.g. media player, network up/download
4. Background Process
   - Holds an Activity that is currently not visible (onStop())
   - Can be killed at any time to reclaim memory
5. Empty Process
   - Holds no active application components

Exercise 1

- Follow the Hello Android example
- Add a picture to the „Hello Android"-text
- Submit your solution using SVN
  - Create your personal folder „nachname" in the SVN-repository of your group
  - Create a folder for each exercise named „exerciseX" and put all necessary source files there
- **Submit your solution until Wednesday, 23.04.08, 12p.m.**



Source: http://code.google.com/android/index.html

Links

- Android website: http://code.google.com/android/
- YouTube: Androidology

Fragen?
Viel Spaß!