LMU
LUDWIG-
MAXIMILIANS-
UNIVERSITÄT
MÜNCHEN

FAKULTÄT FÜR MATHEMATIK, INFORMATIK UND STATISTIK
INSTITUT FÜR INFORMATIK
ARBEITSGRUPPEN MEDIENINFORMATIK UND
MENSCH-MASCHINE-INTERAKTION

Praktikum Entwicklung Mediensysteme (für Master)

Einführung in Android

Today

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

Schedule

- Phase 1 – Individual Phase:
  - Introduction to basics about Android
  - Exercises 1 to 3
  - Each student works on exercises himself/herself
  - Weekly meetings
- Phase 2 – Project Phase:
  - Concept and implementation of an Android application
  - Topic: Everyday Tools
  - Students work in teams
  - Regular milestone meetings
- Phase 3 – Evaluation:
  - Evaluate your concept
- Phase 4 – Paper Phase:
  - Write academic work
  - (Submit it to a relevant conference)

Topic: Everyday Tools

- Mobiler Datenzugang wird immer beliebter
- „Everyday Tools“
  - Mobile Software
  - Tägliche Einsatzmöglichkeit
  - Unterwegs
    - Einkaufen
    - Spazieren gehen
    - Shoppen
    - Auf dem Weg zur Arbeit
  - Vereinfachung bestimmter Alltagssituation

Timeline

Date	Topic/Activity
27.11.2010	Introduction and Overview of the Android Platform
03.11.2010	Implementing a User Interface
10.11.2010	Storing, Retrieving and Exposing Data
24.11.2010	Brainstorming, Application Design
01.12.2010	Project Phase Starts
	… (Milestones)
05.02.2010	Project Presentation
12.02.2010	Evaluation
	Paper Writing

Organizational Stuff I

- 4 SWS
- Weekly meetings
  - Wednesday 16:00 c.t. – 18:00
  - Room 107, Amalienstraße 17
  - Media lab available, if needed
- Homepage:
  - http://www.medien.ifi.lmu.de/pem

Organizational Stuff II

- Students work in teams
- SVN accounts for each team
  - svn://tracsvn.medien.ifi.lmu.de/repos/pem_team [number] (e.g. svn://tracsvn.medien.ifi.lmu.de/repos/pem_team1)
- Students check their exercises in with their group's SVN repository

Teams

- Team 1 (Diplom)
  - Obolashvili, Weber, Sappler
- Team 2
  - Dietz, Hartmann, Lindemann, Stockinger
- Team 3
  - Bornschlegel, Bürger, Grabs, Koelle

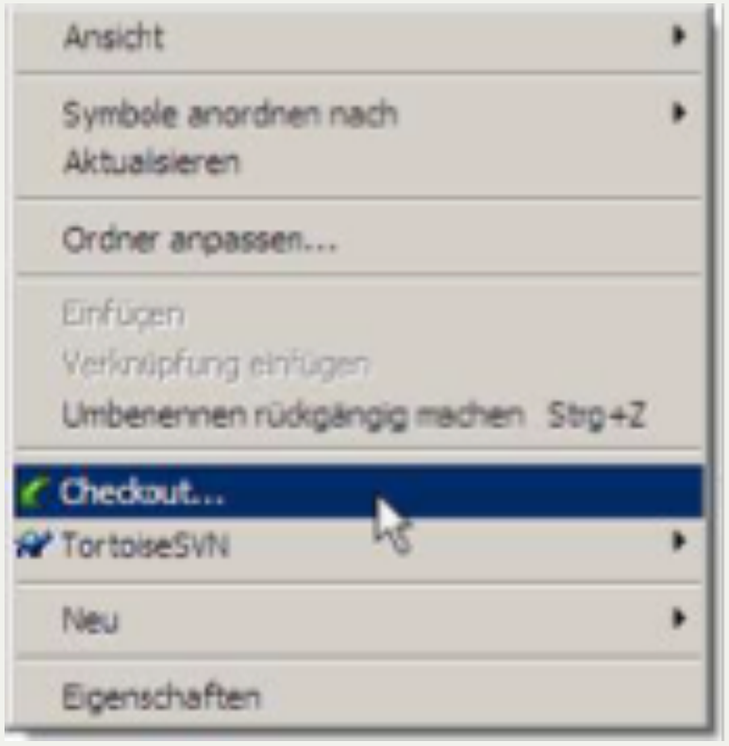
Technology – SVN

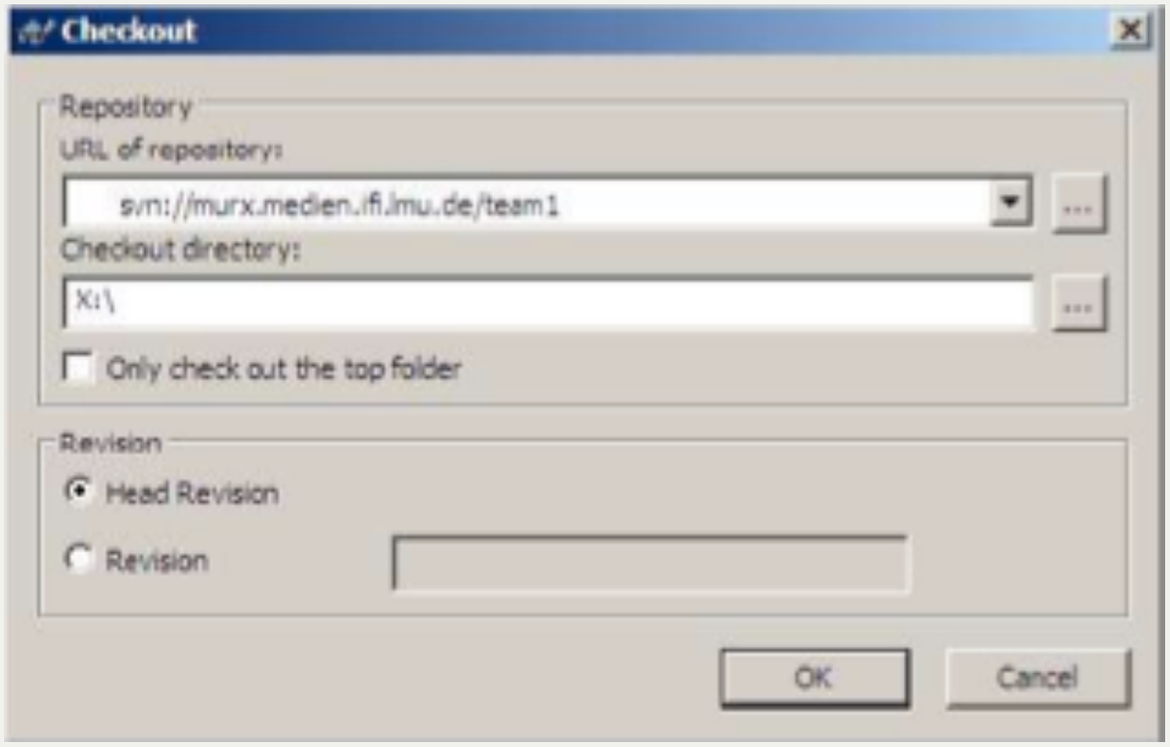
Technology – SVN I

- 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. And do not touch the hidden .svn-Folders, especially do not copy an svn-folder (use Export-Command).

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

- 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

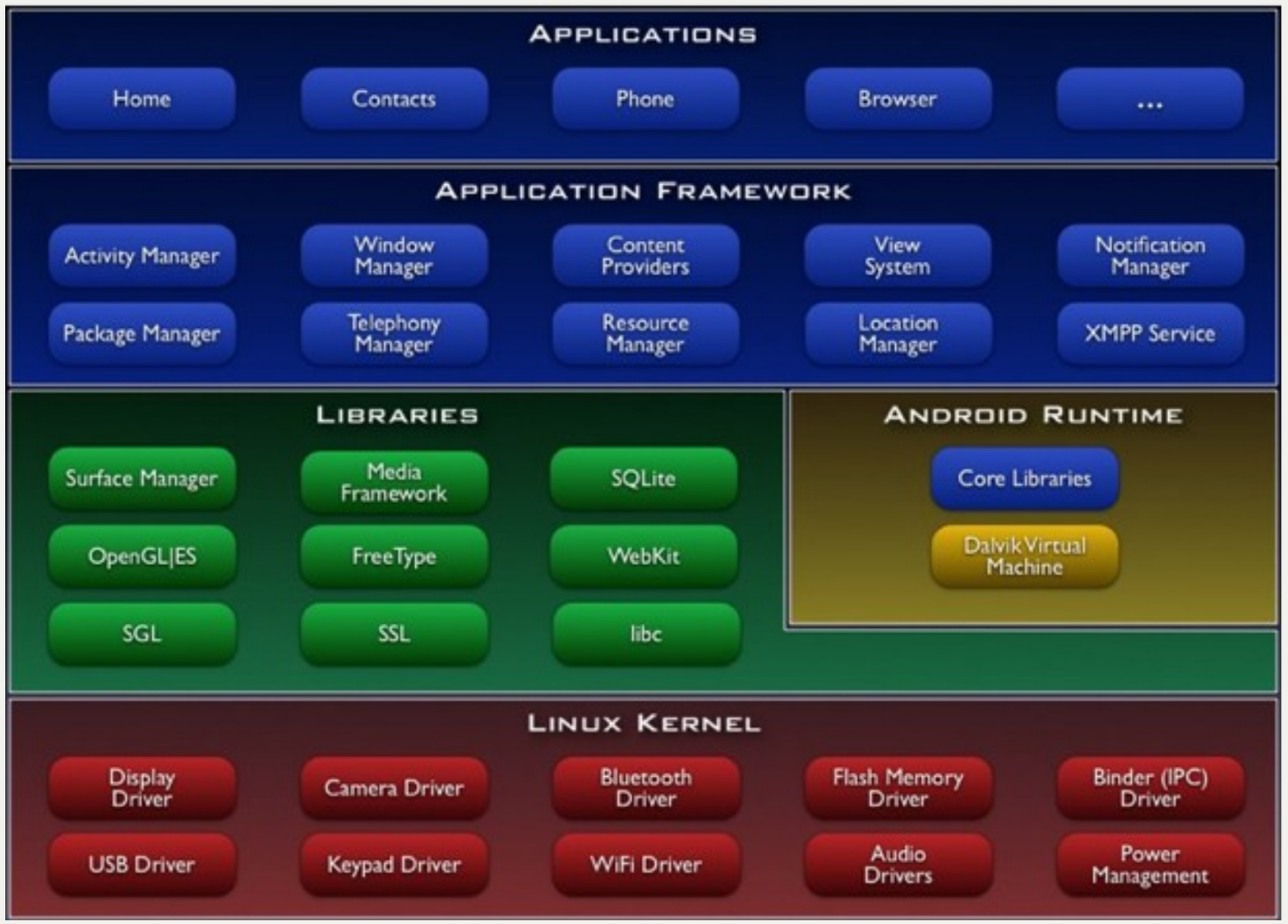
Source: www.openhandsetalliance.com/

Android Features

- **Application framework** enabling reuse and replacement of components
- **Dalvik virtual machine** optimized for mobile devices (register based)
- **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

- 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.

Android Version History

Version	Features
1.5 Cupcake	30.04.2009: Onscreen-Keyboard with „Autocomplete“, Screen switch Animations, Video upload
1.6 Donut	15.09.2009: Screenshots on the android market, Voice Search, WVGA resolutions,
2.0/2.1 Eclair	12.01.2010: Speed improvements, More screen resolutions (dip), Camera flash support, Live wallpapers, Multitouch support
2.2 FroYo	20.05.2010: Speed and performance increase, Flash 10.1 support, Installing apps on SD-Card
Future Versions	2.3 Gingerbread (new copy and paste), 3.0 Honeycomb (tablet optimized), 4.0 Ice Cream

Installing 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 and Download your Platforms

http://developer.android.com/sdk/index.html

Installing SDK

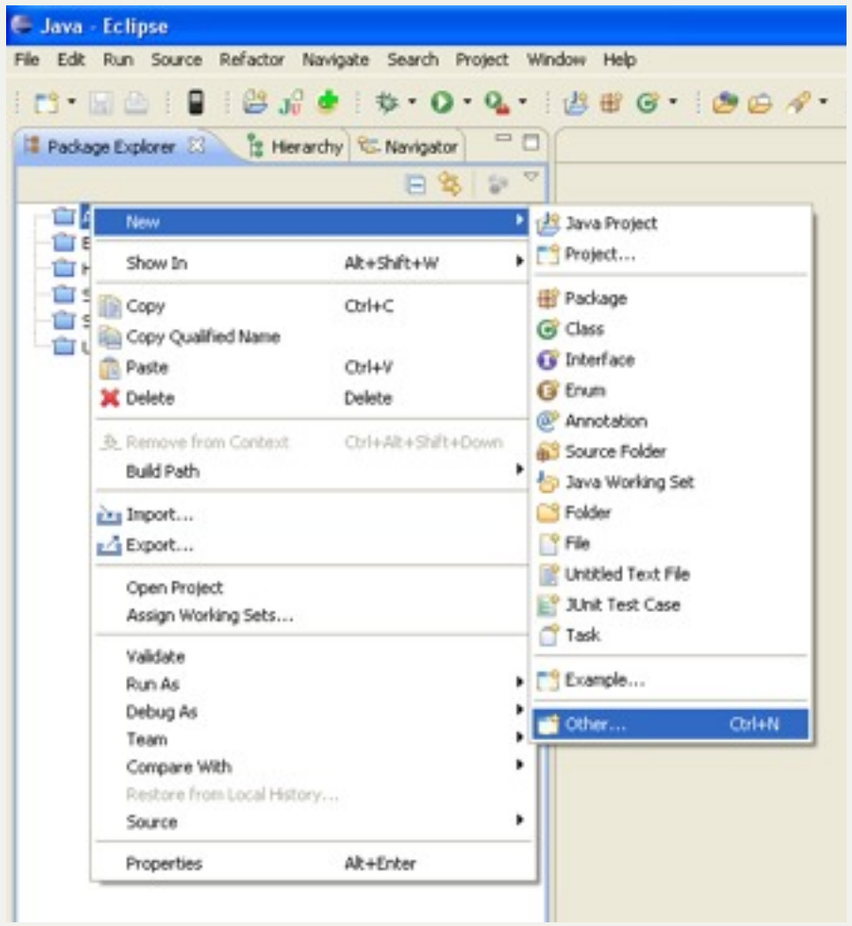
- Create an Android project
  - Standard Eclipse procedure
  - Automatically creates folders and a Manifest file
  - Can also be used to create a demo project
- Set up a launch configuration
  - Run application from menu or
  - Define settings for run configuration (project, activity, emulator options, …) from Run > Open Run Dialog >
- Run Android application in emulator
  - Be Patient! The emulator takes while to boot up.
  - Keep it open once it was started!

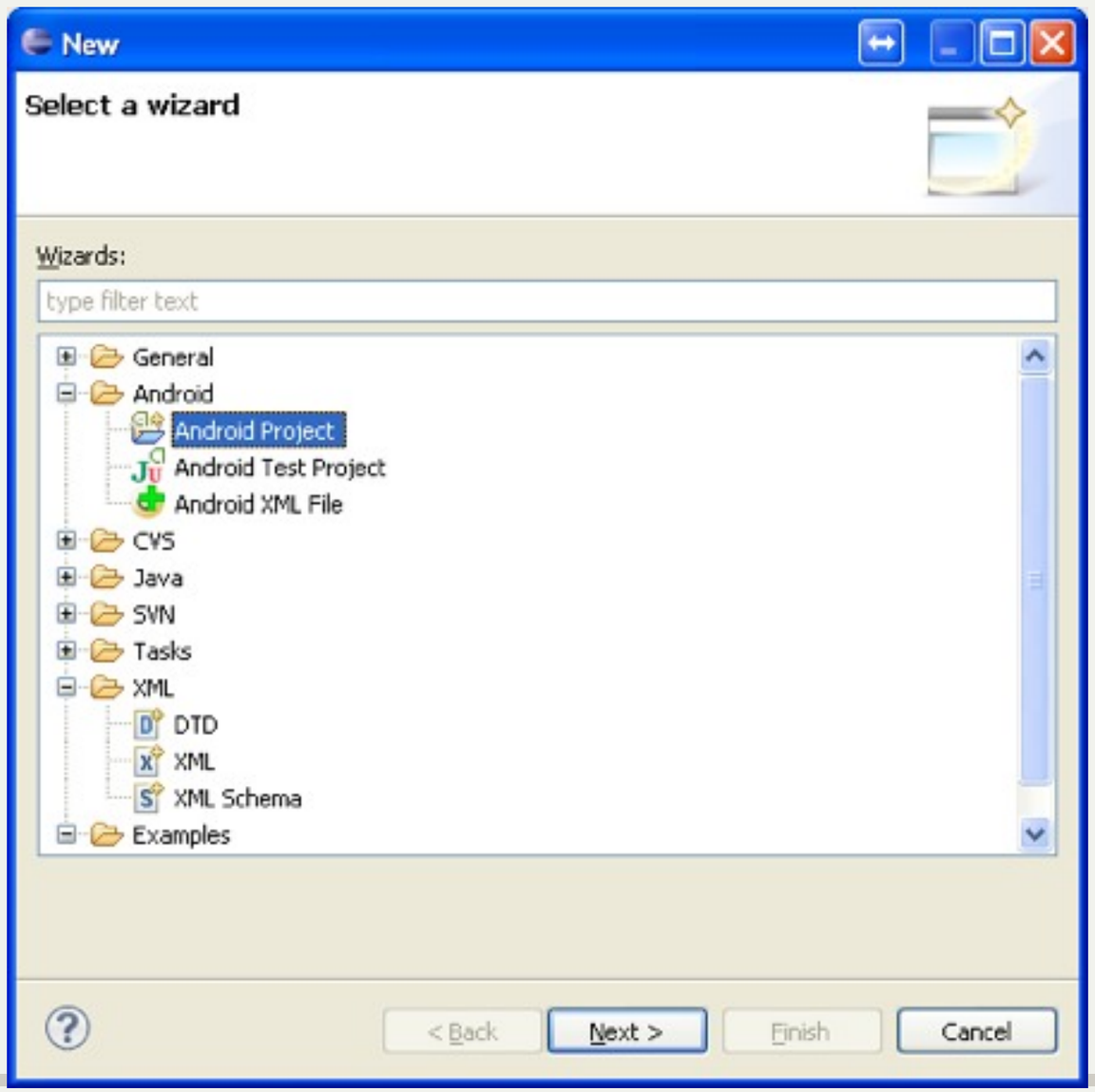
The Nexus One

nexus one™

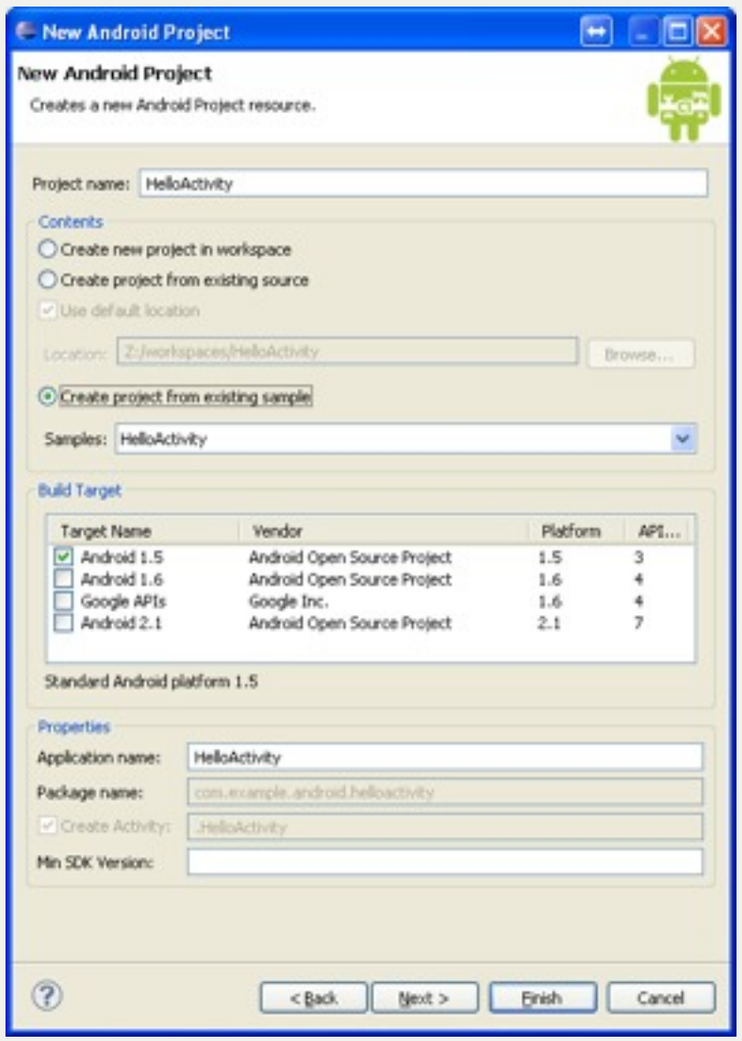
Source: Wikimedia Commons

Hello Android I





Hello Android II



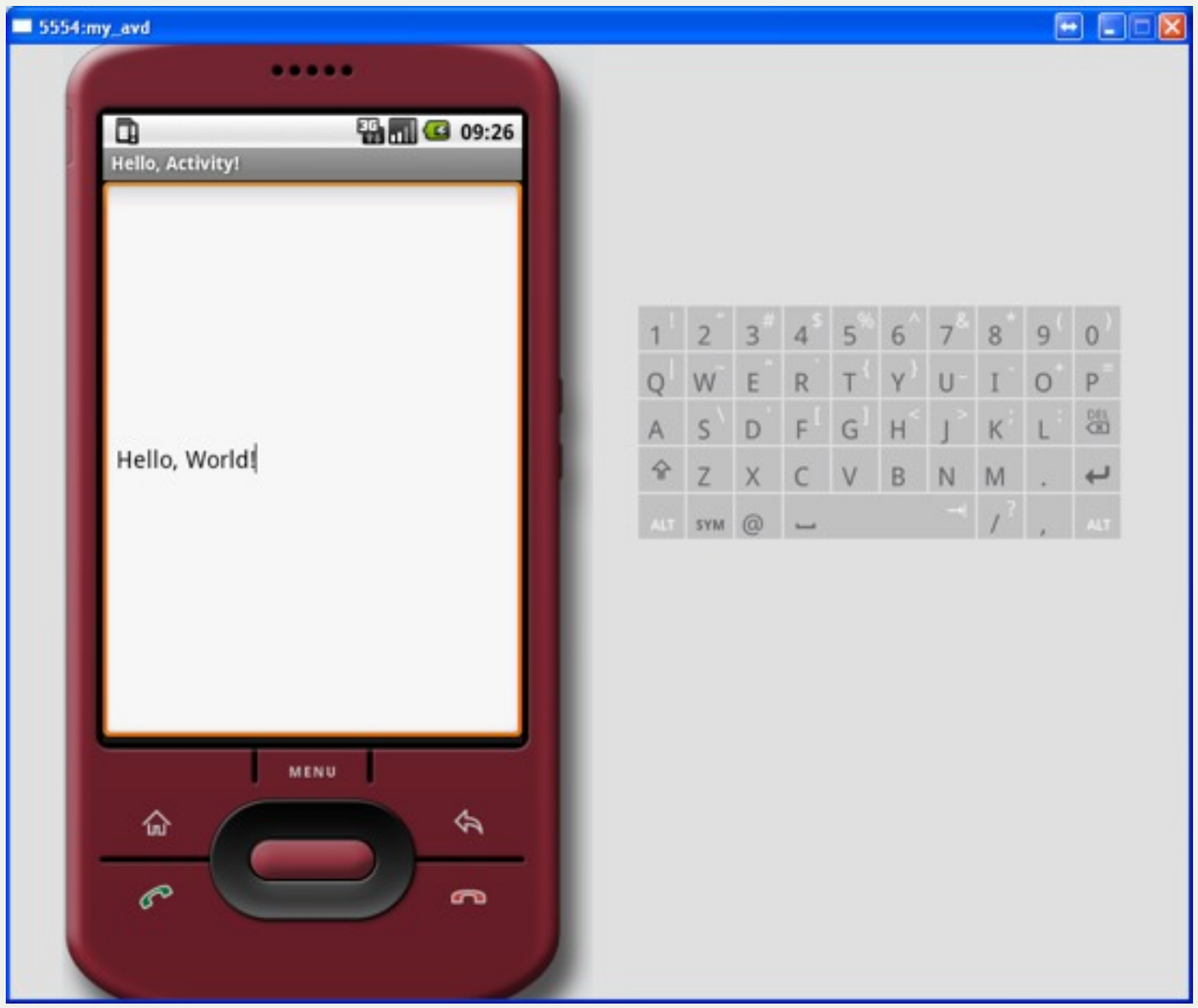
Source: http://code.google.com/android/index.html

Hello Android III

```
 * Copyright (C) 2007 The Android Open Source Project

package com.example.android.helloactivity;

import android.app.Activity;


/**
 * A minimal "Hello, World!" application.
 */
public class HelloActivity extends Activity {
    public HelloActivity() {
    }

    /**
     * Called with the activity is first created.
     */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        // Set the layout for this activity.  You can find it
        // in res/layout/hello_activity.xml
        setContentView(R.layout.hello_activity);
    }
}
```

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

Hello Android IV



Anatomy of an Android

- 4 main building blocks for Android applications
  - Activity
  - Intent Receiver
  - Service
  - Content Provider

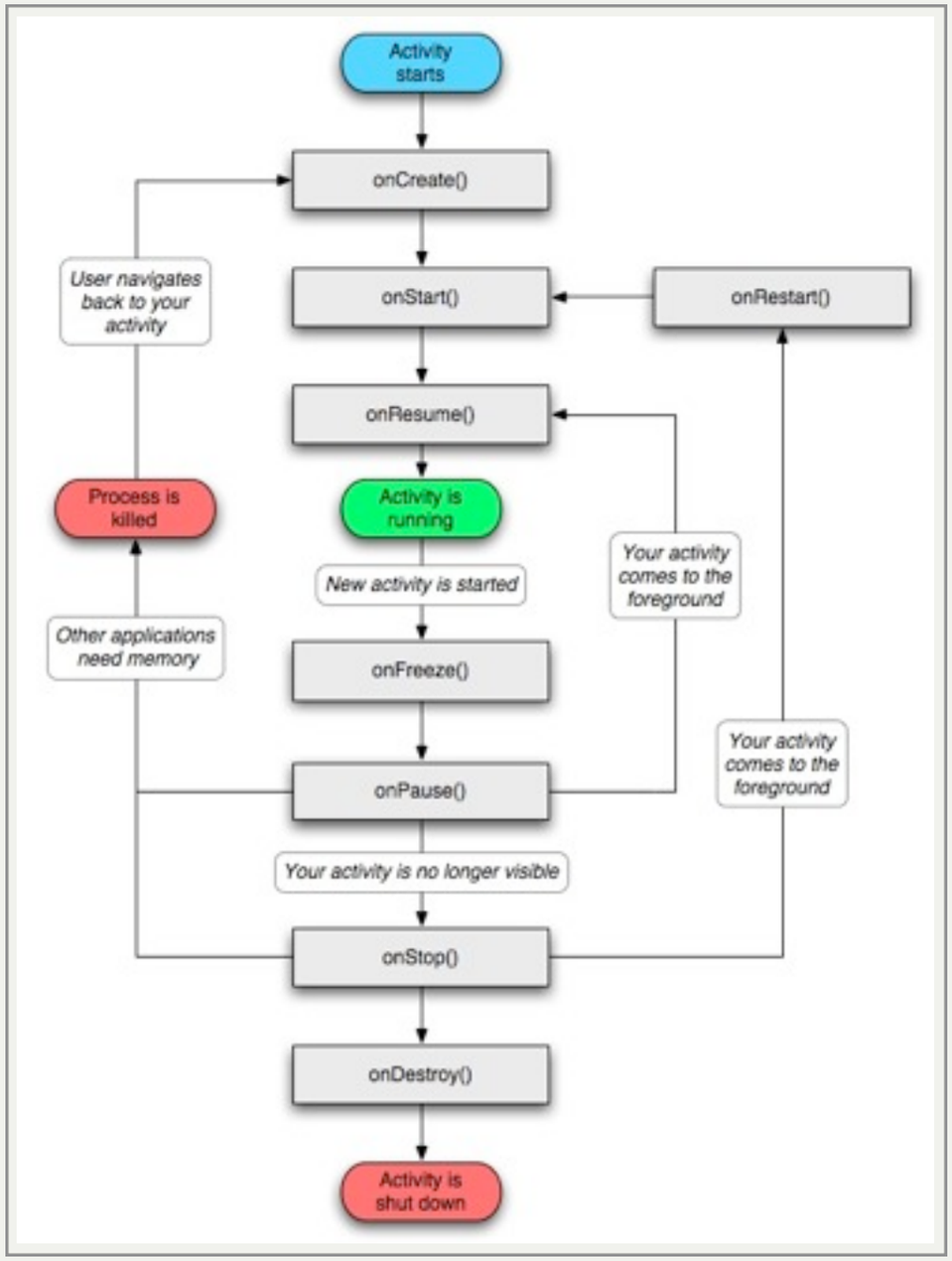
```
<?xml version="1.0" encoding="utf-8"?>

<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>
```

- AndroidManifest.xml lists all components of an application, their capabilities and requirements

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

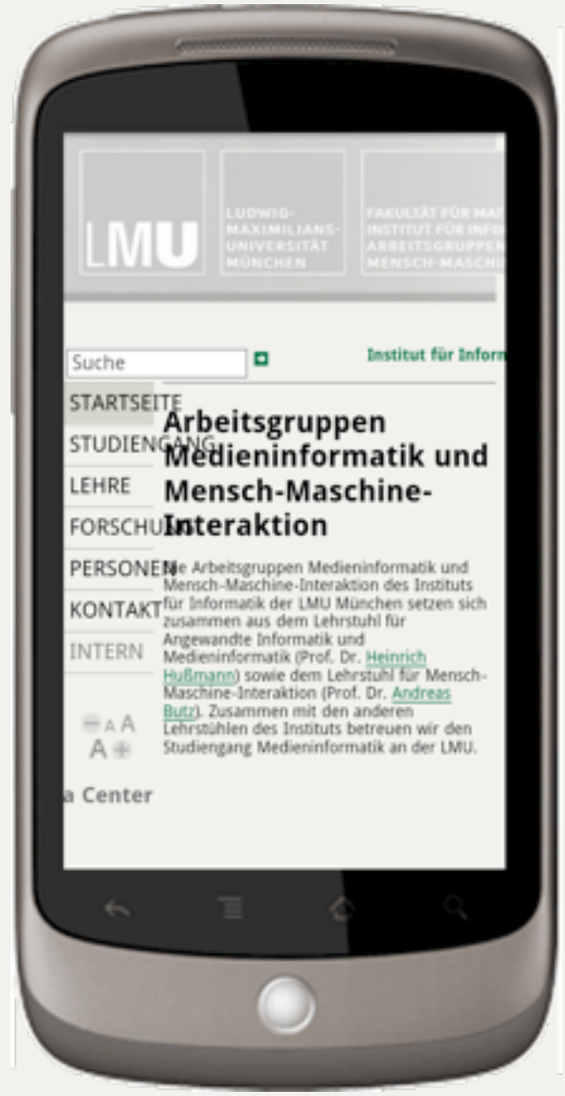
- Each Android application runs in its own Linux process
- Process's lifetime not directly controlled by application
- Determined by the system, depending on running applications, their importance, available memory
- Components (Activity, Service, Intent Receiver) impact the lifetime of the application's 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

- Create a first Andoird Activity
- It should incorporate a mini web browser
- Always load URL: www.medien.ifi.lmu.de
- Make it fullscreen!
  - 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
- **Solutions are always due to following Wednesday, 12 p.m.**



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

Links

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

http://www.youtube.com/watch?v=QBGfUs9mQYY

Fragen?

Viel Spaß!