

Einführung in die Entwicklung mobiler Anwendungen



1. Einführung in die Entwicklung mobiler Anwendungen
2. Erste grafische Oberflächen und Benutzerinteraktionen
3. Weiterführende Konzepte mobiler Plattformen
4. Standortbezogene Dienste, Sensoren und Kamera
5. Dauerhaftes Speichern von Daten (Persistenz)
6. Responsive Design, Weiterführende Interaktionsmuster
7. Asynchrone Verarbeitung



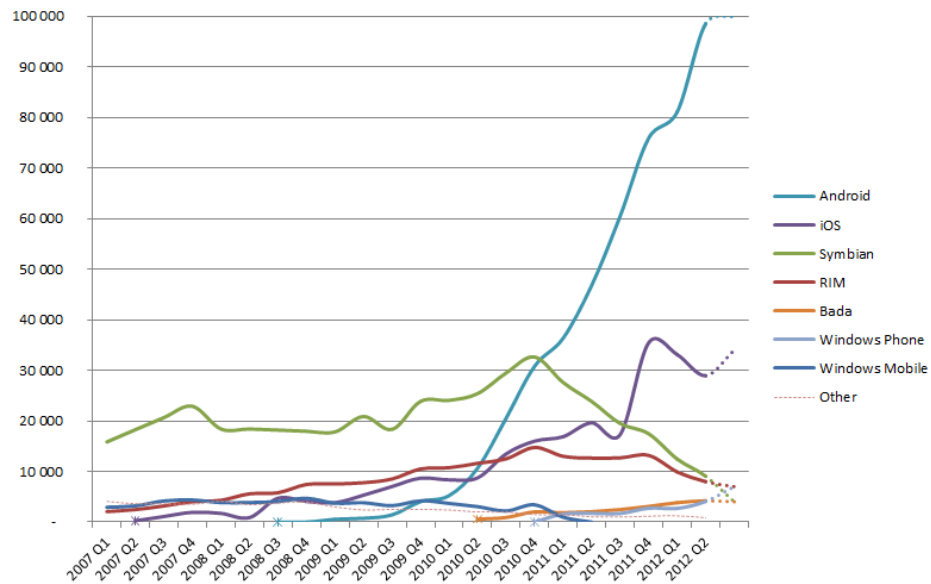
- Eingeschränkte Konnektivität
 - geringe Bandbreite
 - hohe Latenz
- Leistungsschranken
 - Rechenleistung
 - Arbeitsspeicher
- Endliche Energiequelle (Akkulaufzeit)
- Formfaktor
 - Displaygröße
 - Touchscreens, ohne Tastatur
 - neue Eingabemöglichkeiten (zB Kamera, Gesten, Sensoren, Geographische Position, ...)



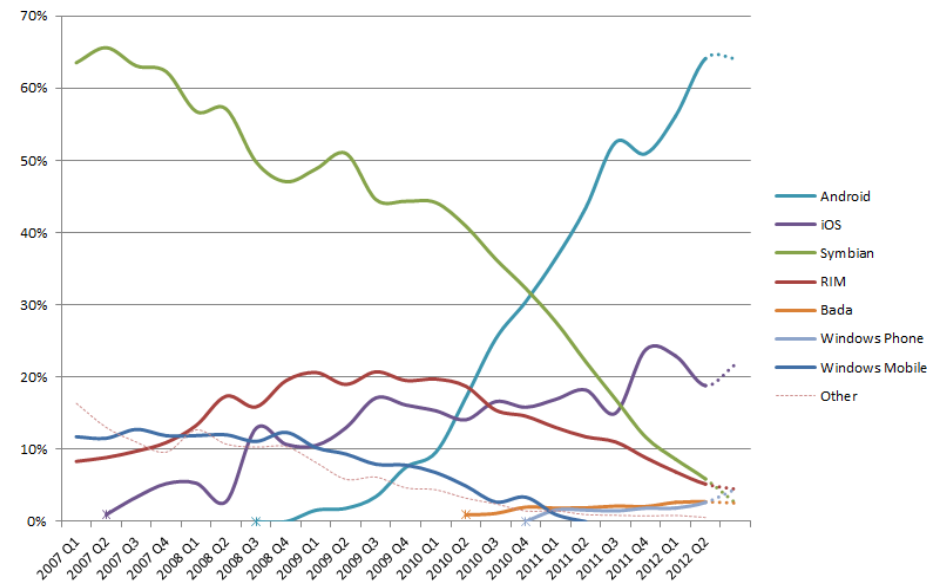
Smartphone Verkaufstrends 2007 - 2012

4

World-Wide Smartphone Sales (Thousands of Units)



World-Wide Smartphone Sales (%)



Quelle: Gartner Smartphone Market Share 2007 Q1 – 2012 Q2

Einführung in die Entwicklung mobiler Anwendungen
Studiengang Web-Business & Technology, WS 2014/15

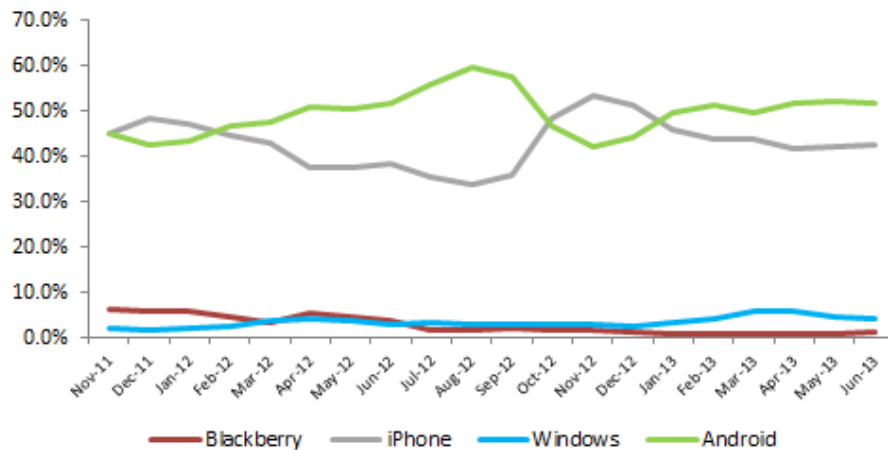


Smartphone Verkaufstrends 2011 – 2013

Unterschied Europa und USA

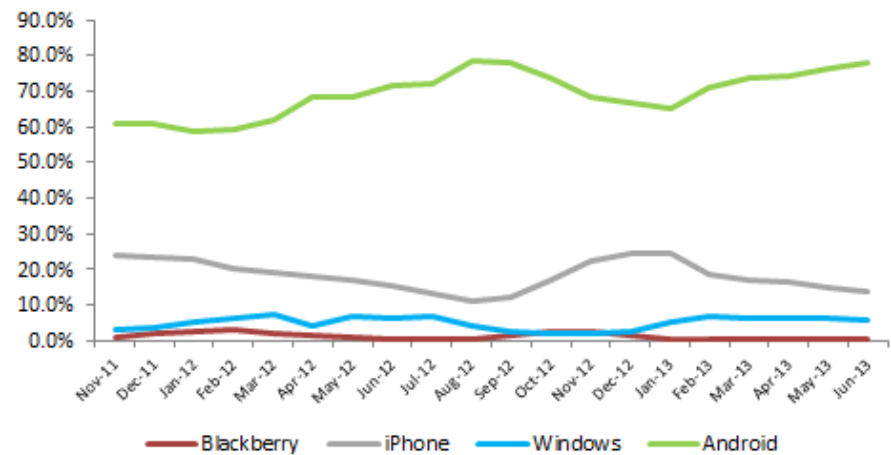
5

US Smartphone Market Share - Kantar



Tech-Thoughts ©

Germany Smartphone Market Share - Kantar

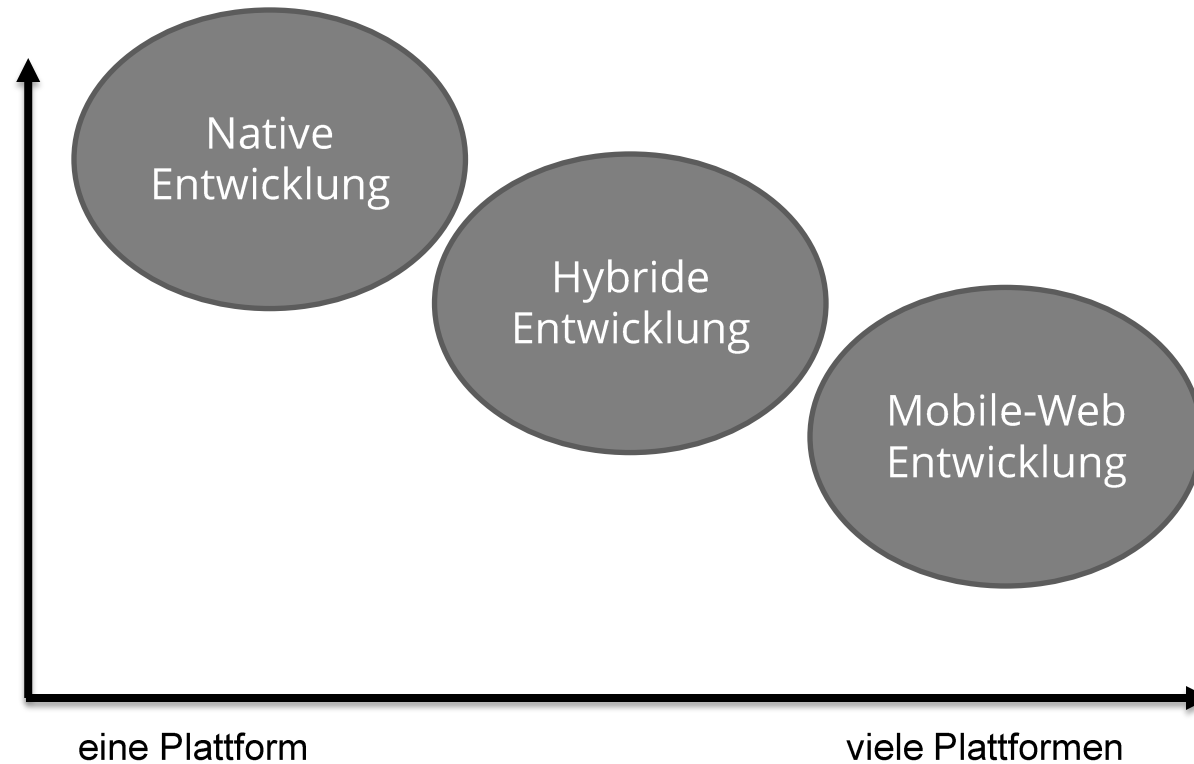


Tech-Thoughts ©

Quelle: Tech-Thoughts, Daten Gartner

Einführung in die Entwicklung mobiler Anwendungen
Studiengang Web-Business & Technology, WS 2014/15





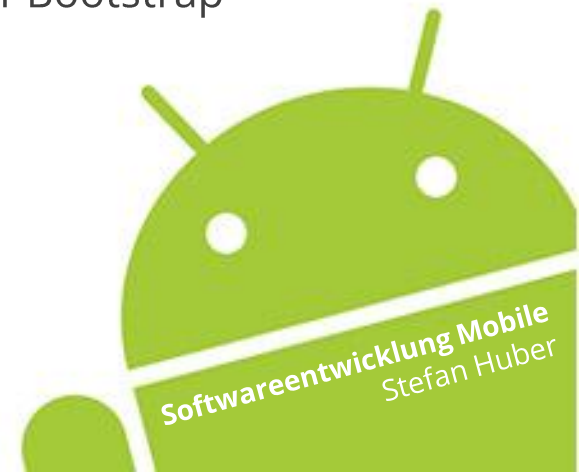
- Möglichkeiten
 - Kapseln die Komplexität des darunterliegenden Betriebssystems und der Spezifika der Hardware (z.B. unterschiedliche Prozessorarchitekturen)
 - Gute Dokumentation, viele Ressourcen und Hilfen (zB MOOCs)
 - Bieten spezielle Entwicklungsumgebungen (zB Android Studio)
- Grenzen
 - Direkter Zugriff auf die Hardware ist beschränkt durch die Möglichkeiten, die das Framework bietet.
 - Einfache Veröffentlichung der App in diversen Stores mit hoher Gewinnaussicht.
 - Verbreitung der Anwendung hängt direkt von der Verbreitung des Frameworks und darauf aufbauender Geräte ab.
- Typische Vertreter:
 - Android SDK, Apple iOS, Tizen, ...



- Möglichkeiten
 - Basieren meist auf bereits bekannte Webstandards (html, css)
 - Schneller Know-How Aufbau für Web-Entwickler
 - Gut geeignet um contentzentrierte Applikationen zu realisieren, die trotzdem die Interaktionen mobiler Geräte nutzen sollen
- Grenzen
 - Direkter Zugriff auf die Hardware oft eingeschränkt
 - Geräteoptimierungen sind nicht möglich.
 - Entwicklungsprozess besteht oft in der Generierung des fertigen Pakets über eine proprietären Build-Prozess (zB iOS Cordova App muss auf MAC kompiliert werden)
 - Performanznachteile gegenüber nativen Frameworks
- Typische Vertreter:
 - Apache Cordova (Phonegap), Appcelerator Titanium, Xamarin



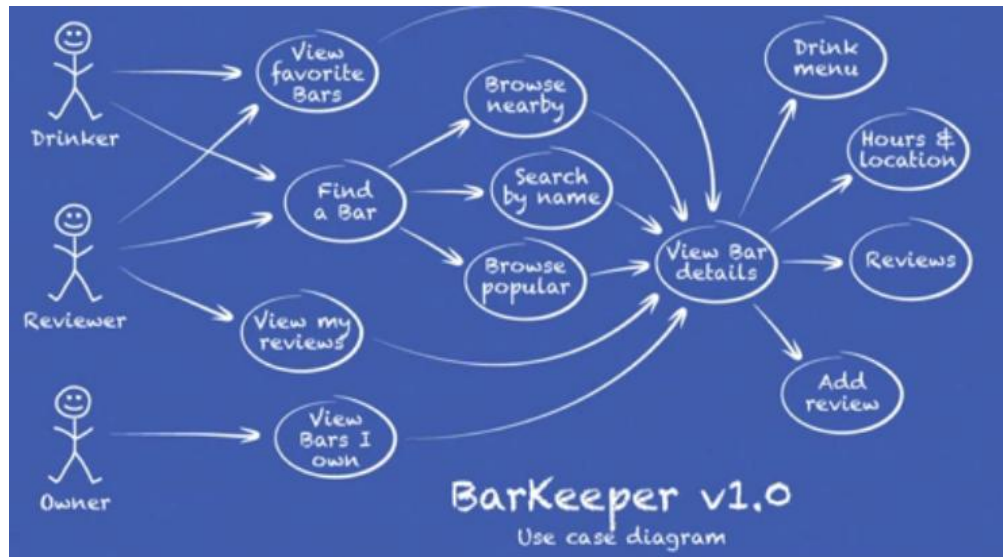
- Möglichkeiten
 - Entwicklung von klassischen Webanwendungen, die auf die Eigenheiten mobiler Geräte (Browser) Rücksicht nehmen
 - Keine spezifische Entwicklung einer Anwendung für eine oder mehrere mobilen Plattformen mehr nötig
 - Am besten für die variable Präsentation von Inhalten geeignet
 - Kein spezifisches Know-How für mobile Plattformen nötig
- Grenzen
 - Viele Einschränkungen in Bezug auf die Nutzung der Hardware (zB Near Field Communication NFC kann nicht genutzt werden)
- Typische Vertreter:
 - Responsive Webdesign mit Media Queries, zB Twitter Bootstrap



App Design und Planung

10

- Use-Case Diagramme, eignen sich hervorragend für App Design und Planung
- Aufgrund des Formfaktors, ist meistens jede Aktion mit einem Screen darstellbar



- Video: <https://www.youtube.com/watch?v=XpqyiBR0lJ4>



Prototyping on Paper

11



ANDROID

Grundlagen der Android Plattform



Warum Android?

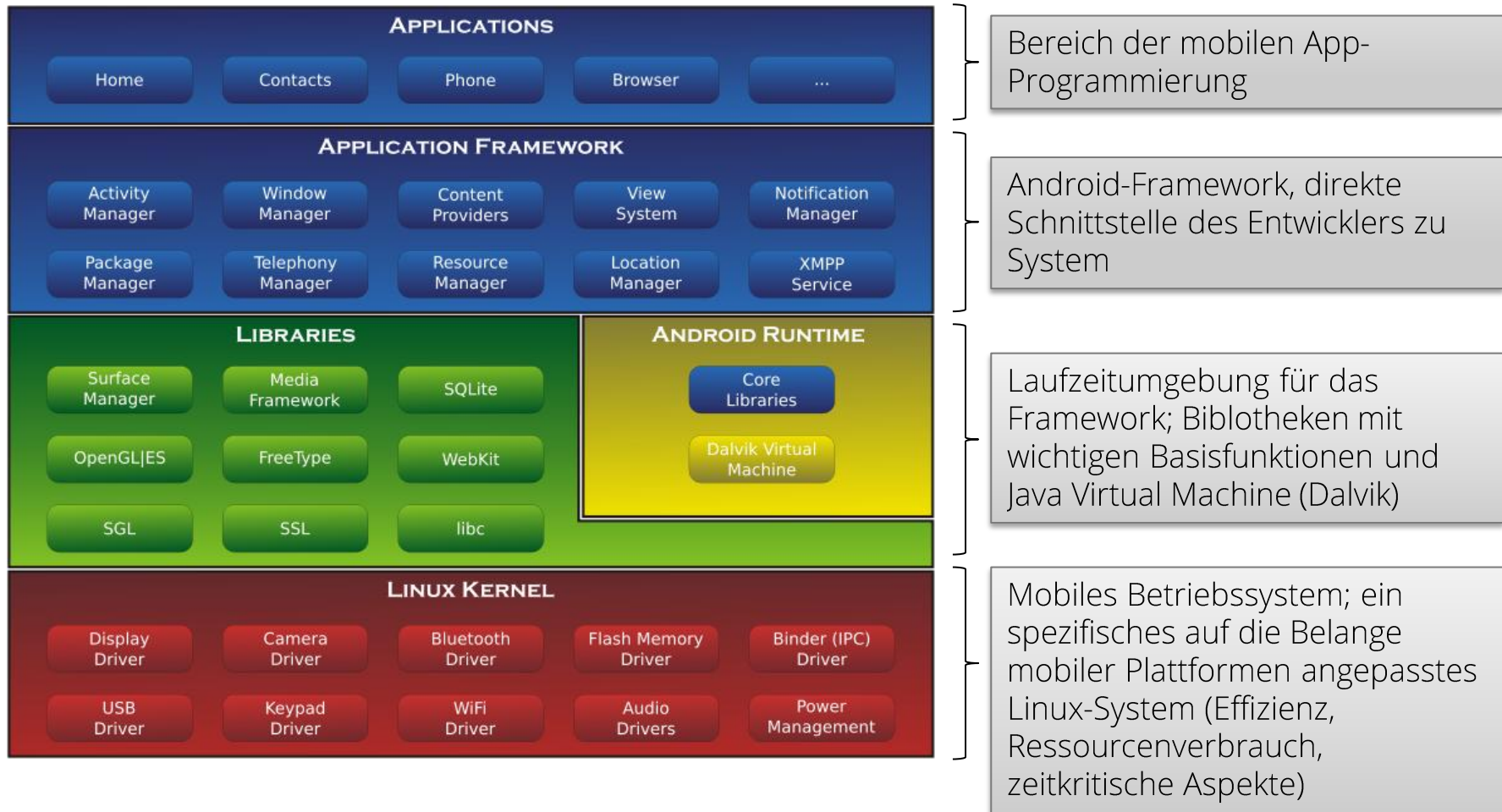
13

- Unterstützung vieler unterschiedlicher Geräteklassen
 - Smartphones & Tablets
 - Android Wear
 - Android TV
 - Android Auto
 - Google Glass
 - Spielkonsolen
 - Ouya
 - Project Shield
 - und vieles mehr...
- Offene Plattform
 - Android ist größtenteils unter der Apache Software License 2.0 veröffentlicht
 - Der Sourcecode kann unter source.android.com heruntergeladen werden



Android Plattform Architektur

14



Android API Levels

15

- Ein API Level klassifiziert die verfügbaren Funktionalitäten des Android Frameworks eindeutig.
- Erweiterungen der Framework API sind additiv (Abwärtskompatibilität)
- Minimal, Maximal und Ziel API Level können für die jeweilige Anwendung festgelegt werden (Manifest)
- Für eine breite (virale) Verbreitung der Anwendung sollte das Minimale API Level **so niedrig wie möglich** sein!

API LEVEL	Bezeichnung
19	KitKat (4.4)
16, 17, 18	Jelly Bean (4.1 – 4.3)
15, 14	Ice Cream Sandwich (4.0)
13, 12, 11	Honeycomb (3.0 – 3.2)
10, 9	Gingerbread (2.3)
8	Froyo (2.2)
7,6,5	Eclair (2.0 – 2.2)
4	Donut (1.6)
3	Cupcake (1.5)
2, 1	Base (1.0 – 1.1)

Für die Highlights des jeweiligen API Level sollte <http://developer.android.com> konsultiert werden!

Grundstruktur einer Android Anwendung

16

- Android Anwendungen bestehen aus einer Reihe von Komponenten, die durch ein sog. Manifest (xml-Datei) konfiguriert werden
- Zu einer Android App können folgende Komponenten gehören:
 - Anwendungskomponenten
 - Activities
 - Services
 - Content Provider
 - Broadcast Receivers
 - Ressourcen
 - Mediendateien (Bilder, Soundfiles, Videos, ...)
 - Vordefinierte Zeichenketten
 - Layout Definitionen
 - Externe Libraries



- Activity
 - Eine Activity ist Verantwortliche für die Darstellung des User Interfaces und für die Verarbeitung von Benutzerinteraktionen.
 - Bsp: In einer Email Anwendung gibt es eine Activity für das Verfassen einer Email, dafür werden Views (Textfelder, Buttons, etc.) bereitgestellt und Interaktionen (Klick auf Button) verarbeitet.
- Service
 - Services verarbeiten Operationen im Hintergrund einer App.
 - Bsp: Mediaplayer (Service) spielt Musik im Hintergrund einer App.
- Content Provider
 - Content Provider verwalten Anwendungsdaten, auf welche über Content Resolver zugegriffen werden kann.
 - Bsp: SQLite Datenbank ermöglicht das Abspeichern Strukturierter Daten.
- Broadcast Receiver
 - Broadcast Receiver reagieren auf einen systemweiten Broadcast.
 - Bsp: Anwendungen können auf einen „Low Battery“ Broadcast reagieren.



Grundstruktur einer Android Anwendung

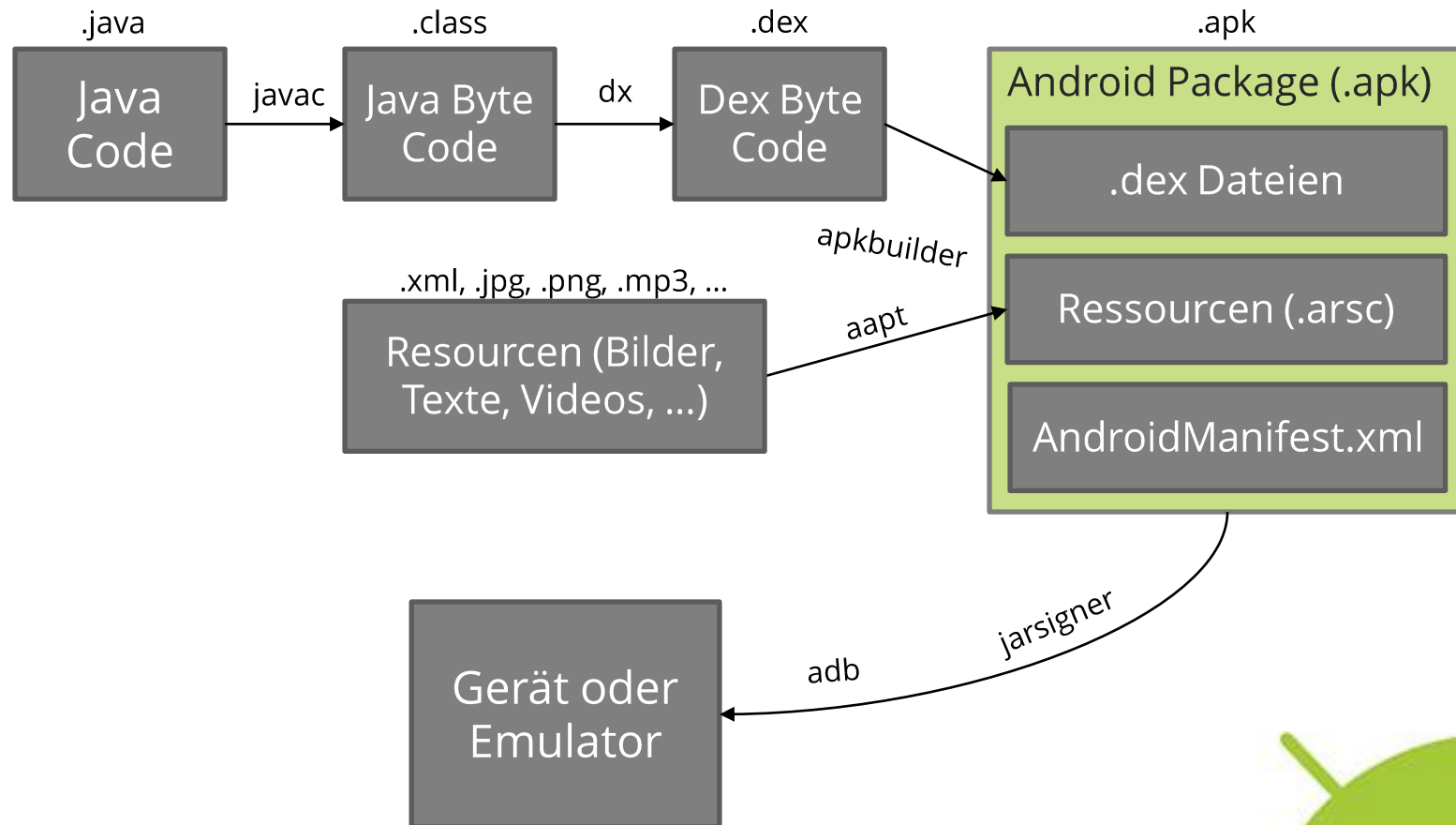
18

- Android Anwendungen bestehen aus einer Reihe von Komponenten, die durch ein sog. Manifest (xml-Datei) konfiguriert werden
- Im AndroidManifest.xml stehen:
 - Alle Komponenten, die zur App gehören (siehe vorherige Folie)
 - Beschreibt alle Möglichkeiten der Interaktion dieser Komponenten untereinander und zu anderen Apps
 - Enthält Metadaten (z.B. Versionsinformationen) und beschreibt den Link zu anderen Ressourcen (z.B. zu Icons)
 - Definiert, welche (Hardware)Ressourcen von der App benötigt werden (z.B. Berechtigungen für Internetzugriff oder Zugriff auf die Kamera)



Grundstruktur einer Android Anwendung

19



Ausführung von Android Anwendungen

20

- Android Anwendungen werden innerhalb einer “Sandbox” ausgeführt:
 - Eigener User je Anwendung
 - Ausführung in einem eigenen Prozess
 - Eigene Dalvik Virtual Machine
 - Eigener Bereich im Dateisystem
 - Eigener Bereich im Hauptspeicher (Heap)



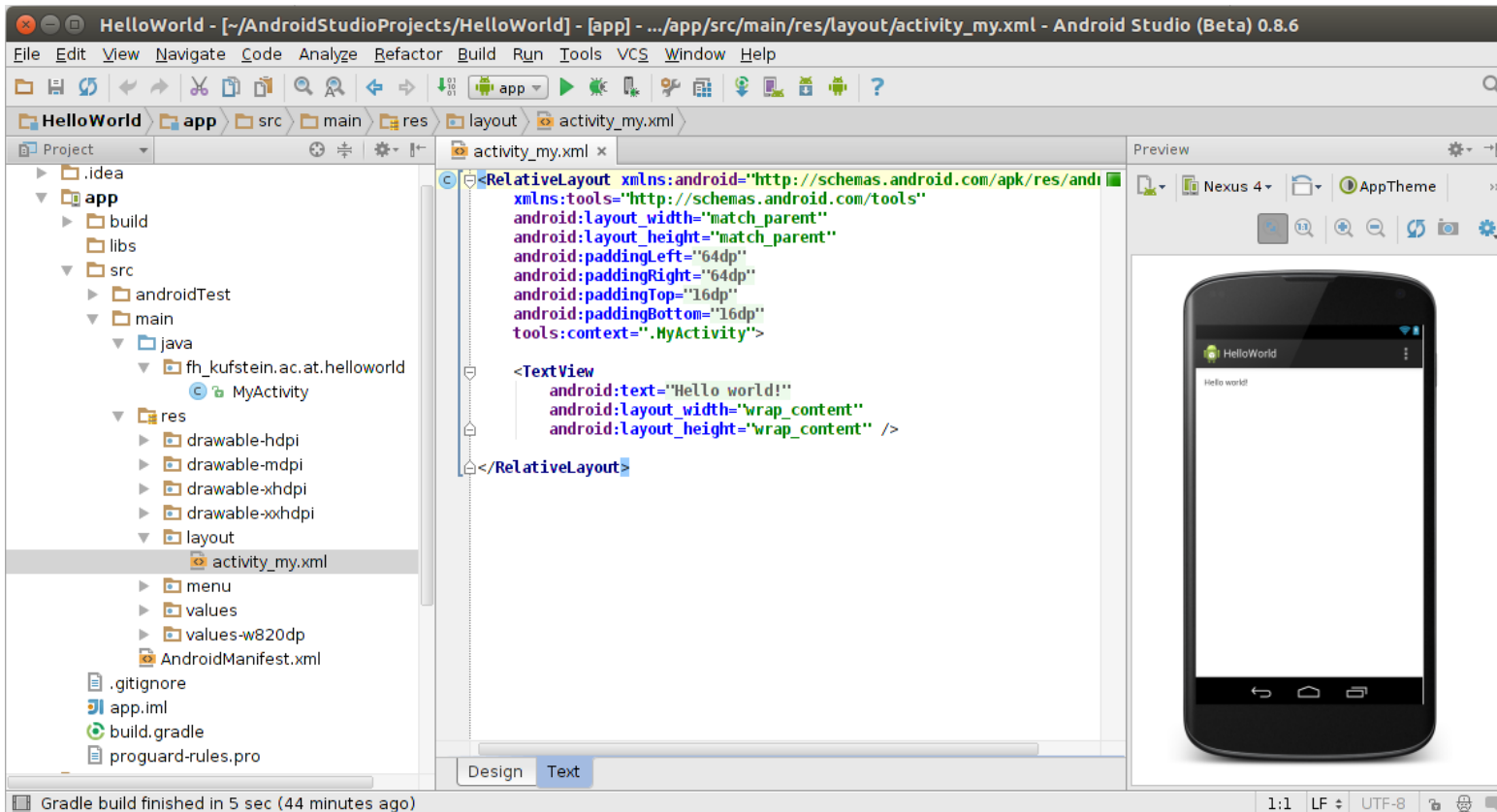
- Die Entwicklung unter Android setzt mehrere Komponenten voraus:
 - Das Android SDK (Software Development Kit), das über den SDK Manager verwaltet wird.
 - Eine Verbindung zu den (virtuellen) Android Geräten, auf denen die programmierte Anwendung laufen soll. Diese Geräte werden über den Android Virtual Device Manager (AVD-Manager) verwaltet.
 - Eine Entwicklungsumgebung die den spezifischen Entwicklungsprozess und die vorgenannten Komponenten integriert. Mögliche IDEs:
 - Netbeans, mit Android Plugin
 - Eclipse
 - **Android Studio**



Android Entwicklungstools

22

Android Studio



Download unter <https://developer.android.com/sdk/installing/studio.html>

Android Entwicklungstools

23

DDMS

DDMS - Eclipse - /Users/robertly/Documents/workspace

Devices

Name	Online	Android-2.1 [2.1
emulator-5556	Online	Android-2.1 [2.1
system_process	64	8600
com.android.inputmethod	116	8601
com.android.phone	118	8602
android.process.acore	142	8604
com.android.alarmclock	164	8603
android.process.media	177	8605
com.android.email	200	8606 / 8700
com.android.mms	212	8607

Emulator Control

Telephony Status

Voice: home Speed: Full

Data: home Latency: None

Telephony Actions

Incoming number:

Threads

ID	Tid	Status	utime	stime	Name
3	200	wait	26	20	main
*5	201	vmwait	2	7	HeapWorker
*7	202	vmwait	0	0	Signal Catcher
*9	203	running	3	7	JDWP
11	204	native	0	0	Binder Thread #1
13	205	native	0	0	Binder Thread #2
15	207	wait	0	0	Thread-8

LogCat

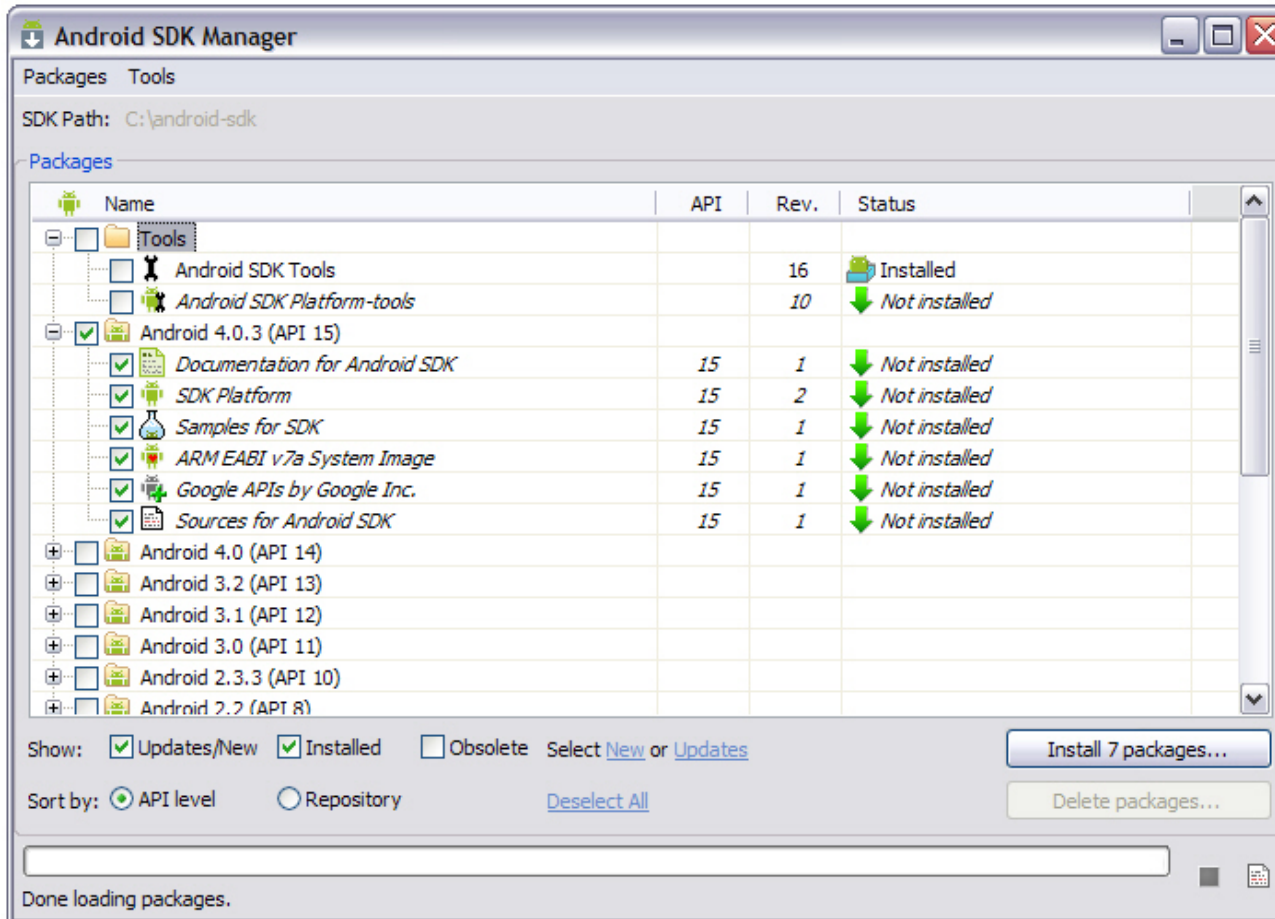
Time	pid	tag	Message
09-07 11:32:47. D 200		EAS Synct	!!! EAS Synctmanager, onDestroy
09-07 11:32:47. I 64		ActivitytStart	proc com.android.mms for broadcast
09-07 11:32:47. D 177		MediaScarstart	scanning volume internal
09-07 11:32:47. D 212		ddm-heap	Got feature list request
09-07 11:32:47. D 31		installD	DexInv: --- BEGIN '/system/app/fms.apk
09-07 11:32:47. I 177		MediaProvUpgrading	media database from version
09-07 11:32:49. D 219		dalvikvm	DexOpt: load 360ms, verify 1098ms, opt



Android Entwicklungstools

24

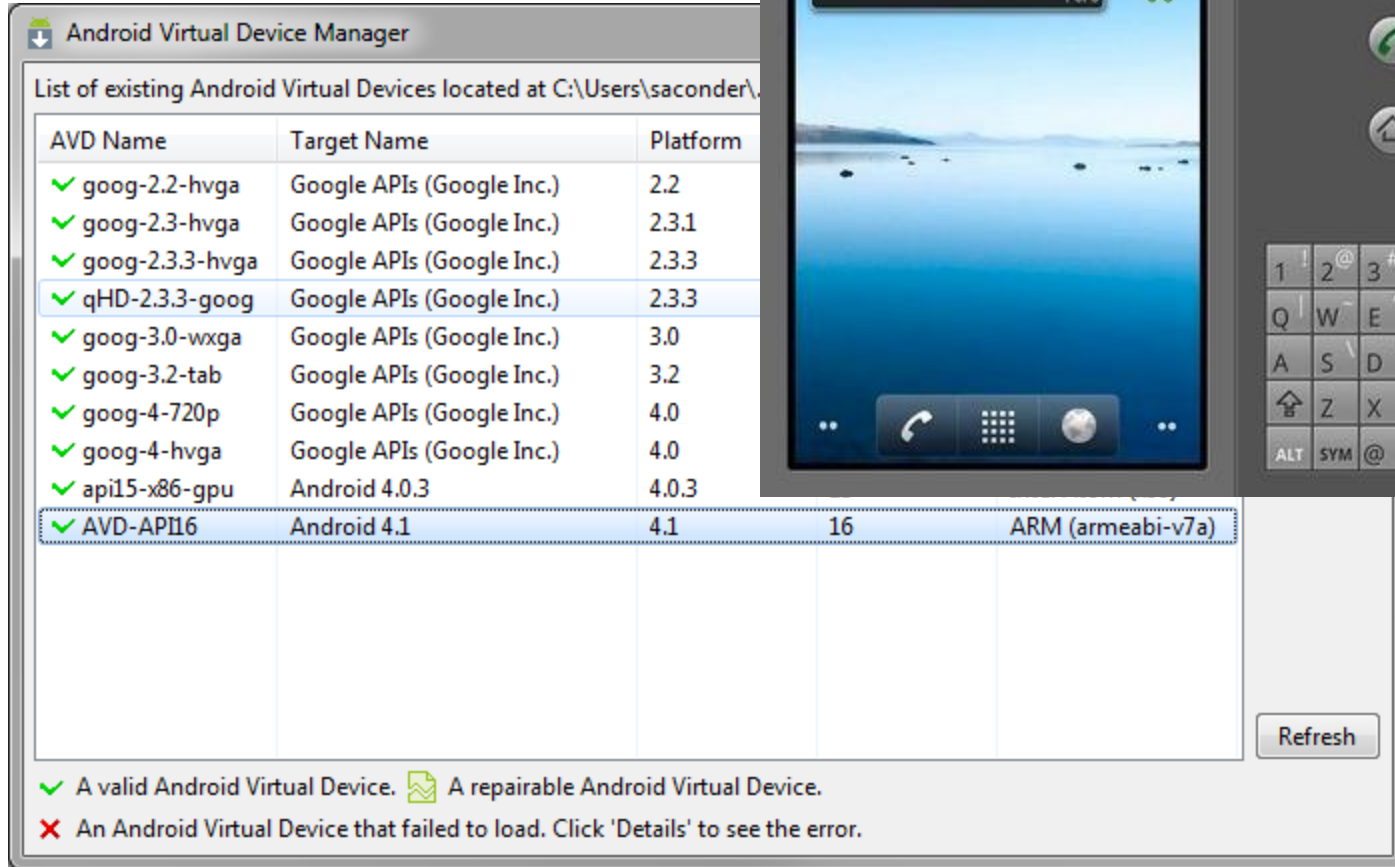
Android SDK Manager



Android Entwicklungstools

25

AVD und Emulator




The image shows the Android Virtual Device Manager (AVD Manager) window on the left and the Android emulator interface on the right. The AVD Manager lists existing virtual devices, with 'AVD-API16' selected. The emulator shows a virtual Android phone screen with a notification and a home screen with a dock.

Android Virtual Device Manager

List of existing Android Virtual Devices located at C:\Users\saconder\...

AVD Name	Target Name	Platform
✓ goog-2.2-hvga	Google APIs (Google Inc.)	2.2
✓ goog-2.3-hvga	Google APIs (Google Inc.)	2.3.1
✓ goog-2.3.3-hvga	Google APIs (Google Inc.)	2.3.3
✓ qHD-2.3.3-goog	Google APIs (Google Inc.)	2.3.3
✓ goog-3.0-wxga	Google APIs (Google Inc.)	3.0
✓ goog-3.2-tab	Google APIs (Google Inc.)	3.2
✓ goog-4-720p	Google APIs (Google Inc.)	4.0
✓ goog-4-hvga	Google APIs (Google Inc.)	4.0
✓ api15-x86-gpu	Android 4.0.3	4.0.3
✓ AVD-API16	Android 4.1	4.1 16 ARM (armeabi-v7a)

Refresh

✓ A valid Android Virtual Device.  A repairable Android Virtual Device.
✗ An Android Virtual Device that failed to load. Click 'Details' to see the error.

Vorbereitung des Gerätes

26

- Zumindest das USB-Debugging muss aktiviert werden



„Take-Away“ für diese Einheit

27



- Unterschiedliche mobile Entwicklungsansätze
- Aufbau einer Android Anwendung
- Android Komponenten
- Android Entwicklungswerkzeuge

