Android - Basics

Heute

- Was ist Android?
- Programmieren für Android
- App-Struktur
- Activities und Intents
- App-Design
- GUI und Layout

Android in a nutshell

- Open-Source (Open Headset Alliance)
- Basiert auf Linux
- Dalvik VM – optimierte Java VM
- Voll nutzbar nur mit Google Services, z.B.
  - Play Store
  - Maps usw.

  Keine Teile von Android!

Programmiersprachen

- Grundsätzlich: Java
  - Java-Code wird in Dalvik-Format konvertiert
  - Sämtliche systemnahen Java-Funktionen sind nicht nutzbar
- Für Layout und Metadaten: XML
  - Wird letzendlich auch zu Java-Code gemacht, ist aber komfortabler
  - R.java enthält Referenzen für diese Objekte
- "Native development kit": C/C++

Probleme - Energie

- Akkus halten meist nicht lange
- Betriebssysteme werden auf Akkulebenszeit optimiert
- Apps mit hohem Verbrauch fallen auf
- Folgen:
  - Viele Callback-Funktionen
  - Activities und Services können "einfach so" vom System beendet werden
  - PassiveLocationProvider

Probleme - Netzwerk

- Es kann nicht von einer vorhandenen/stabilen Netzwerkverbindung ausgegangen werden
  - System verbietet Netzwerkoperationen im Main-Thread
  - Wichtige Daten zwischenspeichern
- Volumenbeschränkte Tarife/Langsame Verbindung
  - Möglichst wenig Daten schicken/empfangen
  - Mobilfunk ist für http optimiert!

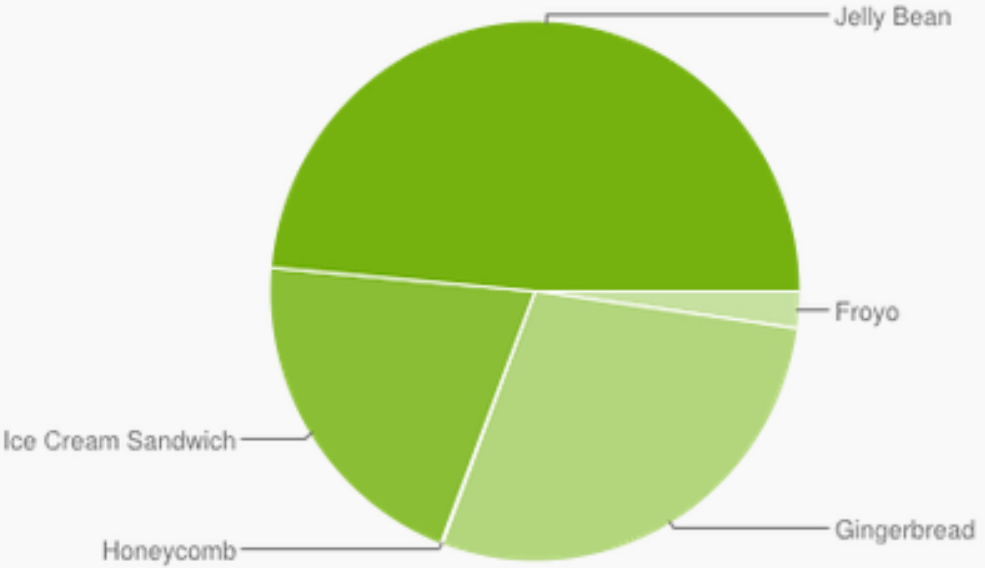
Probleme - Hardware

- Es gibt sehr viele verschiedene Geräte:
  - Angaben in density-independent pixels (dp)
  - Sensoren, Kamera, GPS usw. verschieden
  - Leistung sehr unterschiedlich
- Generell oft wenig Platz und immer nur eine App sichtbar
  - Gutes Layout entscheidend
  - Viel Feedback an den User notwendig

Probleme - Software

Version	Codename	API	Distribution
2.2	Froyo	8	2.2%
2.3.3 - 2.3.7	Gingerbread	10	28.5%
3.2	Honeycomb	13	0.1%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	20.6%
4.1.x	Jelly Bean	16	36.5%
4.2.x		17	10.6%
4.3		18	1.5%

Data collected during a 7-day period ending on October 2, 2013.
Any versions with less than 0.1% distribution are not shown.



http://developer.android.com/about/dashboards/index.html

- Neue Funktionen in neueren Versionen
  - Support-libraries und Fallunterscheidungen

Android Developer Tools

- ADT-bundle (Empfohlen!)
  - http://developer.android.com/sdk/
  - Eclipse mit Android-Plugin
  - SDK-Manager zum Download der benötigten Dateien für jede Android-Version
- Android Studio
  - Von Google als Ersatz für Eclipse entwickelt
  - Basiert auf IntelliJ
  - Beta, relativ instabil

Tutorials, Dokumentation, APIs...

http://developer.android.com/

ist Ausgangspunkt für alles, was man braucht:

- Erklärungen und Einführungen zu allen Funktionen
- Beispiele
- APIs
- Design und Designrichtlinien

...

Geräte

- Hardwarebuttons
  - An/Aus
  - Lautstärkeregler
  - Home

    sollten klar sein.
  - Zurück: führt immer zur letzten Ansicht (nicht notwendigerweise in der gleichen App)
  - Menü: Wird abgeschafft und durch die ActionBar ersetzt

    Man kann die Funktion aber auch überschreiben!

App-Struktur

- Leeres Projekt
- Was ist das Grundgerüst einer App?
  - /src/: Wie in Java, beinhaltet Java-Quelldateien
  - /gen/ und /bin/: Gebautes Projekt
  - /libs/: Verwendete Libraries, z.B. Support-library
  - /res/: Resource-Files, z.B. Layout, Grafiken, Strings
  - AndroidManifest.xml: Meta-Informationen zur App und ihren Bestandteilen

Manifest.xml

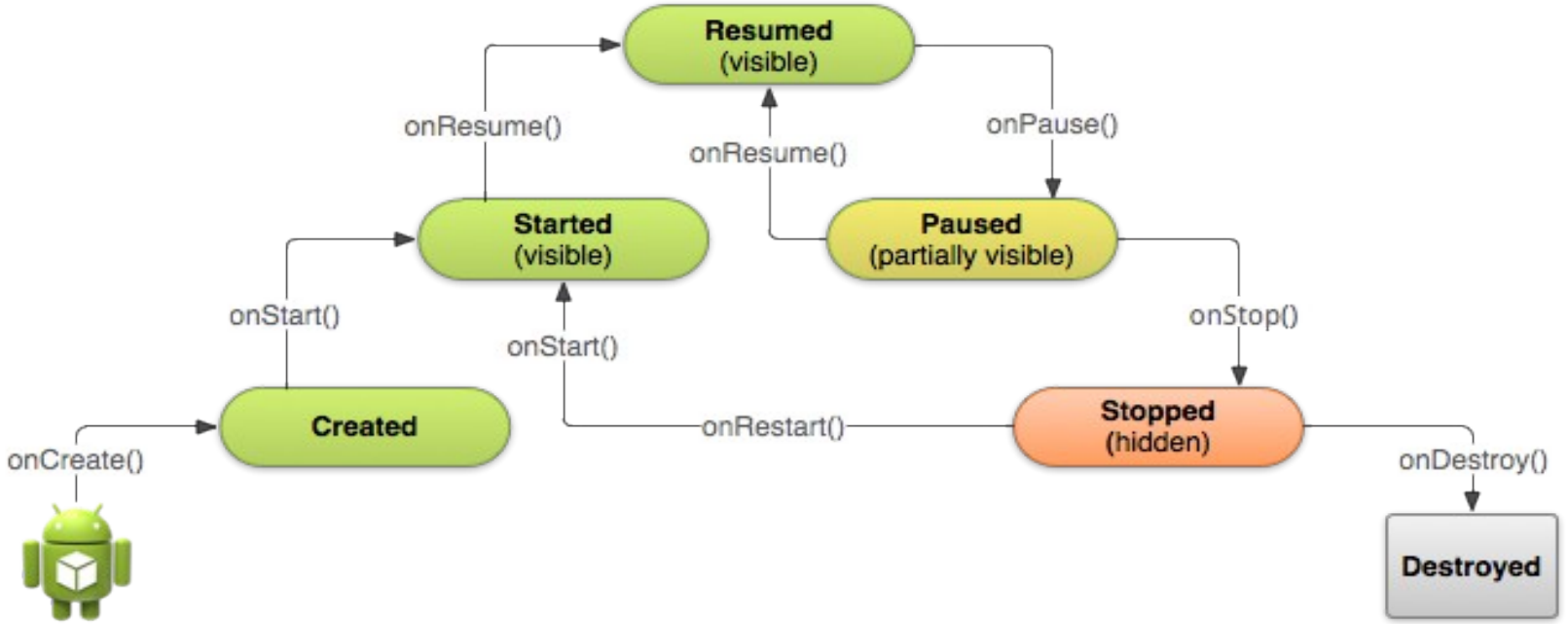
- API-Versionen
- Packagename (wichtig v.a. für Google Play)
- Auflistung von Activities und Services
- Permissions
  - Apps brauchen für viele Aktionen (z.B. Zugriff auf Kontakte) eine Genehmigung
  - Wird diese Permission nicht im Manifest gesetzt, wird eine Exception geworfen
- Intent-Filter (mehr dazu kommt noch)

Hallo Welt

- Live-Demo
- Was sieht man und wo kommt es her?
- Localization mit values-xx
  - Funktioniert ähnlich mit Bildschirmgrößen-/und Formaten (z.B. res/drawable-small-land-stylus)

Activity

- Meist in voller Größe angezeigt
- Nur eine Ansicht innerhalb der App
- Demo Lifecycle:



Intent

- Systemnachricht, die immer eine Aufgabe enthält
- Parameter möglich

```
Intent callIntent = new
Intent(Intent.ACTION_CALL);
callIntent.setData(Uri.parse("tel:"+number));
startActivity(callIntent);
```

- Aufruf anderer Activities innerhalb der App:

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

IntentFilter

- Wertet aus, ob ein Aufgabe übernommen werden kann
- Erlaubt es z.B., sich für Aktionen (Telefon) oder Protokolle (im Sinne einer URI) zu registrieren

```
<action android:name="android.intent.action.VIEW"></action>

<category
android:name="android.intent.category.DEFAULT"></category>

<category
android:name="android.intent.category.BROWSABLE"></category>

<data android:host="www.medien.ifi.lmu.de"
path="/lehre/ws1314/pem/" android:scheme="http"></data>
```

Soweit die Theorie...

- Fragen?
- Pause?
- Danach:

  Crashkurs Android-App-Design

  Layout

  Wichtigste Views und GUI-Elemente

Design-Philosophie

- Einheitliches Design erleichtert dem User das "erlernen" neuer Apps
  - Nicht iOS- oder Windows Phone-Elemente nachbauen, weil sie besser aussehen oä.
  - Am besten nur von Google-Apps abschreiben
  - Wie man es nicht machen sollte: DB-Navigator

    Konsistenz über Plattformen hinweg ist meist sinnlos!
- Details auf http://developer.android.com/design/get-started/creative-vision.html

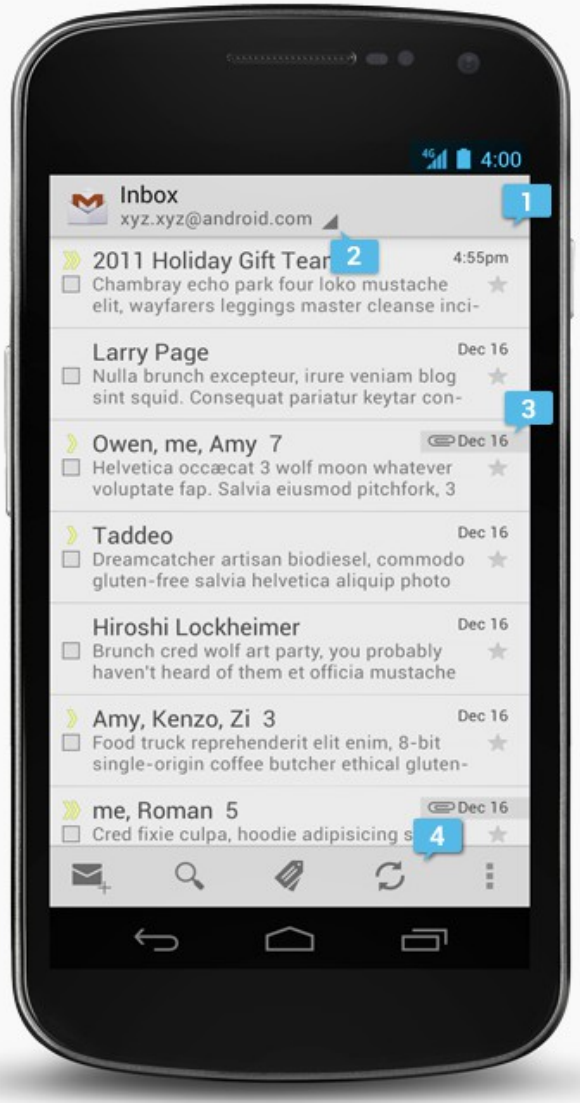
Design-Prinzipien

- Real objects are more fun than buttons and menus
- Get to know me
- Keep it brief
- Pictures are faster than words
- Only show what I need when I need it
- If it looks the same, it should act the same
- Only interrupt me if it's important
- Make important things fast

http://developer.android.com/design/get-started/principles.html

UI-Aufbau

Common App UI



A typical Android app consists of action bars and the app content area.

1. **Main Action Bar**

   The command and control center for your app. The main action bar includes elements for navigating your app's hierarchy and views, and also surfaces the most important actions.

   More on the Action Bar

2. **View Control**

   Allows users to switch between the different views that your app provides. Views typically consist of different arrangements of your data or different functional aspects of your app.

3. **Content Area**

   The space where the content of your app is displayed.

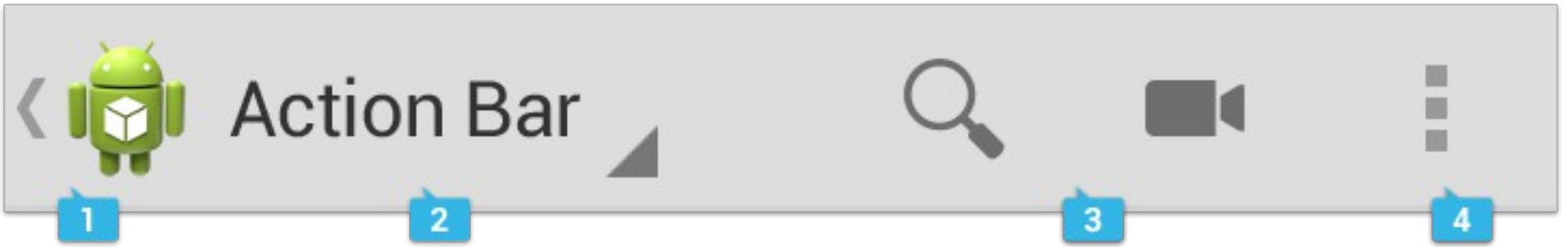
4. **Split Action Bar**

   Split action bars provide a way to distribute actions across additional bars located below the main action bar or at the bottom of the screen. In this example, a split action bar moves important actions that won't fit in the main bar to the bottom.

Alte Version von Gmail!

http://developer.android.com/design/get-started/ui-overview.html

Action Bar



1. Icon – führt eine Hierarchiestufe nach oben
2. App-Name oder Auswahl der darunterliegenden View (z.B. In der Galerie: Alben, Orte, Zeiten…) über Spinner
3. Häufig benutzte Aktionen (Action buttons)
4. "Action overflow" – weniger häufig benutzt

Navigation Drawer

Spiegel Online-App

- Kann die Navigation zu allen Views übernehmen
- Wann sollte man den Navigation Drawer einem Spinner oder einer Tabbed View vorziehen?

Layout

- Der statische Teil wird in xml-Dateien definiert
- Demo LinearLayout
- Demo Button
- sp und dp

Nächste Woche

- Praxis oder Theorie?
- Habt ihr Themenwünsche?

  Beispiele:

  - Maps
  - Facebook-API
  - Mehr GUI
  - Services (für Hintergrunddienste)

  ...