

Einführung: Android Programmierung (Teil 1)

Einführung: Android Programmierung (Teil 1)

Grundlagen - Life Cycle - UI

David Tanzer

business@davidtanzer.net
<http://davidtanzer.net>

Fragen? - Fragen!



Qualifikation

- Über den Vortragenden
 - Freiberuflicher SW-Entwickler und Berater seit 2006
 - Certified Scrum Master
 - Bakk. Techn. (Johannes Kepler Universität Linz)
- Android – Erfahrung
 - Generelles Interesse an Mobiler Softwareentwicklung
 - AndroHud
 - aWoman

<http://davidtanzer.net>

Aufbau

- Grundlagen
- Life Cycle
- Ressourcen
- Benutzerschnittstelle

Aufbau

- Grundlagen
- Life Cycle
- Ressourcen
- Benutzerschnittstelle

Android – Überblick

- „Freies“ Betriebssystem für mobile Geräte
 - Smartphones
 - „Tablets“
 - Netbooks
- Software – Plattform
- Initiiert von Google
- Entwickelt von der Open Handset Alliance



Android Verbreitung

- Mehr als 20 Hardware – Hersteller erzeugen oder planen Android – Handsets
 - HTC, Motorola, Samsung, Dell, Sony Ericsson, ...
- 60.000 verkaufte Handsets pro Tag (lt. Google – Februar 2010)
- In Deutschland erhältlich bei T-Mobile, Vodafone, E-Plus und o2 [1]
- Billige Geräte ohne Vertrag

[1] <http://www.teltarif.de/android-google-ein-jahr-jubilaem/news/37450.html>

Android Versionen

- 1.1 (10.2.2009)
- 1.5 (Cupcake – 30.4.2009)
 - Bildschirm-Tastatur
 - Aufnahme und Wiedergabe von Videos
- 1.6 (Donut – 15.9.2009)
 - Virtual Private Networks
- 2.0 (Eclair – 26.10.2009)
 - Microsoft Exchange Unterstützung
- 2.1 (Eclair - 12.1.2010)

Architektur (1/3)

- Linux Kernel (2.6.29 in Eclair)
- Java
 - Apache Harmony Klassenbibliothek
 - Weder J2SE noch J2ME
 - Eigene GUI – Bibliothek
- Dalvik – VM
 - Registerbasiert
 - Bytecode wird durch Cross-Assembler aus Java – Bytecode erstellt
 - Optimiert für mobile Geräte

Architektur (2/3)

- Native Libraries
 - Surface Manager: Window – Manager
 - 2D und 3D Graphik (OpenGL ES)
 - Media Codecs (Audio, Video)
 - SQL Datenbank (SQLite)
 - Browser Engine (Webkit)
- Java – Wrapper für alle Native Libraries
- Native – Entwicklung mit NDK möglich

http://developer.android.com/sdk/ndk/1.6_r1/index.html

Architektur (3/3)

- Application framework
 - Activity Manager: Life Cycle von Anwendungen, Navigationsstack
 - Content Providers: Datenaustausch zwischen Anwendungen
 - Resource Manager: Übersetzungen, Bilder, ...
 - Sensor Manager: GPS, Kompass, Neigung, Temperatur, ...
 - Notification Manager: Dem Benutzer Ereignisse präsentieren
- Alle Anwendungen sind gleichwertig
 - Alle können durch eigene Anwendungen ausgetauscht werden
 - Auch Adressbuch, Dialer, Kalender, ...

Android SDK

- Eclipse – Plugin, Emulator, AVDs, ...
- → Demo

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

Bausteine einer Android - Anwendung

- Activities: Jeder Bildschirm ist eine Activity
- Services: Hintergrundprozesse
- Content Providers: Datenaustausch zwischen Anwendungen
- Intents: Message – Passing (innerhalb der Anwendung oder systemweit)
- Broadcast Receivers: „Hören“ auf Broadcast – Intents
- Notifications: Benutzer benachrichtigen (z.B. SMS, eingehender Anruf, ...)
- Application Manifest: XML – Metadaten der Anwendung und deren Komponenten

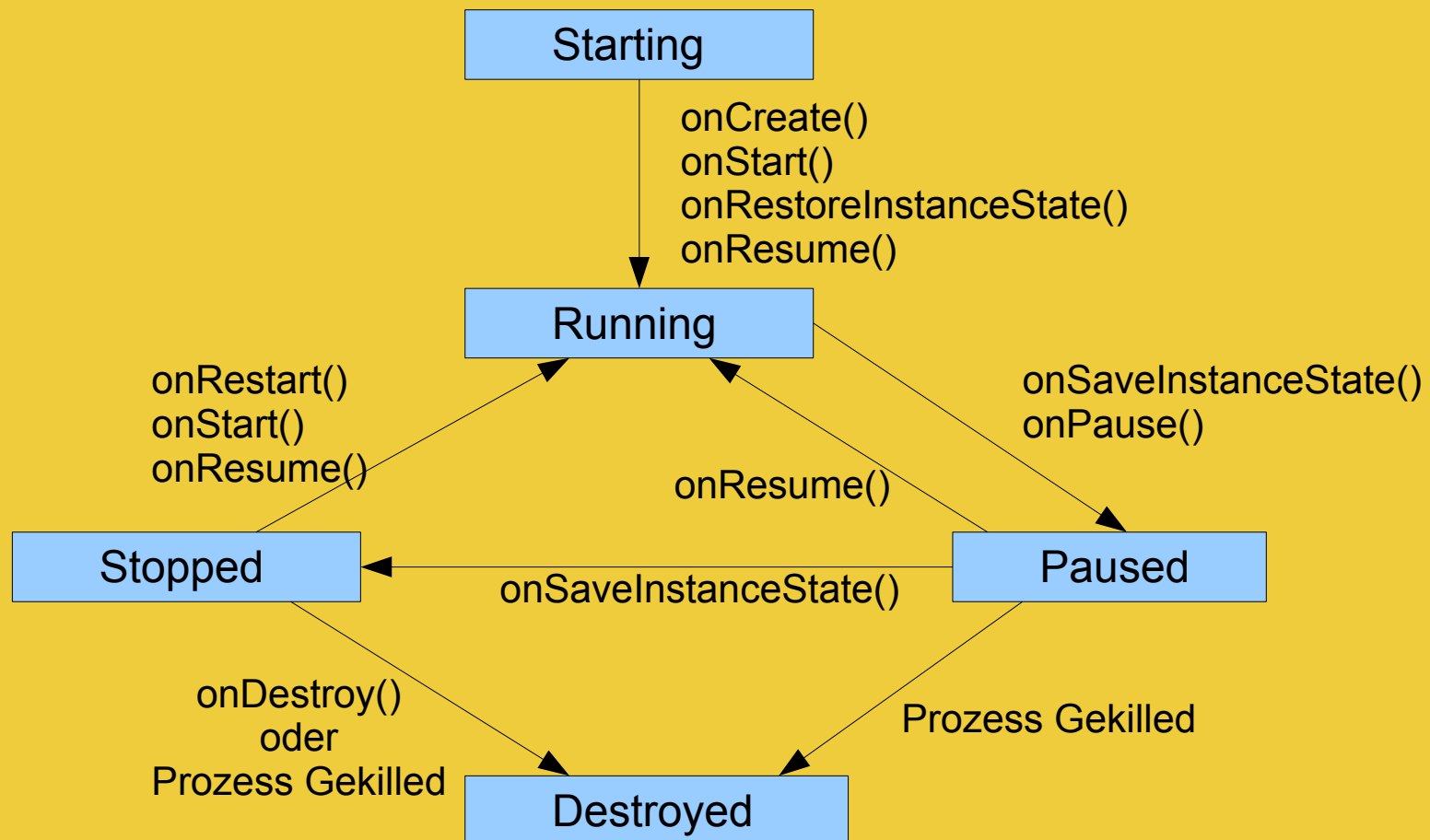
Aufbau

- Grundlagen
- Life Cycle
- Ressourcen
- Benutzerschnittstelle

Prozess - Hierarchie

- Prozesse werden so lange wie möglich am Leben gehalten
- Können jederzeit gestoppt werden, um Ressourcen für höher priorisierte Prozesse freizugeben
- Hierarchie:
 - Vordergrundprozess: z.B. aktuelle Activity
 - Sichtbarer Prozess: z.B. Activity hinter einem Dialog
 - Service – Prozess: gestarteter Hintergrund-Service
 - Hintergrundprozess: z.B. gestoppte Activity
 - Leerer Prozess: keine aktiven Komponenten

Life Cycle einer Activity (1/5)



Life Cycle einer Activity (2/5)

- Voller Lebenszyklus:
 - `onCreate()` bis `onDestroy()`
 - `onDestroy()` kann entfallen
- Sichtbar:
 - `onStart()` bis `onStop()`
 - `onStop()` kann in seltenen Fällen entfallen
- Aktiv:
 - `onResume()` bis `onPause()`
 - `onPause()` sollte überschrieben werden

<http://developer.android.com/guide/topics/fundamentals.html>

Reto Meier: „Professional Android Application Development“, Kapitel 3

Life Cycle einer Activity(3/5)

- `onCreate()` : Einmalige Initialisierung (z.B. UI)
- `onStart()` : Danach wird die Activity angezeigt
- `onResume()` : Danach kann mit dem Benutzer interagiert werden
 - Hier z.B. Animationen oder Musik starten
- `onPause()` : Activity geht in den Hintergrund
 - Hier z.B. persistenten Status speichern
- `onRestart()` : Start nachdem die Activity gestoppt war
- `onDestroy()` : Aufruf bevor die Activity zerstört wird

Life Cycle einer Activity (4/5)

- `onSaveInstanceState ()`:
 - z.B. UI Status speichern
 - Default - Implementierung macht das
- `onRestoreInstanceState ()`:
 - z.B. UI Status laden
 - Default - Implementierung macht das

Life Cycle einer Activity (5/5)

- Demo

Aufbau

- Grundlagen
- Life Cycle
- Ressourcen
- Benutzerschnittstelle

Ressourcen (1/4)

- „Non-Code“ Informationen
 - Strings für Internationalisierung
 - Farben
 - Bilder
 - ...
- Im „res“ - Verzeichnis des Projekts
 - Bilder: `res/drawable`
 - Strings: `res/values`
 - UI: `res/layout`

Ressourcen (2/4)

- Können andere Ressourcen referenzieren

```
<EditText  
    android:text="@string/message"  
    ...  
>
```

- Werden im Code über die Klasse „R“ verwendet
- Resource-Compiler (aapt) erzeugt die Klasse „R“
 - In Eclipse (ADT) automatisch
- System - Ressourcen: Klasse „android.R“

Ressourcen (3/4)

- Internationalisierung:

```
Projekt/
```

```
  res/
```

```
    values/
```

```
      strings.xml
```

```
  values-de/
```

```
    strings.xml
```

```
  valuse-de-rAT/
```

```
    strings.xml
```

Ressourcen (4/4)

- Demo: Layout; String Ressourcen und Internationalisierung

Aufbau

- Grundlagen
- Life Cycle
- Ressourcen
- **Benutzerschnittstelle**

• Benutzerschnittstelle (1/3)

- Kann deklarativ oder prozedural erstellt werden
- Deklarativ: .xml – Dateien in `res/layout`
 - Ist die von Google empfohlene Variante
- Prozedural: Im Java – Code
 - Vergleichbar mit Swing oder anderen Toolkits

• Benutzerschnittstelle (2/3)

- View: Basisklasse für Elemente der Benutzerschnittstelle
- ViewGroup: Können mehrere Views enthalten
 - Layout
 - Widget
- Activity: Ein Fenster bzw. Screen
- Widget Toolbox: Standard – Views von Android
 - Button, TextView, ListView, RadioButton, ...
- Layout Managers
 - FrameLayout, LinearLayout, TableLayout, ...

- **Benutzerschnittstelle (3/3)**

- Demo: Button, der eine andere Activity öffnet

Fragen? - Fragen!



Fragen?

Vielen Dank!

business@davidtanzer.net