



Android Entwicklung

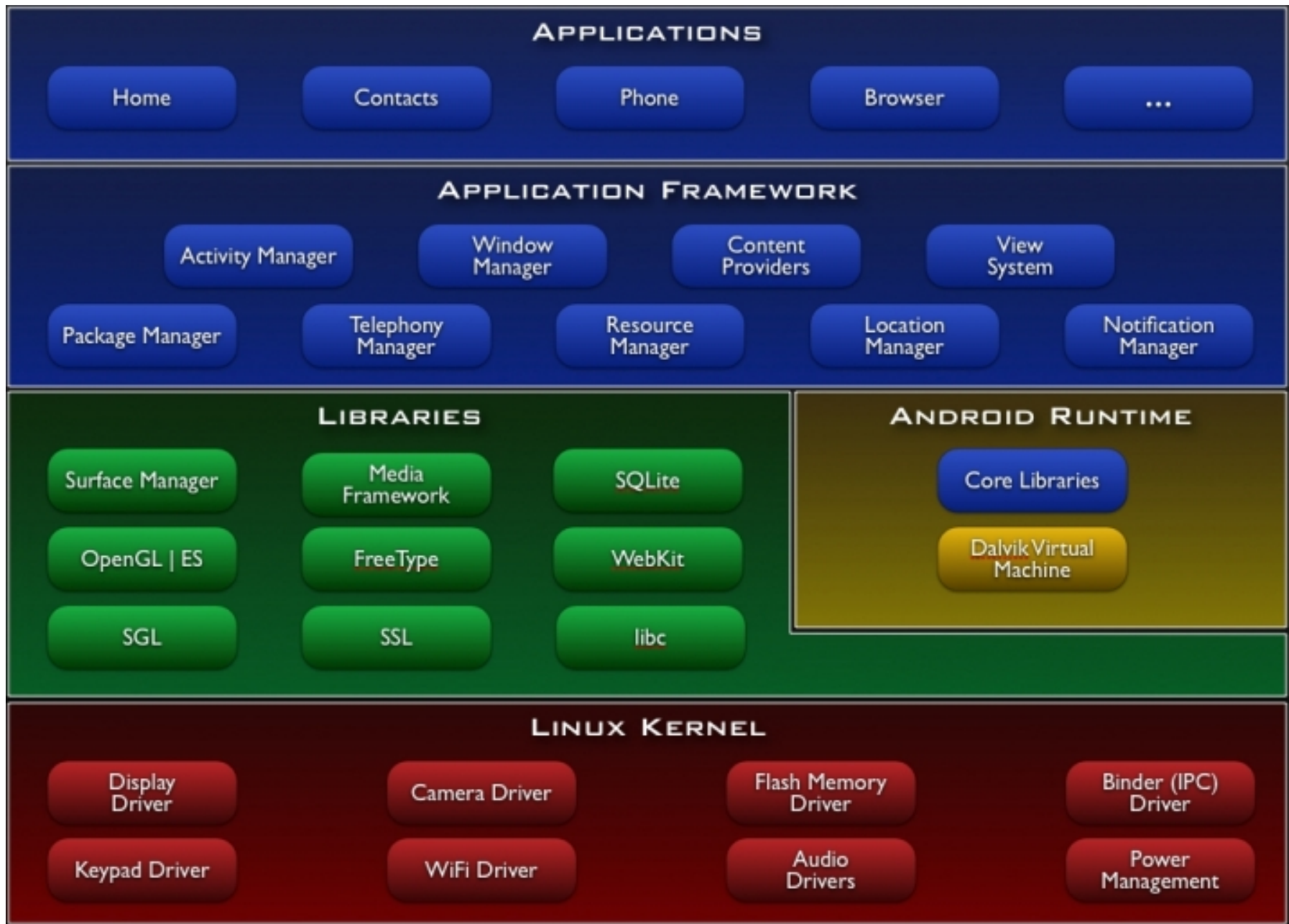
Mobile Camp Dresden 2009

Markus Junginger

greenrobot

Outline

- Android Entwicklung allgemein
- Activities & Intents
- Hello World
- AndroidManifest.xml
- UI Grundlagen
- 2D, Farbe, Performance

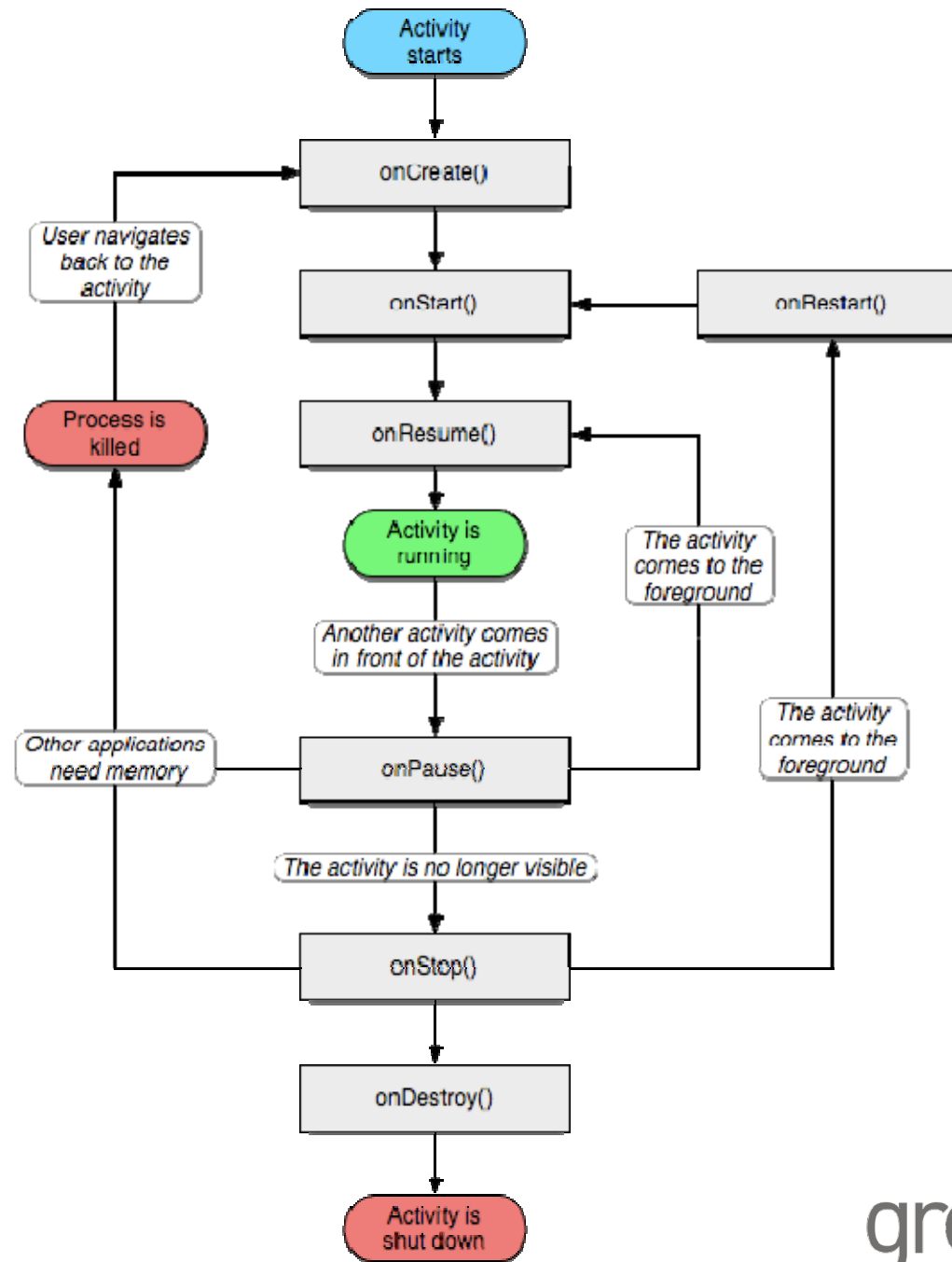


Android Entwicklung

- Java 5 (Scala, JRuby)
- Java 5 SE APIs teilweise vorhanden (io,nio, lang, util, math, etc.)
- Proprietäre Android APIs (android.*)
- Java Bytecode wird in DEX umgewandelt
- Dalvik Virtual Machine
- IDE: Eclipse (oder von Hand)
- Device Emulator

Activities

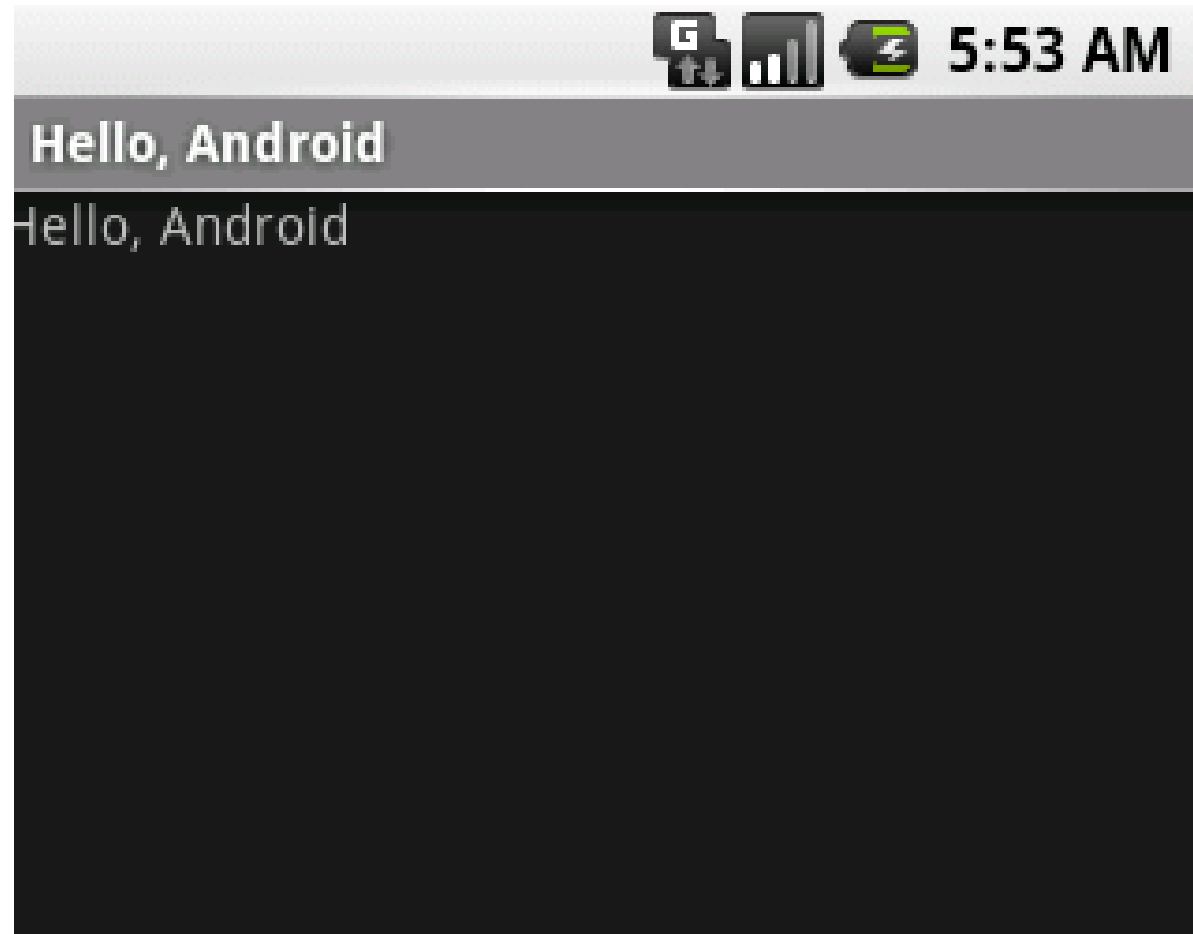
- Einstiegspunkt in Android Programm
- Aktiver Programmteil
- Eine Activity setzt (mind.) ein View (GUI)
- Activities unterliegen Life Cycle
- onResume und onPause (Persistenz!)
- Android OS kann Activities killen
- Alternative: Background Services



Intents

- Eine Absicht etwas zu tun ohne zu definieren wie es getan werden soll
- Ermöglicht Vernetzung von Apps über lose Kopplung (eins meiner Lieblingsfeatures)
- Angelehnt an URLs (Beispiel: „tel:123“)
- Auch für das Aktivieren eines neuen Zustand einer App: Starten einer Activity

Hello World



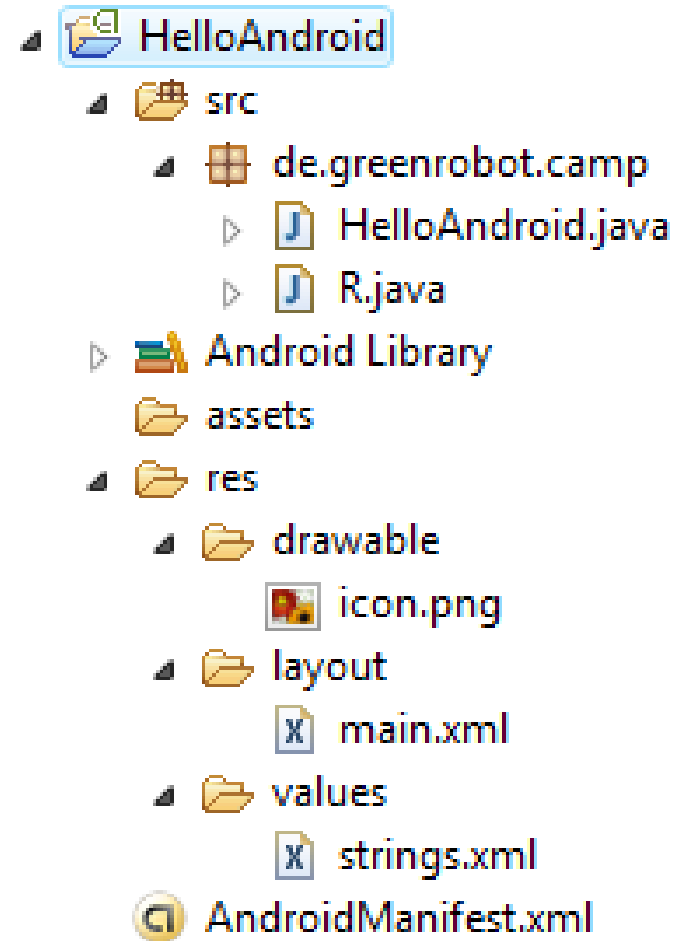
Hello World

```
public class HelloAndroid extends Activity {  
    @Override  
    public void onCreate(Bundle state) {  
        super.onCreate(savedInstanceState);  
        TextView tv = new TextView(this);  
        tv.setText("Hello, Android");  
        setContentView(tv);  
    }  
}
```

→ Zusätzlich AndroidManifest.xml nötig

Neues Android Eclipse Projekt

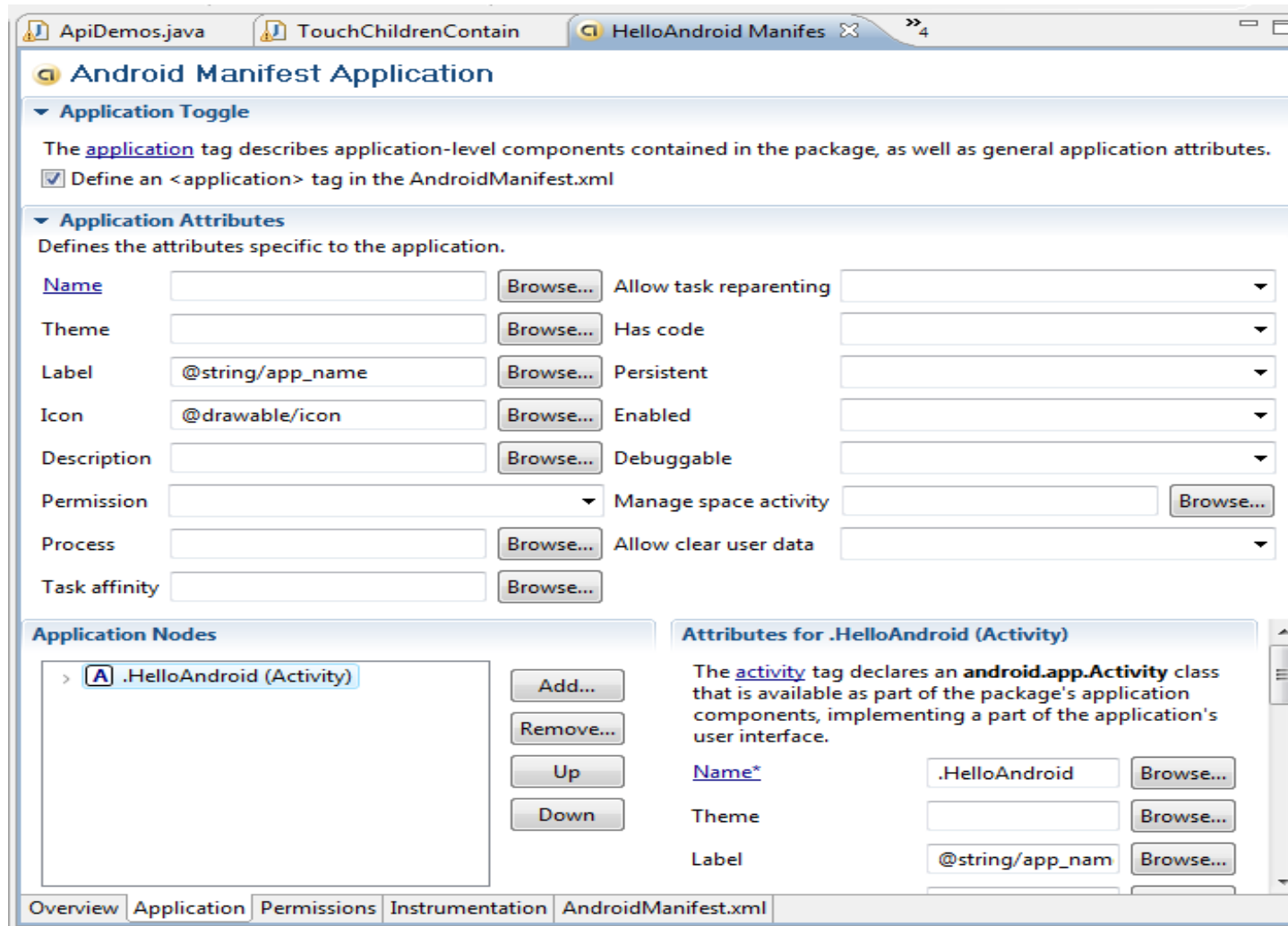
- IDE erstellt Gerüst
- R.java ist generiert
- res Verzeichnis



AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
  xmlns:android="http://schemas.android.com/apk/res/android"
  package="de.greenrobot.camp"
  android:versionCode="1"
  android:versionName="1.0.0">
  <application android:icon="@drawable/icon"
    android:label="@string/app_name">
    <activity android:name=".HelloAndroid"
      android:label="@string/app_name">
      <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER"/>
      </intent-filter>
    </activity>
  </application>
</manifest>
```

AndroidManifest Editor



Funktion des AndroidManifest

- Activities und Services registrieren
- Permissions setzen; z.B.
 - Internet
 - Zugriff auf Kontakte
 - Power Management
- Registrieren auf Events (IntentFilter)

UI Grundlagen

- Als Ressourcen oder programmatisch
- Views (Widgets): Basisklasse ist View
- Layouts (Linear, Grid, Table, Relative, ...)
- UI Thread: Änderungen an der GUI
- Events (OnClick, ...)
- Visueller GUI Editor in Eclipse

The screenshot shows the Android Studio IDE with the following components:

- Top Bar:** Tabs for 'HelloAndroid Manifest', 'main.xml', and 'strings.xml'. A configuration bar with various settings (MCC, MNC, Lang, Region, Orient, Density, Touch, Keybrd, Input, Nav) and a 'Create...' button.
- Main Editor:** A preview of the app showing 'Hello World!' on a black background.
- Outline Pane:** A tree view showing a 'LinearLayout' containing a 'TextView'.
- Properties Pane:** A table of properties for the selected 'TextView'.

Property	Value
Shadow color	
Shadow dx	
Shadow dy	
Shadow radius	
Single line	
Sound effects enabled	
Tag	
Text	@string/hello
Text appearance	
Text color	
Text color highlight	
Text color hint	
Text color link	
Text scale X	
Text size	
Text style	
Typeface	
Visibility	
Width	

greenrobot

Typische Views

- TextView
- Button
- Checkbox
- ImageView
- WebView (Achtung: JavaScript aktivieren)
- MapView (benötigt Registrierung/Key)

ListView

- Adapter hält die Daten der Liste vor
- ListActivity vereinfacht Handhabung
 - setListAdapter z.B. mit ArrayAdapter aufrufen
- Eigene Adapter von BaseAdapter ableiten
 - Object **getItem**(int position)
 - View **getView**(int position, View convertView, ViewGroup parent)

2D Grafik für Spiele etc.

- View ableiten und onDraw implementieren
- Canvas Objekt wird übergeben
 - Bitmaps, Rechtecke, Linien, etc. zeichnen
- Empfohlen: SurfaceView da effizienter
 - Losgelöst vom UI Thread
 - Game Loop kann so direkt UI aktualisieren
 - Für OpenGL nötig

Farben, Farbmodelle

- Verschiedene Farbmodelle, z.B.
 - ARGB 8888: 24 bit Farbe mit 8 Bit Alpha
 - RGB 565: 16 bit Farbe
- RGB 565 ist natives Format auf G1:
sehr performant
- Farbwerte über Ressourcen definierbar
- Farbwerte als ints: `0xff00ff00` → grün

Performance

- Gute Performance für normale Apps
- Optimierungen nötig für:
 - Background Services (be nice)
 - Spiele, Multimedia (flüssiger Ablauf)
- GC legt VM für 100-200ms lahm
- Regel #1: GC vermeiden. Keine neuen Objekte in zentralen Schleifen anlegen
- Versteckte Anlage von Objekten

Weitere Ressourcen

- [Android.com](https://android.com)
- Beim SDK dabei: Dev Guide, Referenz und API Demos mit Source
- Dev Guide hat gute How-tos
- [Android Developer Blog](https://android-developer.blogspot.com/)
- jars.de und greenrobot.de 😊

Android Branchentreffen

- Ca. Herbst des Jahres
- Unterstützt von Telekom, Google, etc.
- Entwickler- und Businessstrack
- Gesucht: weitere Inhalte und Vortragende
→ Bei Interesse einfach bei mir melden
- Android XING Gruppe

Sneak Preview: Android Game

- Kugel & Beschleunigungssensor
- „Bumper“, Gegner, Transporter, ...
- Große Spielfelder
- Grafik und Sounds fehlen noch
- Leveleditor
- Beta Ankündigung im greenrobot Blog



Vielen Dank! Fragen?

Markus Junginger

markus@greenrobot.de

<http://greenrobot.de>

<http://jars.de>

greenrobot