

Android

Applikationsentwicklung auf
mobilen Endgeräten



Vortrag von Mathias Menninghaus
Universität Osnabrück, 15.04.2010



Grundlagen

Übersicht

- Systemarchitektur
- Einführung in grundlegende Konzepte
- Beispielapplikation
- Entwicklungsumgebung
- weiterführende Hinweise

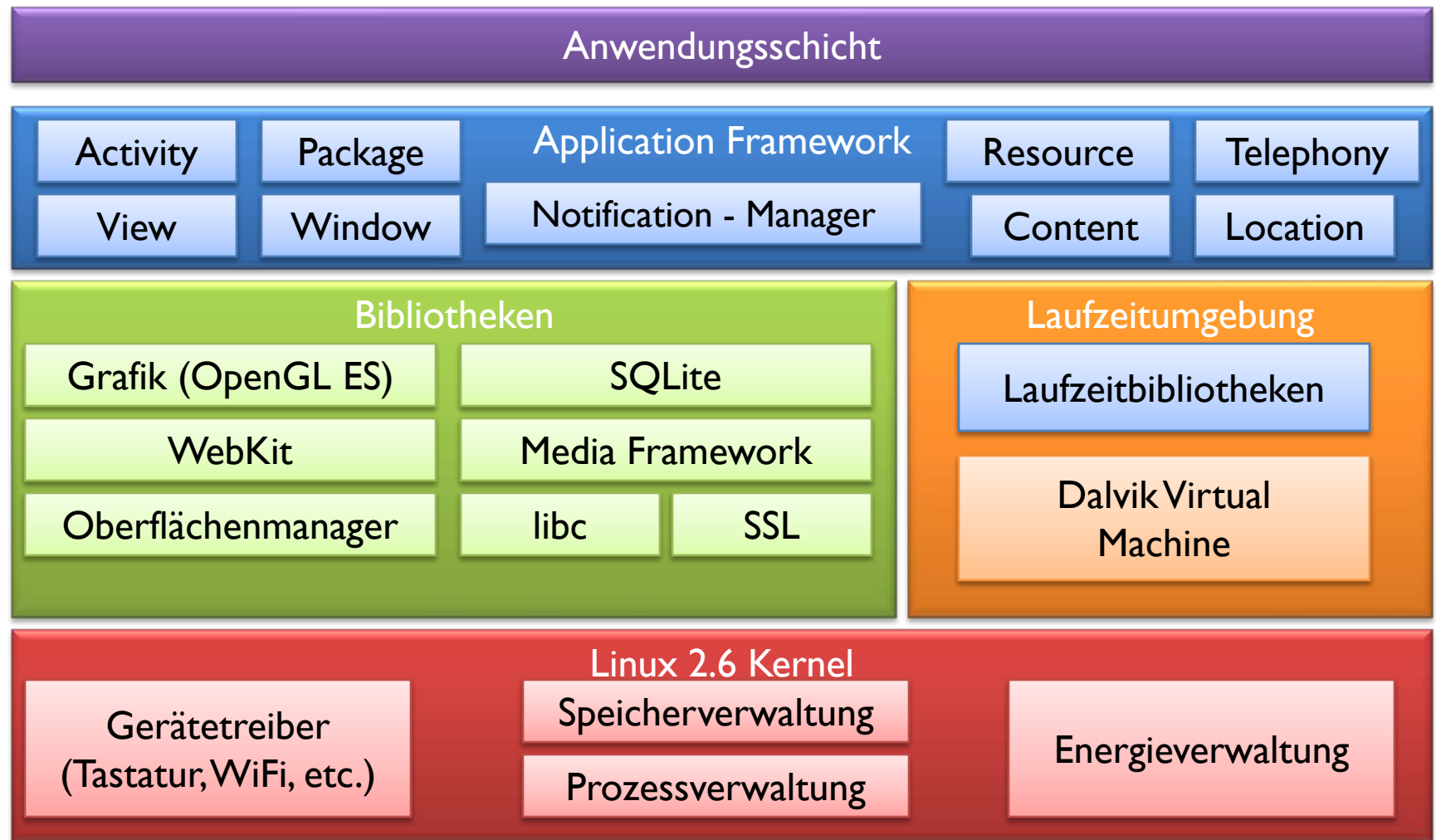
Android

- Freies Open-Source Betriebssystem
- auf Basis von Java und Linux
- betreut von Google
- entwickelt von der Open Handset Alliance



Grundlagen

Systemarchitektur



Systemschicht

- basierend auf Linux 2.6
- unzugänglich für den Entwickler
- Prozess-, Speicher- und Energieverwaltung
- Treiberverwaltung
- sorgt für Sicherheit



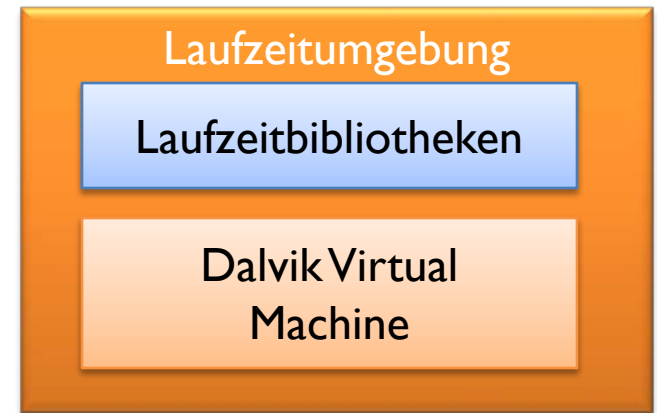
Bibliotheken

- C/C++ Bibliotheken
- Native Ausführung
- können vom Entwickler nicht geändert werden

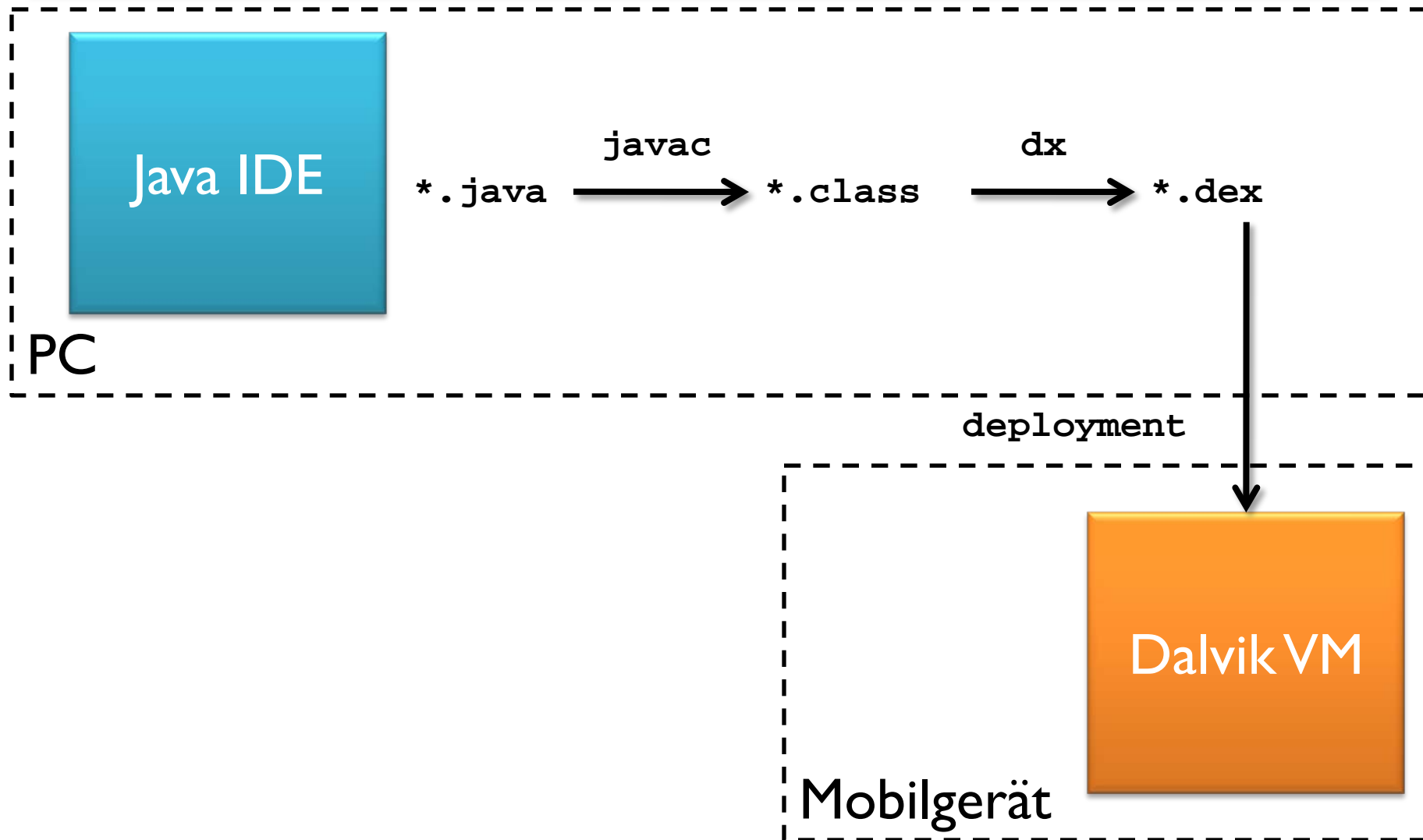


Dalvik Virtual Machine

- benannt nach dem Ort Dalvik (Island)
- JVM vs. DVM:
 - nutzt Register
 - und RISC – Architektur
- Abgrenzung zu J2ME
- nicht unter Sun Lizenz
- eine Virtual Machine pro Applikation



Erzeugung von Bytecode



Sicherheit

- pro Applikation:
 - ein User
 - fester Speicherbereich (GI ~ 16 MB)
 - eine Dalvik VM
 - Sandbox
- in anfälligen Applikationen zusätzlich eigene Sicherheitsschicht einfügen
- Zugriff auf sicherheitsrelevante Inhalte über Permissions

Permissions

- über 100 Permissions verfügbar
- eigene können erzeugt werden
- Verankerung der Berechtigungen im Manifest der Applikation:

```
<uses-permission  
android:name="android.permission.INTERNET" />
```

- Benutzer wird bei der Installation informiert
- ebenso wie bei einer Security-Exception

Application Framework



- Java - Klassen aus Android Bibliotheken
- Telephony, Notification und Package Manager
- Location Manager und Content Provider
- Activity – Manager

Grundlagen

Prozess - Organisation

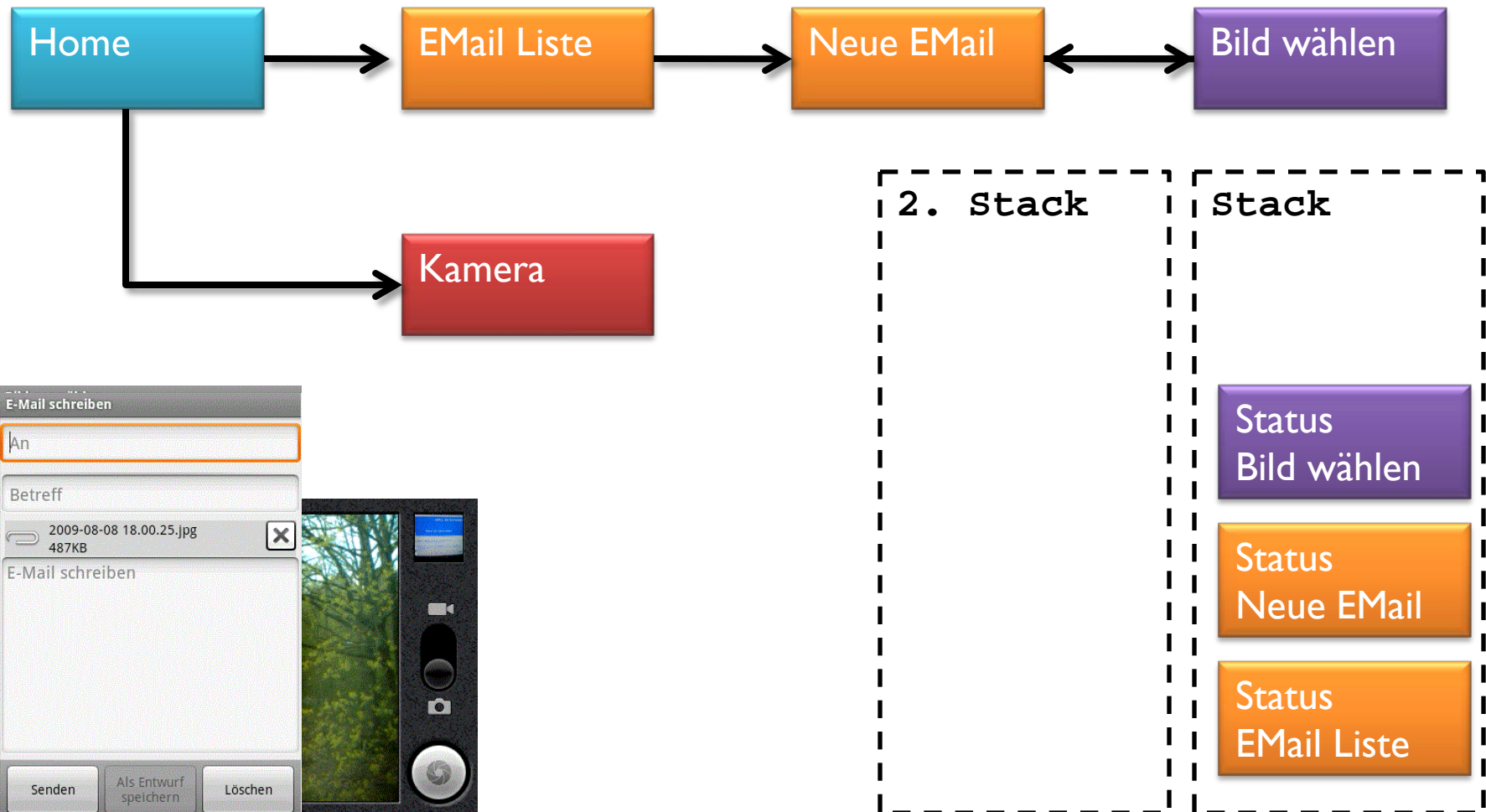
Activity

Application Framework

Linux 2.6 Kernel

Prozessverwaltung

Prozess - Organisation

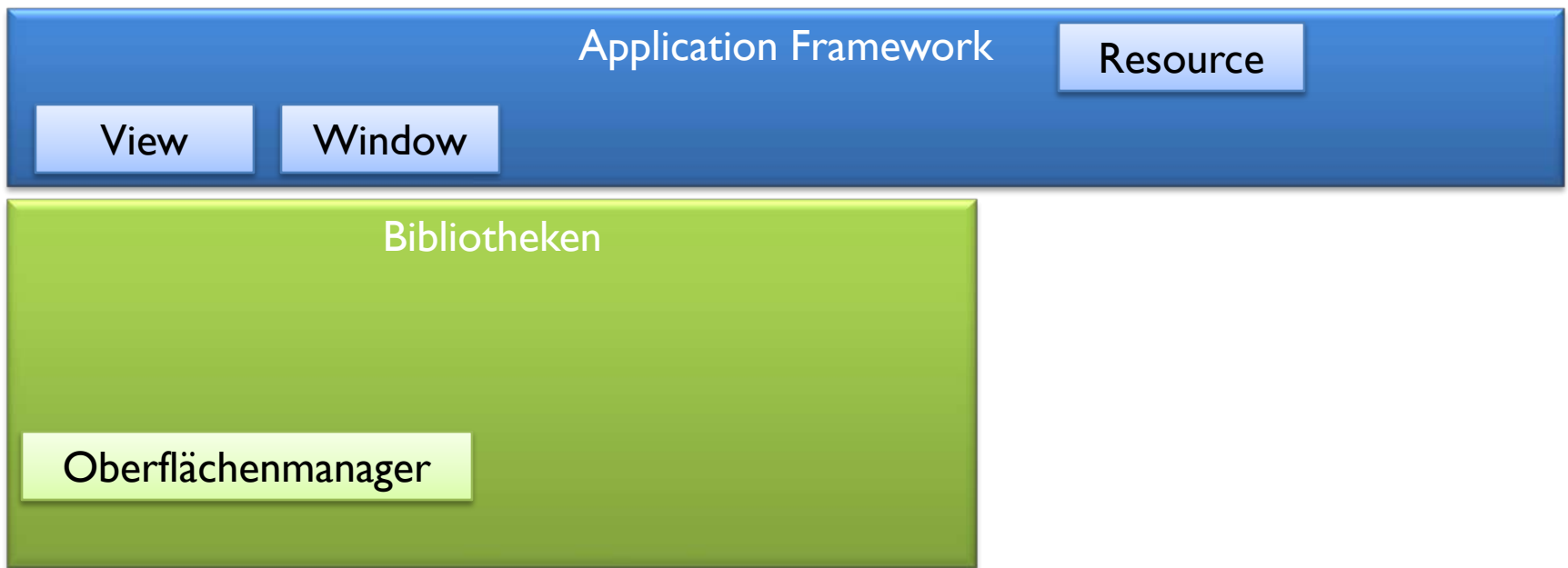


Prozess - Organisation

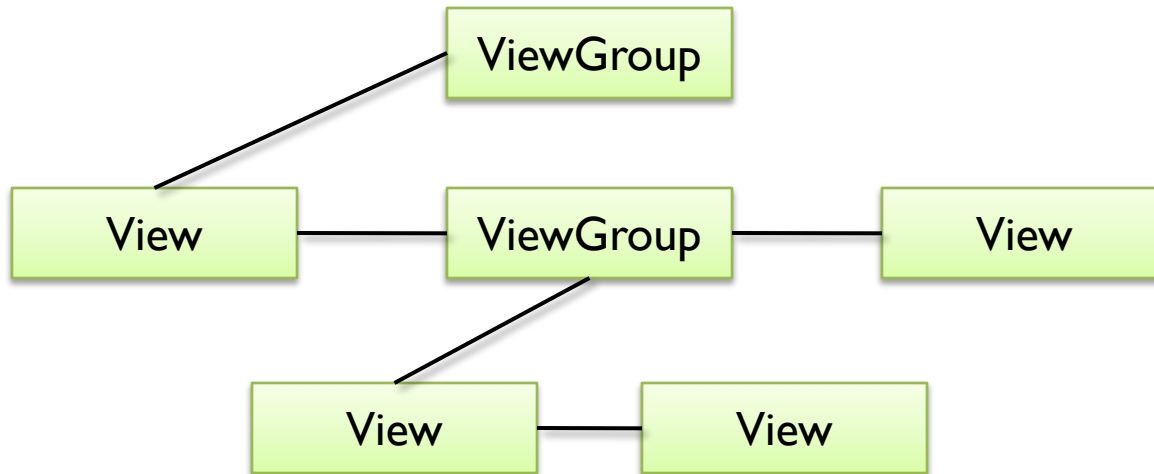
- ein Task enthält den Ablauf einer Applikation
- zwischen Tasks kann gewechselt werden
- Applikationen können von verschiedenen Punkten aus gestartet werden
- sie bestehen aus Activities
- ist eine aktive Activity nicht im Vordergrund, wird ihr Status in einem Keller vorgehalten
- eine Activity kann sich den Funktionalitäten anderer Activities bedienen

Grundlagen

View System



View System



1. Größe bestimmen
2. relative Größe bestimmen
3. Layout berechnen

Praxis

Layouts

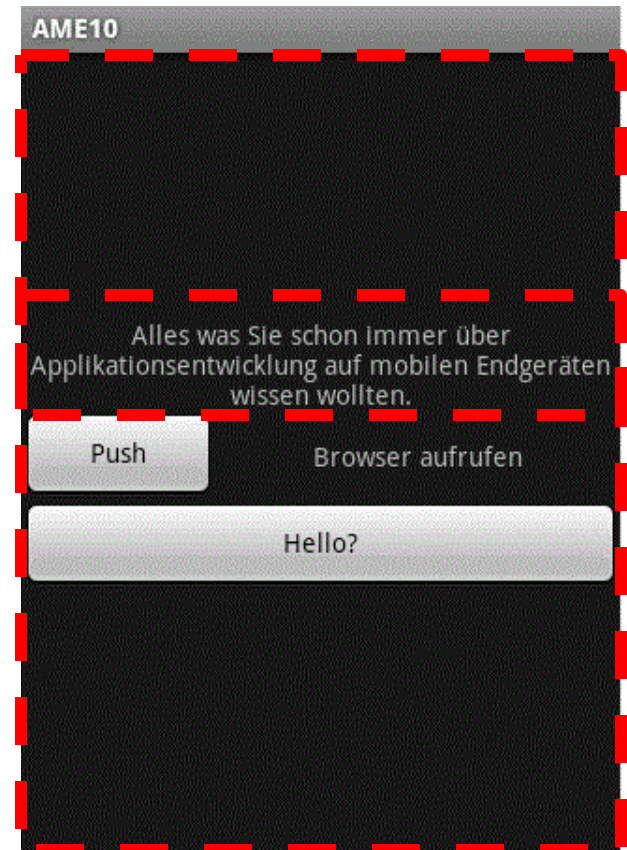
- Frame Layout
- Relative Layout
- Table Layout
- Grid Layout
- Linear Layout

Beispielview

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android=
"http://schemas.android.com/apk/res/android"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
android:orientation="vertical"
android:gravity="center_vertical">

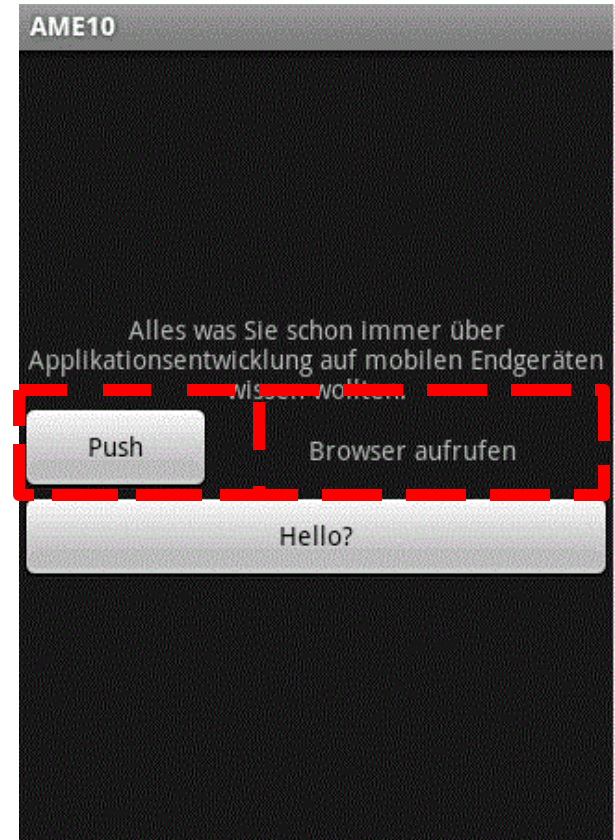
<TextView android:id="@+id/description"
android:gravity="center"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text="@string/description"
/>
...
```

@ Ressourcen-Typ
+ muß noch erzeugt werden
: Packet - Trennung



Beispielview

```
...  
<LinearLayout  
  android:layout_width="fill_parent",  
  android:layout_height="wrap_content"  
  android:orientation="horizontal"  
  android:gravity="center_horizontal">  
  
  <Button android:id="@+id/ame10:browser"  
    android:layout_width="wrap_content"  
    android:layout_height="fill_parent"  
    android:text="Push"  
    android:gravity="center"  
    android:layout_weight="1"/>  
  
  <TextView  
    android:layout_width="wrap_content"  
    android:layout_height="fill_parent"  
    android:text="@string/browser"  
    android:gravity="center",  
    android:layout_weight="2"/>  
  
</LinearLayout>  
...
```



Beispielview

```
...  
<Button android:id="@+id/ame10:helloandroid"  
  android:layout_width="fill_parent"  
  android:layout_height="wrap_content"  
  android:text="Hello?,"  
  android:gravity="center" />  
  
</LinearLayout>
```

Typ

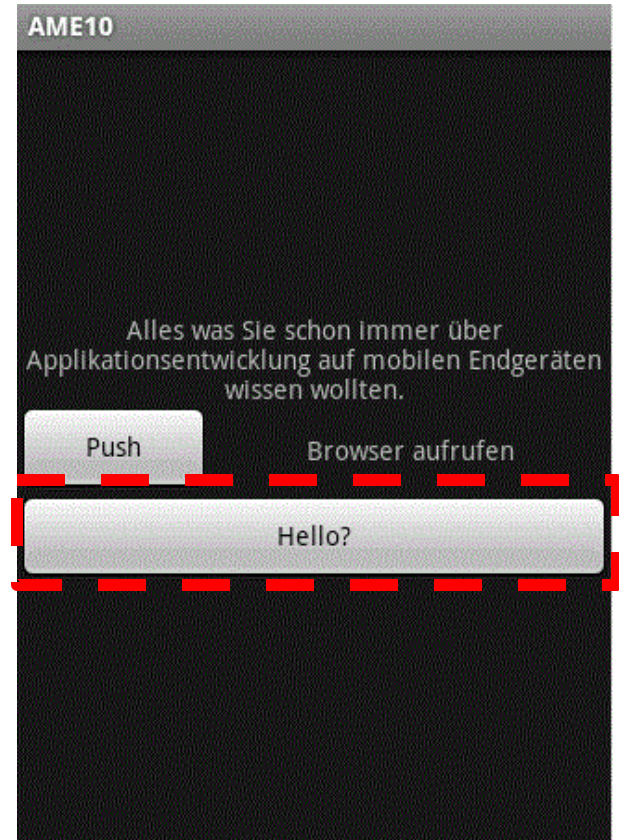
Views

Konstanten (z.B. Strings) /res/values

Bilder/Animationen /res/drawable

Ort

/res/layout



Anbindung an Java

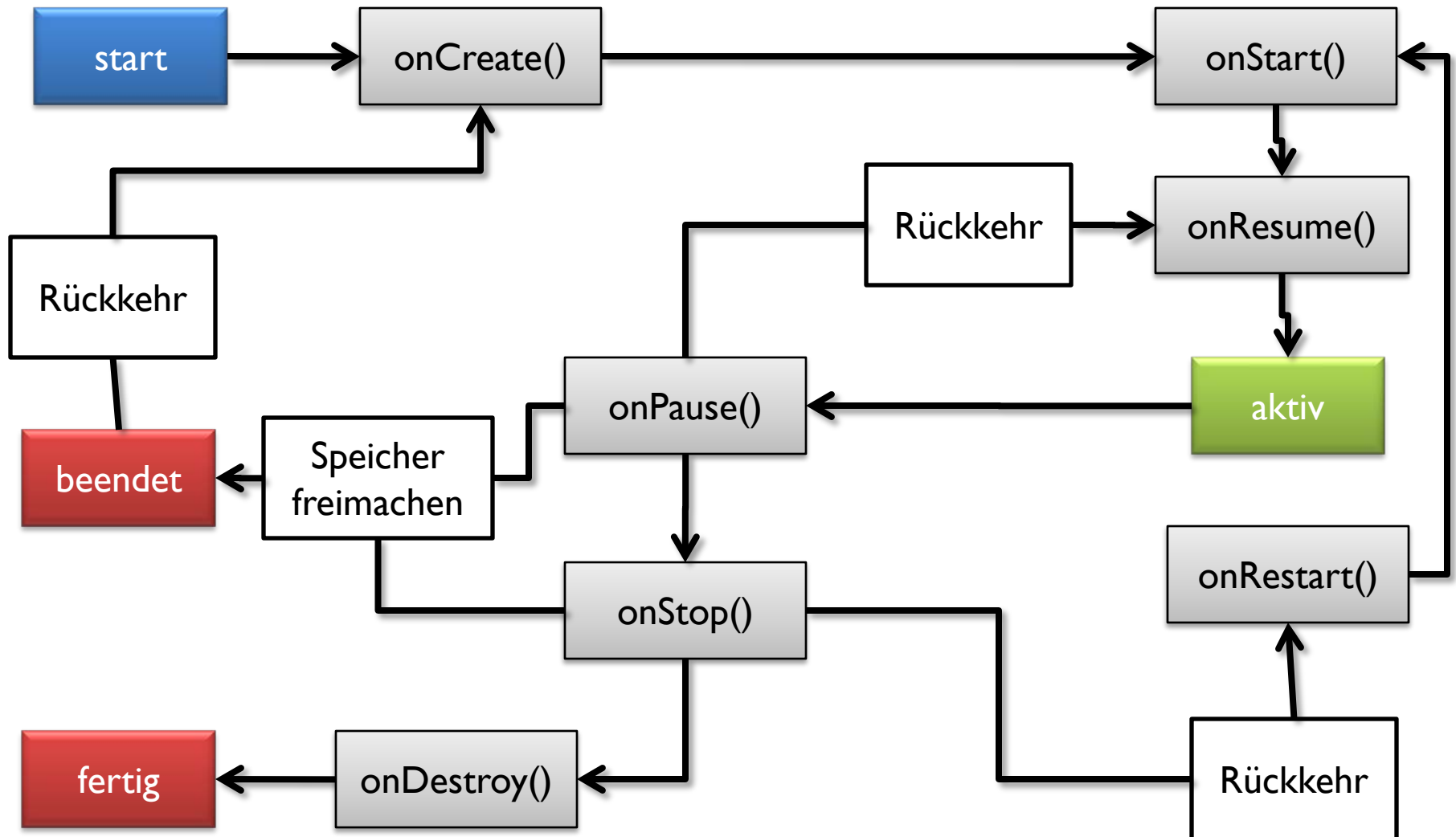
```
public class AME10 extends Activity {  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
  
        setContentView(R.layout.ame10);  
  
        Button callBrowser =  
            (Button) this.findViewById(R.id.ame10_browser);  
  
        ...  
    }  
}
```

Die erste Android Applikation

- View erstellt (XML)
- Activity angebunden (Java)
- alle XML – Objekte haben eine Repräsentation aus Java – Objekten (Klasse R, Views)

Praxis

Activity - Lifecycle



Aufruf einer fremden Activity

```
public class AME10 extends Activity {
    public void onCreate(Bundle savedInstanceState) {

        ...
        Button callBrowser =
            (Button) this.findViewById(R.id.ame10_browser);
        callBrowser.setOnClickListener(new OnClickListener() {

            public void onClick(View v) {
                AME10.this.startActivity(
                    new Intent(Intent.ACTION_VIEW,
                        Uri.parse("http://www-lehre.inf.uos.de/ame10")
                    )
                );
            }
        });
    }
}
```

Intents

- systeminterne Nachrichten
- Kommunikation zwischen einzelnen Android – Komponenten
- folgen dem Sicherheitsprinzip
- Kommunikation zwischen einzelnen Sandboxes
- Implizite Adressierung möglich
- Aufgabe wird an die beste verfügbare Anwendung weitergeleitet

Intent-Filter

```
<intent-filter>
    <action
        android:name="android.intent.action.VIEW" />
    <category
        android:name="android.intent.category.DEFAULT" />
    <data
        android:scheme="http"/>
</intent-filter>
```

`android:mimetype` (z.B. `image/jpeg`)

`android:host`

`android:path`

`android:port`

Intent-Vermittlung

- vergleiche Action-Parameter mit action-Tag im Intent-Filter
- alle Kategorien des Intents mit denen des Filters vergleichen
- URI mit data – Tags vergleichen
- bei mehreren Übereinstimmungen erscheint ein Auswahl - Dialog

Aufruf einer eigenen Activity

```
public class AME10 extends Activity {
    public void onCreate(Bundle savedInstanceState) {

        ...
        Button callHello =
            (Button) this.findViewById(R.id.ame10_helloandroid);
        callHello.setOnClickListener(new OnClickListener() {

            public void onClick(View v) {
                AME10.this.startActivityForResult(
                    new Intent(AME10.this, HelloAndroid.class),
                    HelloAndroid.CALL_HELLO);
            }
        });
    }
}
```

Abfragen von Activities

```
public class HelloAndroid extends Activity {  
  
    private static int calls = 0;  
    public static final String CALLS = "CALLS";  
    public static final int CALL_HELLO = 1000;  
  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.helloandroid);  
  
        calls++;  
        Intent data = this getIntent();  
        data.putExtra(CALLS, calls);  
        this.setResult(Activity.RESULT_OK, data);  
    }  
}
```

Verarbeitung von Intents

```
public class AME10 extends Activity {

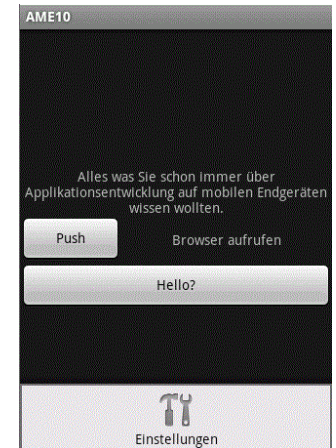
    private static final String TAG = "AME10";
    ...
    protected void onActivityResult(
        int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);

        if (requestCode == HelloAndroid.CALL_HELLO
            && resultCode == Activity.RESULT_OK) {

            Log.d(TAG,
                this.getResources().getString(R.string.count)
                + data.getExtras().getInt(HelloAndroid.CALLS)
            );
        }
    }
}
```

Menüs

```
public class AME10 extends Activity {  
    private static final int MENU_PREFERENCES = 1;  
    ...  
    public boolean onCreateOptionsMenu(Menu menu) {  
        menu.add(0, MENU_PREFERENCES, 0, R.string.preferences)  
            .setIcon(android.R.drawable.ic_menu_preferences);  
        return super.onCreateOptionsMenu(menu);  
    }  
  
    public boolean onOptionsItemSelected(int featureId, MenuItem item) {  
        if (item.getItemId() == MENU_PREFERENCES) {  
            Intent i = new Intent(this, Preferences.class);  
            this.startActivity(i);  
        }  
  
        return super.onOptionsItemSelected(featureId, item);  
    }  
}
```



Einstellungen

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

<PreferenceScreen
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical">

    <PreferenceCategory android:title="Allgemeine Einstellungen">
        <CheckBoxPreference
            android:title="Blau"
            android:key="@+string/makeblue"
            android:summary="Einen bisschen blau ist immer schön"
        />
    </PreferenceCategory>

</PreferenceScreen>
```



Einstellungen anbinden

```
public class Preferences extends PreferenceActivity {  
  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
  
        addPreferencesFromResource(R.layout.preferences);  
    }  
}
```

Das Manifest

- Basis der gesamten Applikation
- Rechteverwaltung über Permissions
- Verwaltung der Activities
- und zugehöriger Intent – Filter
- Informationen über Kompabilität
 - unterstützte Betriebssystemversionen
 - und Gerätekonfigurationen
- Einbindung von externen Bibliotheken

Praxis

Entwicklung

DEMO

Weiterführendes

Dialoge und Toasts

- Benutzerbenachrichtigung und –eingabe ohne extra Activity

- einfach

- Toast

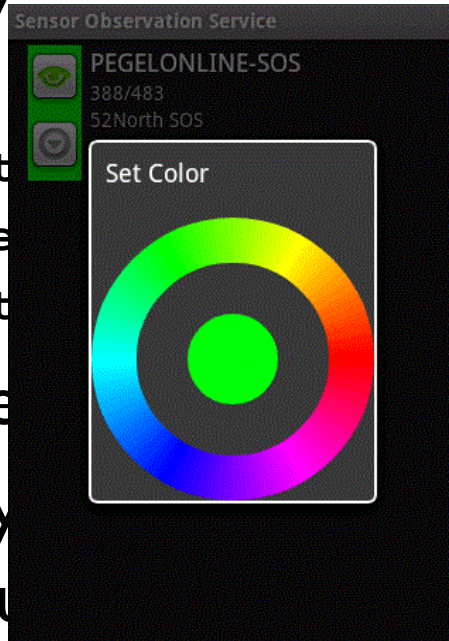
- Content

- Alert

- Dialoge

- Activity

Erzeugt



Styles und Themes

- einheitliches Design
 - durch Gruppen von Attributwerten (Styles)
 - und Formatvorlagen für Bildschirmfenster (Themes)
- werden durch XML – Files vordefiniert
- Android liefert Styles in `android:style` mit
- zusätzlich bessere Strukturierung durch `include` - Tags

Weiterführendes

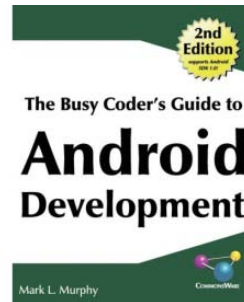
Performance

- Effizienz vor „Schönheit“
- Interfaces, Getter und Setter vermeiden
- direkte Variablenaufrufe
- Activity Lifecycle ausnutzen
- statische Variablen
- Konstanten

Weiterführendes

Literatur

- „Schlecht“:



- „Mittel“:



- A. Becker, M. Pant: Android: Grundlagen und Programmierung (dpunkt – Verlag, ISBN 978-3-89864-574-4)

- „Gut“:

- <http://developer.android.com>
- <http://www.anddev.org>

Zusammenfassung und Ausblick

- leistungsstarkes SDK
- effiziente Programmierung durch komponentenbasierten Aufbau
- vielfache Möglichkeiten zur Einbindung von Multimedia-Inhalten
- „Die Vorstellung ist das Limit“

