

# Einführung: Android Programmierung (Teil 2)

## UI (Details), Dateien, Services

David Tanzer

[business@davidtanzer.net](mailto:business@davidtanzer.net)  
<http://davidtanzer.net>

# Übersicht

- UI Details
- Dateien / Einstellungen
- Services

## Übersicht: UI Details

- UI Details
  - Layouts
  - Eigene Views
  - 2D Graphik
  - Intents / Broadcast Receivers
  - Dialoge
- Dateien / Einstellungen
- Services

## Layouts

- Am besten in XML definiert
- Für Activities oder Compound Controls
- Compound Controls:
  - Layout in XML definieren
  - Control – Klasse implementieren
- Demo ...

## Übersicht: UI Details

- UI Details
  - Layouts
  - Eigene Views
  - 2D Graphik
  - Intents / Broadcast Receivers
  - Dialoge
- Dateien / Einstellungen
- Services

## Eigene Views

- Klasse „View“ oder „SurfaceView“ ableiten
  - View: „Normale“ - Views (nur 2D Graphik, Vordergrundthread)
  - SurfaceView: 3D – Graphik, zeichnen aus Hintergrundthread
- `onMeasure()`: Überschreiben, um die Größe des Views zu setzen
- `onDraw()`: View zeichnen
- `onKeyDown()`, `onKeyUp()`, `onTouchEvent()`, ...: Benutzerinteraktion
- Keinen Speicher in diesen Methoden reservieren!
- Demo ...

## Übersicht: UI Details

- UI Details
  - Layouts
  - Eigene Views
  - 2D Graphik
  - Intents / Broadcast Receivers
  - Dialoge
- Dateien / Einstellungen
- Services

## 2D Graphik

- Color: 32 Bit, ARGB
- Paint: Enthält Farbe, Linientyp, Stil, ...
- Canvas: „Leinwand“, auf der gezeichnet wird
- Path: Enthält Vektor-Zeichenbefehle
- Demo ...

## Übersicht: UI Details

- UI Details
  - Layouts
  - Eigene Views
  - 2D Graphik
  - Intents / Broadcast Receivers
  - Dialoge
- Dateien / Einstellungen
- Services

## Intents / Broadcast Receivers (1/2)

- Intents
  - Activities und Services starten
  - Kommunikation zwischen Komponenten
  - Broadcasts (eingehender Anruf, SMS, ...)
- Intent – Filter
  - Um auf Intents zu hören
- Broadcast Receivers
  - Über Intent – Filter konfiguriert
- Demo ...

## Intents / Broadcast Receivers (2/2)

- Sub-Activities
  - Können an den Aufrufer ein Ergebnis zurückgeben
  - z.B. Auswahl des Benutzers aus einer Liste
- Demo ...

## Übersicht: UI Details

- UI Details
  - Layouts
  - Eigene Views
  - 2D Graphik
  - Intents / Broadcast Receivers
  - Dialoge
- Dateien / Einstellungen
- Services

## Dialoge

- Dialog – Klasse ableiten
- oder Activity mit „Dialog-Theme“ stylen

```
<activity android:name=".DialogActivity"  
    android:label="@string/dialog_activity_title"  
    android:theme="@android:style/Theme.Dialog"/>
```

- oder AlertDialogBuilder verwenden
- Demo...

## Übersicht: UI Details

- UI Details
- Dateien / Einstellungen
  - Preferences
  - Programmeinstellungen (Settings)
  - Statische Dateien (aus Ressourcen)
- Services

## Preferences

- View für Preferences in XML definieren
  - Für Eingabefelder `android:key="..."` setzen
- Activity von `PreferencesActivity` ableiten
- In `onCreate` **statt** `setContentView(...)` die Methode `addPreferencesFromResource(...)` aufrufen
- Lesen:
  - `PreferenceManager`
    - `.getDefaultSharedPreferences(context)`
    - `.getString(key, defaultWert);`
- Demo...

## Übersicht: UI Details

- UI Details
- Dateien / Einstellungen
  - Preferences
  - Programmeinstellungen (Settings)
  - Statische Dateien (aus Ressourcen)
- Services

## Programmeinstellungen (Settings) (1/3)

- SharedPreferences
  - Key/Value Paare
- Schreiben:

```
String name = „mySharedPreferences“;  
int mode = Activity.MODE_PRIVATE;  
prefs = getSharedPreferences(name, mode);  
editor = prefs.edit();  
editor.put...(...); //putBoolean, putString, ...  
//...  
editor.commit();
```

## Programmeinstellungen (Settings) (2/3)

### ■ Lesen:

```
String name = „mySharedPreferences“;  
int mode = Activity.MODE_PRIVATE;  
prefs = getSharedPreferences(name, mode);  
someVar = prefs.get...(...); //getBoolean, ...
```

## Programmeinstellungen (Settings) (3/3)

### ■ Instance State:

```
@Override  
public void onSaveInstanceState(Bundle state) {  
    state.putString(„key“, value);  
    super.onSaveInstanceState(state);  
}
```

```
@Override  
public void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    someVar = savedInstanceState.getString(„key“);  
}
```

## Übersicht: UI Details

- UI Details
- Dateien / Einstellungen
  - Preferences
  - Programmeinstellungen (Settings)
  - Statische Dateien (Ressourcen)
- Services

## Statische Dateien (Ressourcen)

- FileInputStream bzw. FileOutputStream

- z.B. für temporäre Dateien

- keine Pfadangabe möglich, nur lokale Dateien

- `FileOutputStream fos = openFileOutput(...);`

- `FileInputStream fis = openFileInput(...);`

- SD-Karte (spezielle Permission benötigt):

```
File root =
```

```
    Environment.getExternalStorageDirectory();
```

```
if (root.canWrite()) {
```

```
    File file = new File(root, "file.txt");
```

```
    FileWriter filewriter = new FileWriter(file);
```

# Übersicht

- UI Details
- Dateien / Einstellungen
- Services

## Services

- Erstellen: Klasse „Service“ ableiten
  - `onCreate`, `onBind` und `onStart` überschreiben
  - Muss im Manifest bekanntgegeben werden
- Starten: Explizit über Service-Klasse oder implizit über „Action“:
  - `startService(new Intent(MyService.MY_ACTION));`
  - `startService(new Intent(this, MyService.class));`
- Services können an Activities gebunden werden (`onBind`)
  - Activity kann dann Methoden des Service aufrufen

## Hintergrund - Threads

- `java.lang.Thread` und `java.lang.Runnable`
- Synchronisation mit GUI – Thread:

```
private Handler handler = new Handler();  
private Runnable runInUiThread =  
    new Runnable {...};  
private void methodInBackgroundThread() {  
    handler.post(runInUiThread);  
}
```

## Toasts

- Toasts: Transiente Dialog-Boxen

- Nicht Modal und kein Fokus
- Nur einige Sekunden sichtbar

- Erzeugen:

```
Toast toast =
```

```
    Toast.makeText(context, message, duration);
```

```
toast.show();
```

- Können durch eigene Layouts angepasst werden

## Notifications

- Benachrichtigungen, die den Benutzer nicht unterbrechen
  - z.B. neue SMS, Roaminginformationen, eigene Nachrichten
- Können folgende Aktionen ausführen:
  - Icon im Status-Bar anzeigen
  - Zusatzinfos im erweiterten Status-Bar-Fenster anzeigen
  - LED aufleuchten lassen
  - Vibrieren bzw. Klingeltöne und andere Medien spielen
- `NotificationManager` ist ein `System Service`
- Methode `notify(...)` aufrufen und eine `Notification` übergeben

# Fragen? - Fragen!



Fragen?

**Vielen Dank!**

[business@davidtanzer.net](mailto:business@davidtanzer.net)