Android - Layout einer Benutzeroberfläche

Hinweise zur Gestaltung der Bildschirmseite

- Eine Bildschirmseite sollte eine Aufgabe abbilden.
- Auf einer Bildschirmseite befinden sich nur, die für die Aufgabe benötigten Elemente.
- Die wichtigsten Informationen werden zuerst präsentiert.
- Verzicht auf horizontales Scrollen der Bildschirmseite.
- Vermeidung von vertikalen Bildschirmlaufleisten durch Teilung der Aufgabe.

Elemente auf einer Bildschirmseite

- Das wichtigste Element wird zuerst präsentiert.
- Menüs befinden sich immer am oberen Rand einer Bildschirmseite.
- Elemente kleben nicht an einander. Nutzen Sie Außenabstände (margin) zwischen den einzelnen Elementen.
- Die Größe der Elemente wird so groß wie nötig und so klein wie möglich gewählt. Symbole können klar erkannt werden. Text kann gut gelesen werden. Antippbare Elemente können auch mit großen Fingern sicher gedrückt werden.

Formulare auf einer Bildschirmseite

- Formularfelder werden von oben nach unten abgearbeitet. Das wichtigste Feld sollte ganz oben stehen.
- Formularfelder werden mit einem Stichwort als Hilfe beschriftet.
- Felder, die Text anzeigen, sollten Innenabstände (padding) nutzen.

Sonstige Hinweise

- Aufwendige Berechnungen etc. werden im Hintergrund ausgeführt.
- Die Zurück-Schaltfläche ruft die vorherige Bildschirmseite auf, egal wofür diese in einer Activity sonst noch genutzt werden kann.
- Von einem Anwender gestartete Activities werden nicht durch Benachrichtigungen etc. unterbrochen.

Guidelines im Internet

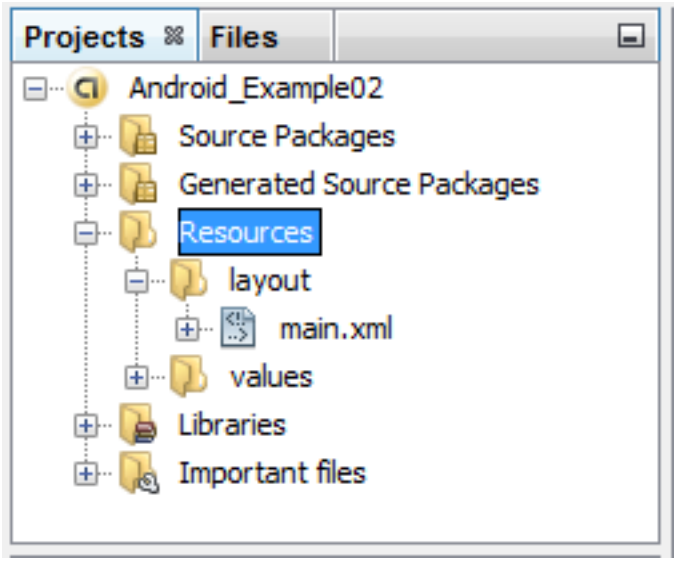
- http://developer.android.com/guide/topics/ui/index.html
- http://developer.android.com/design/index.html
- http://www.androidauthority.com/android-design-guidelines-514918/
- http://usabilitygeek.com/official-usability-user-experience-user-interface-guidelines-from-companies/
- http://t3n.de/news/wichtigsten-design-guidelines-ios-466152/design-guidelines-ios-android-2/

Layout-Dateien

- Für jede Bildschirmseite wird eine XML-Datei für die Definition des Layoutes erstellt.
- Die Dateien werden in dem Verzeichnis *res / layout* abgelegt.
- Sobald ein neues Android-Projekt in NetBeans angelegt wird, wird die Layout-Datei *main.xml* automatisiert in dem Verzeichnis angelegt.

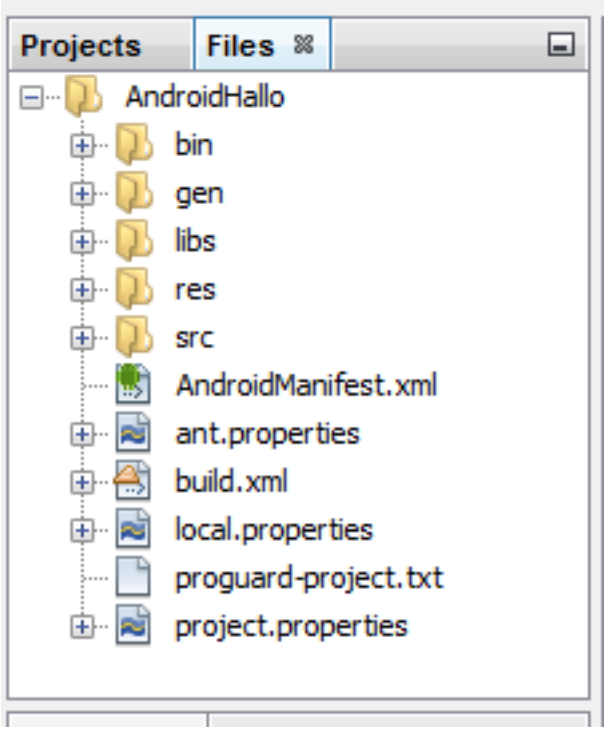
Layout-Datei im Projekt-Explorer öffnen

- Die Registerkarte *Projects* ist aktiv.
- Klick auf das Pluszeichen des Ordners *Ressources*.
- Klick auf das Pluszeichen des Ordners *layout*.
- Mausklick auf die gewünschte Datei in dem Ordner.
- Die Datei wird im Code-Fenster angezeigt.



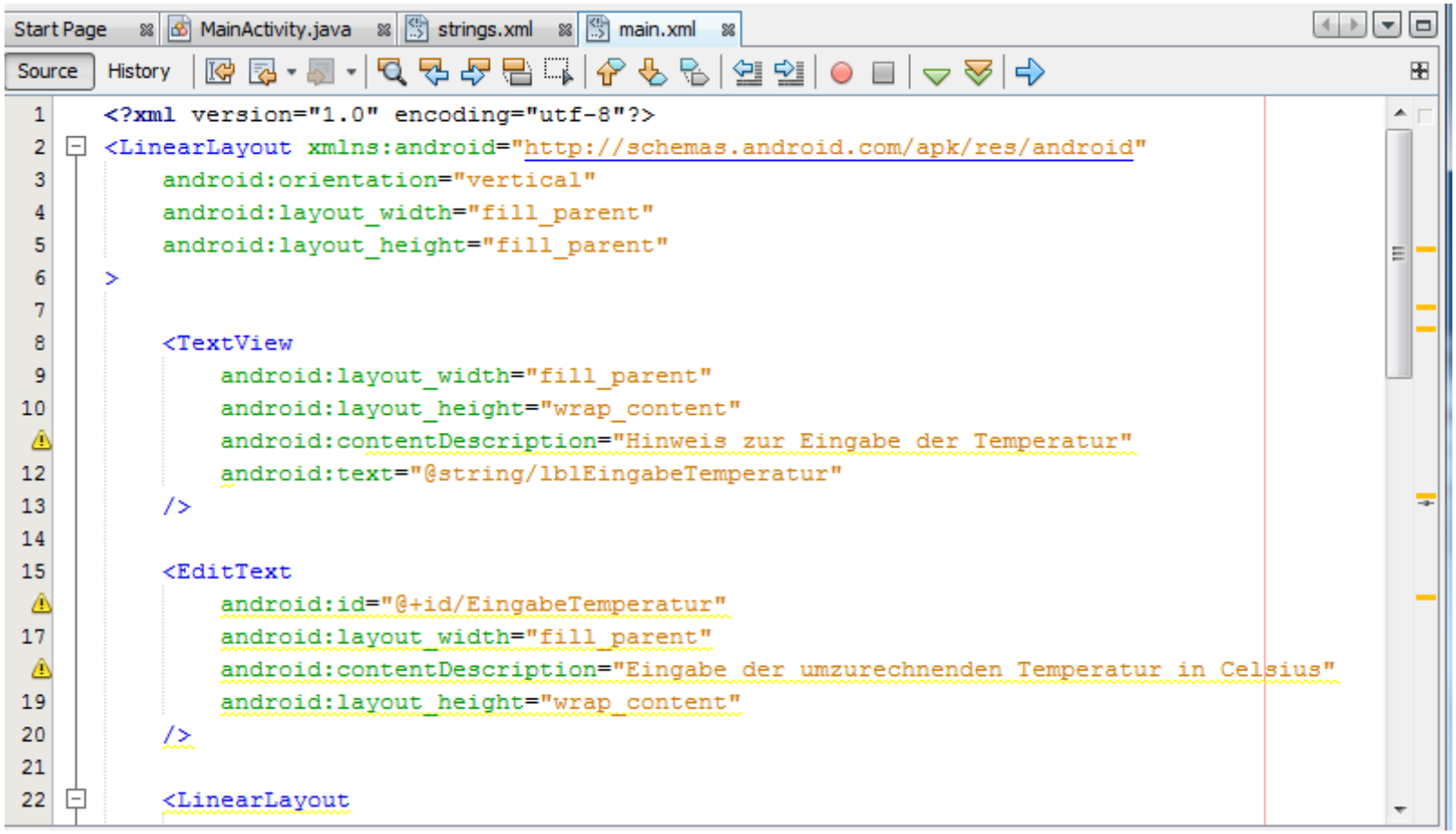
Layout-Datei im File-Explorer öffnen

- Die Registerkarte *Files* ist aktiv.
- Klick auf das Pluszeichen des Ordners *res*.
- Klick auf das Pluszeichen des Ordners *layout*.
- Mausklick auf die gewünschte Datei in dem Ordner.
- Die Datei wird im Code-Fenster angezeigt.



Leibniz Universität Hannover

Layout-Datei im Code-Fenster



Hinweise zum Codefenster

- Für jede Datei wird eine eigene Registerkarte angelegt.
- Der Dateiname wird im Reiter angezeigt.
- Durch einen Klick auf den Reiter wird die passende Registerkarte aktiviert.

Aufbau der XML-Datei

Prolog

Wurzelelement (LayoutView)

View-Elemente

XML-Elemente

- Start-Tag in XML: <element>. Ende-Tag in XML: </element>.
- Zusammenfassung von Start- und Ende-Tag: <element />.
- Bei einer Verschachtlung von Elementen darf der Start- und Ende-Tag nicht zusammengefasst werden.

Prolog

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

- Der Prolog beginnt mit <?xml und endet mit ?>.
- Die Zeile ist optional.
- Die genutzte Version wird angegeben (version="1.0").
- Die Zeichencodierung (encoding="utf-8") zum Speichern der XML-Datei wird festgelegt. Hier wird der Webstandard UTF-8 genutzt.
- Beiden Attributen werden die Werte als String übergeben.

Beispiel: XML-Elemente für Apps

Erläuterung

- Die Klasse View repräsentiert alle Elemente auf einer Benutzeroberfläche einer App. Views stellen ein Rechteck bereit, in dem gezeichnet oder mit dem Benutzer interagiert werden kann.
- Die Klasse ViewGroup ordnet View-Elemente auf einer Benutzeroberfläche an.
- Die Klasse TextView stellt Steuerelemente zur Interaktion mit dem Benutzer bereit.

Hierarchie der Elemente

- Voraussetzung: Das Android Projekt wird als Emulation auf dem Bildschirm angezeigt.
- Das Android Projekt hat durch einen Klick auf die Bildschirmseite den Fokus bekommen.
- Klick auf *hierachyviewer.bat* im Ordner *Android – android-sdk – tools*.
- Klick im Hierarchy-Viewer auf den Eintrag der Aktivität der zu untersuchende Bildschirmseite.
- Klick auf die Schaltfläche *Load View Hierarchy*.
- Nach einer kurzen Weile erscheint die View-Hierachie.

Wurzel-Element

- Anordnung von View-Elementen.
- Einbettung von View-Elementen.
- Rahmen für die Positionierung von View-Elementen mit Hilfe der Klasse ViewGroup.

Lineares Layout als Wurzelelement

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



</LinearLayout>
```

Erläuterung

- Die View-Elemente werden mit Hilfe des Attributs android:orientation="vertical" vertikal und durch android:orientation="horizontal" horizontal angeordnet.
- Ein lineares Layout sollte nicht mehr als zwei Ebenen tief verschachtelt werden.

Verankerung der Elemente im Layout

```
<Button
        android:layout_weight="1"
/>
```

- Das Attribut android:layout_weight="1.0" legt bei zwei Elementen fest, dass sich die Elemente den Platz im Verhältnis 1:1 teilen. Andere Möglichkeit: 0.
- Das Attribut android:layout_gravity verankert die Elemente zum Beispiel oben (top), unten (bottom), links (left), rechts (right).

Relatives Layout als Wurzelelement

```
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >



</RelativeLayout>
```

Erläuterung

- Die Elemente werden relativ zu anderen Elementen oder den Rändern des Layouts angeordnet.
- Umsetzung von komplexen Layouts ohne Verschachtlung.
- Jedes Element in einem relativen Layout muss eine ID haben.

Positionierung der Elemente im Layout (Beispiel)

```
<Button
        android:layout_below="@+id/EingabeTemperatur"
        android:layout_toRightOf="@+id/btnKelvin"
        android:layout_alignParentRight="true"
/>
```

- In diesem Beispiel wird die Schaltfläche unterhalb des Elements „EingabeTemperatur“ und rechts von dem Element „btnKelvin“ positioniert.
- Die Schaltfläche wird am rechten Rand ausgerichtet.

Positionierung der Elemente im Layout

```
<Button
        android:layout_below="@+id/EingabeTemperatur"
        android:layout_toRightOf="@+id/btnKelvin"
        android:layout_alignParentRight="true"
/>
```

- Die Elemente werden mit Hilfe von Attributen in Abhängigkeit des Eltern-Elements oder eines anderen Widget positioniert.
- Siehe http://developer.android.com/reference/android/widget/RelativeLayout.LayoutParams.html).

Laden des Layouts

```
public class MainActivity extends Activity
{
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
}
```

Erläuterung

- Mit Hilfe der Methode `setContentView()` wird das Layout geladen.
- Der Methode wird die Ressourcen-ID der Layout-Datei übergeben.
- Ressourcen-IDs werden automatisch generiert und in der Datei *R.java* gespeichert.
- Für Layout-Dateien ist die ID in der Klasse `R.layout` gespeichert.

Hinweise

- Im ersten Schritt sollte die Größe des Widget festgelegt werden.
- Im zweiten Schritt sollte das Widget in Abhängigkeit des Wurzelelements positioniert werden.

XML-Element

<RelativeLayout >

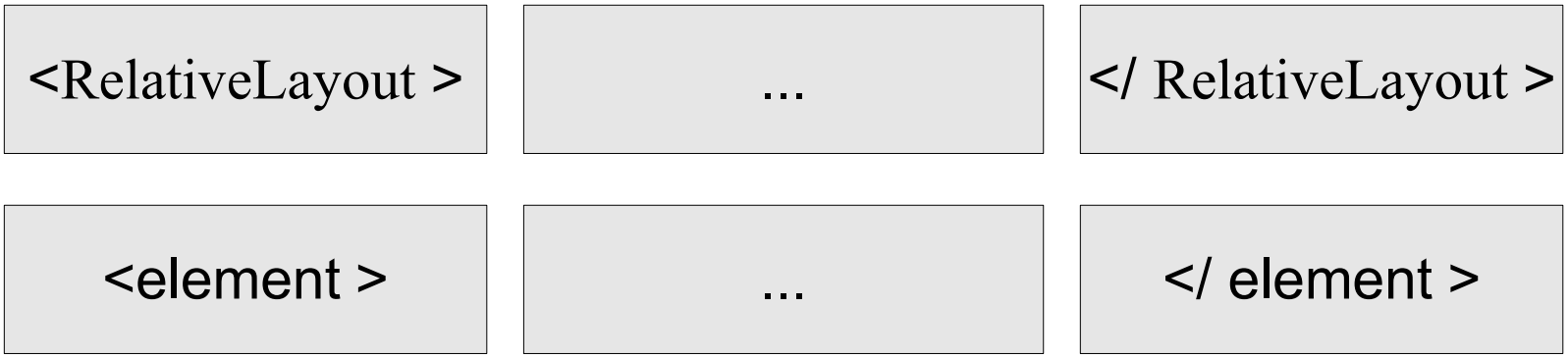
...

</ RelativeLayout >

<Button />

- Jedes Element hat einen eindeutigen Namen.
- Der Name drückt die Verwendung des Elements aus.
- Der Name wird im Start- und Ende-Tag wiederholt.

Start- und Ende-Tag



- Start-Tag in XML: <element>. Der Element-Name wird in spitze Klammern gefasst.
- Ende-Tag in XML: </element>. Der Tag beginnt mit der spitzen Klammer und einem Schrägstrich. Zwischen der Klammer und dem Schrägstrich darf kein Leerzeichen stehen.
- Zwischen dem Start- und Ende-Tag werden untergeordnete Elemente oder Ressourcen definiert.

Zusammenfassung von Start- und Ende-Tag

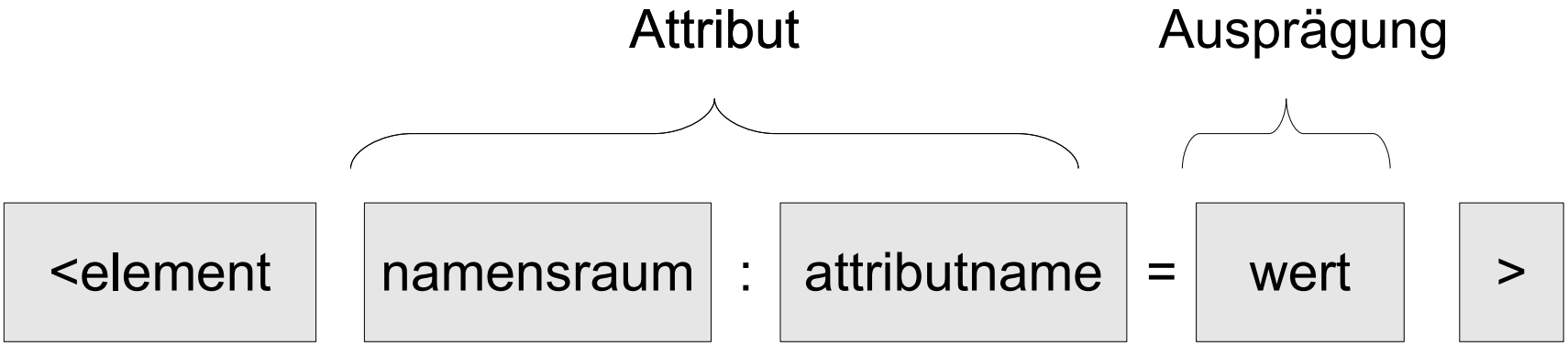
<Button />

<element />

- Das Element beginnt mit der spitzen Klammer und endet mit dem Schrägstrich plus spitze Klammer.
- Unterhalb des definierten Elements werden keine weiteren Elemente angeordnet. Das Element definiert keine Ressource.

Leibniz Universität Hannover

Attribute eines XML-Elemente



- Attribute eines XML-Elementes werden im Start-Tag definiert.
- Mit Hilfe von Attributen werden Eigenschaften eines XML-Elements beschrieben.
- Attribute können von dem Eltern-Element geerbt werden. Zum Beispiel die Attribute layout_width und layout_height sind Basis-Attribute, die jedes View-Element besitzt.

Leibniz Universität Hannover

Name eines Attributs

<TextView	...	:	layout_width
<element	...	:	attributname

- Der Name kennzeichnet eindeutig ein Attribut.
- Bitte beachten Sie die Groß- und Kleinschreibung.

Kategorisierung mit Hilfe des Namensraumes

<TextView	android	:	layout_width
<element	namensraum	:	attributname

- Die Attribute werden mit Hilfe von Kategorien zusammengefasst, um Namenskonflikte aufzulösen.
- Der Namensraum und der Name des Attributs werden durch den Doppelpunkt miteinander verbunden.
- Die Angabe des Namensraumes ist zwingend erforderlich.
- Der Namensraum wird im Wurzelelement des Layouts definiert (xmlns:android).

Beispiel für die Einbindung eines Namensraumes

```
<LinearLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
>
```

- Einbindung über das Attribut xmlns:android im Wurzelelement.
- Die angegebene URL dient der eindeutigen Identifizierung des Namensraumes.
- Wenn das Layout nicht korrekt geladen werden kann, wird ein Hinweis auf den fehlenden Namensraum eingeblendet.

Leibniz Universität Hannover

Ausprägung eines Attributs

<TextView	android	:	layout_width	=	"fill_parent"	>
<element	namensraum	:	attributname	=	"wert"	>

- Die Ausprägung spiegelt den Attributwert wieder.
- Die Ausprägung wird einem Attribut als String übergeben. Der String wird durch Anführungszeichen begrenzt.
- Der Typ des Wertes selber ist abhängig vom Attribut.

Größe eines Elements

```
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:layout_width="150dp"
```

- Die Angaben für die Breite und Höhe müssen für jedes Layout-Element gesetzt werden.
- Die Größe kann durch eine Ganzzahl plus der Maßeinheit als String angegeben werden.
- Für die Größe können vordefinierte Konstanten wie `wrap_content` oder `fill_parent` genutzt werden.

… in Abhängigkeit des Inhaltes

```
android:layout_height="wrap_content"
```

- Die Größe des Elements wird in Abhängigkeit des anzuzeigenden Inhaltes ausgewählt.
- Hinweis: Die Höhe eines Elements wird häufig in Abhängigkeit des Inhaltes ermittelt.

… in Abhängigkeit der Eltern

```
android:layout_width="fill_parent"
android:layout_width="match_parent"
```

- Die Größe des Elements wird in Abhängigkeit des nicht ausgefüllten Bereiches des Eltern-Elements ermittelt.
- Die Größe wird immer in Abhängigkeit des übergeordneten Elementes ermittelt.
- Ab Android API Level 8 wird die Konstante `fill_parent` genutzt.

Hinweise zu der Ausprägung „fill_parent“

- Das Wurzelelement füllt den gesamten Touchscreen aus.
- Untergeordnete Elemente füllen die Restbreite / Resthöhe des Eltern-Elements aus.
- Wenn mehr als ein Element in einer Zeile angeordnet ist, wird ein Fehler ausgegeben.

Maßeinheiten für die Größe eines Elements

```
android:layout_width="12dp"
android:layout_height="12sp"
```

- Die Größe wird als Ganzzahl angegeben.
- Der Maßgröße folgt die Maßeinheit. Zwischen Maßgröße und -einheit darf kein Leerzeichen stehen.

Maßeinheiten

- dp: Pixelwert, der an die Dichte des Ausgabegerätes angepasst wird.
- sp: Pixelwert, der an die Dichte des Ausgabegerätes angepasst wird. Die Schriftgröße wird entsprechend der Benutzereinstellung skaliert. Die Maßeinheit wird häufig für die Darstellung von Schrift genutzt.
- Weitere Möglichkeiten: in (Inch), mm (Milimeter), px (Pixelwert), pt (Punktgröße).

Hintergrundfarbe

```
android:background="#112233"
```

- Dem Attribut wird eine RGB-Farbe als String übergeben.
- Der String enthält eine Hexadezimal-Zahl. Die Hexadezimalzahl beginnt mit einem Hash-Zeichen.
- Immer zwei Ziffern stellen einen bestimmten Licht-Anteil der gewünschten Farbe dar.

Farben in Android

- Das RGB-Farbsystem addiert (mischt) Licht in den Farben Rot, Grün und Blau.
- Jede der drei Farbe wird in 256 Helligkeitsstufen unterteilt. Um so eine Farbe sich dem Wert „Weiß“ nähert, um so heller wird diese.
- Für die Farbanteile Rot, Grün und Blau kann jeweils ein hexadezimaler Wert von 00 (= 0) bis FF (= 255) eingegeben werden.

Leibniz Universität Hannover

Widgets

- Interaktion mit dem Benutzer.
- TextView. Anzeige von Text.
- EditText. Eingabe von Text durch den Benutzer.
- Button. Schaltfläche zum Starten von Aktionen.

Leibniz Universität Hannover

ID eines Widgets

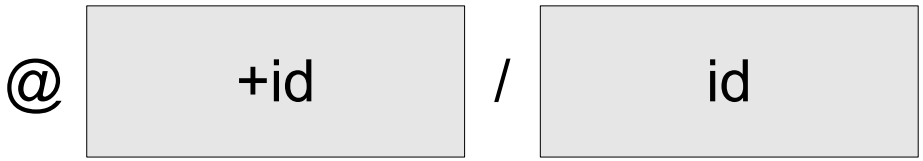
```
android:id="@+id/BeschriftungEingabefeld"
```

- Dem Attribut android:id wird eine Ressourcen-ID übergeben.
- Die ID wird in die Datei R.java eingetragen.

Hinweise zum Namen

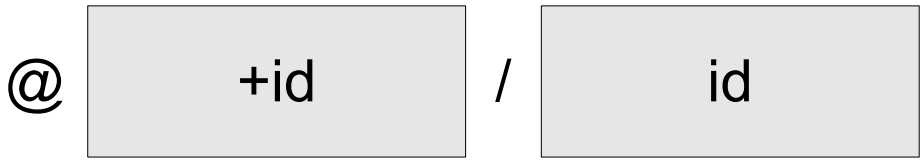
- Der Name muss eindeutig sein.
- Der Name beginnt immer mit einem Buchstaben.
- Es sollten nur die Buchstaben A ... Z, a .. z und die Ziffern von 0 bis 9 genutzt werden. Der Name darf keine Sonderzeichen enthalten.
- Wörter in zusammengesetzten Namen können durch den Unterstrich getrennt werden. Andere Möglichkeit: Kamel-Notation. Das erste Wort beginnt mit einem Kleinbuchstaben. Alle nachfolgenden Wörter beginnen mit einem Großbuchstaben.

Erzeugung einer Ressourcen-ID



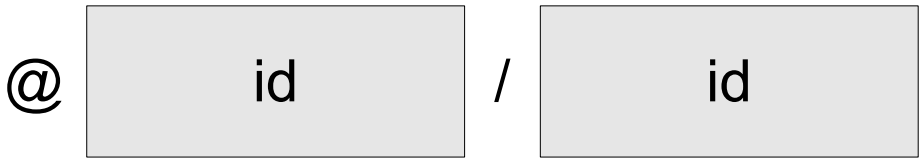
- Die ID eines Widgets oder Layouts wird in der Layout-Datei mit Hilfe des Attributs android:id definiert.
- Der Kasten „id“ kann durch einen beliebigen benutzerdefinierten Namen ersetzt werden. Dieser Name wird in der Datei *R.java* als Variablennamen für einen Integer-Wert angezeigt.
- +id erzeugt eine neue Ressourcen-ID.

At-Zeichen



- Der XML-Parser liest und zergliedert den String.
- Links vom Schrägstrich wird der Typ angegeben. In diesem Beispiel soll der Wert als ID interpretiert werden.
- Rechts vom Schrägstrich wird der Wert definiert, der entsprechend der Angabe links vom Schrägstrich vom Parser gelesen wird.

Hinweis auf eine Ressourcen-ID



- Der Kasten „id“ wird durch einen Variablennamen aus der Datei *R.java* ersetzt.
- id kennzeichnet eine vorhandene Ressourcen-ID. Es wird auf eine definierte Ressoucen-ID in der Datei *R.java* verwiesen.

Leibniz Universität Hannover

Außenabstände von Widgets

```
android:layout_marginTop="10dp"
android:layout_marginBottom="10dp"
android:layout_marginLeft="10dp"
android:layout_marginRight="10dp"
```

- Abstand zwischen den eingebetteten Elementen.
- Der Zwischenabstand wird immer mit der Hintergrundfarbe des Containers, in dem es eingebettet ist, ausgefüllt.

Innenabstände

```
android:paddingLeft="10dp"
android:paddingRight="10dp"
android:paddingTop="5dp"
android:paddingBottom = "5dp"

android:padding="10dp"
```

- Abstand zwischen dem Rand und dem Inhalt des Widgets.
- Hinweis: Bevor der anzuzeigende Text definiert wird, sollten die Größe, Position, die Außen- und Innenabstände definiert werden.

Anzeige von Text

- android:text.
- Der anzuzeigende Text im TextView.
- Der Text im Feld EditText.
- Die Beschriftung einer Schaltfläche (Button).

Hinweise für den Benutzer

- android:hint bei dem Widget EditText. Wenn das Textfeld leer ist, wird der Hinweis angezeigt.
- android:contentDescription. Der Text wird gesprochen, wenn der Nutzer über das Widget mit der Maus fährt.

Schriftgröße und -farbe in Bezug auf „android:text“

- android:textSize. Die Schriftgröße wird angepasst.
- android:textColor. Die Schriftfarbe wird als RGB-Farbe angegeben.

Hinweise zur Schriftgröße

```
android:textSize="12sp"
```

- Die Schriftgröße sollte immer in der Maßeinheit sp eingegeben werden.
- Die Schriftgröße ist bei dieser Maßeinheit abhängig von der Auflösung und der Pixeldichte des Ausgabegerätes. Die Textanzeigeoptionen des Benutzers werden beachtet.

Einschränkung der Tastatur bei InputText

```
android:inputType="numberDecimal"
```

- Spezifizierung der virtuellen Tastatur.
- Was darf der Benutzer eingeben? Welche Tasten sind erlaubt?
- Der Wert `numberDecimal` erlaubt nur die Eingabe von Dezimalzahlen.
- Siehe: http://developer.android.com/guide/topics/ui/controls/text.html.