PROGRAMMIEREN

Android SDK - Einrichten und benutzen

2 Entwicklungsumgebung einrichten

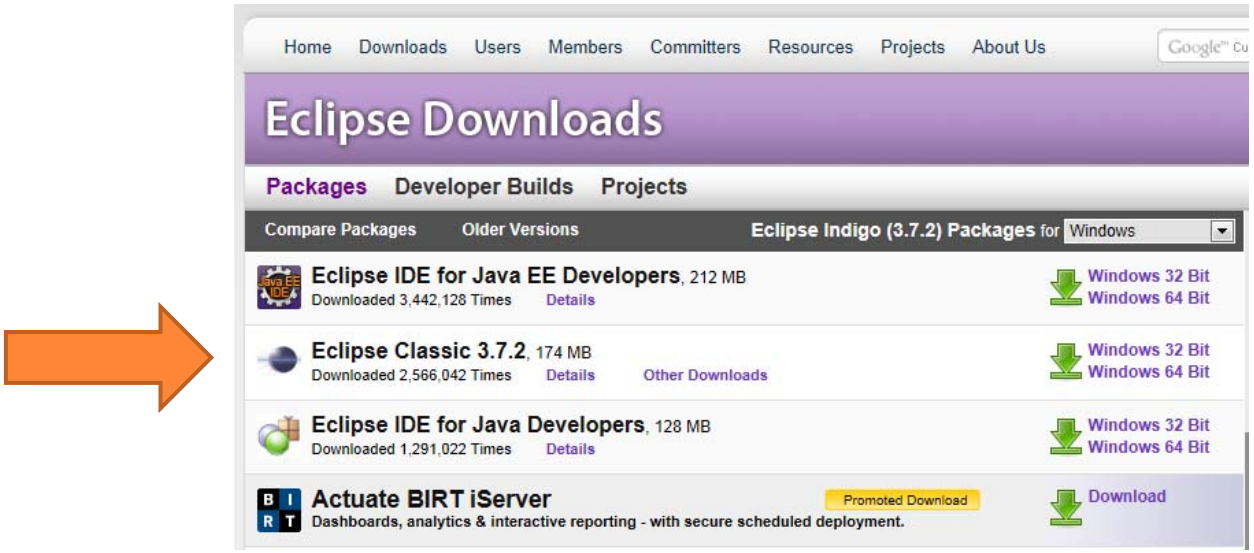
Komponenten

- Das *Android Software Development Kit* (SDK) ist das Hauptwerkzeug

- Am besten wird das Android SDK in Verbindung mit *Eclipse* verwendet. Die *Android Develpment Tools* (ADT) sind ein sehr leistungsfähiges Plug-in für Eclipse.

Installation - Eclipse

- Download der Eclipse Classic Version von http://www.eclipse.org/downloads/



Ausschnitt der Download-Seite

- Heruntergeladenes zip-File in den Zielordner entpacken.

Android SDK

- Inhalt des SDKs
  - Emulator
  - Tools
  - Plattformen (= bekannte Android-Versionen z.B. 2.3.x „Gingerbread“)
  - Dokumentation
  - Beispiele

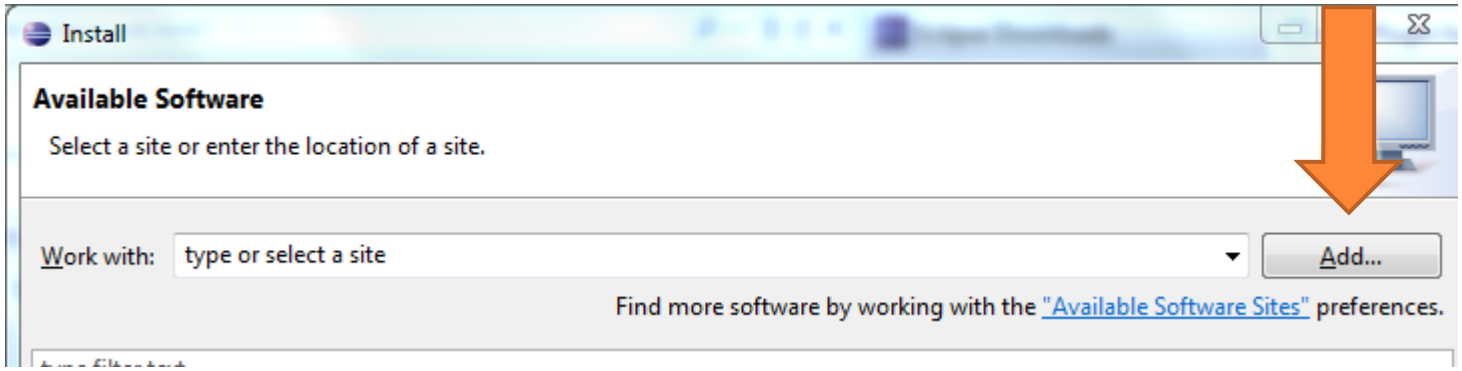
Installation – Android SDK

- Schritt 1: SDK Starter Package herunterladen und installieren.
  Quelle: http://developer.android.com/sdk/index.html

- Das SDK enthält nach der Installation weder das ADT (Eclipse Plug-In) noch die einzelnen Plattformen

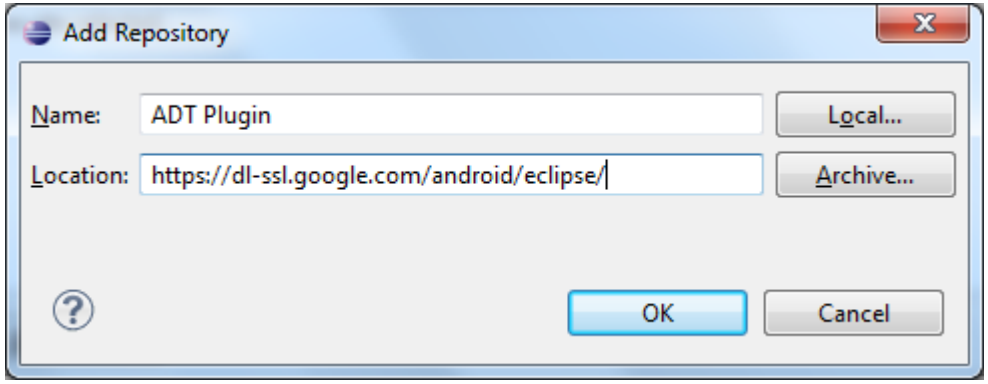
Installation – Android SDK

- Schritt 2: ADT installieren
  1. Eclipse als Administrator starten
  2. Menü-Eintrag *Help* > *Install New Software*....
  3. Im Dialog *Install* den *Add*-Button anklicken.



Installation – Android SDK

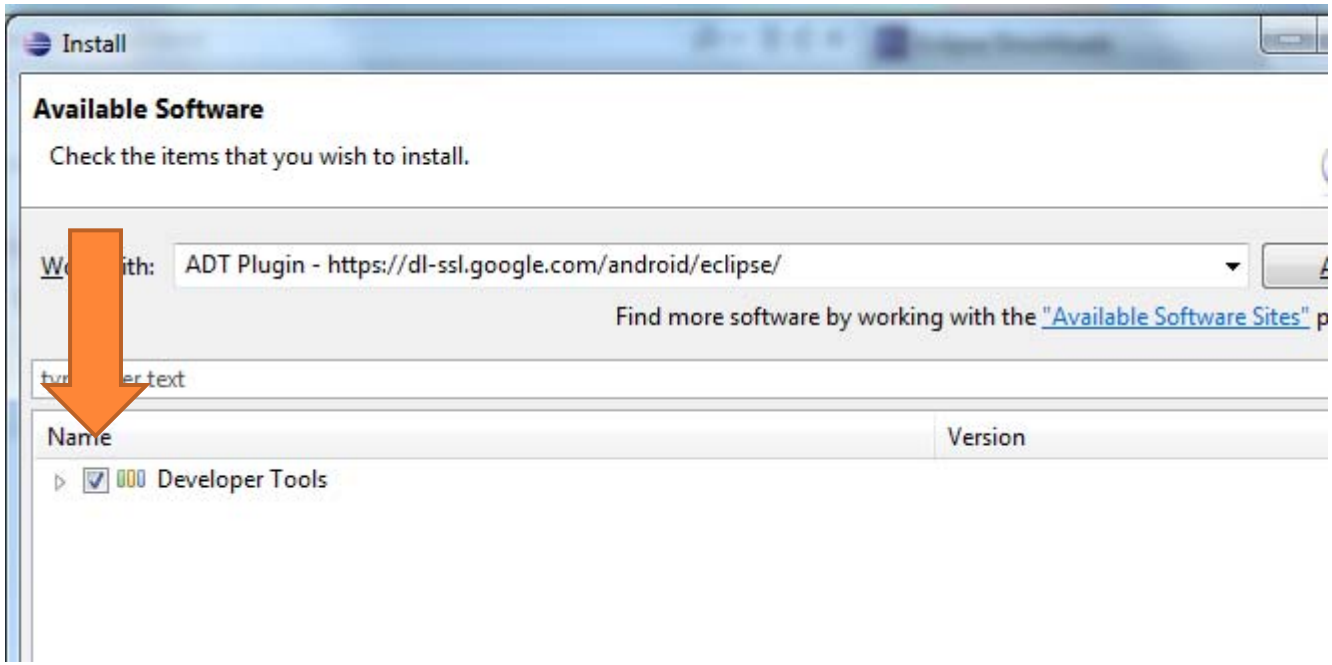
(4) Im Dialog *Add Repository* den Namen ADT Plugin und die folgende Location eingeben:
https://dl-ssl.google.com/android/eclipse/



(5) *OK* Button klicken

Installation – Android SDK

(6) Im Dialog *Available Software* die Checkbox vor Developer Tools aktivieren und den *Next*-Button drücken.



Installation – Android SDK

(7) Im nächsten Dialog werden die zu ladenden Tools angezeigt. *Next*-Button klicken.

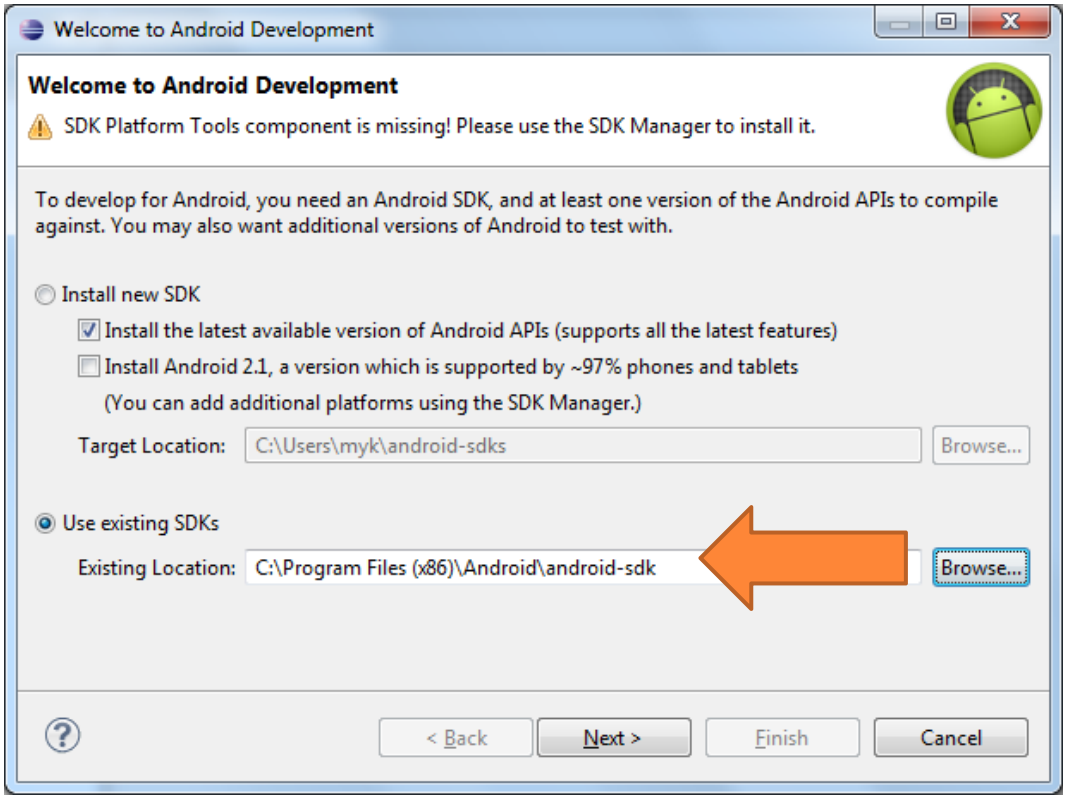
(8) Im nächsten Dialog Lizenzvereinbarung akzeptieren und *Finish* drücken.

(9) Eclipse neu starten.

Installation – Android SDK

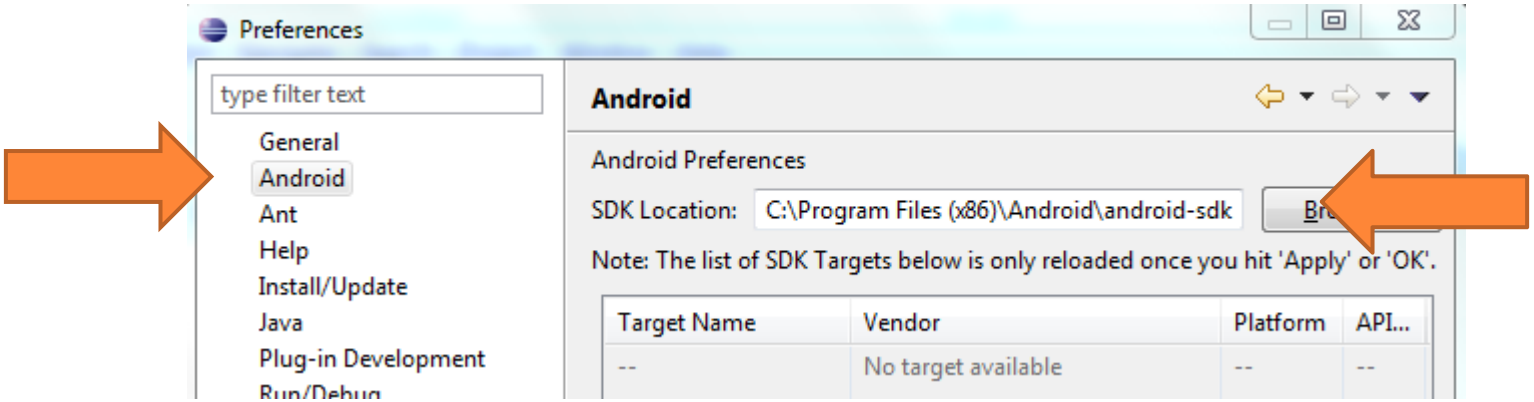
- Schritt 3: ADT konfigurieren

  Nach Eclipse Neustart im erscheinenden Begrüßungsdialog den Pfad zum SDK angeben.



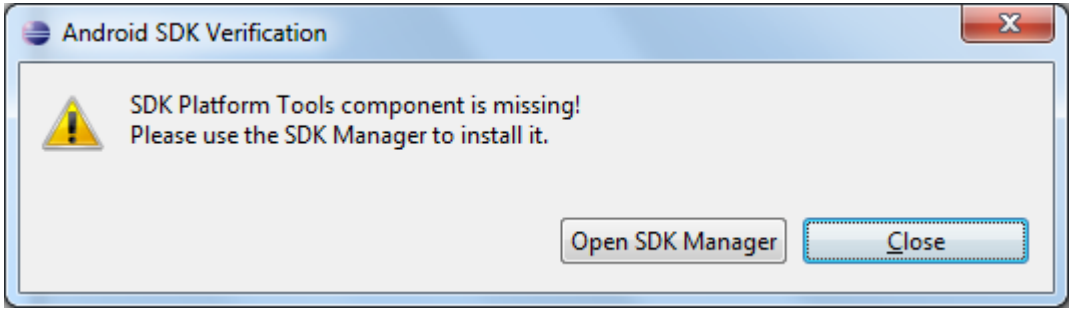
Installation – Android SDK

- ADT konfigurieren: Alternative, falls der Abfragedialog nach dem Start von Eclipse nicht automatisch erscheint.
  (1) Über Menüeintrag *Window* > *Preferences*… den Einstelldialog öffnen.
  (2) Im linken Panel *Android* auswählen.
  (3) Pfad zum SDK eingeben.



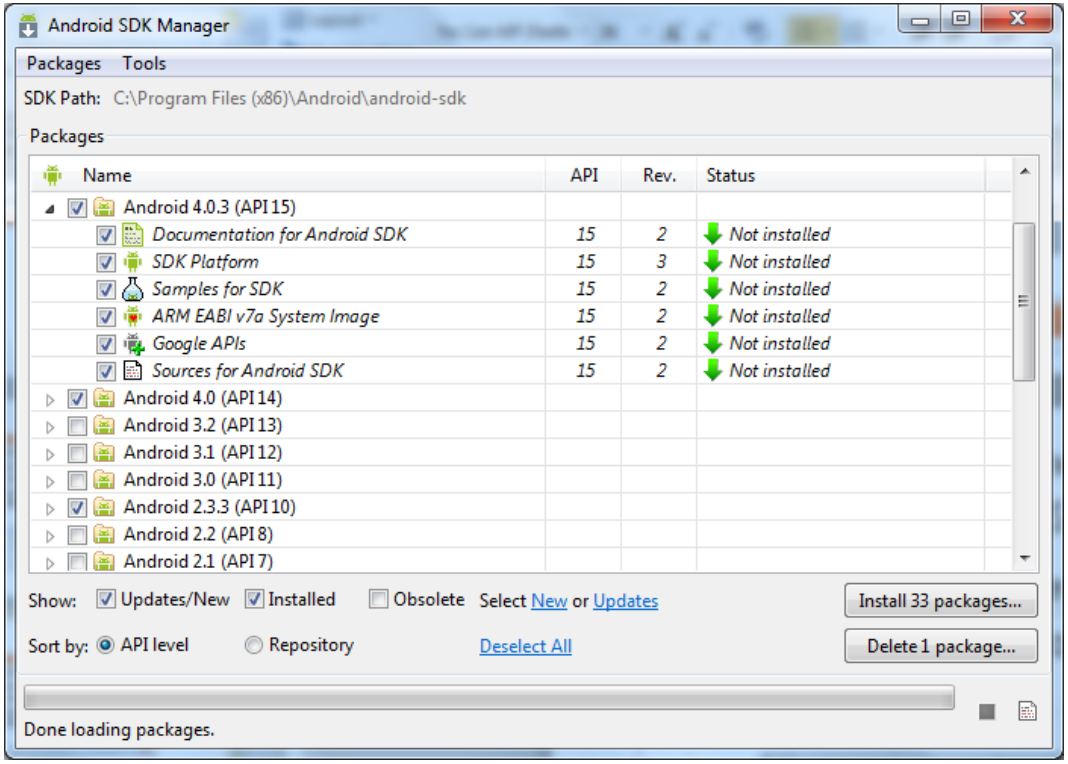
Installation – Android SDK

- Schritt 4: Plattformen installieren
  - Die Zielplattformen sind nicht im SDK enthalten. Die können mit dem SDK Manager installiert werden.
  - Nach Angabe des SDK-Pfads wird man von einem Eclipse-Dialog darauf hingewiesen



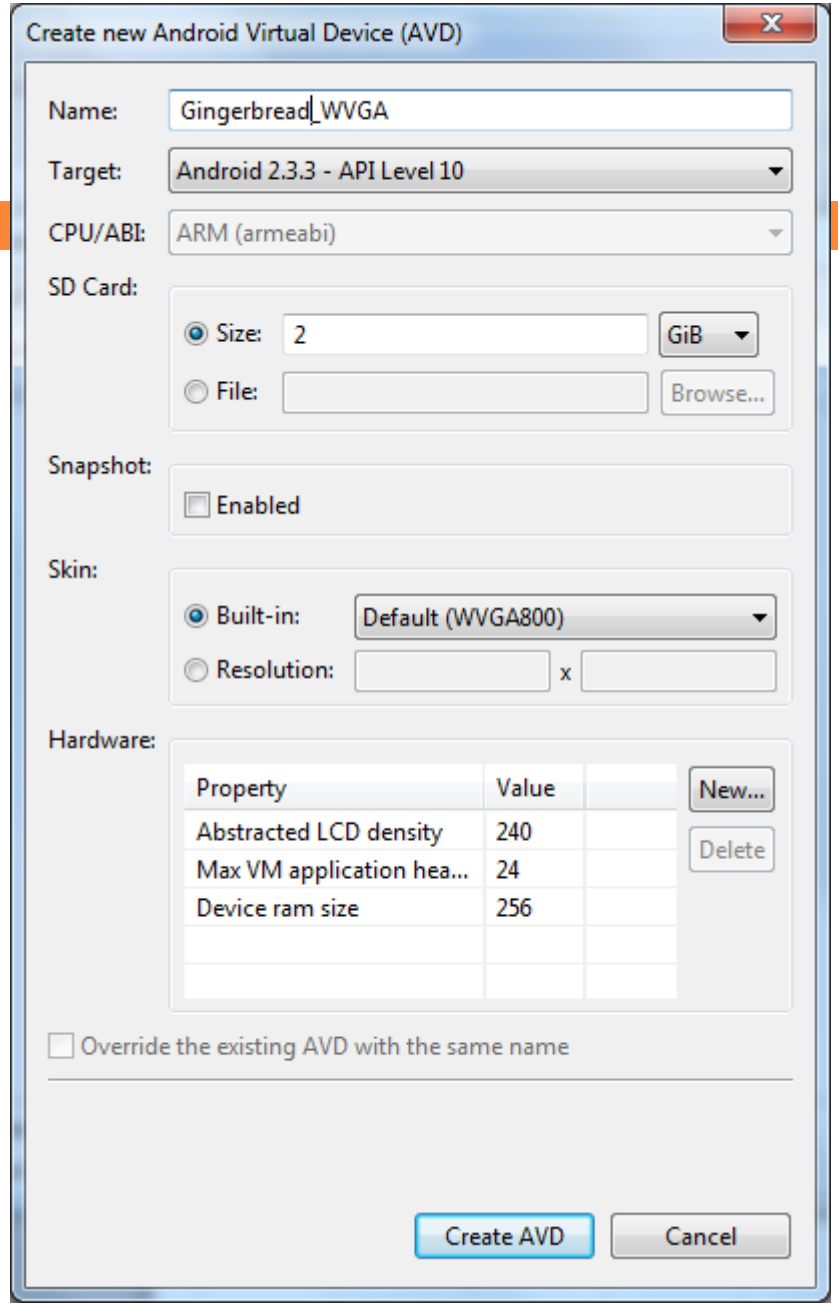
Installation – Android SDK

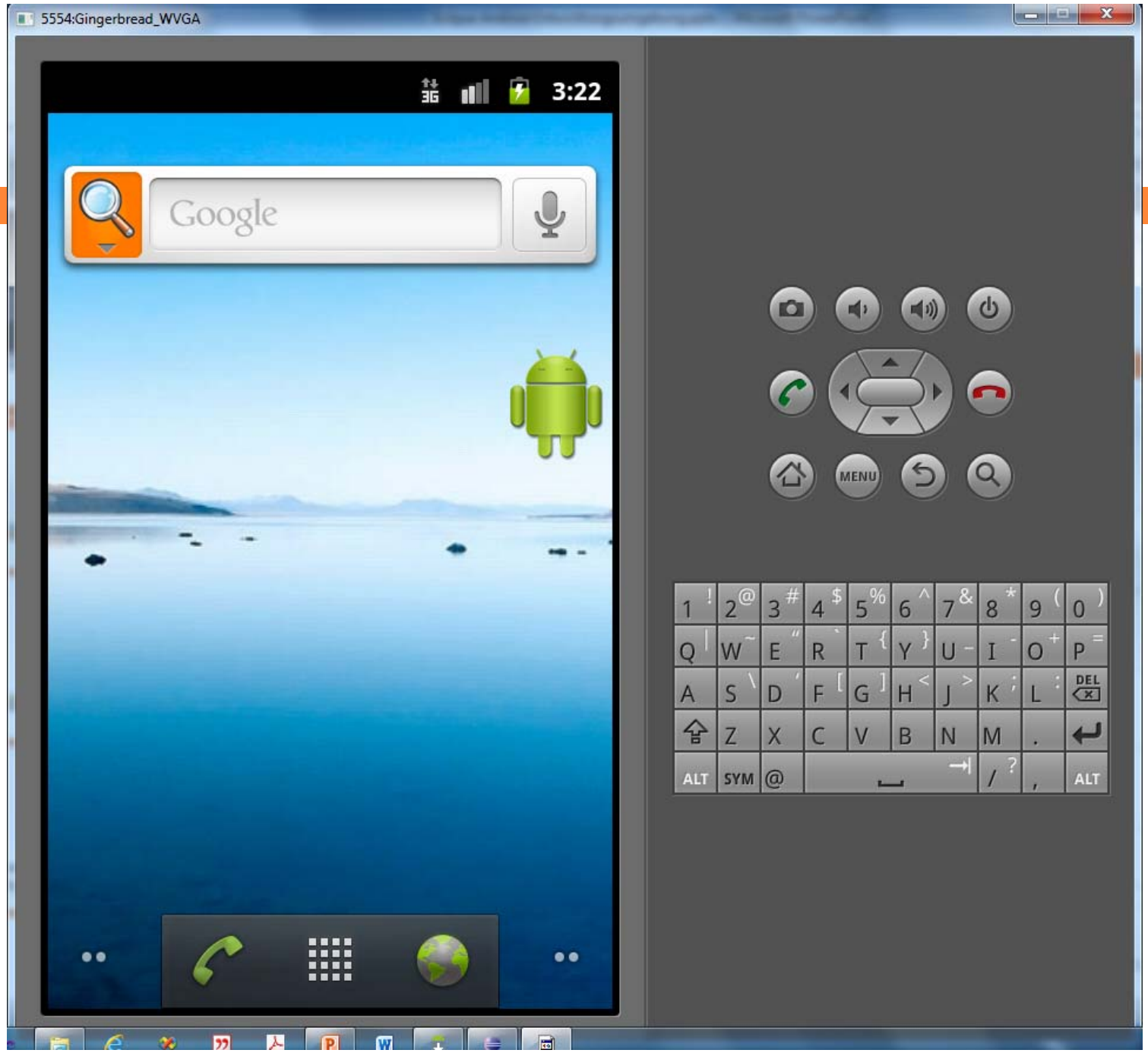
- Den SDK Manager öffnen.
- Die Plattformen 4.0.3 und 2.3.3 wählen und installieren



Installation – Android SDK

- Schritt 5: *Android Virtual Devices* (AVDs) erzeugen
  (1) Im Android SDK Manager den Menüeintrag *Tools* > *Manage AVDs…* auswählen.
  (2) Im Dialog *Android Virtual Device Manager* den Button *New…* klicken.
  (3) Neues Device anlegen wie auf der nächsten Folie beschrieben.
  (4) Neues Device anschließend im Dialog *Android Virtual Device Manager* starten. Das Ergebnis sollte so aussehen wie auf der übernächsten Folie.


Create new Android Virtual Device (AVD)
Name:
Gingerbread_WVGA
Target:
Android 2.3.3 - API Level 10
CPU/ABI:
ARM (armeabi)
SD Card:
Size:
2
GiB
File:
Browse...
Snapshot:
Enabled
Skin:
Built-in:
Default (WVGA800)
Resolution:
x
Hardware:
Property
Value
New...
Abstracted LCD density
240
Delete
Max VM application hea...
24
Device ram size
256
Override the existing AVD with the same name
Create AVD
Cancel


5554:Gingerbread_WVGA
3:22
Google
MENU

Hello World

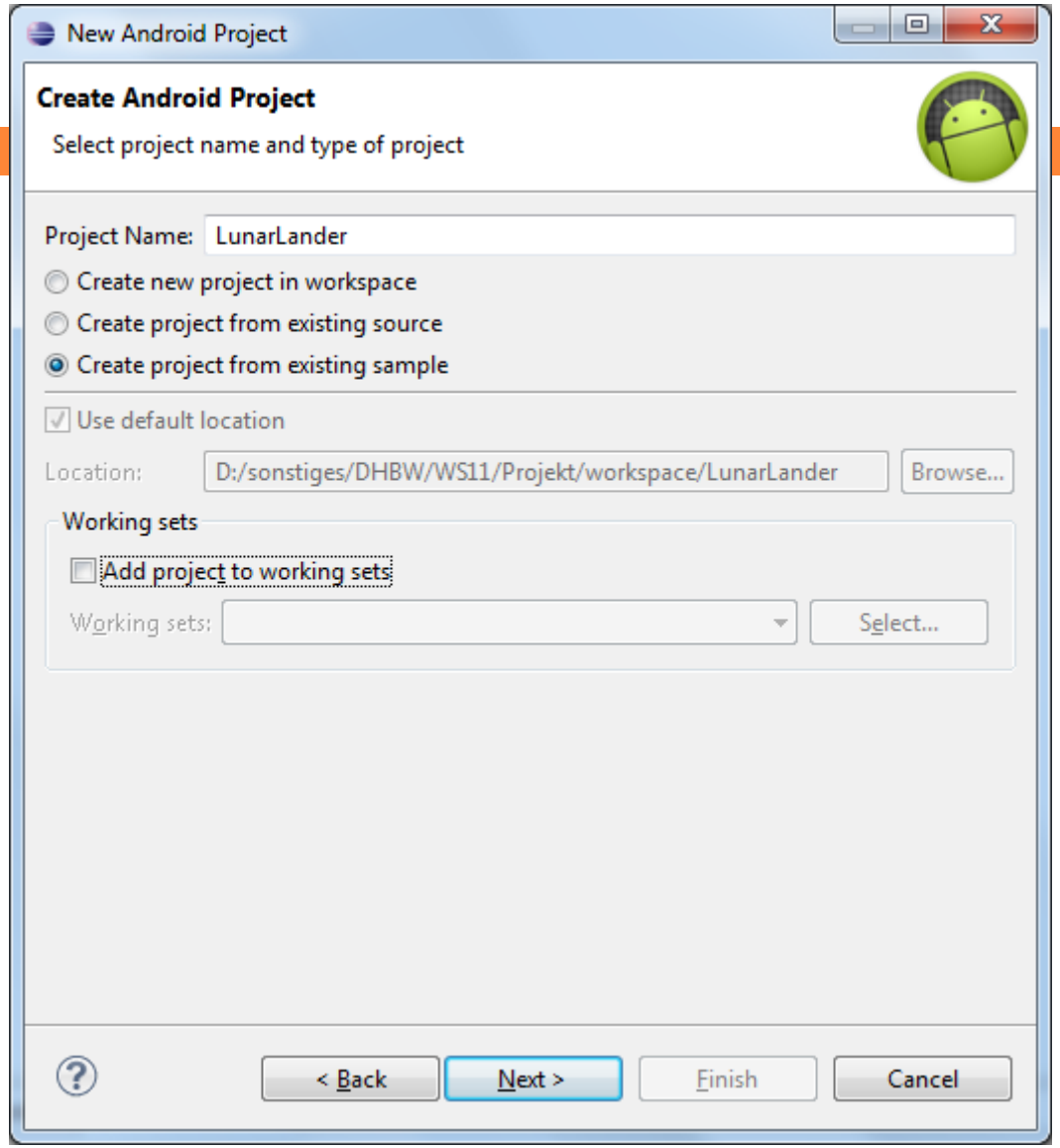
Erste Schritte mit Eclipse/Android

Erste Anwendung starten

(1) In Eclipse den Menüeintrag *File* > *New* > *Project…* auswählen

(2) Im Dialog *New Project* den Eintrag *Android* > *New Android* Project auswählen und den Button *Next* klicken.

(3) Projekt wie auf der nächsten Folie gezeigt anlegen und den Button *Next* klicken.

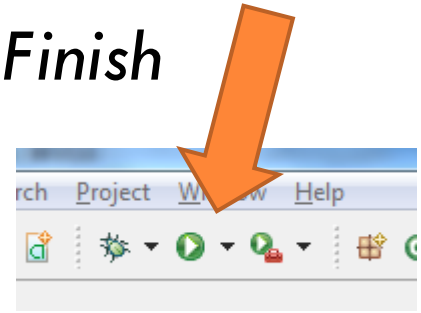

New Android Project
Create Android Project
Select project name and type of project
Project Name: LunarLander
Create new project in workspace
Create project from existing source
Create project from existing sample
Use default location
Location: D:/sonstiges/DHBW/WS11/Projekt/workspace/LunarLander
Browse...
Working sets
Add project to working sets
Working sets:
Select...
< Back
Next >
Finish
Cancel

Erste Anwendung starten

(4) Target 2.3.3 auswählen und *Next* drücken.

(5) Sample „Lunar Lander“ auswählen und *Finish* drücken.

(6) Den Run-Button in der Toolbar klicken.

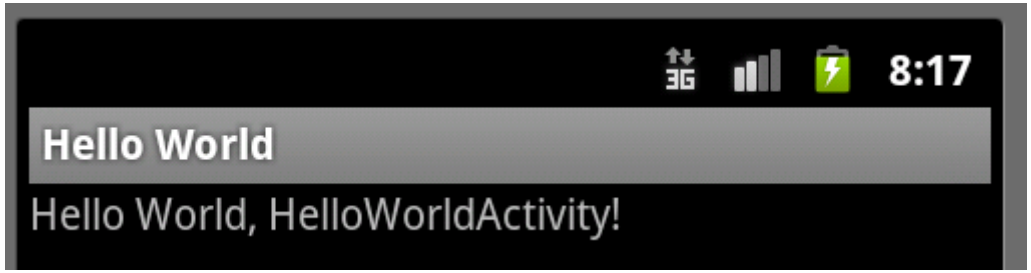


(7) Im Dialog *Run as* den Eintrag *Android Application* wählen und OK klicken.


5554:Gingerbread_WVGA
3:47
Lunar Lander
Press Up To Play
MENU

Hello World

- Wir schreiben unsere erste eigene App. Wie soll den Text „Hello World“ in einem Fenster ausgeben



Hello World

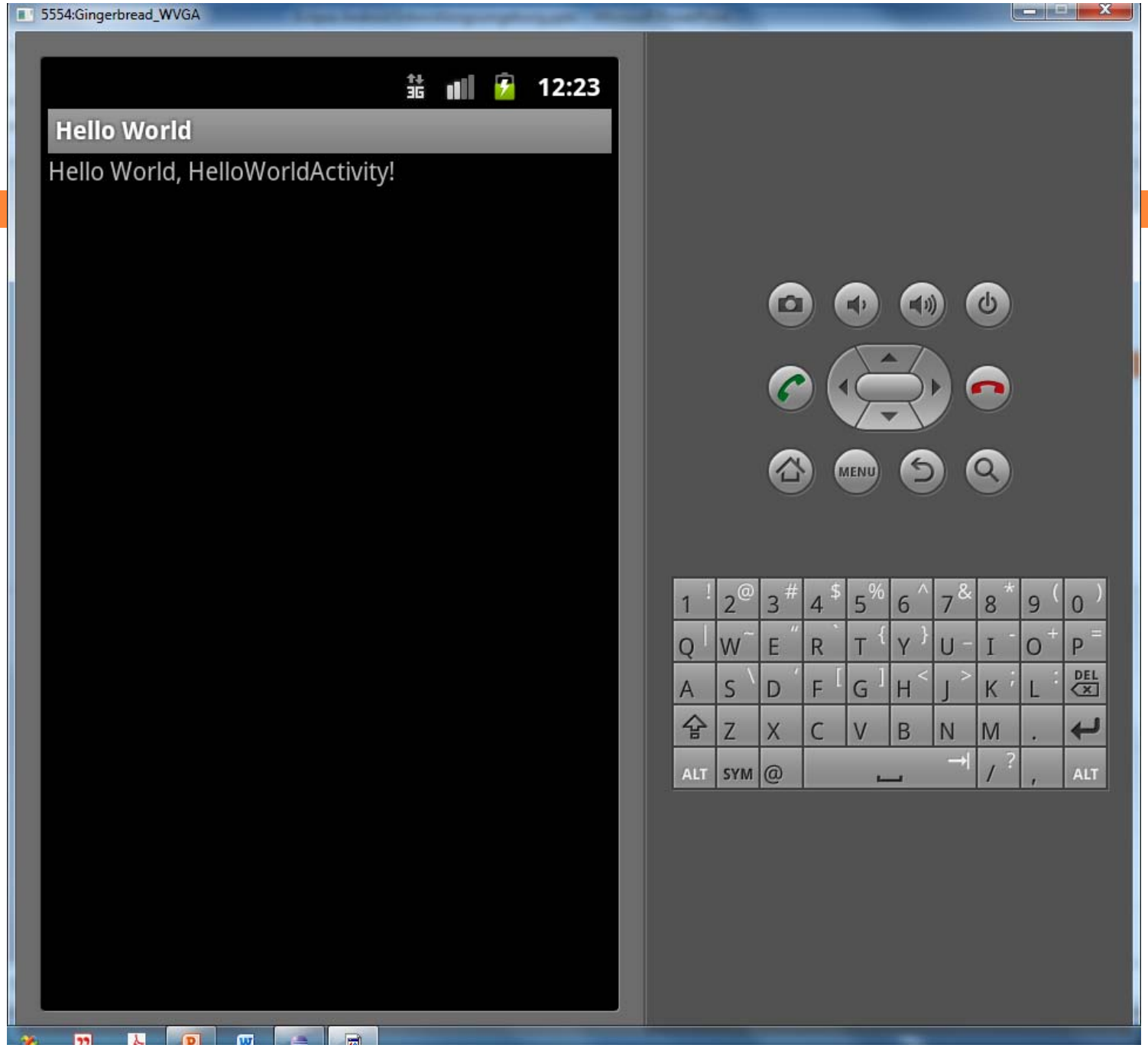
- Schritt 1: Projekt anlegen
  1. In Eclipse wieder den Menüeintrag *File* > *New* > *Project…* wählen.
  2. Im Auswahlbaum wieder *Android* > *Android Project* auswählen und auf *Next* klicken.
  3. Im Dialog *New Android Project* einen Projektnamen vergeben z.B. Hello World. Sicherstellen, dass der Radio-Button *Create New Project in Workspace* aktiviert ist. Auf *Next* klicken.

Hello World

(4) Als Build Target die *Version 2.3.3* auswählen und *Next* klicken.

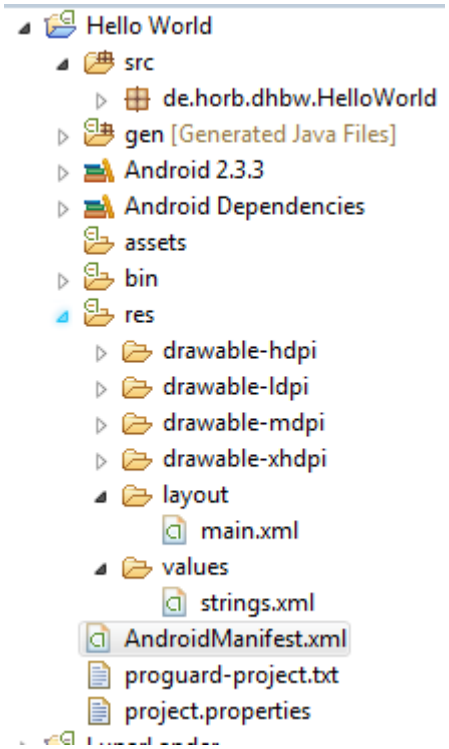
(5) Im Dialog *Application Info* noch einen Package Name vergeben. Hier wählen wir *de.horb.dhbw.HelloWorld*. Den Button *Finish* klicken.

(6) Anwendung übersetzen und ausführen (s. nächste Folie)


5554:Gingerbread_WVGA
12:23
Hello World
Hello World, HelloWorldActivity!
MENU

Hello World

- Projektstruktur
  - Order *src* enthält den Java-Quellcode
  - res enthält die Ressourcen (Strings, Benutzeroberflächen, Grafiken, …)
  - AndroidManifest.xml ist die zentrale Beschreibungsdatei. Sie spielt später eine wesentliche Rolle bei der Veröffentlichung der App



Hello World

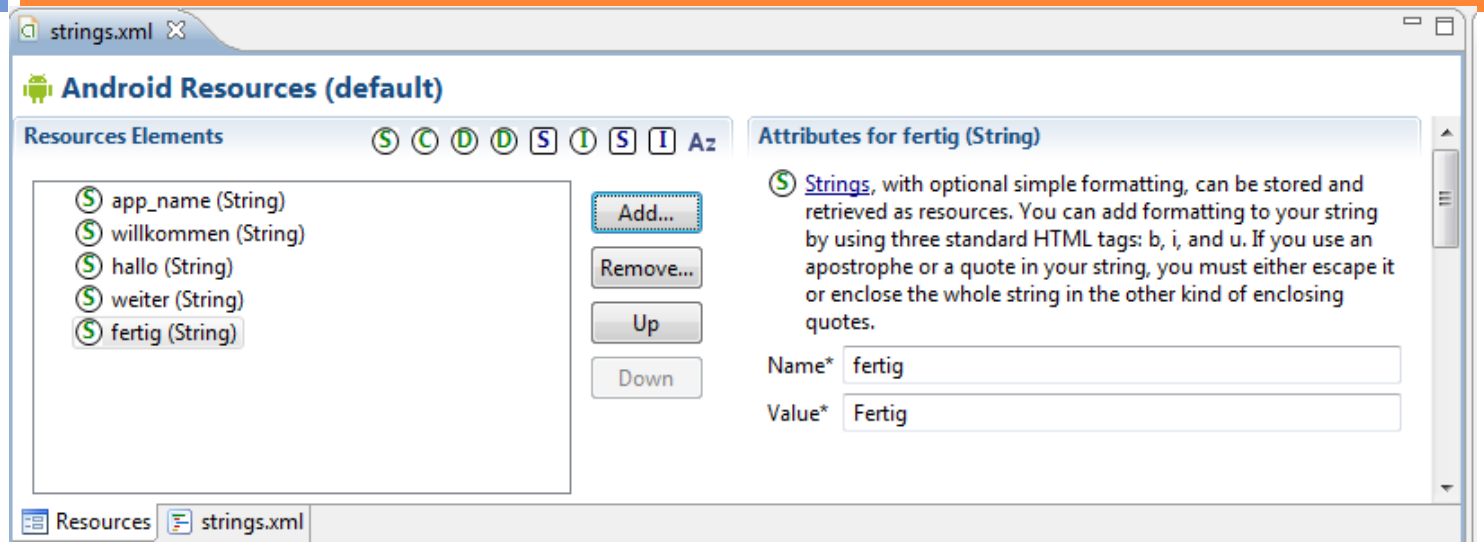
- Erweiterungen: Die Hello-World-Anwendung soll so erweitert werden, dass sie
  - den Benutzer nach seinem Namen fragt.
  - den Benutzer persönlich mit dem eingegebenen Namen begrüßt.

Hello World

Umsetzung der Erweiterung:

- Schritt 1: Ausgabestrings anlegen.
  - Die Anwendung soll folgende Ausgaben liefern
    - „Hallo. Wie heißt Du?“
    - „Hallo <Platzhalter>. Schön Dich kennenzulernen.“
    - Für die Dialog-Buttons: „Weiter“, „Fertigstellen“.
  - Unter Android werden alle Texte in der Datei *strings.xml* abgelegt →Erleichtert die Internationalisierung.

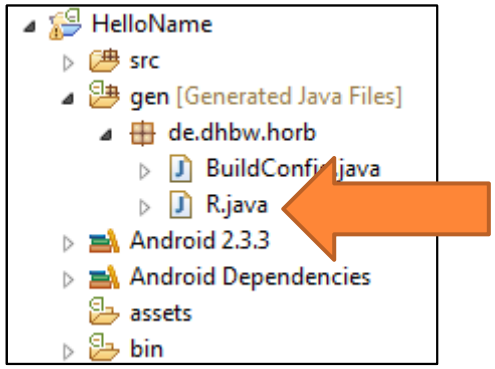
Hello World




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

    <string name="app_name">HelloName</string>
    <string name="willkommen">Hallo. Wie heißt Du?</string>
    <string name="hallo">Hallo %1$s. Schön Dich kennenzulernen.</string>
    <string name="weiter">Weiter</string>
    <string name="fertig">Fertig</string>

</resources>
```

Hello World

- Die String-Konstanten werden automatisch in eine Klasse umgewandelt. Diese Klasse heißt R.




```
/* AUTO-GENERATED FILE.  DO NOT MODIFY.
 *
 * This class was automatically generated by the
 * aapt tool from the resource data it found.  It
 * should not be modified by hand.
 */

package de.dhbw.horb;

public final class R {
    public static final class attr {
    }
    public static final class drawable {
        public static final int ic_launcher=0x7f020000;
    }
    public static final class id {
        public static final int eingabe=0x7f050001;
        public static final int nachricht=0x7f050000;
        public static final int weiter_fertig=0x7f050002;
    }
    public static final class layout {
        public static final int main=0x7f030000;
    }
    public static final class string {
        public static final int app_name=0x7f040000;
        public static final int fertig=0x7f040004;
        public static final int hallo=0x7f040002;
        public static final int weiter=0x7f040003;
        public static final int willkommen=0x7f040001;
    }
}
```

Hello World

- Schritt 2: Benutzeroberfläche erstellen
  - In Android sind alle Bedienelemente direkt oder indirekt von *android.view.View* abgeleitet.
  - Jedes View belegt einen rechteckigen Bereich auf dem Bildschirm.
  - Position und Größe der einzelnen Views wird durch *Layouts* bestimmt.
  - Layouts haben keine grafische Darstellung. Sie sind nur die Container für die Views.

Hello World

- Textfelder, Eingabefelder und Buttons sind Views.
- Wir verwenden die Klassen *Button*, *TextView* und *EditText*.
- Die Elemente platzieren wir mit einem *LinearLayout*.
- Die Benutzeroberfläche wird durch eine xml-Datei beschrieben.

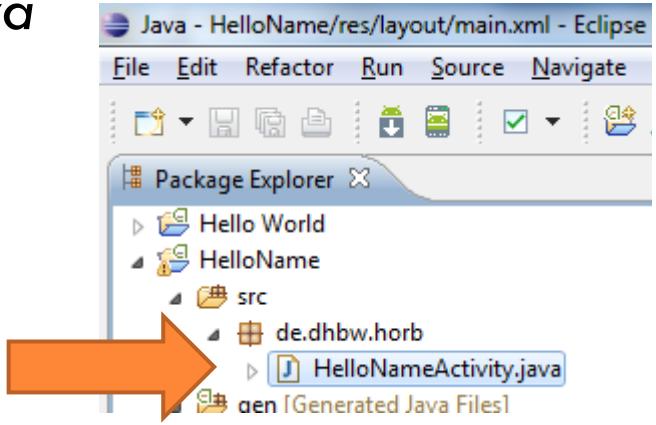
Hello Android


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

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

<TextView
    android:id="@+id/nachricht"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    />

<EditText
    android:id="@+id/eingabe"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
/>

<Button
    android:id="@+id/weiter_fertig"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="right"
/>

</LinearLayout>
```

Inhalt der Datei main.xml

Hello World

- Schritt 3: Activity erstellen
  - Apps sind meistens in mehrere Activities aufgeteilt.
  - Üblicherweise ist jeder Activity eine Benutzeroberfläche zugeordnet.
  - Die Vorwärtsnavigation innerhalb einer Anwendung wird durch Activities realisiert.
  - Für unsere einfache App ist eine Activity ausreichend.
  - Wir modifizieren *HelloNameActivity.java*



Hello World


```
package de.dhbw.horb;

import android.app.Activity;
import android.os.Bundle;
import android.widget.Button;
import android.widget.TextView;


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

        nachricht = (TextView) findViewById(R.id.nachricht);
        weiter_fertig = (Button) findViewById(R.id.weiter_fertig);

        nachricht.setText(R.string.willkommen);
        weiter_fertig.setText(R.string.weiter);
    }

    private TextView nachricht;
    private Button weiter_fertig;

}
```

Inhalt von HelloNameActivity.java

Hello World

- Schritt 4: Benutzereingabe auswerten
  - Jetzt soll beim Klick auf *Weiter* der persönliche Begrüßungstext ausgegeben werden.
  - Ein Klick auf *Fertig* soll das Programm beenden.

```
package de.dhbw.horb;

import android.app.Activity;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.view.View;
import android.view.View.OnClickListener;

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

        nachricht = (TextView)
findViewById(R.id.nachricht);
        weiter_fertig = (Button)
findViewById(R.id.weiter_fertig);
        eingabe = (EditText)
findViewById(R.id.eingabe);


        erster_klick = true;

        nachricht.setText(R.string.willkommen);
        weiter_fertig.setText(R.string.weiter);

weiter_fertig.setOnClickListener(new OnClickListener()
{
            public void onClick(View v) {
                if (erster_klick) {

nachricht.setText(getString(R.string.hallo,
                                    eingabe.getText()));

eingabe.setVisibility(View.INVISIBLE);

weiter_fertig.setText(R.string.fertig);
                    erster_klick = false;
                } else {
                    finish();
                }
            }
        });
    }

    private TextView nachricht;
    private Button weiter_fertig;
    private EditText eingabe;
    private boolean erster_klick;
}
```