

Programming with Google Android and Eclipse

This article describes how to create Android applications with Eclipse.

Version 1.6

Copyright © 2009 Lars Vogel

30.12.2009

The article is based on Eclipse 3.5 and Android 2.0.

Table of Contents

[1. Overview](#)

[1.1. Android](#)

[1.2. Android Application](#)

[2. Installation](#)

[2.1. Eclipse](#)

[2.2. Android](#)

[2.3. Configuration](#)

[2.4. Device](#)

[3. Your first Android project](#)

[3.1. Create Project](#)

[3.2. Add UI Elements](#)

[3.3. Create and use attributes](#)

[3.4. Code your application](#)

[3.5. Define the button handler](#)

[3.6. Start Project](#)

[3.7. Using the phone menu](#)

[4. Important views](#)

[4.1. Log](#)

[4.2. File explorer](#)

[5. Networking](#)

[5.1. Networking](#)

[5.2. Proxy](#)

[5.3. Permissions](#)

[5.4. Example](#)

[6. ContentProvider](#)

[6.1. Overview](#)

[6.2. Create Contacts](#)

[6.3. Example](#)

[7. Shell](#)

[7.1. Opening the Shell](#)

[7.2. Emulator Console](#)

[7.3. Uninstall an application](#)

[8. Location API](#)

[8.1. Device with Google API](#)

[8.2. Project and Permissions](#)

[8.3. Google Map library](#)

[8.4. Layout](#)

[8.5. Activity](#)

[8.6. Run and Test](#)

[9. Thank you](#)

[10. Questions and Discussion](#)

[11. Links and Literature](#)

[11.1. Source Code](#)

[11.2. Android Resources](#)

[11.3. Other Resources](#)

1. Overview

1.1. Android

Android is an operating system based on Linux with a Java programming interface. It provides tools, e.g. a compiler, debugger and a device emulator as well as its own Java Virtual machine (Dalvik).

Android is created by the Open Handset Alliance which is lead by Google.

Android uses a special Java virtual machine (Dalvik) which is based on the Apache Harmony Java implementation. Dalvik uses a special Bytecode so that you have to use the Android compiler to create this special byte-code.

Android supports 2-D and 3-D graphics using the OpenGL libraries and supports data storage in a SQLite database.

For development Google provides the Android Development Tools (ADT) for Eclipse to develop Android applications.

1.2. Android Application

An Android application consists out of the following parts:

- Activity - A screen in the Android application

- Intent / Broadcast Receiver - allow the application to request and / or provide services from other application. For example the application call ask via an intent for a contact application. Application register themself via an IntentFilter
- Services - Background activities without UI
- Content Provider - provides data to applications, Android contains a SQLite DB which can serve as data provider

An Android application is described the file "AndroidManifest.xml". This files contains all classes of the application and the required permissions for the application, e.g. if the application requires network access. "AndroidManifest.xml" can be thought as the deployment descriptor for an Android application.

The following assume that you have already Eclipse install. For installing and using Eclipse please see [Eclipse Tutorial](#)

To use Android you need to install the Eclipse Android Plugin and the base Android SDK. Afterwards you can install different Android versions via the Eclipse Android plugin . You will also need to configure a device which will be used to emulate your real device.

2.1. Eclipse

Use the update manager of Eclipse to install all available plugins for the Android Development Tools (ADT) from the URL <https://dl-ssl.google.com/android/eclipse/> . See [Using the Eclipse update manager](#) for details on how to use the update manager and how to install new plugins.

Tip

Ehe Eclipse Android SDK does not seem to have an option to install the Android (Java) source code to make it available in Eclipse. Please join me in starring at bug [Make Android Source available in Eclipse - Bug report](#) .

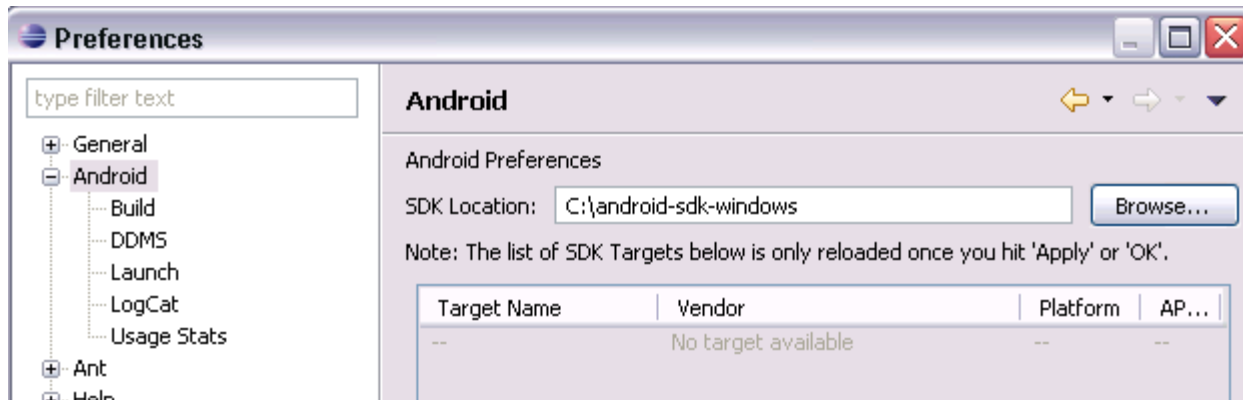
2.2. Android

Download the Android SDK from the Android homepage under [Android Homepage](#) .

The download contains a zip file which you can extract to any place in your file system, e.g. I placed it under "c:\android-sdk-windows" .

2.3. Configuration

In Eclipse open the Preferences dialog via Windows -> Preferences. Select Android and maintain the installation path of the Android SDK.



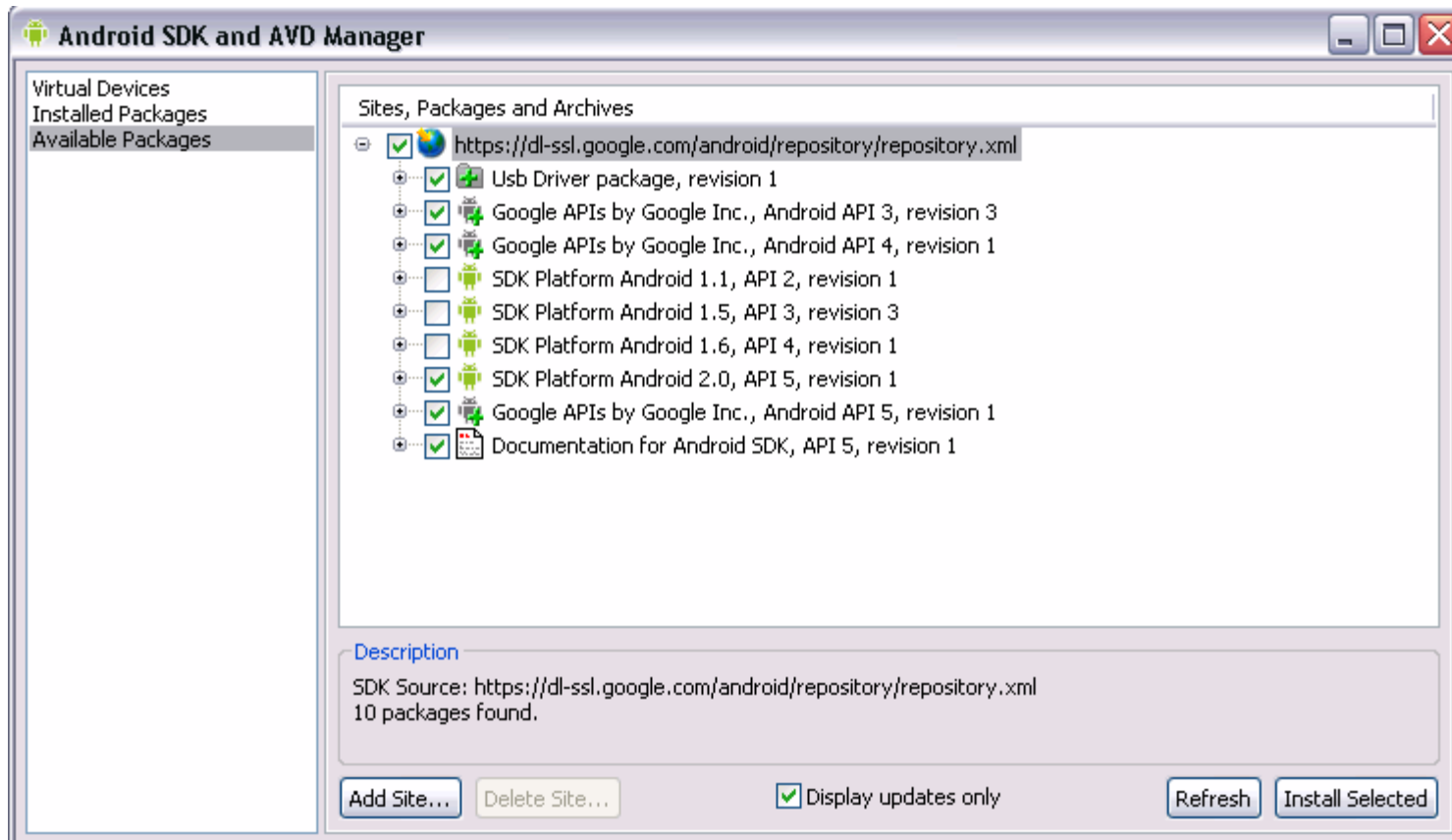
Tip

If you maintain the location the Android plugin will remind you frequently (and for every workspace). Join me in starring at [Bug 3210](#) to get this improved.

Select now Window -> Android SDK and AVD Manager from the menu.



Select available packages and select everything except the older version of the SDK.

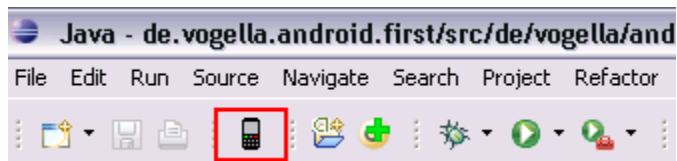


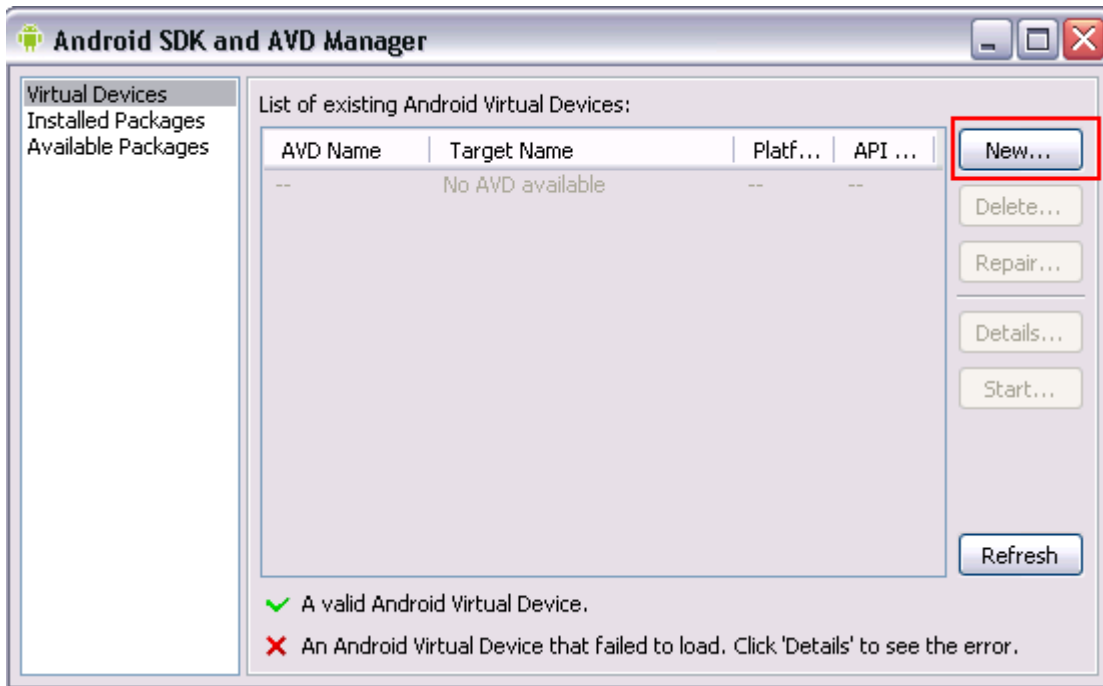
Press "Install selected" and confirm the license for all package.

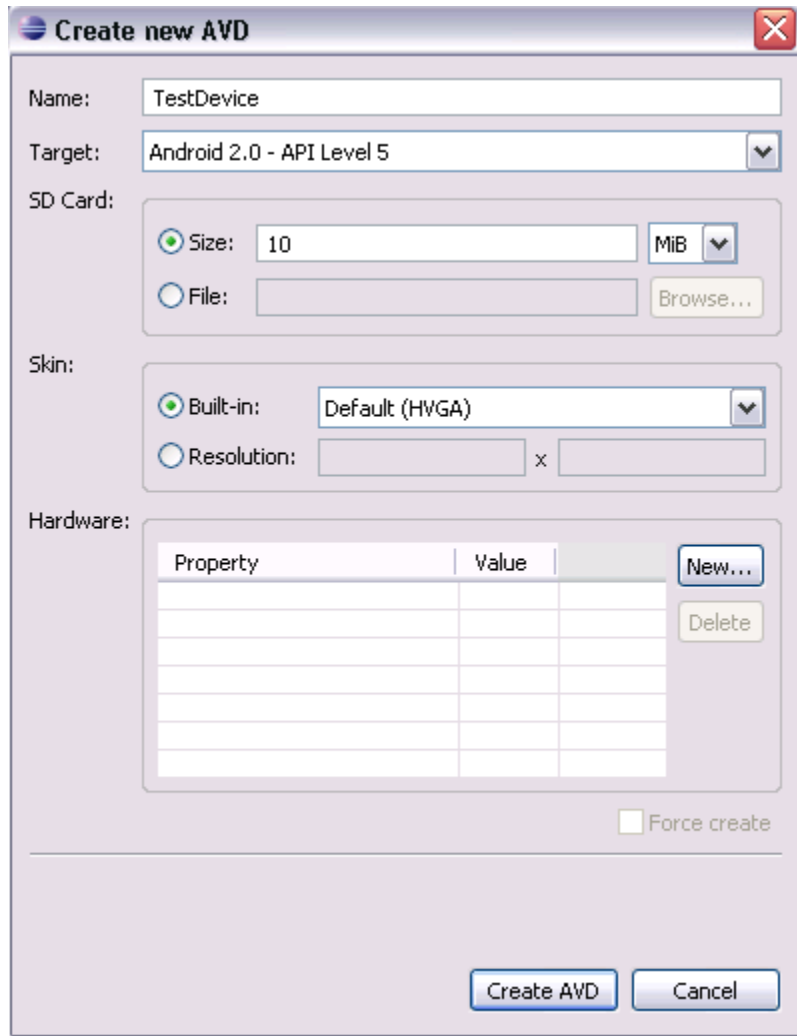
After the installation restart Eclipse.

2.4. Device

You need to define a device which can be used for emulation. Press the device manager button, press "New" and maintain the following.

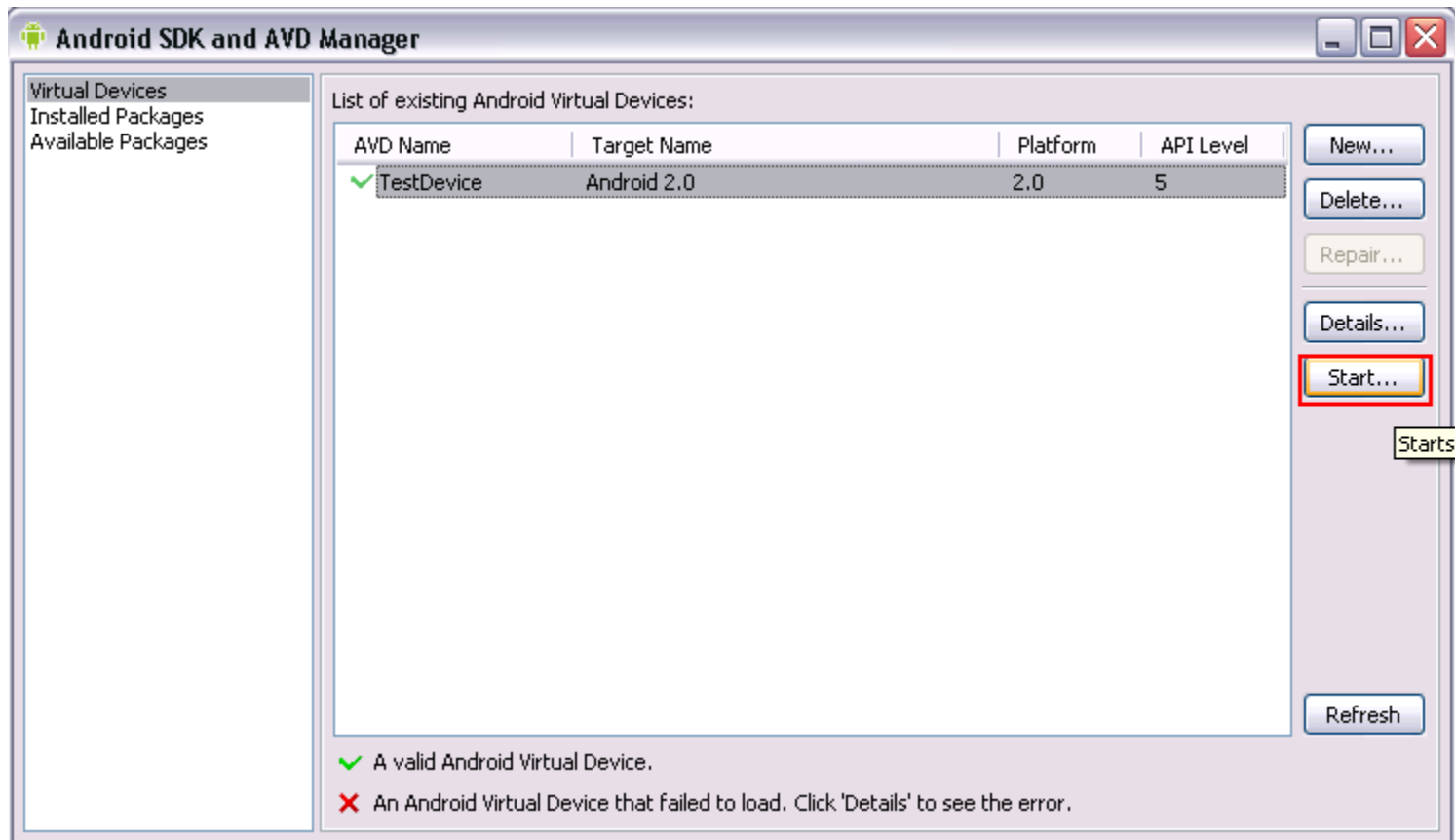




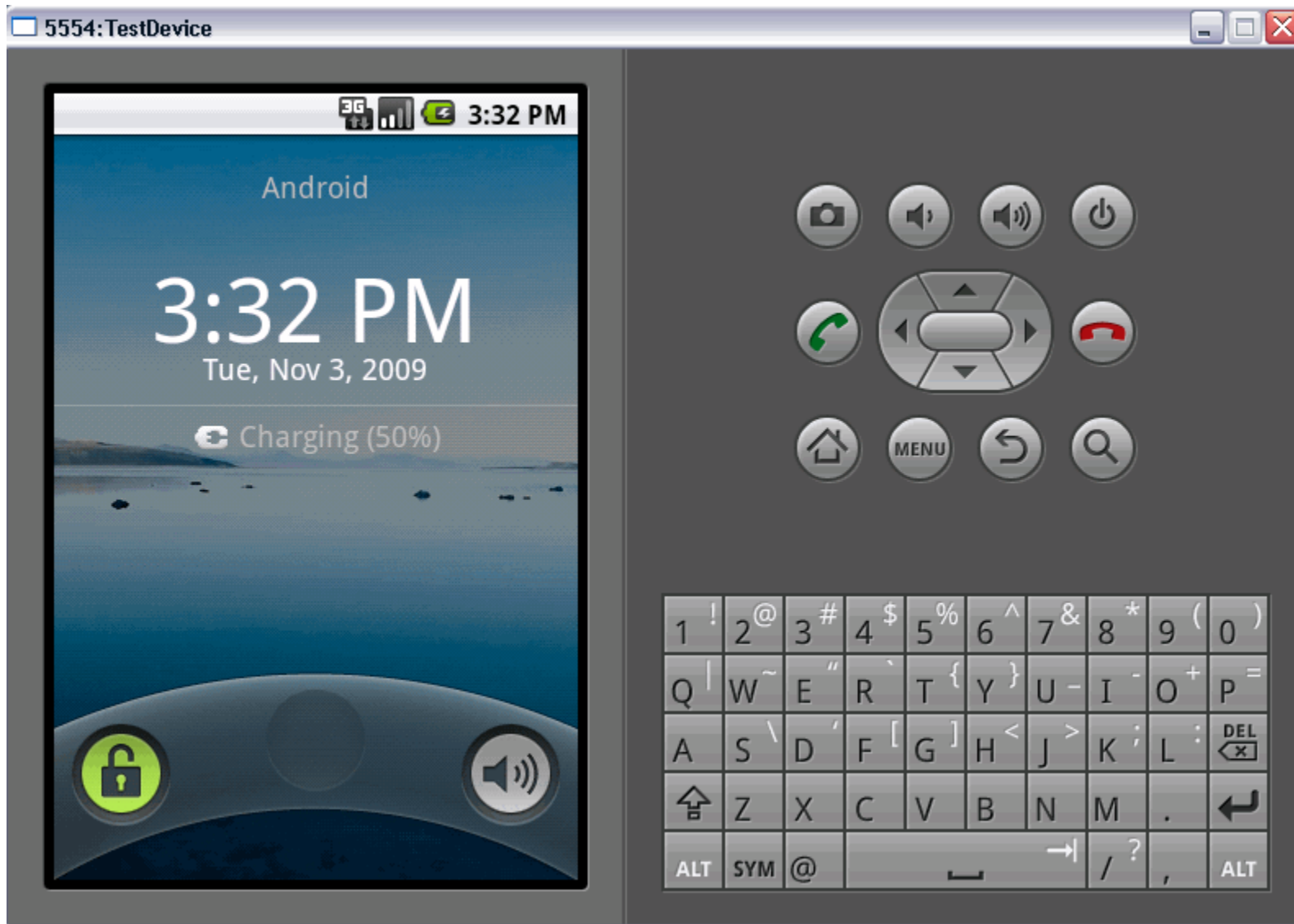


Press "Create AVD".

To test if you setup is correct, select your device and press "Start".



After (a long time) your device should be started.



Tip

You can use the perspective "DDMS" to monitor your device.

3. Your first Android project

3.1. Create Project

. Select File -> New -> Other -> Android -> Android Project and create the Android project "de.vogella.android.first" Maintain the following.

New Android Project

Creates a new Android Project resource.

Project name:

Contents

Create new project in workspace
 Create project from existing source
 Use default location

Location:

Create project from existing sample

Samples:

Build Target

Target Name	Vendor	Platform	API Level
<input checked="" type="checkbox"/> Android 2.0	Android Open Source Project	2.0	5
<input type="checkbox"/> Google APIs	Google Inc.	2.0	5

Standard Android platform 2.0

Properties

Application name:

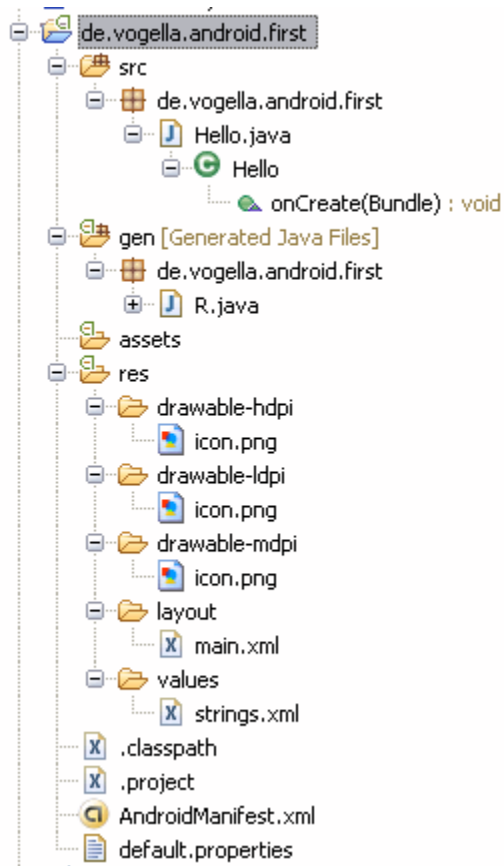
Package name:

Create Activity:

Min SDK Version:

Press "Finish".

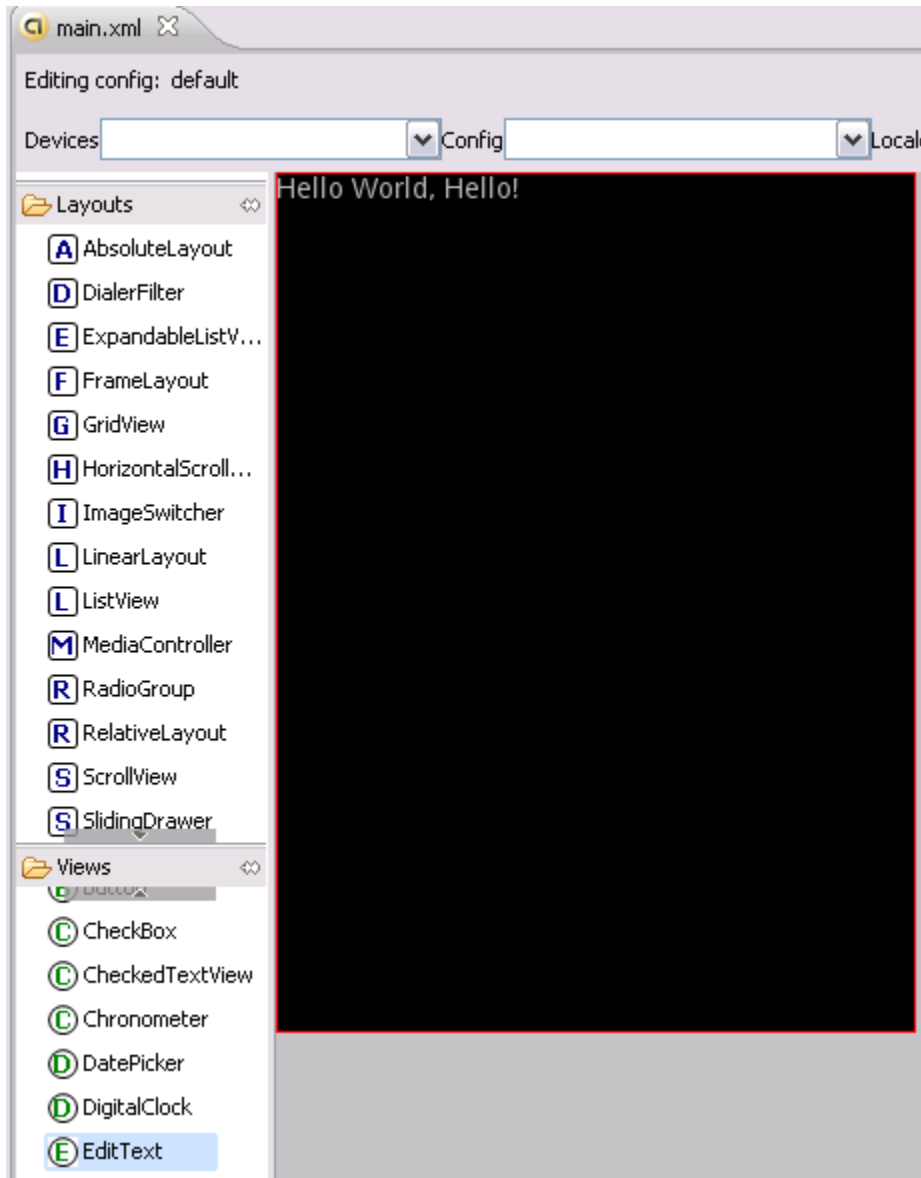
This should create the following directory structure.



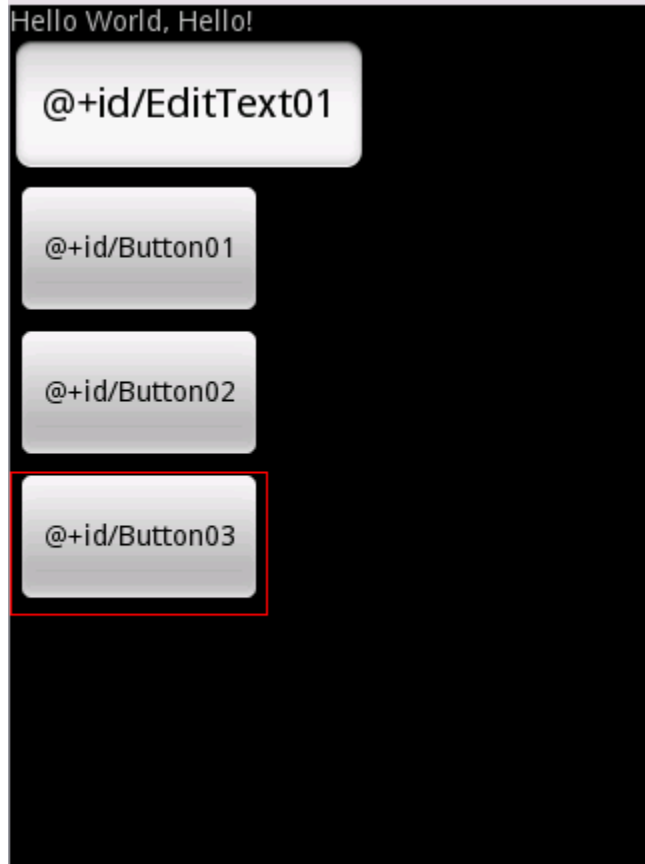
R.java is a generated class which contains the text and the UI elements. Please do not try to modify this class manually.

3.2. Add UI Elements

Select main.xml and open the editor via double-click. The result should look like the following.



From the "Views" bar, drag in an "EditText" and three "Buttons". The result should look like the following.

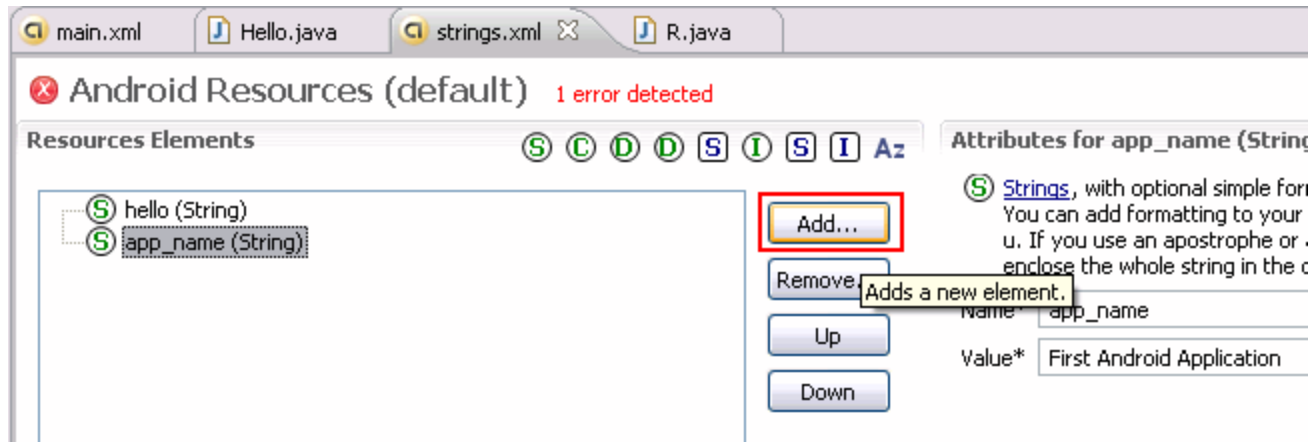


Tip

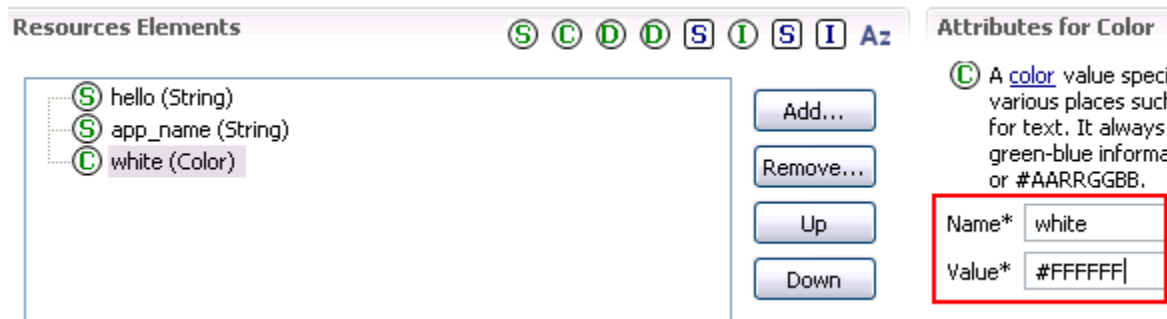
Check R.java it will contain your new elements.

3.3. Create and use attributes

Select "string.xml" and press "Add".



Select color and maintain "white" as the name and "#FFFFFF" as the value.



Go back to "main.xml", select the complete widget and use the Properties view to set the background to this attribute.

main.xml Hello.java strings.xml R.java

Editing config: default

Devices: ADP1 Config: Landscape, closed Locale: Theme:

Layouts

- AbsoluteLayout
- DialerFilter
- ExpandableListV...
- FrameLayout
- GridView
- HorizontalScroll...
- ImageSwitcher
- LinearLayout
- ListView
- MediaController

Views

- AutoCompleteT...
- Button
- CheckBox
- CheckedTextView
- Chronometer
- DatePicker
- DigitalClock
- EditText
- Gallery

Layout: main.xml

CVS R Proble Javad Declar Consol Error L Progre Call Hi Out

Property Value

Property	Value
LinearLayout	
Add states from children	
Always drawn with cache	
Animation cache	
Background	@color/white
Baseline aligned	
Baseline aligned child index	
Clickable	
Clip children	
Clip to padding	
Content description	

Reference Chooser

Choose a resource

type filter text

- Color
 - white
- Drawable
- ID
- Layout
- String

New Color...

OK Cancel

3.4. Code your application

Change your code in "Hello.java" to the following.

```
package de.vogella.android.first;

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

public class Hello extends Activity {

    private EditText text;

    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main); // bind the layout to the activity
        text = (EditText) findViewById(R.id.EditText01);
        text.setText("No button pressed");

    }

    // Will be connected with the buttons via XML
    public void myClickHandler(View view) {
        switch (view.getId()) {
            case R.id.Button01:
                text.setText("Button 1 was clicked");
                break;
            case R.id.Button02:
                text.setText("Button 2 was clicked");
                break;

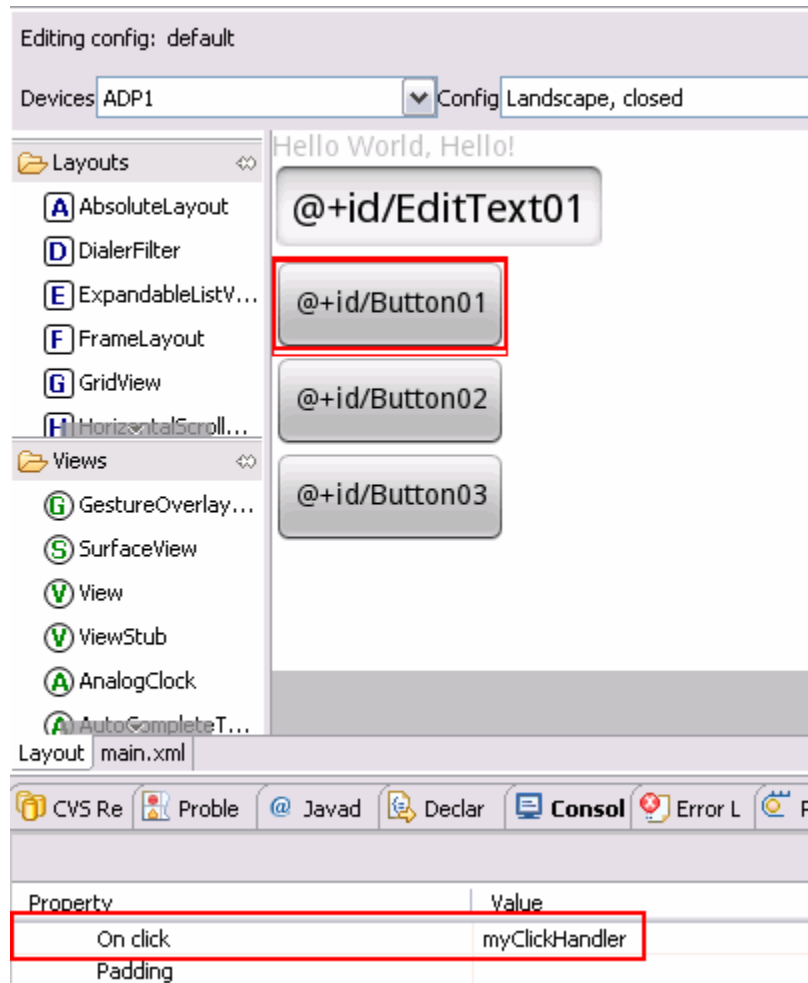
            case R.id.Button03:
                text.setText("Button 3 was clicked");
                break;
        }
    }
}
```

Tip

The next chapter will connect the handler methods with the buttons via XML.

3.5. Define the button handler

Open again main.xml and select your first button. Via the property view assign the method "myClickHandlerButton" to the "on Click" property of the first button.



Assign "myClickHandlerButton" to the other buttons.

The resulting main.xml should look like the following.

```
<?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"
```

```
        android:background="@color/white">
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/hello"
    />
<EditText android:text="@+id/EditText01" android:id="@+id/EditText01" android:layout_width="wrap_content"
android:layout_height="wrap_content"></EditText>
<Button android:text="@+id/Button01" android:id="@+id/Button01" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:onClick="myClickHandler"></Button>
<Button android:text="@+id/Button02" android:id="@+id/Button02" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:onClick="myClickHandler"></Button>
<Button android:text="@+id/Button03" android:id="@+id/Button03" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:onClick="myClickHandler"></Button>
</LinearLayout>
```

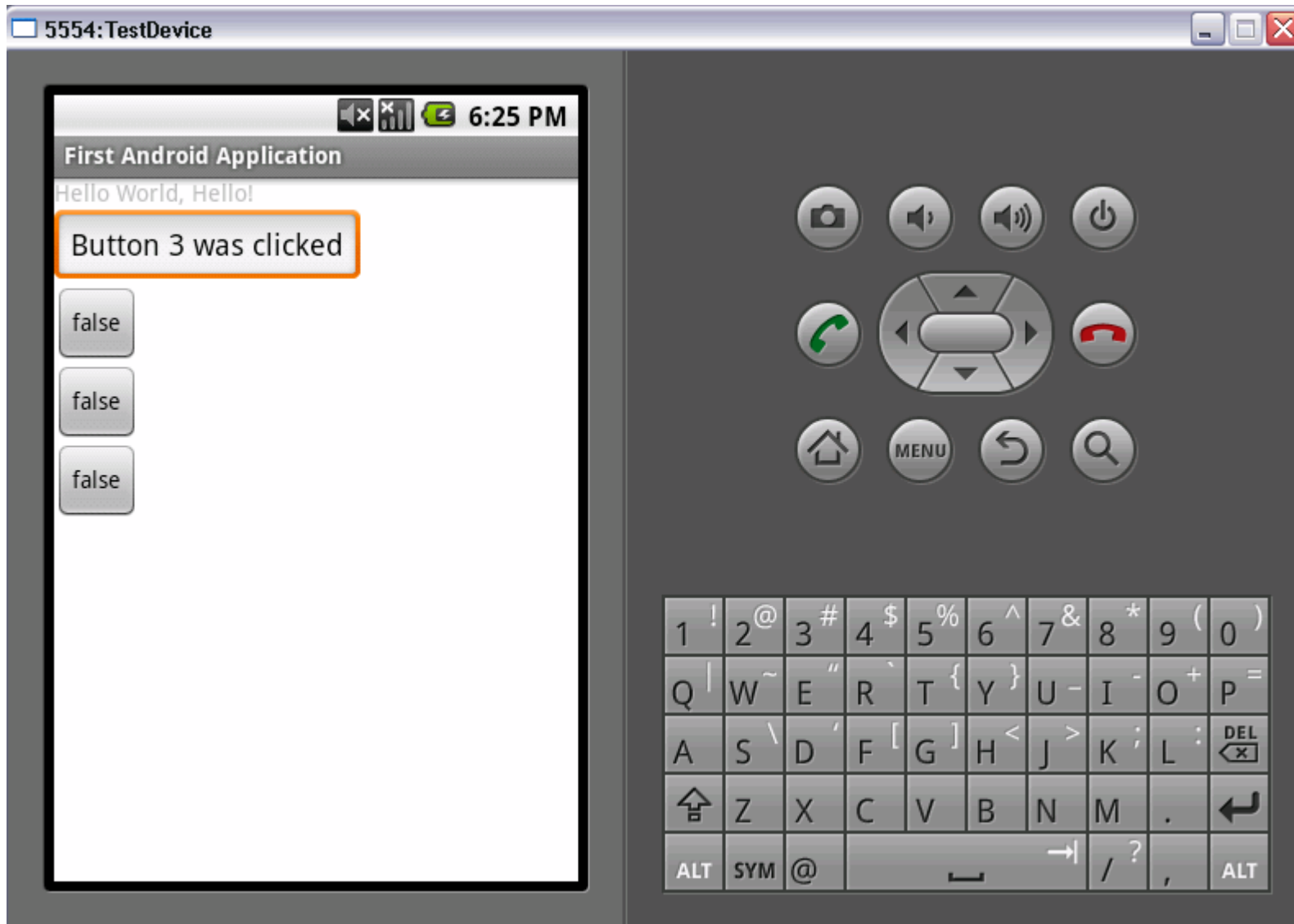
3.6. Start Project

To start the Android Application, select your project, right click on it, Run-As-> Android Application

Tip

Be patient, the emulator is sometimes very slow.

You should get the following result.



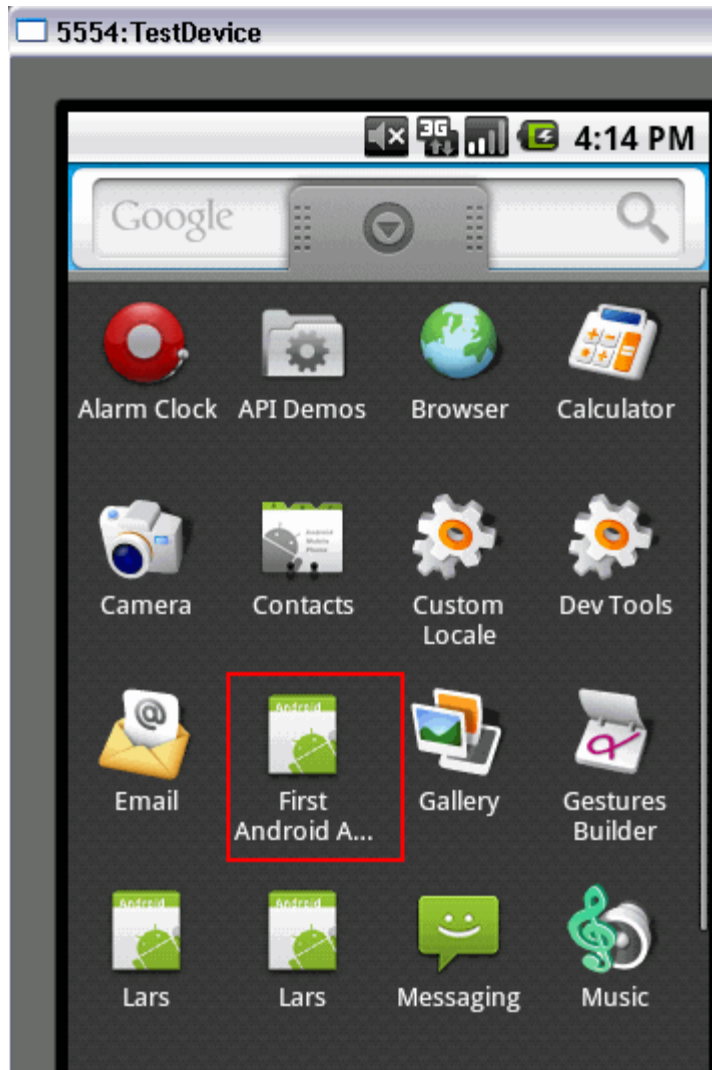
3.7. Using the phone menu

If you press the Home button you can also select your application.



Home

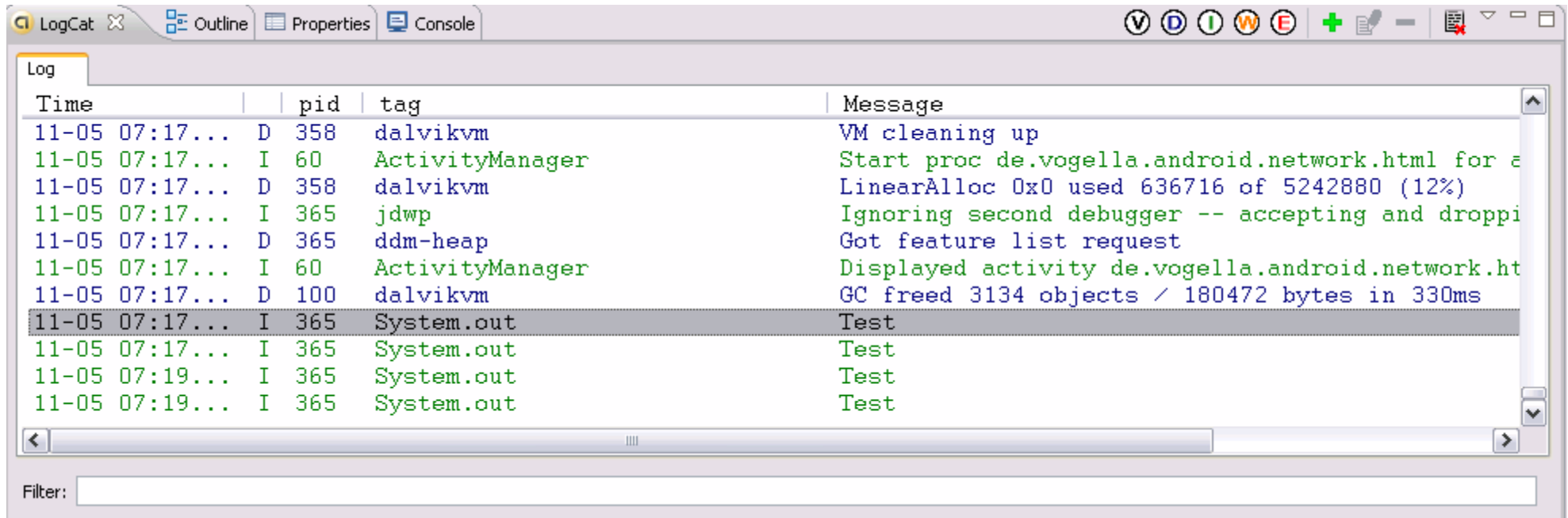




4. Important views

4.1. Log

You can see the log (including `System.out.print()` statements) via the LogCat view.



4.2. File explorer

The file explorer allows to see the files on the android simulator.

Name	Size	Date	Time	Permissions	Info
data		2009-10-22	01:34	drwxrwx--x	
anr		2009-11-03	15:26	drwxrwx--x	
app		2009-10-22	01:34	drwxrwx--x	
ApiDemos.apk	2184352	2009-10-22	01:34	-rw-r--r--	com.examp...
SoftKeyboard.apk	35613	2009-10-22	01:34	-rw-r--r--	com.examp...
de.vogella.android.first.apk	14677	2009-11-03	18:24	-rw-r--r--	de.vogella...
de.vogella.android.network.html.apk	15107	2009-11-05	09:54	-rw-r--r--	de.vogella...
de.vogella.android.test.apk	13300	2009-11-03	15:36	-rw-r--r--	de.vogella...
de.vogella.android.test2.apk	13301	2009-11-03	15:34	-rw-r--r--	de.vogella...
app-private		2009-11-03	15:26	drwxrwx--x	
backup		2009-11-03	15:27	drwx-----	
dalvik-cache		2009-11-03	15:26	drwxrwx--x	
data		2009-11-03	15:26	drwxrwx--x	
dontpanic		2009-11-03	15:26	drwxr-x---	
local		2009-11-03	15:26	drwxrwx--x	
lost+found		2009-11-03	15:26	drwxrwx---	
misc		2009-11-03	15:26	drwxrwx-t	
property		2009-11-03	15:26	drwx-----	
system		2009-11-03	15:26	drwxrwxr-x	
sdcard		1969-12-31	16:00	d---rwxr-x	
system		2009-10-22	01:33	drwxr-xr-x	

5. Networking

5.1. Networking

Android allows to access the network via the the `java.net.URL` class.

Tip

You can also read XML, e.g. RSS feeds. Unfortunately Android does not have a Stax parser included in it SDK. Vote for [Android should have a Stax parser](#) to get support. Currently you have to use the android specific class `XmlPullParser`.

5.2. Proxy

To set the proxy you can use the class Settings. For example you could add the following line to your onCreate method in your activity.

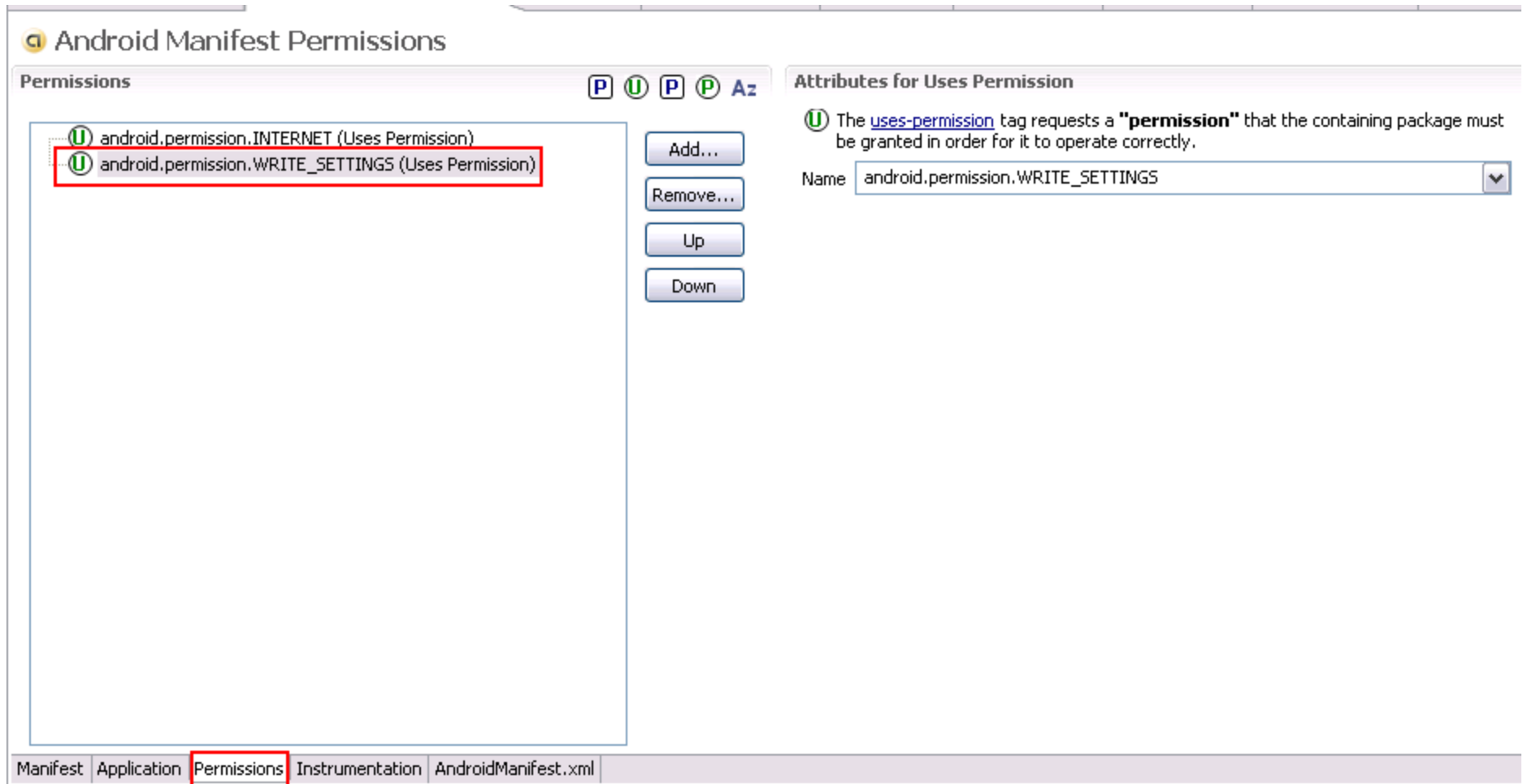
```
Settings.System.putString(getContentResolver(), Settings.System.HTTP_PROXY, "myproxy:8080");
```

Tip

It seems that DNS resolving doesn't work behind a proxy. See [Bug 2764](#)

5.3. Permissions

You also have to give your application the right to change the settings "android.permission.WRITE_SETTINGS" in "AndroidManifest.xml".



5.4. Example

Create the project "de.vogella.android.network.html". Add the following elements to your activity:

- EditText with the ID "address"
- TextView with the ID "pagetext"
- Button with the ID "ReadWebPage"

Create the following code to read a webpage and show the HTML code in the TextView.

This example also demonstrate the usage of Android preferences to store user data. The URL which the user has typed is stored in the preferences in the method onPause(). This method is called whenever the Activity is send into the background.

```

package de.vogella.android.network.html;

import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.net.URL;

import android.app.Activity;
import android.content.SharedPreferences;
import android.content.SharedPreferences.Editor;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.TextView;

public class ReadWebpage extends Activity {
    private static final String PREFERENCES = "PREFERENCES";
    private static final String URL = "url";
    private String lastUrl;
    private EditText urlText;
    private TextView textView;

    /** Called when the activity is first created. */

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        urlText = (EditText) findViewById(R.id.address);
        textView = (TextView) findViewById(R.id.pagetext);

        loadPreferences();
        urlText.setText(lastUrl);
    }

    /**
     * Demonstrates loading of preferences The last value in the URL string will
     * be loaded
     */
    private void loadPreferences() {
        SharedPreferences preferences = getSharedPreferences(PREFERENCES,
            Activity.MODE_PRIVATE);
        // Set this to the Google Homepage
        lastUrl = preferences.getString(URL, "http://209.85.229.147");
    }

    @Override
    protected void onPause() {
        super.onPause();
        SharedPreferences preferences = getSharedPreferences(PREFERENCES,

```

```

        Activity.MODE_PRIVATE);
        Editor preferenceEditor = preferences.edit();
        preferenceEditor.putString(URL, urlText.getText().toString());
        // You have to commit otherwise the changes will not be remembered
        preferenceEditor.commit();
    }

    // Will be connected with the buttons via XML
    public void myClickHandler(View view) {
        switch (view.getId()) {
            case R.id.ReadWebPage:
                try {
                    textView.setText("");
                    // Perform action on click
                    URL url = new URL(urlText.getText().toString());
                    // Get the response
                    BufferedReader rd = new BufferedReader(
                        new InputStreamReader(url.openStream()));
                    String line = "";
                    while ((line = rd.readLine()) != null) {
                        textView.append(line);
                    }
                }
                catch (Exception e) {
                    System.out.println("Nay, did not work");
                    textView.setText(e.getMessage());
                }
                break;
        }
    }
}

```

Assign the handler "buttonHandler" to the button in the property "on Click". via your XML.

6. ContentProvider

6.1. Overview

ContentProvider are used to provide data from an application to another. ContentProvider do not store the data but provide the interface for other applications to access the data.

The following example will use an existing context provider from "Contacts".

6.2. Create Contacts

Start the contacts application and create a few contacts.



6.3. Example

Create a new Android project "de.vogella.android.contentprovider" with the activity "ContactsView".

Rename the id of the the existing TextView from the example wizard to "contactview". Delete the default text. Also change the layout_height to "fill_parent".

The resulting main.xml should look like the following.

```
<?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:layout_width="fill_parent"
        android:layout_height="fill_parent" android:id="@+id/contactview" />
</LinearLayout>
```

In Application.xml add the Permission that the application can use "android.permission.READ_CONTACTS".

Change now your coding of your activity.

```
package de.vogella.android.contentprovider;

import android.app.Activity;
import android.database.Cursor;
import android.os.Bundle;
import android.provider.ContactsContract.Contacts;
import android.widget.TextView;

public class ContactsView extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        TextView contactView = (TextView) findViewById(R.id.contactview);
        Cursor cursor = getContentResolver().query(Contacts.CONTENT_URI, null, null, null, null);
        while(cursor.moveToNext()){
            String displayName = cursor.getString(cursor.getColumnIndex(Contacts.DISPLAY_NAME));
            contactView.append("Name: ");
            contactView.append(displayName);
            contactView.append("\n");
        }
    }
}
```

Tip

Currently the contentProvider does not return any data. Looks like the content provider has been changed but [the content provider documentation](#) still refers to the People class. If you find a solution please email me.

7. Shell

7.1. Opening the Shell

You can access your Android emulator also via the console. Open a shell, switch to your "android-sdk" installation directory into the folder "tools". Start the shell via the command "adb shell".

7.2. Emulator Console

The emulator console lets you dynamically access your simulated device. Use "telnet localhost 5554" to connect to your simulated device. To exit the console session, use the command "quit" or "exit".

For example you can set your geolocation in the emulator via "geo fix -121.45356 46.51119 4392"

For more information on the emulator console please see [Emulator Console manual](#)

7.3. Uninstall an application

You can uninstall an android application via the shell. Switch to the data/app directory (cd /data/app) and simply delete your android application.

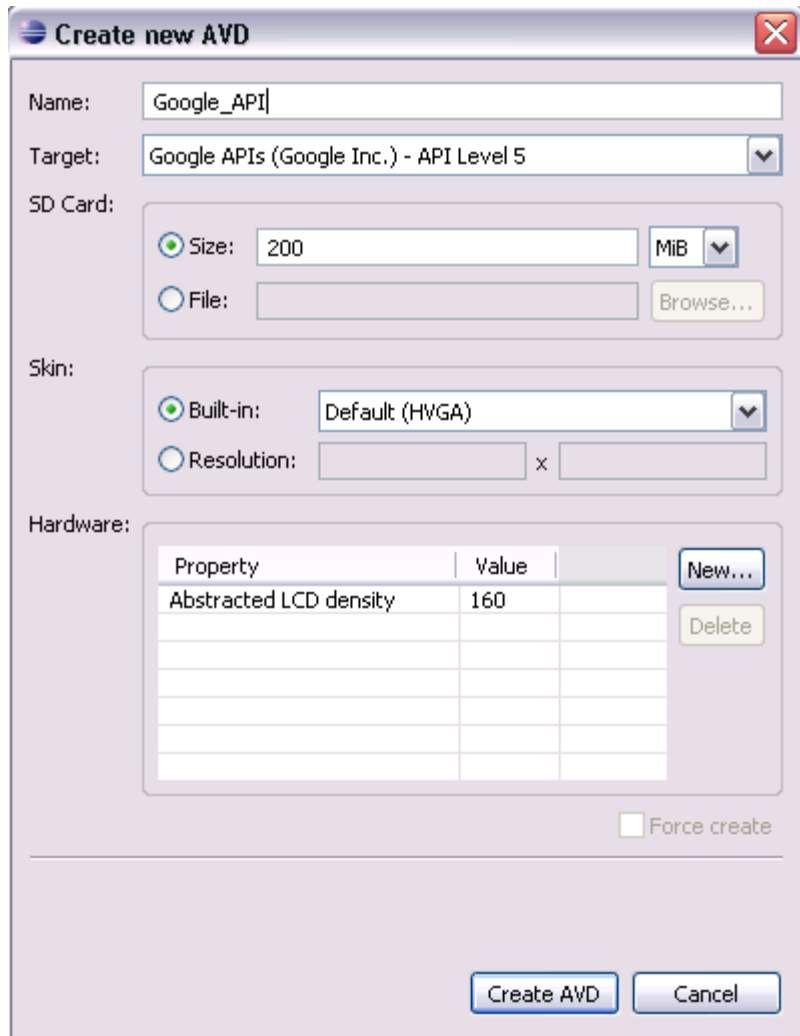
8. Location API

The location API allow you to determine your current location.

The following requires that you have installed the Google API (see installation) and a valid Google map API key. Go to [Obtaining a Maps API Key](#) to get one.

8.1. Device with Google API

Create a new device which supports the Google API's.



8.2. Project and Permissions

Create a new project "de.vogella.android.locationapi". Make sure to select the Google API

New Android Project

Creates a new Android Project resource.

Project name:

Contents

Create new project in workspace
 Create project from existing source
 Use default location

Location:

Create project from existing sample

Samples:

Build Target

Target Name	Vendor	Platform	API Level
<input type="checkbox"/> Android 2.0	Android Open Source Project	2.0	5
<input checked="" type="checkbox"/> Google APIs	Google Inc.	2.0	5

Android + Google APIs

Properties

Application name:

Package name:

Create Activity:

Min SDK Version:

Add the following permissions to your application in Android.xml.

- INTERNET

- ACCESS_FINE_LOCATION
- ACCESS_COARSE_LOCATION

Tip

The maintenance of "Uses permissions" should be enhanced. Please stare at [Bug: Permissions should support field assists](#) to get this improved.

8.3. Google Map library

You need to add the Google maps library to your application. Open Android.xml, tab Application and add a "Uses library".

Android Manifest Application

Application Toggle

The `application` tag describes application-level components contained in the package, as well as general application attributes.

Define an `<application>` tag in the `AndroidManifest.xml`

Application Attributes

Defines the attributes specific to the application.

Name	<input type="text"/>	<input type="button" value="Browse..."/>	Persistent	<input type="text"/>	<input type="button" value="Browse..."/>
Theme	<input type="text"/>	<input type="button" value="Browse..."/>	Enabled	<input type="text"/>	<input type="button" value="Browse..."/>
Label	@string/app_name	<input type="button" value="Browse..."/>	Debuggable	<input type="text"/>	<input type="button" value="Browse..."/>
Icon	@drawable/icon	<input type="button" value="Browse..."/>	Manage space activity	<input type="text"/>	<input type="button" value="Browse..."/>
Description	<input type="text"/>	<input type="button" value="Browse..."/>	Allow clear user data	<input type="text"/>	<input type="button" value="Browse..."/>
Permission	<input type="text"/>	<input type="button" value="Browse..."/>	Test only	<input type="text"/>	<input type="button" value="Browse..."/>
Process	<input type="text"/>	<input type="button" value="Browse..."/>	Backup agent	<input type="text"/>	<input type="button" value="Browse..."/>
Task affinity	<input type="text"/>	<input type="button" value="Browse..."/>	Allow backup	<input type="text"/>	<input type="button" value="Browse..."/>
Allow task reparenting	<input type="text"/>	<input type="button" value="Browse..."/>	Kill after restore	<input type="text"/>	<input type="button" value="Browse..."/>
Has code	<input type="text"/>	<input type="button" value="Browse..."/>	Restore needs application	<input type="text"/>	<input type="button" value="Browse..."/>

Application Nodes

- .CurrentLocation (Activity)
- Intent Filter
- com.google.android.maps (Uses Library)

The "uses-libraries" specifies a shared library that this package requires to be linked against.

Name	com.google.android.maps
Required	true

Manifest Application Permissions Instrumentation AndroidManifest.xml

8.4. Layout

Define your view layout like the following.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/mainlayout"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >

    <com.google.android.maps.MapView
        android:id="@+id/mapview"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:clickable="true"
        android:apiKey="Your Maps API Key"
    />

</RelativeLayout>
```

Tip

Replace "Your Maps API Key" with your Google API key.

8.5. Activity

Create the following activity. This activity use an LocationListner to update the map with the current location.

```
package de.vogella.android.locationapi;

import android.content.Context;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.widget.RelativeLayout;
import android.widget.ZoomControls;

import com.google.android.maps.GeoPoint;
import com.google.android.maps.MapActivity;
import com.google.android.maps.MapController;
import com.google.android.maps.MapView;

public class CurrentLocation extends MapActivity {

    private MapController mapController;
    private MapView mapView;
    private LocationManager locationManager;
```

```

public void onCreate(Bundle bundle) {
    super.onCreate(bundle);
    setContentView(R.layout.main); // bind the layout to the activity

    // create a map view
    RelativeLayout linearLayout = (RelativeLayout) findViewById(R.id.mainlayout);
    mapView = (MapView) findViewById(R.id.mapview);
    ZoomControls mZoom = (ZoomControls) mapView.getZoomControls();
    linearLayout.addView(mZoom);
    mapController = mapView.getController();
    // Zoon 1 is world view

    mapController.setZoom(14);

    locationManager = (LocationManager) getSystemService(Context.LOCATION_SERVICE);

    locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER, 0,
        0, new GeoUpdateHandler());
}

@Override
protected boolean isRouteDisplayed() {
    return false;
}

public class GeoUpdateHandler implements LocationListener {

    @Override
    public void onLocationChanged(Location location) {
        int lat = (int) (location.getLatitude() * 1E6);
        int lng = (int) (location.getLongitude() * 1E6);
        GeoPoint point = new GeoPoint(lat, lng);
        mapController.setCenter(point);
        setContentView(mapView);
    }

    @Override
    public void onProviderDisabled(String provider) {
    }

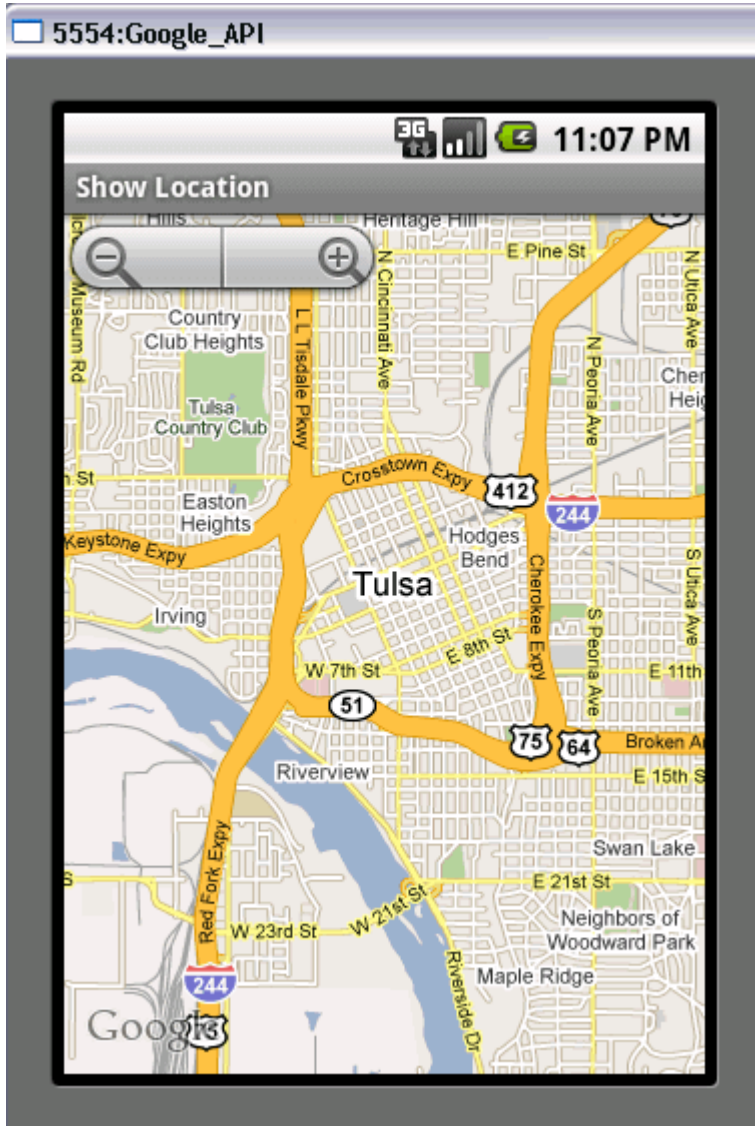
    @Override
    public void onProviderEnabled(String provider) {
    }

    @Override
    public void onStatusChanged(String provider, int status, Bundle extras) {
    }
}
}

```

8.6. Run and Test

Run and test your application. You should be able to zoom in and out. Use [Emulator console](#) to send geo-coordinates to your device for example



```
geo fix 13.24 52.31
```

Tip

See also [Hello, MapView](#) for an example how to put graphics on the map.

9. Thank you

Thank you for practicing with this tutorial.

Please note that I maintain this website in my private time. If you like the information I'm providing please help me by donating.

10. Questions and Discussion

For questions and discussion around this article please use the [www.vogella.de Google Group](http://www.vogella.de). Also if you note an error in this article please post the error and if possible the correction to the Group.

I believe the following is a very good guideline for asking questions in general and also for the Google group [How To Ask Questions The Smart Way](#).

11.1. Source Code

<http://www.vogella.de/code/codeclipse.html> Source Code of Examples

11.2. Android Resources

[Android Homepage](#)

[Android Developer Homepage](#)

[Android Issues / Bugs](#)

[Android Market](#)

[SQL light database homepage](#)

[Content Provider Tutorial by Google](#)

[Notepad Tutorial from Google](#)

<http://www.ibm.com/developerworks/opensource/library/os-android-networking/index.html> Networking with Android from Frank Ableson

<http://www.ibm.com/developerworks/opensource/library/os-android-sensor/index.html> Androids Sensors from Frank Ableson

<http://www.ibm.com/developerworks/opensource/library/x-android/index.html> Android and XML from Michael Galpin

11.3. Other Resources

[Eclipse.org Homepage](#)

[Articles about Java, Eclipse and Webdevelopment from www.vogella.de](#)

[Articles about Eclipse development from www.vogella.de](#)

[Articles about Web development from www.vogella.de](#)