Android Workshop

Probestudium, 15.-20.4.2011

Prof. Dr. Michael Rohs

michael.rohs@ifi.lmu.de

Dipl.-Inform. Sven Kratz

sven.kratz@ifi.lmu.de

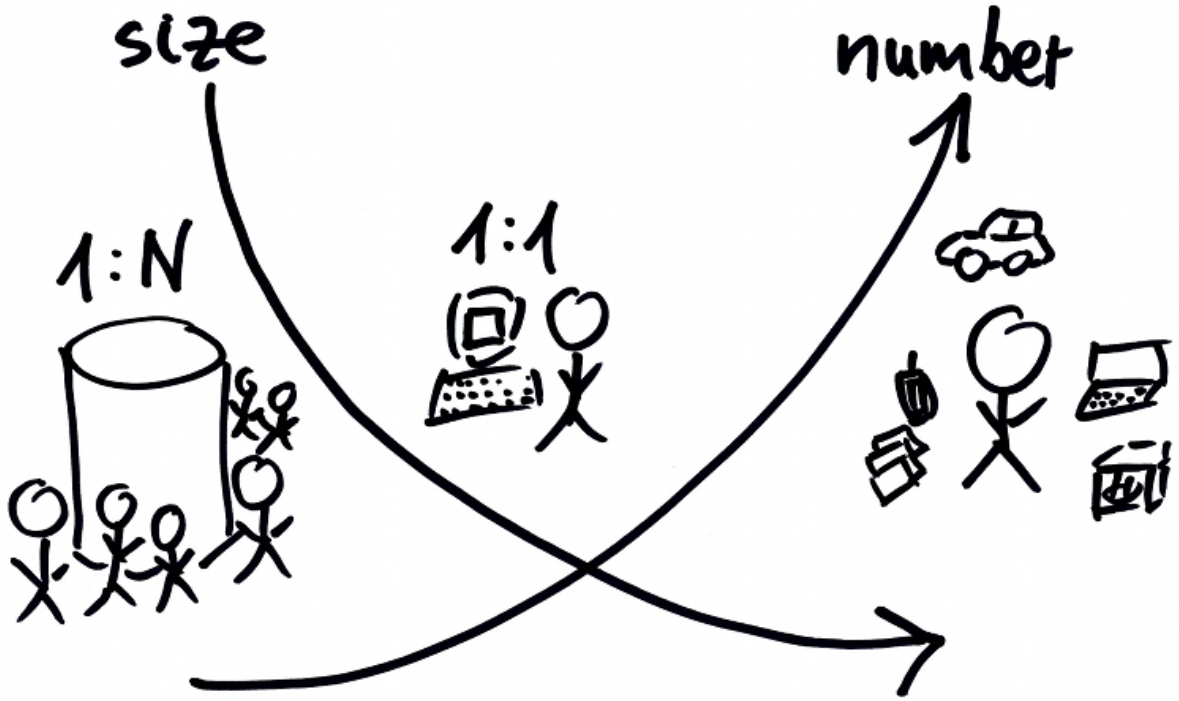
Mobile Interaction Lab, LMU München

Mobile Interaction is Usage in Context

- Primary real-world task

Adapted from a slide of Albrecht Schmidt at T-Labs

Ubiquitous Computing



- Computers embedded in everyday things
- Technology moves into the background
- Computers in the world, instead of world in the computer


INFORMATION

iPhone Sandwich Pressure Input

Michael Rohs, Sven Kratz
Deutsche Telekom
Laboratories, TU Berlin

Mobile Application Development

- Who of you owns a mobile phone?
  - Is it possible to develop applications for this device?
  - Which platforms are supported?
  - Which programming languages and tools can be used?
  - How to install the programs on the device?

Zeitplanung

Fr Vormittag	Begrüßung, Einführung Workshop
Fr Nachmittag	Workshop
Mo Vormittag	Probevorlesungen
Mo Nachmittag	Workshop
Di Vormittag	Vertiefungsvorlesung
Di Nachmittag	Workshop
Mi Vormittag	Workshop
Mi Nachmittag	Präsentationen

Ziel: Ein Orts-basiertes Quiz

- Was macht der Benutzer? (→ „Szenario“)

- „Jan ist zu Besuch in München. Er möchte mehr über die Stadt erfahren und lädt sich das neue mobile Quiz auf sein Handy. Er geht durch die Stadt. Wenn er sich einer Sehenswürdigkeit nähert, vibriert sein Handy und zeigt ein Bild und eine Erklärung dazu. Unter dem Bild sind eine Frage und vier mögliche Antworten zu sehen. Durch Antippen der richtigen Antwort bekommt er Pluspunkte. Wählt er eine falsche Antwort aus, handelt er sich Minuspunkte ein. Wenn er genügend Punkte ergattert hat, bekommt er einen günstigeren Eintritt im Museum.“

Bonus: Orts-basiertes Quiz + Gesten

- „[...] Jan nähert sich einer Sehenswürdigkeit. Das Handy vibriert und zeigt an dass Magie detektiert wurde. Jan versucht eine der magischen Gesten einzugeben. Ein Geist erscheint und Jan bekommt ein Rätsel, das er mit einer weiteren Geste lösen kann. Nach dem richtigen Lösen schickt der Geist Jan an den nächsten Ort.“

- https://wiki.medien.ifi.lmu.de/Main/2DGestenAndroid

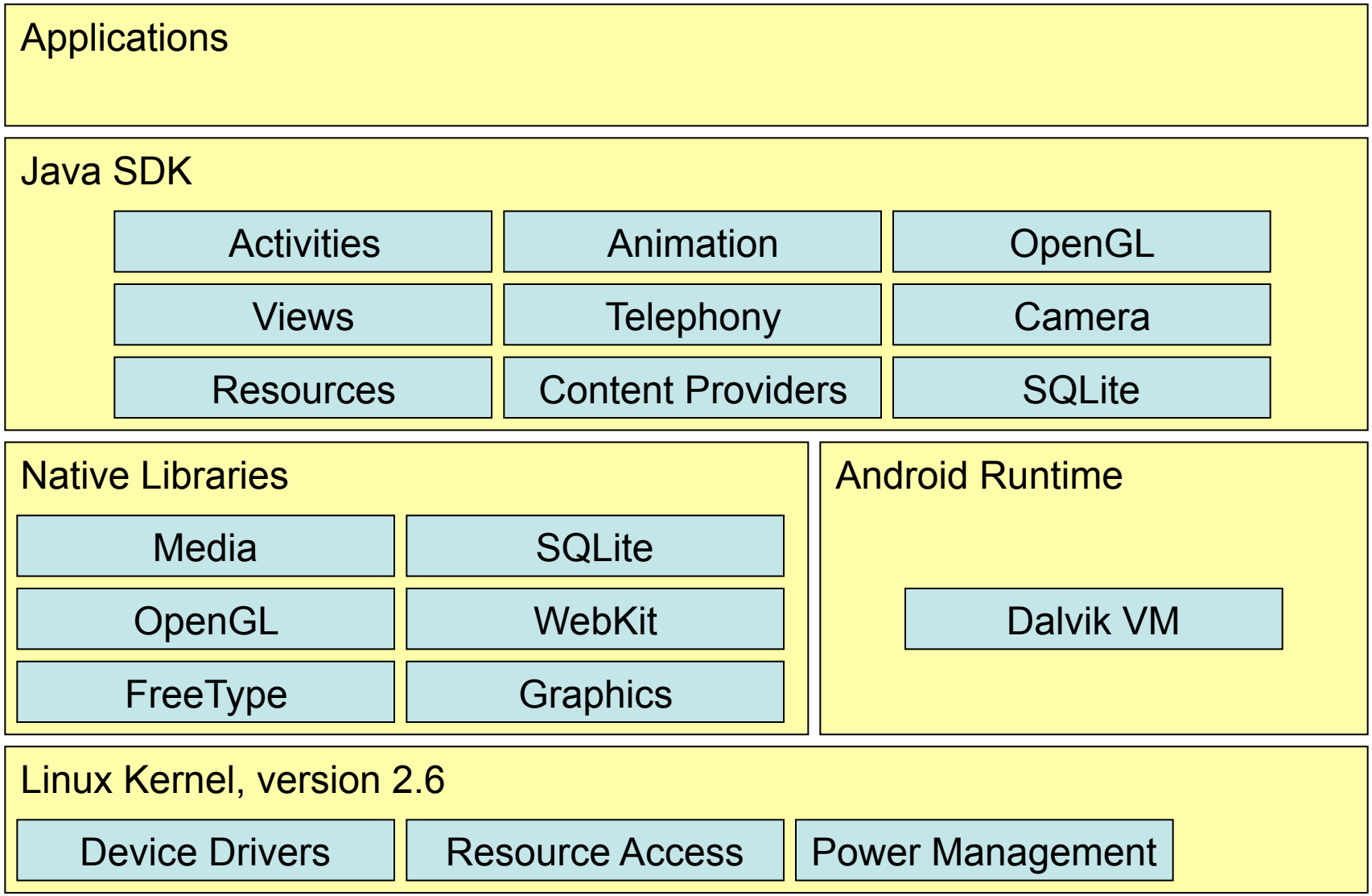
Realisierung

- Ortsinformation verarbeiten
- Bild und Text auf dem Display anzeigen
- Berührpunkt erkennen
- zwischen Bildschirmen hin- und herschalten

- Basis Eingaben vom Touch-Screen verarbeiten

Android

Android Software Stack



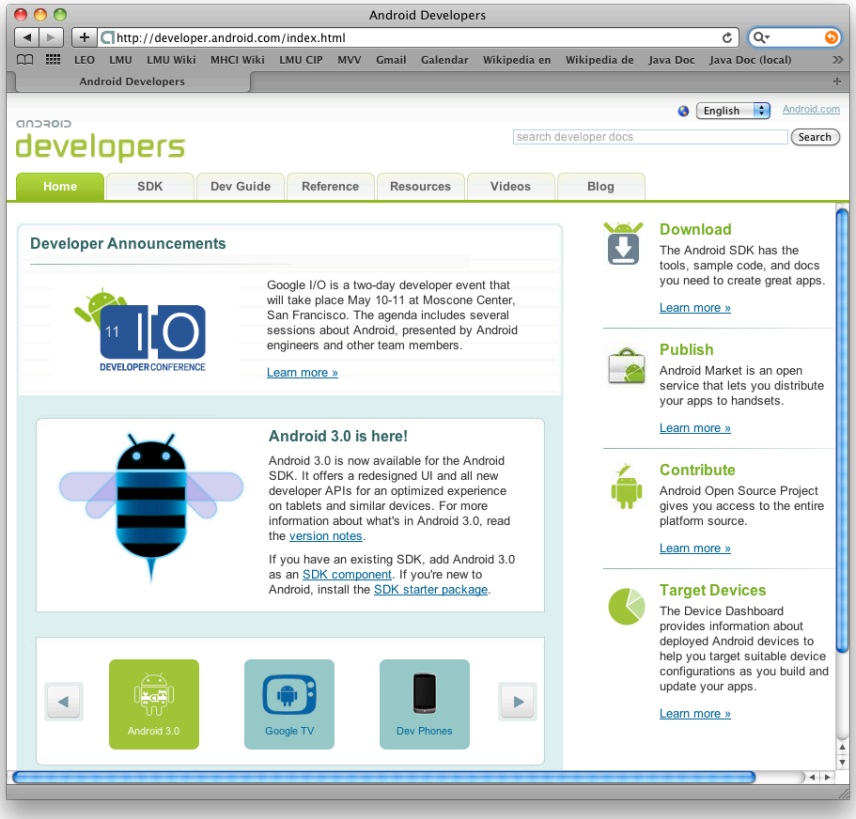
Android Characteristics

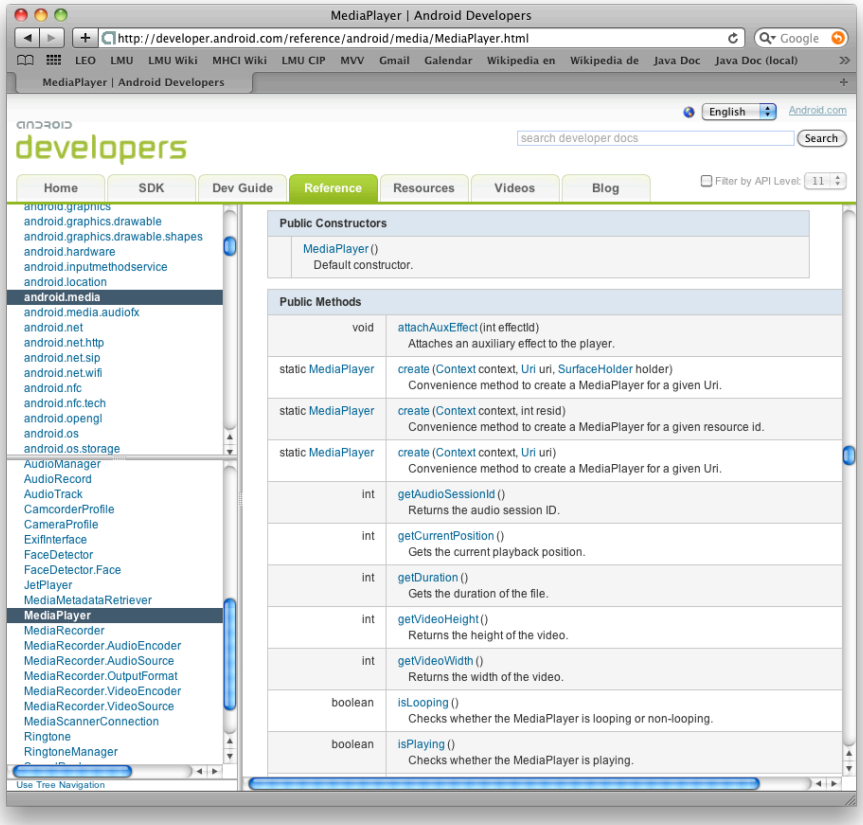
- Activity
  - Activities are the components of an application
  - Represent a logical unit of user action
  - Typically represented by a screen containing views
  - Can be invoked externally
- Declarative UI definition
  - XML files specify user interface resources
  - Resources (layout definitions, strings, bitmaps)
  - Separation of code and user interface
- User events handled programmatically

Installing Android

Android Resources

- Android developer pages (platform documentation)
  - http://developer.android.com





Installing Android (1/2)

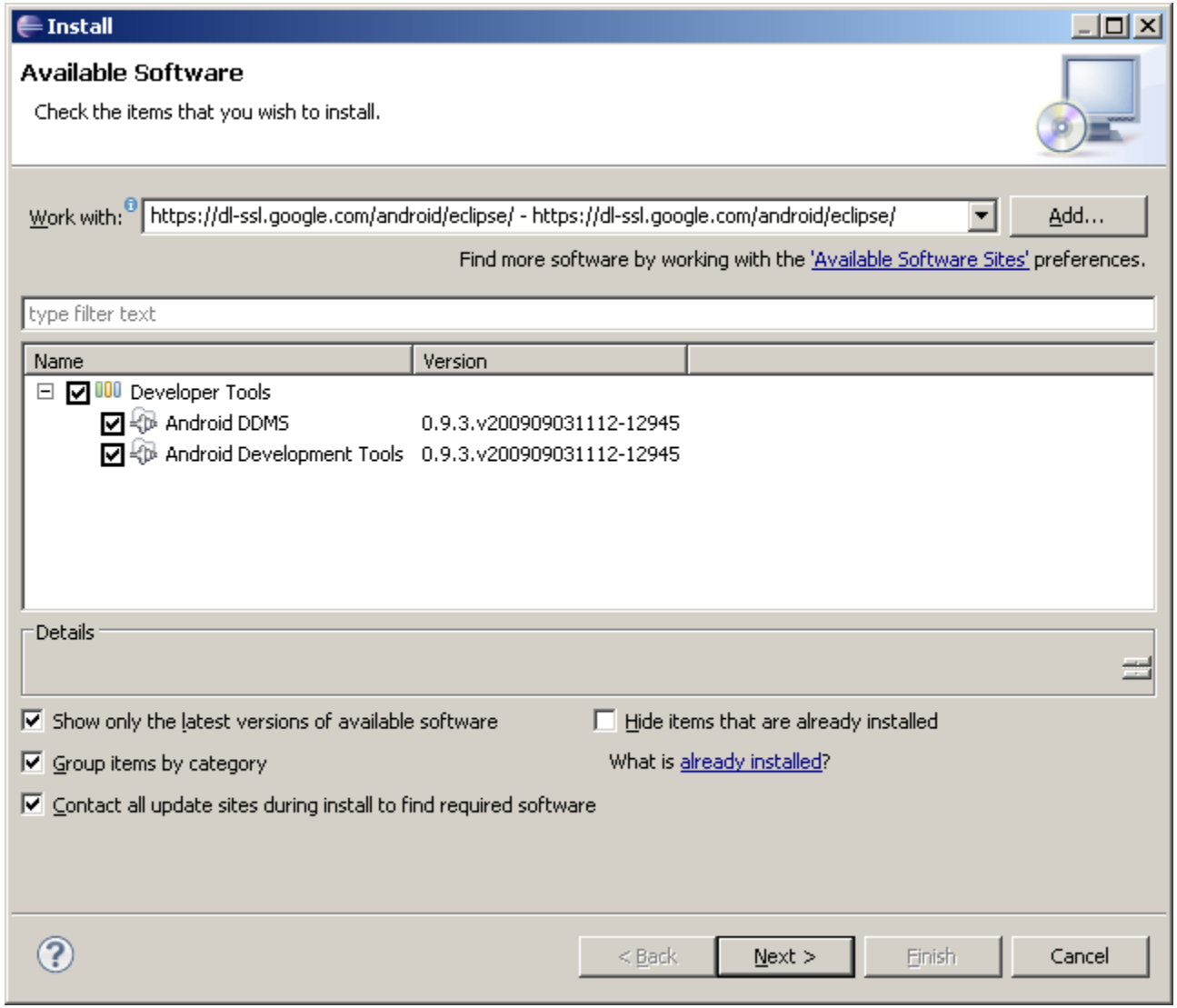
- Java JDK 6, Standard Edition (not only JRE)
  - http://java.sun.com/javase/downloads/index.jsp
- Eclipse IDE (3.4 or newer)
  - http://www.eclipse.org/downloads/
  - Eclipse IDE for Java Developers
- Android SDK starter package (depending on your platform)
  - http://dl.google.com/android/android-sdk_r08-windows.zip
  - http://dl.google.com/android/android-sdk_r08-mac_86.zip
  - http://dl.google.com/android/android-sdk_r08-linux_86.tgz
- See also: “Quick Steps”
  - http://developer.android.com/sdk/index.html

Installing Android in Eclipse (2/2)

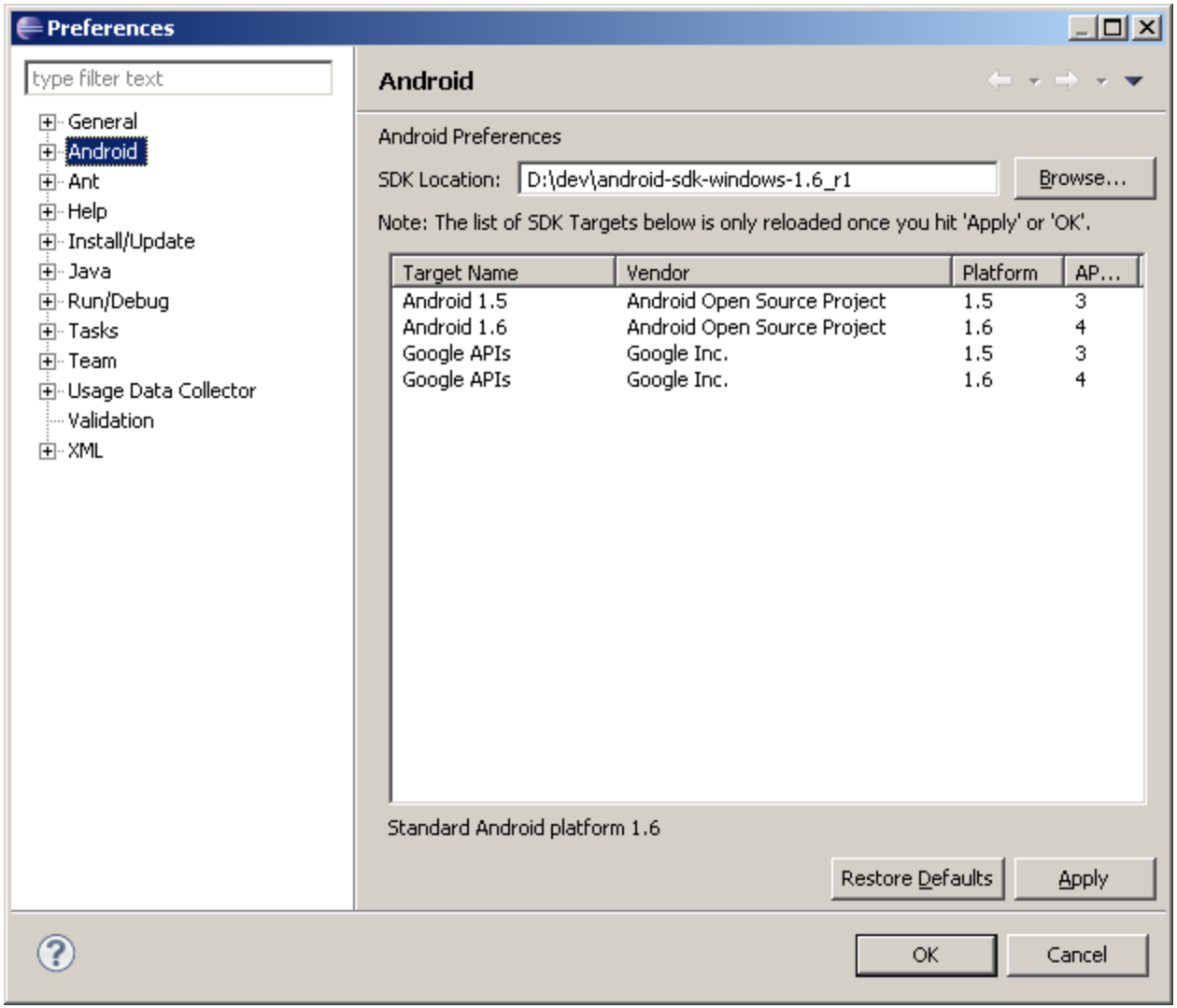
- Start Eclipse
  - Terminal oder Alt-F2: “eclipse-ide-3.6” eintippen
- In Eclipse: Install Android SDK
  - Menu: Help, Install New Software…
  - https://dl-ssl.google.com/android/eclipse/
- Point Eclipse to the Android SDK starter package
  - Menu: Window, preferences, Android, SDK Location
  - /soft/IFI/lang/android-sdk-r10/iX86-unknown-linux
- In Eclipse: Android SDK and AVD Manager
  - Window / Android SDK and AVD Manager
  - New... / Virtual Devices / 2.2 (oder 1.6) mit Google API
- Mobile Phone
  - Anwendungen, Entwicklung: USB-Debugging, ...

In Eclipse: Install New Software...

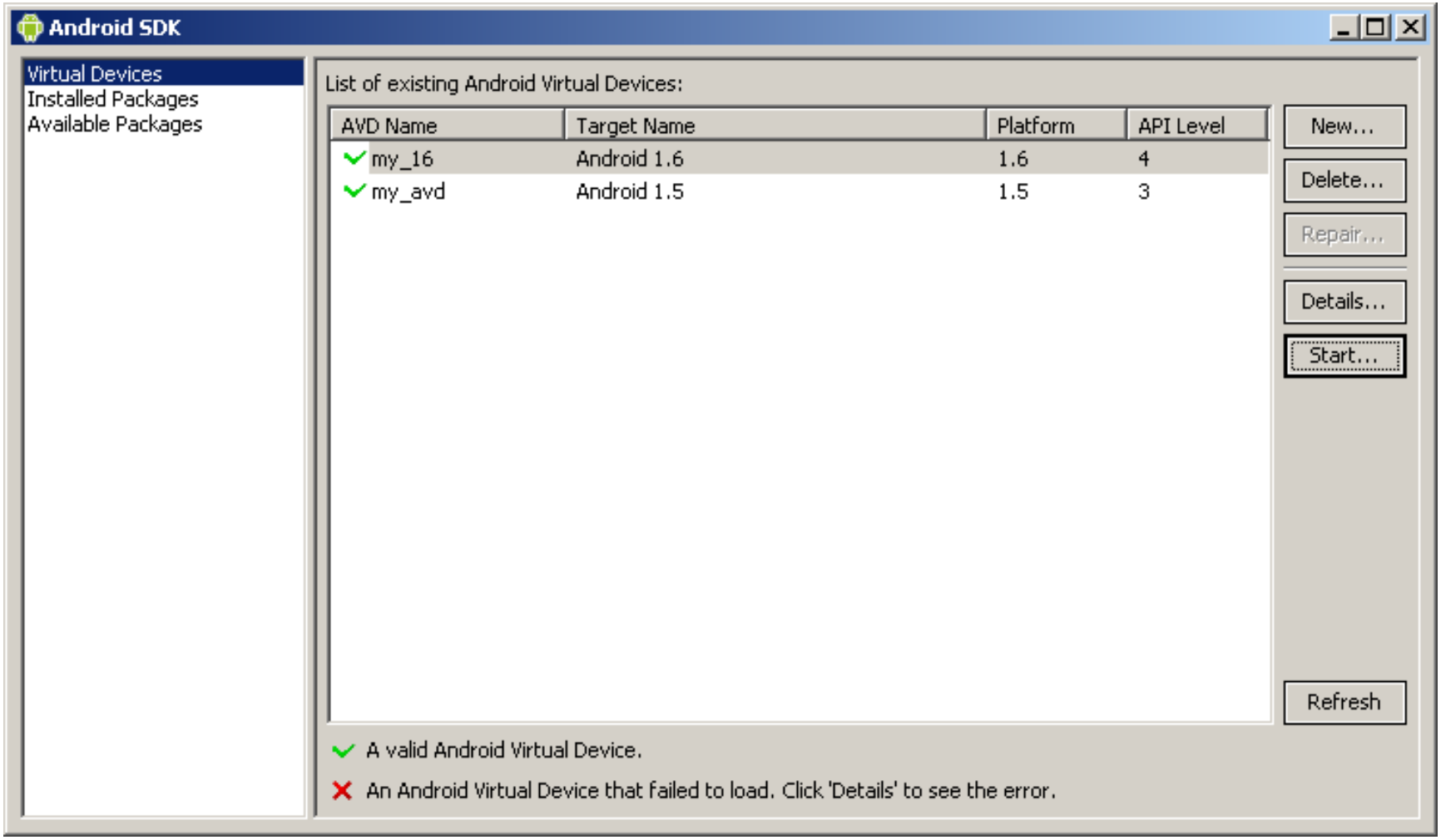
Android Plugin – https://dl-ssl.google.com/android/eclipse/



Set Path to Android SDK Starter Package



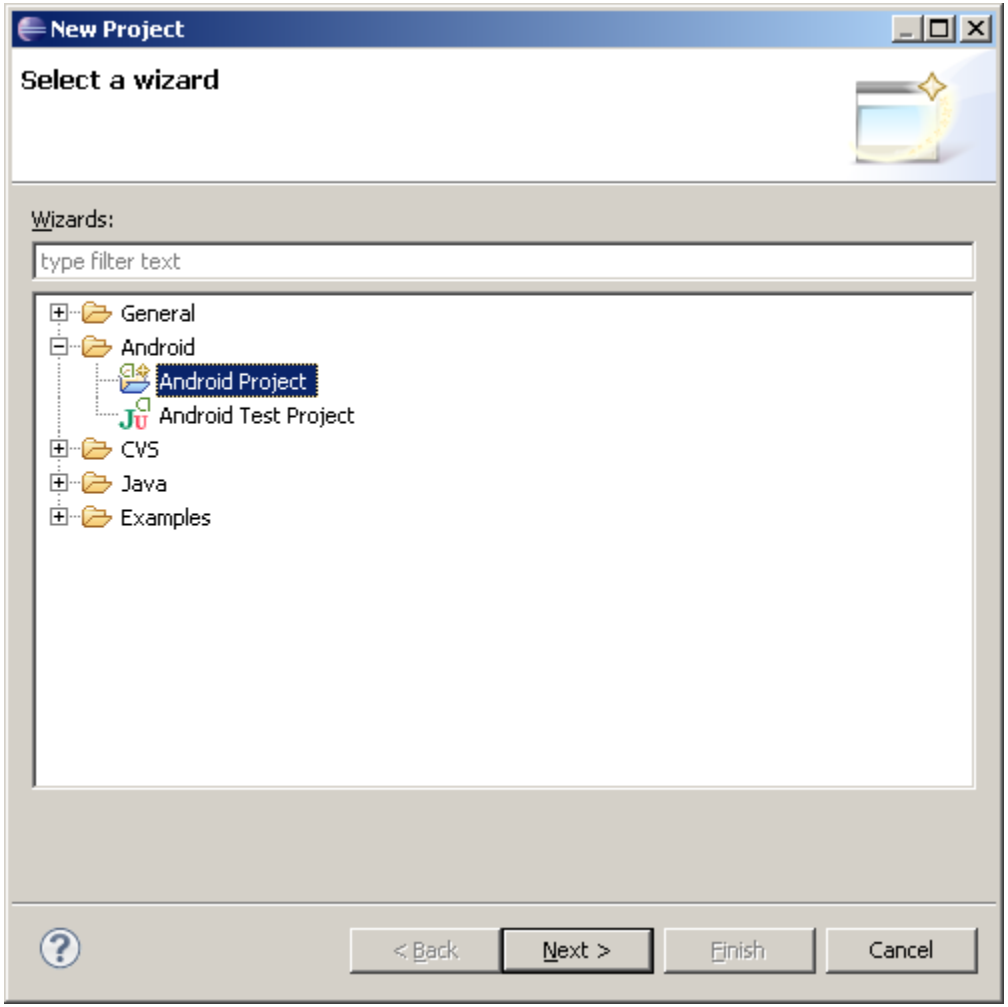
Define Android Virtual Device

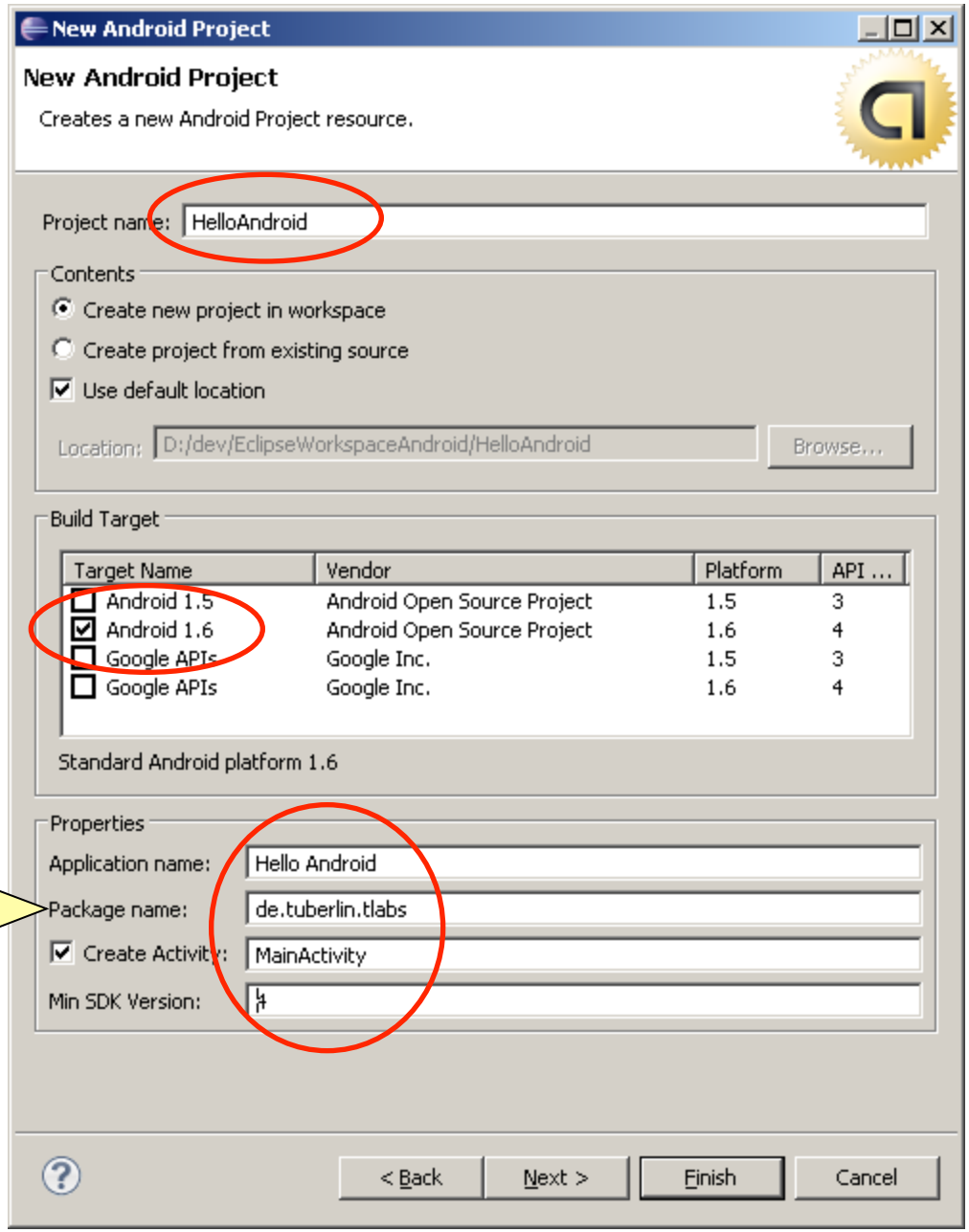


“Hello World”

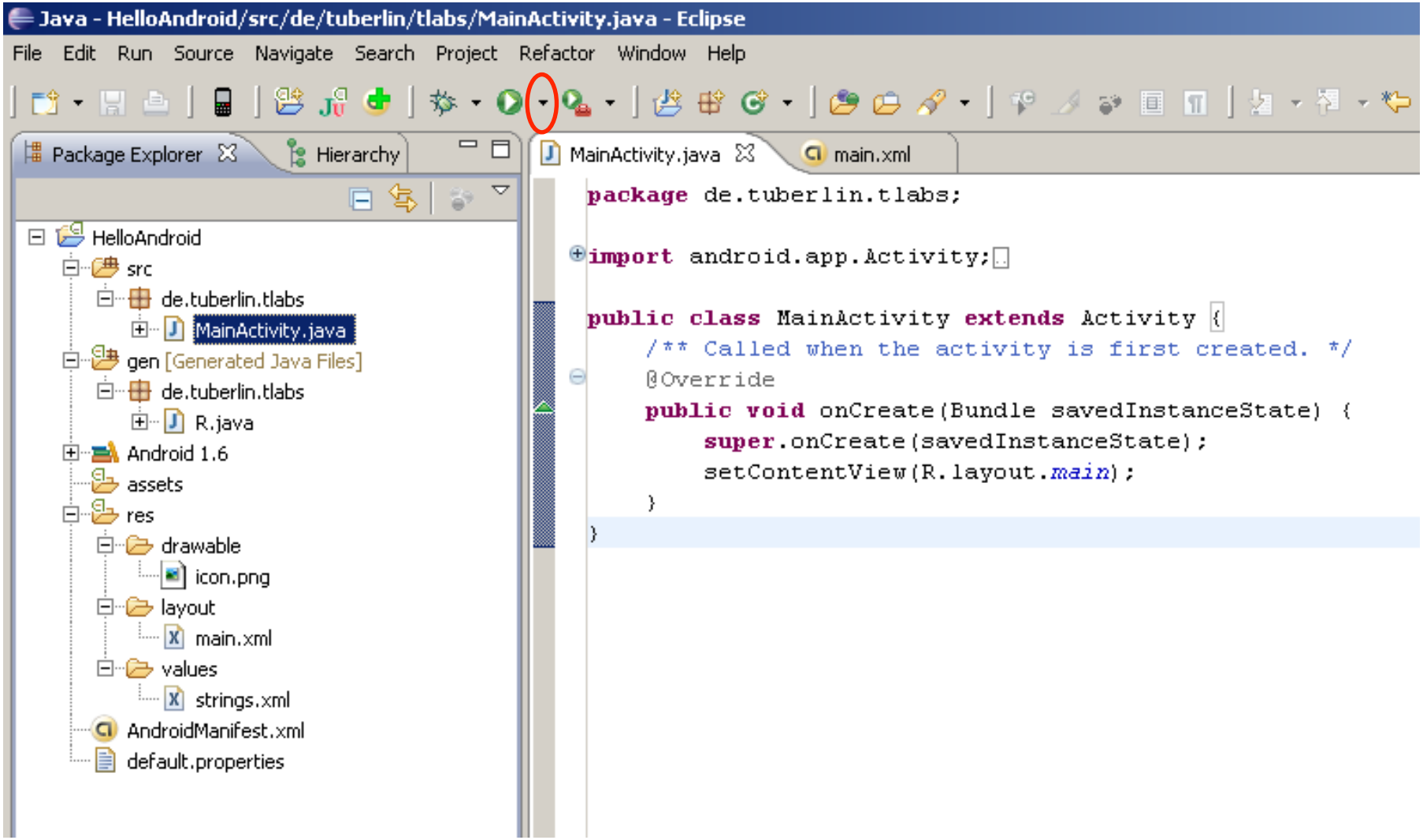
Creating Your First Android Project

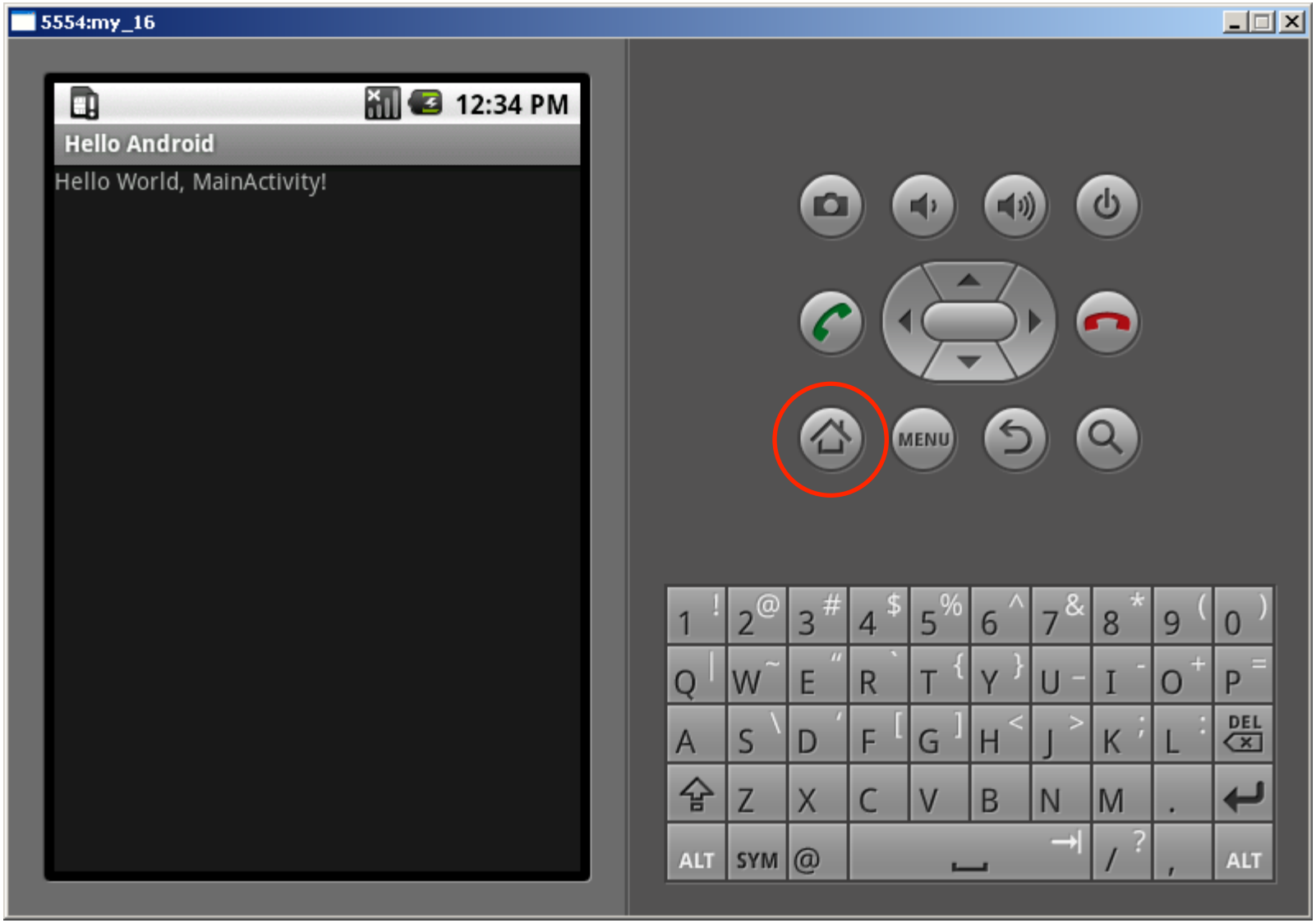
File → New Project → Android → Android Project

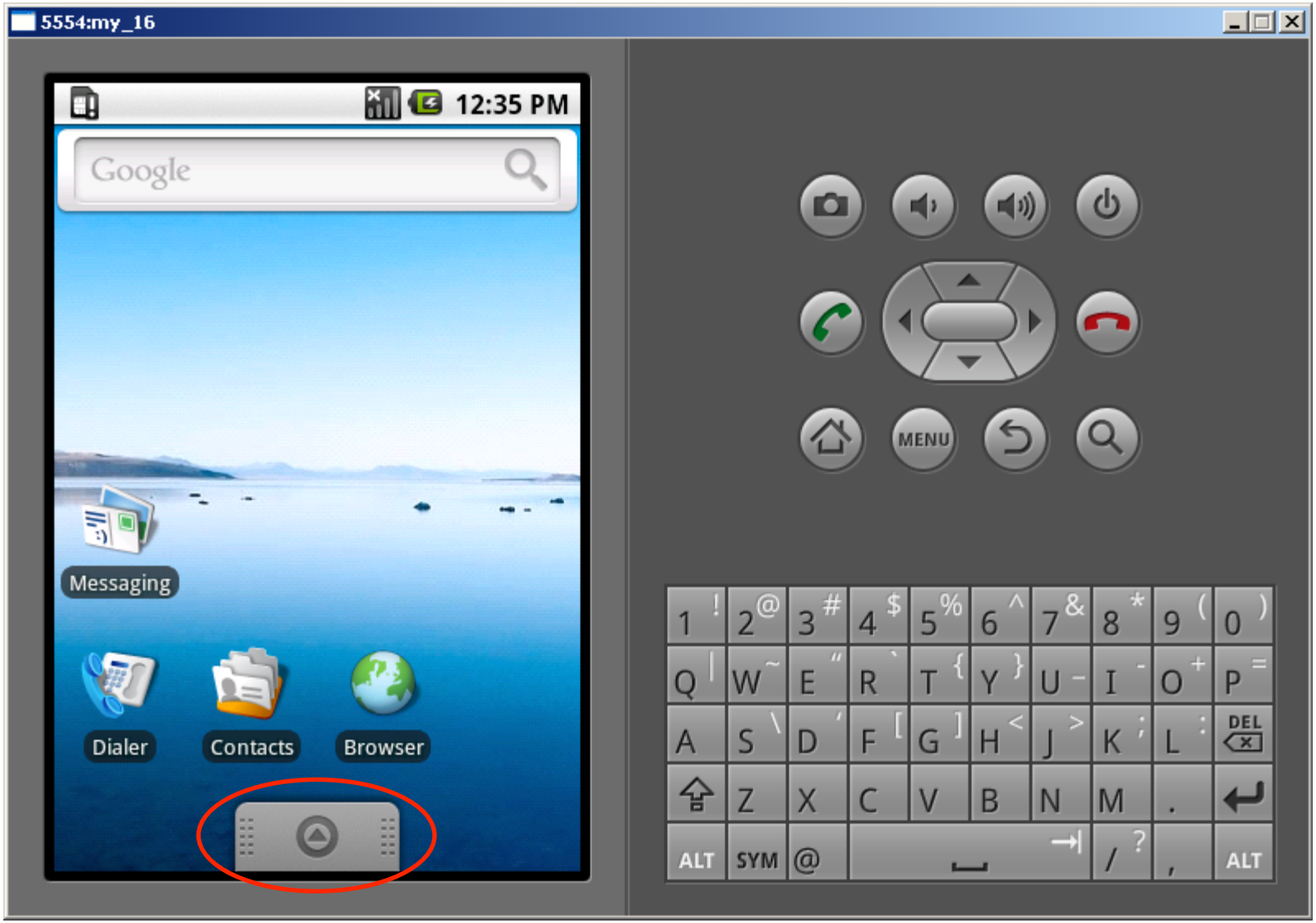


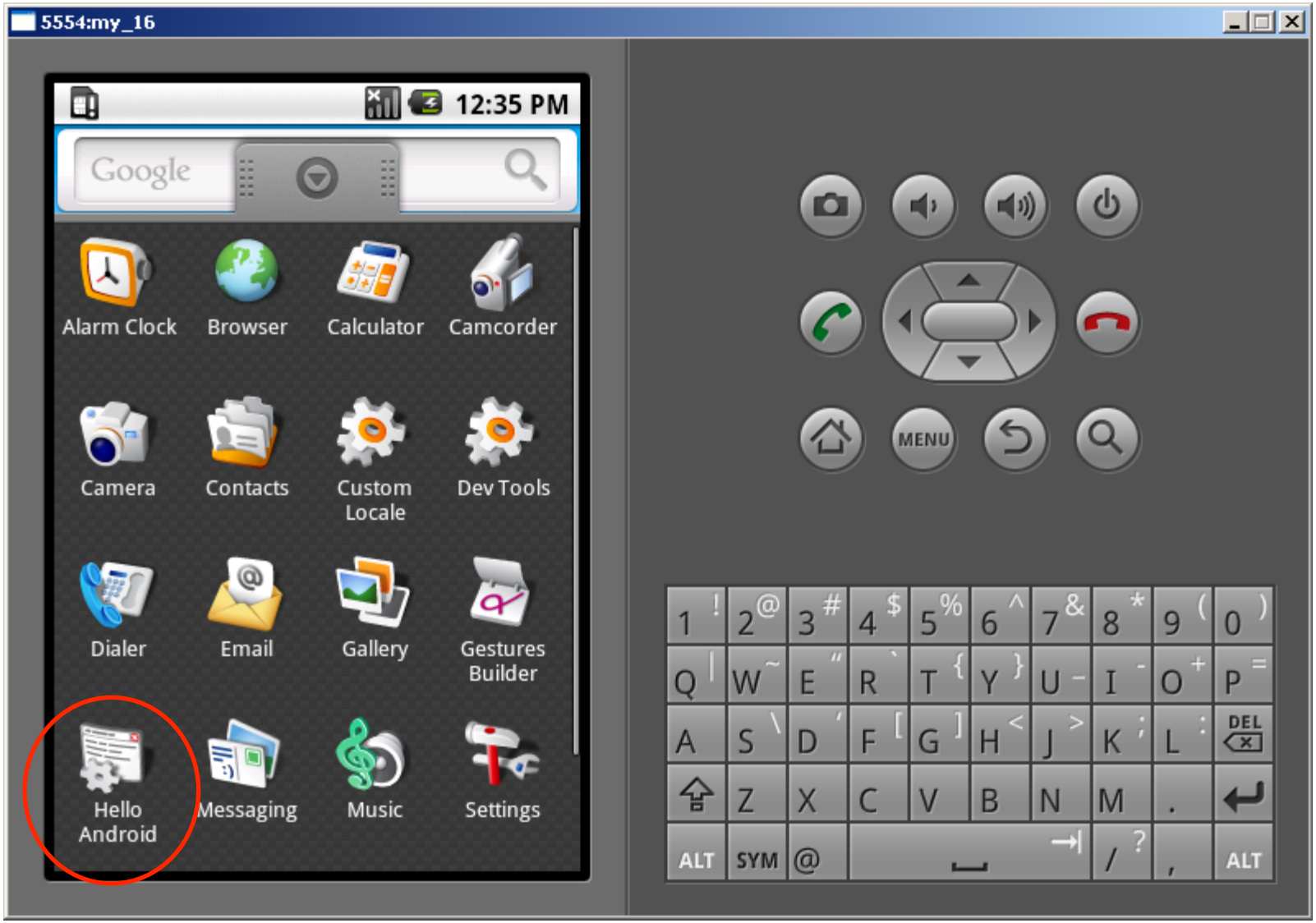


Uniquely identifies the application!


Java - HelloAndroid/src/de/tuberlin/tlabs/MainActivity.java - Eclipse
File Edit Run Source Navigate Search Project Refactor Window Help
Package Explorer
Hierarchy
HelloAndroid
src
de.tuberlin.tlabs
MainActivity.java
gen [Generated Java Files]
de.tuberlin.tlabs
R.java
Android 1.6
assets
res
drawable
icon.png
layout
main.xml
values
strings.xml
AndroidManifest.xml
default.properties
MainActivity.java
main.xml
package de.tuberlin.tlabs;
import android.app.Activity;
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);
}
}


5554:my_16
12:34 PM
Hello Android
Hello World, MainActivity!
MENU
ALT
SYM
ALT


5554:my_16
12:35 PM
Google
Messaging
Dialer
Contacts
Browser
MENU
ALT
SYM
ALT
DEL

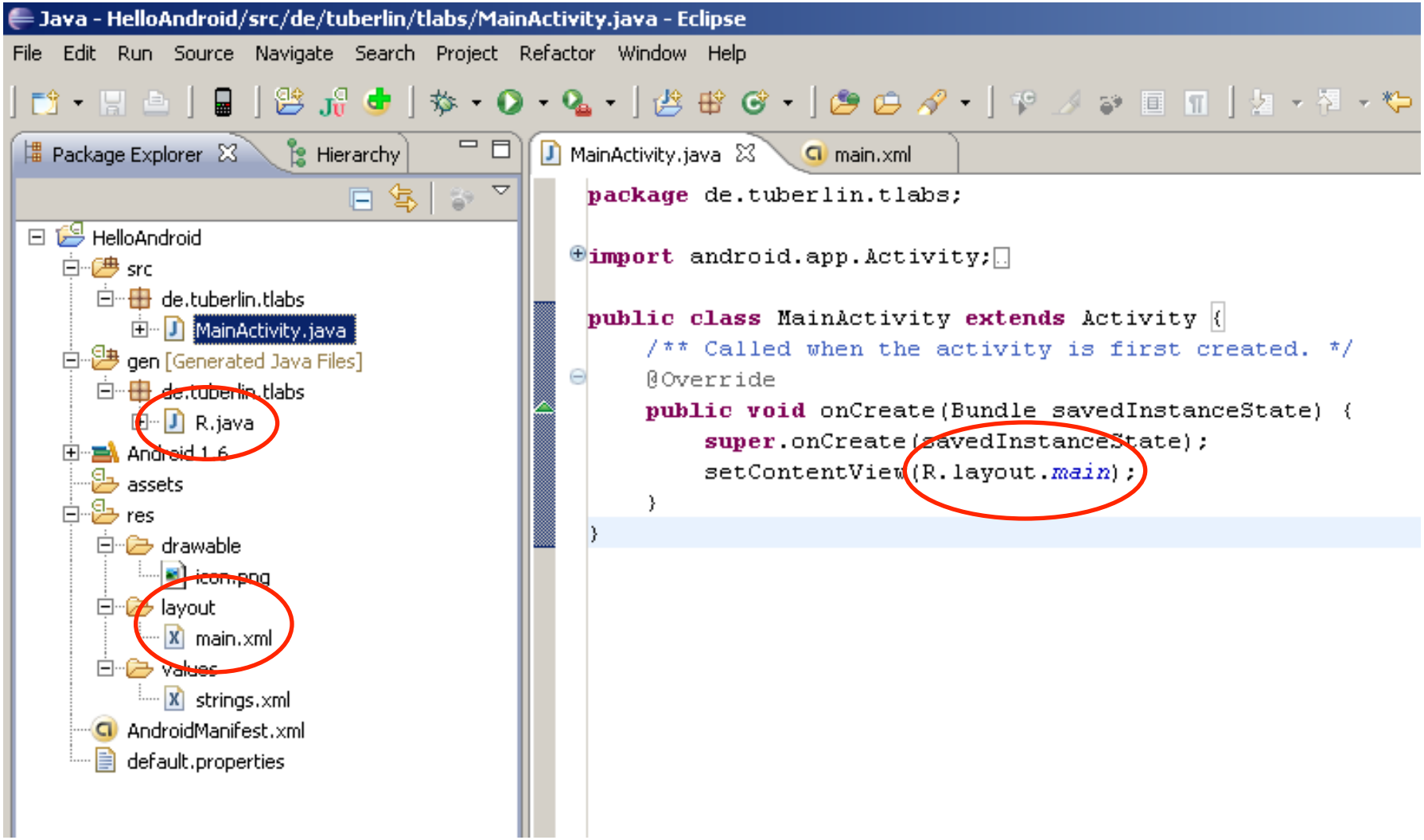

5554:my_16
12:35 PM
Google
Alarm Clock
Browser
Calculator
Camcorder
Camera
Contacts
Custom Locale
Dev Tools
Dialer
Email
Gallery
Gestures Builder
Hello Android
Messaging
Music
Settings
MENU
ALT
SYM
DEL

Exercise:

Install Android + Create “Hello World”

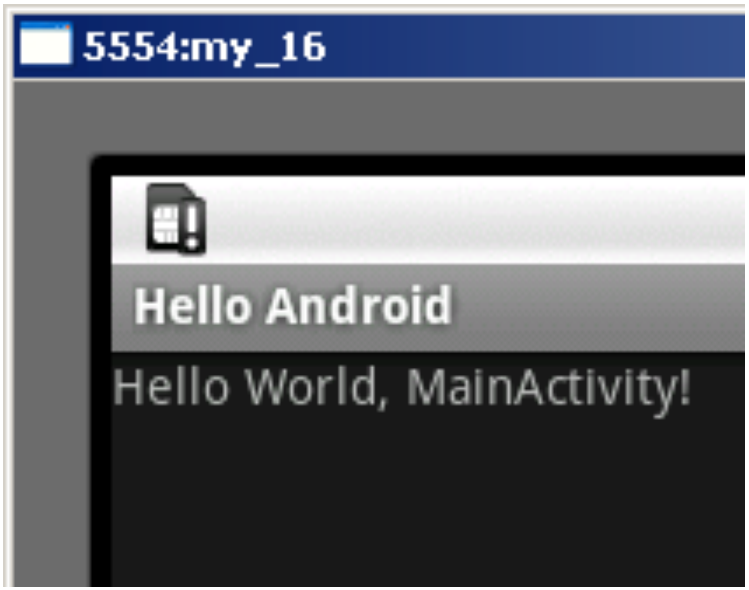
Diese Folien:

http://tiny.cc/probeandroid


Java - HelloAndroid/src/de/tuberlin/tlabs/MainActivity.java - Eclipse
File Edit Run Source Navigate Search Project Refactor Window Help
Package Explorer
Hierarchy
HelloAndroid
src
de.tuberlin.tlabs
MainActivity.java
gen [Generated Java Files]
de.tuberlin.tlabs
R.java
Android 1.6
assets
res
drawable
icon.png
layout
main.xml
values
strings.xml
AndroidManifest.xml
default.properties
MainActivity.java
main.xml
package de.tuberlin.tlabs;
import android.app.Activity;
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);
}
}

Declarative definition of UIs
main.xml

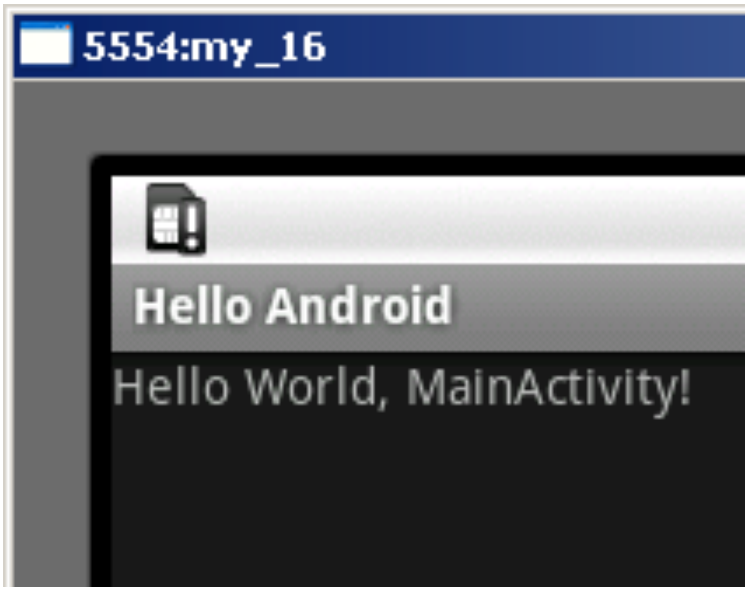
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    >
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/hello"
    />
</LinearLayout>
```



Separating text strings from source code strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello World, MainActivity!</string>
    <string name="app_name">Hello Android</string>
</resources>
```

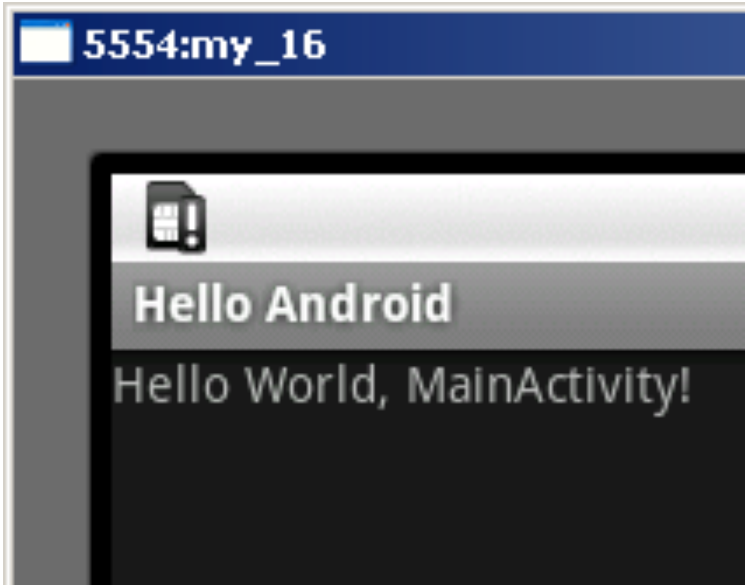
- Default language in res/values/strings.xml
- Localized languages in res/values-xx (language qualifier)
  - French in res/values-fr/strings.xml
  - Hindi in res/values-hi/strings.xml
  - etc.

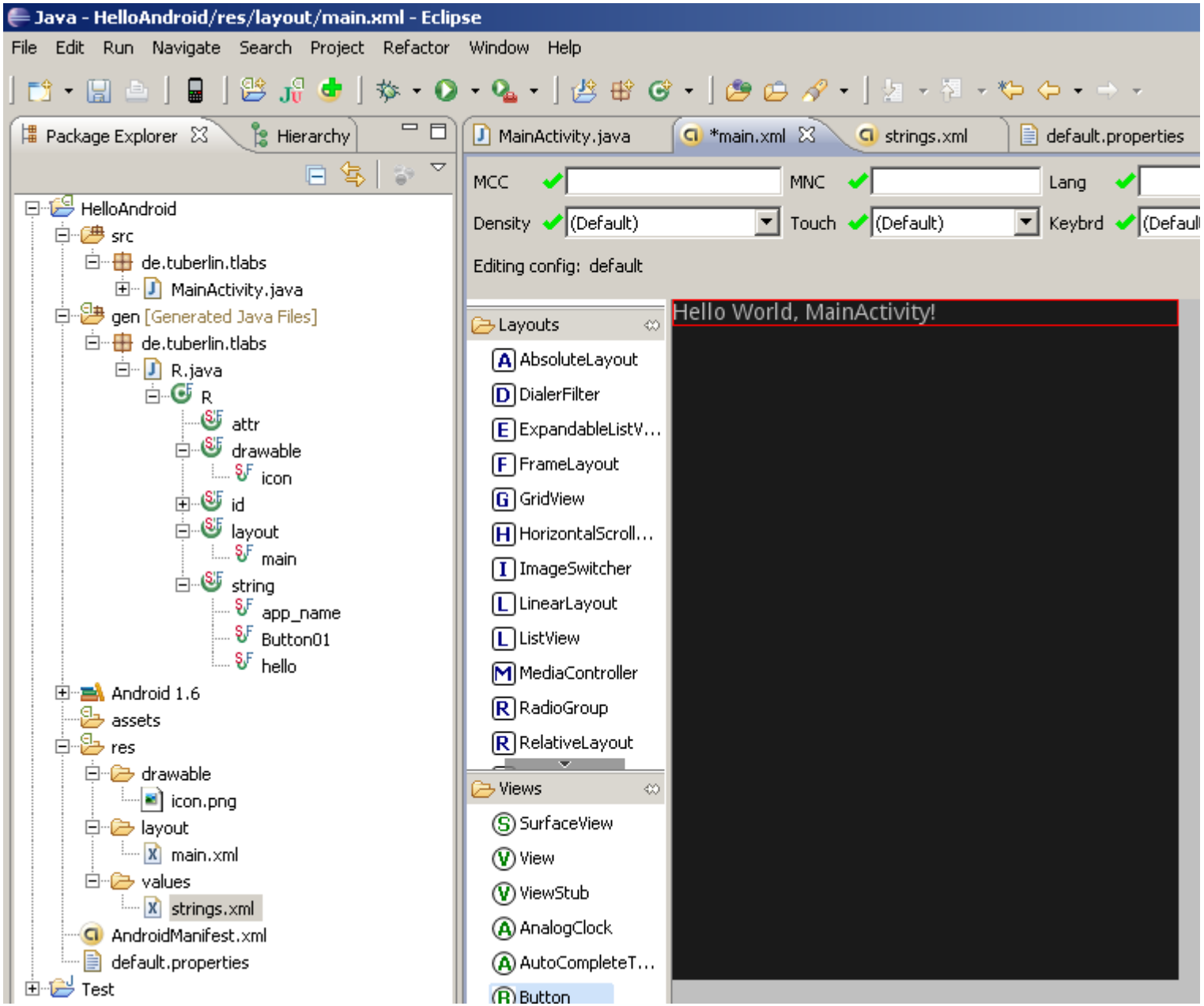


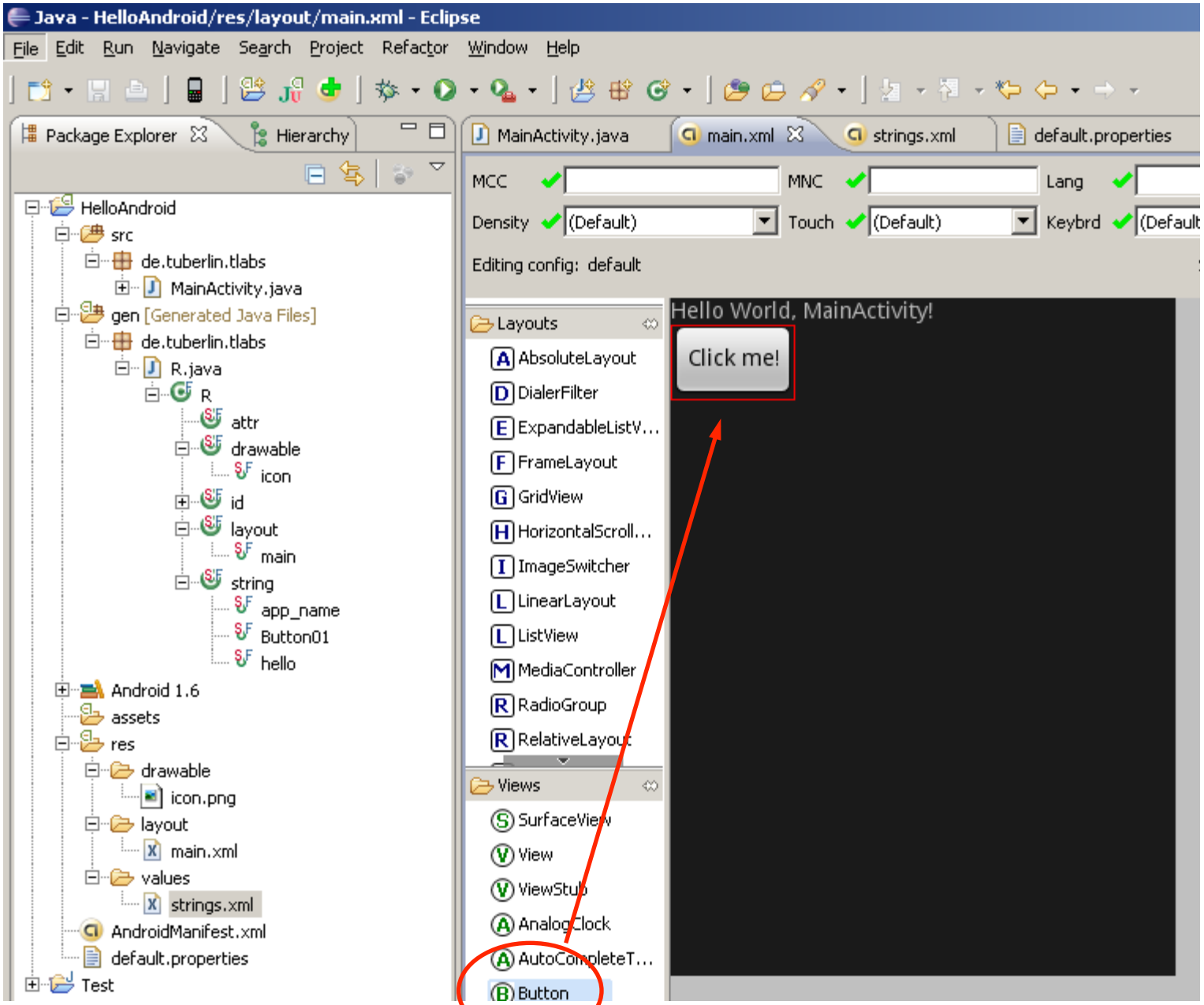
R.java

```
/* 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.tuberlin.tlabs;

public final class R {
    public static final class attr {
    }
    public static final class drawable {
        public static final int icon=0x7f020000;
    }
    public static final class id {
        public static final int Button01=0x7f050000;
    }
    public static final class layout {
        public static final int main=0x7f030000;
    }
    public static final class string {
        public static final int Button01=0x7f040002;
        public static final int app_name=0x7f040001;
        public static final int hello=0x7f040000;
    }
}
```

Never ever edit R.java!!!




Java - HelloAndroid/res/layout/main.xml - Eclipse
File Edit Run Navigate Search Project Refactor Window Help
Package Explorer
Hierarchy
MainActivity.java
*main.xml
strings.xml
default.properties
HelloAndroid
src
de.tuberlin.tlabs
MainActivity.java
gen [Generated Java Files]
de.tuberlin.tlabs
R.java
R
attr
drawable
icon
id
layout
main
string
app_name
Button01
hello
Android 1.6
assets
res
drawable
icon.png
layout
main.xml
values
strings.xml
AndroidManifest.xml
default.properties
Test
MCC
MNC
Lang
Density
(Default)
Touch
(Default)
Keybrd
Editing config: default
Layouts
AbsoluteLayout
DialerFilter
ExpandableListV...
FrameLayout
GridView
HorizontalScroll...
ImageSwitcher
LinearLayout
ListView
MediaController
RadioGroup
RelativeLayout
Views
SurfaceView
View
ViewStub
AnalogClock
AutoCompleteT...
Button
Hello World, MainActivity!

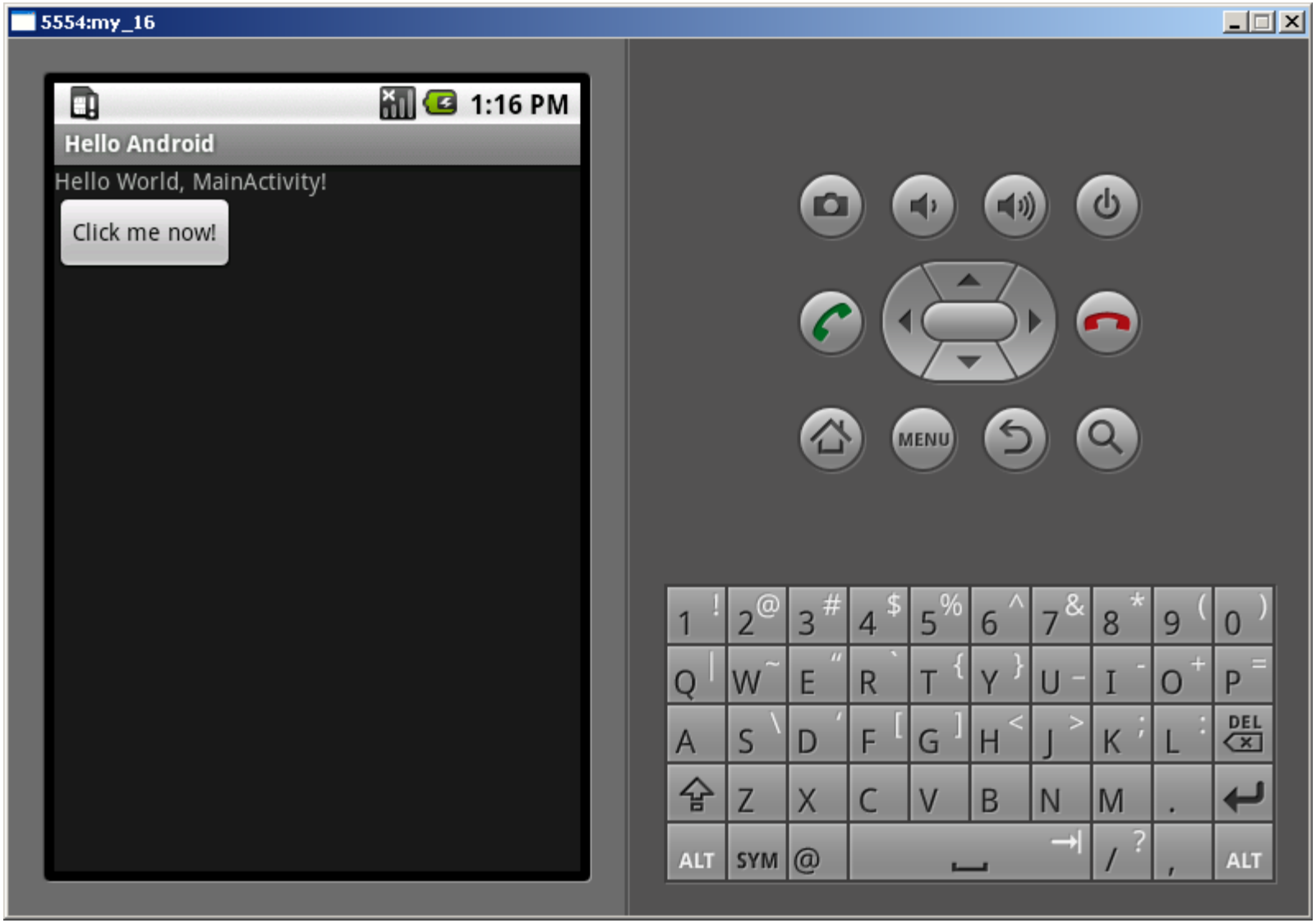

Java - HelloAndroid/res/layout/main.xml - Eclipse
File Edit Run Navigate Search Project Refactor Window Help
Package Explorer
Hierarchy
MainActivity.java
main.xml
strings.xml
default.properties
HelloAndroid
src
de.tuberlin.tlabs
MainActivity.java
gen [Generated Java Files]
de.tuberlin.tlabs
R.java
R
attr
drawable
icon
id
layout
main
string
app_name
Button01
hello
Android 1.6
assets
res
drawable
icon.png
layout
main.xml
values
strings.xml
AndroidManifest.xml
default.properties
Test
MCC
MNC
Lang
Density
(Default)
Touch
(Default)
Keybrd
Editing config: default
Layouts
AbsoluteLayout
DialerFilter
ExpandableListV...
FrameLayout
GridView
HorizontalScroll...
ImageSwitcher
LinearLayout
ListView
MediaController
RadioGroup
RelativeLayout
Views
SurfaceView
View
ViewStub
AnalogClock
AutoCompleteT...
Button
Hello World, MainActivity!
Click me!

Declarative Definition of UIs
main.xml

```
<?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="wrap_content"
    android:text="@string/hello"
    />
<Button
    android:text="@string/Button01"
    android:id="@+id/Button01"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    />
</LinearLayout>
```

strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello World, MainActivity!</string>
    <string name="app_name">Hello Android</string>
    <string name="Button01">Click me now!</string>
</resources>
```


5554:my_16
1:16 PM
Hello Android
Hello World, MainActivity!
Click me now!
MENU

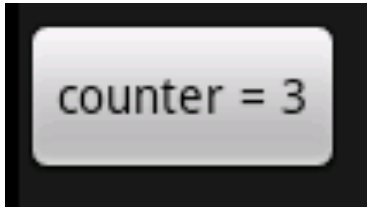
Handling Button Click Events

- XML

```
<Button android:id="@+id/button1" android:text="Basic Button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />
```

- Java

```
public class MainActivity extends Activity implements
                                                View.OnClickListener {
    public void onCreate(Bundle savedInstanceState) {
        ...
        Button b = (Button) findViewById(R.id.button1);
        b.setOnClickListener(this);
    }

    private int counter = 0;

    public void onClick(View v) {
        Button b = (Button)v;
        b.setText("counter = " + (++counter));
    }
}
```



Exercise:

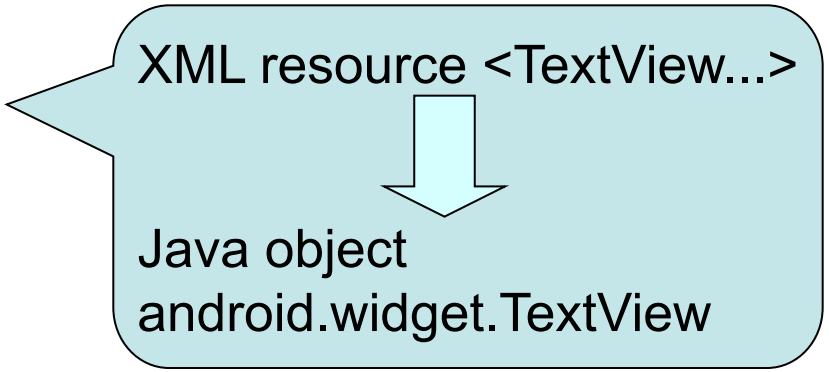
- Add a button to “Hello World”

UI from XML resources
MainActivity.java

```
import android.app.Activity;
import android.os.Bundle;

public class MainActivity extends Activity {

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
}
```

UI programmatically defined MainActivity.java

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

public class MainActivity extends Activity {

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
//        setContentView(R.layout.main);
        TextView tv = new TextView(this);
        tv.setText("Hello World (TextView)!");
        setContentView(tv);
    }

}
```

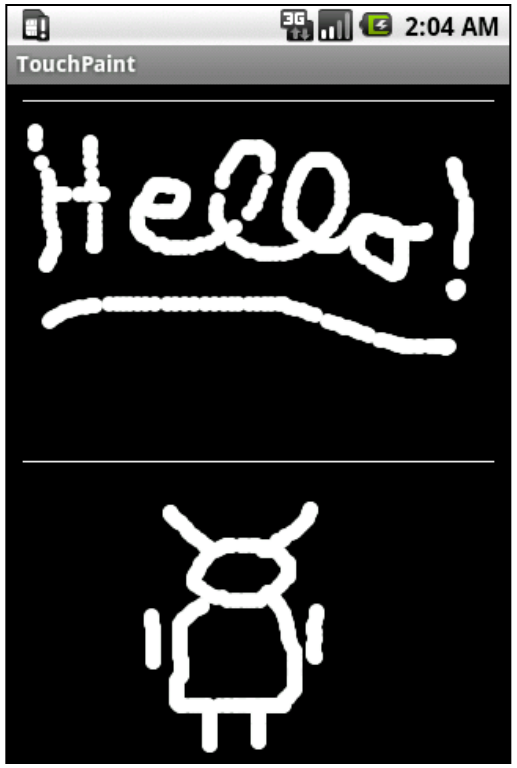


Touch Input: MotionEvent

- Method View.onTouchEvent(MotionEvent e)
- Motion event data
  - x, y, time, action, source, pressure, size
- Sources depend on hardware
  - Mouse, pen, finger, trackball
- Actions
  - ACTION_DOWN
  - ACTION_MOVE
  - ACTION_UP
  - ACTION_CANCEL
- Motion history
  - Sequence of coordinates between events

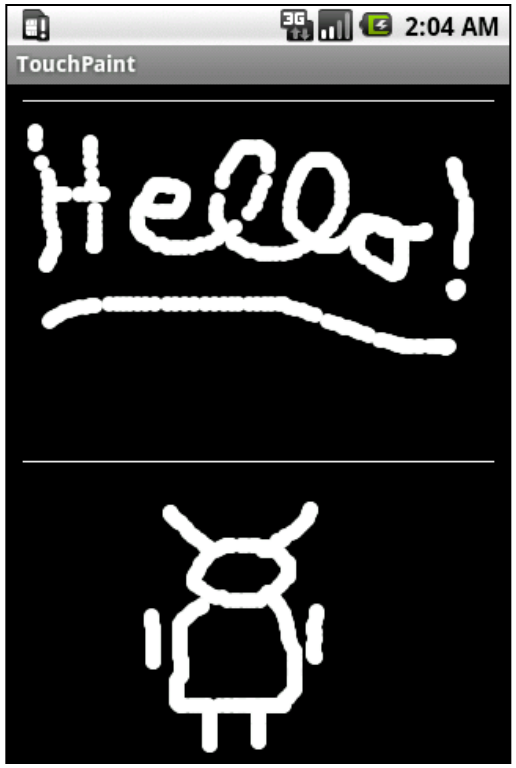
Touch Input Painting

```
public class TouchPaint extends Activity {

    private MyView myView;

    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        myView = new MyView(this);
        setContentView(myView);
    }
}
```



Touch Input Painting

```
public class MyView extends View {
    private final Paint paint = new Paint();
    private int x = 0, y = 0;

    public MyView(Context c) {
        super(c);
        paint.setARGB(255, 255, 255, 255);
    }

    protected void onDraw(Canvas c) {
        c.drawCircle(x, y, 3, paint);
    }

    public boolean onTouchEvent(MotionEvent e) {
        x = (int)e.getX(); y = (int)e.getY();
        invalidate();
        return true;
    }
}
```



Concepts so far

- Project directory structure
  - src, gen, res, AndroidManifest.xml
- Resources
  - Declarative view definitions in XML
  - Localization of string resources
  - Resource identifiers
- Touch input
  - Motion events

Activities

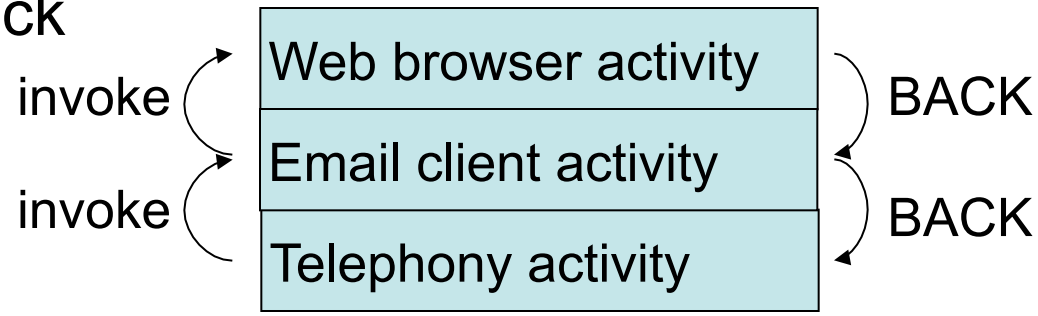
- Independent components of the application
  - Components “crash” individually
- Represent data and behavior of one View
  - Roughly: the model and controller of the MVC pattern
- Example: text messaging application
  - Activity 1 shows list of contacts
  - Activity 2 to write a message to a chosen contact
  - Activity 3 to review sent messages
- View of an Activity typically fills the screen
  - Views grouped in hierarchy
  - Parents control layout of children
  - Leaf view react to user actions
  - Associate root view with activity: activity.setContentView(view id);

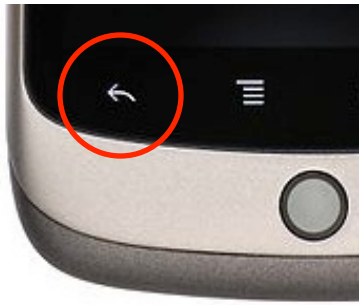
Activity Lifecycle

- Managed by system based on resources and user needs
- States
  - Running: in foreground (at top of activity stack)
  - Paused: partially visible, lost focus (e.g. dialog on top)
  - Stopped: invisible
- Lifecycle callback methods of an Activity
  - **protected void** onCreate(Bundle savedInstanceState);
  - **protected void** onStart();
  - **protected void** onRestart();
  - **protected void** onResume();
  - **protected void** onPause();
  - **protected void** onStop();
  - **protected void** onDestroy();

Tasks

- Task: what the user experiences as an “application”
  - Notion of an “application” blurry in component-based system
  - Tasks can span multiple activities and applications
- Example scenario for a task
  - User talks on the phone, looks up an email to answer a question, follows a link to a Web page with the desired information
  - Talk on phone: telephony application
  - Look up email: email client
  - Reading Web page: web browser
- Activity stack of a task:





Intents

- Intents are
  - Messages to the system
  - (Passive) representations of an operation to be performed
  - “Glue” between activities
  - Enable late runtime binding across applications
- Primary pieces: action and data
  - Example: action: ACTION_VIEW, data: URI to view
- Intents used to
  - Invoke other applications
  - Represent actions to be performed in the future
  - Register for events (→ publish-and-subscribe)

Example: Invoking an Activity

- Activity to be invoked

```
public class BasicActivity extends Activity {
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
}   }
```

- In AndroidManifest.xml

```
<activity android:name="BasicActivity" android:label="My Basic Activity">
    <intent-filter>
        <action android:name="de.lmu.intent.action.ShowBasicView" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
```

- From another activity

```
Intent intent = new Intent("de.lmu.intent.action.ShowBasicView");
startActivity(intent);
```

Available Intents in Android

- Available intents
  - Browser: open a browser window
  - Dialer: calling phone numbers
  - Google Maps: open to the given location
  - Google Streetview: open to the given location

- Examples

```
Intent intent = new Intent(Intent.ACTION_VIEW);
intent.setData(Uri.parse("http://www.lmu.de"));
startActivity(intent);

Intent intent = new Intent(Intent.ACTION_VIEW);
intent.setData(Uri.parse("geo:52.5127,13.3210?z=17"));
startActivity(intent);
```

Matching Intents to Activities

- Generic action ACTION_VIEW

```
Intent intent = new Intent(Intent.ACTION_VIEW);
intent.setData(Uri.parse("http://www.lmu.de"));
startActivity(intent);
```

- Intent registration names scheme

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

Define the contents of the application AndroidManifest.xml

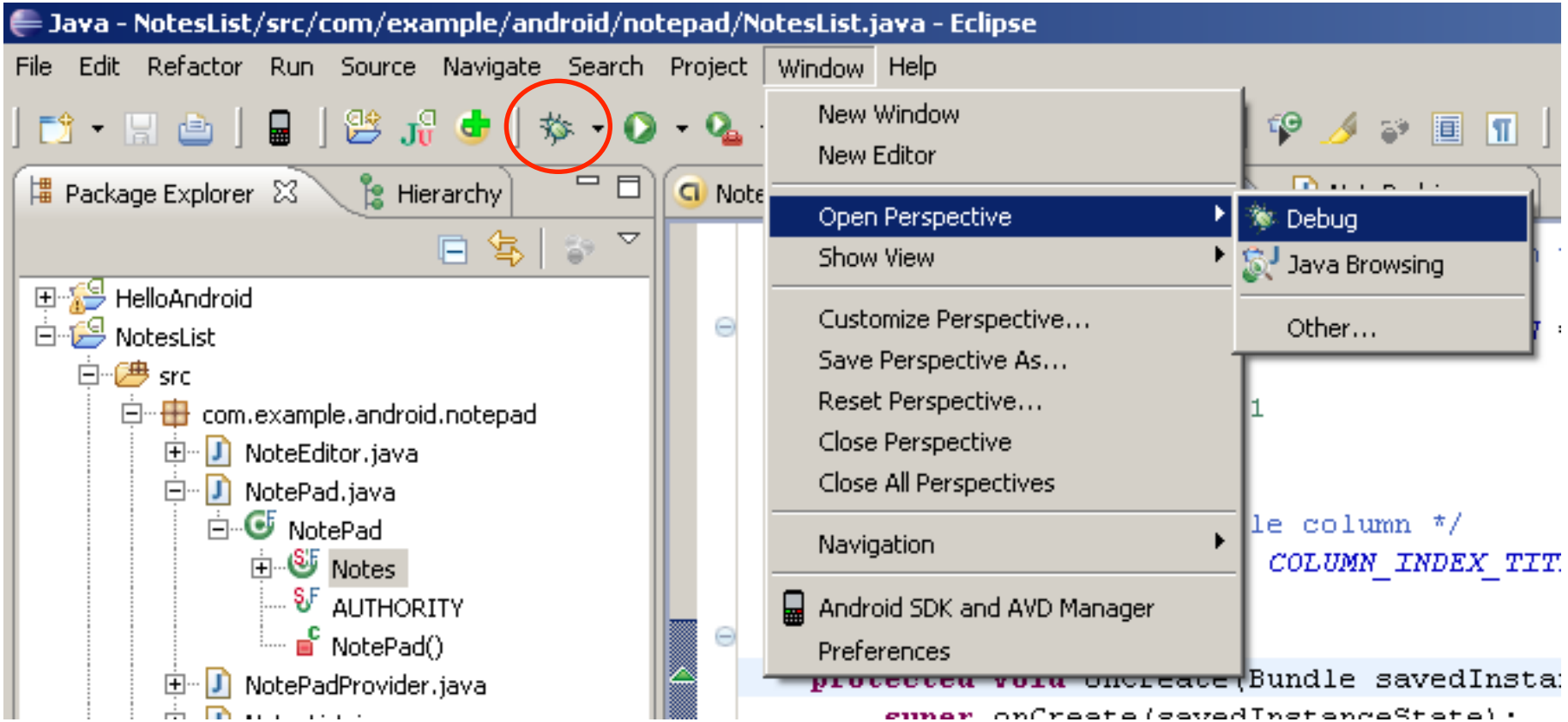
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="de.tuberlin.tlabs"
    android:versionCode="1"
    android:versionName="1.0">
  <application android:icon="@drawable/icon" android:label="@string/app_name">
    <activity android:name=".MainActivity“android:label="@string/app_name">
      <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
    </activity>
  </application>
  <uses-sdk android:minSdkVersion="4" />
</manifest>
```

Uniquely identifies the application!

Add for android:debuggable="true" on-device debugging!

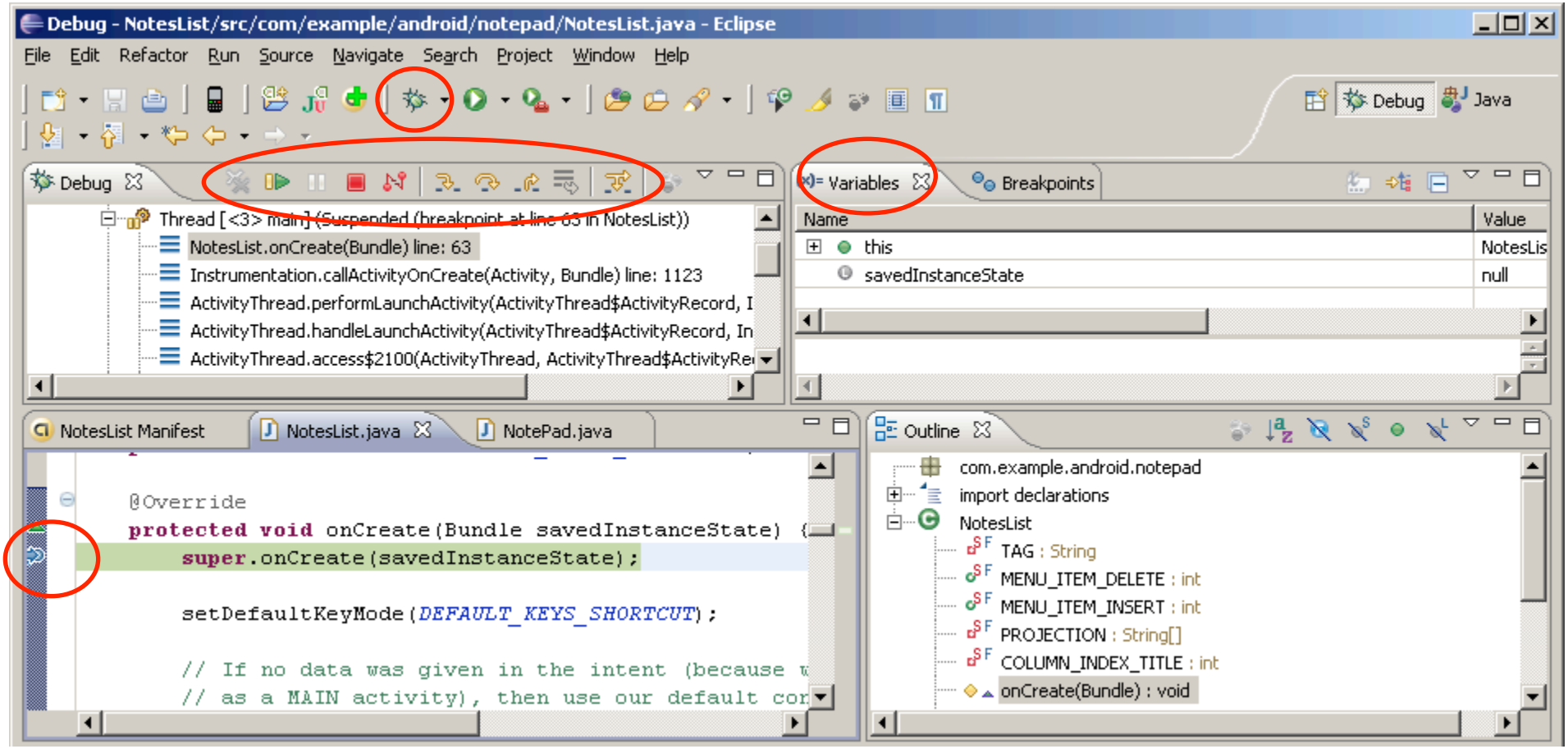
- Initial activity of application
- Listed in application launcher

Eclipse Perspectives

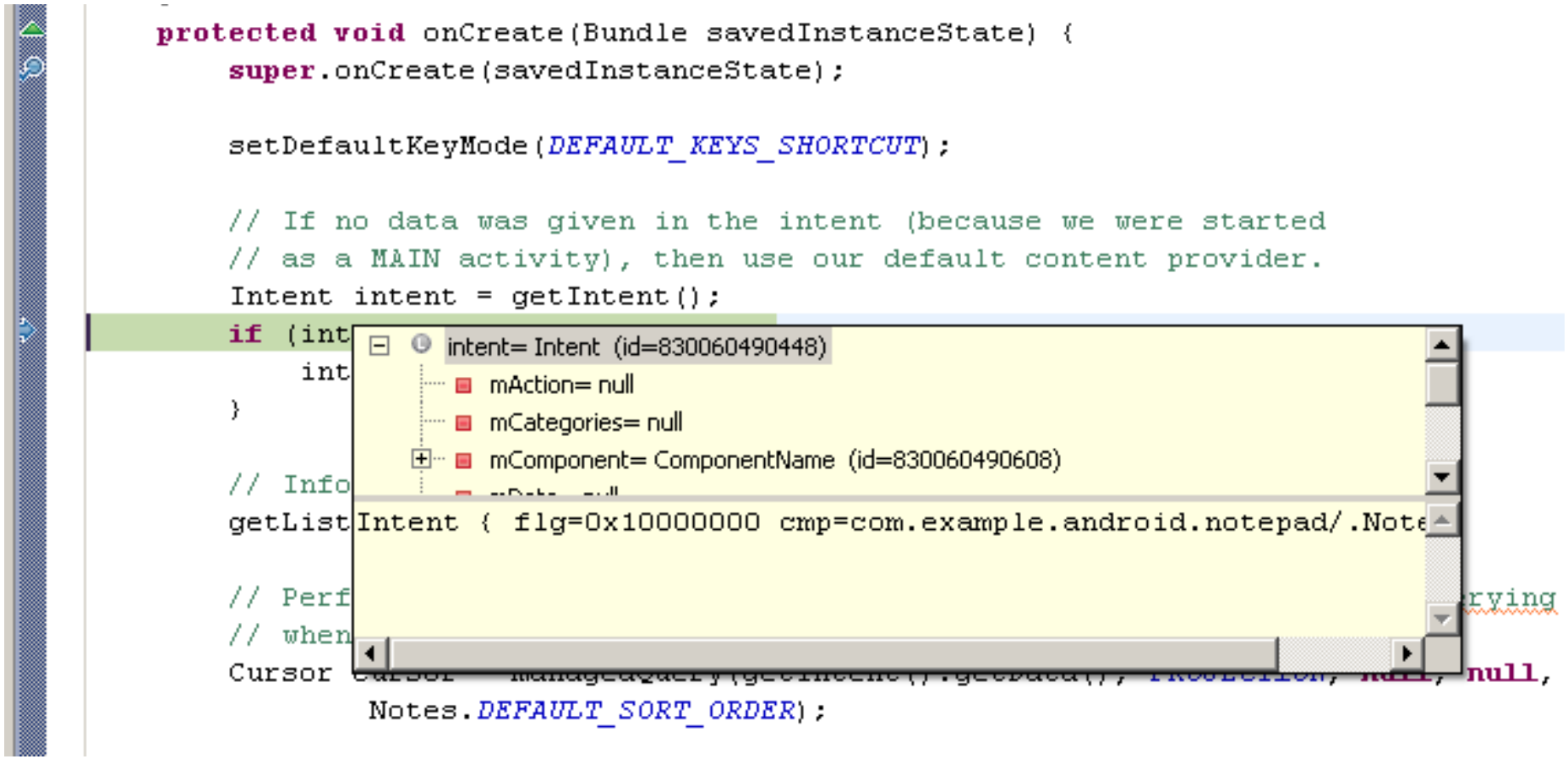


Debugging in the Emulator

- Set Breakpoint with Ctrl+Shift+B (⌘ +Shift+B)
- Step through code with F5, F6, F7 (*fn* + F5, F6, F7)



Inspecting Variables

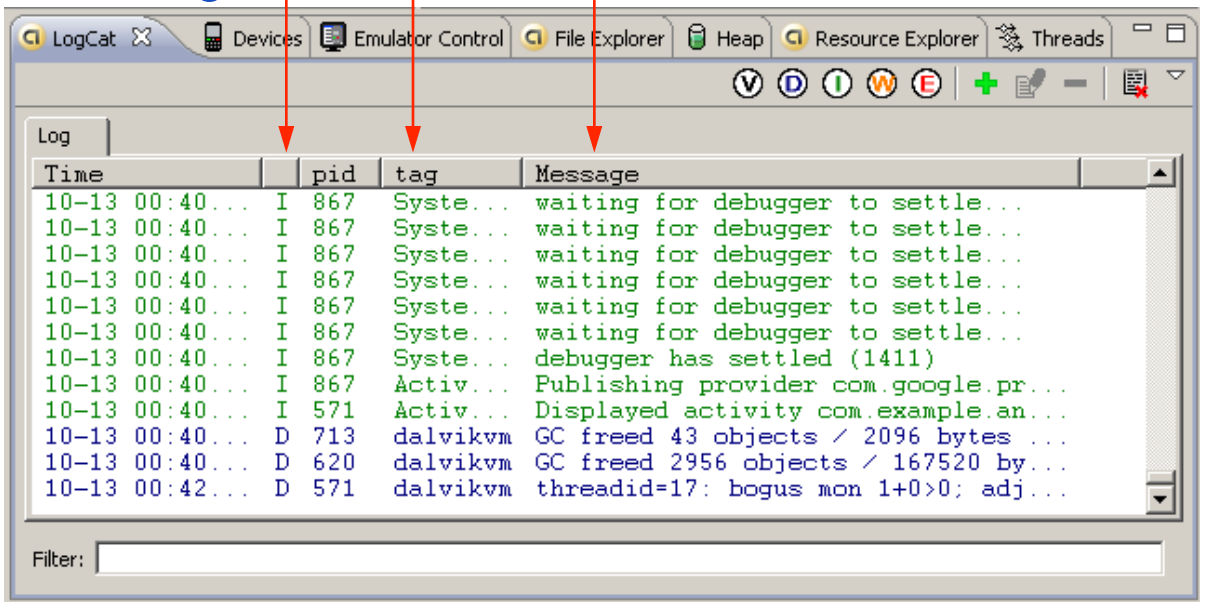


Logging and Tracing

- android.util.Log
  - informational, warning, error methods
  - Example:

Log.d(TAG, "getAddress: " + s);

- android.os.Debug
  - Debug.startMethodTracing
  - Debug.stopMethodTracing
  - trace viewer tool
- File explorer tool to view files on the device

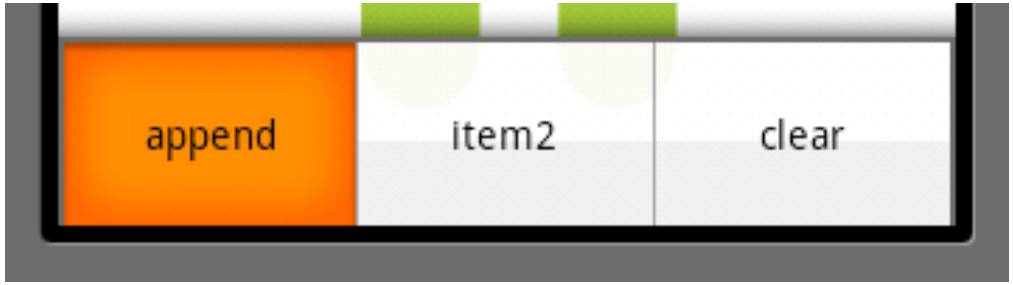


Menus

Menus

- An activity is associated with a single menu
- Use onCreateOptionsMenu(Menu m) to populate menu
- Creating an options menu

```
public boolean onCreateOptionsMenu(Menu menu) {
    super.onCreateOptionsMenu(menu);
    menu.add(0, 1, 0, "append"); // group, id, order, title
    menu.add(0, 2, 1, "item2");
    menu.add(0, 3, 2, "clear");
    return true; // return true to enable menu
}
```



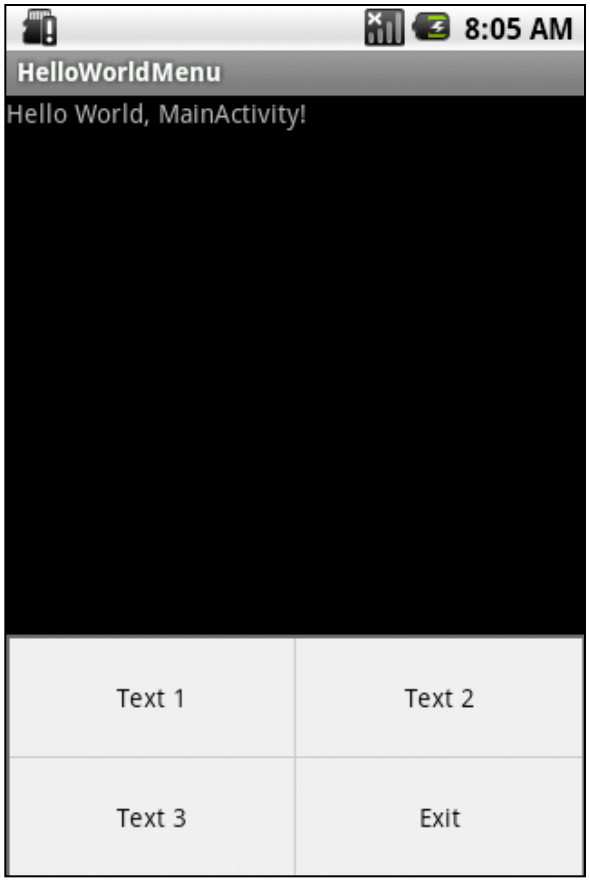
Responding to Menu Selection

- Overriding onOptionsItemSelected

```
public boolean onOptionsItemSelected(MenuItem item) {
    Log.d("MainActivity", "menu id = " + item.getItemId() +
            ", title = " + item.getTitle().toString());
    switch (item.getItemId()) {
        case X: // id of handeled item
        // handle item X
        return true;
        ...
    }
}
```

Exercise: A Menu for Hello World

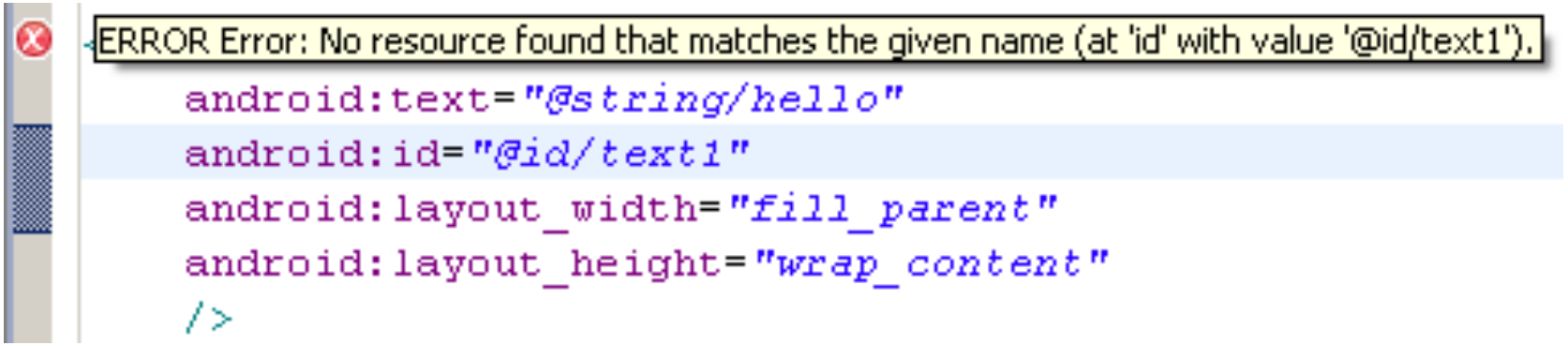
- Add a menu with four items to “Hello World”
- Menu items 1-3 changes text shown in the top of the display
  - setText(...)
- Menu item 1 → Probestudium
- Menu item 2 → LMU
- Menu item 3 → Android
- Menu item 4: Exit the application
  - finish()



Resources

Resource-Reference Syntax

- “+” Use id if it already exists, otherwise create new id
- @id/text1



- @+id/text1

```
<TextView
    android:text="@string/hello"
    android:id="@+id/text1"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    />
<Button
```

Image Resources

- Automatic id generation for images in /res/drawable
  - Example: /res/drawable/sample_image.jpg → R.drawable.sample_image
- Supported types: .gif, .jpg, .png

- Usage in Java

```
Button b = (Button)this.findViewById(R.id.Button01);
b.setBackgroundResource(R.drawable.sample_image);
```

- Usage in XML

```
<Button android:text="@string/Button01"
…
android:background="@drawable/sample_image" />
```

UI Components

- Common Controls
- Layout Managers
- Menus
- Dialogs

Core UI Component Classes

java.lang.Object
↑ android.view.View
↑ android.view.ViewGroup
↑ android.widget.LinearLayout

- android.view.View
  - Rectangular area on the screen
  - Responsible for drawing and event handling
  - Base class for widgets (buttons, text fields, etc.)
- android.view.ViewGroup
  - Is a view and contains other views (“container”)
  - Base class for layouts
- Layouts
  - Invisible containers that hold other Views
  - Define their layout properties (position, padding, size, etc.)
  - Example: LinearLayout (horizontal / vertical list of children)

Design UI in XML, Reference in Java

- Assign IDs in XML

```
<TextView android:id="@+id/nameValue" .../>
<TextView android:id="@+id/addrValue" ... />
```

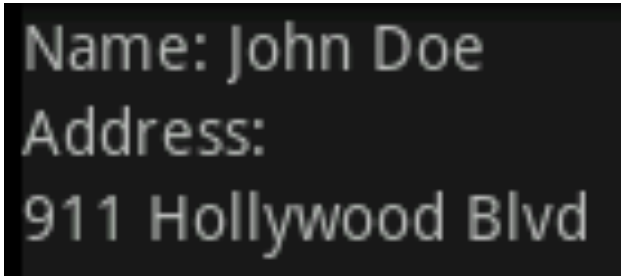
- Refer to controls using IDs

```
TextView nameValue = (TextView) findViewById(R.id.nameValue);
nameValue.setText("John Doe");
TextView addrValue = (TextView)findViewById(R.id.addrValue);
addrValue.setText("911 Hollywood Blvd.");
```

- View must have been loaded before referencing IDs

```
setContentView(R.layout.test);
```

Creating a UI in XML (/res/layout/test.xml)

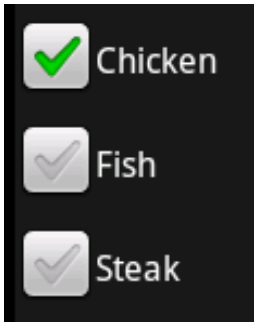


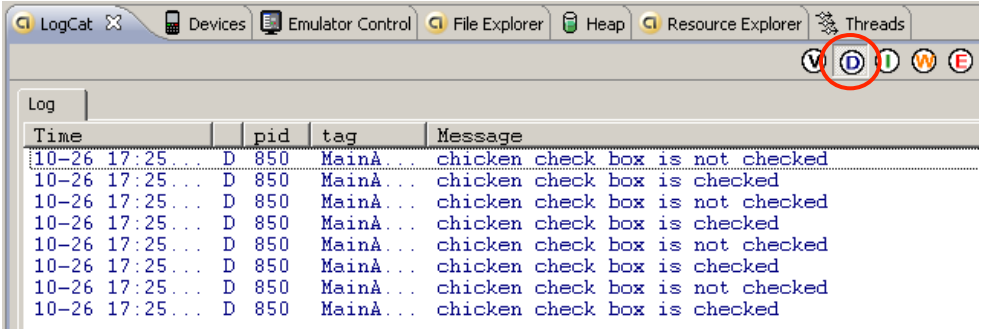

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="fill_parent"
    android:layout_height="fill_parent">
  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:orientation="horizontal" android:layout_width="fill_parent"
      android:layout_height="wrap_content">
    <TextView android:layout_width="wrap_content"
      android:layout_height="wrap_content" android:text="Name: " />
    <TextView android:layout_width="wrap_content"
      android:layout_height="wrap_content" android:text="John Doe" />
  </LinearLayout>
  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical" android:layout_width="fill_parent"
    android:layout_height="wrap_content">
  <TextView android:layout_width="fill_parent"
    android:layout_height="wrap_content" android:text="Address:" />
  <TextView android:layout_width="fill_parent"
    android:layout_height="wrap_content" android:text="911 Hollywood Blvd." />
  </LinearLayout>
</LinearLayout>
```

Setting the XML UI in Java

```
public class MainActivity extends Activity {
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.test);
    }
}
```

CheckBox





- XML

```
<LinearLayout android:orientation="vertical" ... >
    <CheckBox android:id="@+id/chicken" android:text="Chicken" ... />
    <CheckBox android:id="@+id/fish" android:text="Fish" ... />
    <CheckBox android:id="@+id/steak" android:text="Steak" ... />
</LinearLayout>
```

- Java

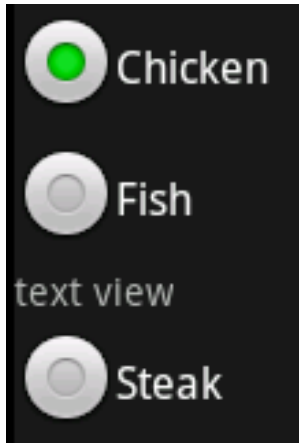
```
CheckBox cb = (CheckBox) findViewById(R.id.chicken);
cb.setChecked(true);
cb.setOnCheckedChangeListener(new OnCheckedChangeListener() {
  public void onCheckedChanged(CompoundButton b, boolean isChecked) {
     Log.d("MainActivity", "chicken check box is " +
        (isChecked ? "" : "not ") + "checked");
  }
});
```

Radio Button

- XML

```
<LinearLayout android:orientation="vertical"
        android:layout_width="wrap_content“
        android:layout_height="wrap_content">
    <RadioGroup android:layout_width="wrap_content“
        android:layout_height="wrap_content">
        <RadioButton android:text="Chicken"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />
        <RadioButton android:text="Fish"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" />
        ...
    </RadioGroup>
</LinearLayout>
```

- Radio groups can contain arbitrary views



Location-Based Services

Location-Based Services

- Location APIs: Access location data (GPS, WiFi, GSM)
  - android.location
  - LocationManager
  - Geocoder
- Mapping APIs: Display and navigate maps
  - com.google.android.maps
  - MapView
  - MapActivity

Permissions (in AndroidManifest.xml)

- Permissions for location-based services

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

- Child of element <application>
  - `<uses-library android:name="com.google.android.maps" />`

- Example
  - http://developer.android.com/intl/fr/guide/tutorials/views/hello-mapview.html

Location Manager Service

- Obtain device's geographical location
- Get notification upon entering a specified location

Example: Last Location

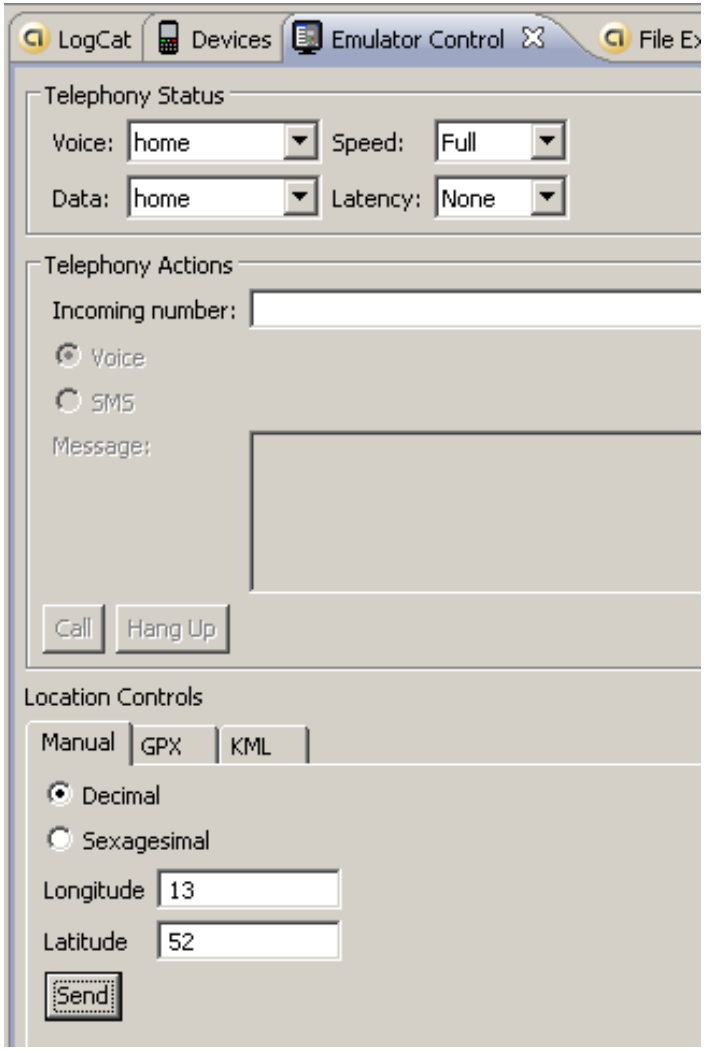
```
public class LocationManagerDemoActivity extends Activity {
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        LocationManager locMgr = (LocationManager)
                getSystemService(Context.LOCATION_SERVICE);
        Location loc = locMgr
                .getLastKnownLocation(LocationManager.GPS_PROVIDER);
        Toast.makeText(this, loc.toString(), 10000).show();
        Log.d("last location", loc.toString());
        List<String> providerList = locMgr.getAllProviders();
        Iterator<String> iter = providerList.iterator();
        while (iter.hasNext()) {
            Log.d("provider", iter.next().toString());
} } }
```

Example: Location Updates

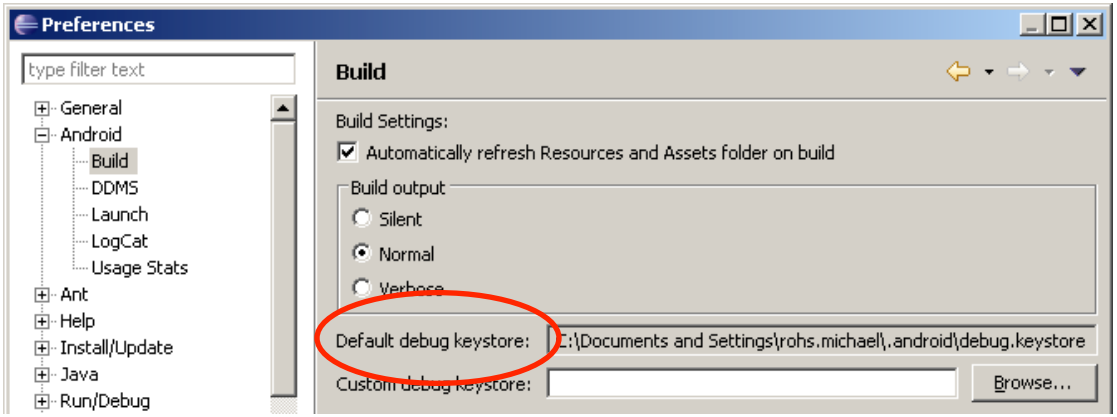
```
public class LocationUpdateDemoActivity extends Activity {
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      LocationManager locMgr = (LocationManager)
            getSystemService(Context.LOCATION_SERVICE);
      LocationListener locListener = new LocationListener() {
         public void onLocationChanged(Location location) {
            if (location != null) {
               Toast.makeText(getBaseContext(),
                  "New location (" + location.getLatitude() + ", " +
                  location.getLongitude() + ")", Toast.LENGTH_LONG).show();
               }
            }
            public void onProviderDisabled(String provider) {}
            public void onProviderEnabled(String provider) {}
            public void onStatusChanged(String provider, int status, Bundle extras) {}
         };
      locMgr.requestLocationUpdates(LocationManager.GPS_PROVIDER,
            0, 0, locListener);
}}
```

Simulated Location for the Emulator

- Dalvik Debug Monitor Service
- Play back GPS traces
  - GPX: GPS Exchange Format
  - KML: Keyhole Markup Language
- Telnet to a running emulator
  - telnet localhost <emulator port>
  - geo fix <lon> <lat>
  - geo nmea <nmea sentence>
- Example
  - telnet localhost 5554
  - geo fix 13 52
  - http://developer.android.com/intl/fr/guide/developing/tools/emulator.html



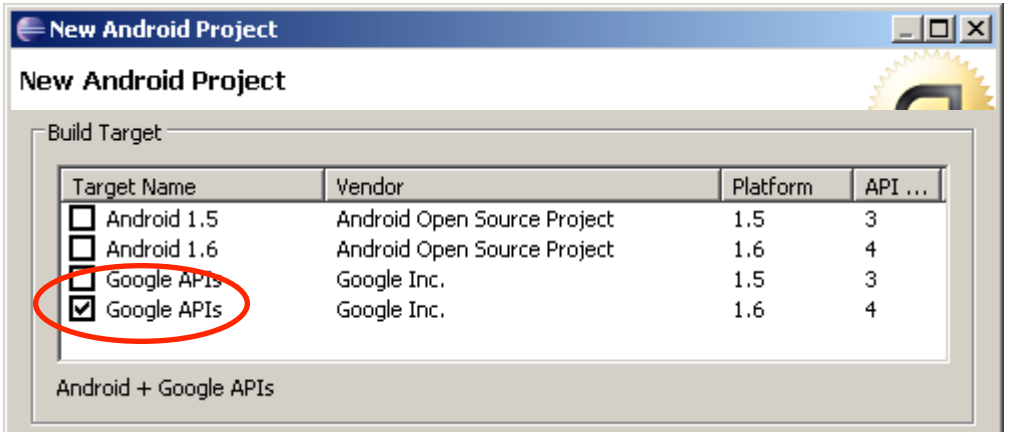
Map API Key



- Locate keystore
- Open command line
- Get MD5 hash of debug certificate

  keytool -list -alias androiddebugkey
  -keystore "C:\Documents and Settings\<user>\.android\debug.keystore"
  -storepass android -keypass android

- Get the key from Google
  - http://code.google.com/android/maps-api-signup.html
- Projects using maps need build target "GoogleAPIs"
  - Potentially needs a new AVD



Example Map View

- XML

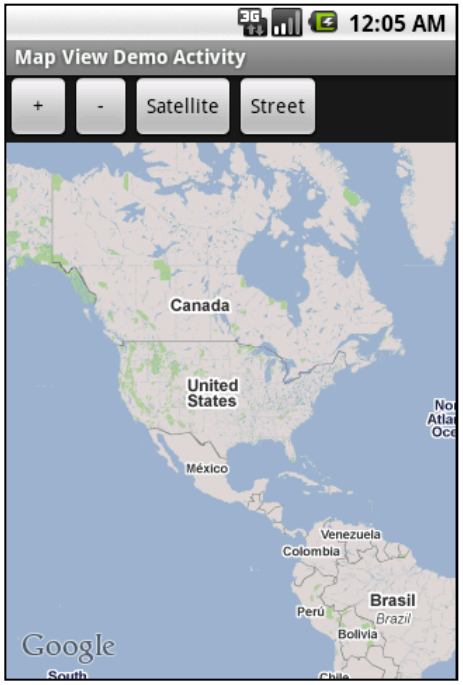
```
<LinearLayout xmlns:android="http://schemas..."
    android:orientation="vertical" android:layout_... >
    <com.google.android.maps.MapView android:layout_...
        android:apiKey="02LvHoUW1Z_HVYZWU..." />
</LinearLayout>
```

- Java

```
public class MapViewDemoActivity extends MapActivity {
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.mapview);
    }
    protected boolean isRouteDisplayed() { return false; }
}
```

Example Map View with Controls

```
<LinearLayout xmlns:android="http://schemas..."
                android:orientation="vertical" ...>
  <LinearLayout android:orientation="horizontal" android:layout_...>
    <Button android:id="@+id/zoomin" android:text="  +  " ... />
    <Button android:id="@+id/zoomout" android:text="  -  " ... />
    ...
  </LinearLayout>
  <com.google.android.maps.MapView
    android:id="@+id/mapview“
    android:apiKey="02Lv..." ... />
</LinearLayout>
```



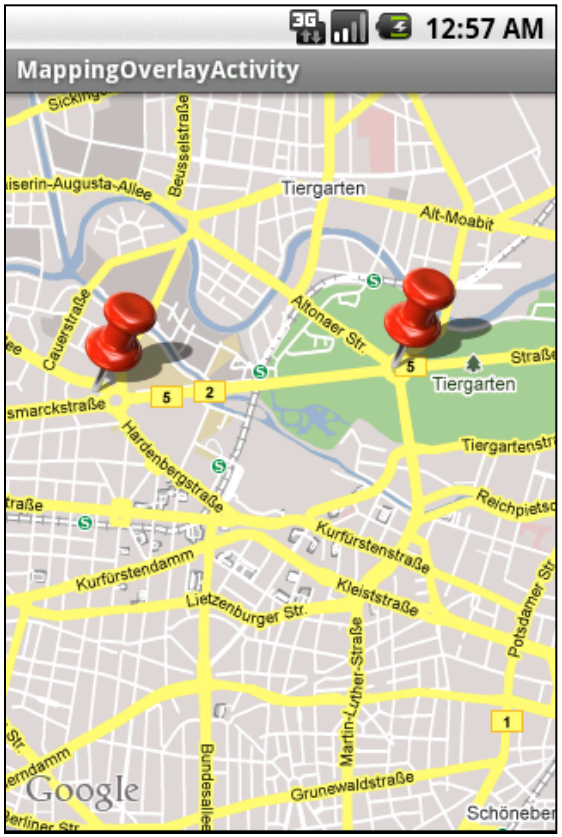
Example Map View with Controls

```
public class MapViewDemoActivity extends MapActivity {
    private MapView mapView;
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.mapview);
        mapView = (MapView) findViewById(R.id.mapview);
        Button zoominBtn = (Button) findViewById(R.id.zoomin);
        zoominBtn.setOnClickListener(new OnClickListener() {
            public void onClick(View view) {
                mapView.getController().zoomIn();
            }});
        ...
    }
    protected boolean isRouteDisplayed() { return false; }
}
```

Using Overlays

- /res/layout/mapviewoverlay.xml

```
<LinearLayout xmlns:android="http://schemas..."
                    android:orientation="vertical" ...>
   <com.google.android.maps.MapView
      android:id="@+id/mapviewoverlay"
      android:apiKey="02Lv..." ... />
</LinearLayout>
```



Using Overlays

```
public class MappingOverlayActivity extends MapActivity {
    private MapView mapView;
    private GeoPoint tlabs = new GeoPoint((int)(
        52.513036 * 1000000), (int)(13.320281 * 1000000));
    private GeoPoint saeule = new GeoPoint((int)(
        52.514495 * 1000000), (int)(13.350130 * 1000000));

    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.mapviewoverlay);
        mapView = (MapView) findViewById(R.id.mapviewoverlay);
        mapView.setBuiltInZoomControls(true);
        mapView.setClickable(true);
        mapView.getController().setCenter(tlabs);
        mapView.getController().setZoom(14);
        Drawable marker = getResources().getDrawable(R.drawable.pushpin);
        mapView.getOverlays().add(new InterestingLocations(marker));
    }
...}
```

Using Overlays

```
class InterestingLocations extends ItemizedOverlay<OverlayItem> {
    private List<OverlayItem> locations = new ArrayList<OverlayItem>();
    private Drawable marker;
    public InterestingLocations(Drawable marker) {
        super(marker);
        this.marker = marker;
        locations.add(new OverlayItem(tlabs, "T-Labs", "T-Labs"));
        locations.add(new OverlayItem(saeule, "Siegessäule", "Siegessäule"));
        populate();
    }
    public void draw(Canvas canvas, MapView mapView, boolean shadow) {
        super.draw(canvas, mapView, shadow);
        boundCenterBottom(marker);
    }
    protected OverlayItem createItem(int i) {
        return locations.get(i);
    }
    public int size() {
        return locations.size();
    }
}
```

Marker hotspot: bottom center

Prof. Dr. Michael Rohs
michael.rohs@ifi.lmu.de
Dipl.-Inform. Sven Kratz
sven.kratz@ifi.lmu.de
Mobile Interaction Lab, LMU München