



Seminararbeit

Business Information Systems

4219 Seminar aus BIS

im SS 2018

Developing for Android

Sebastian Höfinger (h1451987)

LV-Leitung: ao.Univ.Prof. Dr. Rony G. Flatscher

Wien , 26.03.2018

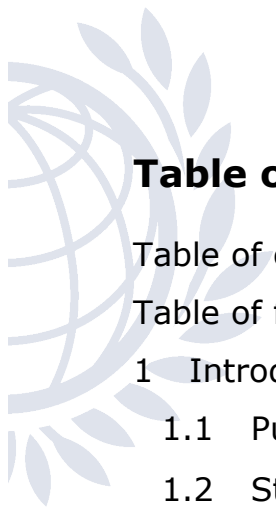


Table of content

Table of content	I
Table of figures	II
1 Introduction	1
1.1 Purpose.....	1
1.2 Structure of this project paper	2
2 Android and Java	2
2.1 Learn Java	2
2.2 History of Android and Tizen	3
2.3 Tools.....	4
2.3.1 Android Studio.....	4
3 Applications Set-up	5
3.1 Creating a new Project in Android Studio	6
3.2 Emulate a device in Android Studio	12
3.3 Connect a smartphone with an emulated Gear	15
4 Applications	23
4.1 Sound application	23
4.2 E-Mail shortcut application	25
4.3 Google-search application	28
5 Conclusion	31
Anhang.....	A



Table of figures

All screenshots from „Android Studio“ are self-made, but all copyright goes to
<https://developer.android.com/studio/>.

1 Introduction

The reason why I chose the topic "Developing for Android" is because I am using an android smartphone (Samsung Galaxy S7 edge) for nearly two years now and I am really satisfied with it. I also have a good comparison to the big competitor Apple because before my Samsung smartphone I was using an iPhone 6 for over two years. For a few months now, I also own a new smartwatch, the Samsung Gear S3 frontier. I never was really convinced by smartwatches and their benefit in our everyday life, but after using and testing it for some time now I have to say that I am really impressed. It helps with so many short tasks, for example jumping to the next music title when you are in a crowded subway or checking your calendar. For that reasons I choose the topic mentioned above, because I really like the android system on smartphones, and I think "wearables" like the Samsung Gear S3 have a lot of potential in the future.

1.1 Purpose

The purpose of this project paper is to learn and understand the basics of programming for android wearable devices without any previous knowledge. After reading this paper and doing some self-learning in the internet (I will provide links and sources where you can do this) you should be able to write some basic android applications-. Also, you will have a much better understanding of how applications on your smartphone work and how complex some small applications can be.

1.2 Structure of this project paper

First, this project paper will provide some basic knowledge about the android operating system. Then it will continue with some basic knowledge about the devices and software used in this project paper.

I will also provide sources where I learned the basics of Java.

Then it will show 3 small applications, how they work and what their purpose is.

This paper will also provide a step-by-step explanation how to set up the different tools.

2 Android and Java

2.1 Learn Java

To write your own applications, you must learn Java. This sounds simple, but Java can be very complex and vast. If you have never programmed before (like me) I would suggest you to do some basic online Java courses. Most of them are free and give a very good basic knowledge about Java. For example this free course: <https://www.sololearn.com/Play/Java> .

Also, there is a very good online course from "codecademy" which is even better than the course from "sololearn" above, but for the full course you must pay.

There you learn the basic concepts, conditionals, loops, arrays, classes and objects and some exceptions.

2.2 History of Android and Tizen

Android

Android is based on a modified version of Linux and other open source software. It is developed by Google and is primarily used for smartphones and tablets. In general, it is used for touchscreen devices, so it is also used for the Samsung Gear Series.

Android Inc. was founded in October 2003 by Andy Rubin, Rich Miner, Nick Sears and Chris White in California. The first Idea was that Android should work as the operating system in digital cameras, but they soon mentioned that the market for digital cameras is too small and so they switch to using Android as an operating system for handset devices. Android developed so good that in July 2005, Android Inc. was acquired by Google for about \$50 Million.¹

Tizen

Tizen is an open source mobile operating system developed by Samsung and runs on a lot of Samsung mobile devices for example smartwatches, smartphones, tablets, smart televisions, smart cameras, smart home

¹

[https://en.wikipedia.org/w/index.php?title=Android_\(operating_system\)&oldid=843029452](https://en.wikipedia.org/w/index.php?title=Android_(operating_system)&oldid=843029452)

appliances and more. It is based on Linux and written in HTML5, C and C++.²

It was a business strategy from Samsung to try to launch their own operating system because they did not want to depend on Google. So, they developed Tizen and wanted to use it for their smartphones and smartwatches. Because there were a lot of android apps and android application developers and Tizen was so new there were almost non-native Tizen applications or Tizen application developers. So, they tried to make an intelligent move and add compatibility for android applications. They made a layer in the operating system, that supports the android applications. But since the android applications run that efficiently on Tizen as they used to on android devices most of the applications, and new applications, stayed on android.

2.3 Tools

To develop an application for android, or in this case also for a wearable android device, you need a lot of tools and programs. In the following points I will briefly explain the function of this tools and how you get them to run on your computer.

2.3.1 Android Studio

The most important tool is Android Studio. Android Studio is the official Integrated Development Environment (IDE) for android application development. It is based on IntelliJ IDEA which is an IDE developed by JetBrains. With the powerful code editor from IntelliJ and it's developer

² <https://en.wikipedia.org/w/index.php?title=Tizen&oldid=845594071>

tools, Android Studio offers a lot of features to build very complex and great android apps.³

To download Android Studio go to <https://developer.android.com/studio/> and press on „Download Android Studio“. You always get the newest version there. In my presentation and following screenshots, I have used version 3.1.2 so if you have a newer version it may look a bit different.

3 Applications Set-up

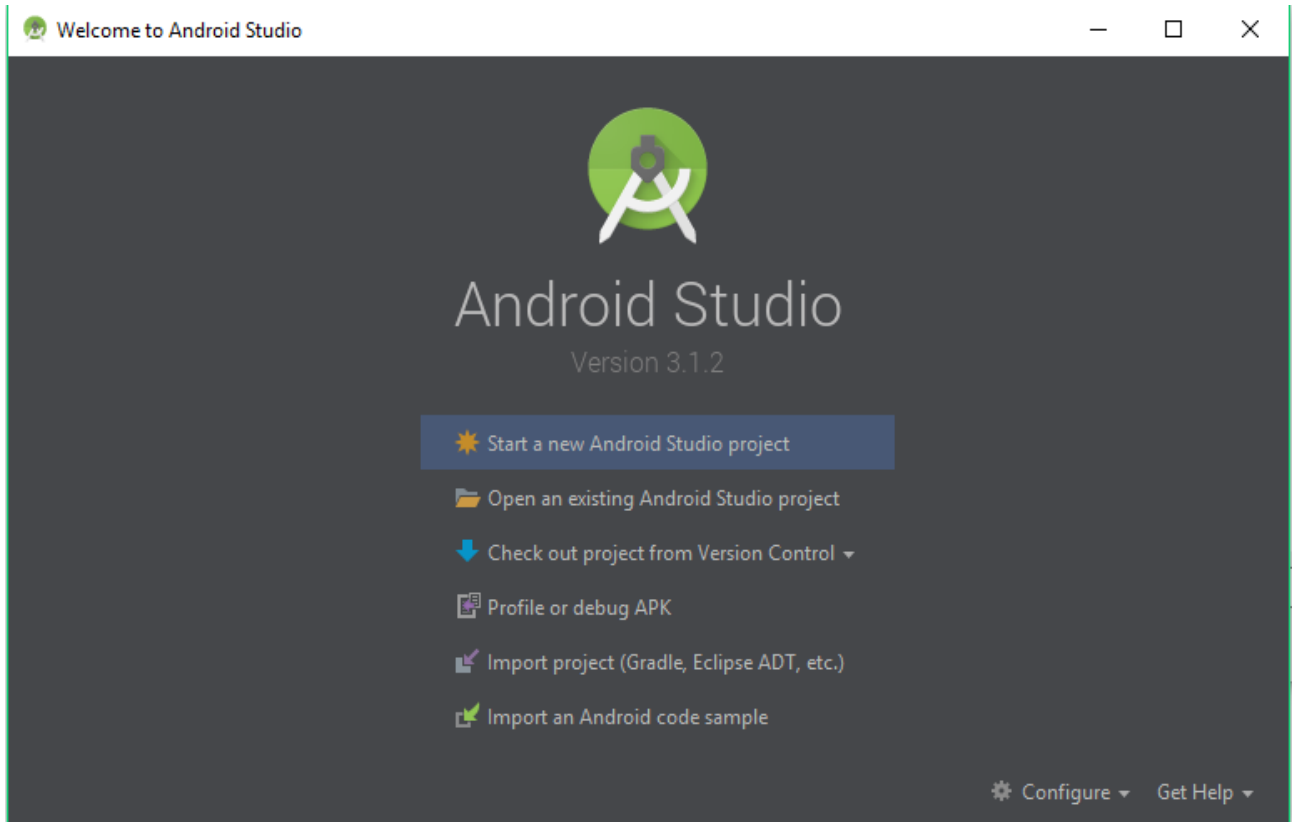
The following section will contain 3 Gear android applications, explain the purpose of them and how to set them up. There will also be a general “How-to” to create a new project in Android Studio.

³ https://en.wikipedia.org/w/index.php?title=Android_Studio&oldid=845831535

3.1 Creating a new Project in Android Studio

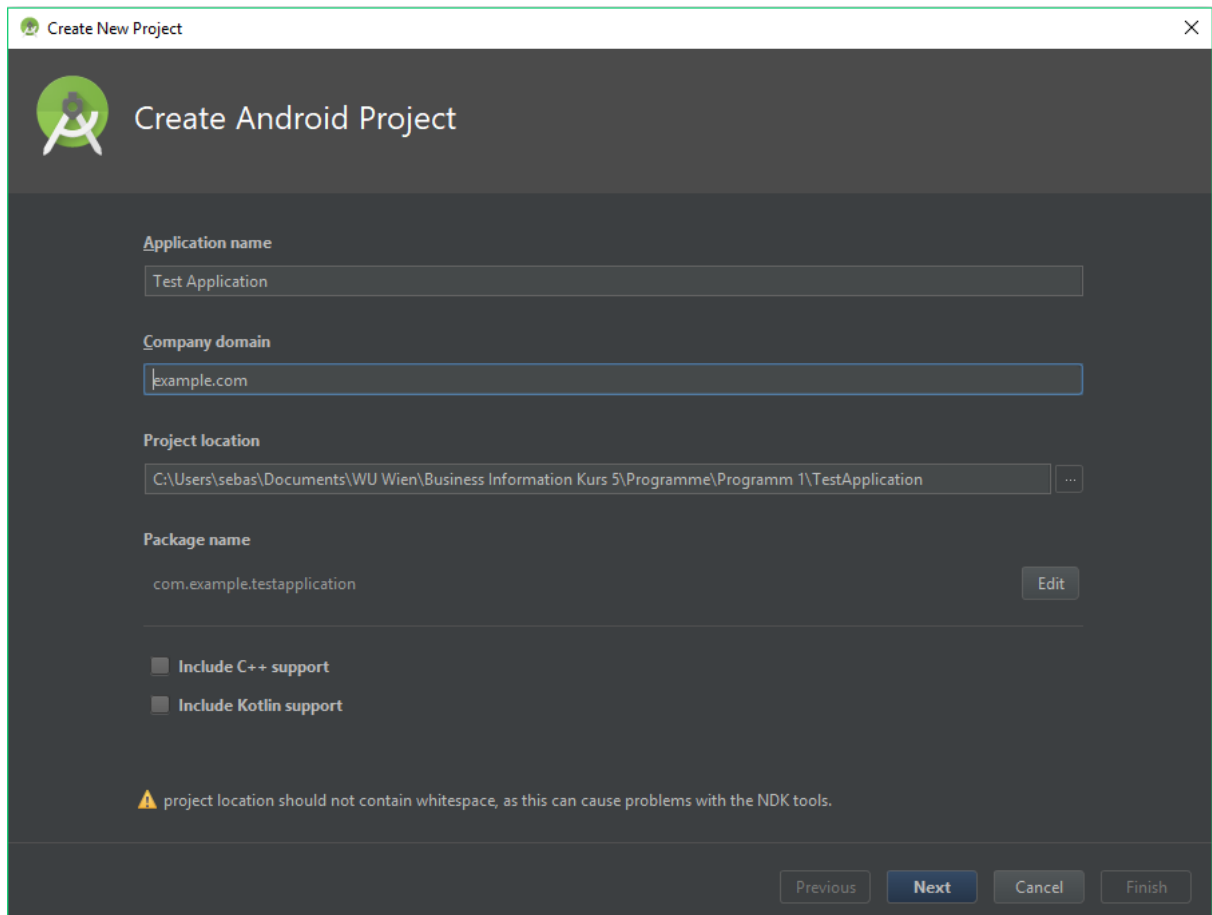
1. Step:

When you have set up Android Studio properly and open it, there is a button with "Start a new Android Studio project". It looks like this:



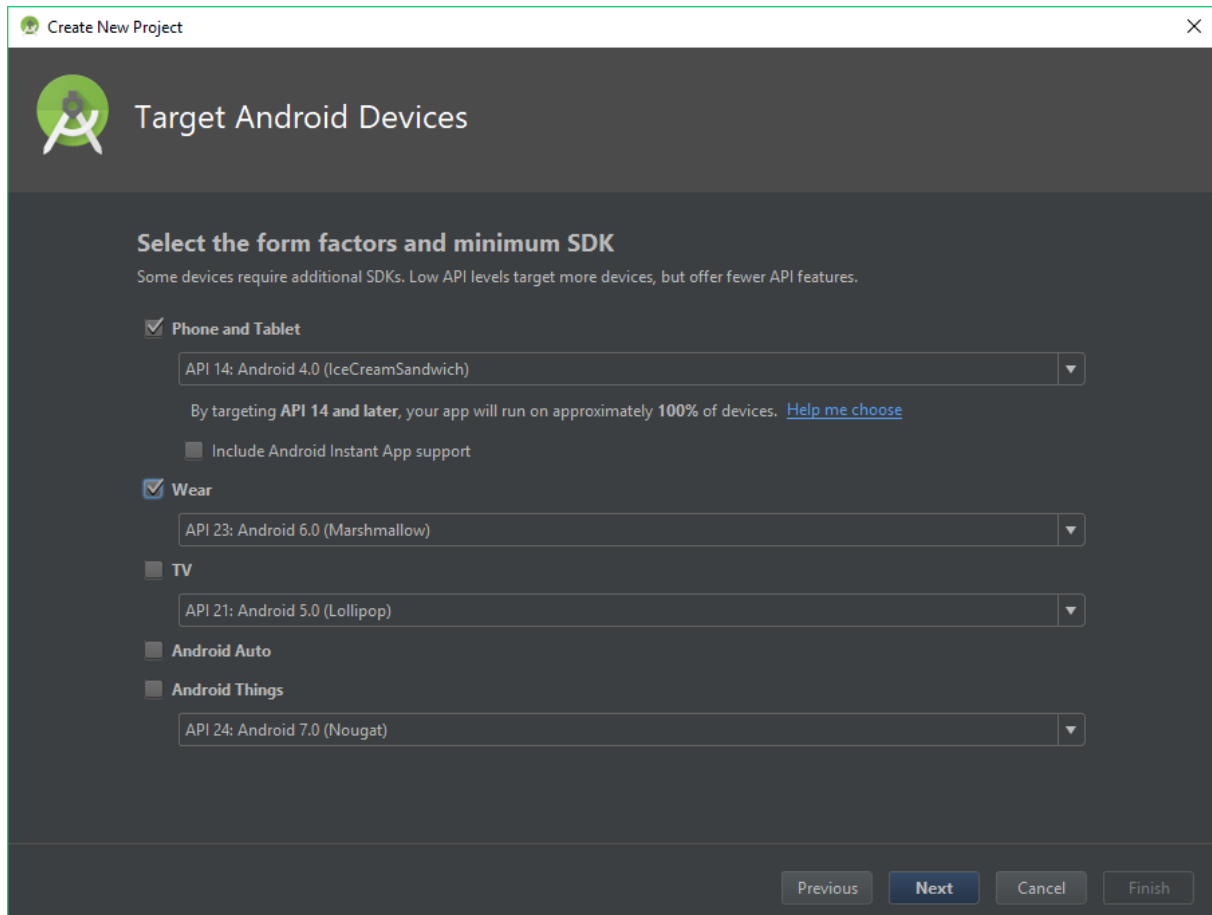
2. Step:

In the next window, you must choose a name for the new application, this can be anything.



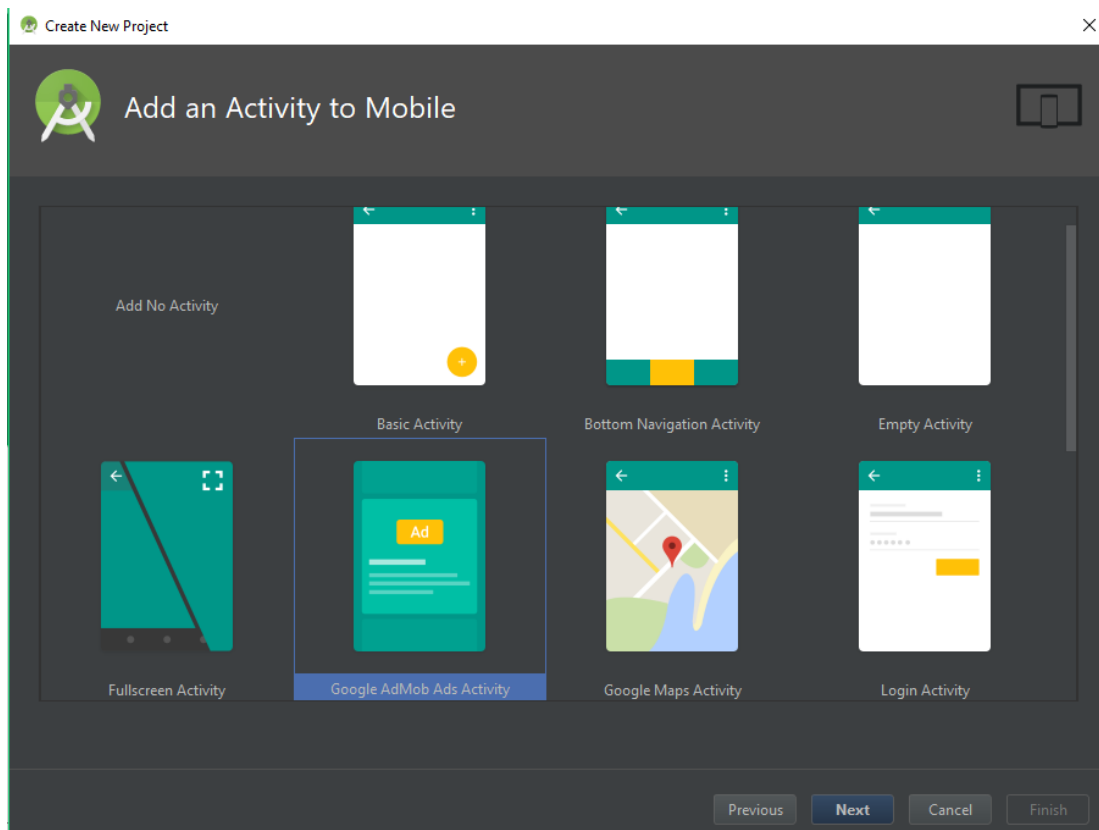
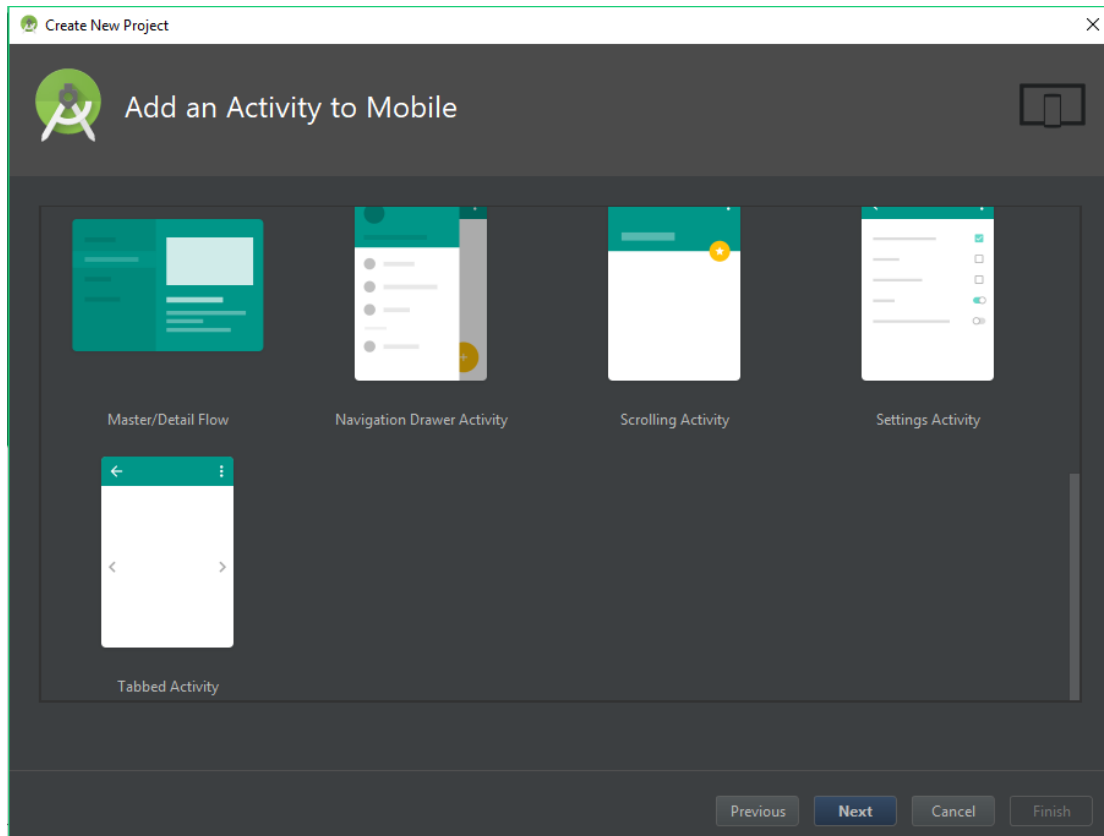
3. Step:

The following window you must choose the API (Application Programming Interface) for the new Application. That means you must choose the lowest Android Version on that the new Application is running. The lower API you choose the higher the number of compatible devices. But with every lower API you cannot use the features of the newer versions. This applies to both, the Phone and the Gear device. So, it is good to know which features you want to use and how many devices you want to support with the app.



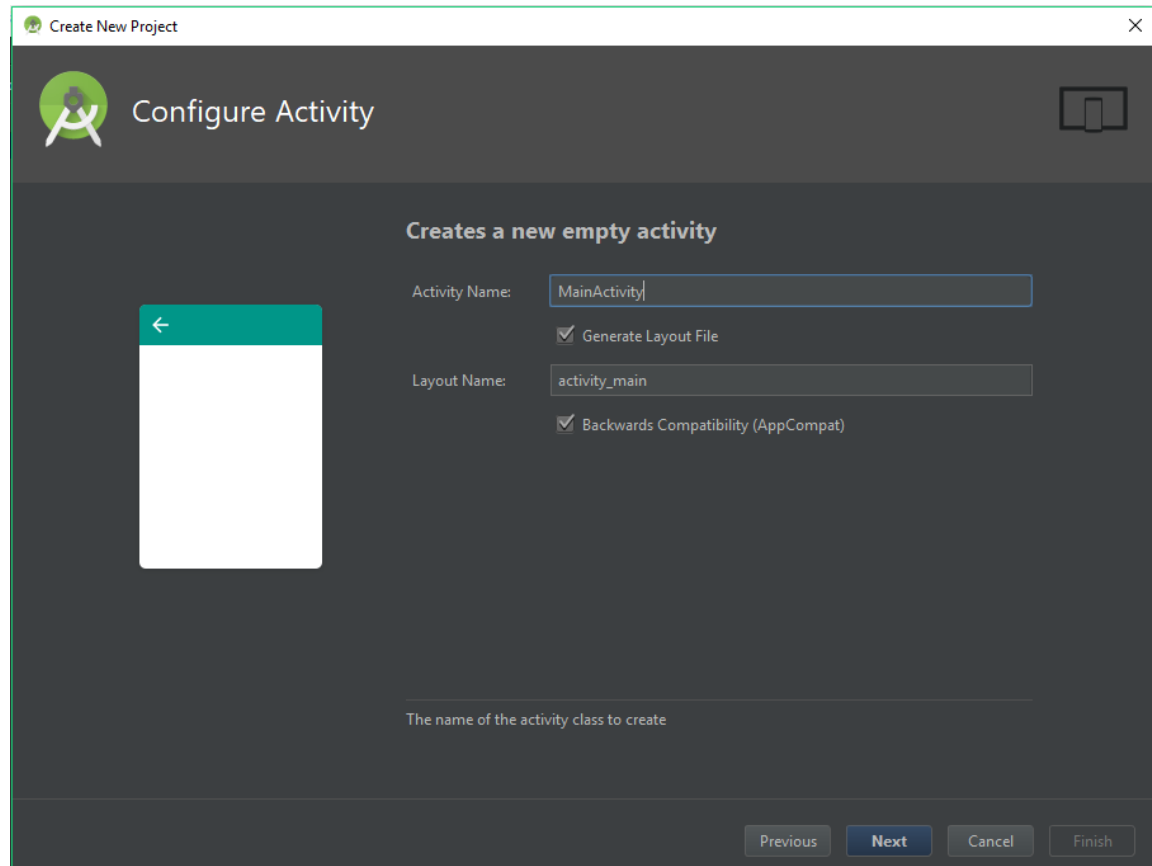
4. Step:

In the next window you add an activity to your application. You can choose between "Basic Activity", "Bottom Navigation Activity", "Empty Activity", "Full screen Activity", "Google Ad Mob Ads Activity", "Google Maps Activity", "Login Activity", "Master/Detail Flow", "Navigation Drawer Activity", "Scrolling Activity", "Tabbed Activity". So, you have a lot of choice here, and it really depends what you are planning for the main activity of your program. For my following programs the main activity on the Smartphone device always is "Empty Activity", because the applications on the phone are only function as receiver. (A more detailed information what that means follows above in section 4.)



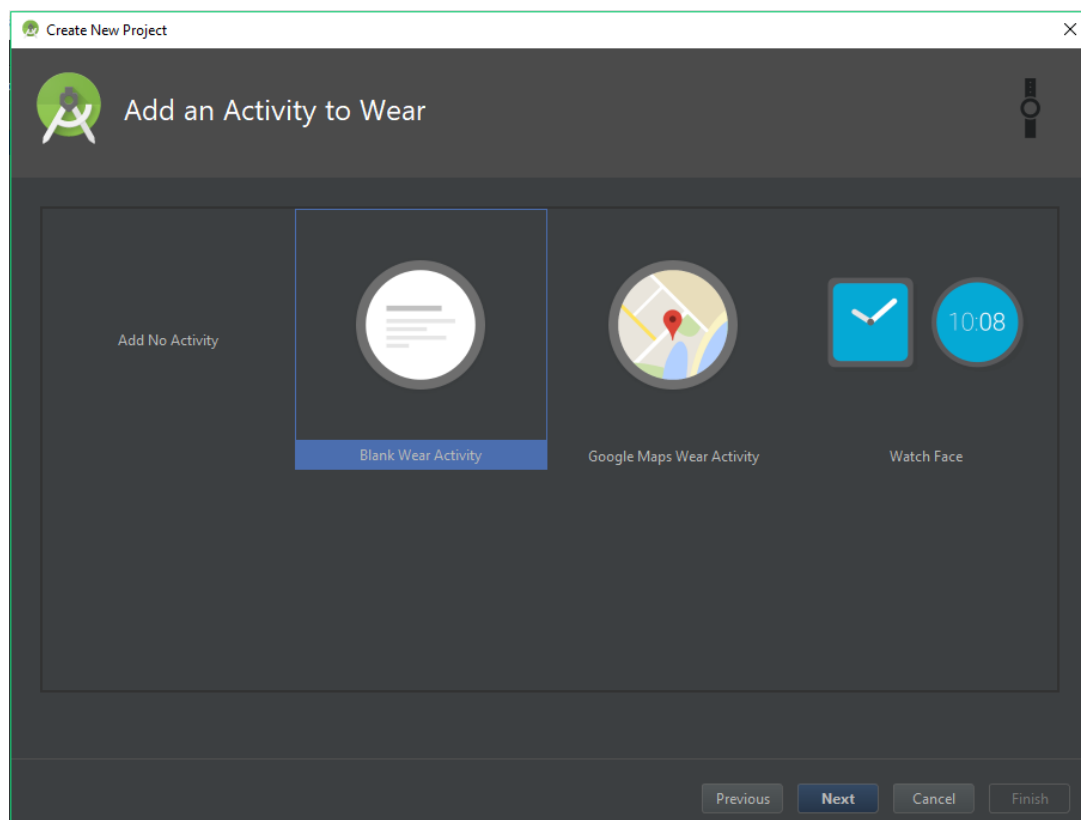
5. Step:

The last step for the application that is based on the phone, is to name the activity.



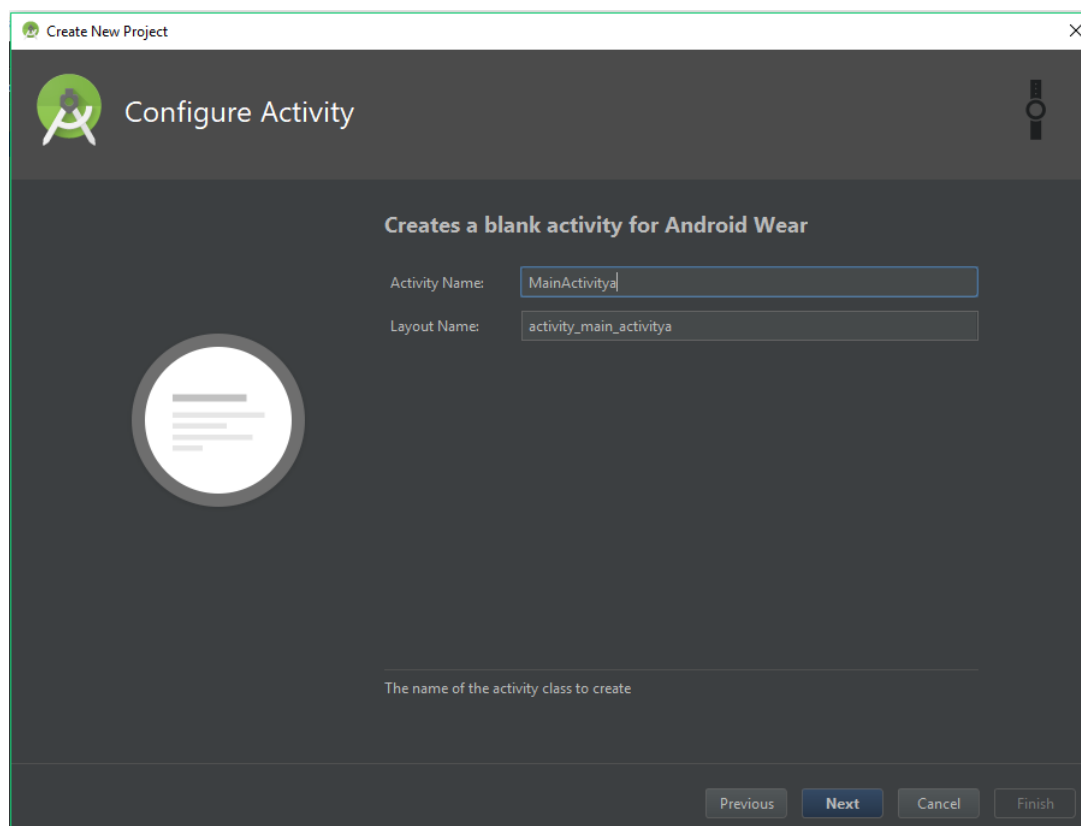
6. Step:

Now you must do very similar steps for the Gear device. Here you can choose between "Blank Wear Activity", "Google Maps Wear Activity", "Watch Face". Again: Your choice depends on your planned use for the application.)



7. Step:

In the very last step here you have again to name your activity.



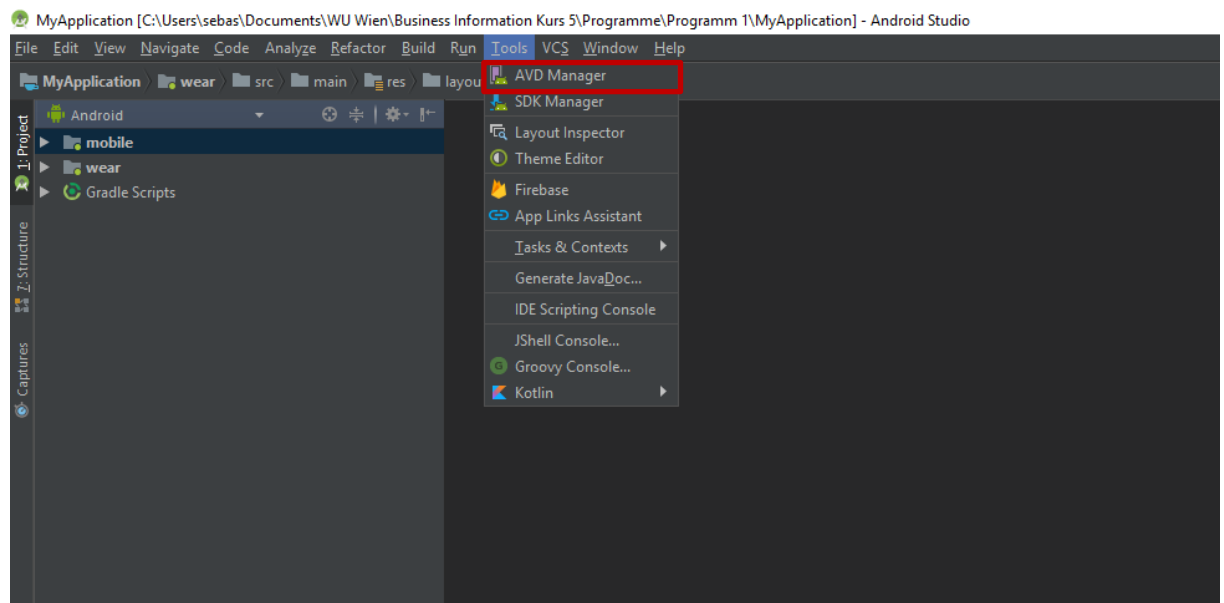
3.2 Emulate a device in Android Studio

To recreate my applications, or to create new applications you normally need a smartphone that runs android as well as a Samsung Gear device that runs Tizen. If you do not have these devices to test and run your applications, you can also emulate them with Android Studio.

An emulator tries to recreate a specific environment (e.g. a specific version of Android or Windows). This makes it possible to use the structure of the emulated operating system, to test programs or apps without even owning the device that normally runs that system.⁴

1. Step:

Create a new project (as explained above) or open an existing project to get into your programming environment. There you click on tools in the top navigation bar and select AVD manager.



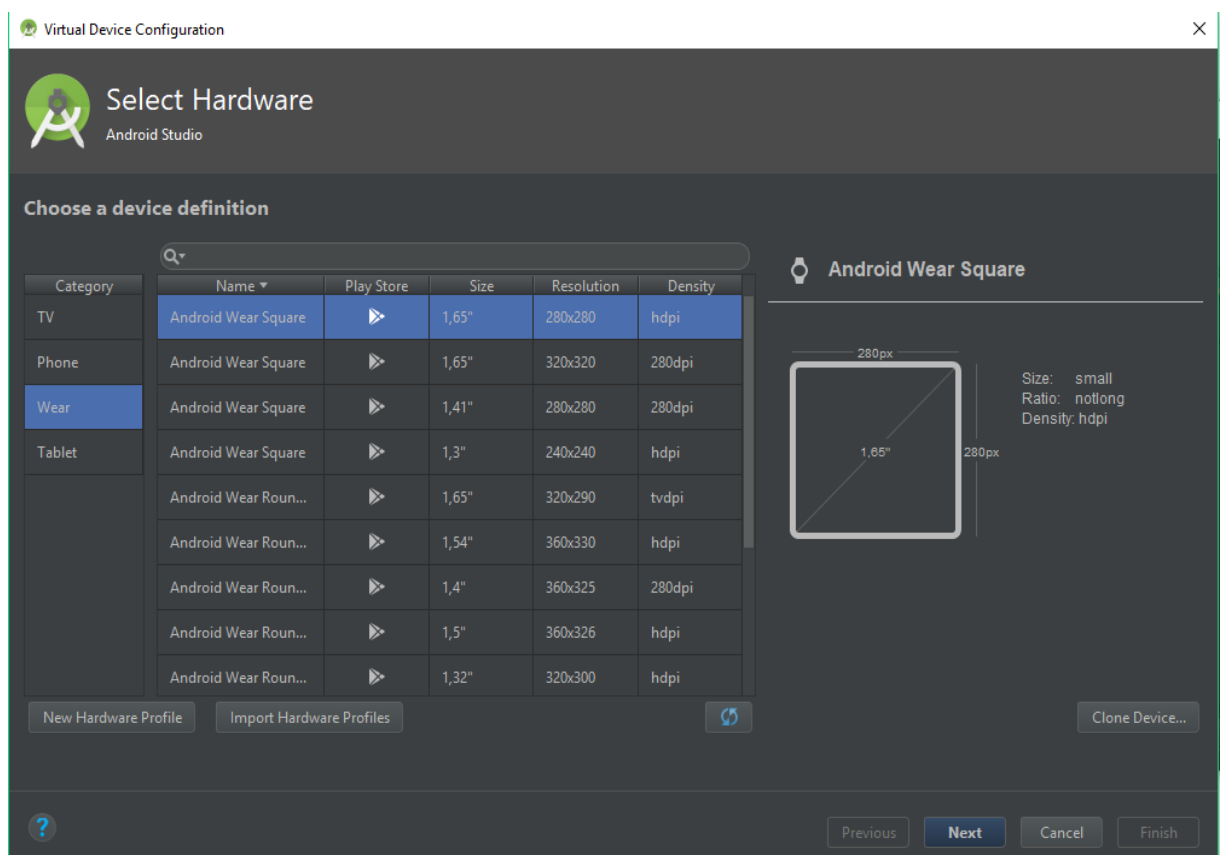
⁴ <https://de.wikipedia.org/w/index.php?title=Emulator&oldid=172913064>

2. Step:

When you have clicked on "AVD Manager" a new pop-up window opens. There you just click on "Create Virtual Device".

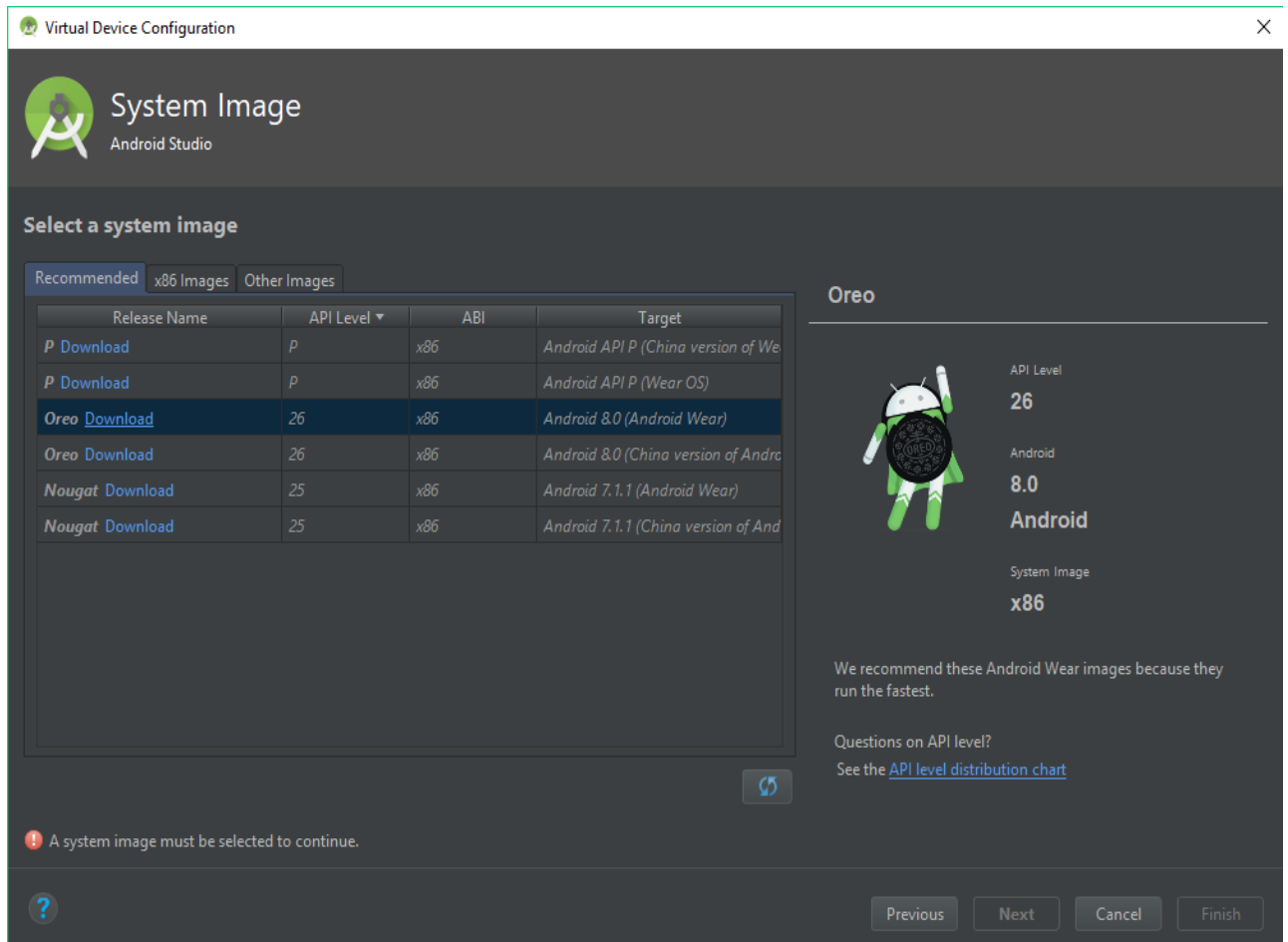
3. Step:

In the next window you have a lot of options. You can choose with device you want to emulate and what size the display should have. Just choose the device you want to emulate and experiment with and click on next.



4. Step:

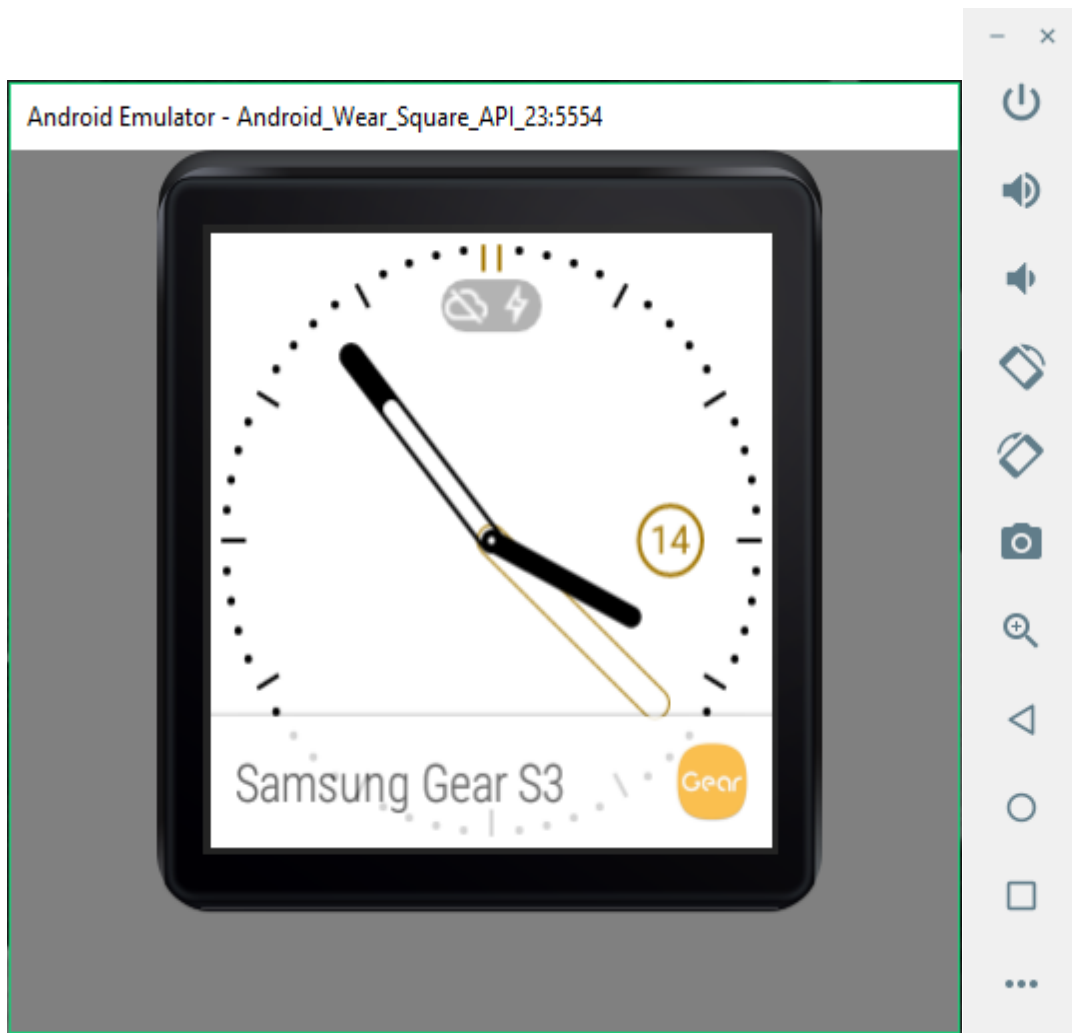
In the next step you must choose on which android Version your emulator should run. Your choice depends on the features you want to use in your application and with how many devices your application should be compatible (the higher the version the less are the compatible devices).



5. Step:

In the last step you can name the emulator and have an overview of all the options you have selected in the last steps. When you click on "Finish" your emulator is ready to run. Just navigate to the "AVD Manager" again (like in Step 1). There you now have a list with an entry of the new created emulator. Click on the green arrow under the "Actions" settings and you start your emulator.

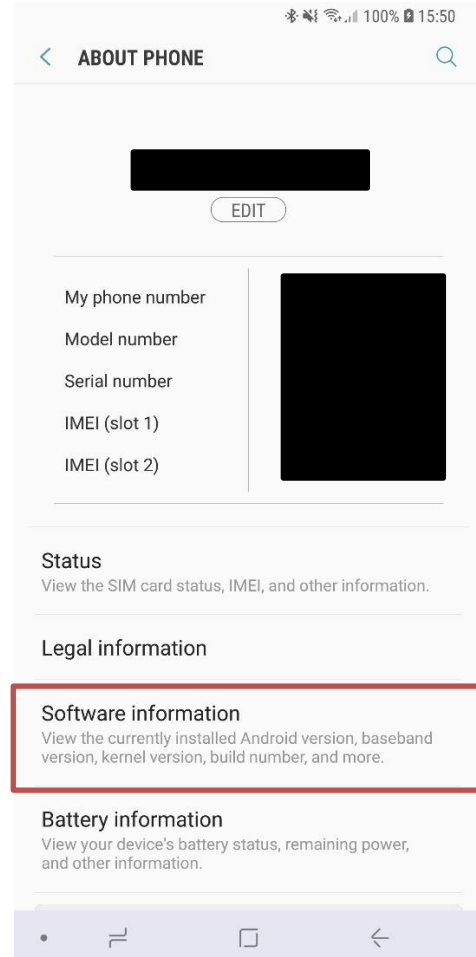
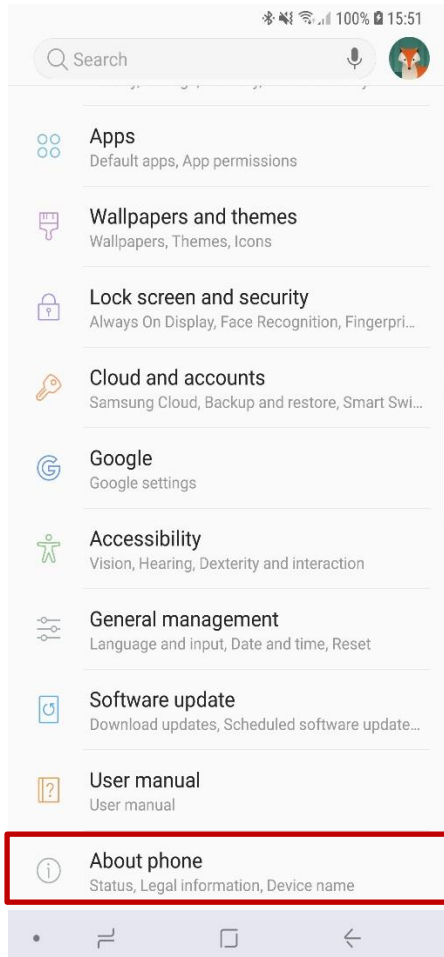
If you have done everything properly your emulator should look like this:



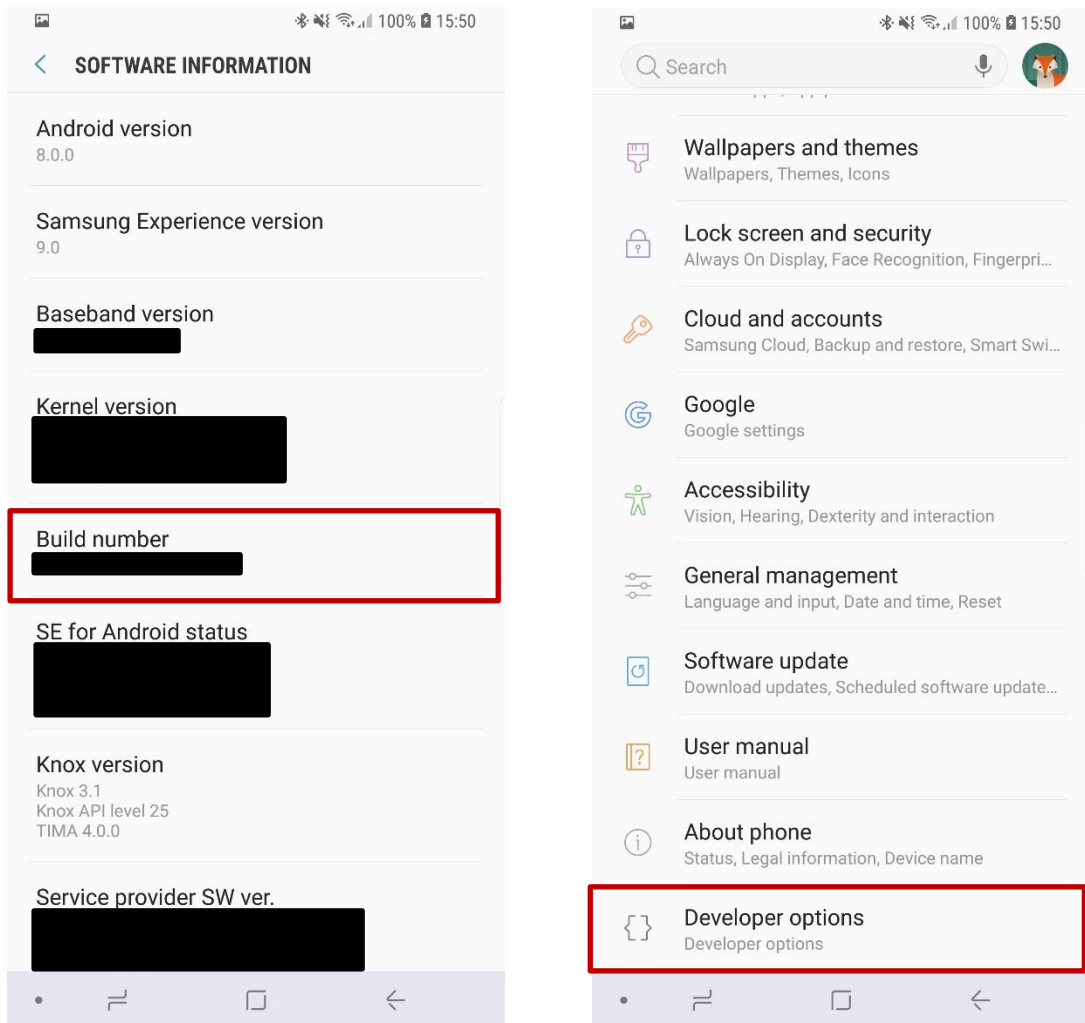
3.3 Connect a smartphone with an emulated Gear

When the emulator is set up properly and is running, you can connect it to a "physical" smartphone device. (Attention: this explanation is when you chose to emulate a Gear – a smartwatch device). To be able to install applications with android studio on your Samsung smartphone device you must enable the "USB - debugging" mode.

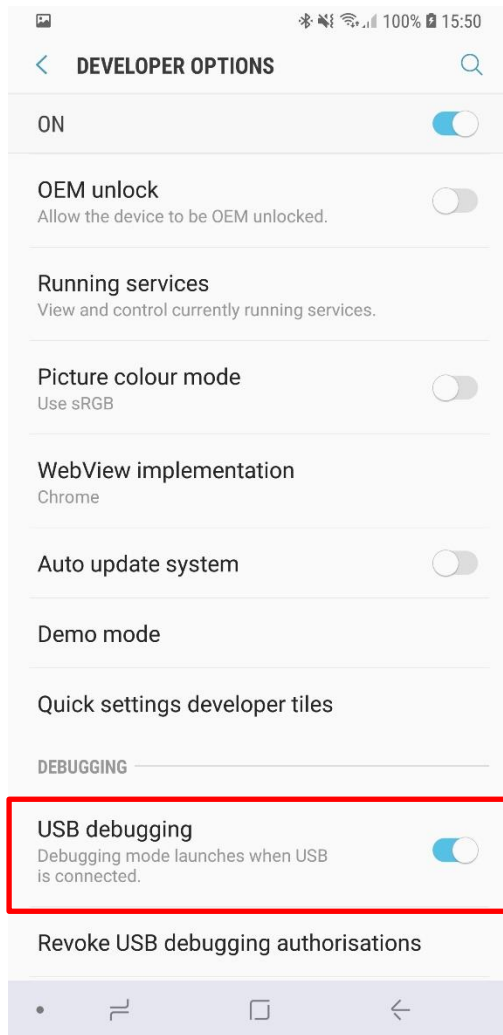
1. Step: Go to the settings on your smartphone and search for "About device".
2. Step: In the next window select the point "Software information".



3. Step: In the next window, search for the point "Build number" and tap it 7 times.
4. Step: Go back to the settings and scroll all the way down to the bottom. There is now a new point that is called "Developer options".

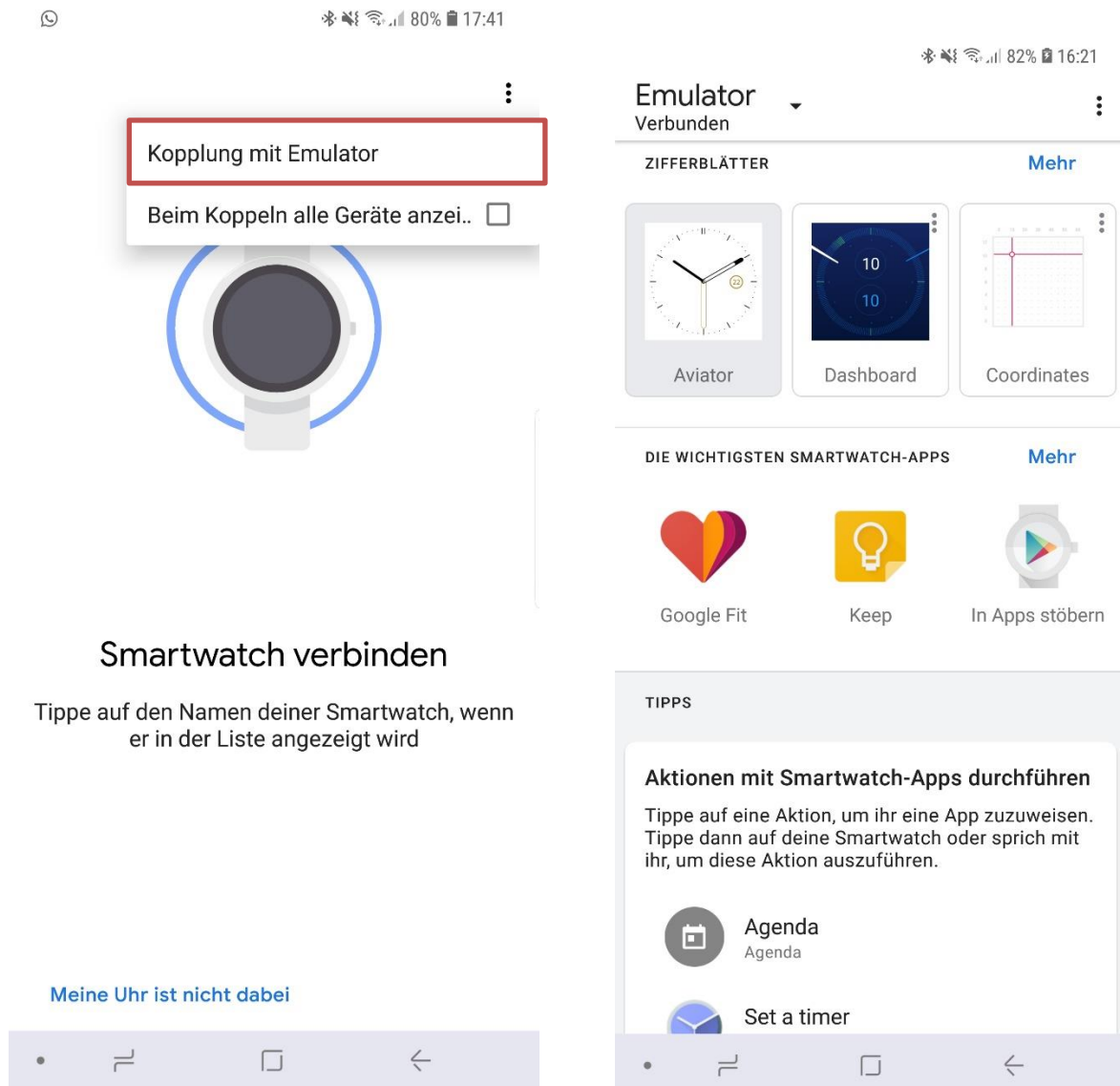


5. Step: In the "Developer options" scroll down until you reach the point "USB -debugging" and enable it.



Now you must download the Wear OS application from the google play store (<https://play.google.com/store/apps/details?id=com.google.android.wearable.app>) to be able to connect your smartphone with your emulated gear.

Make sure you have your Smartphone connected via USB-cable to your computer and that your emulator Gear is opened and running. Then open the Wear OS application on your smartphone and tap on the button on the top right corner and select "Kopplung mit Emulator". Your smartphone now should automatically connect with the emulator.



Smartwatch verbinden

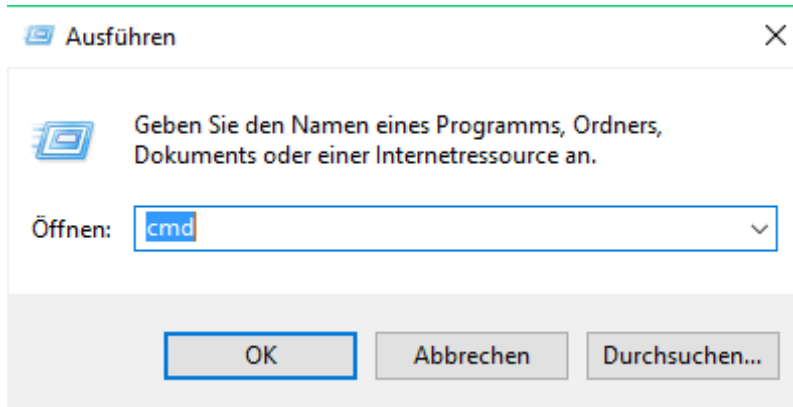
Tippe auf den Namen deiner Smartwatch, wenn er in der Liste angezeigt wird

Possible Problem:

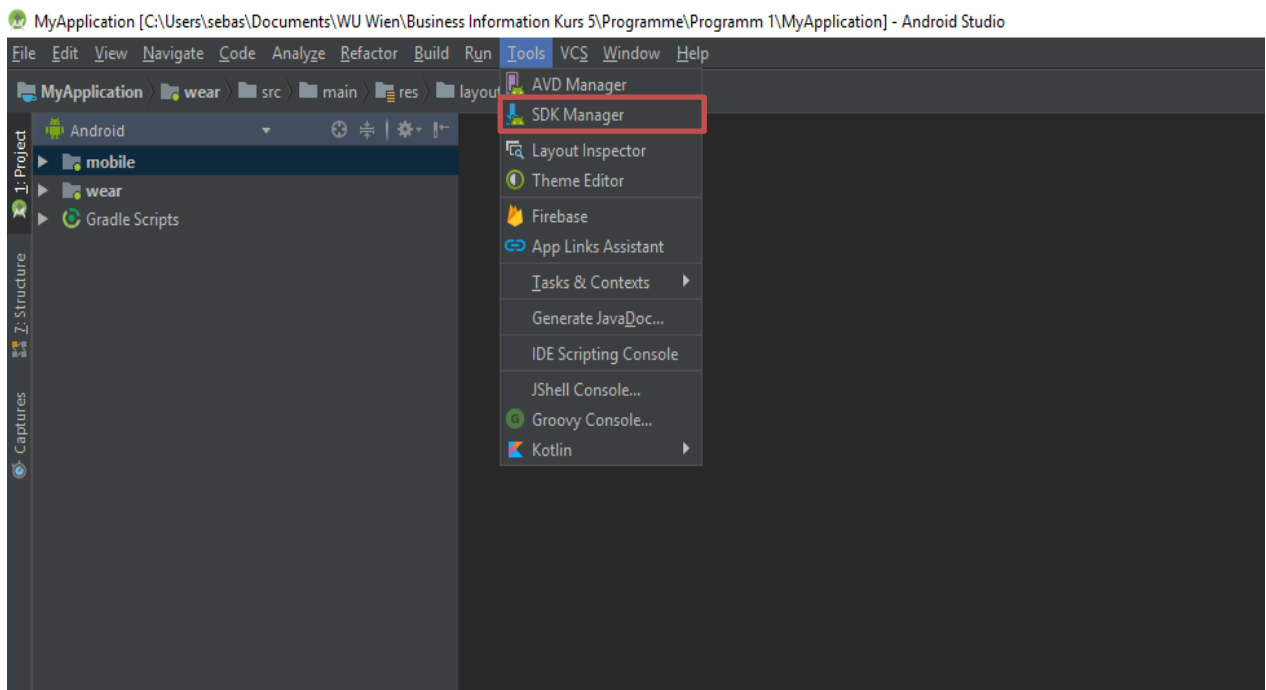
If the emulator does not connect automatically you can connect it directly over the windows command prompt. Make sure you have your smartphone connected and the emulator running.

1. Step:

Open "Run" ("Ausführen" if you run your windows in german) and type in "cmd" and press "OK".

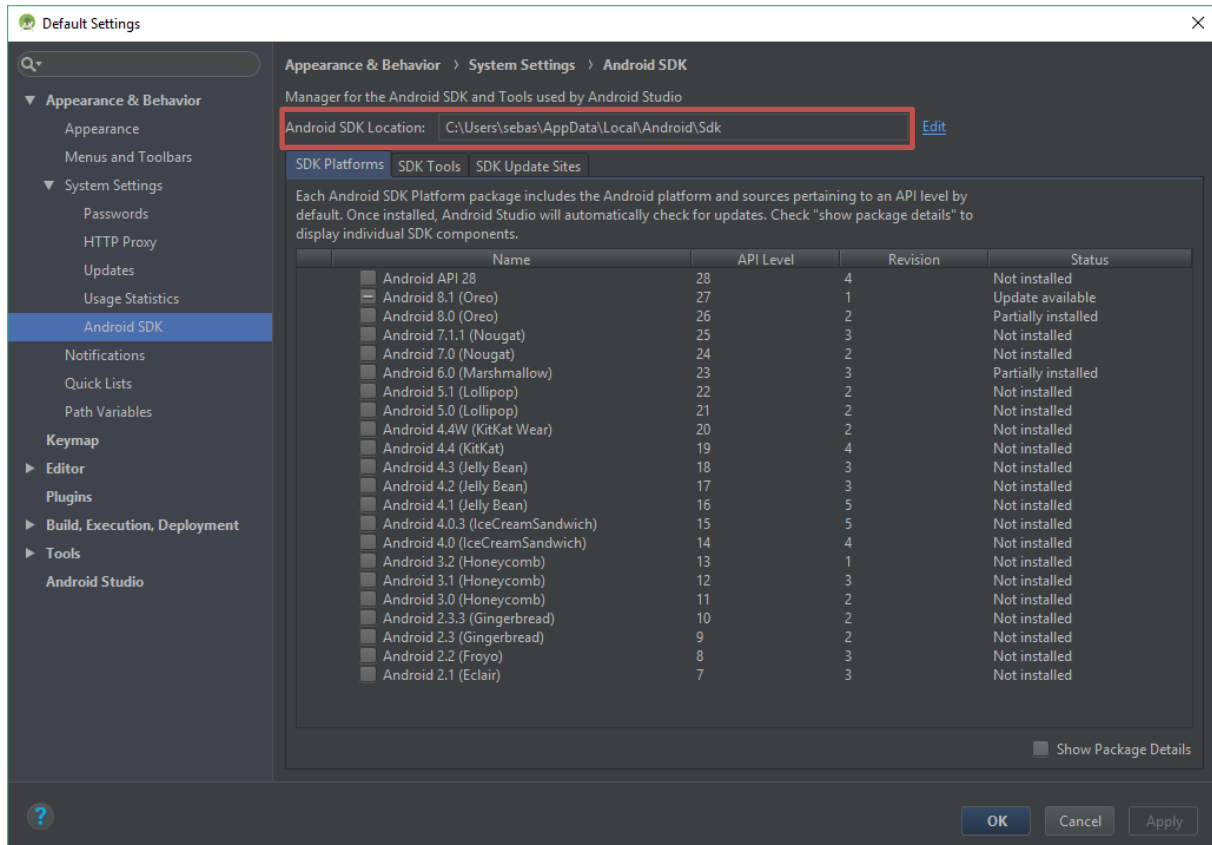


2. Step: Open Android Studio and search in the top navigation bar for "Tools". In the drop down, menu you choose "SDK Manager". A now pop-up window opens.



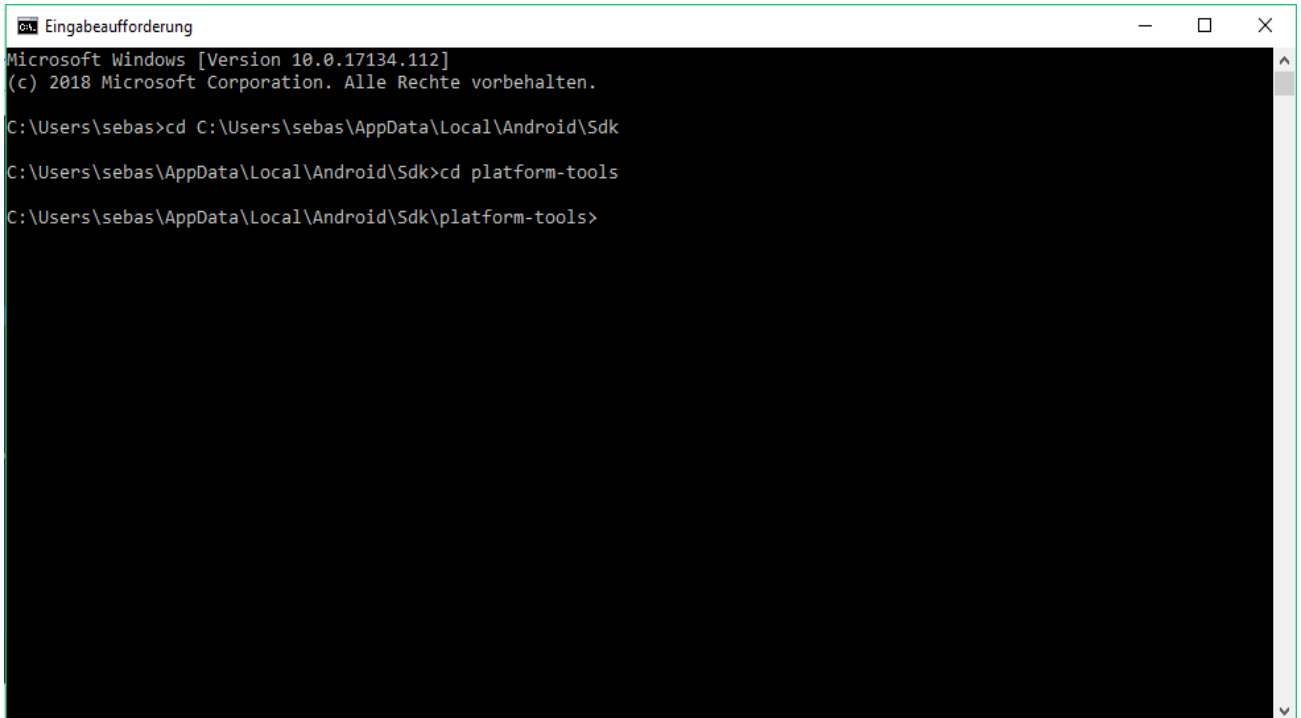
3. Step:

In the new window you search for your “Android SDK Location”. Copy the whole location-path and save it in a new word or text editor file.



4. Step:

Now open the windows command prompt from step 1 again and type in “cd [your Android SDK Location directory]”. After that you must type in “cd platform-tools”.



```
Microsoft Windows [Version 10.0.17134.112]
(c) 2018 Microsoft Corporation. Alle Rechte vorbehalten.

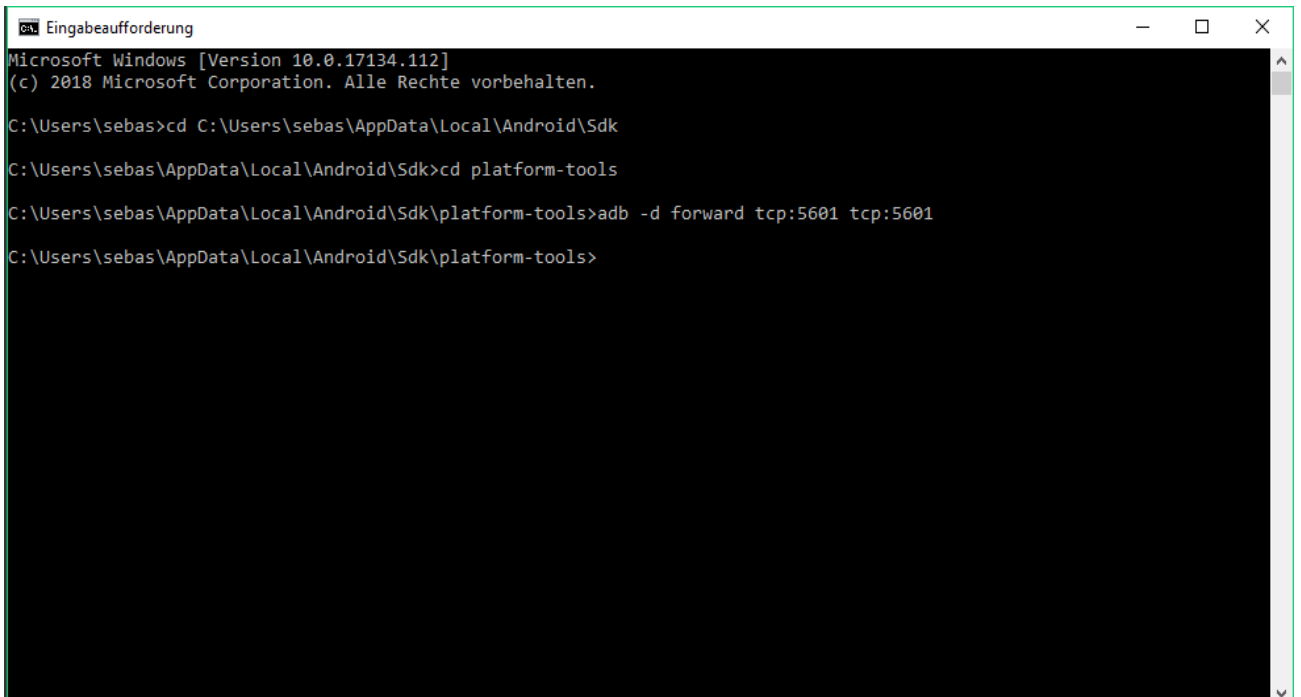
C:\Users\sebas>cd C:\Users\sebas\AppData\Local\Android\Sdk

C:\Users\sebas\AppData\Local\Android\Sdk>cd platform-tools

C:\Users\sebas\AppData\Local\Android\Sdk\platform-tools>
```

5. Step:

In the last step you must enter a command that now directly connects your smartphone with your emulated watch. So as last command you type in “adb -d forward tcp:5601 tcp:5601”. Now your emulated smartwatch and smartphone are connected.



```
Microsoft Windows [Version 10.0.17134.112]
(c) 2018 Microsoft Corporation. Alle Rechte vorbehalten.

C:\Users\sebas>cd C:\Users\sebas\AppData\Local\Android\Sdk

C:\Users\sebas\AppData\Local\Android\Sdk>cd platform-tools

C:\Users\sebas\AppData\Local\Android\Sdk\platform-tools>adb -d forward tcp:5601 tcp:5601

C:\Users\sebas\AppData\Local\Android\Sdk\platform-tools>
```

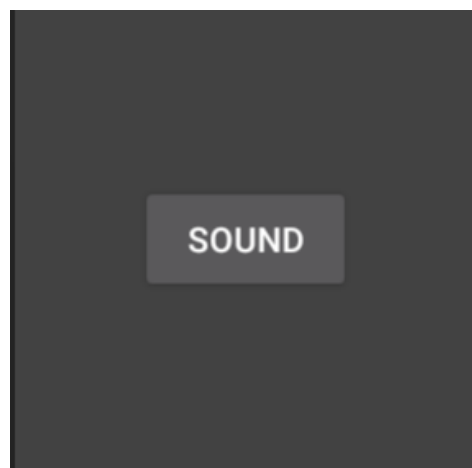
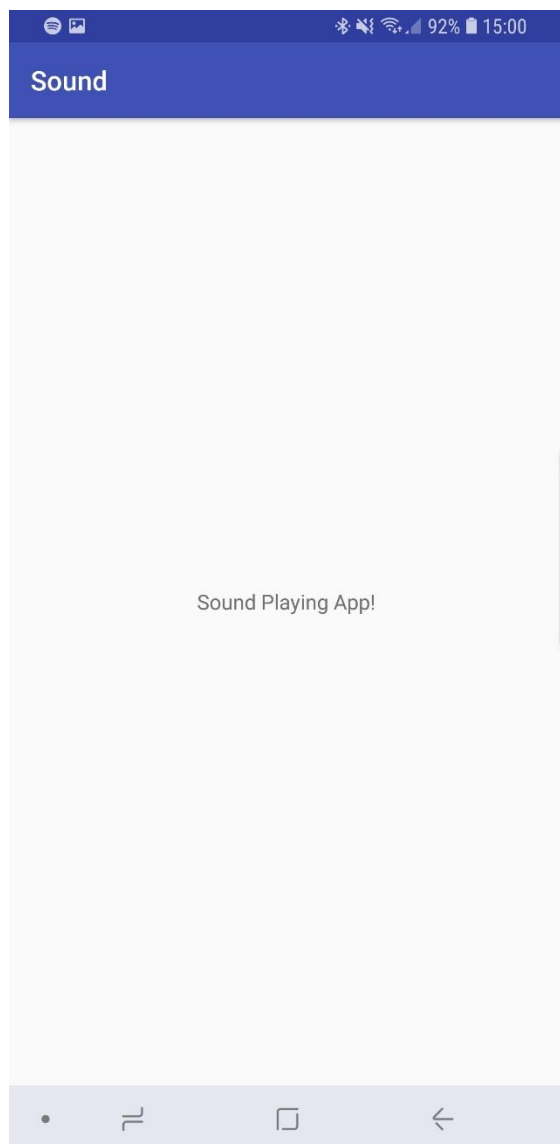
4 Applications

In the next whole section, I will concentrate on the three small and simple applications I made during this project. All three have a connection with a Samsung Gear device. So, there is always an interaction between the Gear and the smartphone when the application is used.

The structure of the three programs is similar. Every program has 2 applications, one on the smartphone and one on the Gear. The application on the smartphone is always a listener application. That means you have no executable function on the smartphone itself. It is there to wait for instructions from the Gear. The other application on the Gear has buttons that if pressed send a defined instruction to the smartphone that receives and executes it via the listener application mentioned above.

4.1 Sound application

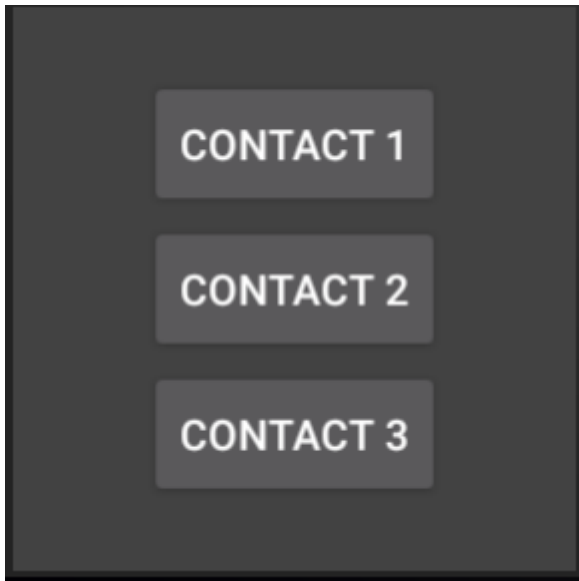
The purpose of the first application is simple, when you press the button on the Samsung Gear device, the Smartphone plays a sound. If you cannot find your smartphone, you can just use your Gear to find it.



This is how the sound app looks like on the smartphone device (left) and on the watch (right). As mentioned above the app on the smartphone has only the function to listen and wait for the message from the smartwatch. When the button on the smartwatch is pressed, the message is sent to the smartphone and it plays a sound.

4.2 E-Mail shortcut application

The email shortcut program gives you the option to define your 3 most important email contacts. It provides three buttons on the watch that then opens the email application of your choice, creates a new mail and fills in the defined e-mail, subject and text from the pressed button.

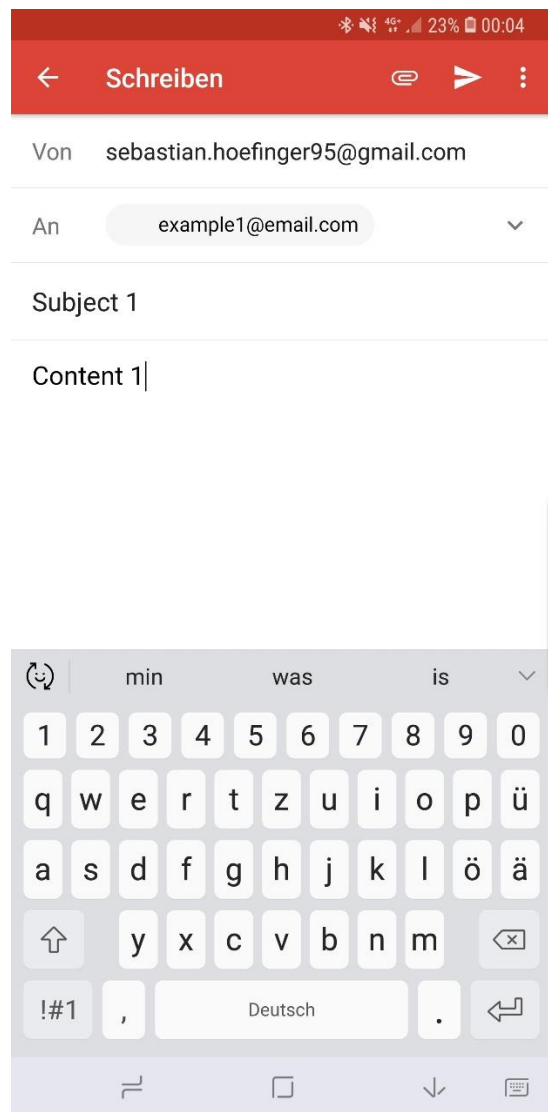
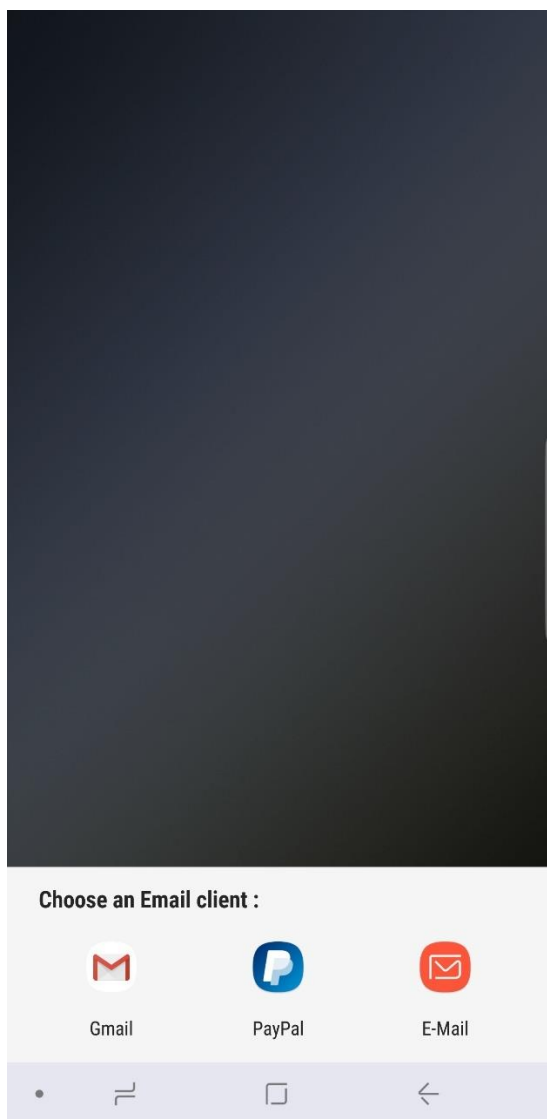


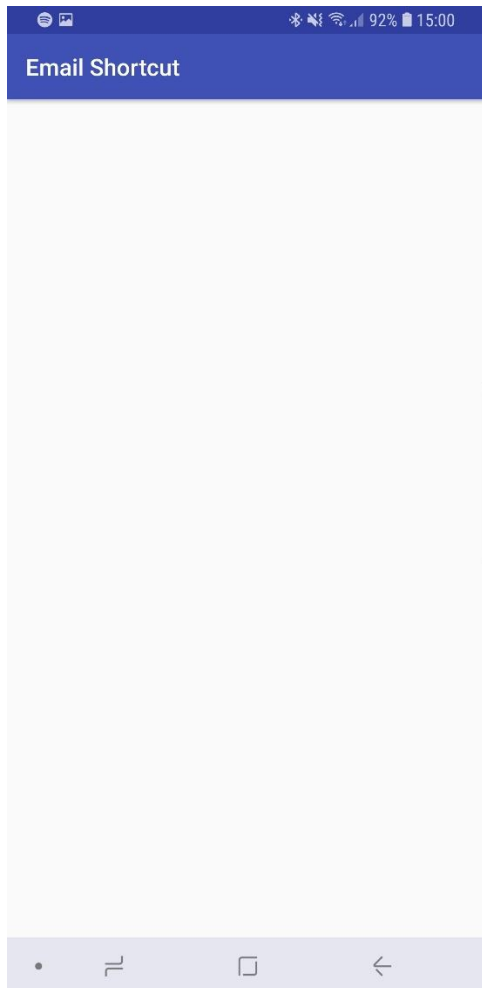
First you have the three different buttons that you can define for your 3 most important email contacts. The following screenshot shows how you can change the e-mail, the subject and the text.

```
public class ListenerServiceFromWear extends WearableListenerService {

    @Override
    public void onMessageReceived(MessageEvent messageEvent) {
        super.onMessageReceived(messageEvent);
        String path = messageEvent.getPath();
        Log.d( tag: "PATH", msg: "===== " + path);
        if (path.equals("/email1"))
        {
            Intent intent = new Intent(Intent.ACTION_SENDTO, Uri.fromParts(
                scheme: "mailto", ssp: "example1@email.com" fragment: null));
            intent.putExtra(Intent.EXTRA_SUBJECT, value: "Subject 1");
            intent.putExtra(Intent.EXTRA_TEXT, value: "Content 1");
            intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
            startActivity(Intent.createChooser(intent, title: "Choose an Email client :"));
        }
        else
        {
            if (path.equals("/email2"))
            {
                Intent intent = new Intent(Intent.ACTION_SENDTO, Uri.fromParts(
                    scheme: "mailto", ssp: "example2@email.com" fragment: null));
                intent.putExtra(Intent.EXTRA_SUBJECT, value: "Subject 2");
                intent.putExtra(Intent.EXTRA_TEXT, value: "Content 2");
                intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
                startActivity(Intent.createChooser(intent, title: "Choose an Email client :"));
            }
            else
            {
                if (path.equals("/email3"))
                {
                    Intent intent = new Intent(Intent.ACTION_SENDTO, Uri.fromParts(
                        scheme: "mailto", ssp: "example3@email.com" fragment: null));
                    intent.putExtra(Intent.EXTRA_SUBJECT, value: "Subject 3");
                    intent.putExtra(Intent.EXTRA_TEXT, value: "Content 3");
                    intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
                    startActivity(Intent.createChooser(intent, title: "Choose an Email client :"));
                }
            }
        }
    }
}
```

If you press one of these buttons your smartphone will ask you with which e-mail program you want to open the new mail. As soon as you choose your preferred e-mail client it fills the e-mail, the subject and the text automatically with the defined values





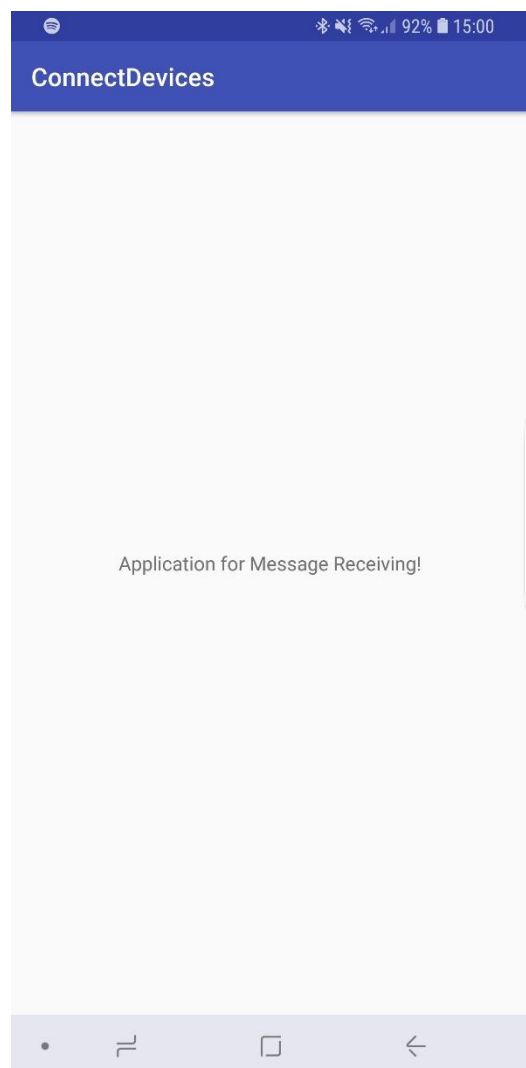
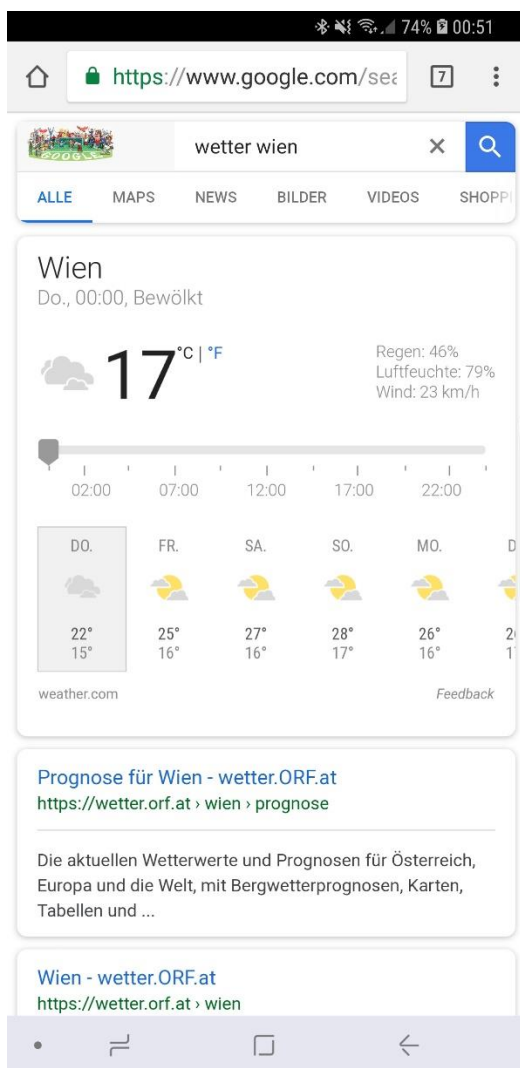
The application on the smartphone device again just acts as a listener and waits for the message from the smartwatch.

4.3 Google-search application

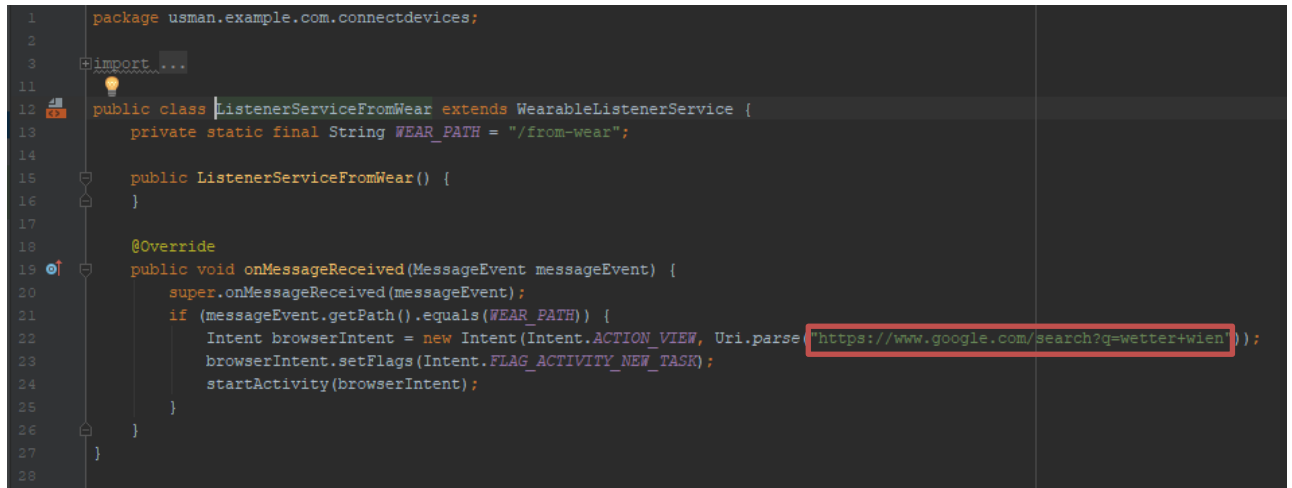
With my last application you can define a term that you want to google. When you press the button on the Gear, the Google Chrome browser opens on the smartphone and searches for the defined term.



Again, a simple button on the smartwatch.



As soon as you push this button, Google chrome opens on the smartphone and searches for the defined term. The application on the smartphone is again just a listener that waits for the command from the smartwatch.

A screenshot of the Android Studio code editor. The code is in Java and defines a class `ListenerServiceFromWear` that extends `WearableListenerService`. The class has a private static final String `WEAR_PATH` set to `"/from-wear"`. It implements the `onMessageReceived` method. Inside this method, it checks if the message path equals `WEAR_PATH`. If true, it creates an `Intent` with `Intent.ACTION_VIEW` and a URI pointing to a Google search for "wetter wien". It then sets the `FLAG_ACTIVITY_NEW_TASK` and calls `startActivity`. The URI is highlighted with a red rectangle in the original image.

```
1 package usman.example.com.connectdevices;
2
3 import ...
4
11
12 public class ListenerServiceFromWear extends WearableListenerService {
13     private static final String WEAR_PATH = "/from-wear";
14
15     public ListenerServiceFromWear() {
16     }
17
18     @Override
19     public void onMessageReceived(MessageEvent messageEvent) {
20         super.onMessageReceived(messageEvent);
21         if (messageEvent.getPath().equals(WEAR_PATH)) {
22             Intent browserIntent = new Intent(Intent.ACTION_VIEW, Uri.parse("https://www.google.com/search?q=wetter+wien"));
23             browserIntent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
24             startActivity(browserIntent);
25         }
26     }
27 }
28
```

The screenshot above shows where you can change which website your smartphone opens when you press the button on the smartwatch.

5 Conclusion

I must admit that this project was very hard and complex to work out for me. I have some very good basic knowledge in windows and with PC in general, but until this project I never have written any kind of code so, it was very difficult at the beginning to get into it. You must get familiar with the whole programming interface of android studio and it functions. For this I would suggest you watch YouTube videos because there are a lot of good videos with guides and tips for beginners.

The next step, which was the hardest one for me, is to learn the basics of Java. I did some free online courses, where they teach it to you step-by-step. I then began testing some code for myself in Android Studio and I took some time until my first real progress. During the programming of my three small applications, I signed up to different forums where I could ask for help if my code was not working. I think this is a very important step in learning Java, because you learn a lot new tricks from different people.

To sum up I have to say that in the end this project was very interesting to make. I have learned the basics of a program language and in general how applications work.

Anhang

Google-search application code:

Smartphone application code:

```
package usman.example.com.connectdevices;

import android.app.Service;
import android.content.Intent;
import android.net.Uri;
import android.os.IBinder;
import android.widget.Toast;

import com.google.android.gms.wearable.MessageEvent;
import com.google.android.gms.wearable.WearableListenerService;

public class ListenerServiceFromWear extends WearableListenerService {
    private static final String WEAR_PATH = "/from-wear";

    public ListenerServiceFromWear() {
    }

    @Override
    public void onMessageReceived(MessageEvent messageEvent) {
        super.onMessageReceived(messageEvent);
        if (messageEvent.getPath().equals(WEAR_PATH)) {
            Intent browserIntent = new Intent(Intent.ACTION_VIEW,
Uri.parse("https://www.google.com/search?q=wetter+wien"));
            browserIntent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
            startActivity(browserIntent);
        }
    }
}
```

```
package usman.example.com.connectdevices;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

Gear application code:

```
package usman.example.com.connectdevices;

import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.annotation.Nullable;
import android.support.wearable.activity.WearableActivity;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import com.google.android.gms.common.ConnectionResult;
import com.google.android.gms.common.api.GoogleApiClient;
import com.google.android.gms.common.api.ResultCallback;
import com.google.android.gms.wearable.MessageApi;
import com.google.android.gms.wearable.Node;
import com.google.android.gms.wearable.NodeApi;
import com.google.android.gms.wearable.Wearable;

public class MainActivity extends WearableActivity implements
    GoogleApiClient.ConnectionCallbacks,
    GoogleApiClient.OnConnectionFailedListener {

    private Button button;

    private Node mNode;
    private GoogleApiClient mGoogleApiClient;
    private static final String WEARABLE_PATH = "/from-wear";
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        button = findViewById(R.id.wetterwien);

        mGoogleApiClient = new GoogleApiClient.Builder(this)
            .addApi(Wearable.API)
            .addConnectionCallbacks(this)
            .addOnConnectionFailedListener(this)
            .build();

        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {

                sendMessage("Search");
            }
        });

        // Enables Always-on
        setAmbientEnabled();
    }

    private void sendMessage(String search) {
        if (mNode != null && mGoogleApiClient != null)
        {

```

```
        Wearable.MessageApi.sendMessage(mGoogleApiClient,
mNode.getId(), WEARABLE_PATH, search.getBytes());

    }

}

@Override
public void onConnected(@Nullable Bundle bundle) {
    Wearable.NodeApi.getConnectedNodes(mGoogleApiClient)
        .setResultCallback(new
ResultCallback<NodeApi.GetConnectedNodesResult>() {
        @Override
        public void onResult(@NonNull
NodeApi.GetConnectedNodesResult nodes) {
            for (Node node: nodes.getNodes())
            {
                if (node != null && node.isNearby())
                {
                    mNode = node;
                }
            }
            if (mNode == null)
            {
            }
        }
    });
}

@Override
public void onConnectionSuspended(int i) {
}

@Override
public void onConnectionFailed(@NonNull ConnectionResult
connectionResult) {
}

@Override
protected void onStart() {
    super.onStart();
    mGoogleApiClient.connect();
}

@Override
protected void onStop() {
    super.onStop();
    mGoogleApiClient.disconnect();
}
}
```

Email shortcut application code:

Smartphone application code:

```
package usman.example.com.aouslen1;

import android.content.Intent;
import android.net.Uri;
import android.util.Log;

import com.google.android.gms.wearable.MessageEvent;
import com.google.android.gms.wearable.WearableListenerService;

public class ListenerServiceFromWear extends WearableListenerService {

    @Override
    public void onMessageReceived(MessageEvent messageEvent) {
        super.onMessageReceived(messageEvent);
        String path = messageEvent.getPath();
        Log.d("PATH",
"=====
" + path);
        if (path.equals("/email1"))
        {
            Intent intent = new Intent(Intent.ACTION_SENDTO, Uri.fromParts(
                "mailto","example1@email.com", null));
            intent.putExtra(Intent.EXTRA_SUBJECT, "Subject 1");
            intent.putExtra(Intent.EXTRA_TEXT, "Content 1");
            intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
            startActivity(Intent.createChooser(intent, "Choose an Email
client :"));
        }
        else
        {
            if (path.equals("/email2"))
            {
                Intent intent = new Intent(Intent.ACTION_SENDTO,
Uri.fromParts(
                "mailto","example2@email.com", null));
                intent.putExtra(Intent.EXTRA_SUBJECT, "Subject 2");
                intent.putExtra(Intent.EXTRA_TEXT, "Content 2");
                intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
                startActivity(Intent.createChooser(intent, "Choose an Email
client :"));
            }
            else
            {
                if (path.equals("/email3"))
                {
                    Intent intent = new Intent(Intent.ACTION_SENDTO,
Uri.fromParts(
                "mailto","example3@email.com", null));
                    intent.putExtra(Intent.EXTRA_SUBJECT, "Subject 3");
                    intent.putExtra(Intent.EXTRA_TEXT, "Content 3");
                    intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
                    startActivity(Intent.createChooser(intent, "Choose an
Email client :"));
                }
            }
        }
    }
}
```

```
        }  
    }  
}  
  
}
```

```
package usman.example.com.aouslen1;  
  
import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;  
  
import com.google.android.gms.wearable.WearableListenerService;  
import java.io.File;  
  
public class MainActivity extends AppCompatActivity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
  
        ListenerServiceFromWear listenerServiceFromWear = new  
ListenerServiceFromWear();  
    }  
}
```

Smartwatch application code:

```
package usman.example.com.aouslen1;  
  
import android.os.Bundle;  
import android.support.annotation.NonNull;  
import android.support.annotation.Nullable;  
import android.support.wearable.activity.WearableActivity;  
import android.util.Log;  
import android.view.View;  
import android.widget.Button;  
import android.widget.TextView;  
import android.widget.Toast;  
  
import com.google.android.gms.common.ConnectionResult;  
import com.google.android.gms.common.api.GoogleApiClient;  
import com.google.android.gms.common.api.ResultCallback;  
import com.google.android.gms.wearable.MessageApi;  
import com.google.android.gms.wearable.Node;  
import com.google.android.gms.wearable.NodeApi;  
import com.google.android.gms.wearable.Wearable;  
  
public class MainActivity extends WearableActivity implements
```

```

GoogleApiClient.ConnectionCallbacks,
GoogleApiClient.OnConnectionFailedListener {

    private Button button1;
    private Button button2;
    private Button button3;

    private Node mNode;
    private GoogleApiClient mGoogleApiClient;
    private String wearAblePath;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        button1 = findViewById(R.id.email1);
        button2 = findViewById(R.id.email2);
        button3 = findViewById(R.id.email3);

        mGoogleApiClient = new GoogleApiClient.Builder(this)
            .addApi(Wearable.API)
            .addConnectionCallbacks(this)
            .addOnConnectionFailedListener(this)
            .build();

        button1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                wearAblePath = "/email1";
                sendMessage("email1");
            }
        });

        button2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                wearAblePath = "/email2";
                sendMessage("email2");
            }
        });

        button3.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                wearAblePath = "/email3";
                sendMessage("/email3");
            }
        });
        // Enables Always-on
        setAmbientEnabled();
    }

    private void sendMessage(String search) {
        if (mNode != null && mGoogleApiClient != null)
        {
            Wearable.MessageApi.sendMessage(mGoogleApiClient,
mNode.getId(), wearAblePath, search.getBytes())
                .setResultCallback(new
ResultCallback<MessageApi.SendMessageResult>() {
                    @Override

```



```
        public void onResult(@NonNull
MessageApi.SendMessageResult sendMessageResult)
        {
            }
        });
    }

    @Override
    public void onConnected(@Nullable Bundle bundle) {

        Wearable.NodeApi.getConnectedNodes(mGoogleApiClient)
            .setResultCallback(new
ResultCallback<NodeApi.GetConnectedNodesResult>() {
            @Override
            public void onResult(@NonNull
NodeApi.GetConnectedNodesResult nodes) {
                for (Node node: nodes.getNodes())
                {
                    if (node != null && node.isNearby())
                    {
                        mNode = node;
                        Log.d("TAG", "In Node Assigned!");
                    }
                }
                if (mNode == null)
                {
                    Log.d("TAG", "Not connected to any device!!");
                }
            }
        });
    }

    @Override
    public void onConnectionSuspended(int i) {

        Log.d("TAG", "In Connection Suspended!");
    }

    @Override
    public void onConnectionFailed(@NonNull ConnectionResult
connectionResult) {

        Log.d("TAG", "In Connection Failed!");
    }

    @Override
    protected void onStart() {
        super.onStart();
        mGoogleApiClient.connect();
    }

    @Override
    protected void onStop() {
        super.onStop();
        mGoogleApiClient.disconnect();
    }
}
```

```
}  
}
```

Sound application code:

Smartphone application code:

```
package usman.example.com.sound;  
  
import android.content.Context;  
import android.content.Intent;  
import android.media.Ringtone;  
import android.media.RingtoneManager;  
import android.net.Uri;  
import android.widget.Toast;  
  
import com.google.android.gms.wearable.MessageEvent;  
import com.google.android.gms.wearable.WearableListenerService;  
  
public class ListenerServiceFromWear extends WearableListenerService {  
    private static final String WEAR_PATH = "/from-wear";  
  
    public ListenerServiceFromWear() {  
    }  
  
    @Override  
    public void onMessageReceived(MessageEvent messageEvent) {  
        super.onMessageReceived(messageEvent);  
        if (messageEvent.getPath().equals(WEAR_PATH)) {  
            Uri notification =  
RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION);  
            Ringtone r =  
RingtoneManager.getRingtone(getApplicationContext(), notification);  
            r.play();  
        }  
    }  
}
```

```
package usman.example.com.sound;  
  
import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;  
  
public class MainActivity extends AppCompatActivity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
    }  
}
```

Smartwatch application code:

```
package usman.example.com.sound;  
  
import android.os.Bundle;
```

```
import android.support.annotation.NonNull;
import android.support.annotation.Nullable;
import android.support.wearable.activity.WearableActivity;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import com.google.android.gms.common.ConnectionResult;
import com.google.android.gms.common.api.GoogleApiClient;
import com.google.android.gms.common.api.ResultCallback;
import com.google.android.gms.wearable.MessageApi;
import com.google.android.gms.wearable.Node;
import com.google.android.gms.wearable.NodeApi;
import com.google.android.gms.wearable.Wearable;

public class MainActivity extends WearableActivity implements
    GoogleApiClient.ConnectionCallbacks,
    GoogleApiClient.OnConnectionFailedListener{

    Button sound;

    public static final String WEARABLE_MAIN = "WearableMain";
    private Node mNode;
    private GoogleApiClient mGoogleApiClient;
    private static final String WEARABLE_PATH = "/from-wear";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        sound = findViewById(R.id.sound);

        mGoogleApiClient = new GoogleApiClient.Builder(this)
            .addApi(Wearable.API)
            .addConnectionCallbacks(this)
            .addOnConnectionFailedListener(this)
            .build();

        sound.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                sendMessage("Sound");
            }
        });

        // Enables Always-on
        setAmbientEnabled();
    }

    private void sendMessage(String search) {
        if (mNode != null && mGoogleApiClient != null)
        {
            Wearable.MessageApi.sendMessage(mGoogleApiClient,
mNode.getId(), WEARABLE_PATH, search.getBytes())
                .setResultCallback(new
ResultCallback<MessageApi.SendMessageResult>() {
                    @Override
                    public void onResult(@NonNull
```

```

MessageApi.SendMessageResult sendMessageResult)
    {
        if (!sendMessageResult.getStatus().isSuccess())
        {
            Log.d(WEARABLE_MAIN, "Failed message: " +
sendMessageResult.getStatus().getStatusCode());

            }
            else
            {
                Log.d(WEARABLE_MAIN, "Message Sent!");
            }
        }
    });
}

@Override
public void onConnected(@Nullable Bundle bundle) {

    Wearable.NodeApi.getConnectedNodes(mGoogleApiClient)
        .setResultCallback(new
ResultCallback<NodeApi.GetConnectedNodesResult>() {
            @Override
            public void onResult(@NonNull
NodeApi.GetConnectedNodesResult nodes) {
                for (Node node: nodes.getNodes())
                {
                    if (node != null && node.isNearby())
                    {
                        mNode = node;
                        Log.d(WEARABLE_MAIN, "Connected to " +
node.getDisplayName());
                        Toast.makeText(MainActivity.this,
"Connected!", Toast.LENGTH_SHORT).show();
                    }
                }
                if (mNode == null)
                {
                    Log.d(WEARABLE_MAIN, "Not Connected!");
                    Toast.makeText(MainActivity.this, "Not
Connected!", Toast.LENGTH_SHORT).show();
                }
            }
        });
}

@Override
public void onConnectionSuspended(int i) {

}

@Override
public void onConnectionFailed(@NonNull ConnectionResult
connectionResult) {

}

```

```
@Override
protected void onStart() {
    super.onStart();
    mGoogleApiClient.connect();
}

@Override
protected void onStop() {
    super.onStop();
    mGoogleApiClient.disconnect();
}
}
```