



**+GF+**



**APP**

**blnet.ch**

Autor	Ian Hild und Abeeraam Rahunenthiran
Beruf	Informatiker EFZ
Lehrjahr	1. Lehrjahr
Berufsbilder	Rubén Fructuoso
Erstelldatum	28.04.2020
Semester	1 & 2

# Inhaltsverzeichnis

1	Installation von Android Studio .....	6
2	Java .....	8
2.1	Java Einführung.....	8
2.2	Datentypen.....	8
2.3	If else/Switch Anweisungen.....	9
2.3.1	If/Else .....	9
2.3.2	Switch .....	9
2.4	Variablen .....	10
2.5	Class .....	10
2.6	Methoden.....	10
2.7	Loops.....	11
2.8	Operatoren.....	12
2.9	Arrays .....	12
2.10	Comments .....	12
3	Android Studio .....	13
3.1	Neues Projekt erstellen .....	13
4	Android App Aufbau .....	15
5	Android Studios Grundlagen.....	16
5.1	Bilder hinzufügen .....	16
5.1.1	Bilder ausrichten .....	17
5.2	Links .....	18
6	Android App als APK exportieren .....	20
7	Activity erstellen .....	22
7.1	Activity in Navigation einbinden .....	22
7.1.1	XML-Code .....	22
7.1.2	Java-Code.....	23
8	Fragment erstellen.....	24
8.1	Fragment in Navigation einbinden .....	25
8.1.1	XML-Code .....	25
8.1.2	Java-Code.....	26
9	Debugging .....	29
9.1	Debug-Release über USB installieren .....	29
10	WebView .....	33
10.1	Landscape oder Portrait .....	33
11	Toasts.....	35
11.1	Was sind Toasts?.....	35
11.2	Toasts in Android Studio.....	35
12	Push Nachrichten.....	37

12.1	Push durch Knopf .....	37
12.2	Reminder erstellen .....	40
13	Login und Registration .....	42
13.1	XML-Code .....	42
13.2	Java-Code .....	43
13.3	PHP-Code.....	44
13.4	SQL-Tabelle.....	44
13.5	Passwort ändern .....	45
14	Checklisten.....	49
14.1	XML-Code .....	49
14.2	Java-Code (Werte speichern).....	49
14.3	Zweite Checkliste .....	51
15	Notenformular .....	54
15.1	Konzept .....	54
15.2	Design .....	54
15.3	Validation .....	55
15.4	User Berechtigungen.....	56
15.5	SQL-Tabelle.....	58
15.6	Absende-Request.....	58
15.7	App Script (Google Scripts) .....	60
16	Firebase .....	62
16.1	Was ist Firebase? .....	62
16.2	Firebase mit Android Studio verbinden.....	62
16.3	Cloud Messaging.....	65
16.4	Spezifizierte Benachrichtigungen .....	66
16.4.1	User Properties .....	66
16.4.2	Spezifizierte Nachricht .....	66
17	Word Press Rest API .....	68
17.1	Was ist WP Rest API? .....	68
17.2	Versuche.....	69
17.2.1	1.Versuch YouTube Tutorial.....	69
17.2.2	WebView.....	70
17.2.3	3.Versuch Internet Tutorial.....	71
18	Zeitplan .....	72
19	<b>Arbeitsjournal</b> .....	73
19.1.1	08.01.2019.....	73
19.1.2	13.01.2019.....	73
19.1.3	27.01.2020.....	74
19.1.4	28.01.2020.....	74
19.1.5	29.01.2020.....	75

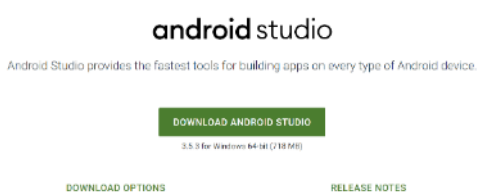
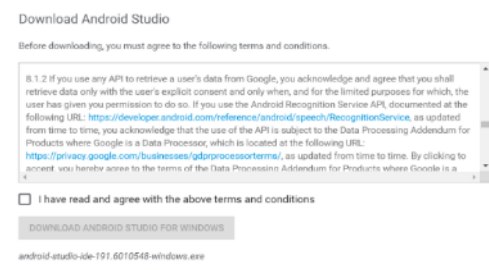
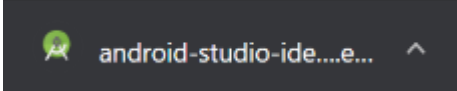

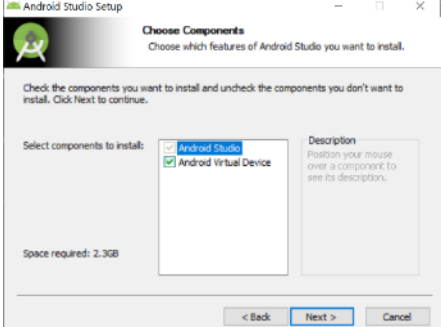
19.1.6	30.01.2020 .....	75
19.1.7	10.02.2020 .....	76
19.1.8	26.02.2020 .....	76
19.1.9	02.03.2020 .....	77
19.1.10	03.03.2020 .....	77
19.1.11	04.03.2020 .....	78
19.1.12	09.03.2020 .....	78
19.1.13	25.03.2020 .....	79
19.1.14	31.03.2020 .....	79
19.1.15	06.04.2020 .....	80
19.1.16	07.04.2020 .....	80
19.1.17	08.04.2020 .....	80
19.1.18	14.04.2020 .....	81
19.1.19	15.04.2020 .....	82
19.1.20	16.04.2020 .....	83
19.1.21	17.04.2020 .....	83
19.1.22	27.04.2020 .....	83
19.1.23	28.04.2020 .....	84
19.1.24	29.04.2020 .....	84
19.1.25	04.05.2020 .....	85
19.1.26	05.05.2020 .....	85
19.1.27	06.05.2020 .....	86
19.1.28	11.05.2020 .....	86
19.1.29	12.05.2020 .....	87
19.1.30	13.05.2020 .....	88
19.1.31	18.05.2020 .....	88
19.1.32	19.05.2020 .....	89
19.1.33	20.05.2020 .....	89
20	Testprotokoll blnet App .....	90
21	Ausgangslage .....	91
21.1	Testgegenstand .....	91
22	Testfälle .....	92
22.1	Testfall 1 App installieren .....	92
22.2	Testfall 2 – Anmelden und Registrieren .....	93
22.3	Testfall 3 Passwort zurücksetzen .....	94
22.4	Testfall 4 Notenformular .....	95
22.5	Testfall 5 Kalender .....	96
22.6	Testfall 6 Checklisten .....	97
22.7	Testfall 6 Links .....	98
23	Testergebnis .....	99
24	Glossar .....	100

25	Abbildungsverzeichnis .....	101
26	Tabellenverzeichnis .....	104
27	Index .....	105

Den Quellcode der App finden Sie auf:

<https://github.com/RaA-BLNET/BLNETAppAndroid/>

# 1 Installation von Android Studio

Erklärung	Bild
Android Studio kann und wird über den Browser heruntergeladen, um dies zu tun einfach das grüne Feld anklicken	 <p>Abbildung 1 Android Studio Download</p>
Als nächsten Schritt muss man die Lizenzbedingungen durchlesen, wenn man damit einverstanden ist, dann kann man einfach auf «gelesen und akzeptieren» drücken.	 <p>Abbildung 2 Android Studios Lizenzbedingungen</p>
Hat man die Lizenzbedingungen gelesen wird das Android Studio Setup heruntergeladen.	 <p>Abbildung 3 Android Studios Download</p>
Bei der ersten Seite des Setups wird einfach erklärt was dieses Setup macht. Hier kann man einfach auf Next drücken.	 <p>Abbildung 4 Willkommensfenster Android Studio</p>
Hier kann man die Komponenten wählen.	 <p>Abbildung 5 Installation Komponenten</p>

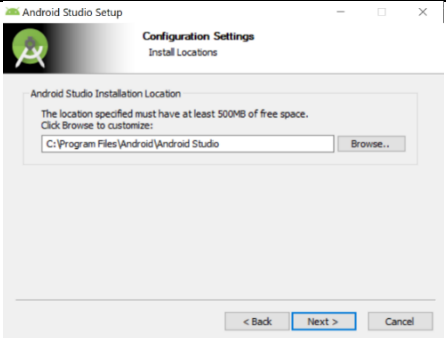
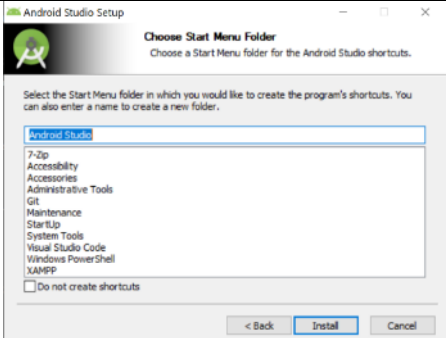
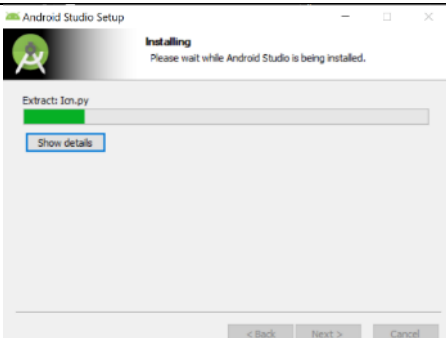

Erklärung	Bild
In diesem Fenster wird ausgewählt wo das ganze gespeichert wird.	 <p>Abbildung 6 Auswahl Speicherort</p>
Im letzten Fenster vor der Installation kann noch der Shortcut gewählt werden	 <p>Abbildung 7 Shortcut Auswahl</p>
Jetzt wird die Installation durchgeführt, dies dauert meistens so ca. 2 Minuten	 <p>Abbildung 8 Durchführung Installation</p>
Die Installation ist hiermit abgeschlossen.	 <p>Abbildung 9 Abgeschlossene Installation</p>

Tabelle 1 Android Studio Installation

## 2 Java

### 2.1 Java Einführung

Java ist eine objektorientierte Programmiersprache. Sie wurde von James Gosling entwickelt. Eine spezielle Eigenschaft ist, dass die nicht direkt über die Hardware läuft, sondern über einer virtuellen Maschine. Das wird so gelöst, damit der Code nicht für jede Plattform (z.B. Windows, MacOS...).

Java strebte hauptsächlich fünf Ziele an:

1. Sie soll einfach, objektorientiert, verteilt und vertraut sein
2. Sie soll robust sein
3. Sie soll Cross-Plattform-fähig und portabel sein.
4. Sie soll leistungsfähig sein
5. Interpretierbar und parallelisierbar sollte sich auch sein.

### 2.2 Datentypen

Datentyp	Beschreibung
char	Der Datentyp char ist ein elementarer Typ. Er umfasst Werte von 0 bis 65535, die meist als Kennzahlen für Zeichen interpretiert werden. Man kann damit aber nur 1 Zeichen speichern
int	Der Datentyp int ist wohl der am häufigsten eingesetzte primitive Typ. Er belegt 4 Bytes, was in der Regel für viele Anwendungsbereiche ausreicht.
boolean	Boolean kann einen von zwei Zuständen annehmen: true und/oder false.
double	Ist wie int, man kann aber Zahlen mit Dezimalstellen angeben.
String	Mit einem String kann man mehrere Zeichen speichern.

Tabelle 2 Datentypen



## 2.3 If else/Switch Anweisungen

### 2.3.1 If/Else

If else Statements in Java sind eigentlich fast identisch zu denen in Java Script. Es wird zuerst ein If geschrieben und dann eine Klammer geöffnet. In die Klammer wird dann die Bedingung geschrieben. Um danach noch eine Aktion schreiben zu können öffnet man noch eine geschweifte Klammer und in dieser Klammer kann man rein schreiben was passiert, wenn der Fall zutrifft oder eben nicht. Es zeigt sehr viele Ähnlichkeiten mit Java Script in diesem Fall.

```
int time = 22;
if (time < 10) {
    System.out.println("Guten Morgen.");
} else if (time < 20) {
    System.out.println("Guten Tag.");
} else {
    System.out.println("Guten Abend.");
}
// Gibt "Guten Abend" aus."
```

Else if benötigt man, wenn man mehrere Konditionen benutzen möchte.

### 2.3.2 Switch

Auch Switch Anweisungen verhalten sich relativ gleich wie in Java Script. Der Unterschied ist bei der Ausgabe. Hier ein Beispiel indem der 3 Case gesucht wird, aus diesem Grund definiert man einen Wert für den Switch Case.

```
public static void main(String[] args) {
    int month = 3;
    switch(month){
        case 1:
            System.out.println("January ");
            break;
        case 2:
            System.out.println("February");
            break;
        case 3:
            System.out.println("March")
            break;
    }
}
```

In diesem Fall wird der Case 3 also March herausgegeben, weil month mit dem Wert 3 deklariert worden ist.

## 2.4 Variablen

Man kann auch bei Java Variablen deklarieren. Man muss zuerst angeben, welcher Datentyp gebraucht wird.

```
int age;  
double salaryRequirement;  
boolean isEmployed;
```

Dann kann man Werte vergeben. Dazu muss man den Namen der Variable kennen.

```
age = 85;
```

Um es einfacher zu machen, kann man auch alles in einer Zeile schreiben.

```
int age = 31;
```

## 2.5 Class

Eine Class ist wie eine Vorlage in Java. Darin kann man Objects erstellen

```
public class Car {  
    // Scope von der Class Car beginnt nach dem Curly Brace.  
  
    public static void main(String[] args) {  
        // Scope von main() beginnt nach Curly Brace.  
  
        // Befehle  
  
    }  
    // Scope von main() ist nach dem Curly Brace zu Ende  
}  
// Scope von der Class Car ist nach dem Curly Brace zu Ende.
```

Mit public wird definiert, dass andere Classes damit interagieren können. Mit Class wird deklariert, dass dies eine Class ist.

## 2.6 Methoden

Mit Methoden kann man Classes Sachen tun lassen. Damit kann man auch mehrere Befehle zusammenfassen, die oft gebraucht werden

```
public class MyClass {  
    static void myMethod() {  
        System.out.println("I just got executed!");  
    }  
  
    public static void main(String[] args) {  
        myMethod();  
    }  
}  
// Outputs "I just got executed!"
```

Sie müssen in einer Class geschrieben werden.

In main befindet sich der Code, der ausgeführt wird. Static ist nötig, da sie aufgerufen wird, ohne dass vorher ein Objekt einer Klasse gebildet wurde. Void wird benutzt, da die Methode keinen Return-Wert besitzt. Der Parameter String[] args ist ein Array, das die vom Aufruf entgegengenommenen Parameter enthält.

## 2.7 Loops

In Java gibt es auch wie in Java Script drei verschiedenen Schleifen: For, While, Do-While.

Schleife	Beschreibung	Bild
For	Die For-Schleife nimmt man immer dann, wenn man die Anzahl der benötigten Schleifen-Durchläufe schon im Vorhinein kennt. Für die for-Schleife braucht man drei Parameter: Initialisierung, Zielwert, Schrittweite.	<pre>for(int i=0; i&lt;10; i++) {     System.out.println("Hello World " + i); }</pre>
While	Die while-Schleife führt wiederholt Anweisungen nach Prüfung einer Bedingung aus. Im Gegensatz zur for-Schleife muss bei der while-Schleife vorher eine Zählvariable deklariert werden.	<pre>int i = 0; while (i &lt; 5) {     System.out.println(i);     i++; }</pre>
Do-While	Die do-while-Schleife führt wiederholt Anweisungen abhängig von der Prüfung einer Bedingung aus. Auch anders als bei der for-Schleife ist dasselbe wie bei der while-Schleife. Es muss meist vorher eine Zählvariabel deklariert werden.	<pre>int i = 0; do {     System.out.println(i);     i++; } while (i &lt; 5);</pre>

Tabelle 3 Loops

## 2.8 Operatoren

Jede Programmiersprache enthält Operatoren. So auch in Java. Hier im Überblick die speziellen Operatoren von Java.

Operator	Beschreibung
!	Bedeutet nicht gleich wie die Bedingung. Kommt bei If Anweisung vor, wenn der Wert eben nicht gleich ist wie die Bedingung.
&&	&& wird benutzt, um mehrere Bedingungen aufzuzählen die, allerdings dass alle true, also sprich korrekt sein müssen.
	Der    Operator bedeutet nichts mehr als oder und wird bei If Statements gebraucht wo Zwei werte drin sind und nur 1 davon true sein muss

Tabelle 4 Operatoren

## 2.9 Arrays

Arrays werden verwendet, um mehrere Werte in einer einzelnen Variablen zu speichern, anstatt für jeden Wert separate Variablen zu deklarieren. Um ein Array zu deklarieren, definiert man den Variablentyp in eckigen Klammern:

```
public static void main(String[] args) {  
    String[] months = {"January", "February", "March", "April"};  
    System.out.println(months[2]);  
    //Gibt February heraus  
}
```

## 2.10 Comments

In Java hat man zwei Möglichkeiten, Sachen zu kommentieren:

Mit // kann man eine Zeile auskommentieren

```
// Gibt February heraus
```

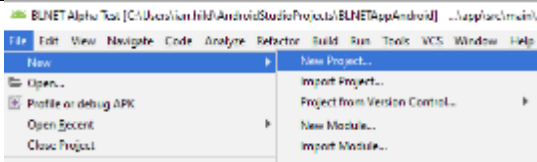
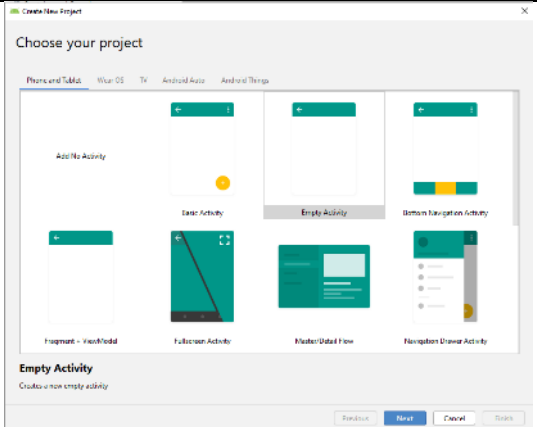
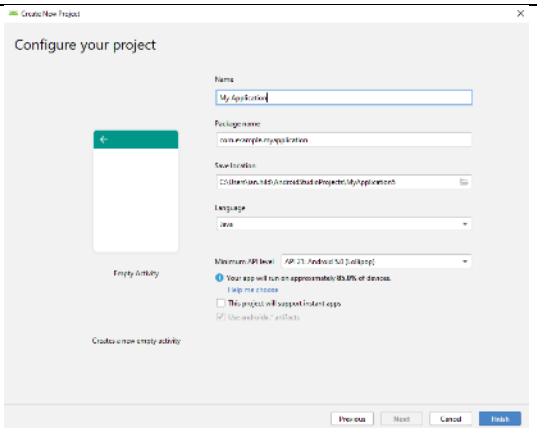
Mit /\* \*/ kann man mehrere Zeilen auskommentieren

```
/* Gibt February  
heraus */
```

### 3 Android Studio

Android Studio ist eine Software die gebraucht werden kann um Apps für Android herzustellen. Diese Software wurde im Jahr 2013 veröffentlicht. Zu diesem Zeitpunkt ist die Version 3.6.3 die neuste Version. Android Studio wurde mit Java, Kotlin, C und C++ programmiert. Ausserdem wird die Java-IDE von JetBrains (IntelliJ IDEA) verwendet.

#### 3.1 Neues Projekt erstellen

Erklärung	Bild
Um ein neues Projekt auf Android Studios zu erstellen muss man oben Links auf das Feld File.	 <p>Abbildung 10 Neues Projekt Android Studio</p>
Dann erscheint dieses Fenster, wo man ein Projekt auswählen kann. Bei der frischen Installation kommt dieses Fenster automatisch direkt nach der Installation.	 <p>Abbildung 11 Projekt auswählen</p>
Hat man ein Projekt gewählt kann man dem Projekt einen Namen geben, entscheiden wo man das Projekt abspeichern will und die Sprache wählen.	 <p>Abbildung 12 Konfiguration des Projekts</p>

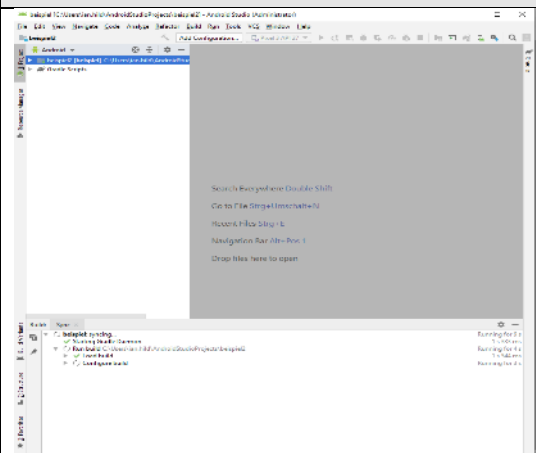
Erklärung	Bild
Nun wird das Projekt erstellt dies dauert meistens maximal eine Minute.	 <p>Abbildung 13 Projekt wird erstellt</p>

Tabelle 5 Neues Projekt Android Studio

## 4 Android App Aufbau

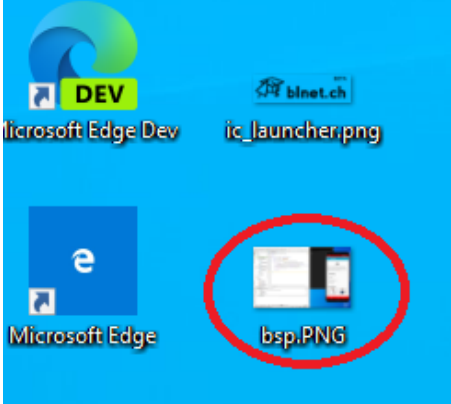
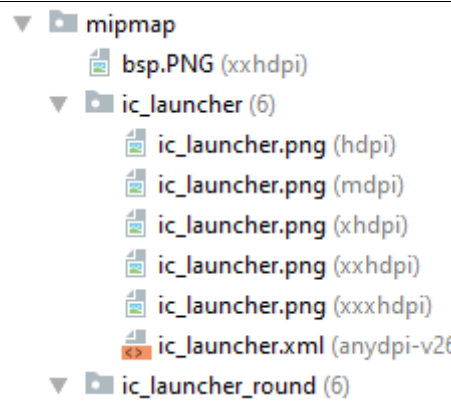
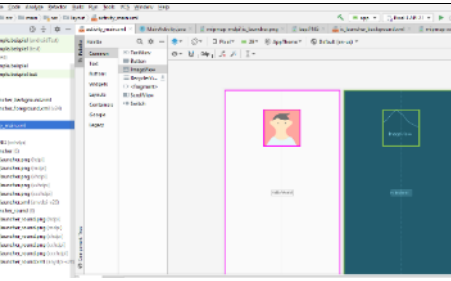
Eine Android App ist in Android Studio so aufgebaut.



Abbildung 14 Android Studio Aufbau

## 5 Android Studios Grundlagen

### 5.1 Bilder hinzufügen

Beschreibung	Bild
<p>Wenn man ein Bild vom Desktop in Android Studio haben will, kann man es per Drag&amp;Drop in einen Ordner legen.</p>	 <p>Abbildung 15 Bild auf dem Desktop</p>
<p>Danach kann man das Bild in den Ordner «mipmap» ziehen. Wenn das erledigt ist kann man das Bild überall einfügen</p>	 <p>Abbildung 16 Ordner mipmap</p>
<p>Wenn das Bild sich im Ordner befindet, kann man auf eine XML Datei gehen. dort auf Design klicken und dann ein Imageview hineinziehen.</p>	 <p>Abbildung 17 Design erstellen</p>



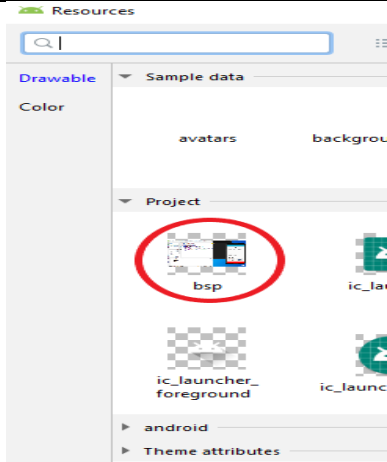
Beschreibung	Bild
Sobald dieser Platzhalter für Bilder eingefügt wurde. Kommt automatisch ein Fenster wie auf der Abbildung. Wenn diese Fenster sich öffnet kann man unter das Register Project und dort befindet sich dann das gewünschte Bild.	 <p>Abbildung 18 Bild nach Wahl auswählen</p>

Tabelle 6 Bilder in Android Studios

### 5.1.1 Bilder ausrichten

Erklärung	Bild
Wenn man das Bild nicht richtig ausrichtet klebt es in der oberen linken Ecke fest. Daher ist es wichtig Bilder zu positionieren.	 <p>Abbildung 19 Bild in der APP</p>
Wenn man das Bild zentriert haben will muss man an den vorher wissen, jetzt blauen Punkten ziehen. Wenn man den oberen Punkt an die obere Kante zieht muss man auch noch den unteren Punkt an die untere Kante ziehen. Folge ist das Bild wird über die Höhe mittig ausgerichtet, das sieht man an der blauen geschweiften Linie. Nun muss man das noch für die Breite machen und das Bild ist schön zentriert auf der App.	<div style="display: flex; justify-content: space-around;"> <div style="text-align: center;">  <p>Abbildung 21 Zentrierung (Breite)</p> </div> <div style="text-align: center;">  <p>Abbildung 20 Zentrierung (Höhe)</p> </div> </div>

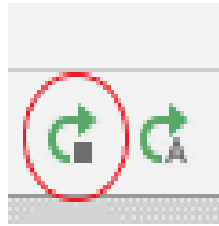
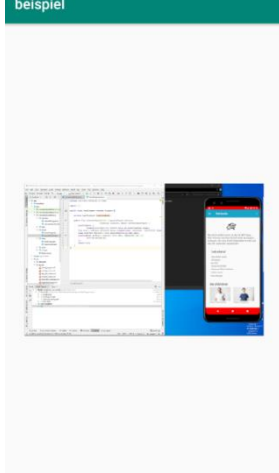

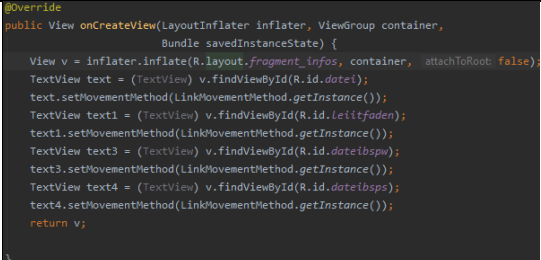
Erklärung	Bild
Wichtig ist um eine Veränderung auf der App jedes Mal den grünen Pfeil mit dem Rechteck zu drücken ansonsten passiert nichts. Auf der zweiten Abbildung sieht man das zentrierte Bild auf der App.	 <p>Abbildung 22 Debug Emulator</p>  <p>Abbildung 23 Zentriertes Bild in der App</p>

Tabelle 7 Bilder ausrichten in Android Studios

## 5.2 Links

Zu einer App gehören auch Links. Wir haben Links gebraucht um auf der Seite mit den wichtigen Infos.

Erklärung	Bild
Zuerst muss man in der Datei strings.XML den String konfigurieren. Dazu gehört Name, der Link zur Seite und der Text.	 <p>Abbildung 24 Text im strings.XML</p>
In der entsprechenden Java Klasse muss man dann noch den Link eine Funktion geben, weil er bis jetzt nur blau markiert wäre. Man sucht den Link per ID und gibt ihm die Funktion für den Link. Da wir vier Links gebraucht haben wir auch dementsprechend vier konfiguriert	 <p>Abbildung 25 Funktionsgebung</p>

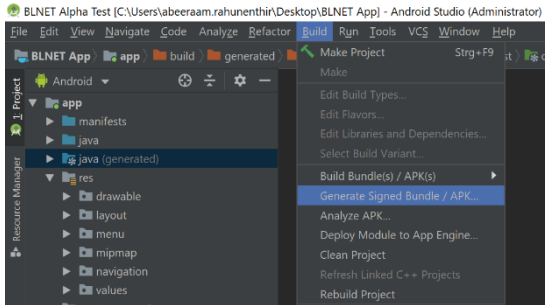
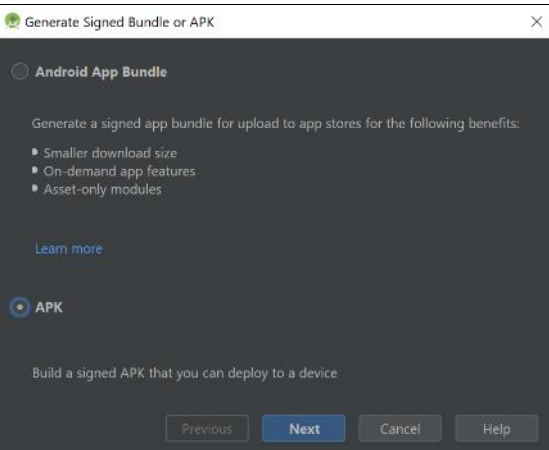
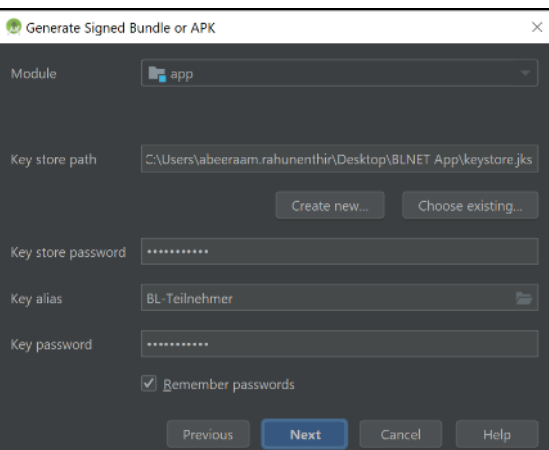
Dann muss noch in der XML Datei der richtige String angegeben werden.

```
<TextView
    android:id="@+id/datei"
    android:layout_width="409dp"
    android:layout_height="25dp"
    android:layout_marginTop="480dp"
    android:clickable="false"
    android:paddingLeft="60dp"
    android:text="@string/datei"
    android:textSize="16dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="1.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
```

Abbildung 26 XML String

## 6 Android App als APK exportieren

Damit man die App auf einem Gerät installieren kann, muss man die APK-Datei davon haben. Hier zeigen wir euch, wie man in Android Studio die App als APK exportiert.

Beschreibung	Bild
<p>Als erstes gehen wir auf «Build» und im Menü dann auf «Generated Signed Bundle /APK».</p>	 <p>Abbildung 27 Kontext Menü</p>
<p>Hier wählen wir dann APK aus, da wir ja eine APK-Datei haben möchten.</p>	 <p>Abbildung 28 Auswahl Bundle/APK</p>
<p>Hier muss man einen Keystore auswählen. Falls Sie keinen haben, müssen sie auf «Create New» klicken und die nötigen Infos dort eingeben.</p>	 <p>Abbildung 29 Key-Store</p>

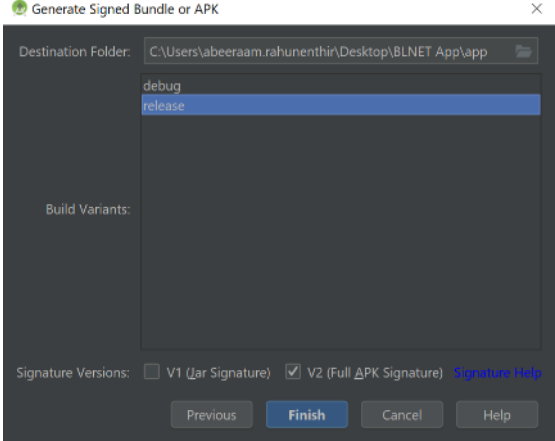
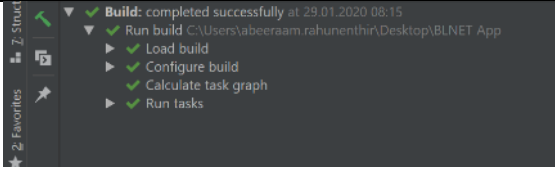
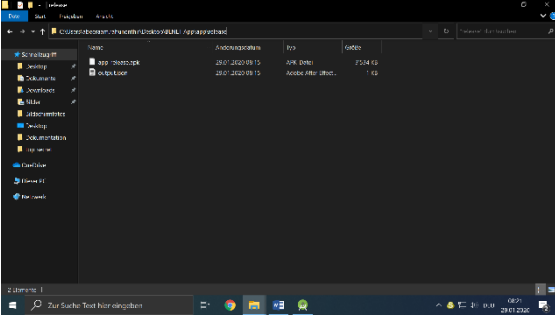
Beschreibung	Bild
Jetzt können wir auswählen, ob wir eine Debug-Variante oder eine Release-Variante haben möchten. Wir wählen hier die Release-Variante aus, da wir diese App veröffentlichen möchten. Anschliessend klicken wir auf «Finish»	 <p>Abbildung 30 Auswahl Debug/Release</p>
Jetzt warten wir, bis überall grüne Häkchen angezeigt werden. Falls ein Fehler angezeigt wird, Sie es beheben.	 <p>Abbildung 31 Build Fortschritte</p>
Die fertige APK-Datei befindet sich im Ordner ihrprojektname\app\release. Diese Datei können Sie entweder auf einem Handy ziehen und installieren oder im Internet veröffentlichen.	 <p>Abbildung 32 Zielordner</p>

Tabelle 8 APK

## 7 Activity erstellen

Activitäts sind Seiten einer App. Hier wird gezeigt, wie man sie erstellen kann.

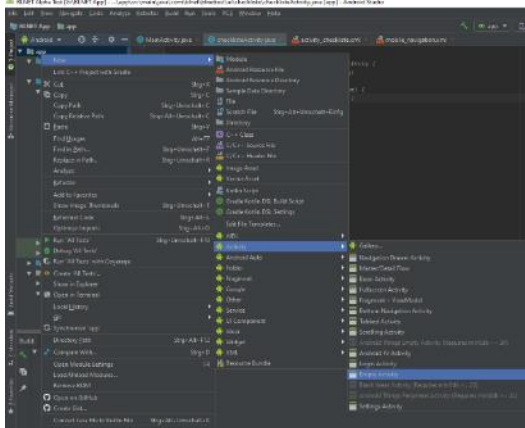
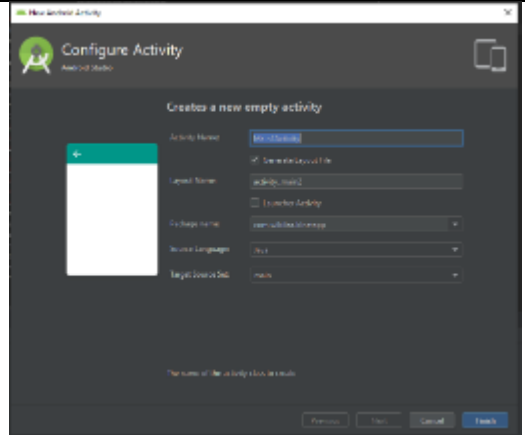
Beschreibung	Bild
Zuerst einen Rechtsklick auf den Root-Ordner «app». Dort wählt man dann New -> Activity -> Empty Activity	 <p>Abbildung 33 Activity erstellen</p>
Hier stellt man seine Einstellungen an. Wir haben nur den Namen geändert. Dann anschliessend auf «Finish» klicken.	 <p>Abbildung 34 Activity erstellen</p>

Tabelle 9 Activity erstellen

### 7.1 Activity in Navigation einbinden

Man kann auch Activity in die Navigation einbinden. Dafür müssen wir ein bisschen Java und XML anwenden.

#### 7.1.1 XML-Code

Zuerst müssen wir in der XML-Datei mobile\_navigation.XML ein paar Änderungen vornehmen. Dieses Code-Snippet müssen wir einfügen.

```
<activity
    android:id="@+id/nav_checkliste"
    android:name="com.bl.net.bl.nettest.ui.checkliste.checklisteActivity"
    android:label="@string/menu_checkliste"
    tools:layout="@layout/activity_checkliste" />
```

Damit wird in der Navigation ein Eintrag erstellt. Die Werte müsste man eventuell abändern, da sie App-spezifisch sind.

Als nächstes müssen wir in der Datei activity\_main\_drawer.XML dieses Snippet einfügen.

```
<item
    android:id="@+id/nav_checkliste"
    android:icon="@drawable/ic_menu_checkliste"
    android:title="@string/menu_checkliste" />
```

Hier können wir das Logo und den Text angeben.

Als letztes müssen wir in der Manifest-Datei angeben, dass ein Backbutton eingefügt werden soll. Das ist einfacher als es klingt. Dafür müssen wir im Eintrag von ChecklistActivity folgendes eingeben:

```
android:parentActivityName = ".ui.MainActivity"
```

Das definiert, welche Activity beim Betätigen des Backbuttons gestartet werden soll.

### 7.1.2 Java-Code

In der Java-Datei MainActivity.java müssen wir der Funktion mAppBarConfiguration die ID des Navigation-Eintrags angeben.

```
mAppBarConfiguration = new AppBarConfiguration.Builder(
    R.id.nav_home, R.id.nav_send, R.id.nav_infos,
    R.id.nav_calendar, R.id.nav_checkliste)
```

Unter diesem Befehl

```
NavigationUI.setupWithNavController(navigationView, navController);
```

schreiben wir folgenden Code:

```
navController.addOnDestinationChangedListener(new
NavController.OnDestinationChangedListener() {
    @Override
    public void onDestinationChanged(@NonNull NavController controller,
    @NonNull NavDestination destination, @Nullable Bundle arguments) {
        if (destination.getId() == R.id.nav_checkliste) {
            startActivity(new Intent(MainActivity.this,
            checklistActivity.class));
        }
    }
});
```

Hier wird bestimmt, dass die Checkliste gestartet werden soll, falls das den Eintrag geklickt wird.

Somit haben wir erfolgreich eine Activity in die Navigation eingebunden.

## 8 Fragment erstellen

Fragments braucht man, um verschiedene Seite in einer App zu erstellen. Fragments befinden sich in einer Activity. Hier wird erklärt, wie man ein Fragment erstellt und in der Navigation einbindet.

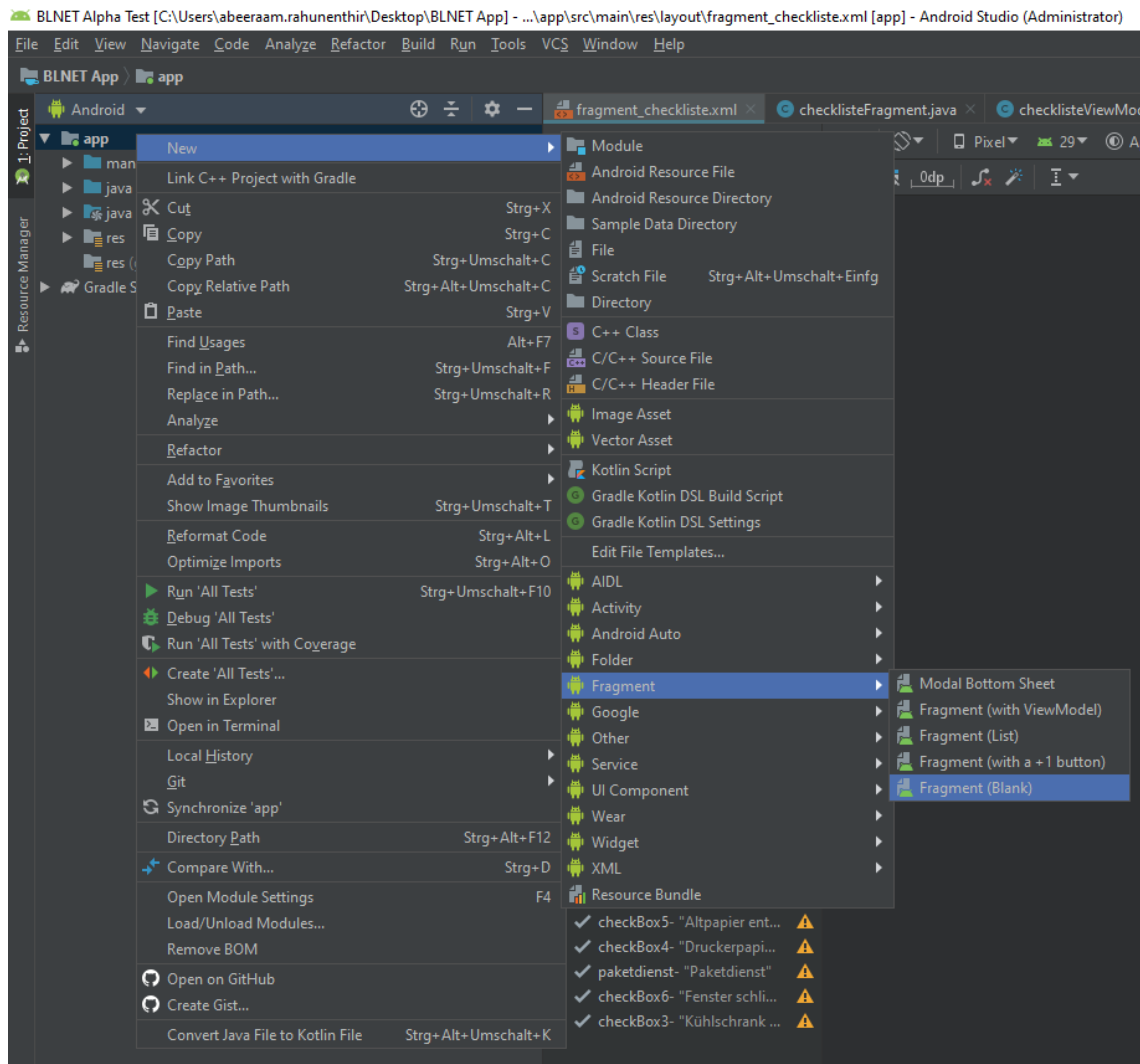


Abbildung 35 - XML-Datei für Fragment erstellen

Zuerst muss man in das Kontextmenü vom root-Ordner namens «app» auf «New», dann auf «Fragment» und zuletzt auf «Fragment (blank)».



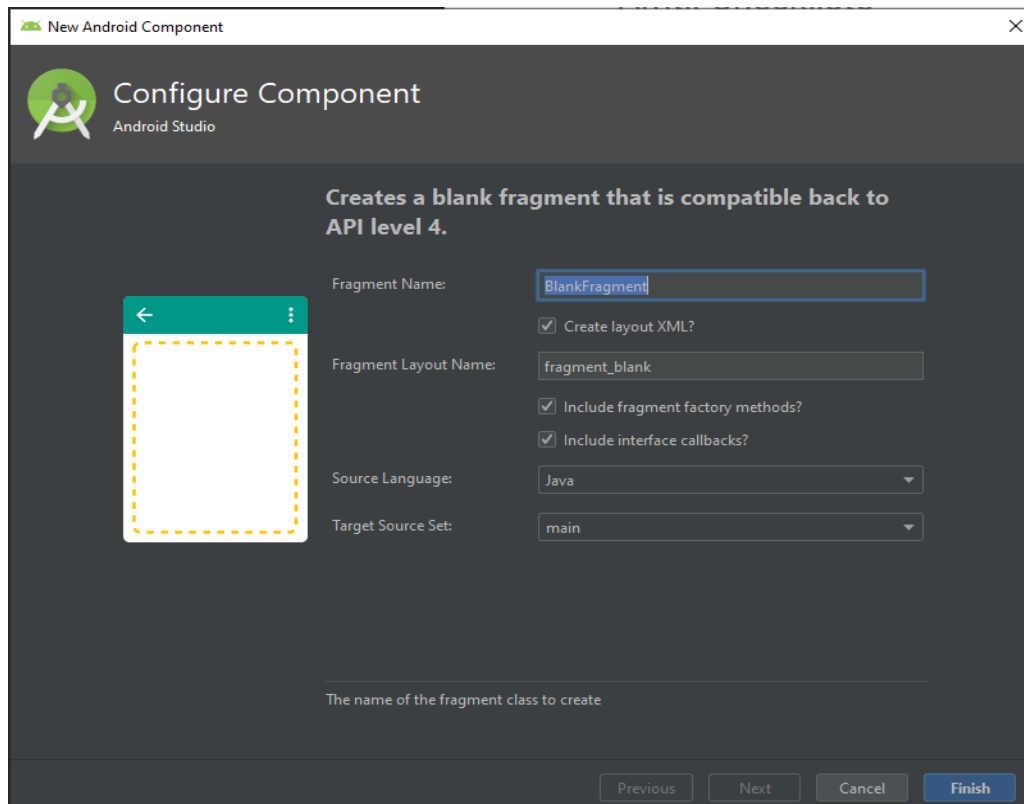


Abbildung 36 Fragment erstellen

Hier müssen wir dann den Namen des Fragments wählen. Wir haben uns für «ChecklisteFragment» entschieden.

## 8.1 Fragment in Navigation einbinden

### 8.1.1 XML-Code

Dann müssen wir das neue Fragment in der Navigation verlinken. Dafür öffnen wir die Datei «activity\_main\_drawer.XML» und dort geben wir im Element menu diesen Code ein.

```
<item
    android:id="@+id/nav_checkliste"
    android:icon="@drawable/ic_menu_checkliste"
    android:title="@string/menu_checkliste" />
```

Damit wird der Eintrag eingefügt. Bei id geben wir die ID an, bei icon die drawable Datei (Vektor) und bei title einen Namen, der in der Liste angezeigt werden soll.

Wir müssen noch in der Datei «mobile\_navigation.XML» diesen Code eingeben.

```
<fragment
    android:id="@+id/nav_checkliste"
    android:name="com.blnet.blnettest.ui.checkliste.checklisteFragment"
    android:label="@string/menu_checkliste"
    tools:layout="@layout/fragment_checkliste" />
```

Bei id geben wir wie vorher die ID der XML-Datei vom Fragment an.

Bei "name" geben wir den Paketnamen des Fragments an, bei "label" den gleich Wert wie oben bei "title", und bei "layout" geben wir den Pfad der XML-Datei vom erstellten Fragment an.

### 8.1.2 Java-Code

Jetzt müssen wir noch im Java-Code ein paar Änderungen vornehmen. Dafür müssen wir als allererstes eine ViewModel Java-Datei erstellen. Der Name der Datei ist [fragmentname]ViewModel, in unserem Fall ChecklisteViewModel.

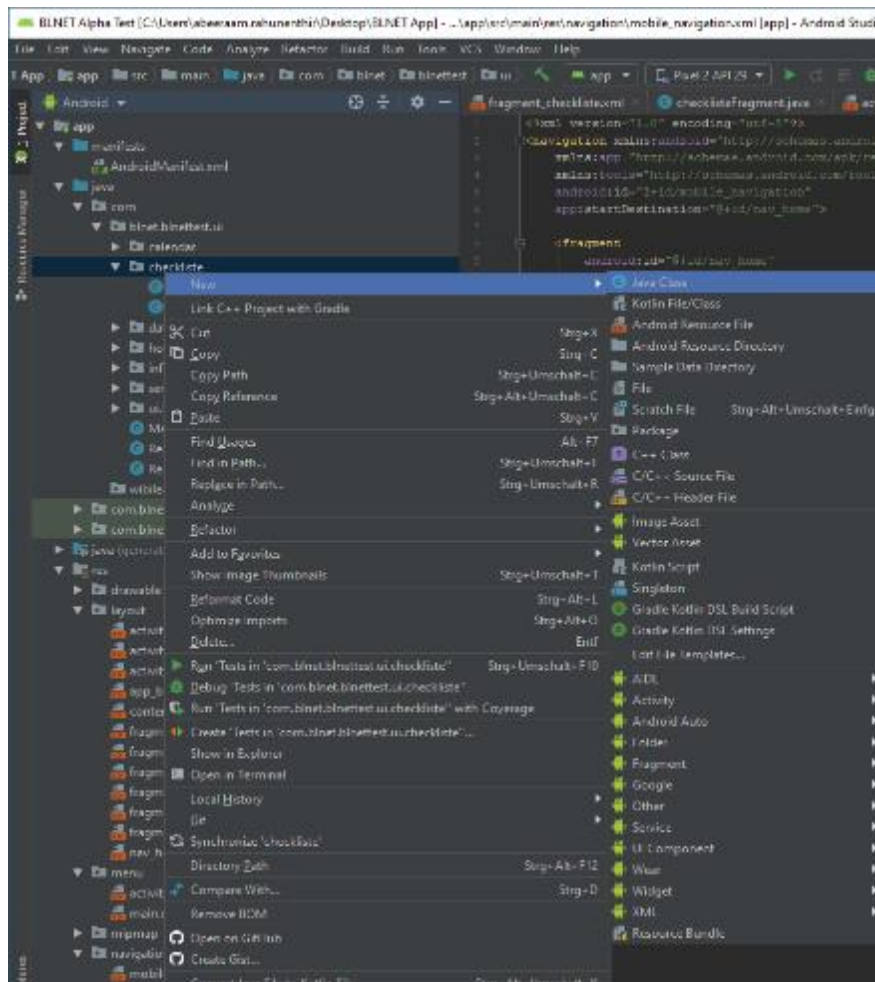


Abbildung 37 Java Code

Dort geben wir folgenden Code ein:

```
package com.blnet.blnettest.ui.checkliste;

import androidx.lifecycle.LiveData;
import androidx.lifecycle.MutableLiveData;
import androidx.lifecycle.ViewModel;

public class checklisteViewModel extends ViewModel {
    private MutableLiveData<String> mText;

    public checklisteViewModel() {
        mText = new MutableLiveData<>();
    }

    public LiveData<String> getText() {
        return mText;
    }
}
```

Dieser Code sorgt dafür, dass die Eigenschaften aus der XML-Datei für das Design auf der Seite selber angezeigt wird.

Dann müssen wir in `checkListeFragment.java` diesen Code eingeben (Den evtl. vorher vorhandenen Code einfach löschen):

```
package com.blnet.blnettest.ui.checkliste;

import android.os.Bundle;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.fragment.app.Fragment;
import androidx.lifecycle.Observer;
import androidx.lifecycle.ViewModelProviders;

import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;

import com.blnet.blnettest.R;

public class checklisteFragment extends Fragment {

    private com.blnet.blnettest.ui.checkliste.checklisteViewModel
    checklisteViewModel;

    public View onCreateView(@NonNull LayoutInflater inflater,
                             ViewGroup container, Bundle savedInstanceState) {
        checklisteViewModel =
        ViewModelProviders.of(this).get(com.blnet.blnettest.ui.checkliste.checklisteVi
        ewModel.class);
        View root = inflater.inflate(R.layout.fragment_checkliste, container,
        false);
        final TextView textView = root.findViewById(R.id.text_checkliste);
        checklisteViewModel.getText().observe(this, new Observer<String>() {
            @Override
            public void onChanged(@Nullable String s) {
                textView.setText(s);
            }
        });
        return root;
    }
}
```

Dieser Code verlinkt den ViewModel und den XML-Layout miteinander.

Jetzt müsste man das Fragment benutzen können.

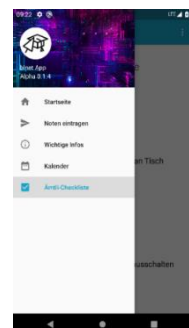


Abbildung 38  
Fragment

## 9 Debugging

Debugging ist ein wichtiger Bestandteil beim Programmieren, um Probleme zu beheben. In den Logs steht warum z.B. die App abgestürzt ist usw.

### 9.1 Debug-Release über USB installieren

Man kann mit dem Handy Apps debuggen. Dafür braucht man Android Studio und ein USB-Kabel.

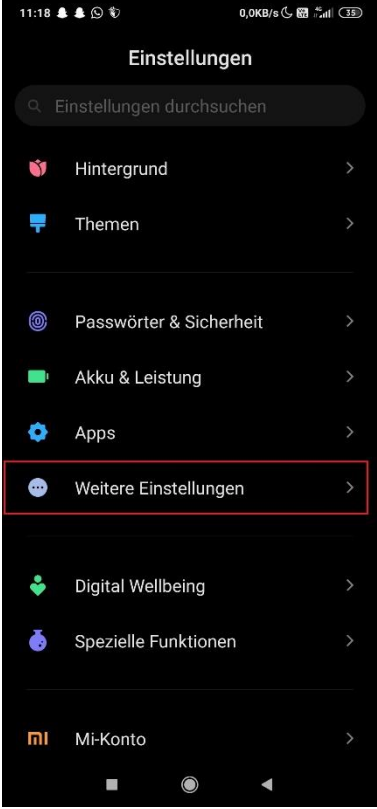

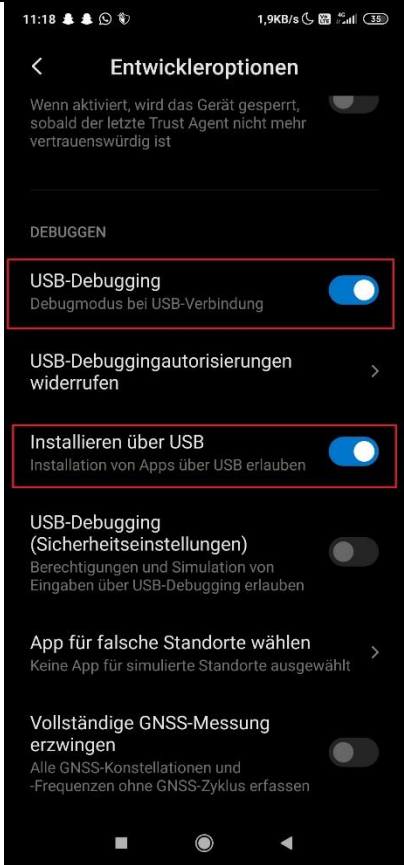
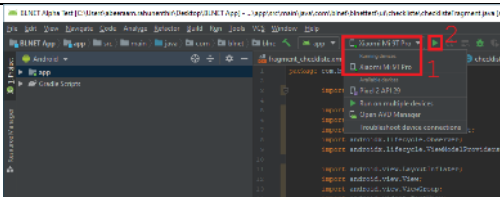
Beschreibung	Bild
Als Erstes muss man USB-Debugging aktivieren. Dies ist je nach Android-Version/ROM anders. Hier wird es für Xiaomi erklärt. In den Einstellungen gehen wir auf «Weitere Einstellungen».	 <p>The screenshot shows the 'Einstellungen' (Settings) menu of an Android phone. The 'Weitere Einstellungen' (More settings) option is highlighted with a red rectangle. The menu includes options like Hintergrund, Themen, Passwörter &amp; Sicherheit, Akku &amp; Leistung, Apps, Weitere Einstellungen, Digital Wellbeing, Spezielle Funktionen, and Mi-Konto.</p>

Abbildung 39 USB-Debugging Weitere Einstellungen

Beschreibung	Bild
<p>Hier wählen wir dann «Entwickleroptionen» aus. Um die Entwickleroptionen freizuschalten, muss man bei den Softwareinfos auf «Buildnummer» spammen. Auch dies ist je nach Android-Version/ROM anders.</p>	 <p>Abbildung 40 USB-Debugging Entwickleroptionen</p>

Beschreibung	Bild
<p>In den Entwickleroptionen schalten wir «USB-Debugging» und «Installieren über USB» ein.</p>	 <p>Abbildung 41 USB-Debugging einschalten</p>
<p>Dann schliessen wir das Smartphone mit dem USB-Kabel am PC an. Dort starten wir Android Studio und wählen oben das Gerät aus. In unserem Fall ist es ein Xiaomi Mi 9T Pro. Dann klicken wir aus das Play-Zeichen.</p>	 <p>Abbildung 42 USB-Debugging von Android Studio</p>


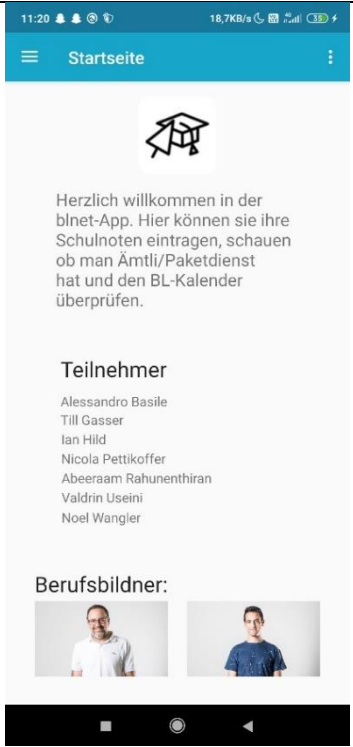
Beschreibung	Bild
<p>Diese Meldung müsste jetzt auf dem Handy erscheinen. Logischerweise wählen wir hier jetzt «Installieren» aus.</p>	 <p>Abbildung 43 App installieren</p>
<p>Die App startet nach einer kurzen Zeit, bis die App fertig installiert ist. Jetzt kann man auch in Android Studio die Logs des Gerätes anschauen.</p>	 <p>Abbildung 44 Die App auf dem Handy</p>

Tabelle 10 USB-Debugging



## 10 WebView

WebView wird gebraucht, um eine Website/HTML-Code in einer Android-App anzuzeigen. Die Funktionsweise ist sehr ähnlich wie bei iframe aus HTML. Man kann auch damit die Anzeige bestimmter ID's oder Classes verhindern.

Um WebView zu verwenden, muss man zuerst bei XML die style-informationen festlegen.

```
<WebView android:id="@+id/calendar_html"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"/>
```

Hier werden Höhe & Breite des WebViews-Elements angegeben. Bei android:id sollte man eine eigene ID angeben, damit wir es später im Java ansprechen können.

In der Java-Datei können angegeben, wie es funktionieren soll.

```
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
    Bundle savedInstanceState) {
    View v = inflater.inflate(R.layout.fragment_calendar, container, false);
    WebView webView = (WebView) v.findViewById(R.id.calendar_html);
    webView.getSettings().setJavaScriptEnabled(true);
    webView.setWebViewClient(new WebViewClient() {
        @Override
        public void onPageFinished(WebView view, String url) {
            super.onPageFinished(view, url);
            view.loadUrl("javascript:(function() { " +
"document.getElementById('masthead').style.display='none';})()");
            view.loadUrl("javascript:(function() { " +
"document.getElementById('colophon').style.display='none';})()");
        }
    });
    webView.loadUrl("https://www.blnet.ch/kalender/");
    return v;
}
```

In diesem Code-Snippet wird angegeben, dass im fragment\_calendar nach einem WebView mit der id calendar\_html gesucht werden soll, und darauf die folgenden Regeln angewendet werden sollen. Wenn die Seite fertig geladen ist, sollten header und footer nicht mehr angezeigt werden auf der Website blnet.ch/kalender/.

### 10.1 Landscape oder Portrait

In unserem Fall haben wir den Kalender der blnet-Website in unsere App mit einem WebView reingeladen. Allerdings wurde der Kalender in der Portrait-Ansicht nur halb angezeigt. Erst als das Handy gedreht wurde konnte der Kalender komplett angezeigt werden. Dieses Problem haben wir mit einem If Else Statement gelöst.

```
int orientation = getResources().getConfiguration().orientation;

if (orientation == Configuration.ORIENTATION_PORTRAIT) {
    webView.setVisibility(View.GONE);
    AlertDialog.Builder builder = new AlertDialog.Builder(getContext());
    builder.setTitle(getString(R.string.titlecal))
        .setMessage(getString(R.string.infocal))
}
```

```
        .setNegativeButton(getString(R.string.okcal), null)
        .create()
        .show();
    } else {
        webView.setVisibility(View.VISIBLE);
    }
```

Wir haben gegoogelt nach der Bedingung für Hochformat und Querformat. Danach haben wir nach der Bedingung, das WebView versteckt und die Fehlermeldung konfiguriert. Danach hingeschrieben was sonst passieren soll (beim Querformat) also das WebView anzeigen.

## 11 Toasts

### 11.1 Was sind Toasts?

Toasts sind Benachrichtigungen von Ereignissen die für eine kurze Zeit auf dem Display erscheinen. Zum Beispiel, wenn man sich irgendwo anmeldet sieht man ab und zu eine Benachrichtigung wo steht: «Anmeldung erfolgreich» oder «Anmeldung felgeschlagen».

### 11.2 Toasts in Android Studio

Erklärung	Bild
Zu aller erst muss man logischerweise einen Button reinziehen. Den Button kann man dann nach Wunsch ausrichten. Denn Button kann man auf einer Datei nach Wahl im Ordner App>res>Layout machen.	 <p>Abbildung 45 Auslöser Button</p>
In der Datei string.XML muss man noch einen «name» und ein Label definieren. In diesem Fall ist der Name "toast" und das Label des Buttons "Drücke mich".	<pre>&lt;string name="toast"&gt;Drücke mich!&lt;/string&gt;</pre> <p>Abbildung 46 Definition des Buttons im string.XML</p>
Nun muss man noch (in der Datei wo man denn Button reingezogen hat) ein onClick Statement setzten. Man schreibt also «android:onClick="showToast"». Einfach gesagt soll er den Toast zeigen, wenn der Button gedrückt wird.	<pre>&lt;Button     android:id="@+id/button2"     android:layout_width="wrap_content"     android:layout_height="wrap_content"     android:onClick="showToast"     android:text="@string/toast"     app:layout_constraintBottom_toBottomOf="parent"     app:layout_constraintEnd_toEndOf="parent"     app:layout_constraintHorizontal_bias="0.498"     app:layout_constraintStart_toStartOf="parent"     app:layout_constraintTop_toTopOf="parent"     app:layout_constraintVertical_bias="0.202" /&gt;</pre> <p>Abbildung 47 onClick Statement Button</p>
In diesem Schritt definiert man den Button in der Java Datei (MainActivity.java).	<pre>package com.example.myapplication;  import ...  public class MainActivity extends AppCompatActivity {      private AppBarConfiguration mAppBarConfiguration;     private Button btn;      @Override</pre> <p>Abbildung 48 Button Definition Java</p>


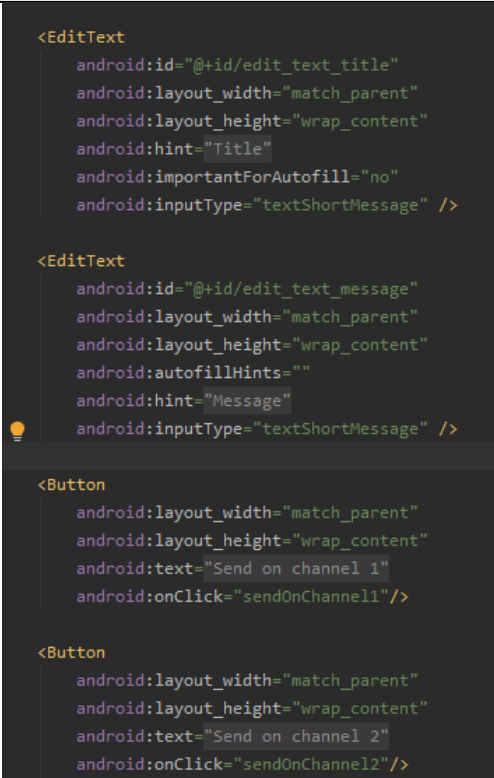
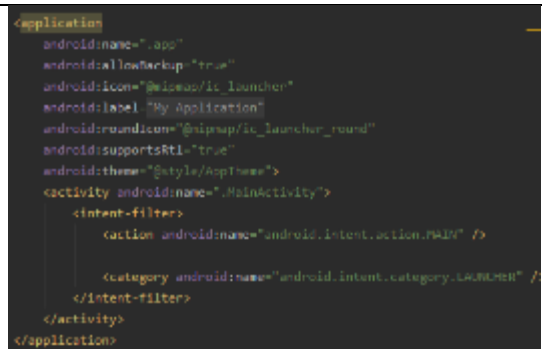

<p>Danach muss man <code>findViewById</code> einsetzen indem man die ID des buttons angibt. In unserem Fall die ID <code>button</code>.</p>	<pre><code>@Override protected void onCreate(Bundle savedInstanceState) {     super.onCreate(savedInstanceState);     setContentView(R.layout.activity_main);     btn = (Button) findViewById(R.id.button);     Toolbar toolbar = findViewById(R.id.toolbar);     setSupportActionBar(toolbar);     FloatingActionButton fab = findViewById(R.id.fab);     fab.setOnClickListener((view) -&gt; {</code></pre> <p>Abbildung 49 <code>findViewById</code></p>
<p>Zum Schluss muss man noch angeben was es ausgeben soll. Es sollte also auf Mausklick den Text Registrierung erfolgreich ausgeben.</p>	<pre><code>public void showToast(View v) {     Toast.makeText(getApplicationContext(), text: "Registrierung erfolgreich", Toast.LENGTH_SHORT).show(); }</code></pre> <p>Abbildung 50 Ausgabe des Toasts</p>
<p>Das kommt raus, wenn man denn Button gedrückt hat.</p>	 <p>Abbildung 51 Ausgabe nach dem Click</p>

Tabelle 11 Toasts

## 12 Push Nachrichten

### 12.1 Push durch Knopf

Erklärung	Bild
<p>Zuerst sollte man die Seite designen, bei der die Aktion durchgeführt werden soll. Im Bild sieht man das wir 2 Textfelder erstellt haben und zwei Buttons zum Senden. Im ersten Textfeld wird der Titel oder auch Sender angegeben und beim zweiten wird das die eigentliche Nachricht reingeschrieben. Das kann man in irgendeiner Layout-datei sprich XML-Datei machen.</p>	 <pre> &lt;EditText     android:id="@+id/edit_text_title"     android:layout_width="match_parent"     android:layout_height="wrap_content"     android:hint="Title"     android:importantForAutofill="no"     android:inputType="textShortMessage" /&gt;  &lt;EditText     android:id="@+id/edit_text_message"     android:layout_width="match_parent"     android:layout_height="wrap_content"     android:autoFillHints=""     android:hint="Message"     android:inputType="textShortMessage" /&gt;  &lt;Button     android:layout_width="match_parent"     android:layout_height="wrap_content"     android:text="Send on channel 1"     android:onClick="sendOnChannel1"/&gt;  &lt;Button     android:layout_width="match_parent"     android:layout_height="wrap_content"     android:text="Send on channel 2"     android:onClick="sendOnChannel2"/&gt; </pre> <p>Abbildung 52 Layout der Seite</p>
<p>In der Android Manifest.XML-Datei muss man noch unter «application» android:name“.app” eintragen.</p>	 <pre> &lt;application     android:name=".app"     android:allowBackup="true"     android:icon="@mipmap/ic_launcher"     android:label="@string/app_name"     android:roundIcon="@mipmap/ic_launcher_round"     android:supportRtl="true"     android:theme="@style/AppTheme"&gt;     &lt;activity android:name=".MainActivity"&gt;         &lt;intent-filter&gt;             &lt;action android:name="android.intent.action.MAIN" /&gt;             &lt;category android:name="android.intent.category.LAUNCHER" /&gt;         &lt;/intent-filter&gt;     &lt;/activity&gt; &lt;/application&gt; </pre> <p>Abbildung 53 Manifest Dokument</p>

Erklärung	Bild
<p>In diesem Schritt konfiguriert man die Push Nachricht. Für diese Konfiguration haben wir eine externe Java-datei gemacht. Zu aller erst schreibt man onCreate (Shortcut) um eine Methode zu bestimmen. In diesem Fall wird es eine onCreate Methode. Die Methode wird automatisch erstellt, wenn man die Kürzel eingibt. Danach muss man die Kanäle bestimmen (die zwei erstellten Buttons). Um dies zu erreichen machen wir für diese Buttons zwei Konstanten. Diese Konstanten schreiben wir oberhalb des onCreate. Als nächstes haben wir in einer separaten Methode mit dem Namen (createNotificationChannel()) den Kanal erstellt, die Methode selber folgt direkt danach. Diese Art Benachrichtigungen funktioniert erst ab der Oreo Version von Android. Deswegen setzen wir ein if Statement, um nachzufragen ob die Benachrichtigungen funktionieren würden. Wenn die Version Oreo oder höher ist wird die Bedingung ausgeführt. Danach kann man doch die Notifications konfigurieren und einstellen</p>	 <pre> public class App extends Application {     public static final String Channel_1_ID = "channel1";     public static final String Channel_2_ID = "channel2";      @Override     public void onCreate() {         super.onCreate();          createNotificationChannels();     }      private void createNotificationChannels(){         if (Build.VERSION.SDK_INT &gt;= Build.VERSION_CODES.O){             NotificationChannel channel1 = new NotificationChannel(                 Channel_1_ID,                 "Channel 1",                 NotificationManager.IMPORTANCE_HIGH             );             channel1.setDescription("This is Channel 1");              NotificationChannel channel2 = new NotificationChannel(                 Channel_2_ID,                 "Channel 2",                 NotificationManager.IMPORTANCE_LOW             );             channel2.setDescription("This is Channel 2");              NotificationManager manager = getSystemService(NotificationManager.class);             manager.createNotificationChannel(channel1);             manager.createNotificationChannel(channel2);         }     } } </pre> <p>Abbildung 54 Eigene Java-datei</p>

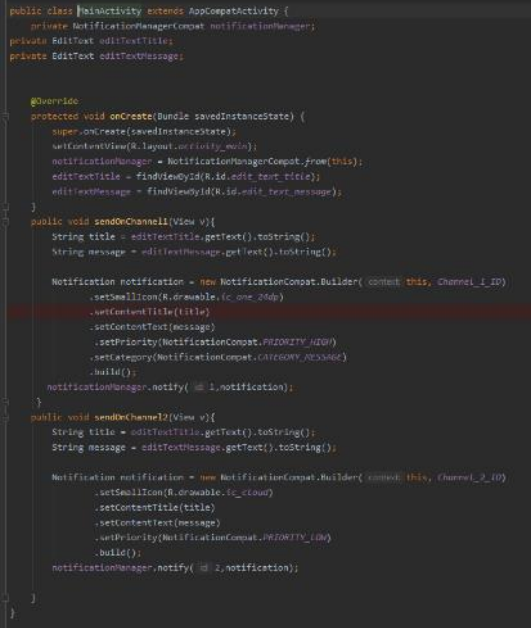
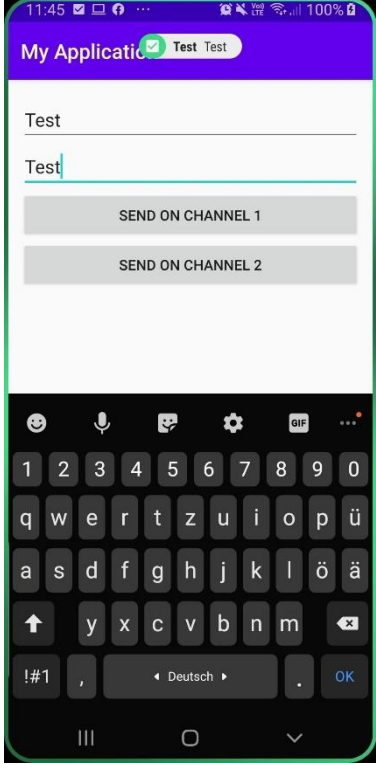
Erklärung	Bild
<p>In der Main.java Datei kann man dann noch einstellen wie sich die Nachricht verhalten soll und was passieren soll, wenn diese angeschickt wird.</p>	 <p>Abbildung 55 Konfiguration Benachrichtigungen</p>
<p>Unser Script ergab folgendes Ergebnis.</p>	 <p>Abbildung 56 Endresultat</p>

Tabelle 12 Push Knopf

## 12.2 Reminder erstellen

Erklärung	Bild
<p>Zuerst haben wir uns um das Layout der Seite gekümmert dafür haben wir einfach einen TimePicker (Uhr) und einen Button auf die Seite getan.</p>	 <pre> &lt;RelativeLayout     xmlns:android="http://schemas.android.com/apk/res/android"     xmlns:app="http://schemas.android.com/apk/res-auto"     xmlns:tools="http://schemas.android.com/tools"     android:layout_width="match_parent"     android:layout_height="match_parent"     tools:context=".MainActivity"&gt;      &lt;LinearLayout         android:layout_width="match_parent"         android:layout_height="wrap_content"         android:layout_centerVertical="true"         android:orientation="vertical"&gt;          &lt;TimePicker             android:id="@+id/timePicker"             android:layout_width="match_parent"             android:layout_height="wrap_content" /&gt;          &lt;Button             android:id="@+id/buttonSetAlarm"             android:layout_width="match_parent"             android:layout_height="wrap_content"             android:text="Alarm aktivieren" /&gt;      &lt;/LinearLayout&gt;  &lt;/RelativeLayout&gt; </pre> <p>Abbildung 57 MainActivity Datei</p>
<p>Als zweites muss man eine zusätzliche Java-Klasse hinzufügen. Die muss man dann noch in der Manifest-datei angeben. Für das muss man unter dem activity Tag einen Receiver Tag öffnen und den Namen der Klasse eingeben.</p>	 <pre> &lt;category android:name="android.intent.category.DEFAULT" /&gt; &lt;/intent-filter&gt; &lt;/activity&gt;  &lt;receiver     android:name=".BroadcastReminder"     android:enabled="true"     android:exported="true" /&gt;  &lt;/application&gt;  &lt;/manifest&gt; </pre> <p>Abbildung 58 AndroidManifest Datei</p>



<p>In der externen Datei konfiguriert man die Nachricht an sich. Man bestimmt also was drin steht oder was für eine Art die Nachricht ist (Anruf, Alarm).</p>	<pre>package com.example.myapplication;  import androidx.appcompat.app.AppCompatActivity;  public class BroadcastReminder extends BroadcastReceiver {     public static final String Channel_2_ID = "channel1";     @Override     public void onReceive(Context context, Intent intent) {         Notification.Builder builder2 =             new Notification.Builder(context, Channel_2_ID)                 .setSmallIcon(R.drawable.ic_android_black_24dp)                 .setContentTitle("Message")                 .setContentText("Hello!")                 .setPriority(Notification.PRIORITY_DEFAULT);         NotificationManagerCompat notificationManagerCompat =             NotificationManagerCompat.from(context);         notificationManagerCompat.notify(1, builder2.build());     } }</pre> <p>Abbildung 59 MainActivity(Java) Datei</p>
<p>Hier macht man noch ein if-else Statement damit es für alle Versionen passt. Ab API 23 wurden Sachen geändert die relevant sind für Notifications.</p>	<pre>@Override protected void onCreate(Bundle savedInstanceState) {     super.onCreate(savedInstanceState);     setContentView(R.layout.activity_main);      timePicker = (TimePicker) findViewById(R.id.timePicker);      findViewById(R.id.buttonSetAlarm).setOnClickListener((v) -&gt; {         Calendar calendar = Calendar.getInstance();         if (Build.VERSION.SDK_INT &gt;= 23) {             calendar.set(                 calendar.get(Calendar.YEAR),                 calendar.get(Calendar.MONTH),                 calendar.get(Calendar.DAY_OF_MONTH),                 timePicker.getHour(),                 timePicker.getMinute(),                 second: 0             );         } else {             calendar.set(                 calendar.get(Calendar.YEAR),                 calendar.get(Calendar.MONTH),                 calendar.get(Calendar.DAY_OF_MONTH),                 timePicker.getHour(),                 timePicker.getMinute(),                 second: 0             );         }     }); }</pre> <p>Abbildung 60 Externe Java Datei</p>

Tabelle 13 Reminder erstellen

Diese Variante funktionierte zu Beginn allerdings ging es auf einmal nicht mehr. Wir versuchten es zum Funktionieren zu bringen allerdings sahen wir ein das es nicht mehr funktioniert und beschlossen dann dass wir Firebase brauchen werden.

## 13 Login und Registration

Wir haben Login und Registration programmiert, damit nicht Fremde Zugriff auf gewisse private Inhalte (z.B. Notenformular etc.) haben. Wir haben dafür das Tutorial von Tonikami TV benutzt.

Was brauchten wir bevor wir anfangen konnten? Natürlich Android Studio, Internetzugriff, einen Webserver, einen SQL-Server und ein bisschen PHP- und SQL-Kenntnisse.

### 13.1 XML-Code

Wir fangen mit dem Design der App an. Das wird alles mit XML gelöst. Aus diesem Code:

```
<?XML version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:id="@+id/container"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingTop="@dimen/activity_vertical_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
android:paddingBottom="@dimen/activity_vertical_margin"
tools:context="com.blnet.blnettest.ui.ui.login.LoginActivity">

    <EditText
        android:id="@+id/username"
        android:layout_width="322dp"
        android:layout_height="57dp"

        android:hint="@string/prompt_email"
        android:inputType="textEmailAddress"
    />
</androidx.constraintlayout.widget.ConstraintLayout>
```

geht weiter...

wird:

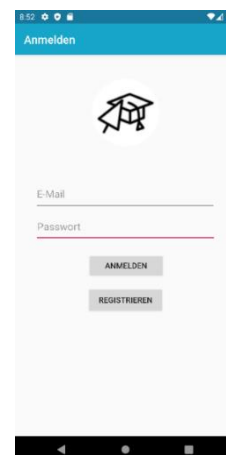


Abbildung 61 BL App Login Fenster

Java war dann aber ein bisschen herausfordernder als XML.

## 13.2 Java-Code

In Java mussten wir das logische Teil des Logins realisieren. Dazu gehört z.B. was passieren sollte, wenn das Passwort falsch ist, wenn man sich 2-mal mit der gleichen E-Mail-Adresse registrieren möchte oder was passiert, wenn ein Account erfolgreich erstellt wurde.

Dafür mussten wir 4 verschiedene Java-Dateien erstellen 2 für das Login und 2 für die Registration. In der ersten Datei (LoginActivity.java/RegisterActivity.java) befindet sich der Code für das Anmelde/Registrierfenster und in der anderen (RegisterRequest.java/LoginRequest.java) befindet sich der Code für die Abfrage mittels HTTP-POST der Benutzerdaten die sich auf dem Server befinden oder der Code für die Übertragung der Benutzerdaten, die in das Registrierfenster eingegeben wurde.

Hier ein kleiner Code-Snippet:

```
private static final String LOGIN_REQUEST_URL =
"https://app.bl.net.ch/scripts/login.php";
private Map<String, String> params;

public LoginRequest(String email, String password, Response.Listener<String>
listener){
    super (Request.Method.POST, LOGIN_REQUEST_URL, listener, null);
    params = new HashMap<>();
    params.put ("email", email);
    params.put ("password", password);
}
```

Hier werden die Werte «email» und «password» dem Server geschickt, der ein PHP-Skript geladen, der die Daten aus der App mit den Daten mit der Datenbank vergleicht. Falls die Daten übereinstimmen, wird «success= true» an die App geschickt, falls die Daten falsch sind wird «success= false» an die App geschickt. Wie der PHP-Code aufgebaut ist, wird noch gezeigt.

```
private static final String REGISTER_REQUEST_URL =
"https://tante gemmas.000webhostapp.com/androidlogin/register.PHP";
private Map<String, String> params;

public RegisterRequest(String vorname, String nachname, String email, String
password, Response.Listener<String> listener){
    super (Method.POST, REGISTER_REQUEST_URL, listener, null);
    params = new HashMap<>();
    params.put ("vorname", vorname);
    params.put ("nachname", nachname);
    params.put ("email", email);
    params.put ("password", password);
}
```

Bei der Registration werden die Daten, die in den Eingabefeldern eingegeben wurden, direkt zum Server geschickt. Dort werden die Daten verarbeitet und zurückgewiesen. falls z.B. die eingegebene E-Mail schon in der Tabelle der Datenbank existiert.

### 13.3 PHP-Code

Im PHP hatten wir die meisten Probleme. Wenn es nur nach dem Code in Java ginge, hätten wir das Login schon lange fertig. Nach dem üK307 hatten wir aber schnell erkannt, dass das falsch war und Probleme verursachte. Die PHP-Meldungen hatten `<br/>` Elemente eingefügt, und das konnte die App nicht in eine JSON-Datei umwandeln. Also haben wir mit

```
error_reporting(0);
```

alle Meldungen deaktiviert. Seitdem werden keine JSON-Parse Fehler mehr ausgegeben. Also musste der Fehler irgendwo anders liegen. Wir haben den SQL-Befehl angeschaut und haben gesehen, dass die dort angegebene Tabelle nicht existiert. Wir haben dann den neuen Namen angegeben und dann funktionierte das Registrieren einwandfrei.

Als nächstes knüpften wir uns das Login vor. Dort haben wir festgestellt, dass das PHP-Skript alle Spalten der Tabelle abrufen, aber wir nur einen Teil davon im PHP-Skript einer Variable zugewiesen hatten. Das haben wir behoben und dann ging das Login auch ohne Probleme.

### 13.4 SQL-Tabelle

Damit wir die Daten auch irgendwo speichern konnten, mussten wir eine Datenbank erstellen. Dafür verwendeten wir bis zur Version 0.4.0 einen Freehoster. Dann haben wir auf Hostpoint gewechselt (als Sub-Domäne von blnet.ch). Die Datenbank + Tabelle zu erstellen war nicht schwierig. Man musste nur die richtigen Datensätze auswählen und `auto_increment` bei der ID anhängen.

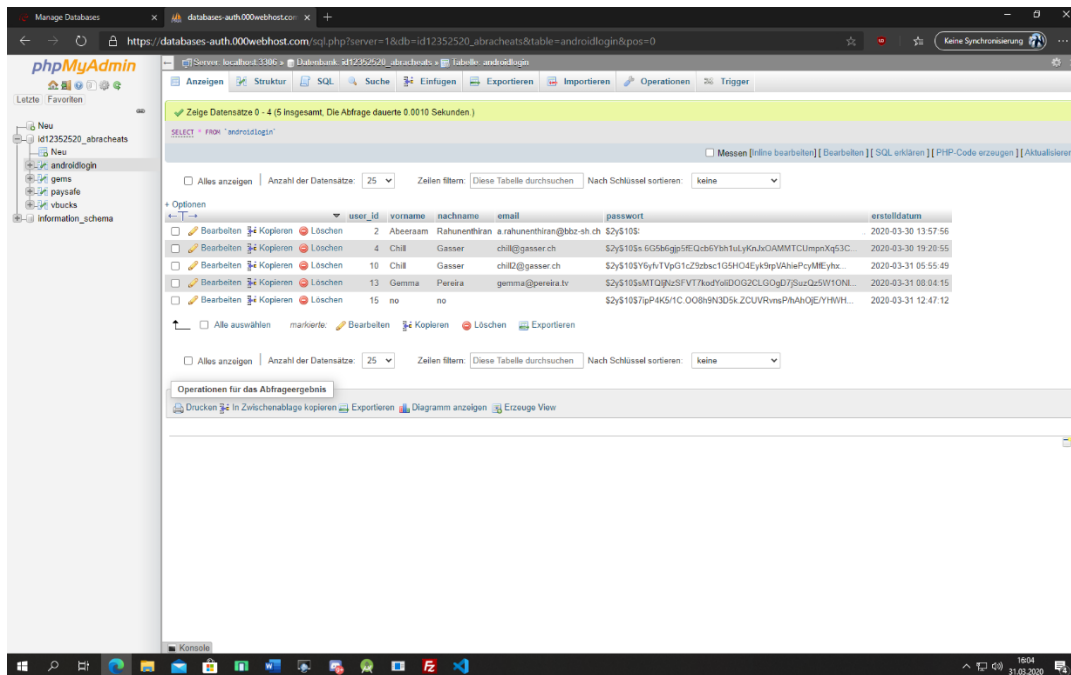


Abbildung 62 phpMyAdmin WebGUI

#	Nama	Typ	Kollation	Attribute	Null	Standard	Kommentare	Extra	Aktion
1	user_id	Int(11)			Nein	kein(e)		AUTO_INCREMENT	Bearbeiten Löschen Mehr
2	vorname	varchar(100)	utf8_unicode_ci		Nein	kein(e)			Bearbeiten Löschen Mehr
3	nachname	varchar(100)	utf8_unicode_ci		Nein	kein(e)			Bearbeiten Löschen Mehr
4	email	varchar(100)	utf8_unicode_ci		Nein	kein(e)			Bearbeiten Löschen Mehr
5	password	varchar(9000)	utf8_unicode_ci		Nein	kein(e)			Bearbeiten Löschen Mehr
6	erstelldatum	timestamp			Nein	current_timestamp()			Bearbeiten Löschen Mehr

Abbildung 63 Datenbankeinträge

auto\_increment sorgt dafür, dass die Zahl der ID bei jedem Eintrag um 1 grösser wird. Das hilft, die verschiedenen Benutzer auseinander zu halten. Das Erstelldatum (timestamp) ist nicht unbedingt nötig für die Funktion.

## 13.5 Passwort ändern

Damit die User der App das Passwort ändern können, mussten wir eine dafür eine Unterseite erstellen. In dieser Unterseite werden folgenden Daten eingegeben: E-Mail, Altes Passwort und neues Passwort.

Wir haben zwei Volley-Request mit if else Anweisungen ineinander verschachtelt. (Volley ist das http-API für Android). Die App prüft zuerst, ob die Login-Daten (E-Mail und Altes Passwort) korrekt sind. Wenn ja, wird die nächste Anfrage ausgeführt, wo das Passwort geändert wird. Wenn nicht, wird eine Fehlermeldung ausgegeben. Dafür die bereits vorhandene Class vom Login (loginRequest) verwendet.

```
Response.Listener<String> responseListener = new Response.Listener<String>()
{
```

```
@Override
public void onResponse(String response) {
    try {
        JSONObject jsonResponse = new JSONObject(response);
        successlogin = jsonResponse.getBoolean("success");
        if (successlogin == true) {
            loadingButton.setVisibility(View.GONE);

getWindow().clearFlags(WindowManager.LayoutParams.FLAG_NOT_TOUCHABLE);
            RequestQueue queue =
Volley.newRequestQueue(pwresetActivity.this);
            StringRequest stringRequest = new
StringRequest(Request.Method.POST,
"https://app.blnet.ch/scripts/resetpass.php", new Response.Listener<String>()
{
                @Override
                public void onResponse(String response) {
                    loadingButton.setVisibility(View.GONE);

getWindow().clearFlags(WindowManager.LayoutParams.FLAG_NOT_TOUCHABLE);
                    CharSequence toasttext =
getString(R.string.successchange);
                    int duration = Toast.LENGTH_LONG;
                    Toast successinfo =
Toast.makeText(pwresetActivity.this, toasttext, duration);
                    successinfo.show();
                    Intent backToMain = new Intent(pwresetActivity.this,
LoginActivity.class);
                    startActivity(backToMain);
                }
            }, new Response.ErrorListener() {
                @Override
                public void onErrorResponse(VolleyError error) {
                    loadingButton.setVisibility(View.GONE);

getWindow().clearFlags(WindowManager.LayoutParams.FLAG_NOT_TOUCHABLE);
                    AlertDialog.Builder builder = new
AlertDialog.Builder(pwresetActivity.this);
                    builder.setTitle(getString(R.string.error))
                        .setMessage(getString(R.string.neterror))

.setNegativeButton(getString(R.string.tryagainLogin), null)
                        .create()
                        .show();
                }
            }) {
                protected Map<String, String> getParams() {
                    Map<String, String> params = new HashMap<String,
String>();

                    params.put("email", rstemailstr);
                    params.put("newpassword", rstnewpassstr);
                    return params;
                }

                @Override
                public Map<String, String> getHeaders() throws
AuthFailureError {
```

```
        Map<String, String> params = new HashMap<String,
String>();
        params.put("Content-Type", "application/x-www-form-
urlencoded");
        return params;
    }
};
queue.add(stringRequest);
} else {
    loadingButton.setVisibility(View.GONE);

getWindow().clearFlags(WindowManager.LayoutParams.FLAG_NOT_TOUCHABLE);
AlertDialog.Builder builder = new
AlertDialog.Builder(pwresetActivity.this);
    builder.setTitle(getString(R.string.error))
        .setMessage(getString(R.string.wrongemailoldpwd))
        .setNegativeButton(getString(R.string.tryagainlogin),
null)
        .create()
        .show();
    }
} catch (Exception e) {
    e.printStackTrace();
}
}
};
```

Während der Zeit, wo eine Anfrage bearbeitet wird, erscheint ein Lade-Icon, um dem Nutzer zu signalisieren, dass er warten muss. Ausserdem werden auch noch alle Eingaben in der App deaktiviert. Das Passwort wird mit einem PHP-File geändert. Die eingegebenen Werte werden an das PHP-File geschickt, wo die Werte in den SQL-Befehl eingesetzt werden. Dann wird der Befehl ausgeführt.

```
<?php
error_reporting (0);
require("password.php");

$con = mysqli_connect('blnetch.mysql.db.internal', 'blnetch_app',
'XXXXXXX', 'blnetch_app');

$email = mysqli_real_escape_string($con, $_REQUEST["email"]);
$password = mysqli_real_escape_string($con, $_REQUEST["newpassword"]);

function registerUser() {
    global $con, $email, $password;
    $passwordHash = password_hash($password, PASSWORD_DEFAULT);
    $statement = mysqli_prepare($con, "UPDATE androidlogin SET password =
? WHERE email = ?");
    mysqli_stmt_bind_param($statement, "ss", $passwordHash, $email);
    mysqli_stmt_execute($statement);
    mysqli_stmt_close($statement);
}
```

```
$response = array();  
    registerUser();  
    $response["success"] = true;  
  
echo json_encode($response);  
?>
```

Das Passwort kann auch im Webinterface geändert werden. Das Webinterface benutzt die gleichen Scripts, aber ohne Response. Sonst ist der Rest HTML/CSS und ein bisschen JavaScript für die Validierung.



## 14 Checklisten

Mit der Checkliste kann man beim Ämtli machen sicherstellen, dass man nichts vergisst.

### 14.1 XML-Code

Zuerst müssen wir die Checkliste gestalten. Wir haben im visuellen Editor Checkbox-Element eingefügt. Dieser Code muss dann entstanden sein.

```
<CheckBox
    android:id="@+id/ablagefach"
    android:layout_width="340dp"
    android:layout_height="50dp"
    android:layout_marginBottom="8dp"
    android:text="Ablagefach (BACK TO LE)"
    android:textSize="18sp"
    app:layout_constraintBottom_toTopOf="@+id/paketdienst"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.492"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="1.0" />
```

Im XML haben wir die ID's und den String, deren angezeigt werden soll definiert.

### 14.2 Java-Code (Werte speichern)

Diese Funktion ist nötig, damit unsere Checkliste nicht alles wieder vergisst. Das passiert, wenn man die App schliesst oder das Gerät die App aus dem RAM löscht. Um die Dateien zu speichern, müssen wir SharedPreferences benutzen.

Wir schreiben den folgenden Code in die Datei checklisteActivity.java und in die onCreate-Methode.

Zuerst erstellen wir Variablen und teilen ihnen Werte zu:

```
final CheckBox chk1;
// ...
chk1 = findViewById(R.id.ablagefach);
// ...
```

Das müssen wir für jedes Kontrollkästchen machen. Die Namen der Variablen dürfen nicht mehrmals vorhanden sein + Man muss die ID pro Kästchen angeben.

Dann müssen wir die SharedPreferences erstellen. Dafür müssen wir folgenden Befehl schreiben. Wir haben den Namen «spCheckliste» vergeben.

```
SharedPreferences spCheckliste = getSharedPreferences("save", MODE_PRIVATE);
// ...
chk1.setChecked(spCheckliste.getBoolean("ablagefach", false));
// ..
```

Den unteren Befehl muss man für jedes Kontrollkästchen wiederholen.

Hier muss man den Namen des Wertes (ablagefach...) pro Kästchen ändern, sonst überschreiben sich die Werte gegenseitig.

Bis jetzt haben wir es geschafft, dass die App schaut, ob ein Wert abgespeichert worden ist. Wenn ja, lädt es diesen Wert und zeigt es im Kontrollkästchen an (angekreuzt/nicht angekreuzt.)

Jetzt wird aber immer angezeigt, dass es das Kontrollkästchen nicht angekreuzt ist. Damit der richtige Wert ausgegeben wird, müssen wir noch ein bisschen weiter Code schreiben.

```
chk1.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View v) {  
        if (chk1.isChecked()) {  
            SharedPreferences.Editor editor = getSharedPreferences("save",  
MODE_PRIVATE).edit();  
            editor.putBoolean("ablagefach", true);  
editor.putLong("loeschennach", System.currentTimeMillis() +  
TimeUnit.MINUTES.toMillis(720));  
  
            editor.apply();  
            chk1.setChecked(true);  
        } else {  
            SharedPreferences.Editor editor = getSharedPreferences("save",  
MODE_PRIVATE).edit();  
            editor.putBoolean("ablagefach", false);  
editor.putLong("loeschennach", System.currentTimeMillis() +  
TimeUnit.MINUTES.toMillis(720));  
  
            editor.apply();  
            chk1.setChecked(false);  
        }  
    }  
});
```

Hier wird eine If-Else Anweisung gebraucht. Sie wird ausgelöst, wenn das Kontrollkästchen angetippt wird. Alles was dieser Code-Snippet ausführt, ist:

Wenn das Kontrollkästchen angekreuzt ist, sollte der Boolean-Wert von «ablagefach» auf «true» gesetzt werden. Ausserdem sollte das Kontrollkästchen auf den «true»-State gewechselt werden.

Falls nicht, wird der Boolean-Wert «ablagefach» auf «false» gesetzt, und das Kontrollkästchen wird auf den «false»-State gewechselt

In beiden Fälle wird eine maximale Gültigkeit von 12 Stunden angegeben

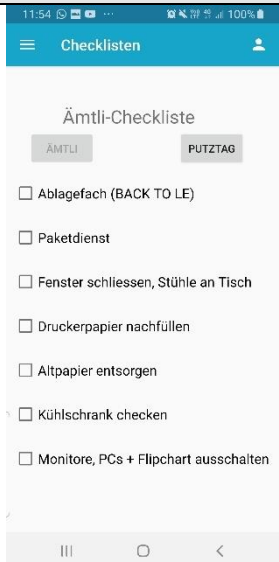
Das muss man noch für jedes Kontrollkästchen wiederholen.

```
if (!(spCheckliste.getLong("loeschennach", -1) > System.currentTimeMillis()))
{
    SharedPreferences.Editor editor = spCheckliste.edit();
    editor.clear();
    editor.apply();
}
```

Am Schluss muss man noch angeben, dass die Werte gelöscht werden, falls sie älter als 12 Stunden sind.

## 14.3 Zweite Checkliste

Wir haben auch noch eine zweite Checkliste auf der noch selben Seite erstellt. Es kam das Bedürfnis nach einer Putztag-Checkliste.

Erklärung	Bild
Zuerst haben wir über die Ämtli-Checkliste, die Putztag-Checkliste erstellt und die Putztag standardmässig ausgeblendet.	 <p>Abbildung 64 Checklisten</p>
In der Java Datei haben wir die die Checkboxes also Kriterien in eine Variabel deklariert.	<pre>final View view = inflater.inflate(R.layout.fragment_checkliste, container, false); final TextView titelchk; final CheckBox chk1; final CheckBox chk2; final CheckBox chk3; final CheckBox chk4; final CheckBox chk5; final CheckBox chk6; final CheckBox chk7; final CheckBox chk8; final CheckBox chk9; final CheckBox chk10; final CheckBox chk11; final CheckBox chk12; final CheckBox chk13; final CheckBox chk14; final CheckBox chk15; super.onCreate(savedInstanceState);</pre> <p>Abbildung 65 Variabel Deklaration</p>

Erklärung	Bild
<p>Danach haben wir unsere Variabel (chk1) einem Kriterium zugewiesen. Wir haben also jeder Variabel eine Checkbox zugewiesen.</p> <p>Unten haben wir noch die beiden Buttons für Ämtli und Putztag deklariert.</p>	<pre> titelchk = view.findViewById(R.id.text_checkliste); chk1 = view.findViewById(R.id.ablagefach); chk2 = view.findViewById(R.id.paketsdienst); chk3 = view.findViewById(R.id.fenster); chk4 = view.findViewById(R.id.drucker); chk5 = view.findViewById(R.id.altpapier); chk6 = view.findViewById(R.id.kuehl); chk7 = view.findViewById(R.id.monitor); chk8 = view.findViewById(R.id.aeschen); chk9 = view.findViewById(R.id.abfall); chk10 = view.findViewById(R.id.tische); chk11 = view.findViewById(R.id.baenke); chk12 = view.findViewById(R.id.boden); chk13 = view.findViewById(R.id.vollabfall); chk14 = view.findViewById(R.id.mikro); final Button button3 = (Button) view.findViewById(R.id.button3); final Button button2 = (Button) view.findViewById(R.id.button2); </pre> <p>Abbildung 66 Zuweisung der Buttons</p>
<p>Da man standardmässig zuerst auf die Ämtli-Checkliste kommt haben wir den Button der die Ämtli-Checkliste öffnet disabled, damit man weiss das man sich auf dieser Seite befindet.</p> <p>Danach noch alle Kriterien der Putztag-Checkliste versteckt damit man nur die Ämtli-Checkliste sieht.</p>	<pre> button3.setEnabled(true); button2.setEnabled(false); titelchk.setText(R.string.checkliste_text); chk8.setVisibility(View.GONE); chk9.setVisibility(View.GONE); chk10.setVisibility(View.GONE); chk11.setVisibility(View.GONE); chk12.setVisibility(View.GONE); chk13.setVisibility(View.GONE); chk14.setVisibility(View.GONE); </pre> <p>Abbildung 67 Checkliste verstecken</p>

Erklärung	Bild
<p>Als nächsten Schritt haben wir den Button konfiguriert. Mit einem OnClickListener haben wir dann programmiert, dass die Ämtli-Checkliste verschwindet und die Putztag-Checkliste erscheint und umgekehrt. Zudem sollte der Button der aktiven Seite disabled werden und der andere Button enabled. Auch das der Titel der Seite wechselt wenn man denn Button klickt haben wir programmiert.</p>	<div data-bbox="767 271 1262 763"> <pre>button3.setOnClickListener(new View.OnClickListener() {     @Override     public void onClick(View v) {         titelchk.setText(R.string.checkliste_text_putz);         button3.setEnabled(false);         button2.setEnabled(true);         chk1.setVisibility(View.GONE);         chk2.setVisibility(View.GONE);         chk3.setVisibility(View.GONE);         chk4.setVisibility(View.GONE);         chk5.setVisibility(View.GONE);         chk6.setVisibility(View.GONE);         chk7.setVisibility(View.GONE);         chk8.setVisibility(View.VISIBLE);         chk9.setVisibility(View.VISIBLE);         chk10.setVisibility(View.VISIBLE);         chk11.setVisibility(View.VISIBLE);         chk12.setVisibility(View.VISIBLE);         chk13.setVisibility(View.VISIBLE);         chk14.setVisibility(View.VISIBLE);     } });</pre> <p>Abbildung 68 Button-Konfiguration</p> </div> <div data-bbox="767 819 1275 1341"> <pre>button2.setOnClickListener(new View.OnClickListener() {     @Override     public void onClick(View v) {         titelchk.setText(R.string.checkliste_text);         button3.setEnabled(true);         button2.setEnabled(false);         chk1.setVisibility(View.VISIBLE);         chk2.setVisibility(View.VISIBLE);         chk3.setVisibility(View.VISIBLE);         chk4.setVisibility(View.VISIBLE);         chk5.setVisibility(View.VISIBLE);         chk6.setVisibility(View.VISIBLE);         chk7.setVisibility(View.VISIBLE);         chk8.setVisibility(View.GONE);         chk9.setVisibility(View.GONE);         chk10.setVisibility(View.GONE);         chk11.setVisibility(View.GONE);         chk12.setVisibility(View.GONE);         chk13.setVisibility(View.GONE);         chk14.setVisibility(View.GONE);     } });</pre> <p>Abbildung 69 Button-Konfiguration</p> </div>
<p>Am Ende hat das Fragment so ausgesehen. Um den Inhalt zu speichern musste man wieder dasselbe wie vorher bei Kapitel 7.2 machen.</p>	<div data-bbox="778 1400 1007 1861"> <p>Abbildung 70 Checklisten</p> </div>

Tabelle 14 Zweite Checkliste

## 15 Notenformular

### 15.1 Konzept

Das Notenformular muss so einfach wie möglich eingerichtet sein, damit es auch Laien verstehen können. Es soll möglich sein, den Lehrlingen den Link von Google Spreadsheets zuzuweisen, ohne in phpMyAdmin einzuloggen und den Code der App zu verändern.

Die erste Möglichkeit, worüber ich nachgedacht habe, war: Ein Webinterface erstellen, wo die Berufsbildner den Lehrlingen den Link des G-Spreadsheets zuweisen können und eine Seite in der App, wo die Lehrlinge ihr Noten abschicken konnten. Die Noten werden auf Google Spreadsheet und die Zuweisung des G-Spreadsheets in der SQL-Datenbank gespeichert. Das alles wird mit PHP, HTML&CSS (Webinterface), JavaScript (Validierung Webinterface), Java (Android Studio) und SQL (Datenbank) gelöst.

<https://github.com/dwyl/learn-to-send-email-via-google-script-html-no-server>

### 15.2 Design

Da das bisherige Formular schon bereits optimal war, haben wir genau die gleichen Felder übernommen. Das Design sieht so aus:

Vorgehensweise

1. Bei Erhalt einer Note ist diese im Formular einzutragen.
2. Wenn die Note ungenügend ist, ist eine Begründung anzugeben, weshalb die Note so ausgefallen ist.
3. Auf den Button Absenden klicken. → E-Mail an Berufsbildner wird ausgelöst.

Fach\*

Thema Prüfung\*

☐ Note-E ☐ Note-A

Note\*

Begründung <4.0

ABSENDEN

\* → erforderlich

Note-E: Noten die effektiv zu 100% gelten

Note-A: Noten die bei Kurztests oder zum Aufrunden benötigt werden

NOTENFORMULAR

Abbildung 71 Notenformular

Für das Dropdown-Menü mussten wir Code schreiben:

```
String [] values =  
    {"Bitte auswählen*", "Deutsch", "Englisch", "Französisch",
```

```
"Geschichte & Politik", "Gesellschaft IDAF", "INF-Schulmodule", "Mathematik  
Grundlagen", "Mathematik Schwerpunkt", "Naturwissenschaften NW I - Physik",  
"NW II - Chemie", "Sportunterricht", "Sprache & Kommunikation", "Wirtschaft &  
Recht"};  
Spinner spinner = (Spinner) root.findViewById(R.id.fachSend);  
ArrayAdapter<String> adapter = new ArrayAdapter<String>(this.getActivity(),  
android.R.layout.simple_spinner_item, values);  
adapter.setDropDownViewResource(android.R.layout.simple_dropdown_item_1line);  
spinner.setAdapter(adapter);
```

Wir hatten zuerst ein Array mit den Werten der Dropdown-Liste erstellt. Dann haben wir ein Spinner-Element erstellt, der auf das Spinner-Element aus dem XML zugreift. Dann haben wir das Dropdown-Menü erstellt und ihm die Werte aus dem Array zugewiesen.

### 15.3 Validation

Dann hatten wir die Validierung programmiert. Diese Validierung habe ich auch dann auf das Login angewendet. Wir benutzten für Textfelder AwesomeValidation und für andere Felder die Android-eigene Validation.

So sieht ein mit AwesomeValidation validiertes Feld aus:

```
awesomeValidation = new AwesomeValidation(BASIC);  
awesomeValidation.addValidation(editDatum, "([0-9]{2}\\.|){0,2}([0-9]{4})",  
getString(R.string.dateerror));
```

Zuerst erstellen wir ein AwesomeValidation Element. Dann benutzen wir dieses Element, um einem Element die Validation hinzuzufügen. Der erste Wert in den Klammern (editDatum) ist das ausgewählte Textfeld, die kryptischen sind RegEx Begriffe (Das zu erklären würde den Rahmen sprengen, aber was es macht ist eine Regel, wie etwas eingegeben werden darf. Hier ist es für ein europäisches Datum (dd.mm.yyyy.) Nachher geben wir die Fehlermeldung an. Hier nehmen wir es aus der strings.XML Datei.

```
public void onClick(View root) {  
    if (dropFach.getSelectedItemAt().toString().trim().equals("Bitte  
auswählen*")) {  
        errorSpinner.setError(getString(R.string.facherror));  
    } else {  
        errorSpinner.setError(null);  
    }  
}
```

Da der Spinner keine Fehlermeldungen unterstützt, mussten wir ein Textfeld über den Spinner platzieren, der die Fehlermeldung ausgibt.

Mit der if else Anweisung stellten wir ein, dass wenn «Bitte auswählen\*» ausgewählt war, eine Fehlermeldung ausgegeben wurde. Wenn nicht, wurde die Meldung annulliert. Mit dem nächsten Befehl wird die Validation ausgeführt.

```
awesomeValidation.validate();
```

## 15.4 User Berechtigungen

Wir mussten Sicherheitsfeatures einbauen, sodass nicht irgendwer, der nicht zum BL gehört, sich einloggen kann und Noten eintragen kann. Das verhindern wir mit einer Datenbank. In dieser Datenbank sind die Spreadsheet-Links und die dazugehörigen Links abgelegt. Die App prüft nach, ob die E-Mail des eingeloggten Users in der Datenbank vorhanden ist. Wenn ja, wird das Formular des Benutzers geladen. Wenn nicht, verschwindet das Formular und eine Fehlermeldung wird angezeigt. Dafür mussten wir zuerst ein PHP-File erstellen, mit der die App kommuniziert. Das PHP-Skript schaut, ob die empfangene E-Mail-Adresse in der Datenbank existiert und wenn ja schickt sie den dazugehörigen Spreadsheet-Link + Script-Link wieder zurück zu der App.

```
<?PHP
error_reporting(0);

$con = mysqli_connect('blnetch.mysql.db.internal', 'blnetch_app',
'XXXXXXX', 'blnetch_app');
$getemail = mysqli_real_escape_string($con, $_REQUEST["email"]);
$stmt = mysqli_prepare($con, "SELECT * FROM emailsheet WHERE email =
?");
mysqli_stmt_bind_param($stmt, "s", $getemail);
mysqli_stmt_execute($stmt);
mysqli_stmt_store_result($stmt);
mysqli_stmt_bind_result($stmt, $colemail, $colsheetlink,
$colscriptlink);
$response = array();
while(mysqli_stmt_fetch($stmt)){
    if ($getemail == "") {
        $response["success"] = false;
    } else {
        $response["success"] = true;
        $response["email"] = $colemail;
        $response["sheetlink"] = $colsheetlink;
        $response["scriptlink"] = $colscriptlink;
    }
}
echo json_encode($response);
?>
<title>blnet App Notenformular Authentification</title>
```

Wenn nicht, schickt sie «success=false» zurück, was die Fehlermeldung auslöst.



In der App selbst mussten wir coden, was die App mit den Antworten vom Server machen sollte. Weil der Server die Antwort als JSON-File sendet, müssen wir einen JSONResponse Element erstellen. Dann haben wir die Antworten aus dem JSON entnommen und in eine Variable eingesetzt. Die Werte werden dann direkt in SharedPreferences gespeichert. Try und catch ist nötig für JSONResponse. In try kommt die Funktion, in catch kommt Code, der ausgeführt wird, wenn die JSON-Datei fehlerhaft ist oder fehlt.

```
Response.Listener<String> responseListener = new Response.Listener<String>()
{
    @Override
    public void onResponse(String response) {
        try {
            JSONObject jsonResponse = new JSONObject(response);
            boolean success = jsonResponse.getBoolean("success");
            final String tempEmailjson = jsonResponse.getString("email");
            final String sheetlinkjson = jsonResponse.getString("sheetlink");
            SharedPreferences.Editor editor =
getActivity().getSharedPreferences("noteform", MODE_PRIVATE).edit();
            editor.putString("tempMail", tempEmailjson);
            editor.apply();
            editor.putString("sheetlink", sheetlinkjson);
            editor.apply();
            if (success == true) {
                titleSend.setText(getString(R.string.concatnoteformtitle) + "
" + nachname);
            } else if (success == false) {
                titleSend.setText(getString(R.string.norights));
                editDatum.setVisibility(View.GONE);
                // usw...
            }
        } catch (JSONException e) {
            e.printStackTrace();
            titleSend.setText(getString(R.string.jsonexceptsend));
            editDatum.setVisibility(View.GONE);
            // usw...
        }
    }
};
```

## 15.5 SQL-Tabelle

Die Links werden in der Datenbank gespeichert. Dafür haben wir eine einfache Tabelle mit den Spalten email, sheetlink & scriptlink erstellt. Für alle drei haben wir den Datentyp varchar benutzt.

#	Name	Typ	Kollation	Attribute	Null	Standard	Kommentare	Extra	Aktion
<input type="checkbox"/> 1	email	varchar(100)	utf8_unicode_ci		Nein	kein(e)			Bearbeiten  Löschen  Mehr
<input type="checkbox"/> 2	sheetlink	varchar(100)	utf8_unicode_ci		Nein	kein(e)			Bearbeiten  Löschen  Mehr
<input type="checkbox"/> 3	scriptlink	varchar(500)	utf8_unicode_ci		Nein	kein(e)			Bearbeiten  Löschen  Mehr

## 15.6 Absende-Request

Jetzt wird der Absende-Request programmiert. Das brauchen wir, damit das Formular abgeschickt werden kann. Das Formular funktioniert wie ein stinknormales HTML-Formular, mit dem Unterschied, dass wir die Ergebnisse nicht zu einem PHP-File schicken, sondern direkt an Google Spreadsheets. Da das Formular von der blnet Seite auch so funktioniert, haben wir uns entschieden, das auch zu benutzen. Dann haben wir alle Noten zentral an einem Ort.

```
private void submitForm() {
    if (awesomeValidation.validate()) {
        // Notenformular abschicken
        loadingButton.setVisibility(View.VISIBLE);

        getActivity().getWindow().setFlags(WindowManager.LayoutParams.FLAG_NOT_TOUCHABLE,
            WindowManager.LayoutParams.FLAG_NOT_TOUCHABLE);
        final String date = editDatum.getText().toString();
        final String subject = dropFach.getSelectedItemAt().toString();
        final String thema = editThema.getText().toString();
        final int notetypepre = radioType.getCheckedRadioButtonId();
        RadioButton notetypeSelected = (RadioButton)
        getActivity().findViewById(notetypepre);
        final String notetype = notetypeSelected.getText().toString();
        final String note = editNote.getText().toString();
        final String begr = begrSend.getText().toString();
        SharedPreferences spNote =
        getActivity().getSharedPreferences("noteform", MODE_PRIVATE);
        final String scriptlink = spNote.getString("scriptlink", null);
        RequestQueue queue = Volley.newRequestQueue(getContext());
        StringRequest stringRequest = new StringRequest(Request.Method.POST,
        scriptlink, new Response.Listener<String>() {
            @Override
            public void onResponse(String response) {
                loadingButton.setVisibility(View.GONE);

                getActivity().getWindow().clearFlags(WindowManager.LayoutParams.FLAG_NOT_TOUCHABLE);

                CharSequence toasttext = getString(R.string.successsend);
                int duration = Toast.LENGTH_LONG;
                Toast successinfo = Toast.makeText(getContext(), toasttext,
                duration);
                successinfo.show();
            }
        });
    }
}
```

```
        Intent backToMain = new Intent(getActivity(),
MainActivity.class);
        startActivity(backToMain);
    }
}, new Response.ErrorListener() {
    @Override
    public void onErrorResponse(VolleyError error) {
        loadingButton.setVisibility(View.GONE);

getActivity().getWindow().clearFlags(WindowManager.LayoutParams.FLAG_NOT_TOUCHABLE);

        AlertDialog.Builder builder = new
AlertDialog.Builder(getActivity());
        builder.setTitle(getString(R.string.error))
            .setMessage(getString(R.string.notefail))
            .setNegativeButton(getString(R.string.tryagainlogin),
null)

            .create()
            .show();
    }
}) {
    protected Map<String, String> getParams() {
        Map<String, String> params = new HashMap<String, String>();
        params.put("action", "addItem");
        params.put("datum", date);
        params.put("fach", subject);
        params.put("themaPruefung", thema);
        params.put("notenTyp", notetype);
        params.put("note", note);
        params.put("begruendung", begr);
        return params;
    }

    @Override
    public Map<String, String> getHeaders() throws AuthFailureError {
        Map<String, String> params = new HashMap<String, String>();
        params.put("Content-Type", "application/x-www-form-
urlencoded");
        return params;
    }
};
queue.add(stringRequest);
}
```

Das ist ganz viel Code, aber die Funktion ist simpel. Der Code in der Funktion submitForm wird ausgeführt, wenn auf den «Absenden» geklickt wird und die Validation erfolgreich ist. Mit if (awesomeValidation.validate()) stellen wir sicher, ob alles validiert worden ist. Als nächstes deaktivieren wir alle Eingaben und blenden einen Ladebalken ein. Dann entnehmen wird von allen Eingabefeldern den eingegebenen Wert. Aus den SharedPreferences haben wir den Link genommen, an den die Daten geschickt werden. Jetzt erstellen wir ein StringRequest Element, wo wir auch den Link aus den SharedPreferences als Ziellink festlegen. Darin erstellen wir auch noch einen ResponseListener (wartet

auf eine Antwort vom Server). Die Funktion onResponse wird ausgeführt, wenn der Server eine Antwort gibt. In unserem Fall wird als erstes die Eingabebeschränkung wieder aufgehoben und der Ladebalken verschwindet wieder. Dann wird eine Toast-Nachricht erstellt und der User wird zu der Startseite weitergeleitet.

In der Funktion onErrorResponse wird deklariert, was geschehen soll, falls der Server keine/eine fehlerhafte Antwort gibt. Bei uns wird die Eingabebeschränkung wieder aufgehoben und der Ladebalken verschwindet. Ausserdem wird eine Meldung erzeugt, die den User aufklärt, dass ein Fehler passiert ist.

Als nächstes geben wir die Parameter ein, die zu Google Spreadsheets geschickt werden sollen. Als erstes wird der Name-Attribut eingegeben (Wie aus HTML), und dann der Wert, der damit verschickt werden soll. Hier ist dieser Wert eine Variable vom Eingabefeld.

Der Code ähnelt leicht an den vom Login, da beide mit Volley programmiert worden sind. Volley eine von Google entwickelte Bibliothek, um HTTP/HTTPS An/Abfragen zu verarbeiten.

## 15.7 App Script (Google Scripts)

Serverseitig müssen wir auch ein kleines Skript haben, um die empfangenen Daten von der App in die Tabellenzeilen einfügen zu können. Dafür haben wir Google Scripts benötigt.

```
var ss =
SpreadsheetApp.openByUrl("https://docs.google.com/spreadsheets/d/1xLH5-
WE45NVLwiGiDHNE42KDd4qvqIbkgz3vNg9wy5k/");
var sheet = ss.getSheetByName('Formularantworten'); // be very careful
... it is the sheet name .. so it should match

function doPost(e){
var action = e.parameter.action;

if(action == 'addItem'){
    return addItem(e);
}
}

function addItem(e){
var date = new Date();
var datefromuser = e.parameter.datum;
var fach = e.parameter.fach;
var thema = e.parameter.themaPruefung;
var notentyp = e.parameter.notenTyp;
```

```
var note = e.parameter.note;
var begruendung = e.parameter.begruendung;
sheet.appendRow([date,datefromuser,fach,thema,notentyp,note,begruendung])
;
    return
ContentService.createTextOutput("Success").setMimeType(ContentService.Mim
eType.TEXT);
}
```

Dieses Skript nimmt die Daten, die über POST empfangen wurden, macht sie zu einer Variable, und fügt sie in eine Tabelle ein.

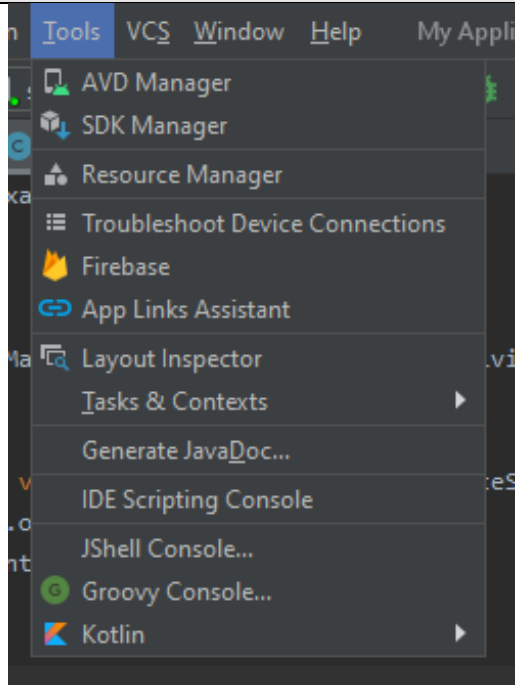
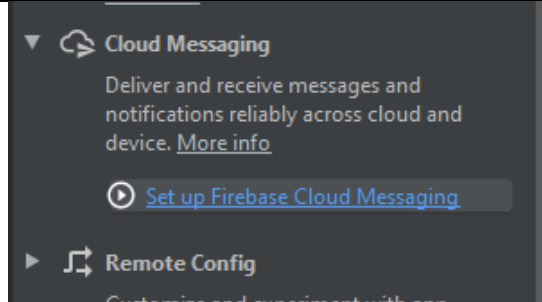
In der Variable ss gehört der Link zu der Spreadsheets-Tabelle.

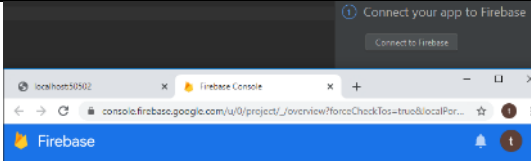
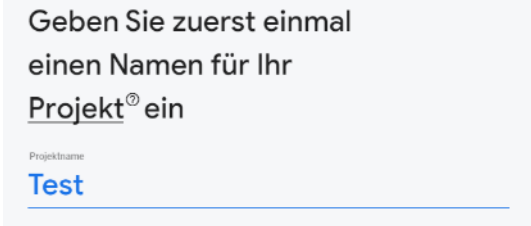
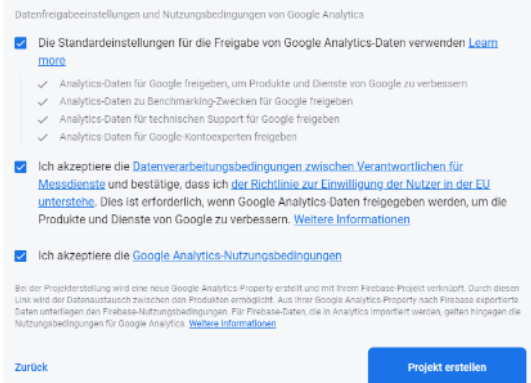

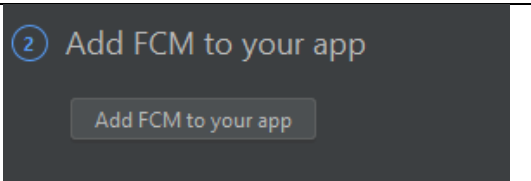
## 16 Firebase

### 16.1 Was ist Firebase?

Google ist eine Plattform (von Google), wo man Applikationen, ob Android, iOS und ganz normal im Web, weiterentwickeln kann. Firebase bietet gewisse Tools an, welche Entwickler verwenden können um hochwertige Apps zu entwickeln. Wir brauchen diese Firebase für Push Notifications. Firebase stellt viele Features dar, so dass Entwickler mit ihrem Unternehmen Geld verdienen und sich auf ihre User konzentrieren können.

### 16.2 Firebase mit Android Studio verbinden

Erklärung	Bild
Um Firebase einzubinden muss man in Android Studio unter Tools auf Firebase. Danach wird auf der rechten Seite im Android Studio ein kleines Fenster aufgeschlagen.	 <p>Abbildung 72 Android Studios Tools</p>
In diesem Fenster kann man dann wählen für was man Firebase brauchen will. In unserem Fall für Benachrichtigungen also Cloud Messaging.	 <p>Abbildung 73 Cloud Messaging Android Studio</p>

Erklärung	Bild
Wenn man dann "Set up Firebase Cloud Messaging" öffnet sich im Browser automatisch die Startseite von Firebase.	 <p>Abbildung 74 Firebase im Browser</p>
Auf der Firebase Website muss man dann auf Projekt erstellen und den Projektnamen eingeben.	 <p>Abbildung 75 Firebase Projektname</p>
Um die Verbindung abzuschliessen einfach Weiter klicken und am Ende die Lizenzbedingungen lesen und akzeptieren.	 <p>Abbildung 76 Firebase Lizenzbedingungen</p>
Falls kein Fehler auftritt sollte dieses Fenster auftauchen wo man einfach verbinden klicken muss.	 <p>Abbildung 77 Android Studio und Firebase verbinden</p>
Um Firebase Cloud Messaging hinzuzufügen muss man noch den zweiten Schritt machen.	 <p>Abbildung 78 FCM einfügen</p>

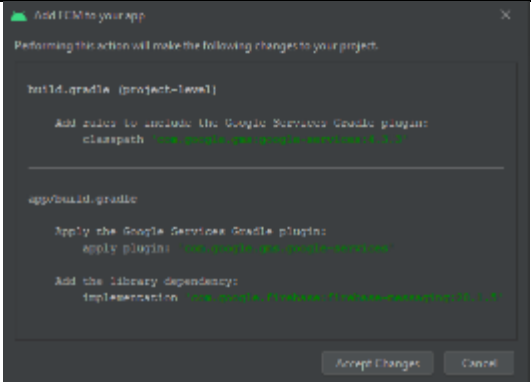
Erklärung	Bild
Hier muss man wesentlich weniger machen und einfach "Accept Changes" klicken.	 <p>Abbildung 79 FCM akzeptieren</p>

Tabelle 15 Firebase verbinden



## 16.3 Cloud Messaging

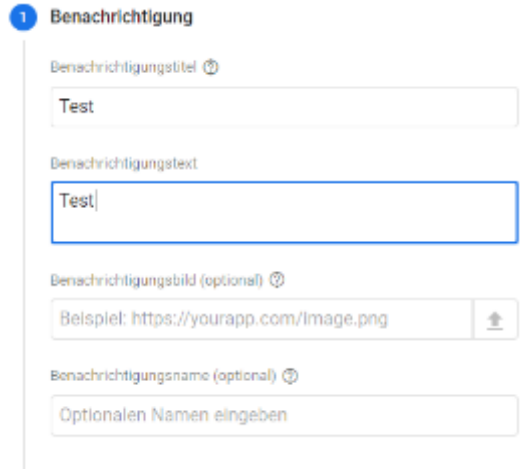
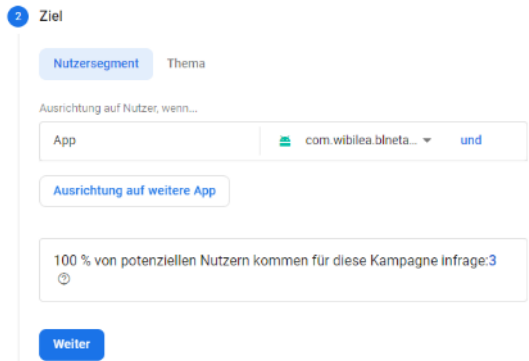
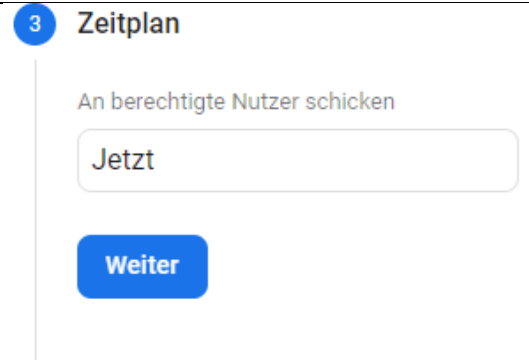
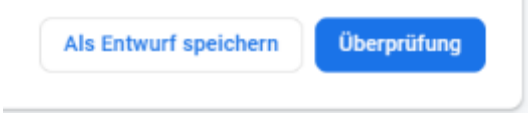
Erklärung	Bild
Wenn man bei Firebase sein neues Projekt erstellt hat und mit Android Studio verbunden hat gibt es eine Kategorie Cloud Messaging. Geht man auf diese Kategorie kann man eine Neue Benachrichtigung erstellen (Ist es die erste Nachricht steh Mache deine erste Benachrichtigung). Click man auf Neue Benachrichtigung kommt ein Fenster wo man die Nachricht konfigurieren kann. Im Benachrichtigungstitel steht dann meistens der Sender und im Benachrichtigungstext die eigentliche Information	 <p>Abbildung 80 Benachrichtigung erstellen</p>
Im nächsten Schritt kann man angeben zu welcher Applikation man senden will. Diese Applikation wird durch die ApplicationID identifiziert. Diese ID ist auch im gradle.script(Module App)-File zu finden.	 <p>Abbildung 81 Ziel Definition</p>
Beim dritten Schritt kann man bestimmen wann die Nachricht denn abgesendet wird. Für die Ämtlis und dem Putzdienst haben wir beim Zeitplan eine Benutzerdefinierte Variante genommen. Mit Benutzerdefinierten Zeitplänen kann man Wöchentlich an ausgewählten tagen eine Benachrichtigung konfigurieren. Eine Nachricht wird dann zum Beispiel Jeden Montag um 06:00 Uhr automatisch gesendet.	 <p>Abbildung 82 Zeitplan Firebase</p>
Die restlichen Punkte können übersprungen werden da sie irrelevant für die Nachricht sind.	 <p>Abbildung 83 Nachricht Überprüfung</p>

Tabelle 16 Cloud Messaging

## 16.4 Spezifizierte Benachrichtigungen

### 16.4.1 User Properties

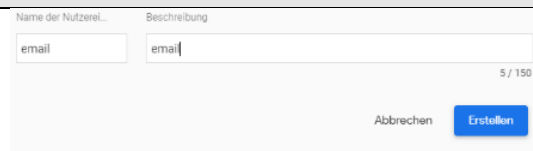
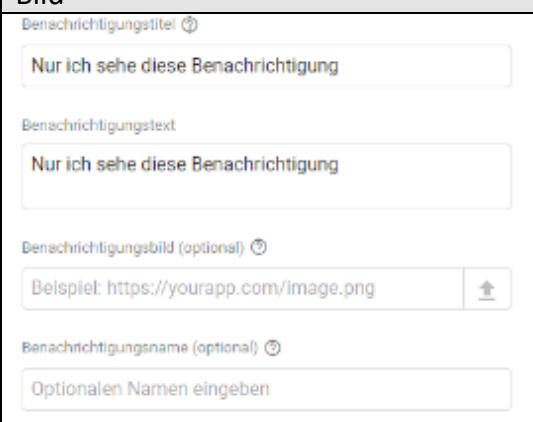
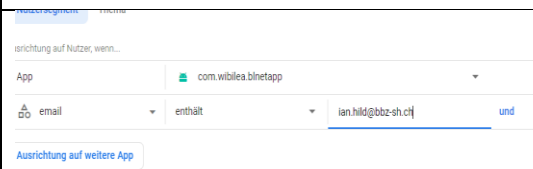
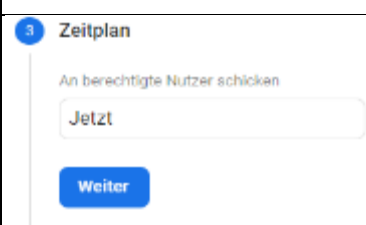
Erklärung	Bild
Eine User Property wird ganz leicht erstellt. Einfach unter User Properties eine neue Nutzereigenschaft erstellen.	 <p>Abbildung 84 Nutzereigenschaft</p>

Tabelle 17 User Properties

### 16.4.2 Spezifizierte Nachricht

Erklärung	Bild
Um einem bestimmten Benutzer eine Nachricht zu machen muss man nicht viel anders machen wie bei der normalen Benachrichtigung für alle. Man schreibt zu Beginn einfach den Titel und Text der Benachrichtigung.	 <p>Abbildung 85 Benachrichtigung Text</p>
Hier wählt man aber nicht nur die ApplicationID, sondern «Und» klicken. Dort muss man dann «Nutzereigenschaften» auswählen und zudem die gewünschte User Property. Wenn man die Property E-Mail genommen hat muss man noch zudem die entsprechende E-Mail-Adresse des Empfängers eingeben	 <p>Abbildung 86 Ziel-E-Mail Angabe</p>
Hier kann man wieder wie vorhin einen eigenen Zeitplan machen oder einfach auf "Jetzt" lassen.	 <p>Abbildung 87 Zeitplan Firebase</p>

Erklärung	Bild
Nachdem die Benachrichtigung abgeschickt wurde wird sie angezeigt. Diese Nachricht bekommt also nur der User mit dieser E-Mail.	 <p>The screenshot shows a smartphone's notification shade pulled down. At the top, the status bar shows the time 09:56, date Di., 28. April, and battery level at 74%. Below the status bar are icons for Wi-Fi, cellular data, Bluetooth, and airplane mode. The notification shade contains three items: a 'blinet' notification from 09:56 with the text 'Nur ich sehe diese Benachrichtigung' repeated twice; a 'Software-Update' notification stating 'Update bereit für die Installation'; and a 'Teams' notification from 19:35 from 'Abeeraam Rahunenthiran (Gast)' with the text 'Zur Rückkehr zum Anruf tippen'.</p>

Abbildung 88 Benachrichtigung auf dem Handy

Tabelle 18 Spezifizierte Benachrichtigung

## 17 Word Press Rest API

### 17.1 Was ist WP Rest API?

WP Rest API ist eine Schnittstelle, um Wordpressinhalte ausserhalb der Seite in Apps reinzuladen. Dabei steht Rest für Representational State Transfer und API für Application Programming Interface.

In Wordpress muss man noch zwei Plugins installieren um WP Rest API überhaupt zu ermöglichen.



WordPress REST API (Version 2)	REST API – Filter Fields
 <p><b>WordPress REST API (Version 2)</b></p> <p><a href="#">Aktiv</a> <a href="#">Weitere Details</a></p> <p>Access your site's data through an easy-to-use HTTP REST API. (Version 2)</p> <p>Von <i>WP REST API Team</i></p> <p>★★★★☆ (34) <span style="float: right;">Zuletzt aktualisiert: vor 4 Jahren</span></p> <p>30'000+ aktive Installationen <span style="float: right;">Ungetestet mit Ihrer WordPress-Version</span></p> <p><i>Abbildung 89 WordPress REST API (Version 2)</i></p>	 <p><b>REST API – Filter Fields</b></p> <p><a href="#">Aktiv</a> <a href="#">Weitere Details</a></p> <p>Filter the properties returned by the Wordpress rest api V2</p> <p>Von <i>Stephan van Rooij</i></p> <p>★★★★★ (10) <span style="float: right;">Zuletzt aktualisiert: vor 3 Jahren</span></p> <p>2'000+ aktive Installationen <span style="float: right;">Ungetestet mit Ihrer WordPress-Version</span></p> <p><i>Abbildung 90 REST API – Filter Fields</i></p>

Tabelle 19 Wordpress Rest API

## 17.2 Versuche

### 17.2.1 1.Versuch YouTube Tutorial

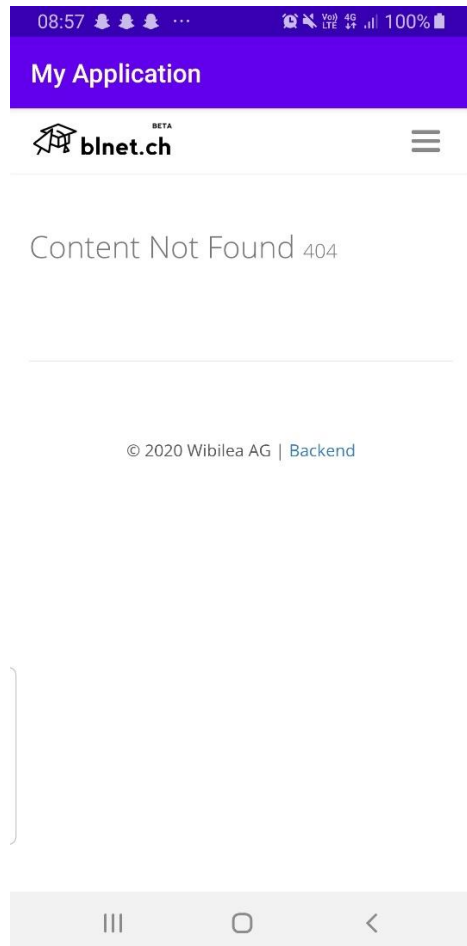
Beim erste Versuch probierten wir mit einem YouTube [Tutorial](#) über WP Rest API in Android Studio. Das Tutorial scheiterte nach mehreren Build-Versuchen. Das Video ging recht lange und vieles konnte nicht erklärt werden im Video. Ich habe viel abgeschrieben und icht viel mitgenommen

```
Caused by: android.view.InflateException: Binary XML file line #13: Error inflating class android.support.v7.widget.RecyclerView
Caused by: java.lang.ClassNotFoundException: Didn't find class "android.support.v7.widget.RecyclerView" on path: DexPathList[[zip file "/data
at dalvik.system.BaseDexClassLoader.findClass(BaseDexClassLoader.java:134)
at java.lang.ClassLoader.loadClass(ClassLoader.java:379)
at java.lang.ClassLoader.loadClass(ClassLoader.java:312)
at android.view.LayoutInflater.createView(LayoutInflater.java:606)
at android.view.LayoutInflater.createViewFromTag(LayoutInflater.java:790)
at android.view.LayoutInflater.createViewFromTag(LayoutInflater.java:730)
at android.view.LayoutInflater.inflate(LayoutInflater.java:863)
at android.view.LayoutInflater.inflateChildren(LayoutInflater.java:824)
at android.view.LayoutInflater.inflate(LayoutInflater.java:866)
at android.view.LayoutInflater.inflateChildren(LayoutInflater.java:824)
at android.view.LayoutInflater.inflate(LayoutInflater.java:515)
at android.view.LayoutInflater.inflate(LayoutInflater.java:423)
at android.view.LayoutInflater.inflate(LayoutInflater.java:374)
at androidx.appcompat.app.AppCompatActivityDelegateImpl.setContentView(AppCompatActivityDelegateImpl.java:555)
at androidx.appcompat.app.AppCompatActivity.setContentView(AppCompatActivity.java:161)
at com.example.myapplication.MainActivity.onCreate(MainActivity.java:40)
at android.app.Activity.performCreate(Activity.java:7327)
at android.app.Activity.performCreate(Activity.java:7318)
at android.app.Instrumentation.callActivityOnCreate(Instrumentation.java:1271)
at android.app.ActivityThread.performLaunchActivity(ActivityThread.java:3094)
at android.app.ActivityThread.handleLaunchActivity(ActivityThread.java:3257)
at android.app.servertransaction.LaunchActivityItem.execute(LaunchActivityItem.java:78)
at android.app.servertransaction.TransactionExecutor.executeCallbacks(TransactionExecutor.java:108)
at android.app.servertransaction.TransactionExecutor.execute(TransactionExecutor.java:68)
at android.app.ActivityThread$H.handleMessage(ActivityThread.java:1948)
at android.os.Handler.dispatchMessage(Handler.java:106)
at android.os.Looper.loop(Looper.java:214)
at android.app.ActivityThread.main(ActivityThread.java:7050)
at java.lang.reflect.Method.invoke(Native Method)
at com.android.internal.os.RuntimeInit$MethodAndArgsCaller.run(RuntimeInit.java:494)
2020-05-05 07:42:48.021 6567-6567/com.example.myapplication E/AndroidRuntime: at com.android.internal.os.ZygoteInit.main(ZygoteInit.java:965)
2020-05-05 07:42:48.066 6567-6567/com.example.myapplication I/Process: Sending signal. PID: 6567 SIG: 9
```

Abbildung 91 Logcat

### 17.2.2 WebView

WebView ist der wahrscheinlich einfachste Variante die aber einen Haken hat. Da die Seite die wir in unsere App haben wollen ein Login benötigt wird bei einem WebView eine 404-Seite hervorgerufen.



© 2020 Wibilea AG | [Backend](#)

Abbildung 92 WebView

### 17.2.3 3.Versuch Internet Tutorial

Als drittes haben wir probiert über ein Internet [Tutorial](#) ausprobiert. Zu Beginn funktionierte alles ganz gut die wurde gebildet. Allerdings stürzte die App immer ab wenn man den Post aufrufen wollte.

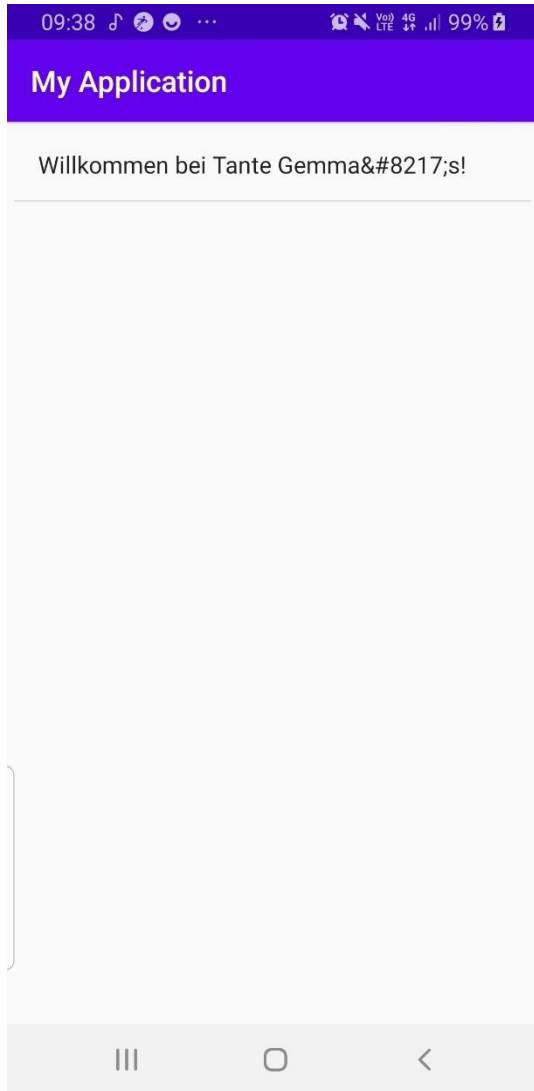
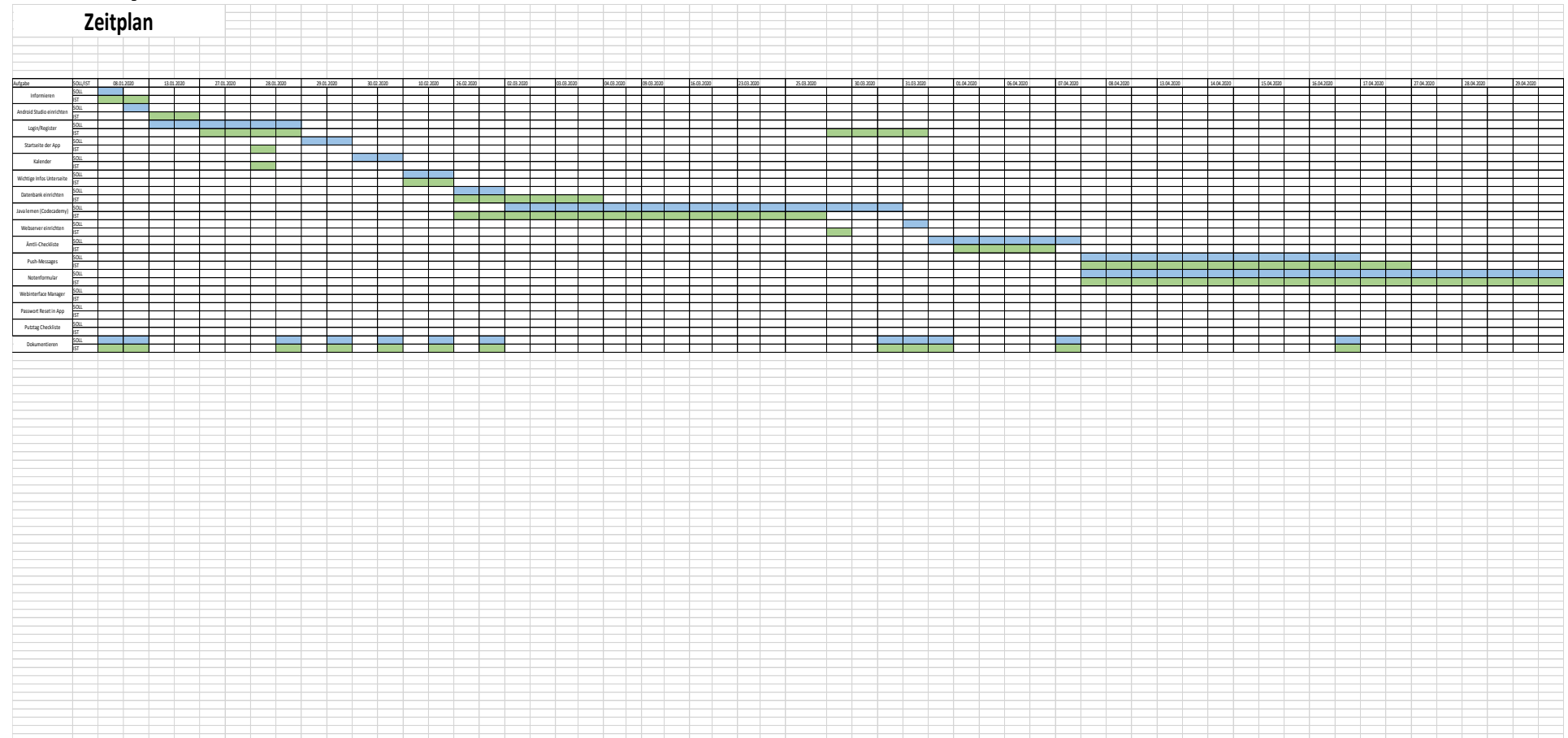


Abbildung 93 Wordpress in Android Studio

## 18 Zeitplan





# 19 Arbeitsjournal

19.1.1 08.01.2019

Tätigkeit	Erledigte Arbeiten / Erreichte Ziele	Erfolge & neu gelernt	Aufgetretene Probleme	Genutzte Quellen
Informieren	Informiert/ Android Studio	Android Studio	-	<a href="https://www.google.com">google.com</a>

19.1.2 13.01.2019

Tätigkeit	Erledigte Arbeiten / Erreichte Ziele	Erfolge & neu gelernt	Aufgetretene Probleme	Genutzte Quellen
Android Studio				
Java	Tutorials über Java	Java kennengelernt	----- -----	<a href="https://www.w3schools.com/JAVA/default.asp">https://www.w3schools.com/JAVA/default.asp</a>

### 19.1.3 27.01.2020

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
Login Button			Fragment Verlinkung ging nicht	stackoverflow, android studio help, YouTube

### 19.1.4 28.01.2020

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
Startseite	Startseite fertig designed	Layout einer Seite auf Android Studio	-	YouTube
Login	Button+Seite	Gelernt wie man Activitys verlinkt	-	YouTube
Kalender	Der Kalender des Basislehrjahrs wurde zur entsprechenden Seite hinzugefügt	Gelernt wie man WebView benutzt	Der Kalender wird nicht ganz zur entsprechenden Grösse verändert. (Media Queries funktioniert nicht ganz)	Stackoverflow

### 19.1.5 29.01.2020

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
Dokumentieren	Dokumentation erweitert	-	-	Word/Internet

### 19.1.6 30.01.2020

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
Datenbank einbinden	Datenbank für App erstellt mit Tutorial gearbeitet	Funktionen einer Datenbank. PHP Kenntnisse	Browser zeigt nicht an was ich im PHP programmiere.	YouTube
Startseite	Responsive gemacht	Constraint layout	Fehlerhafte Darstellungen	
Registrieren bei Login	XML Design		Im Java-code machte Android Studio Probleme	YouTube

### 19.1.7 10.02.2020

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
Datenbank einbinden	<b>Bei Registrierung wird Datenbankeintrag erstellt jedoch funktioniert das Anmelden nicht</b>	<b>PHP und MySQL Kenntnisse</b>	<b>Anmelden funktioniert noch nicht. Ämtli-Checkliste kann noch nicht aufgerufen werden.</b>	<b>YT Internet</b>

### 19.1.8 26.02.2020

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
Mit Codecademy den Java Kurs begonnen	<b>Java (ging den ganzen Tag)</b>	<b>Erste Einblicke in Java</b>	-	<a href="https://www.codecademy.com/">https://www.codecademy.com/</a>

### 19.1.9 02.03.2020

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
Codecademy Java Kurs	<b>Java</b> <b>(ging den ganzen Tag)</b>	<b>System Print out, variables etc.</b>	-	<a href="https://www.codecademy.com/">https://www.codecademy.com/</a>

### 19.1.10 03.03.2020

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
Codecademy Java Kurs	<b>30% Abeeraam</b> <b>30% Ian</b>	<b>Classes, objects</b>	-	<a href="https://www.codecademy.com/">https://www.codecademy.com/</a>
Checkliste App	<b>Checkliste funktionsfähig</b>	<b>Fragment ordnungsgemäss erstellen</b>	<b>Daten werden nicht gespeichert (z.B nach Neustart von App)</b>	-

**19.1.11      04.03.2020**

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
Codecademy Java Kurs	<b>50% Abeeraam</b>	<b>Conditional operators</b>	-	-
Dokumentieren				

**19.1.12      09.03.2020**

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
Codecademy Java Kurs	<b>92% Abeeraam</b> <b>38% Ian</b>	<b>loops, strings</b>	-	-

19.1.13 25.03.2020

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
Codecademy Java Kurs	<b>Abeeraam/Ian fertig</b>	-	-	-
Webserver + SQL-Server eingerichtet	<b>eingrichtet</b>	<b>Webserver + Datenbank erstellen</b>	<b>PHP-Code funktioniert noch nicht</b>	<a href="https://www.YouTube.com/watch?v=JQXflidfFMo&amp;list=PL60o7ed8E-TztoF2K3y4VdDgT6APZ0ka&amp;index=3">https://www.YouTube.com/watch?v=JQXflidfFMo&amp;list=PL60o7ed8E-TztoF2K3y4VdDgT6APZ0ka&amp;index=3</a>
Dokumentation Java	-	-	-	-

19.1.14 31.03.2020

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
Lerndokumentation	<b>Java fertig dokumentiert weitere Themen begonnen</b>	<b>Java wiederholt und besser eingeprägt</b>	-	google.ch
Login fertig	<b>Datenbank korrekt eingebunden, Login funktioniert</b>	<b>PHP, JSON, Datenbank in App einbinden.</b>		
Push Benachrichtigungen	<b>Firebase kennengelernt</b>	<b>Firebase</b>	-	YouTube.com, Google.ch

19.1.15 06.04.2020

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
Push Benachrichtigungen	<b>Durch einen Knopfdruck erscheint eine Push Nachricht. Ziel für Nächstes Mal: Push Nachrichten ohne Knopfdruck mit Counter in App einbinden.</b>	<b>Wie man Push Nachrichten aktiviert und auslöst</b>	<b>Falscher Code, Android Studio</b>	<b>YouTube.com</b>
Checkbox Status speichern	<b>Checkbox State wird gespeichert</b>	<b>Wie man SharedPreferences verwendet</b>	<b>Ganz viele: App stürzte ab, geht nicht in Fragment, etc.</b>	<b>YouTube.com</b>

19.1.16 07.04.2020

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
Checkbox Status speichern	<b>Werte werden nach 12 Stunden gelöscht</b>	<b>Wie SharedPreferences funktionieren</b>		<b>stackoverflow Android Developer Dokumentation</b>

19.1.17 08.04.2020

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
Login speichern	<b>Login Speicherung</b>	<b>SharedPreferences in anderen Activities aufrufen</b>		<b>stackoverflow</b>



19.1.18 14.04.2020

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
Neues Icon bei Actionbarmenu	<b>Neues Icon</b>	<b>Drawable in Java aufrufen</b>		
Reminder programmiert	<b>Einen Reminder programmiert wo man sagen kann wann man Benachrichtigt werden soll.</b>	<b>Nachricht konfigurieren. An einen Button und einer Uhr Funktionen geben.</b>	<p>- Die Benachrichtigung kann nur angezeigt werden, wenn man in der App ist.</p> <p>-Man kann noch kein Datum auswählen.</p>	<p>Google.com</p> <p>YouTube.com</p>
Remindin10sec	<b>Durch ein Knopfdruck wird eine Nachricht nach 10 Sekunden ausgegeben.</b>	<b>Einfacher Reminder erstellen</b>	-	Youtibe.com

19.1.19 15.04.2020

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
Informiert und dokumentiert über Firebase	<b>Informationen über Firebase.</b>	<b>Was ist Firebase? Für was braucht man Firebase?</b>	-	google.com
Reminder programmiert	<b>Einen Reminder programmiert wo man sagen kann wann man Benachrichtigt werden soll.</b>	<b>Nachricht konfigurieren. An einen Button und einer Uhr Funktionen geben.</b>	<b>Funktionierte zu Beginn ganz gut allerdings funktionierte der ganze Zyklus ein Tag nachher nicht mehr. Auch wenn ich nicht geändert habe.</b>	YouTube.com Google.com
Layout von Notenformular	<b>Layout fertiggestellt</b>	<b>Dropdownmenü erstellen + Radio Buttons</b>		<a href="https://www.YouTube.com/watch?v=on_OrrX7Nw4">https://www.YouTube.com/watch?v=on_OrrX7Nw4</a>

**19.1.20      16.04.2020**

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
Notenformular	<b>Validierung</b>	<b>Verschiedene Möglichkeiten für Validierung</b>	<b>Absturz der App</b>	<b>stackoverflow.com</b>

**19.1.21      17.04.2020**

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
Notenformular	<b>Validierung fertig</b>	<b>Wie Formulare auf Android validiert werden</b>		

**19.1.22      27.04.2020**

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
Notenformular	<b>Notentabelle in App integriert. Zugriff auf Notenformular für nicht-autorisierte Personen verweigert</b>	<b>Daten aus Datenbank entnehmen</b>	<b>NullPointerException (Links der Tabellen konnten nicht empfangen werden wegen Fehler in PHP-Skript)</b>	<b>Login Register Dokumentation (Erfahrung von vorher)</b>

19.1.23 28.04.2020

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
Dokumentation	<b>Bilder beschriftet, Tabellen beschriftet, Index, Fehlende Themen ergänzt(Notifications)</b>			<b>Internet</b>
Wp-Rest API	<b>Informiert über WP Rest API. Tutorial begonnen wie man Inhalte von Wordpresseiten auf eine App bringt.</b>	<b>Wp Rest API Informationen.</b>		<b>YouTube.com, Internet</b>
Notenformular	<b>Code zum Abschieken in der App geschrieben Benutzerrechte für Notenformular extern verwaltbar</b>		<b>Absenden Knopf funktioniert nicht richtig (Nach Validierung passiert nichts)</b>	

19.1.24 29.04.2020

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
WP Rest API	<b>Tutorial beendet</b>	<b>Wp Rest API in Android Studio</b>	<b>Build-Fehler. App stürzt ab wenn man sie aufmachen will. Fehler im Code die noch nicht behandelt wurden</b>	<b>YouTube.com, Internet</b>
Notenformular	<b>Absenden-Knopf funktioniert wie gewünscht</b>	<b>Dateien über POST abschieken</b>	<b>Daten werden nicht in Spreadsheets gespeichert</b>	<b>Erfahrung von Login Register</b>

19.1.25 04.05.2020

Tätigkeit	Erledigte Arbeiten / Erreichte Ziele	Erfolge & neu gelernt	Aufgetretene Probleme	Genutzte Quellen
Notenformular	Daten werden in Spreadsheets eingetragen	Daten über POST zu Spreadsheets schicken	-	<a href="https://github.com/dwyl/learn-to-send-email-via-google-script-html-no-server">https://github.com/dwyl/learn-to-send-email-via-google-script-html-no-server</a>
Push Nachrichten	Übrige Ämtlis & usw. in Firebase eingetragen	-	-	-

19.1.26 05.05.2020

Tätigkeit	Erledigte Arbeiten / Erreichte Ziele	Erfolge & neu gelernt	Aufgetretene Probleme	Genutzte Quellen
WP REST API fixen	Habe nochmal alle * Varianten angeschaut und probiert die App zum Laufen zu bringen allerdings klappte kein Versuch. Habe die Logcats gegoogelt es funktioniert nicht	-	App konnte nicht gebaut werden. Habe den Code nochmal verglichen und angepasst allerdings half nichts	YouTube.com Internet
Server zügeln	Server mit Datenbank und Skripts vom Freehoster zu Hostpoint gezügelt.	Lokales PhpMyAdmin für externe Server verwenden	Seite war nicht im DNS des blnet Netzwerks eingetragen	-

19.1.27 06.05.2020

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
blnet App Webinterface	<b>Webinterface für Notenformular Linkzuweisung fertig</b>	<b>Datenbank von Website aus manageable</b>	<b>PHP-Fehler</b>	-
blnet App Website	<b>Kleine Website zum Vorstellen der blnet-App eingerichtet</b>	-	-	-
Wichtige Infos erstellt	<b>Wichtige Infos-Seite fertig.</b>	<b>XML</b>	<b>Ämtli-Tabelle hat ein paar Probleme verursacht. Jedoch nichts wildes</b>	

19.1.28 11.05.2020

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
blnet App Webinterface	<b>Usertabelle wird in Interface angezeigt + Formularfelder für Management einfügt</b>	-	<b>PHP-Fehler</b>	-
Links (Wichtige Infos)	<b>Links funktionieren jetzt</b>	<b>Die Links funktionieren jetzt. Nachdem ich Probleme hatte mit den Link sah ich den Fehler und er konnte relativ schnell behoben werden</b>		.
Kalender (Portrait -- >Landscape)	<b>Wenn man jetzt auf den Kalender geht dann kommt eine Informations-Meldung die besagt dass man das Handy umdrehen muss um den Kalender zu sehen.</b>	<b>If und else bei einem WebView, Info-Meldung programmieren,</b>	<b>Viele Fehlversuche,</b>	<b>Stackoverflow</b>

19.1.29 12.05.2020

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
blnet App Webinterface	<b>Webinterface für Usertabelle fertig</b>	<b>Datenbank von Website aus manageable</b>	<b>PHP-Fehler</b>	-
blnet App	<b>Unterseite (Passwort zurücksetzen) erstellt</b>	-	<b>Code für die Aktion fehlt noch</b>	-
Kalender (Portrait -->Landscape)	<b>Wenn man jetzt auf den Kalender geht dann kommt eine Informations-Meldung die besagt dass man das Handy umdrehen muss um den Kalender zu sehen.</b>	<b>If und else bei einem WebView, Info-Meldung programmieren,</b>	<b>Viele Fehlversuche,</b>	<b>Stackoverflow</b>
Dokumentiert	<b>Die Erfolge und Misserfolge der letzten Tage dokumentiert</b>	-	.	<b>Google.com</b>
Beginn Putztag-Checkliste	<b>Beginn und der Versuch mit TabLayout verschieden Register auf eine Seite zu bringen (Dies war nicht geplant und ein Wunsch unsere User)</b>	<b>Tab-Layout</b>	<b>Die Variante mit dem TabLayout funktioniert noch nicht. Werden versuchen es mit dem TabLayout zu schaffen.</b>	<b>Google.com, YouTube.com</b>

**19.1.30 13.05.2020**

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
Putztag-Checkliste fertig	<b>Putztag-Checkliste fertig programmiert und erstellt.</b>		<b>Wir haben es nach mehreren erfolglosen beschlossen es ohne TabLayout zu machen. Die Endlösung besteht nun aus zwei Buttons die man betätigen kann um zwischen Ämtli und Putztag-Checkliste hin und her zu wechseln. Zudem wird der Button nach dem Click enabled das man weiss auf welcher Liste man sich befindet.</b>	

**19.1.31 18.05.2020**

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
Dokumentation	<b>Dokumentation: erweitert, angepasst, überarbeitet, geändert, ersetzt, korrigiert.</b>	-	-	-
Zeitplan	<b>Zeitplan: erweitert, angepasst, überarbeitet, geändert, ersetzt, korrigiert und komprimiert.</b>	-	-	-



19.1.32      19.05.2020

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
Dokumentation	<b>Dokumentation: erweitert, angepasst, überarbeitet, geändert, ersetzt, korrigiert.</b>	-	-	-
Zeitplan	<b>Zeitplan: erweitert, angepasst, überarbeitet, geändert, ersetzt, korrigiert und komprimiert.</b>	-	-	-

19.1.33      20.05.2020

<b>Tätigkeit</b>	<b>Erledigte Arbeiten / Erreichte Ziele</b>	<b>Erfolge &amp; neu gelernt</b>	<b>Aufgetretene Probleme</b>	<b>Genutzte Quellen</b>
Dokumentation	<b>Dokumentation: erweitert, angepasst, überarbeitet, geändert, ersetzt, korrigiert.</b>	-	-	-
Zeitplan	<b>Zeitplan: erweitert, angepasst, überarbeitet, geändert, ersetzt, korrigiert und komprimiert.</b>	-	-	-

## 20 Testprotokoll blnet App

### Release-Management

Version	Datum	Autor	Kommentar
1.0	28.04.2020	Abeeraam Rahunenthiran Ian Hild	Erstellung Testprotokoll

*Tabelle 20 Release Management*

## 21 Ausgangslage

Auftraggeber	Wibilea AG, Abeeraam Rahunenthiran, Ian Hild, Industrieplatz 8212 Neuhausen
Auftragnehmer	Herr Abeeraam Rahunenthiran & Herr Ian Hild
Autor	Herr Abeeraam Rahunenthiran & Herr Ian Hild

*Tabelle 21 Ausgangslage*

### 21.1 Testgegenstand

Testgegenstand	Beschreibung
<i>blnet App Version 1.0.X</i>	

*Tabelle 22 Testgegenstand*

## 22 Testfälle

### 22.1 Testfall 1 App installieren

Beschreibung	App installieren
Mindestanforderungen	Internetverbindung, 10MB freier Speicher, Android 7.0
Testschritte	<ol style="list-style-type: none"><li>1. Öffnen die Seite <a href="http://app.blnet.ch">app.blnet.ch</a> im Browser</li><li>2. Scrollen sie hinunter bis Sie den Abschnitt herunterladen sehen.</li><li>3. Drücken Sie auf die Taste «herunterladen»</li><li>4. Wählen Sie die neuste Version der App aus. (die oberste).</li><li>5. Drücken Sie auf Assets</li><li>6. Drücken Sie auf die jetzt erschienene .apk Datei</li><li>7. Installieren Sie die App «auf Öffnen klicken wenn Sie heruntergeladen wurde.»</li></ol>
Erwartetes Ergebnis	Die App startet und Sie sehen die Startseite.
Testresultat	

Tabelle 23 Testfall 1 App installieren

## 22.2 Testfall 2 – Anmelden und Registrieren

Beschreibung	Anmelden und Registrieren
Mindestanforderungen	Internetverbindung, 10MB freier Speicher, Android 7.0
Testschritte	<ol style="list-style-type: none"><li>1. Die App starten</li><li>2. Oben rechts auf das Personenzeichen drücken</li><li>3. Im Menü auf «Registrieren» drücken</li><li>4. Folgende Daten eingeben und auf Registrieren klicken: Vorname: Sara Nachname: Stooob E-Mail: <a href="mailto:sara.stoob@gmail.com">sara.stoob@gmail.com</a> Passwort: Bananen.123</li><li>5. Jetzt müssten Sie auf die Login-Seite weitergeleitet worden sein. Geben sie hier die E-Mail und das Passwort von oben ein und drücken Sie auf «Anmelden».</li></ol>
Erwartetes Ergebnis	Eine Toast-Meldung erscheint. Sie sieht etwa so aus: »Willkommen zurück, Sara Stooob«
Testresultat	

Tabelle 24 Testfall 2 Anmelden und Registrieren

## 22.3 Testfall 3 Passwort zurücksetzen

Beschreibung	Passwort zurücksetzen
Mindestanforderungen	Internetverbindung, 10MB freier Speicher, Android 7.0
Testschritte	<ol style="list-style-type: none"> <li>1. App starten</li> <li>2. Oben rechts auf das Personenzeichen drücken</li> <li>3. Im Menü auf «Passwort zurücksetzen» drücken</li> <li>4. Jetzt folgende Daten eingeben: E-Mail: sara.stoob@blnet.ch Altes Passwort: Bananen.123 Neues Passwort + bestätigen: Gurken.456</li> <li>5. Drücken Sie dann auf «Zurücksetzen»</li> <li>6. Jetzt müssten Sie auf die Login-Seite weitergeleitet worden sein. Geben sie hier die E-Mail und das neue Passwort von oben ein und drücken Sie auf «Anmelden».</li> </ol>
Erwartetes Ergebnis	<p>Eine Toast-Meldung erscheint nach dem Drücken auf «Zurücksetzen». Sie sieht etwa so aus: »Das Passwort wurde erfolgreich zurückgesetzt«</p> <p>Eine Toast-Meldung erscheint nach der Anmeldung. Sie sieht etwa so aus: »Willkommen zurück, Sara Stoob«</p>
Testresultat	

Tabelle 25 Testfall 3 Passwort zurücksetzen

## 22.4 Testfall 4 Notenformular

Beschreibung	Notenformular
Mindestanforderungen	Internetverbindung, 10MB freier Speicher, Android 7.0
Testschritte	<ol style="list-style-type: none"> <li>1. Melden Sie sich zuerst mit diesem Account an: <a href="mailto:debug@blnet.ch">debug@blnet.ch</a> Bananen.123</li> <li>2. Öffnen Sie dann das Notenformular über die Navigation</li> <li>3. Geben Sie dort jetzt folgende Daten ein: Datum: 11.09.2001 Fach: Geschichte und Politik Thema: Ground Zero Notentyp: Note-E Note: 4.69 Begründung nicht ausfüllen</li> <li>4. Drücken Sie dann auf «Absenden»</li> <li>5. Öffnen Sie dann das Notenformular über die Navigation</li> <li>6. Drücken Sie auf «Notentabelle» (zu unterst)</li> <li>7. Wechseln Sie auf das Tabellenblatt «Formularantworten»</li> <li>8. Überprüfen Sie, ob ihr Eintrag vorhanden ist. (Wenn nicht, probieren sie es neu zu laden)</li> </ol>
Erwartetes Ergebnis	<p>Eine Toast-Meldung erscheint nach dem Drücken auf «Absenden». Sie sieht etwa so aus: »Das Formular wurde erfolgreich abgeschickt«</p> <p>Eintrag ist in der Tabelle vorhanden.</p>
Testresultat	

Tabelle 26 Testfall 4 Notenformular

## 22.5 Testfall 5 Kalender

Beschreibung	Kalender
Mindestanforderungen	Internetverbindung, 10MB freier Speicher, Android 7.0
Testschritte	<ol style="list-style-type: none"><li>1. Starten Sie die App,</li><li>2. Öffnen sie in der Navigation die Seite «Kalender» im Portraitmodus</li><li>3. Fehlermeldung erscheint</li><li>4. Drehen Sie ihr Handy zur Landscapeansicht</li><li>5. Schauen ob der Kalender in der Landscapeansicht geladen wird</li></ol>
Erwartetes Ergebnis	Der Kalender wird erst geladen wenn man das Handy gedreht hat und sobald das Handy wieder in die Portraitansicht wechselt erscheint die Fehlermeldung erneut.
Testresultat	

Tabelle 27 Testfall 5 Kalender



## 22.6 Testfall 6 Checklisten

Beschreibung	Die beiden Checklisten und das Umschalten zwischen Putztag und Ämtli-Checkliste funktioniert
Mindestanforderungen	Internetverbindung, 10MB freier Speicher, Android 7.0
Testschritte	<ol style="list-style-type: none"> <li>1. Starten Sie die App,</li> <li>2. Öffnen sie in der Navigation die Seite «Checkliste»</li> <li>3. Haken setzen bei: Paketdienst und Kühlschrank checken.</li> <li>4. Schliessen sie die App im Verlauf.</li> <li>5. Öffnen Sie die App und gehen sie wieder zu Checkliste.</li> <li>6. Überprüfen Sie ob die eingegebenen Werte aktiv geblieben sind.</li> <li>7. Aktivieren sie die Putztag Checkliste</li> <li>8. Haken setzen bei: Boden Reinigung aller Pausenplatz und Aschenbecher leeren.</li> <li>9. Schliessen sie die App im Verlauf.</li> <li>10. Öffnen Sie die App und gehen sie diesmal zur Putztagcheckliste.</li> <li>11. Überprüfen Sie ob die eingegebenen Werte aktiv geblieben sind.</li> </ol>
Erwartetes Ergebnis	Die zu Beginn eingegebenen Werte sind aktiv geblieben und der Haken ist immer noch gesetzt. Auch das Umschalten zwischen Ämtli und Putztagcheckliste funktioniert einwandfrei
Testresultat	

Tabelle 28 Testfall 6 Checklisten

## 22.7 Testfall 6 Links

Beschreibung	Links funktionieren
Mindestanforderungen	Internetverbindung, 10MB freier Speicher, Android 7.0 Firefox Browser
Testschritte	<ol style="list-style-type: none"> <li>1. Starten Sie die App,</li> <li>2. Öffnen sie in der Navigation die Seite «Wichtige Infos»</li> <li>3. Klicken Sie den Link «RE_IT_Richtlinien_Lernende»</li> <li>4. Überprüfen sie ob die IT Richtlinien aufgeschlagen werden.</li> <li>5. Klicken Sie alle restlichen Links und überprüfen ob jeder Link das richtige öffnet. «Neuster Leitfaden auf dem Wibilea Extranet»&gt; <a href="https://www.wibilea.ch/extranet/leitfaden/">https://www.wibilea.ch/extranet/leitfaden/</a> «Beispiel Wibilea»&gt; <a href="https://www.bl-net.ch/wp-content/uploads/2020/03/VO_Abwesenheitsmeldung_beispiel_bl-net.jpg">https://www.bl-net.ch/wp-content/uploads/2020/03/VO_Abwesenheitsmeldung_beispiel_bl-net.jpg</a> «Beispiel Syntegon»&gt; <a href="https://www.bl-net.ch/wp-content/uploads/2020/03/Abwesenheitsmeldung_Syntegon_beispiel.jpg">https://www.bl-net.ch/wp-content/uploads/2020/03/Abwesenheitsmeldung_Syntegon_beispiel.jpg</a></li> </ol>
Erwartetes Ergebnis	Alle Links funktionieren
Testresultat	

Tabelle 29 Testfall 6 Links

## 23 Testergebnis

Tester	
Datum Testdurchführung	
Anzahl Fehler	
Fehlerbeschreibung	

*Tabelle 30 Testergebnis*

## 24 Glossar

Erklärung	Begriff
ADB	Android Debug Bridge
API	Application Programming Interface
APK	Android Package Kit
args	Argument
br	Break
CSS	Cascading Style Sheet
EFZ	Eidgenössisches Fähigkeitszeugnis
FCM	Firestore Cloud Messaging
GF	Georg Fischer
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ID	Identifier
IDE	Integrated Development Environment
iOS	Internetwork Operating System
javac	Java Compiler
JS	JavaScript
JSON	JavaScript Object Notation
PHP	Hypertext Preprocessor
PNG	Portable Network Graphics
println	Print line
RAM	Random Acces Memory
ROM	Read Only Memory
SIG	Schweizerische Industrie-Gesellschaft
SQL	Structured Query Language
SVG	Scalable Vector Graphics
TV	Television
USB	Universal Serial Bus
XML	Extensible Markup Language

## 25 Abbildungsverzeichnis

Abbildung 1 Android Studio Download .....	6
Abbildung 2 Android Studios Lizenzbedingungen .....	6
Abbildung 3 Android Studios Download .....	6
Abbildung 4 Willkommensfenster Android Studio.....	6
Abbildung 5 Installation Komponenten .....	6
Abbildung 6 Auswahl Speicherort .....	7
Abbildung 7 Shortcut Auswahl .....	7
Abbildung 8 Durchführung Installation .....	7
Abbildung 9 Abgeschlossene Installation.....	7
Abbildung 10 Neues Projekt Android Studio.....	13
Abbildung 11 Projekt auswählen.....	13
Abbildung 12 Konfiguration des Projekts .....	13
Abbildung 13 Projekt wird erstellt.....	14
Abbildung 14 Android Studio Aufbau.....	15
Abbildung 15 Bild auf dem Desktop .....	16
Abbildung 16 Ordner mipmap.....	16
Abbildung 17 Design erstellen.....	16
Abbildung 18 Bild nach Wahl auswählen .....	17
Abbildung 19 Bild in der APP .....	17
Abbildung 20 Zentrierung (Höhe).....	17
Abbildung 21 Zentrierung (Breite).....	17
Abbildung 22 Debug Emulator .....	18
Abbildung 23 Zentriertes Bild in der App .....	18
Abbildung 24 Text im strings.XML .....	18
Abbildung 25 Funktiongebung .....	18
Abbildung 26 XML String .....	19
Abbildung 27 Kontext Menü.....	20
Abbildung 28 Auswahl Bundle/APK.....	20
Abbildung 29 Key-Store .....	20
Abbildung 30 Auswahl Debug/Release.....	21
Abbildung 31 Build Fortschritte.....	21
Abbildung 32 Zielordner .....	21
Abbildung 33 Activity erstellen.....	22
Abbildung 34Activity erstellen .....	22
Abbildung 35 - XML-Datei für Fragment erstellen .....	24
Abbildung 36 Fragment erstellen .....	25
Abbildung 37 Java Code.....	26
Abbildung 38 Fragment .....	28
Abbildung 39 USB-Debugging Weitere Einstellungen.....	29

Abbildung 40 USB-Debugging Entwickleroptionen .....	30
Abbildung 41 USB-Debugging einschalten .....	31
Abbildung 42 USB-Debugging von Android Studio .....	31
Abbildung 43 App installieren .....	32
Abbildung 44 Die App auf dem Handy .....	32
Abbildung 45 Auslöser Button.....	35
Abbildung 46 Definition des Buttons im string.XML .....	35
Abbildung 47 onClick Statement Button .....	35
Abbildung 48 Button Definition Java .....	35
Abbildung 49 findElementById .....	36
Abbildung 50 Ausgabe des Toasts .....	36
Abbildung 51 Ausgabe nach dem Click.....	36
Abbildung 52 Layout der Seite .....	37
Abbildung 53 Manifest Dokument .....	37
Abbildung 54 Eigene Java-datei.....	38
Abbildung 55 Konfiguration Benachrichtigungen .....	39
Abbildung 56 Endresultat .....	39
Abbildung 57 MainActivity Datei.....	40
Abbildung 58 AndroidManifest Datei.....	40
Abbildung 59 MainActivity(Java) Datei .....	41
Abbildung 60 Externe Java Datei .....	41
Abbildung 61 BL App Login Fenster .....	42
Abbildung 62 phpMyAdmin WebGUI .....	45
Abbildung 63 Datenbankeinträge .....	45
Abbildung 64 Checklisten .....	51
Abbildung 65 Variabel Deklaration.....	51
Abbildung 66 Zuweisung der Buttons .....	52
Abbildung 67 Checkliste verstecken.....	52
Abbildung 68 Button-Konfiguration .....	53
Abbildung 69 Button-Konfiguration .....	53
Abbildung 70 Checklisten .....	53
Abbildung 71 Notenformular .....	54
Abbildung 72 Android Studios Tools .....	62
Abbildung 73 Cloud Messaging Android Studio .....	62
Abbildung 74 Firebase im Browser.....	63
Abbildung 75 Firebase Projektname .....	63
Abbildung 76 Firebase Lizenzbedingungen .....	63
Abbildung 77 Android Studio und Firebase verbinden .....	63
Abbildung 78 FCM einfügen .....	63
Abbildung 79 FCM akzeptieren .....	64

Abbildung 80 Benachrichtigung erstellen .....	65
Abbildung 81 Ziel Definition .....	65
Abbildung 82 Zeitplan Firebase .....	65
Abbildung 83 Nachricht Überprüfung .....	65
Abbildung 84 Nutzereigenschaft .....	66
Abbildung 85 Benachrichtigung Text.....	66
Abbildung 86 Ziel-E-Mail Angabe .....	66
Abbildung 87 Zeitplan Firebase .....	66
Abbildung 88 Benachrichtigung auf dem Handy .....	67
Abbildung 89 WordPress REST API (Version 2) .....	68
Abbildung 90 REST API – Filter Fields .....	68
Abbildung 91 Logcat .....	69
Abbildung 92 WebView .....	70
Abbildung 93 Wordpress in Android Studio .....	71

## 26 Tabellenverzeichnis

Tabelle 1 Android Studio Installation .....	7
Tabelle 2 Datentypen .....	8
Tabelle 3 Loops .....	11
Tabelle 4 Operatoren.....	12
Tabelle 5 Neues Projekt Android Studio .....	14
Tabelle 6 Bilder in Android Studios .....	17
Tabelle 7 Bilder ausrichten in Android Studios .....	18
Tabelle 8 APK .....	21
Tabelle 9 Activity erstellen .....	22
Tabelle 10 USB-Debugging .....	32
Tabelle 11 Toasts .....	36
Tabelle 12 Push Knopf .....	39
Tabelle 13 Reminder erstellen.....	41
Tabelle 14 Zweite Checkliste .....	53
Tabelle 15 Firebase verbinden bin .....	64
Tabelle 16 Cloud Messaging .....	66
Tabelle 17 User Properties .....	66
Tabelle 18 Spezifizierte Benachrichtigung.....	67
Tabelle 19 Wordpress Rest API .....	68
Tabelle 20 Release Management .....	90
Tabelle 21 Ausgangslage .....	91
Tabelle 22 Testgegenstand .....	91
Tabelle 23 Testfall 1 App installieren .....	92
Tabelle 24 Testfall 2 Anmelden und Registrieren .....	93
Tabelle 25 Testfall 3 Passwort zurücksetzen .....	94
Tabelle 26 Testfall 4 Notenformular .....	95
Tabelle 27 Testfall 5 Kalender .....	96
Tabelle 28 Testfall 6 Checklisten .....	97
Tabelle 29 Testfall 6 Links.....	98
Tabelle 30 Testergebnis.....	99



## 27 Index

### A

Android 6, 7, 13, 14, 15, 16, 17, 18, 20, 29, 30, 31,  
32, 33, 35, 38, 42, 54, 55, 62, 63, 65, 100  
Android Studio 6, 13, 14, 15, 16, 20, 29, 31, 32, 42  
APK 20, 21  
app 21, 24, 28, 42  
App 15, 17, 18, 20, 21, 33  
Arrays 12

### B

Benachrichtigungen 35, 38, 39, 62, 66  
Beschreibung 8, 11, 12, 16, 20, 22, 29  
boolean 8, 10  
Bytes 8

### C

Case 9  
Char 8  
Cloud Messaging 62, 63, 65

### D

Debugging 29  
double 8, 10  
Do-While 11  
Dropdown 54, 55

### E

Entwickleroptionen 30, 31  
Erklärung 6, 13, 17, 35, 37, 38, 40, 62, 65, 66, 100

### F

Firebase 41, 62, 63, 65  
For-Schleife 11  
Fragments 24, 25, 26

### G

G-Spreadsheets 54

### I

id 25, 28, 33, 42  
If else 9  
Imageview 16

Installation 6, 7, 13  
int 8, 9, 10, 11  
iOS 62

### J

Ja va 8  
Java 8, 9, 10, 11, 12, 13, 26, 33, 43  
Java Script 9, 11  
JavaScript 54  
JSON-Datei 44, 57  
JSONResponse 57

### K

Keystore 20

### L

Lizenzbedingungen 6, 63  
Login 42

### M

Methoden 10  
mipmap 16

### N

Notifications 38, 41, 62

### O

Operatoren 12

### P

PHPMyAdmin 54  
public 9, 10, 12, 27, 28, 33

### S

Sicherheitsfeatures 56  
Software 13  
Spreadsheet 54, 56  
SQL-Datenbank 54  
String 8, 9, 10, 12, 27, 28, 33  
Switch 9

## **T**

TimePicker 40

## **U**

USB-Debugging 29, 30, 31, 32  
User Property 66

## **V**

Validierung 54, 55  
Variablen 10, 12

## **W**

Webinterface 54

Webserver 42  
WebView 33  
while-Schleife 11  
WP Rest API 68

## **X**

Xiaomi 29, 31  
xml 25, 42  
XML 16

## **Z**

Zeitplan 65, 66