****

|  |  |
| --- | --- |
| 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](#_Toc40879596)

[2 Java 8](#_Toc40879597)

[2.1 Java Einführung 8](#_Toc40879598)

[2.2 Datentypen 8](#_Toc40879599)

[2.3 If else/Switch Anweisungen 9](#_Toc40879600)

[2.3.1 If/Else 9](#_Toc40879601)

[2.3.2 Switch 9](#_Toc40879602)

[2.4 Variablen 10](#_Toc40879603)

[2.5 Class 10](#_Toc40879604)

[2.6 Methoden 10](#_Toc40879605)

[2.7 Loops 11](#_Toc40879606)

[2.8 Operatoren 12](#_Toc40879607)

[2.9 Arrays 12](#_Toc40879608)

[2.10 Comments 12](#_Toc40879609)

[3 Android Studio 13](#_Toc40879610)

[3.1 Neues Projekt erstellen 13](#_Toc40879611)

[4 Android App Aufbau 15](#_Toc40879612)

[5 Android Studios Grundlagen 16](#_Toc40879613)

[5.1 Bilder hinzufügen 16](#_Toc40879614)

[5.1.1 Bilder ausrichten 17](#_Toc40879615)

[5.2 Links 18](#_Toc40879616)

[6 Android App als APK exportieren 20](#_Toc40879617)

[7 Activity erstellen 22](#_Toc40879618)

[7.1 Activity in Navigation einbinden 22](#_Toc40879619)

[7.1.1 XML-Code 22](#_Toc40879620)

[7.1.2 Java-Code 23](#_Toc40879621)

[8 Fragment erstellen 24](#_Toc40879622)

[8.1 Fragment in Navigation einbinden 25](#_Toc40879623)

[8.1.1 XML-Code 25](#_Toc40879624)

[8.1.2 Java-Code 26](#_Toc40879625)

[9 Debugging 29](#_Toc40879626)

[9.1 Debug-Release über USB installieren 29](#_Toc40879627)

[10 WebView 33](#_Toc40879628)

[10.1 Landscape oder Portrait 33](#_Toc40879629)

[11 Toasts 35](#_Toc40879630)

[11.1 Was sind Toasts? 35](#_Toc40879631)

[11.2 Toasts in Android Studio 35](#_Toc40879632)

[12 Push Nachrichten 37](#_Toc40879633)

[12.1 Push durch Knopf 37](#_Toc40879634)

[12.2 Reminder erstellen 40](#_Toc40879635)

[13 Login und Registration 42](#_Toc40879636)

[13.1 XML-Code 42](#_Toc40879637)

[13.2 Java-Code 43](#_Toc40879638)

[13.3 PHP-Code 44](#_Toc40879639)

[13.4 SQL-Tabelle 44](#_Toc40879640)

[13.5 Passwort ändern 45](#_Toc40879641)

[14 Checklisten 49](#_Toc40879642)

[14.1 XML-Code 49](#_Toc40879643)

[14.2 Java-Code (Werte speichern) 49](#_Toc40879644)

[14.3 Zweite Checkliste 51](#_Toc40879645)

[15 Notenformular 54](#_Toc40879646)

[15.1 Konzept 54](#_Toc40879647)

[15.2 Design 54](#_Toc40879648)

[15.3 Validation 55](#_Toc40879649)

[15.4 User Berechtigungen 56](#_Toc40879650)

[15.5 SQL-Tabelle 58](#_Toc40879651)

[15.6 Absende-Request 58](#_Toc40879652)

[15.7 App Script (Google Scripts) 60](#_Toc40879653)

[16 Firebase 62](#_Toc40879654)

[16.1 Was ist Firebase? 62](#_Toc40879655)

[16.2 Firebase mit Android Studio verbinden 62](#_Toc40879656)

[16.3 Cloud Messaging 65](#_Toc40879657)

[16.4 Spezifizierte Benachrichtigungen 66](#_Toc40879658)

[16.4.1 User Properties 66](#_Toc40879659)

[16.4.2 Spezifizierte Nachricht 66](#_Toc40879660)

[17 Word Press Rest API 68](#_Toc40879661)

[17.1 Was ist WP Rest API? 68](#_Toc40879662)

[17.2 Versuche 69](#_Toc40879663)

[17.2.1 1.Versuch YouTube Tutorial 69](#_Toc40879664)

[17.2.2 WebView 70](#_Toc40879665)

[17.2.3 3.Versuch Internet Tutorial 71](#_Toc40879666)

[18 Zeitplan 72](#_Toc40879667)

[19 Arbeitsjournal 73](#_Toc40879668)

[19.1.1 08.01.2019 73](#_Toc40879669)

[19.1.2 13.01.2019 73](#_Toc40879670)

[19.1.3 27.01.2020 74](#_Toc40879671)

[19.1.4 28.01.2020 74](#_Toc40879672)

[19.1.5 29.01.2020 75](#_Toc40879673)

[19.1.6 30.01.2020 75](#_Toc40879674)

[19.1.7 10.02.2020 76](#_Toc40879675)

[19.1.8 26.02.2020 76](#_Toc40879676)

[19.1.9 02.03.2020 77](#_Toc40879677)

[19.1.10 03.03.2020 77](#_Toc40879678)

[19.1.11 04.03.2020 78](#_Toc40879679)

[19.1.12 09.03.2020 78](#_Toc40879680)

[19.1.13 25.03.2020 79](#_Toc40879681)

[19.1.14 31.03.2020 79](#_Toc40879682)

[19.1.15 06.04.2020 80](#_Toc40879683)

[19.1.16 07.04.2020 80](#_Toc40879684)

[19.1.17 08.04.2020 80](#_Toc40879685)

[19.1.18 14.04.2020 81](#_Toc40879686)

[19.1.19 15.04.2020 82](#_Toc40879687)

[19.1.20 16.04.2020 83](#_Toc40879688)

[19.1.21 17.04.2020 83](#_Toc40879689)

[19.1.22 27.04.2020 83](#_Toc40879690)

[19.1.23 28.04.2020 84](#_Toc40879691)

[19.1.24 29.04.2020 84](#_Toc40879692)

[19.1.25 04.05.2020 85](#_Toc40879693)

[19.1.26 05.05.2020 85](#_Toc40879694)

[19.1.27 06.05.2020 86](#_Toc40879695)

[19.1.28 11.05.2020 86](#_Toc40879696)

[19.1.29 12.05.2020 87](#_Toc40879697)

[19.1.30 13.05.2020 88](#_Toc40879698)

[19.1.31 18.05.2020 88](#_Toc40879699)

[19.1.32 19.05.2020 89](#_Toc40879700)

[19.1.33 20.05.2020 89](#_Toc40879701)

[20 Testprotokoll blnet App 90](#_Toc40879702)

[21 Ausgangslage 91](#_Toc40879703)

[21.1 Testgegenstand 91](#_Toc40879704)

[22 Testfälle 92](#_Toc40879705)

[22.1 Testfall 1 App installieren 92](#_Toc40879706)

[22.2 Testfall 2 – Anmelden und Registrieren 93](#_Toc40879707)

[22.3 Testfall 3 Passwort zurücksetzen 94](#_Toc40879708)

[22.4 Testfall 4 Notenformular 95](#_Toc40879709)

[22.5 Testfall 5 Kalender 96](#_Toc40879710)

[22.6 Testfall 6 Checklisten 97](#_Toc40879711)

[22.7 Testfall 6 Links 98](#_Toc40879712)

[23 Testergebnis 99](#_Toc40879713)

[24 Glossar 100](#_Toc40879714)

[25 Abbildungsverzeichnis 101](#_Toc40879715)

[26 Tabellenverzeichnis 104](#_Toc40879716)

[27 Index 105](#_Toc40879717)

Den Quellcode der App finden Sie auf:

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

# Installation von Android Studio

| Erklärung | Bild |
| --- | --- |
| Android Studio kann und wird über den Browser heruntergeladen, um dies zu tun einfach das grüne Feld anklicken | Abbildung Android Studio Download |
| Als nächsten Schritt muss man die Lizenzbedingungen durchlesen, wenn man damit einverstanden ist, dann kann man einfach auf «gelesen und akzeptieren» drücken. | Abbildung Android Studios Lizenzbedingungen |
| Hat man die Lizenzbedingungen gelesen wird das Android Studio Setup heruntergeladen. | Abbildung Android Studios Download |
| Bei der ersten Seite des Setups wird einfach erklärt was dieses Setup macht. Hier kann man einfach auf Next drücken. | Abbildung Willkommensfenster Android Studio |
| Hier kann man die Komponenten wählen. | Abbildung Installation Komponenten |
| In diesem Fenster wird ausgewählt wo das ganze gespeichert wird. | Abbildung Auswahl Speicherort |
| Im letzten Fenster vor der Installation kann noch der Shortcut gewählt werden | Abbildung Shortcut Auswahl |
| Jetzt wird die Installation durchgeführt, dies dauert meistens so ca. 2 Minuten | Abbildung Durchführung Installation |
| Die Installation ist hiermit abgeschlossen. | Abbildung Abgeschlossene Installation |

Tabelle Android Studio Installation

# Java

## 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.

## 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 Datentypen

## If else/Switch Anweisungen

### 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.

### 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.

## 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;

## 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.

## 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.

## 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. | for(int i=0; i<10; i++)  {  System.out.println("Hello World " + i);  } |
| 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. | int i = 0; while (i < 5) {  System.out.println(i);  i++;  } |
| 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. | int i = 0; do {  System.out.println(i);  i++;  }  while (i < 5); |

Tabelle Loops

## Operatoren

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

|  |  |  |
| --- | --- | --- |
| Operator | Beschreibung | |
| ! | Bedeute 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 wir bei If Statements gebraucht wo Zwei werte drin sind und nur 1 davon true sein muss |

Tabelle Operatoren

## 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  
  
}

## 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 /\*

# 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.

## Neues Projekt erstellen

| Erklärung | Bild |
| --- | --- |
| Um ein neues Projekt auf Android Studios zu erstellen muss man oben Links auf das Feld File. | Abbildung Neues Projekt Android Studio |
| Dann erscheint dieses Fenster, wo man ein Projekt auswählen kann. Bei der frischen Installation kommt dieses Fenster automatisch direkt nach der Installation. | Abbildung Projekt auswählen |
| Hat man ein Projekt gewählt kann man dem Projekt einen Namen geben, entscheiden wo man das Projekt abspeichern will und die Sprache wählen. | Abbildung Konfiguration des Projekts |
| Nun wird das Projekt erstellt dies dauert meistens maximal eine Minute. | Abbildung Projekt wird erstellt |

Tabelle Neues Projekt Android Studio

# Android App Aufbau

Eine Android App ist in Android Studio so aufgebaut.

Hier werden die Einträge der Menüs bestimmt. (z.B. Navigationsmenü)

Im Layout-Ordner befinden sich die ganzen XML-Dateien, die bestimmen, wie eine Seite der App auszusehen hat.

Im Drawable-Ordner werden alle Grafiken, die in der App gebraucht werden, abgelegt.

Im Mipmap-Ordner werden die App-Logos in verschiedenen Grössen gespeichert.

Im Java-Ordner werden alle Java-Dateien abgelegt.

In der Manifest-Datei werden allgemeine Informationen hinterlegt, wie z.B. Name der App, Berechtigungen

In values werden alle Werte (Texte, styles etc.) gespeichert.

In der Manifest-Datei werden allgemeine Informationen hinterlegt, wie z.B. Name der App, Berechtigungen

Hier werden die Einträge der Menüs bestimmt. (z.B. Navigationsmenü)

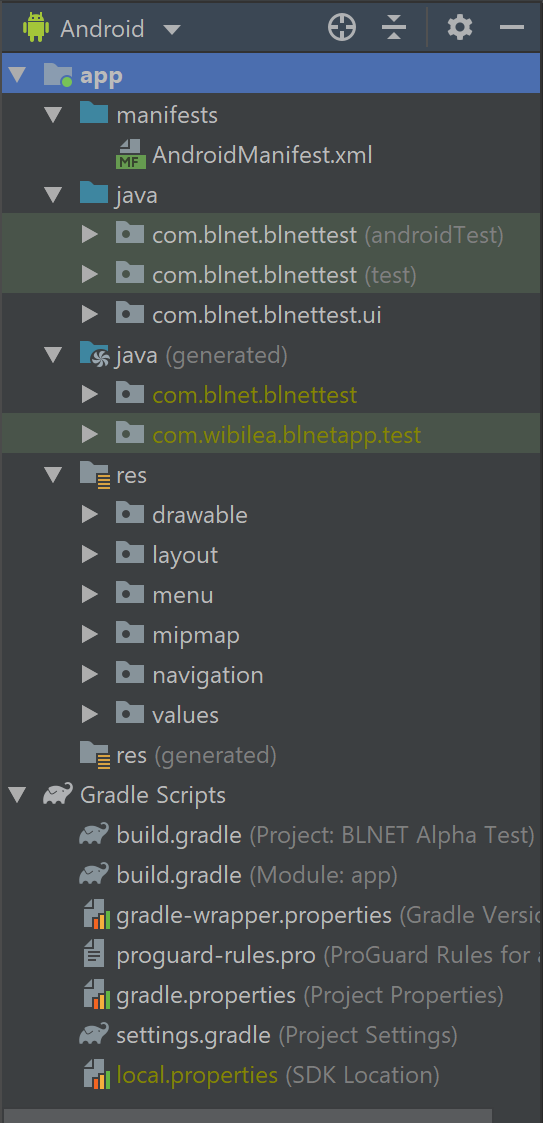


Abbildung Android Studio Aufbau

Gradle basiert auf Java und ist ein Build-Management-Automationstool. Es wird gebraucht, um Code zu kompilieren.

# Android Studios Grundlagen

## Bilder hinzufügen

| Beschreibung | Bild |
| --- | --- |
| Wenn man ein Bild vom Desktop in Android Studio haben will, kann man es per Drag&Drop in einen Ordner legen. | Abbildung Bild auf dem Desktop |
| Danach kann man das Bild in den Ordner «mipmap» ziehen. Wenn das erledigt ist kann man das Bild überall einfügen | Abbildung Ordner mipmap |
| Wenn das Bild sich im Ordner befindet, kann man auf eine XML Datei gehen. dort auf Design klicken und dann ein Imageview hineinziehen. | Abbildung Design erstellen |
| 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. | Abbildung Bild nach Wahl auswählen |

Tabelle Bilder in Android Studios

### 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. | Abbildung Bild in der APP |
| Wenn man das Bild zentriert haben will muss man an den vorher weissen, 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. | Abbildung Zentrierung (Höhe)  Abbildung Zentrierung (Breite) |
| 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. | Abbildung Debug Emulator  Abbildung Zentriertes Bild in der App |

Tabelle Bilder ausrichten in Android Studios

## 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. | Abbildung Text im strings.XML |
| 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 | Abbildung Funktiongebung |
| Dann muss noch in der XML Datei der richtige String angegeben werden. | Abbildung XML String |

# 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 |
| --- | --- |
| Als erstes gehen wir auf «Build» und im Menü dann auf «Generated Signed Bundle /APK». | Screenshot (4)  Abbildung Kontext Menü |
| Hier wählen wir dann APK aus, da wir ja eine APK-Datei haben möchten. | Screenshot (5)  Abbildung Auswahl Bundle/APK |
| Hier muss man einen Keystore auswählen. Falls Sie keinen haben, müssen sie auf «Create New» klicken und die nötigen Infos dort eingeben. | Screenshot (6)  Abbildung Key-Store |
| 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» | Screenshot (7)  Abbildung Auswahl Debug/Release |
| Jetzt warten wir, bis überall grüne Häkchen angezeigt werden. Falls ein Fehler angezeigt wird, Sie es beheben. | Screenshot (8)  Abbildung Build Fortschritte |
| 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. | Screenshot (10)  Abbildung Zielordner |

Tabelle APK

# Activity erstellen

Activitys 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 | Abbildung Activity erstellen |
| Hier stellt man seine Einstellungen an. Wir haben nur den Namen geändert. Dann anschliessend auf «Finish» klicken. | Abbildung Activity erstellen |

Tabelle Activity erstellen

## Activity in Navigation einbinden

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

### 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.blnet.blnettest.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 ChecklisteActivity folgendes eingeben:

android:parentActivityName = ".ui.MainActivity"

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

### 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, checklisteActivity.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.

# 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.

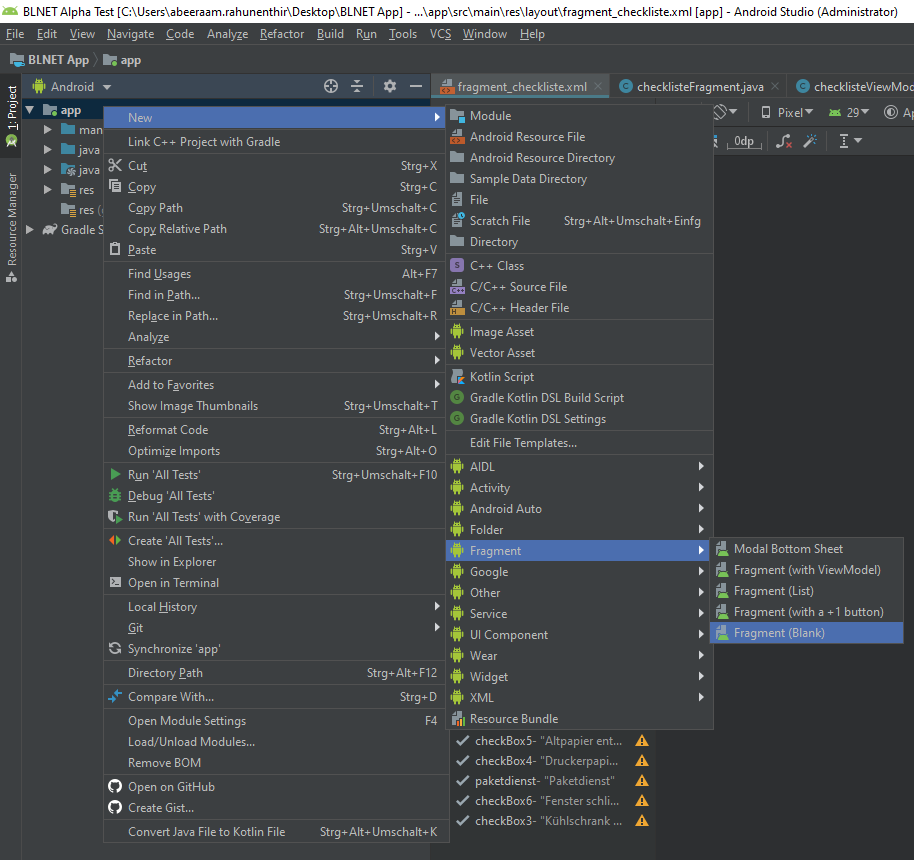


Abbildung - XML-Datei für Fragment erstellen

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

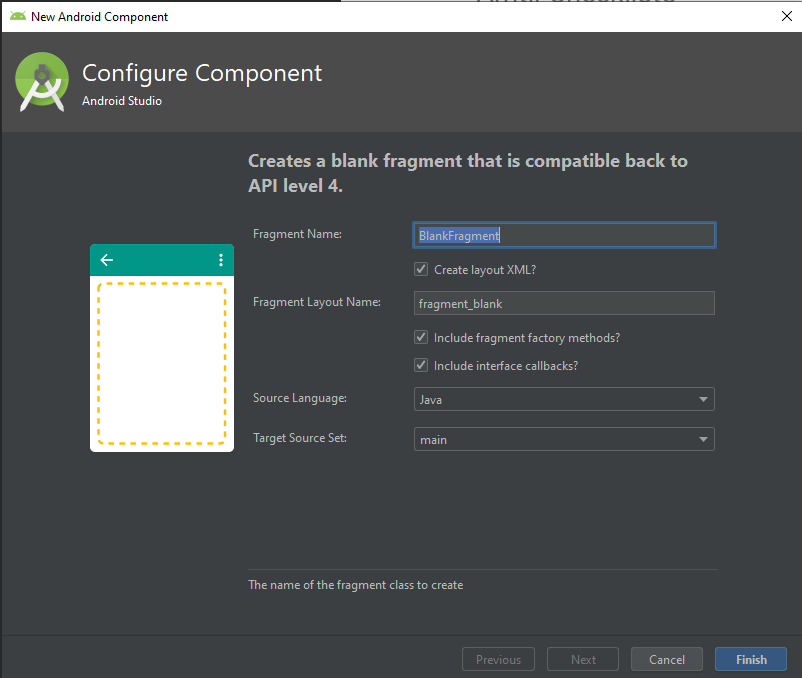


Abbildung Fragment erstellen

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

## Fragment in Navigation einbinden

### 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.

### 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.

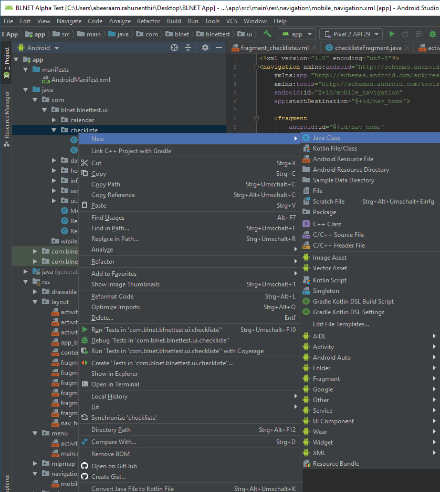


Abbildung Java Code

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.checklisteViewModel.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.

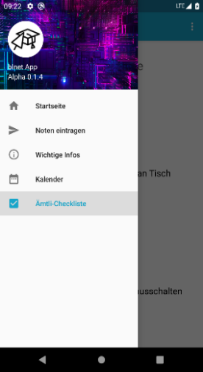
Jetzt müsste man das Fragment benutzen können.

Abbildung Fragment

# Debugging

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

## 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». | C:\Users\abeeraam.rahunenthir\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Screenshot_2020-03-04-11-18-01-842_com.android.settings.jpg  Abbildung USB-Debugging Weitere Einstellungen |
| 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. | Abbildung USB-Debugging Entwickleroptionen |
| In den Entwickleroptionen schalten wir «USB-Debugging» und «Installieren über USB» ein. | C:\Users\abeeraam.rahunenthir\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Screenshot_2020-03-04-11-18-29-539_com.android.settings.jpg  Abbildung USB-Debugging einschalten |
| 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. | C:\Users\abeeraam.rahunenthir\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Screenshot (14).png  Abbildung USB-Debugging von Android Studio |
| Diese Meldung müsste jetzt auf dem Handy erscheinen. Logischerweise wählen wir hier jetzt «Installieren» aus. | C:\Users\abeeraam.rahunenthir\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Screenshot_2020-03-04-11-19-42-909_com.miui.securitycenter.jpg  Abbildung App installieren |
| 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. | C:\Users\abeeraam.rahunenthir\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Screenshot_2020-03-04-11-20-26-613_com.wibilea.blnetapp.jpg  Abbildung Die App auf dem Handy |

Tabelle USB-Debugging

# 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 angeben, 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 werde 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/.

## 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.

# Toasts

## 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».

## 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. | Abbildung Auslöser Button |
| 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’’. | Abbildung Definition des Buttons im string.XML |
| 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. | Abbildung onClick Statement Button |
| In diesem Schritt definiert man den Button in der Java Datei (MainActivity.java). | Abbildung Button Definition Java |
| Danach muss man findElementById einsetzen indem man die ID des buttons angibt. In unserem Fall die ID button. | Abbildung findElementById |
| Zum Schluss muss man noch angeben was es ausgeben soll. Es sollte also auf Mausklick den Text Registrierung erfolgreich ausgeben. | Abbildung Ausgabe des Toasts |
| Das kommt raus, wenn man denn Button gedrückt hat. | C:\Users\ian.hild\Pictures\Screenshots\Screenshot (11).png  Abbildung Ausgabe nach dem Click |

Tabelle Toasts

# Push Nachrichten

## Push durch Knopf

| Erklärung | Bild |
| --- | --- |
| 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. | Abbildung Layout der Seite |
| In der Android Manifest.XML-Datei muss man noch unter «application» android:name’’.app’’ eintragen. | Abbildung Manifest Dokument |

| Erklärung | Bild |
| --- | --- |
| 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 | Abbildung Eigene Java-datei |
| In der Main.java Datei kann man dann noch einstellen wie sich die Nachricht verhalten soll und was passieren soll, wenn diese angeschickt wird. | Abbildung Konfiguration Benachrichtigungen |
| Unser Script ergab folgendes Ergebnis. | Ein Bild, das Computer, sitzend, Monitor, schwarz enthält.  Automatisch generierte Beschreibung  Abbildung Endresultat |

Tabelle Push Knopf

## Reminder erstellen

|  |  |
| --- | --- |
| Erklärung | Bild |
| Zuerst haben wie uns um das Layout der Seite gekümmert dafür haben wir einfach einen TimePicker (Uhr) und einen Button auf die Seite getan. | Abbildung MainActivity Datei |
| 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. | Abbildung AndroidManifest Datei |
| 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). | Abbildung MainActivity(Java) Datei |
| 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. | Abbildung Externe Java Datei |

Tabelle 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.

# Login und Registration

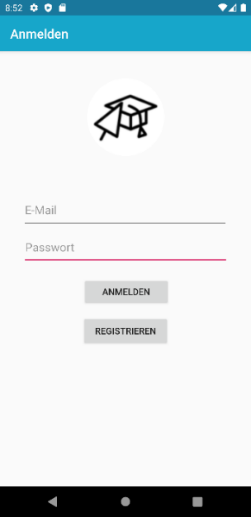
Wir haben Login und Registration programmiert, damit nicht Fremde Zugriff auf gewisse private Inhalte (z.B. Notenformular etc.) haben. Wir haben dafür das Tutorial von Tonikami TV benutzt.

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

## XML-Code

Wir fingen 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"  
geht weiter...



wird:

Abbildung BL App Login Fenster

Java war dann aber ein bisschen herausfordernder als XML.

## 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.blnet.ch/scripts/login.php";   
private Map<String, String> params;

public LoginRequest(String email, String passwort, Response.Listener<String> listener){  
 super (Request.Method.*POST*, *LOGIN\_REQUEST\_URL*, listener, null);  
 params = new HashMap<>();  
 params.put ("email", email);  
 params.put ("passwort", passwort);  
}

Hier werden die Werte «email» und «passwort» 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://tantegemmas.000webhostapp.com/androidlogin/register.PHP";  
private Map<String, String> params;

public RegisterRequest(String vorname, String nachname, String email, String passwort, 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 ("passwort", passwort);  
}

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.

## 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 war 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 abruft, aber wir nur einen Teil davon im PHP-Skript eine Variable zugewiesen hatten. Das hatten wir behoben und dann ging das Login auch ohne Probleme.

## 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 anhaken.

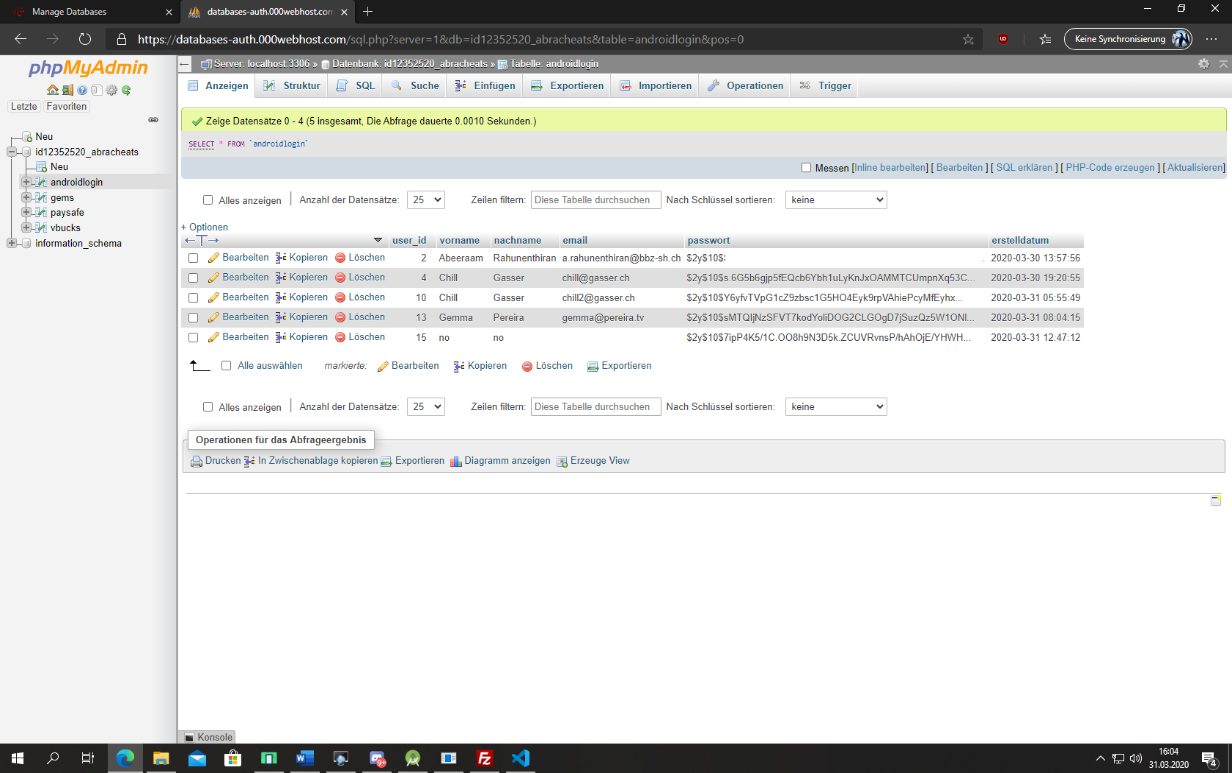


Abbildung phpMyAdmin WebGUI

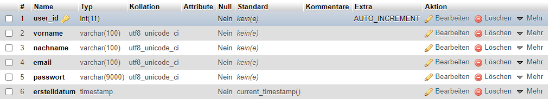


Abbildung Datenbankeinträge

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.

## 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, on 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 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"]);

$passwort = mysqli\_real\_escape\_string($con, $\_REQUEST["newpassword"]);

function registerUser() {

global $con, $email, $passwort;

$passwordHash = password\_hash($passwort, PASSWORD\_DEFAULT);

$statement = mysqli\_prepare($con, "UPDATE androidlogin SET passwort = ? 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.

# Checklisten

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

## 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.

## 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.

## 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. | Abbildung Checklisten |
| In der Java Datei haben wir die die Checkboxes also Kriterien in eine Variabel deklariert. | Abbildung Variabel Deklarierung |
| Danach haben wir unsere Variabel (chk1) einem Kriterium zugewiesen. Wir haben also jeder Variabel eine Checkbox zugewiesen.  Unten haben wir noch die beiden Buttons für Ämtli und Putztag deklariert. | Abbildung Zuweisung der Buttons |
| 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.  Danach noch alle Kriterien der Putztag-Checkliste versteckt damit man nur die Ämtli-Checkliste sieht. | Abbildung Checkliste verstecken |

|  |  |
| --- | --- |
| Erklärung | Bild |
| 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. | Abbildung Button-Konfiguration    Abbildung Button-Konfiguration |
| Am Ende hat das Fragment so ausgesehen. Um den Inhalt zu speichern musste man wieder dasselbe wie vorher bei Kapitel 7.2 machen. | Abbildung Checklisten |

Tabelle Zweite Checkliste

# Notenformular

## 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>

## Design

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

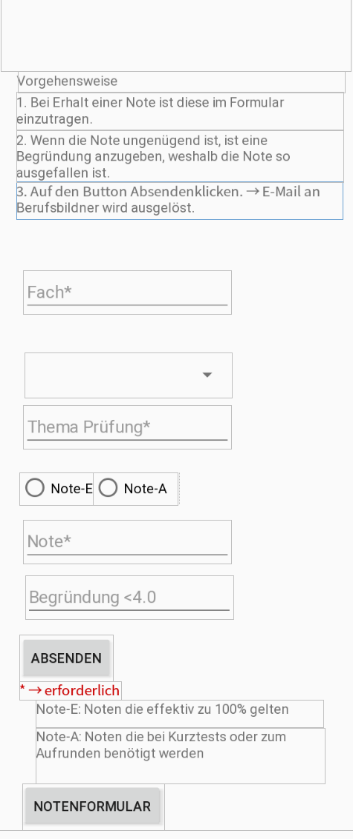


Abbildung Notenformular

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.

## 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.getSelectedItem().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();

## 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 wird 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"]);

$statement = mysqli\_prepare($con, "SELECT \* FROM emailsheet WHERE email = ?");

mysqli\_stmt\_bind\_param($statement, "s", $getemail);

mysqli\_stmt\_execute($statement);

mysqli\_stmt\_store\_result($statement);

mysqli\_stmt\_bind\_result($statement, $colemail, $colsheetlink, $colscriptlink);

$response = array();

while(mysqli\_stmt\_fetch($statement)){

    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>

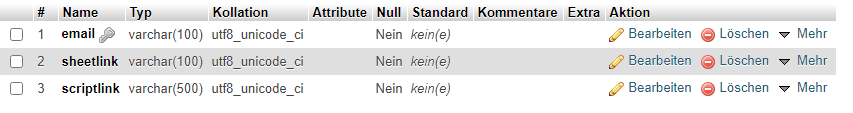
Wenn nicht, schickt sie «success=false» zurück, was die Fehlermeldung auslöst.

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...  
 }  
 }  
};

## 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.



## 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.getSelectedItem().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(getContext());  
 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.

## 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.MimeType.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.

# Firebase

## 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 brauchten 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.

## 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. | Abbildung Android Studios Tools |
| In diesem Fenster kann man dann wählen für was man Firebase brauchen will. In unserem Fall für Benachrichtigungen also Cloud Messaging. | Abbildung Cloud Messaging Android Studio |
| Wenn man dann ‘’Set up Firebase Cloud Messaging’’ öffnet sich im Browser automatisch die Startseite von Firebase. | Abbildung Firebase im Browser |
| Auf der Firebase Website muss man dann auf Projekt erstellen und den Projektnamen eingeben. | Abbildung Firebase Projektname |
| Um die Verbindung abzuschliessen einfach Weiter klicken und am Ende die Lizenzbedingungen lesen und akzeptieren. | Abbildung Firebase Lizenzbedingungen |
| Falls kein Fehler auftritt sollte dieses Fenster auftauchen wo man einfach verbinden klicken muss. | Abbildung Android Studio und Firebase verbinden |
| Um Firebase Cloud Messaging hinzuzufügen muss man noch den zweiten Schritt machen. | Abbildung FCM einfügen |
| Hier muss man wesentlich weniger machen und einfach ‘’Accept Changes’’ klicken. | Abbildung FCM akzeptieren |

Tabelle Firebase verbinden bin

## 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 | Abbildung Benachrichtigung erstellen |
| 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. | Abbildung Ziel Definition |
| 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. | Abbildung Zeitplan Firebase |
| Die restlichen Punkte können übersprungen werden da sie irrelevant für die Nachricht sind. | Abbildung Nachricht Überprüfung |

Tabelle Cloud Messaging

## Spezifizierte Benachrichtigungen

### User Properties

|  |  |  |
| --- | --- | --- |
| Erklärung | Bild | |
| Eine User Property wird ganz leicht erstellt. Einfach unter User Properties eine neue Nutzereigenschaft erstellen. | Abbildung Nutzereigenschaft |

Tabelle User Properties

### 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. | Abbildung Benachrichtigung Text |
| 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 | Abbildung Ziel-E-Mail Angabe |
| Hier kann man wieder wie vorhin einen eigenen Zeitplan machen oder einfach auf ‘’Jetzt’’ lassen. | Abbildung Zeitplan Firebase |
| Nachdem die Benachrichtigung abgeschickt wurde wird sie angezeigt. Diese Nachricht bekommt also nur der User mit dieser E-Mail. | Abbildung Benachrichtigung auf dem Handy |

Tabelle Spezifizierte Benachrichtigung

# Word Press Rest API

## 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 |
| Abbildung WordPress REST API (Version 2) | Abbildung REST API – Filter Fields |

Tabelle Wordpress Rest API

## Versuche

### 1.Versuch YouTube Tutorial

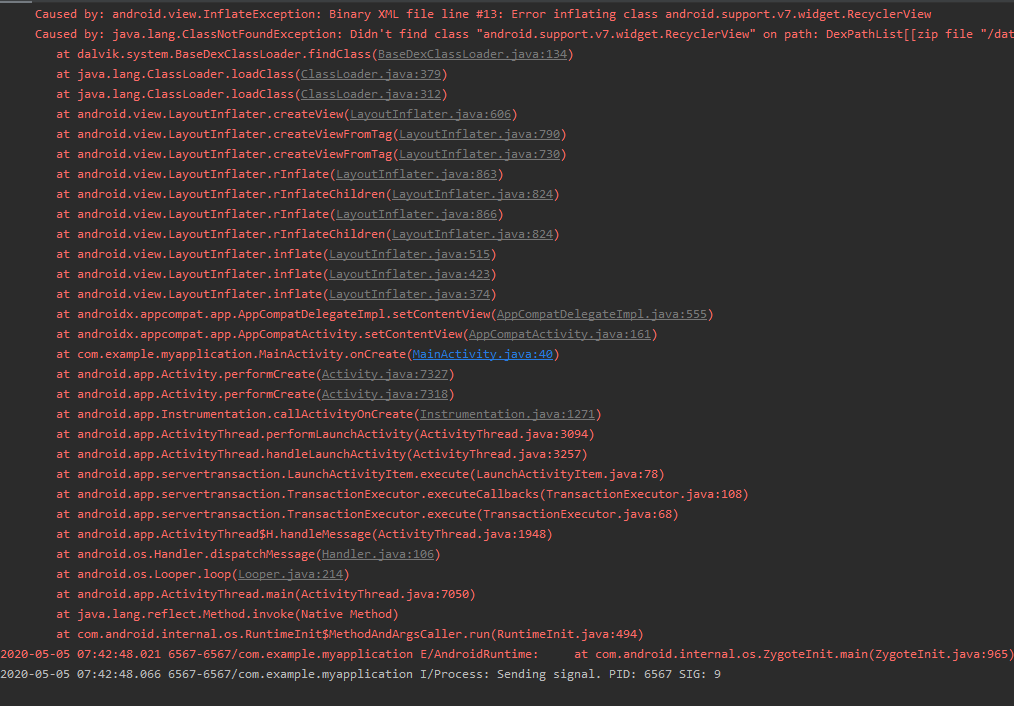
Beim erste Versuch probierten wir mit einem YouTube [Tutorial](https://www.youtube.com/watch?v=W0zTZK3vCeg) über WP Rest API in Android Studio. Das Tutorial scheiterte nach mehreren Build-Versuchen. Das Video ging recht lange und vieles konnte nicht erklärt werden im Video. Ich habe viel abgeschrieben und icht viel mitgenommen

Abbildung Logcat

### 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.

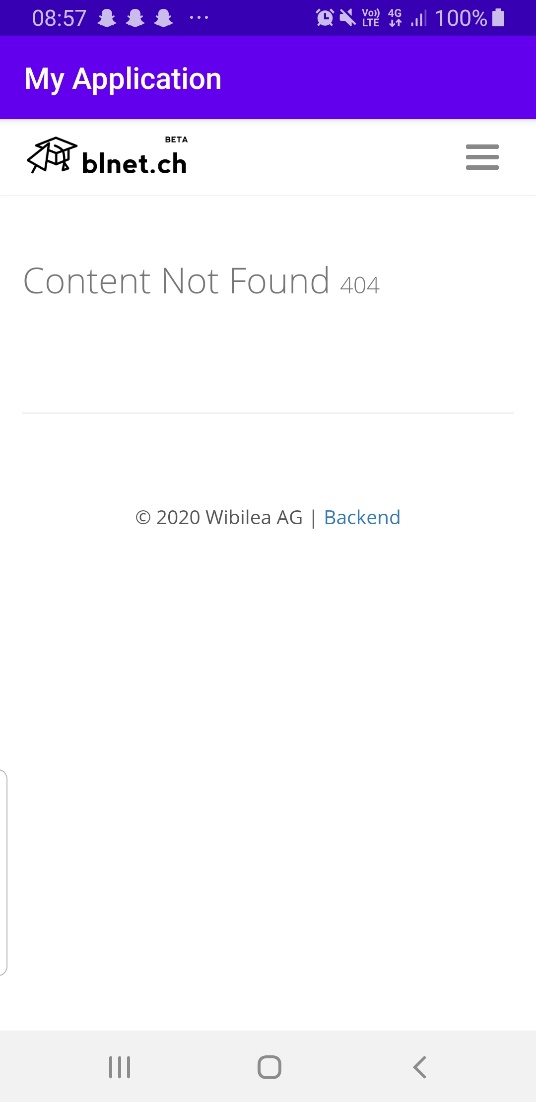


Abbildung WebView

### 3.Versuch Internet Tutorial

Als drittes haben wir probiert über ein Internet [Tutorial](https://www.simplifiedcoding.net/wordpress-to-android-app-tutorial/) ausprobiert. Zu Beginn funktionierte alles ganz gut die wurde gebildet. Allerdings stürzte die App immer ab wenn man den Post aufrufen wollte.

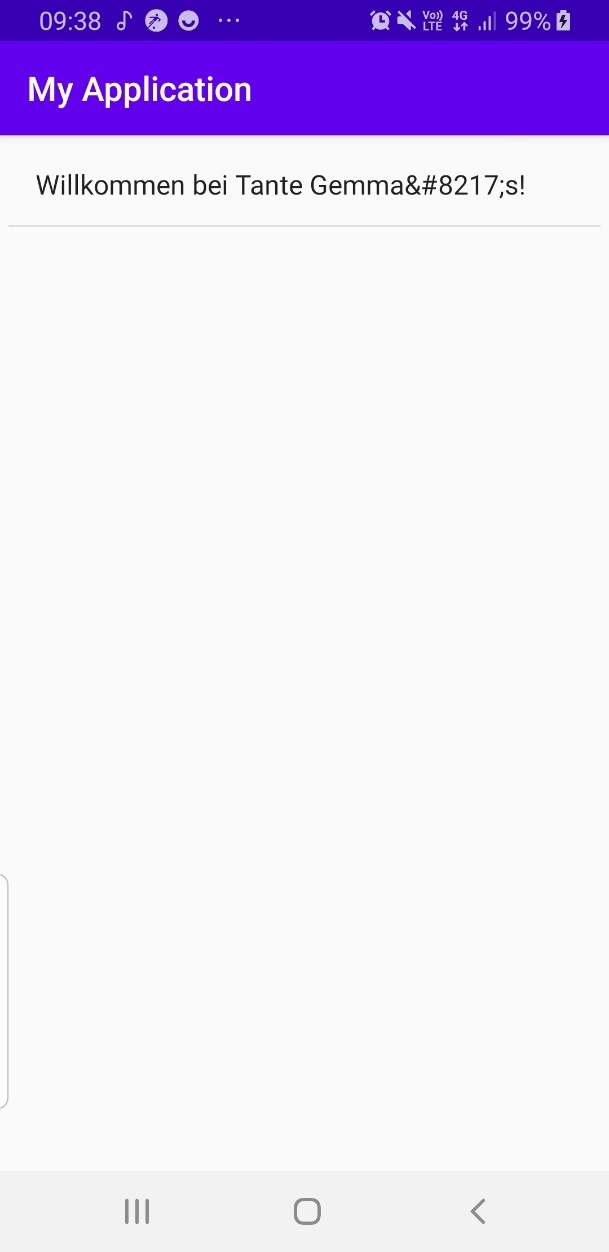


Abbildung Wordpress in Android Studio

# Zeitplan



# Arbeitsjournal

### 08.01.2019

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tätigkeit** | **Erledigte Arbeiten / Erreichte Ziele** | **Erfolge & neu gelernt** | **Aufgetretene Probleme** | **Genutzte Quellen** |
| Informieren | **Informiert/**  **Android Studio** | **Android Studio** | **-** | [**google.com**](https://developer.android.com/studio/?gclid=EAIaIQobChMI9tDXouKA5wIVluF3Ch16kgZlEAAYASAAEgKutPD_BwE) |

### 13.01.2019

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tätigkeit** | **Erledigte Arbeiten / Erreichte Ziele** | **Erfolge & neu gelernt** | **Aufgetretene Probleme** | **Genutzte Quellen** |
| Android Studio |  |  |  |  |
| Java | **Tutorials über Java** | **Java kennengelernt** | **---------------------------------** | [**https://www.w3schools.com/JAVA/default.asp**](https://www.w3schools.com/JAVA/default.asp) |

### 27.01.2020

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tätigkeit** | **Erledigte Arbeiten / Erreichte Ziele** | **Erfolge & neu gelernt** | **Aufgetretene Probleme** | **Genutzte Quellen** |
| Login Button |  |  | **Fragment Verlinkung ging nicht** | **stackoverflow, android studio help, YouTube** |

### 28.01.2020

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tätigkeit** | **Erledigte Arbeiten / Erreichte Ziele** | **Erfolge & neu gelernt** | **Aufgetretene Probleme** | **Genutzte Quellen** |
| 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** |

### 29.01.2020

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tätigkeit** | **Erledigte Arbeiten / Erreichte Ziele** | **Erfolge & neu gelernt** | **Aufgetretene Probleme** | **Genutzte Quellen** |
| Dokumentieren | Dokumentation erweitert | **-** | **-** | **Word/Internet** |

### 30.01.2020

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tätigkeit** | **Erledigte Arbeiten / Erreichte Ziele** | **Erfolge & neu gelernt** | **Aufgetretene Probleme** | **Genutzte Quellen** |
| 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** |

### 10.02.2020

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

### 26.02.2020

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tätigkeit** | **Erledigte Arbeiten / Erreichte Ziele** | **Erfolge & neu gelernt** | **Aufgetretene Probleme** | **Genutzte Quellen** |
| Mit Codecademy den Java Kurs begonnen | **Java**  **(ging den ganzen Tag)** | **Erste Einblicke in Java** | **-** | [**https://www.codecademy.com/**](https://www.codecademy.com/) |

### 02.03.2020

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tätigkeit** | **Erledigte Arbeiten / Erreichte Ziele** | **Erfolge & neu gelernt** | **Aufgetretene Probleme** | **Genutzte Quellen** |
| Codecademy Java Kurs | **Java**  **(ging den ganzen Tag)** | **System Print out, variables etc.** | **-** | [**https://www.codecademy.com/**](https://www.codecademy.com/) |

### 03.03.2020

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tätigkeit** | **Erledigte Arbeiten / Erreichte Ziele** | **Erfolge & neu gelernt** | **Aufgetretene Probleme** | **Genutzte Quellen** |
| Codecademy Java Kurs | **30% Abeeraam**  **30% Ian** | **Classes, objects** | **-** | [**https://www.codecademy.com/**](https://www.codecademy.com/) |
| Checkliste App | **Checkliste funktionsfähig** | **Fragment ordnungsgemäss erstellen** | **Daten werden nicht gespeichert (z.B nach Neustart von App)** | **-** |

### 04.03.2020

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tätigkeit** | **Erledigte Arbeiten / Erreichte Ziele** | **Erfolge & neu gelernt** | **Aufgetretene Probleme** | **Genutzte Quellen** |
| Codecademy Java Kurs | **50% Abeeraam** | **Conditional operators** | **-** | **-** |
| Dokumentieren |  |  |  |  |

### 09.03.2020

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tätigkeit** | **Erledigte Arbeiten / Erreichte Ziele** | **Erfolge & neu gelernt** | **Aufgetretene Probleme** | **Genutzte Quellen** |
| Codecademy Java Kurs | **92% Abeeraam**  **38% Ian** | **loops, strings** | **-** | **-** |

### 25.03.2020

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tätigkeit** | **Erledigte Arbeiten / Erreichte Ziele** | **Erfolge & neu gelernt** | **Aufgetretene Probleme** | **Genutzte Quellen** |
| Codecademy Java Kurs | **Abeeraam/Ian fertig** | **-** | **-** | **-** |
| Webserver + SQL-Server eingerichtet | **eingerichtet** | **Webserver + Datenbank erstellen** | **PHP-Code funktioniert noch nicht** | [**https://www.YouTube.com/watch?v=JQXfIidfFMo&list=PLe60o7ed8E-TztoF2K3y4VdDgT6APZ0ka&index=3**](https://www.youtube.com/watch?v=JQXfIidfFMo&list=PLe60o7ed8E-TztoF2K3y4VdDgT6APZ0ka&index=3) |
| Dokumentation Java | - | - | - | - |

### 31.03.2020

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

### 06.04.2020

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

### 07.04.2020

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tätigkeit** | **Erledigte Arbeiten / Erreichte Ziele** | **Erfolge & neu gelernt** | **Aufgetretene Probleme** | **Genutzte Quellen** |
| Checkbox Status speichern | **Werte werden nach 12 Stunden gelöscht** | **Wie SharedPreferences funktionieren** |  | **stackoverflow**  **Android Developer Dokumentation** |

### 08.04.2020

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tätigkeit** | **Erledigte Arbeiten / Erreichte Ziele** | **Erfolge & neu gelernt** | **Aufgetretene Probleme** | **Genutzte Quellen** |
| Login speichern | **Login Speicherung** | **SharedPreferences in anderen Activities aufrufen** |  | **stackoverflow** |

### 14.04.2020

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

### 15.04.2020

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tätigkeit** | **Erledigte Arbeiten / Erreichte Ziele** | **Erfolge & neu gelernt** | **Aufgetretene Probleme** | **Genutzte Quellen** |
| Informiert und dokumentiert über Firebase | **Informationen über Firebase.** | **Was ist Firebase?**  **Für was braucht man Firebase?** | **-** | **google.com** |
| Reminder programmiert | **Einen Reminder programmiert wo man sagen kann wann man Benachrichtigt werden soll.** | **Nachricht konfigurieren.**  **An einen Button und einer Uhr Funktionen geben.** | **Funktionierte zu Beginn ganz gut allerdings funktionierte der ganze Zyklus ein Tag nachher nicht mehr. Auch wenn ich nicht geändert habe.** | **YouTube.com**  **Google.com** |
| Layout von Notenformular | **Layout fertiggestellt** | **Dropdownmenü erstellen + Radio Buttons** |  | [**https://www.YouTube.com/watch?v=on\_OrrX7Nw4**](https://www.youtube.com/watch?v=on_OrrX7Nw4) |

### 16.04.2020

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tätigkeit** | **Erledigte Arbeiten / Erreichte Ziele** | **Erfolge & neu gelernt** | **Aufgetretene Probleme** | **Genutzte Quellen** |
| Notenformular | **Validierung** | **Verschiedene Möglichkeiten für Validierung** | **Absturz der App** | **stackoverflow.com** |

### 17.04.2020

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tätigkeit** | **Erledigte Arbeiten / Erreichte Ziele** | **Erfolge & neu gelernt** | **Aufgetretene Probleme** | **Genutzte Quellen** |
| Notenformular | **Validierung fertig** | **Wie Formulare auf Android validiert werden** |  |  |

### 27.04.2020

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

### 28.04.2020

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

### 29.04.2020

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

### 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** | **-** | **https://github.com/dwyl/learn-to-send-email-via-google-script-html-no-server** |
| Push Nachrichten | **Übrige Ämtlis & usw. in Firebase eingetragen** | **-** | **-** | **-** |

### 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** | **-** |

### 06.05.2020

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

### 11.05.2020

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tätigkeit** | **Erledigte Arbeiten / Erreichte Ziele** | **Erfolge & neu gelernt** | **Aufgetretene Probleme** | **Genutzte Quellen** |
| blnet App Webinterface | **Usertabelle wird in Interface angezeigt + Formularfelder für Management einfügt** | **-** | **PHP-Fehler** | **-** |
| Links (Wichtige Infos) | **Links funktionieren jetzt** | **Die Links funktionieren jetzt. Nachdem ich Probleme hatte mit den Link sah ich den Fehler und er konnte relativ schnell behoben werden** |  | **.** |
| Kalender (Portrait -->Landscape) | **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.** | **If und else bei einem WebView,**  **Info-Meldung programmieren,** | **Viele Fehlversuche,** | **Stackoverflow** |

### 12.05.2020

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

### 13.05.2020

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tätigkeit** | **Erledigte Arbeiten / Erreichte Ziele** | **Erfolge & neu gelernt** | **Aufgetretene Probleme** | **Genutzte Quellen** |
| Putztag-Checkliste fertig | **Putztag-Checkliste fertig programmiert und erstellt.** |  | **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.** |  |

### 18.05.2020

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

### 19.05.2020

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

### 20.05.2020

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

# Testprotokoll blnet App

**Release-Management**

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

Tabelle Release Management

# 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 Ausgangslage

## Testgegenstand

| **Testgegenstand** | **Beschreibung** |
| --- | --- |
| *blnet App Version 1.0.X* |  |

Tabelle Testgegenstand

# Testfälle

## Testfall 1 App installieren

|  |  |
| --- | --- |
| Beschreibung | App installieren |
| Mindestanforderungen | Internetverbindung, 10MB freier Speicher, Android 5.0 |
| Testschritte | 1. Öffnen die Seite <app.blnet.ch> im Browser 2. Scrollen sie hinunter bis Sie den Abschnitt herunterladen sehen. 3. Drücken Sie auf die Taste «herunterladen» 4. Wählen Sie die neuste Version der App aus. (die oberste). 5. Drücken Sie auf Assets 6. Drücken Sie auf die jetzt erschienene .apk Datei 7. Installieren Sie die App «auf Öffnen klicken wenn Sie heruntergeladen wurde.» |
| Erwartetes Ergebnis | Die App startet und Sie sehen die Startseite. |
| Testresultat |  |

Tabelle Testfall 1 App installieren

## Testfall 2 – Anmelden und Registrieren

|  |  |
| --- | --- |
| Beschreibung | Anmelden und Registrieren |
| Mindestanforderungen | Internetverbindung, 10MB freier Speicher, Android 5.0 |
| Testschritte | 1. Die App starten 2. Oben rechts auf das Personenzeichen drücken 3. Im Menü auf «Registrieren» drücken 4. Folgende Daten eingeben und auf Registrieren klicken:   Vorname: Sara  Nachname: Stoob  E-Mail: [sara.stoob@gmail.com](mailto:sara.stoob@gmail.com)  Passwort: Bananen.123   1. 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». |
| Erwartetes Ergebnis | Eine Toast-Meldung erscheint. Sie sieht etwa so aus: »Willkommen zurück, Sara Stoob» |
| Testresultat |  |

Tabelle Testfall 2 Anmelden und Registrieren

## Testfall 3 Passwort zurücksetzen

|  |  |
| --- | --- |
| Beschreibung | Passwort zurücksetzen |
| Mindestanforderungen | Internetverbindung, 10MB freier Speicher, Android 5.0 |
| Testschritte | 1. App starten 2. Oben rechts auf das Personenzeichen drücken 3. Im Menü auf «Passwort zurücksetzen» drücken 4. Jetzt folgende Daten eingeben:   E-Mail: sara.stoob@blnet.ch  Altes Passwort: Bananen.123  Neues Passwort + bestätigen: Gurken.456   1. Drücken Sie dann auf «Zurücksetzen» 2. 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». |
| Erwartetes Ergebnis | Eine Toast-Meldung erscheint nach dem Drücken auf «Zurücksetzen». Sie sieht etwa so aus: »Das Passwort wurde erfolgreich zurückgesetzt»  Eine Toast-Meldung erscheint nach der Anmeldung. Sie sieht etwa so aus: »Willkommen zurück, Sara Stoob» |
| Testresultat |  |

Tabelle Testfall 3 Passwort zurücksetzen

## Testfall 4 Notenformular

|  |  |
| --- | --- |
| Beschreibung | Notenformular |
| Mindestanforderungen | Internetverbindung, 10MB freier Speicher, Android 5.0 |
| Testschritte | 1. Melden Sie sich zuerst mit diesem Account an:   [debug@blnet.ch](mailto:debug@blnet.ch)  Bananen.123   1. Öffnen Sie dann das Notenformular über die Navigation 2. 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   1. Drücken Sie dann auf «Absenden» 2. Öffnen Sie dann das Notenformular über die Navigation 3. Drücken Sie auf «Notentabelle» (zu unterst) 4. Wechseln Sie auf das Tabellenblatt «Formularantworten» 5. Überprüfen Sie, ob ihr Eintrag vorhanden ist. (Wenn nicht, probieren sie es neu zu laden) |
| Erwartetes Ergebnis | Eine Toast-Meldung erscheint nach dem Drücken auf «Absenden». Sie sieht etwa so aus: »Das Formular wurde erfolgreich abgeschickt»  Eintrag ist in der Tabelle vorhanden. |
| Testresultat |  |

Tabelle Testfall 4 Notenformular

## Testfall 5 Kalender

|  |  |
| --- | --- |
| Beschreibung | Kalender |
| Mindestanforderungen | Internetverbindung, 10MB freier Speicher, Android 5.0 |
| Testschritte | 1. Starten Sie die App, 2. Öffnen sie in der Navigation die Seite «Kalender» im Portraitmodus 3. Fehlermeldung erscheint 4. Drehen Sie ihr Handy zur Landscapeansicht 5. Schauen ob der Kalender in der Landscapeansicht geladen wird |
| 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 Testfall 5 Kalender

## Testfall 6 Checklisten

|  |  |
| --- | --- |
| Beschreibung | Die beiden Checklisten und das Umschalten zwischen Putztag und Ämtli-Checkliste funktioniert |
| Mindestanforderungen | Internetverbindung, 10MB freier Speicher, Android 5.0 |
| Testschritte | 1. Starten Sie die App, 2. Öffnen sie in der Navigation die Seite «Checkliste» 3. Haken setzen bei:   Paketdienst und Kühlschrank checken.   1. Schliessen sie die App im Verlauf. 2. Öffnen Sie die App und gehen sie wieder zu Checkliste. 3. Überprüfen Sie ob die eingegebenen Werte aktiv geblieben sind. 4. Aktivieren sie die Putztag Checkliste 5. Haken setzen bei:   Boden Reinigung aller Pausenplatz und Aschenbecher leeren.   1. Schliessen sie die App im Verlauf. 2. Öffnen Sie die App und gehen sie diesmal zur Putztagcheckliste. 3. Überprüfen Sie ob die eingegebenen Werte aktiv geblieben sind. |
| 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 Testfall 6 Checklisten

## Testfall 6 Links

|  |  |
| --- | --- |
| Beschreibung | Links funktionieren |
| Mindestanforderungen | Internetverbindung, 10MB freier Speicher, Android 5.0  Firefox Browser |
| Testschritte | 1. Starten Sie die App, 2. Öffnen sie in der Navigation die Seite «Wichtige Infos» 3. Klicken Sie den Link «RE\_IT\_Richtlinien\_Lernende» 4. Überprüfen sie ob die IT Richtlinien aufgeschlagen werden. 5. Klicken Sie alle restlichen Links und überprüfen ob jeder Link das richtige öffnet.   «Neuster Leitfaden auf dem Wibilea Extranet»> <https://www.wibilea.ch/extranet/leitfaden/>  «Beispiel Wibilea»> <https://www.blnet.ch/wp-content/uploads/2020/03/VO_Abwesenheitsmeldung_beispiel_blnet.jpg>  «Beispiel Syntegon»> <https://www.blnet.ch/wp-content/uploads/2020/03/Abwesenheitsmeldung_Syntegon_beispiel.jpg> |
| Erwartetes Ergebnis | Alle Links funktionieren |
| Testresultat |  |

Tabelle Testfall 6 Links

# Testergebnis

| Tester |  |
| --- | --- |
| Datum Testdurchführung |  |
| Anzahl Fehler |  |
| Fehlerbeschreibung |  |

Tabelle Testergebnis

# 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 | Firebase 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 |

# Abbildungsverzeichnis

[Abbildung 1 Android Studio Download 6](#_Toc40879718)

[Abbildung 2 Android Studios Lizenzbedingungen 6](#_Toc40879719)

[Abbildung 3 Android Studios Download 6](#_Toc40879720)

[Abbildung 4 Willkommensfenster Android Studio 6](#_Toc40879721)

[Abbildung 5 Installation Komponenten 6](#_Toc40879722)

[Abbildung 6 Auswahl Speicherort 7](#_Toc40879723)

[Abbildung 7 Shortcut Auswahl 7](#_Toc40879724)

[Abbildung 8 Durchführung Installation 7](#_Toc40879725)

[Abbildung 9 Abgeschlossene Installation 7](#_Toc40879726)

[Abbildung 10 Neues Projekt Android Studio 13](#_Toc40879727)

[Abbildung 11 Projekt auswählen 13](#_Toc40879728)

[Abbildung 12 Konfiguration des Projekts 13](#_Toc40879729)

[Abbildung 13 Projekt wird erstellt 14](#_Toc40879730)

[Abbildung 14 Android Studio Aufbau 15](#_Toc40879731)

[Abbildung 15 Bild auf dem Desktop 16](#_Toc40879732)

[Abbildung 16 Ordner mipmap 16](#_Toc40879733)

[Abbildung 17 Design erstellen 16](#_Toc40879734)

[Abbildung 18 Bild nach Wahl auswählen 17](#_Toc40879735)

[Abbildung 19 Bild in der APP 17](file:///N:\Projekte\Projekt_RaA_HiI\blapp_projekt_hii_raa.docx#_Toc40879736)

[Abbildung 20 Zentrierung (Höhe) 17](file:///N:\Projekte\Projekt_RaA_HiI\blapp_projekt_hii_raa.docx#_Toc40879737)

[Abbildung 21 Zentrierung (Breite) 17](file:///N:\Projekte\Projekt_RaA_HiI\blapp_projekt_hii_raa.docx#_Toc40879738)

[Abbildung 22 Debug Emulator 18](file:///N:\Projekte\Projekt_RaA_HiI\blapp_projekt_hii_raa.docx#_Toc40879739)

[Abbildung 23 Zentriertes Bild in der App 18](file:///N:\Projekte\Projekt_RaA_HiI\blapp_projekt_hii_raa.docx#_Toc40879740)

[Abbildung 24 Text im strings.XML 18](#_Toc40879741)

[Abbildung 25 Funktiongebung 18](file:///N:\Projekte\Projekt_RaA_HiI\blapp_projekt_hii_raa.docx#_Toc40879742)

[Abbildung 26 XML String 19](#_Toc40879743)

[Abbildung 27 Kontext Menü 20](#_Toc40879744)

[Abbildung 28 Auswahl Bundle/APK 20](#_Toc40879745)

[Abbildung 29 Key-Store 20](#_Toc40879746)

[Abbildung 30 Auswahl Debug/Release 21](#_Toc40879747)

[Abbildung 31 Build Fortschritte 21](#_Toc40879748)

[Abbildung 32 Zielordner 21](#_Toc40879749)

[Abbildung 33 Activity erstellen 22](#_Toc40879750)

[Abbildung 34Activity erstellen 22](#_Toc40879751)

[Abbildung 35 - XML-Datei für Fragment erstellen 24](#_Toc40879752)

[Abbildung 36 Fragment erstellen 25](#_Toc40879753)

[Abbildung 37 Java Code 26](#_Toc40879754)

[Abbildung 38 Fragment 28](file:///N:\Projekte\Projekt_RaA_HiI\blapp_projekt_hii_raa.docx#_Toc40879755)

[Abbildung 39 USB-Debugging Weitere Einstellungen 29](#_Toc40879756)

[Abbildung 40 USB-Debugging Entwickleroptionen 30](#_Toc40879757)

[Abbildung 41 USB-Debugging einschalten 31](#_Toc40879758)

[Abbildung 42 USB-Debugging von Android Studio 31](#_Toc40879759)

[Abbildung 43 App installieren 32](#_Toc40879760)

[Abbildung 44 Die App auf dem Handy 32](#_Toc40879761)

[Abbildung 45 Auslöser Button 35](#_Toc40879762)

[Abbildung 46 Definition des Buttons im string.XML 35](file:///N:\Projekte\Projekt_RaA_HiI\blapp_projekt_hii_raa.docx#_Toc40879763)

[Abbildung 47 onClick Statement Button 35](file:///N:\Projekte\Projekt_RaA_HiI\blapp_projekt_hii_raa.docx#_Toc40879764)

[Abbildung 48 Button Definition Java 35](#_Toc40879765)

[Abbildung 49 findElementById 36](#_Toc40879766)

[Abbildung 50 Ausgabe des Toasts 36](#_Toc40879767)

[Abbildung 51 Ausgabe nach dem Click 36](#_Toc40879768)

[Abbildung 52 Layout der Seite 37](file:///N:\Projekte\Projekt_RaA_HiI\blapp_projekt_hii_raa.docx#_Toc40879769)

[Abbildung 53 Manifest Dokument 37](#_Toc40879770)

[Abbildung 54 Eigene Java-datei 38](#_Toc40879771)

[Abbildung 55 Konfiguration Benachrichtigungen 39](#_Toc40879772)

[Abbildung 56 Endresultat 39](file:///N:\Projekte\Projekt_RaA_HiI\blapp_projekt_hii_raa.docx#_Toc40879773)

[Abbildung 57 MainActivity Datei 40](#_Toc40879774)

[Abbildung 58 AndroidManifest Datei 40](#_Toc40879775)

[Abbildung 59 MainActivity(Java) Datei 41](#_Toc40879776)

[Abbildung 60 Externe Java Datei 41](#_Toc40879777)

[Abbildung 61 BL App Login Fenster 42](file:///N:\Projekte\Projekt_RaA_HiI\blapp_projekt_hii_raa.docx#_Toc40879778)

[Abbildung 62 phpMyAdmin WebGUI 45](#_Toc40879779)

[Abbildung 63 Datenbankeinträge 45](#_Toc40879780)

[Abbildung 64 Checklisten 51](file:///N:\Projekte\Projekt_RaA_HiI\blapp_projekt_hii_raa.docx#_Toc40879781)

[Abbildung 65 Variabel Deklarierung 51](#_Toc40879782)

[Abbildung 66 Zuweisung der Buttons 52](#_Toc40879783)

[Abbildung 67 Checkliste verstecken 52](#_Toc40879784)

[Abbildung 68 Button-Konfiguration 53](#_Toc40879785)

[Abbildung 69 Button-Konfiguration 53](#_Toc40879786)

[Abbildung 70 Checklisten 53](file:///N:\Projekte\Projekt_RaA_HiI\blapp_projekt_hii_raa.docx#_Toc40879787)

[Abbildung 71 Notenformular 54](#_Toc40879788)

[Abbildung 72 Android Studios Tools 62](#_Toc40879789)

[Abbildung 73 Cloud Messaging Android Studio 62](#_Toc40879790)

[Abbildung 74 Firebase im Browser 63](#_Toc40879791)

[Abbildung 75 Firebase Projektname 63](#_Toc40879792)

[Abbildung 76 Firebase Lizenzbedingungen 63](#_Toc40879793)

[Abbildung 77 Android Studio und Firebase verbinden 63](#_Toc40879794)

[Abbildung 78 FCM einfügen 63](#_Toc40879795)

[Abbildung 79 FCM akzeptieren 64](#_Toc40879796)

[Abbildung 80 Benachrichtigung erstellen 65](#_Toc40879797)

[Abbildung 81 Ziel Definition 65](#_Toc40879798)

[Abbildung 82 Zeitplan Firebase 65](#_Toc40879799)

[Abbildung 83 Nachricht Überprüfung 65](#_Toc40879800)

[Abbildung 84 Nutzereigenschaft 66](#_Toc40879801)

[Abbildung 85 Benachrichtigung Text 66](#_Toc40879802)

[Abbildung 86 Ziel-E-Mail Angabe 66](#_Toc40879803)

[Abbildung 87 Zeitplan Firebase 66](file:///N:\Projekte\Projekt_RaA_HiI\blapp_projekt_hii_raa.docx#_Toc40879804)

[Abbildung 88 Benachrichtigung auf dem Handy 67](file:///N:\Projekte\Projekt_RaA_HiI\blapp_projekt_hii_raa.docx#_Toc40879805)

[Abbildung 89 WordPress REST API (Version 2) 68](#_Toc40879806)

[Abbildung 90 REST API – Filter Fields 68](#_Toc40879807)

[Abbildung 91 Logcat 69](#_Toc40879808)

[Abbildung 92 WebView 70](#_Toc40879809)

[Abbildung 93 Wordpress in Android Studio 71](#_Toc40879810)

# Tabellenverzeichnis

[Tabelle 1 Android Studio Installation 7](#_Toc40879811)

[Tabelle 2 Datentypen 8](#_Toc40879812)

[Tabelle 3 Loops 11](#_Toc40879813)

[Tabelle 4 Operatoren 12](#_Toc40879814)

[Tabelle 5 Neues Projekt Android Studio 14](#_Toc40879815)

[Tabelle 6 Bilder in Android Studios 17](#_Toc40879816)

[Tabelle 7 Bilder ausrichten in Android Studios 18](#_Toc40879817)

[Tabelle 8 APK 21](#_Toc40879818)

[Tabelle 9 Activity erstellen 22](#_Toc40879819)

[Tabelle 10 USB-Debugging 32](#_Toc40879820)

[Tabelle 11 Toasts 36](#_Toc40879821)

[Tabelle 12 Push Knopf 39](#_Toc40879822)

[Tabelle 13 Reminder erstellen 41](#_Toc40879823)

[Tabelle 14 Zweite Checkliste 53](#_Toc40879824)

[Tabelle 15 Firebase verbinden bin 64](#_Toc40879825)

[Tabelle 16 Cloud Messaging 66](#_Toc40879826)

[Tabelle 17 User Properties 66](#_Toc40879827)

[Tabelle 18 Spezifizierte Benachrichtigung 67](#_Toc40879828)

[Tabelle 19 Wordpress Rest API 68](#_Toc40879829)

[Tabelle 20 Release Management 90](#_Toc40879830)

[Tabelle 21 Ausgangslage 91](#_Toc40879831)

[Tabelle 22 Testgegenstand 91](#_Toc40879832)

[Tabelle 23 Testfall 1 App installieren 92](#_Toc40879833)

[Tabelle 24 Testfall 2 Anmelden und Registrieren 93](#_Toc40879834)

[Tabelle 25 Testfall 3 Passwort zurücksetzen 94](#_Toc40879835)

[Tabelle 26 Testfall 4 Notenformular 95](#_Toc40879836)

[Tabelle 27 Testfall 5 Kalender 96](#_Toc40879837)

[Tabelle 28 Testfall 6 Checklisten 97](#_Toc40879838)

[Tabelle 29 Testfall 6 Links 98](#_Toc40879839)

[Tabelle 30 Testergebnis 99](#_Toc40879840)

# 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