Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Bericht

Praxissemester bei portrix.net GmbH

Tim Staats

Fakultät Technik und Informatik
Department Informations- und Elektrotechnik

Faculty of Engineering and Computer Science
Department of Information and Electrical Engineering

Inhaltsverzeichnis

1 Einleitung

In diesem Bericht geht es um den Inhalt meines Praxissemesters vom 01.08.2018 bis 31.12.2018. Ziel dieses Praktikums war es, Einblick in die Abläufe eines modernen Software-Unternehmens zu erhalten und die an der HAW gelernten theoretischen Inhalte durch praktische Erfahrungen zu ergänzen. Die Kompetenzen der Firma portrix.net GmbH bieten dem Kunden maßgeschneiderte ganzheitliche Softwarelösungen, vom zuverlässigem Steuerungssystem bis zur leistungsfähigen Plattform, sowohl für Start-ups als auch für internationale Konzerne. Mein Einsatzgebiet in der Android Entwicklung, hat mir erlaubt erste Eindrücke in der Konzeptionierung, Entwicklung und Pflege der Software zu gewinnen. Voraussetzung für ein erfolgreiches mitarbeiten war zudem die Einarbeitung in die Programmiersprache Java, welche in der Android Programmierung verwendet wird.

2 Einarbeitung

Für den Praktikumsbeginn habe ich mich in Absprache mit meinem Portrix Betreuer Knud Müller auf folgende Themen vorbereitet:

- Linux (Kubuntu)
- Html5/CSS
- Json
- Android Studio

Während meiner Portrix-Zeit haben sich die Eindrücke vertieft und wurden um folgende Aspekte erweitert:

- Java
- Git
- JavaScript

Die generelle Informationsbeschaffung erfolgte in den meisten fällen über das Internet. Besonders hilfreich waren dabei:

- stackoverflow
- Android Developers
- Udemy (Online Kursplattform)
- CodingInFlow (Youtube Channel)
- CodingWithMitch (Youtube Channel)

2.1 Linux (Kubuntu)

Die Firma portrix.net benutzt auf ihren Firmenrechnern Kubuntu als Betriebssystem. Es war also notwendig ein Grundwissen an Terminal-Befehlen (Bash) aufzubauen. Dies wurde als Vorbereitung vor dem Praktikum durch das Youtube-Tutorial "The Complete Linux Course: Beginner to Power User!"[1] geleistet und war besonders in Verbindung mit der Nutzung von GitLab sehr vorteilhaft.

2.2 Html5/CSS

"HTML ist eine Abkürzung für Hypertext Markup Language. Man versteht darunter eine Computersprache, mit deren Hilfe man Webseiten im Internet erstellen kann. HTML ist keine Programmiersprache im eigentlich Sinn, sondern vielmehr eine sogenannte Auszeichnungssprache."[2] Ebenfalls als Vorbereitung auf das Praktikum kennengelernt, kam ich mit HTML5 während der Tätigkeit bei portrix.net nicht direkt in Kontakt. Durch die Ähnlichkeit zu XML war jedoch der Einstieg in das Arbeiten mit den .xml Dateien von Android Studio (AS) sehr viel leichter. Auch das Konzept eines Cascading Style Sheet (CSS) lässt sich bei AS wiederfinden. Dadurch sind die Apps besser wartbar und es ist leichter das Layout einheitlich zu gestalten.

2.3 Json

"Die JavaScript Object Notation, kurz JSON, ist ein kompaktes Datenformat in einer einfach lesbaren Textform zum Zweck des Datenaustauschs zwischen Anwendungen."[3] Die hauseigenen Apps der Firma portrix.net (H2.LIVE und LFI) beziehen aktuelle Daten von einem Server. Diese Daten werden im JSON-Format in den Apps ausgewertet und verarbeitet. Der sichere Umgang mit JSON-Objekten und JSON-Arrays ist daher sehr wichtig. Dies wurde während des Praxissemesters erreicht.

[1] https://www.youtube.com/watch?v=wBp0Rb-ZJak
[2] https://www.as-computer.de/wissen/unterschiede-html-und-xml/
[3] https://de.wikipedia.org/wiki/JavaScript_Object_Notation/

2.4 Android Studio

Als Einstieg in das Arbeiten mit Android Studio (AS) wurde der Udemy Kurs "Der Komplette Android 8 Entwickler Kurs - Erstelle 20+ Apps"[4] absolviert. AS bietet eine Entwicklungsumgebung die sowohl Java als auch Kotlin unterstützt. Da die Android Abteilung bei portrix.net ausschließlich mit Java programmiert, bestand kein Bedarf Kotlin zu lernen. Ein Android Projekt besteht aus vielen Komponenten, welche als Resultat eine Application (App) generieren. Eine einfache *Hello World* App besteht zum Beispiel aus der MainActivity.java welche die programmatische Logik enthält und der dazugehörigen activity_main.xml die das graphische Layout definiert. Ein weitere wichtiger Bestandteil von AS ist der integrierte Emulator der nach den spezifischen Anforderungen des Projekts gestaltet werden kann und zu Testzwecken als ein simuliertes Mobiltelefon auf dem PC fungiert. In der professionellen Software Entwicklung ist es allerdings unerlässlich auf einem realen Testdevice (Smartphone oder Tablet) zu testen, da sich Abweichungen zwischen Emulator und Smartphone kaum vermeiden lassen und der haptische Aspekt nur auf dem Testdevice getestet werden kann. Die Möglichkeit AS Projekte mit Versionskontrollsystemen (VCS) zu nutzen spielt ebenfalls eine große Rolle, da mehrere Entwickler gleichzeitig an den Projekten arbeiten, diese gut aufteilen und dokumentieren können. Bei portrix.net wird mit GitLab gearbeitet, worauf im weiteren Verlauf eingegangen wird.

2.5 Java

Java als Objekt orientierte Programmiersprache (OOP) ist das Fundament für das Arbeiten mit Android Studio bei portrix.net. Die Grundlagen der Objekt orientierten Programmierung die an der HAW vermittelt wurden, konnten dementsprechend erweitert und vertieft werden. Dies geschah zu beginn hauptsächlich durch Kurse der Online-Plattform Udemy zum Thema OOP, wie zum Beispiel "Java leicht gemacht - Der umfassende Java Einsteigerkurs A-Z"[5]. Mit dem dadurch erlangtem Basiswissen, war es möglich bei der Weiterentwicklung der Firmen Apps beteiligt zu sein.

2.6 GitLab

Wie zuvor erwähnt wird GitLab als VCS-Tool verwendet. "GitLab ist eine Webanwendung zur Versionsverwaltung für Softwareprojekte auf Basis von Git."[6] Um Git zu nutzen speichert

[4] https://www.udemy.com/der-komplette-android-8-entwickler-kurs/
[5] https://www.udemy.com/programmieren-lernen-mit-java-ein-kurs-fur-einsteiger/
[6] https://de.wikipedia.org/wiki/GitLab

man sein Inkrement auf dem lokalen PC mit dem *Commit*-Befehl und der *Commit*-Message welche die Änderung beschreibt. Anschließend werden alle gewünschten Commits mit dem push-Befehl ins Online-Repository gespeichert. Dadurch können mehrere Personen gleichzeitig an einem Projekt arbeiten. Es bietet auch die Möglichkeit neue Features in einer parallel laufenden Verzweigung (*Branch*) zu entwickeln und so die Funktionalität der bisherigen Version zu erhalten.

2.7 JavaScript

"JavaScript (kurz JS) ist eine Skriptsprache, die ursprünglich 1995 von Netscape für dynamisches HTML in Webbrowsern entwickelt wurde, um Benutzerinteraktionen auszuwerten, Inhalte zu verändern, nachzuladen oder zu generieren und so die Möglichkeiten von HTML und CSS zu erweitern."[7] Während des Praxissemesters gab es eine Situation in der die H2.LIVE App eine JS Benutzerinteraktion auswerten musste. Hierzu wird im weiteren Verlauf des Berichts eingegangen (Siehe H2.LIVE). Anschließend ergab sich keine Möglichkeit die Kenntnisse in JS zu vertiefen.

[7] https://de.wikipedia.org/wiki/JavaScript

3 Aufgaben

Aufgrund der in der Einarbeitung erworbenen Kenntnisse war es möglich an folgenden Projekten zu partizipieren. Im Fokus stand die **H2.LIVE** App, welche im weiteren Verlauf exemplarisch und detailiert beschrieben wird. Mit einer geringeren Priorität folgte die **LFI** App und zum Schluss gab es noch ein Projekt in dem eine sogenannte **Launcher** App entwickelt werden sollte. Diese beiden Apps werden nur kurz beschrieben.

3.1 Theorie

Die vielfältigen Möglichkeiten und Eigenschaften die Android Studio bietet können im Rahmen dieses Praktikumsberichts nicht im Detail beschrieben werden. Daher wird hier nur auf die für die Aufgabenbewältigung wesentlichen Aspekte eingegangen. Dazu zählen besonders *Activities* und *Fragments*. Im folgenden wird dazu ein kurzer Überblick präsentiert und auschließend auf die verschiedenen Apps eingegangen.

3.1.1 Activity

Eine *Activity* dient als Fenster in dem *User Interface* Elemente (UI) platziert sind, um mit dem Benutzer zu interagieren. Sie füllt den gesamten Bildschirm des Gerätes aus und implementiert verschiedene Methoden die dem *Lifecycle* der *Activity* dienen und um die *Activity* mit der zugehörigen .xml Datei zu verbinden.[1]

3.1.2 Fragment

Fragments sind ein Teil der UI der App und können in einer *Activity* platziert werden. Sie gehören demnach zu einer *Activity*. Es ist auch möglich mehrere *Fragments* in einer *Activity* zu platzieren. Der aktuell beste Ansatz ist es eine *Activity* zu haben und den Rest über

[1] vgl: https://developer.android.com/reference/android/app/Activity

Fragments zu realisieren[2]. Abbildung (3.1) zeigt eine Übersicht zum *Lifecycle* von *Fragment* und *Activity*.

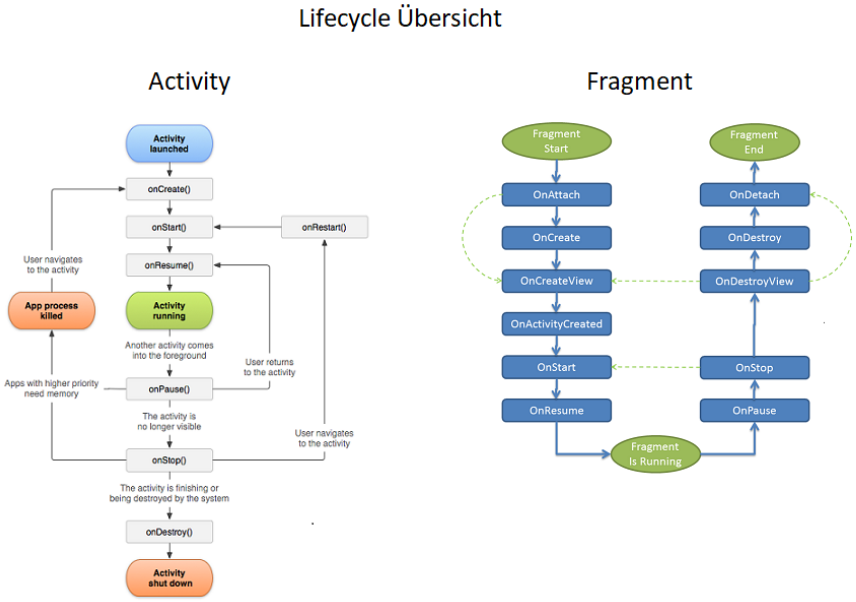

Abbildung 3.1: Lifecycle von Activity und Fragment[3]

3.2 H2.LIVE

Die App ist für Besitzer von Wasserstoffautos, also Autos mit einer Brennstoffzelle, gedacht. "Die Brennstoffzelle wird sich durchsetzen"[4], ist sich Toyota-Motorenentwickler Gerald Killmann sicher. Daher hat die H2 Mobility Deutschland GmbH & Co.KG die Firma portrix.net mit der Entwicklung der H2.LIVE App beauftragt, die den Nutzern unter anderem das europäische Tankstellennetz und die Route zur nächstgelegenen Tankstelle anzeigt, ein Feedback für die Tankstellenbetreiber, Schulungsvideos zum Tankvorgang und viele weitere Funktionen bietet. Zukünftig soll es auch möglich sein mittels Smartphone via App am Tankterminal zu bezahlen. Die Entwicklung der App erfolgt nach Rücksprache mit dem Vertreter von H2 Mobility und dem für das Layout verantwortlichen Designer und wird sowohl für iOS als auch für Android bei portrix.net programmiert. Features und design-Prototypen werden über das digitale Produkt Design Tool InVision an die Programmierer vermittelt und anschließend

[2] vgl: https://stackoverflow.com/questions/20306091/dilemma-when-to-use-fragments-vs-activities
[4] http://www.spiegel.de/auto/aktuell/wasserstoffauto-die-brennstoffzelle-wird-sich-durchsetzen-a-1235431.html

umgesetzt. Aufgrund der hohen Komplexität und des Umfangs den die App mittlerweile angenommen hat, hat sich das Erscheinungsbild zwischenzeitlich stark verändert. Das organische anwachsen der Funktionalität ist am Versionskontrollbaum gut zu erkennen, da das Projekt beinahe 20000 Commits enthält.

3.2.1 Feedback

Über InVision kam die Vorgabe einen Prototyp zu entwickeln, der es ermöglicht einer konkret ausgewählten Tankstelle per *Button* ein Feedback zu geben.

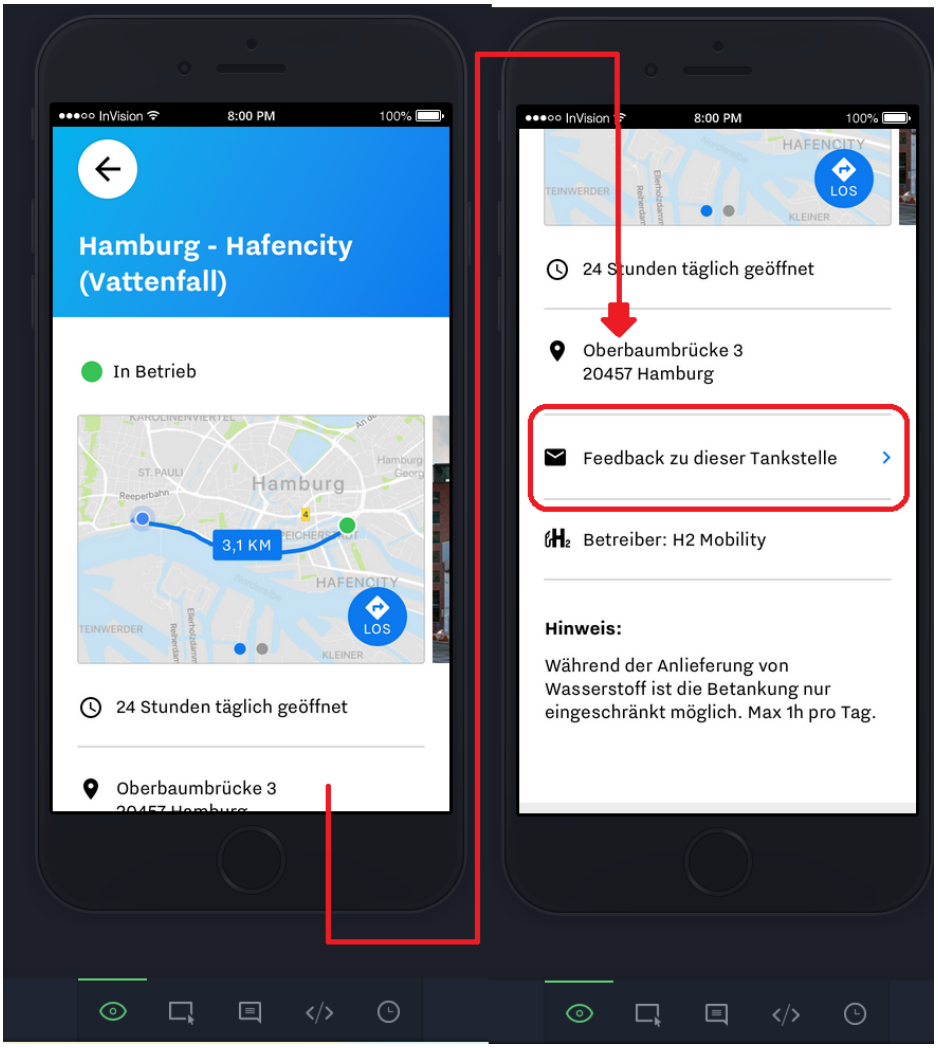

Abbildung 3.2: Prototyp der InVision Vorgabe. Feedback Button für konkret ausgewählte Tankstelle

In der bisherigen Version war es möglich über einen *Button* in der FuelStationsMap.java (MainActivity) das HelpAndFeedback.java (*Fragment*) aufzurufen und von dort in ein allgemeines Feedback.java (*Fragment*) zu kommen (Siehe Abb.3.3) (1 -> 2b -> 3). Dort lässt sich aus einem *Spinner*, der jede Tankstelle auflistet, eine Tankstelle auswählen. Es war also naheliegend den Code zu modifizieren, um aus der FuelStationDetail.java (*Activity*) direkt in das Feedback.java zu wechseln und die angewählte Tankstelle im *Spinner* anzuzeigen. Hierzu mussten diverse Änderungen vorgenommen werden, da das Feedback.java der darunterliegenden FuelStationsMap.java gehört und in der *OnCreate()* Methode des Feedback.java eine FuelStationsMap Variable als *Context* initialisiert wird (3.1). Dieser *Context* gewährleistet unter anderem das der *Spinner* im Feedback.java mit Daten der Tankstellen gefüllt werden kann. Der *Context* macht es allerdings unmöglich das Feedback.java von einer weiteren *Activity*, also der FuelStationDetail.java (*Activity*) gestartet wird. Es war daher notwendig die FuelStationDetail.java (*Activity*) als *Fragment* nachzubauen. Der gewünschte Ablauf wird in Abb.3.3 dargestellt. Zusammenfassend lässt sich sagen, das ein groß des Codes der in der *onCreate()* Methode der FuelStationDetail.java (*Activity*) ausgeführt wird, in die *onCreateView()* Methode der FuelStationDetail.java (*Fragment*) kopiert werden muss. Allerdings ist die Initialisierung der UI Elemente in der korrespondierenden .xml Datei anders zu behandeln. Das Fragment bekommt ein *View* übergeben und dieser *View* muss den UI Elementen vorangestellt werden.

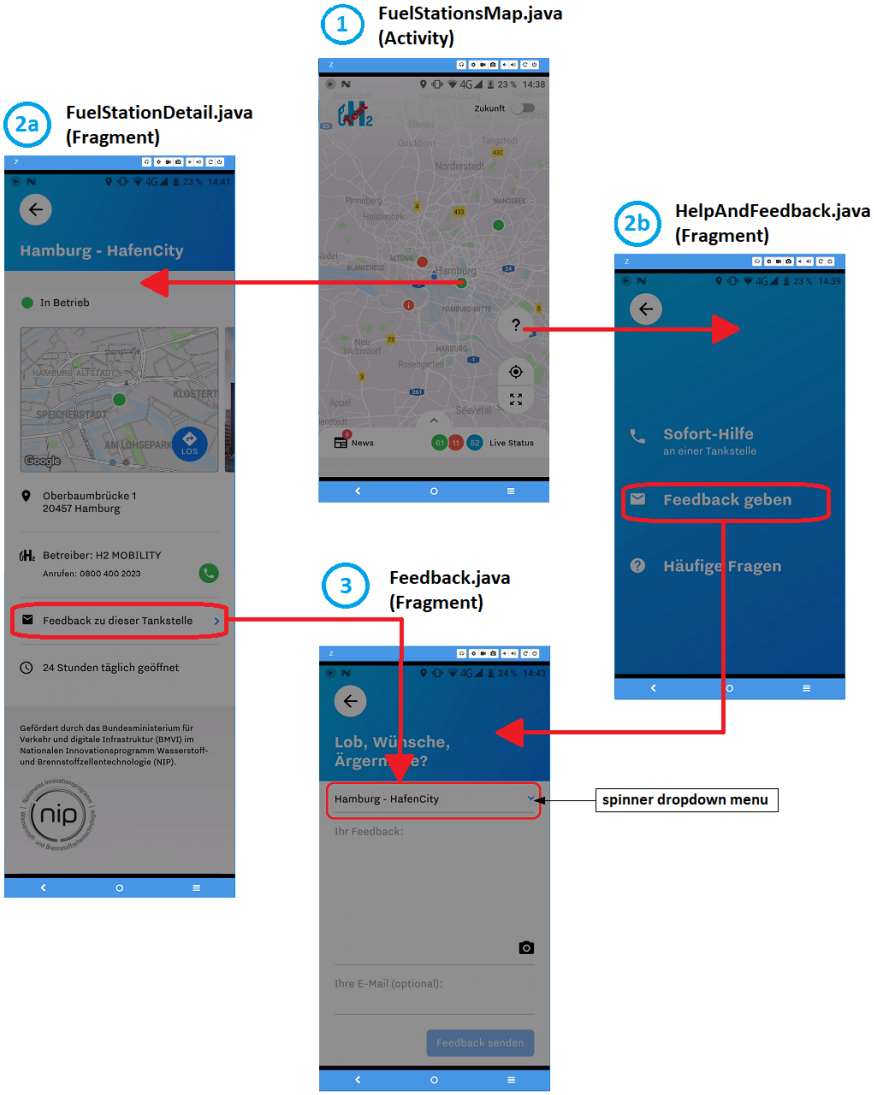

Abbildung 3.3: Workflow der neuen Feedback Funktion

```
1
2  @Override
3  public void onCreate(@Nullable Bundle savedInstanceState) {
4          super.onCreate(savedInstanceState);
5          //mContext = (FuelStationsMap) getActivity(); // initialisierung fÃ¼r veraltete Version
6          mContext = FuelStationsMap.getMap();
7          mContext.addTrackingScreen(getString(R.string.trackingname_view_detail_open));
8          startTimer();
9  }
```

Listing 3.1: onCreate() Methode der Feedback.java

Das Layout einer App wird in der zur *Activity* bzw. *Fragment* zugehörigen .xml Datei definiert. In diesem Fall konnte der meiste Inhalt aus der *activity_fuel_station_detail.xml* in die neue *fragment_fuel_station_detail.xml* kopiert und um das gewünschte neue Feature ergänzt werden (Abb.3.2).

```
1  <!--                    -->
2          <FrameLayout
3                  android:id="@+id/lyFeedbackPin"
4                  style="@style/FuelStationDetail.ValueContainer">
5
6                  <ImageView
7                          android:id="@+id/ivFeedbackPin"
8                          android:layout_width="wrap_content"
9                          android:layout_height="16dp"
10                         android:layout_marginTop="27dp"
11                         android:src="@drawable/ic_feedback_mail_pin"
12                         android:contentDescription="@string/feedback_mail_pin"        />
13
14                 <de.h2.mobility.app.h2live.common.TextViewWithFont
15                         android:id="@+id/tvFeedbackPin"
16                         style="@style/FuelStationDetail.ValueLabel"
17                         tools:text="Feedback zu dieser Tankstelle" />
18
19                 <ImageView
20                         android:id="@+id/ivFeedbackPinArrow"
21                         android:layout_width="wrap_content"
22                         android:layout_height="wrap_content"
23                         android:layout_gravity="end|center_vertical"
24                         android:src="@drawable/ic_cta_arrow_right_pin"
25                         android:contentDescription="@string/cta_arrow_right_pin" />
26
27
28                 <View style="@style/FuelStationDetail.ValueSeparator" />
29
30         </FrameLayout>
```

Listing 3.2: Designelement Feedback Button

Der Transfer (1 -> 2a) erfolgt durch anklicken eines Tankstellen *Markers*. Die bisherige Logik rief die FuelStationDetail.java (*Activity*) auf und musste durch eine *FragmentTransaction* ersetzt werden. Zu sehen in Abb.3.3. Beim Aufruf von *onMarkerClick()* wird der angeklickte *Marker* als Argument übergeben. Es wird über die MapHelper Klasse eine FuelStation Variable **station** zugewiesen und dessen Property **idx**, wenn die **station** ungleich null ist, in ein *Bundle* gespeichert. Dieses *Bundle* **b** wird einem neuen FuelStationDetail (*Fragment*) als Argument übergeben und anschließend wird mit dem *SupportFragmentManager* die *FragmentTransaction* zum FuelStationDetail mit ausgewählter FuelStation durchgeführt.

```
1  @Override
2  public boolean onMarkerClick(final Marker marker) {
3      // Click on a fuel station?
4      FuelStation station = mMapHelper.getMarkerMap().get(marker);
5      if (station == null) {
6          Log.d(TAG, "No Station found");
7          return false;
8      }
9      selectedStation = station.getIdx();
10     Bundle b = new Bundle();
11     b.putInt("station", selectedStation);
12     FuelStationDetail fuelStationDetail = new FuelStationDetail();
13     fuelStationDetail.setArguments(b);
14     getSupportFragmentManager()
15                                 .beginTransaction()
16                                 .setCustomAnimations(
17                                         R.animator.enter_from_right,
18                                         R.animator.exit_to_right,
19                                         R.animator.enter_from_right,
20                                         R.animator.exit_to_right)
21                                 .add(R.id.fragmentContainer, fuelStationDetail, FuelStationDetail.TAG)
22                                 .addToBackStack(null)
23                                 .commit();
24
25     return true;
26 }
```

Listing 3.3: Wechsel von FuelStationsMap zu FuelStationDetail

Um über die Detailansicht der Tankstelle zum gewünschten Feedback (2a -> 3) zu gelangen muss die ID des *FrameLayout* **lyFeedbackPin** (Abb.3.2 Zeile 3) deklariert und initialisiert werden und anschließend über einen *onClickListener()* angeklickt werden (Abb.3.4). Das Prozedere ist dem vorangegangenem ähnlich. Der unterschied besteht darin das in der *onClick()* Methode das Bundle **b** über *getArguments()* den *Marker* übergeben bekommt und an das Feedback.java weiterreicht, damit dort das *Spinner field menu* den Tankstellen Namen vorauswählen kann. Die eigentliche Transaktion von FuelStationDetail zu Feedback funktioniert nach dem gleichen Muster wie zuvor beschrieben.

```
1  // Feedback
2  final ViewGroup lyFeedback = parentView.findViewById(R.id.lyFeedbackPin);
3  lyFeedback.setVisibility(View.VISIBLE);
4  TextView tvFeedback = lyFeedback.findViewById(R.id.tvFeedbackPin);
5  tvFeedback.setText(getString(R.string.fuel_station_detail_feedback));
6  lyFeedback.setOnClickListener(new View.OnClickListener() {
7
8      @Override
9      public void onClick(View v) {
10         Bundle b = getArguments();
11         Feedback feedbackFragment = new Feedback();
12         feedbackFragment.setArguments(b);
13         getSupportFragmentManager()
14                                 .beginTransaction()
15                                 .setCustomAnimations(
16                                         R.animator.enter_from_right,
17                                         R.animator.exit_to_right,
18                                         R.animator.enter_from_right,
19                                         R.animator.exit_to_right)
20                                 .add(R.id.fragmentContainer, feedbackFragment, Feedback.TAG)
21                                 .addToBackStack(null)
22                                 .commit();
23     }
24 });
```

Listing 3.4: Wechsel von FuelStationDetail zu Feedback

3.2.2 Opt-Out

Um das Nutzerverhalten der App zu verbessern werden sämtliche Interaktionen des Benutzers mithilfe von *Google Analytics* registriert. Aufgrund der Datenschutzgrundverordnung[5] (DSVGO) muss es dem Benutzer möglich sein dies zu deaktivieren. Dazu bietet Google Analytics die sogenannte Opt-Out Funktion an. Der *Google Analytics deaktivieren-Button* befindet sich im Impressum (Abb.3.4). Das Impressum ist ein *WebView* indem die Url des Impressums der H2.Live Website aufgerufen und angezeigt wird. Die *HTML* (3.4) enthält JavaScript Elemente die in der App ausgelesen und interpretiert werden, wenn der *Button* gedrückt wird.

[5] https://www.121watt.de/online-marketing/google-analytics-datenschutz-konform/

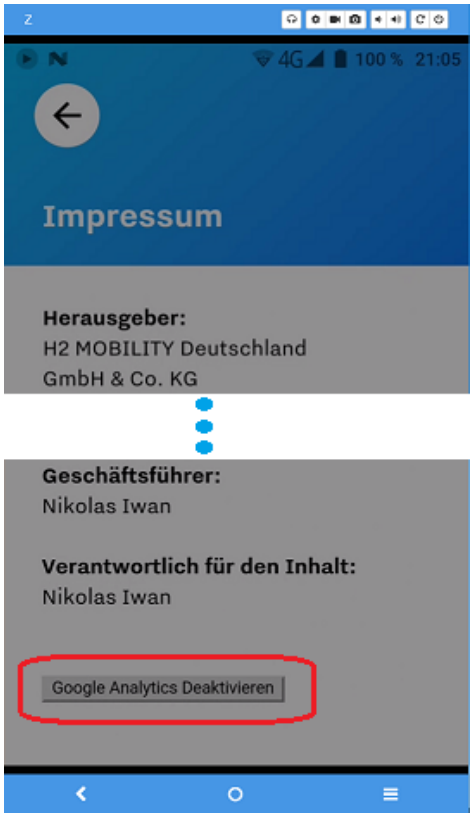

Abbildung 3.4: Impressum mit Google Analytics deaktivieren Button

Abbildung (3.5) (Zeile 7) zeigt das *HTML* CodeSegment in dem der *Google Analytics Deaktivieren* Button definiert wird. Ab Zeile 13 wird mit JavaScript beschrieben, was für die verschiedenen Betriebssysteme bei Knopfdruck passieren soll. Zeile 20 beschreibt das verhalten für Android und Zeile 22 für iOS. Es ist von entscheidender Bedeutung, dass die Klassen, Methoden und Objekte im JavaCode die gleichen Namen haben wie im JavaScript. Damit der JavaCode JavaScript interpretieren kann muss ein JavaScriptInterface implementiert werden. In Abbildung (3.6) ist zu sehen, wie in der inneren Java Klasse GoogleAnalytics.java die Methode postMessage() aufgerufen und als Argument ein JsonObject mit dem Property **action** übergeben wird. Wenn **action** den Wert **optOut** beinhaltet wird der *Google Analytics Tracker* deaktiviert indem der Methode *setTrackerOptOut(true)* als Argument übergeben wird. Die Implementierung des JavaScriptInterface erfolgt durch die *@ Annotation* (Zeile 1) und schließlich durch das verbinden der Java Klasse *GoogleAnalytics()* mit der JavaScript Klasse *googleAnalytics* (Zeile 22).

```
1  <div class="raster-wrapper ">
2
3      <div class="raster float"><div class="inner">
4
5                  <div class="fit"><div class="inner">
6
7                      <button type="button" class="btn" onclick="disableGoogleAnalytics()"
8                          style="margin-bottom: 50px;"> Google Analytics Deaktivieren </button>
9
10                 </div></div>
11         </div></div>
12 </div>
13 <script>
14
15     var disableGoogleAnalytics = function () {
16         jQuery(".btn").prop( "disabled", true ).text("Sendet...");
17
18         try {
19             if (typeof googleAnalytics !== 'undefined') {
20                 googleAnalytics.postMessage(JSON.stringify({"action":"optOut"}));
21             } else {
22                 window.webkit.messageHandlers.googleAnalytics.postMessage({"action":"optOut"});
23             }
24         } catch(err) {
25             console.log("could not disable google analytics, maybe you don't have an iOS device.")
26         }
27
28         jQuery(".btn").prop( "disabled", false ).text('Google Analytics Deaktivieren');
29     }
30
31 </script>
```

Listing 3.5: HTML des Impressums

```
1  @SuppressLint("SetJavaScriptEnabled")
2  private void initView() {
3
4          // GoogleAnalytics opt out
5          class GoogleAnalytics {
6                  @JavascriptInterface
7                  public void postMessage(String action) {
8                          try {
9                                  JSONObject json = new JSONObject(action);
10                                 String a = json.getString("action");
11                                 Log.d(TAG, "Message: " + a);
12                                 if ("optOut".equals(a)) {
13                                         activity.setTrackerOptOut(true); // disable GoogleAnalytics
14
15                                 }
16                         } catch (JSONException e) {
17                                 Log.e(TAG, e.getLocalizedMessage());
18                         }
19
20                 }
21         }
22
23         addJavascriptInterface(new GoogleAnalytics(), "googleAnalytics");
24 }
```

Listing 3.6: Interpretation von JavaScript in Android Studio

3.3 LFI App

Diese App richtet sich an Fotografen mit einer Leica Kamera. Es gibt eine Version für Tablets und eine für Smartphones. Die App hat die folgenden Rubriken:

- News
- Blog
- Galerie
- Magazin
- Video
- Shop

In der Rubrik Magazine werden zum die Magazine als PDF zum kauf angeboten. Über In-Vision kam der Auftrag die Ansicht eines Magazins neu zu gestalten und die Vorschau auf ein nicht gekauftes Magazin zu ermöglichen. Die Vorschau sollte Deckblatt und Inhaltsverzeichnis plus zehn Seiten des Magazins beinhalten. Aus Platzgründen wird die Realisierung in diesem Bericht nicht weiter beschrieben.

3.4 Launcher App

Unter einem Launcher wird beim OS Android die App verstanden die beim Starten des Smartphones aktiviert wird, zum Beispiel "Pixel Launcher". Aus dem Launcher lassen sich alle weiteren Apps starten. Android ermöglicht es die Launcher App zu ändern bzw. eine eigene Launcher App zu verwenden. Diese eigene App sollte bei portrix.net entwickelt werden und nur einen gewissen äbgespecktenÜmfang an Apps beinhalten. Um eine App als Launcher zu deklarieren muss dies im *Manifest* der App festgelegt werden. Zum Ende des Praxissemesters war es möglich auf dem Startbildschirm zuvor ausgewählte Apps und maximal drei *Widgets* anzuzeigen und zu starten. Aus Zeitgründen hat die Launcher App den Status einer Fingerübung nicht überschritten und aus Platzgründen wird sie hier nicht weiter beschrieben.

4 Zusammenfassung

Generell ist der kreative Raum den die Programmierung mit Android Studio bietet eine ganz beeindruckende Erfahrung, sobald man sich etwas zurechtgefunden hat. Obwohl die in den Code Beispielen beschriebenen Aspekte im Nachhinein einfach aussehen, ist es mir nicht leicht gefallen die Lösungen zu erarbeiten. Bestehende komplexe Projekte zu analysieren bedarf viel Übung und Hartnäckigkeit, da Fehler sich schnell einschleichen und es allerhand Details zu beachten gilt. Diese aufzuspüren und zu beseitigen ist eine große Herausforderung, bereitet jedoch im Erfolgsfall ein umso besseres Gefühl. Ich bin der Firma portrix.net und allen Mitarbeitern sehr dankbar, dass ich diese Erfahrung dort sammeln konnte.

Quellenverzeichnis

[1] J. Delgadillo, “The complete linux course: Beginner to power user!” 2017. [Online]. Available: https://www.youtube.com/watch?v=wBp0Rb-ZJak

[2] “Xml vs. html: Die unterschiede einfach erklärt,” 2018. [Online]. Available: https://www.as-computer.de/wissen/unterschiede-html-und-xml/

[3] “Javascript object notation,” 2019. [Online]. Available: https://de.wikipedia.org/wiki/JavaScript_Object_Notation

[4] D. Panjuta, “Der komplette android 8 entwickler kurs - erstelle 20+ apps,” 2018. [Online]. Available: https://www.udemy.com/der-komplette-android-8-entwickler-kurs/

[5] C. Getsy, “Java leicht gemacht - der umfassende java einsteigerkurs a-z,” 2018. [Online]. Available: https://www.udemy.com/programmieren-lernen-mit-java-ein-kurs-fur-einsteiger/

[6] “Gitlab,” 2019. [Online]. Available: https://de.wikipedia.org/wiki/GitLab

[7] “Javascript,” 2019. [Online]. Available: https://de.wikipedia.org/wiki/JavaScript

[8] “Activity,” 2018. [Online]. Available: https://developer.android.com/reference/android/app/Activity#ActivityLifecycle

[9] “Dilemma: when to use fragments vs activities,” 2013. [Online]. Available: https://stackoverflow.com/questions/20306091/dilemma-when-to-use-fragments-vs-activities

[10] “Erstellen eines fragments,” 2018. [Online]. Available: https://docs.microsoft.com/de-de/xamarin/android/platform/fragments/creating-a-fragment

[11] E. Nefzger, “Die brennstoffzelle wird sich durchsetzen,” 2018. [Online]. Available: http://www.spiegel.de/auto/aktuell/wasserstoffauto-die-brennstoffzelle-wird-sich-durchsetzen-a-1235431.html

[12] D. M. Schirmbacher, “Google analytics und datenschutz (dsgvo) ? diese rechtlichen anforderungen musst du erfüllen,” 2018. [Online]. Available: https://www.121watt.de/online-marketing/google-analytics-datenschutz-konform/