Web Project

Autor:
Kevin Zahn
Auftraggeber:
Matthias Studer
Institution:
HFTM
Studiengang:
Systemtechnik ICT
Datum
27.02.2022

Inhaltsverzeichnis

Nutzen der Webseite ... 3
Zielgruppe ... 3
Programmierung ... 3
HTML ... 3
Struktur der HTML Seite ... 3
Bilder ... 3
Schriftart und Texte ... 4
Animieret Elemente ... 4
css ... 6
Datei style.css ... 6
Datei stylePart1.css ... 6
Datei stylePart2.css ... 8
Datei stylePart3.css ... 9
Datei stylePart4.css ... 11
Datei stylePart5.css ... 11
Datei routImage.css ... 12
Datei scrollbar.css ... 12
JavaScript ... 12
jQuery ... 12
Navigationsbar ... 16
Ladezeit Analyse ... 17
Kopieret und Übernommene Elemente ... 17
Schlusswort ... 18

Nutzen der Webseite

Bei vielen Webseiten, bei denen es um Wanderungen geht, ist schwierig abzuschätzen wie es dort aussieht und man kann sich schlecht ein Bild von der Wanderung mache. Ein Grund dafür ist, man kann die verfügbaren Bilder nicht chronologisch einordnen. Es ist schwierig sich die Wanderung bildlich vorzustellen. Dieses Problem wir mit der Webseite gelöst, man kann genau sehen wie, es wo aussieht, aufgebaut ist es ziemlich ähnlich wie „Google Maps Street View“ nur halt für die Berge.

Durch die Webseite soll man eine bessere Übersicht über die Wanderroute bekommen, so dass man besser abschätzten kann, ob sie für einen passt.

Zielgruppe

Die Zielgruppe befindet sich im alter von 18 bis 40 und sind tendenziell gerne draußen. Sicherlich gibt es auch älter Personen, für die die Informationen auf der Webseite nützlich sind, jedoch kann es sein das für älter Gruppen das User Interface kompliziert erscheint. Man kann sich durchaus vorstellen das die Pins auf der Wanderrouten nicht verstanden werden. Wenn die Pins auf der Wanderrouten nicht angeklickt werden bekommt man deutlich weniger Informationen über die Wanderung als sonst.

Programmierung

HTML

Struktur der HTML Seite

Die Webseite ist in 5 Bereiche eingeteilt.

1. Startseite der Webseite
2. Beschreibung der gesamten Wanderung
3. Beschreibung des ersten Tags
4. Beschreibung des zweiten Tags
5. Beschreibung des dritten Tags

In der HTML Struktur wird das in sogenannte „part’s“ eingeteilt *(sieh HTML Code 1.1)*

Diese „part’s“ sind auch dafür da, dass die Links in der Navigationsbar wissen, wo hin sie springen müssen.

```html
<!-- Startseite der Webseite -->
<section id="part_1">
</section>

<!-- Überblick der Wanderung -->
<section id="part_2">
</section>

<!-- Erster Tag -->
<section id="part_3">
</section>

<!-- Zweiter Tag -->
<section id="part_4">
</section>

<!-- Dritter Tag -->
<section id="part_5">
</section>
```

HTML Code 1.1

Bilder

Da die Webseite die Wanderung möglichst grafisch darstellen soll werden viele Bilder verwendet. Insgesamt sind es 34 Bilder, die auf der Webseite dargestellt werden. Alle Bilder, außer die Routen Bilder, wurden selbst fotografiert und bearbeitet. Die Routen Bilder stammen von Google Maps.

Die Bilder haten anfangs eine Auflösung von 6000 x 4000 Pixel das ergab im Durchschnitt eine Größe von 10 MB. Alle Bilder wurden verkleinert, auf folgende Auflösungen.

Bilder Auflösungen

Nach Bearbeitung haten alle Bilder diese Auflösungen:

- Titelbild:
  Auflösung: 1280 x 720 Pixel
  Größe: 2.64 MB
- Routenbilder:
  Auflösung: 1280 x 604 Pixel
  Größe: ca. 2.21 MB
- Bilder auf der Route:
  Auflösung: 960 x 640 Pixel
  Größe: 1.76 MB

Schriftart und Texte

Für die Schriftart wurde die Standard Ubuntu Schriftart verwendet. Da die Schriftart der Browser nicht unterstützt und das Einbinden über „Google Fonts“ eine größere Latenz verursacht, wurde die Schriftart direkt im Projekt abgespeichert. Damit das css weis welche Schriftart verwendet werden muss musste der „font-face“ bestimmt werden. *(sieh HTML Code 1.2)* Die Paragrafen wurde alle selbst erstellt, anhand der erlebten Erlebnisse.

```css
/* Importieren der Schriftart des Projekts */
@font-face {
  font-family: ubuntu;
  src: url(../../rsc/Fonts/Ubuntu/Ubuntu-Light.ttf);
  font-width: light;
}
```

css Code 1.2

Animieret Elemente

Auf der Webseite wurden Animationen teils mit css oder mit JavaScript erstellt. Die Anzeige der Bilder auf der Roadmap wurde mit Hilfe von JavaScript erstelle. Genau gesagt mit einer Bibliothek von JavaScript namens „jQuery“. Auf die genaue Funktion wird im Kapitel JavaScript eingegangen. Die Bibliothek kann auf zwei Wege installiert werden mithilfe von „npm“ (Node Paktet Manager) oder so wie es in der Webseite gelöst ist, über ein API-Schnittstelle. *(siehe HTML Code 1.3)*

```html
<!-- Einbinden von JQuery für die Bilder Animationen und „load data on scroll“-->

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js">
</script>
```

HTML Code 1.3

Über das css wurde die Navigationsbar animiert, die Animation besteht darin das sich die Navigationsbar in 0.3 Sekunden nach oben verschwinden wenn nach unten gescrollt wird. Für den fließenden Übergang ist ein css-Befehl zuständig. *(siehe HTML Code 1.4)*

```css
/* Lässt Navigationsbar fließend verschwinden */
transition: top 0.3s;
```

css Code 1.4

Jedoch reicht diese Animation nicht das die Navigationsbar verschwinde zusätzlich nötig ist JavaScript, mehr in Kapitel JavaScript.

Wenn auf einen Link in der Navigationsbar geklickt wird, wechselt die Webseite auf die richtige Zeile, jedoch funktioniert das mit einer Animation. Die Animation gleitet langsam zur angeklickten Zeile. *(siehe HTML Code 1.5)*

```css
html {
  scroll-behavior: smooth;
}
```

css Code 1.5

Zu den Animationen könnte man noch die „Hover“ Elemente dazuzählen, das sind:

- Navigationsbar
  - Wechselt die Farbe von Hellblau zu Dunkelbau. *(css Code 1.6)*
- Sozial Media Icons
  - Wechselt die Farbe von Weiß zu Hellgrau und wird zusätzlich ein paar Pixel grösser. *(css Code 1.7)*
- Pins / Markierungen auf der Route
  - Vergrößert sich und wechselt die Farbe von Dunkelblau zu Hellblau. *(css Code 1.8)*

```css
/* Verändert die Backgroundfarbe, wenn die Maus darüber kommt */
nav ul a:hover, #btnDropdown:hover, #dropdownContent a:hover
{
  background-color: rgb(0, 60, 255);
}
```

css Code 1.6

```css
/* Vergrössert und wechsel die Farbe wenn darübergefahren wird */
.icon:hover{
  font-size: 28px;
  color:#9b9b9b;
}

/* Vergrössert und wechsel die Farbe wenn darübergefahren wird */
#mail:hover{
  transform: scale(1.25);
  color:#9b9b9b;
}
```

css Code 1.7

```css
.pin:hover {
  border-color: rgb(70, 164, 218); /* Setzt die Farbe des Rahmens */
  height: 1.5vh; /* Setzt die Höhe des Punktes */
  width: 1.5vh; /* Setzt die Breite des Punktes */
  border-width: 0.35vh; /* Setzt die Dicke des Rahmens */
  z-index: 1000; /* Setzt die Position über einen andern Kreis */
}
```

css Code 1.8

css

Da das css mit der Zeit immer umfangreicher wurde, wird in dieser Dokumentation nicht auf das gesamte css eingegangen, um das css besser zu verstehen wird auf die Kommentare des Sourcecode vom css verwiesen, der die meisten css Befehle beschriebt. Das css befindet sich im Ordner „src/styles“.

So wie das HTML in die verschiedenen Part’s aufgeteilt wurde, wurde auch das css aufgeteilt. Die Aufteilung ist wie folgend:

- style.css
- stylePart1.css
- stylePart2.css
- stylePart3.css
- stylePart4.css
- stylePart5.css
- routImage.css
- scrollbar.css

Datei style.css

In der „style.css“ werden alle globalen Styles festgelegt wie Schriftart, Styles der Überschriften und Paragrafen. Auch werden in dieser Datei die responsiven Schriftgrößen für die Mobilen Geräte festgelegt.

Datei stylePart1.css

Auf der Startseite erwies sich die Schwierigkeit ein Bild als Hintergrund Bild zu platzieren. Es gibt ein css Befehl der ein Bild als Hintergrundbild platziert, jedoch erwiess sich dies nicht als nützlich. Das Bild wurde einfach mit dem z-Index in den Hintergrund verschoben und die Elemente darüber geschoben. Damit die Elemente nicht unter dem Bild angezeigt werden, musste die Position auf relativ gesetzt werden. *(css Code 2.1)*

```css
#part_1 {
    /* Mit Position Relativ kann gewährleistet werden, das sich
    alle darin befindenden Element nicht größer alls das Element werden
    und die Elemente könne auf das Bild geschoben werden */
    position: relative;
}
```

Css Code 2.1

Bei der Navigationsbar wurde dann die Position wieder zurück auf fixed gesetzt, um es auf dem Bild zu halten. Fixed bedeutet auch das die Navigationsbar immer an der gleich stelle bleibt, wenn gescrollt wird. Mit Hilfe von JavaScript verschwindet dann die Navigationsbar, wenn nach unten gescrollt wird. *(Kapitel JavaScript)*

Anders ist es bei der Überschrift, der Beschriftung unten links und den Icons, die zu den sozialen Medien führen, unten rechts. Sie haben alle eine absolute Position und werden mit den Befehlen top, bottom, left und right an die richtige Stelle versetzt.

Die Icons unten recht wurden von „Google API Material Icons“ und „fontawesome“ bezogen. Bei der Implementierung der Icons, wurde entdeckt das das Mail-Icon, das von „Google“, nicht die gleichen Dimensionen hat wie die Icons von „fontawesome“. Dieses Problem konnte mit den Transform Befehlen von css behoben werden. *(css Code 2.2)* Transform-Befehle nehmen keine Rücksicht auf das Boxen Modell, das sonst genutzt wird, deshalb sollte man die Transform Befehle sparsam einsetzen.

```css
.mail {
    /* transform achtet nicht auf andere Elemente mit scale wird die
    grösse verändert mit translateY wird das Element nach unten geschoben  */
    transform: scale(1.2) translateY(1px);
}
```

css Code 2.2

Nachfolgen ist noch die Planung der Startseite zu sehen:

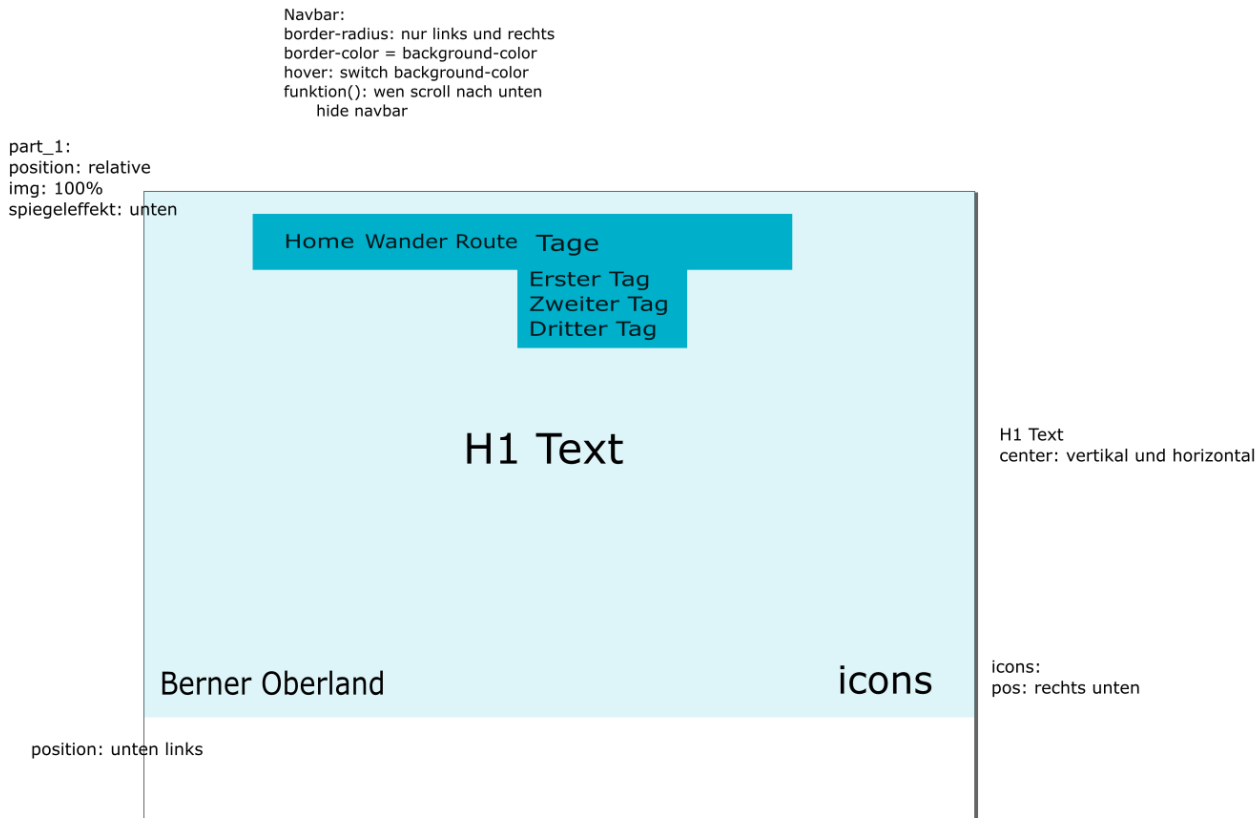

Abbildung 1 Aufbau der Startseite

Datei stylePart2.css

Der Part 2 abschnitt beinhalten ein Gridpanel das wird verwendet um das Responsive designe besser umzusetzen. *(css Code 2.3)* Es wurden zwei spalten erzeuge in der Linken spalte befinden sich Beschreibungen und eine Tabelle, rechts davon eine eingebundenes iframe von Google Maps.

```css
.layout_part_2 {
    /* Setzt ein Grid-Panel fest so kann man
    besser auf
    verändernde Display Größen reagieren. */
    display: grid;
    /* Erzeugt zwei Spalten */
    grid-template-columns: 1fr 1fr;
    /* Setzt die Abstände nach oben 7 % nach
    rechts 4 %
    nach unten kein Abstand und nach links 4 %
    */
    padding: 7% 4% 0% 4%;
}
```

css Code 2.3

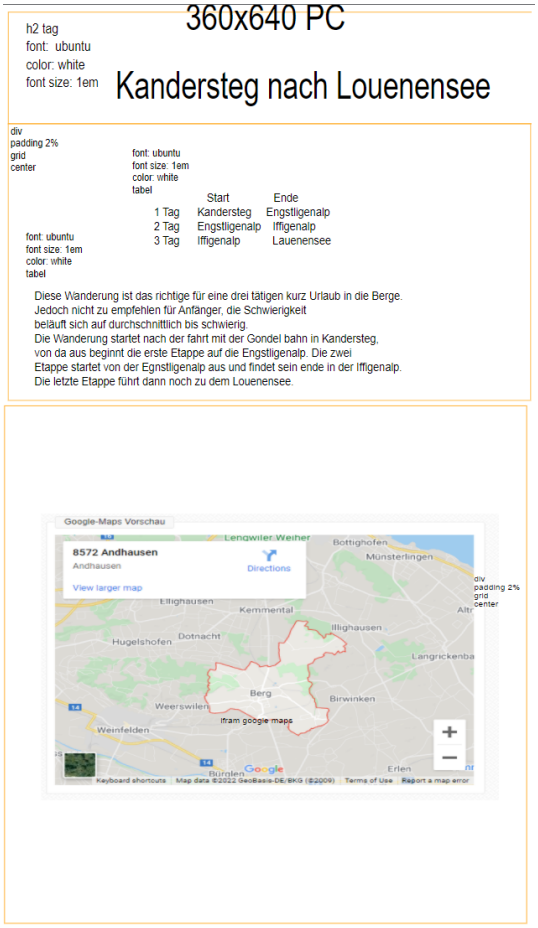

Abbildung 2 Part 2 Mobile

Das Einbinden von Google Maps erwies sich als das schwierigste Element der Webseite. Den als erste wurde versuche die Map über eine API Schnittstell von Google einzubinden jedoch funktionierte lange der API Key nicht. Als der API Key dann funktionierte wurde die Map aber dann nur in einem sogenannten „Developer-Modus“ angezeigt was zur Ursache hatte, dass keine Markierung in der Karte gesetzt werden konnte. Nach langer Recherche wurde ein Tool von „CheckPoll.de“ gefunden was das Problem mit den API Key löste. Auch konnte über diese Webseite eine Markierung auf der Karte gesetzt werden. Ein großer Nachteil dieser Einbindung ist das hinter dem „Google Maps iframe“ ein Werbelink platziert wurde.

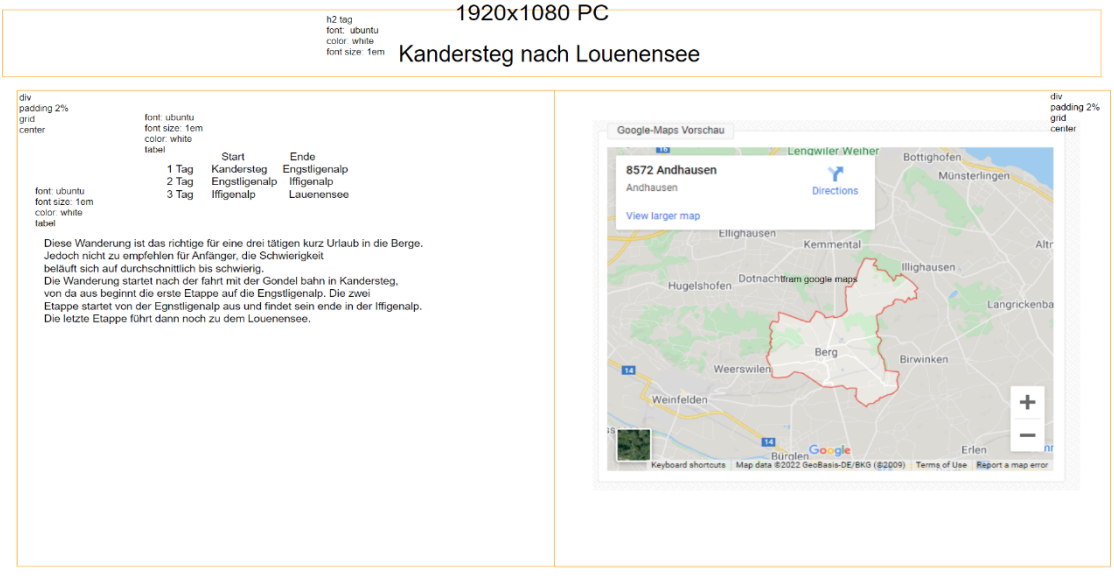

Abbildung 3 Part 2 PC

Datei stylePart3.css

Die Parts 3, 4 und 5 haben viele Elemente gemeinsam, denn auf alle Befindet sich die Routemap die die Wanderroute darstellt. Im Part 3 wurden, die css Befehle für die Punkte, die Routemap Größe und Darstellung der Bilder auf der Routemap designet, für die Parts 3, 4 und 5.

Die Pins auf der Routemap sind leere „div“ im html, die durch die Angabe der Größe und einen Rahmen zu einem Punkt ohne Inhalt werden. *(css Code 2.4)* Um die Position der Pins besser zu bestimmen wurde der Nullpunkt des „div“ mit dem Befehl „transform: translate(-50%, -50%)“ verschoben. Die Positionen werden aber in einer anderen Datei festgelegt, die Befehle für die Positionsbefehle befinden sich in der Datei „routImage.css“.

Auch die Position der Bilder, die auf der Routemap angezeigt werden, werden in der Datei „routeImage.css“ definiert. Die Positionen wurde aus einem Grund ausgelagert, sie verschwendeten sehr viel Platz, insgesamt sind die Positionsbefehle über 300 Zeilen lang.

Wie auch auf Startseite musste das Routemap Bild in den Hintergrund verschoben werden was durch eine relative Position möglich war.

Zurück zu den Pins, alle Pins haben ein „hover“ Event bekommen, das sie beim Berühren hervorhebt. Das Hervorheben funktioniert durch eine Vergrößerung der Größe des Elements und der Rahmenstärke. Auch wird beim berühren die Farbe zu einem dunkleren Blauton und der Z-Index erhöht sich so dass sich der berührte Pin vor andere überschneidende Pins stellt.

Nachfolgend ist die Planung der Part 3 bis 5 zu sehen

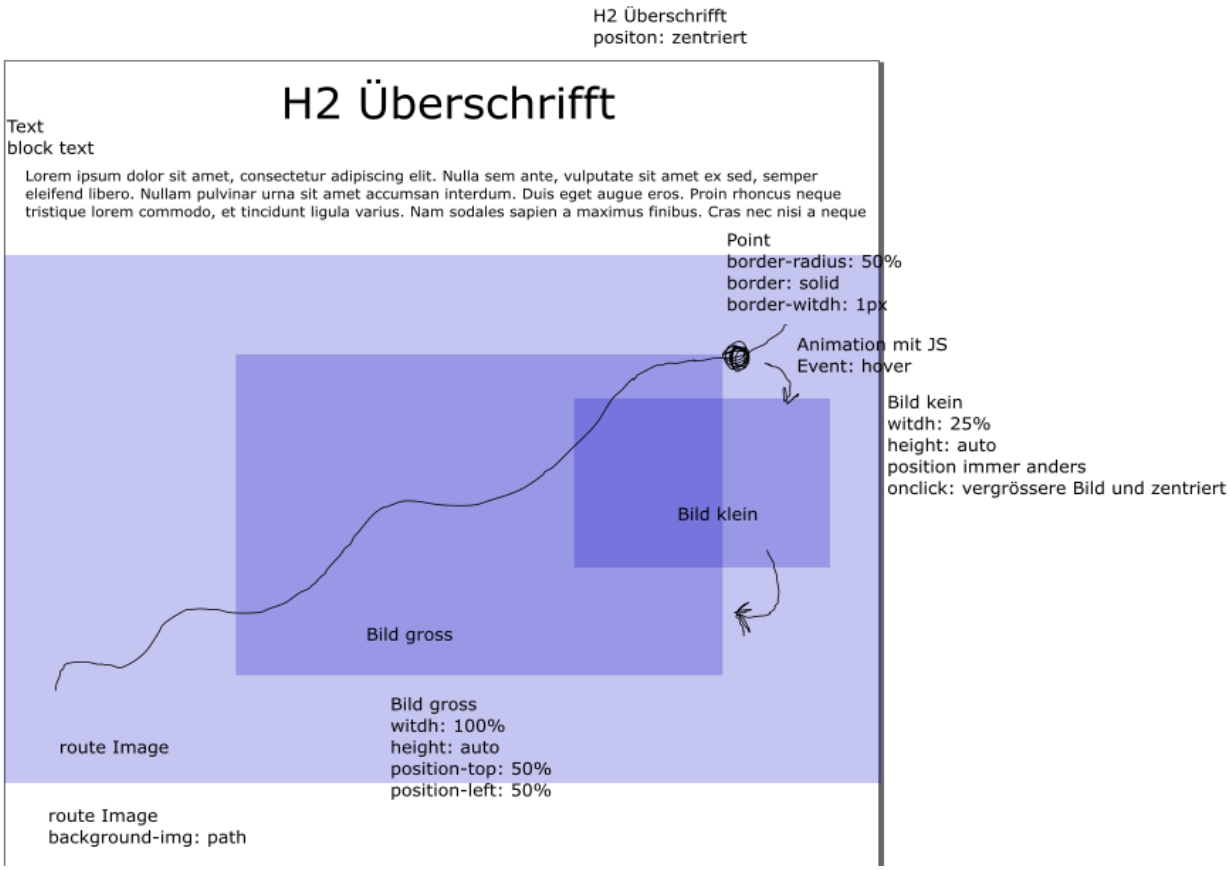

Abbildung 4 Part 3 bis 5

```css
.pin {
    height: 1vmin;
    width: 1vmin;
    /* Blockiert den Punkt so das es nicht an
    das Bild angefügt werden kann, und darüber gefügt wird */
    display: block;
    border: 0.3vmin solid;
    /* Setzt ein Rahmen um das Element */
    border-color: rgba(15, 127, 192, 0.781);
    border-radius: 50%;
    /* Macht den Rahmen zu einem Kreis */
    z-index: 999;
    /* Setzt die Position über das Bild */
    position: absolute;
    /* Verhinder das Anfügen an dem Bild */
    /* Verschiebt die Koordinate es Punktes, so das die Koordinate,
    an dem der Punkt definiert ist, auf die Mitte des Kreises */
    transform: translate(-50%, -50%);
}
```

css Code 2.4

```css
.pin:hover {
    /* Setzt die Farbe des Rahmens */
    border-color: rgb(70, 164, 218);
    /* Setzt die Höhe des Punktes */
    height: 1.5vh;
    /* Setzt die Breite des Punktes */
    width: 1.5vh;
    /* Setzt die Dicke des Rahmens */
    border-width: 0.35vh;
    /* Setzt die Position über einen andern Kreis */
    z-index: 1000;
}
```

css Code 2.5

Datei stylePart4.css

Wie schon erwähnt sind viele css Befehle in der Datei stylePart3 definiert. Damit wird die Datei „stylePart4“ zu den kleinsten css Dateien. Mit einer relativen Position wird die Routemap in den Hintergrund verschoben, wie schon mehrere mal erklärt. Das ganze Element bekommt einen Abstand zu allen anderen Elementen von 4%. Zum Schluss wird noch der Text oberhalb der Routemap bei verkleinert der Webseite angepasst so dass der Text bis zum rechten und linken Bildschirmrand reicht. *(css Code 2.6)*

```
@media only screen and (max-width: 1068px) {
    .text_pos {
        /* Lässt den Text bis an den Bildrand reichen */
        margin: 2% 2% 3% 2%;
    }
}
```

HTML Code 2.6

Datei stylePart5.css

Auch von „stylePart5“ wurde diverse Element schon designet von „stylePart3“. Zusätzlich zum „stylePart4“ kommt noch der „Footer“, der am Ende der Webseite platziert wurde. Im „Footer“ befindet von links nach rechts, der Autor, das Icon von Pinterest, das Icon von Facebook, das Icon von Instagramm, das Icon vom Mail und am Schluss noch das Ersteller Datum. Der „Footer“ wurde in eine sogenannten „Flex Layout“ erstellt. Ein „Flex Layout“ wird verwendet wenn boxen in einer Linie angeordnet werden müssen.

Im „Footer“ befinden sich zwei verschachtelte „Flex Layouts“ das erste „Flex Layout“ beinhaltet die zwei Paragrafen rechts und links und die Icons in der Mitte. *(css Code 2.7)* Die Icons in der Mitte befinden sich in einer Box mit einer bestimmten Größe, das streckt die mittler Box, so haben die Icons einen größeren Lehrraum links und rechts. Das zweite „Flex Layout“ befindet sich in der Box und teil die Icons in gleich große Abstände auf. *(css Code 2.8)*

```
#footer {
    margin: 3% 2% 4% 2%;
    /* Reiht die Element in ein Reihe */
    display: flex;
    /* Setzt ein Element nach recht ganz an den Rand zentriert das in der
    Mitte und setz das Rechte Element ganz nach rechts */
    justify-content: space-between;
}
```

css Code 2.7

```
#footerIcons {
    /* Reiht die Element in ein Reihe */
    display: flex;
    /* Setzt um alle Element den gleichen Abstand */
    justify-content: space-around;
}
```

css Code 2.8

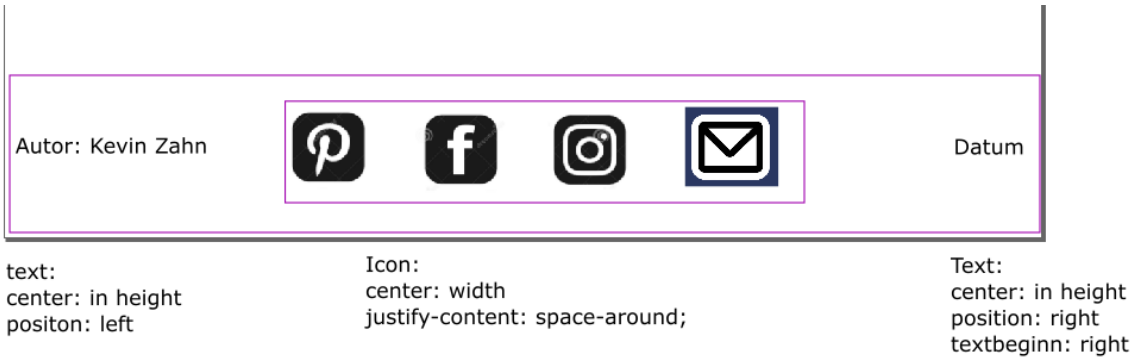

Abbildung 5 Footer Layout

Datei routImage.css

In der Datei „routImage.css“ werden alle Pins und Bilder, die auf der Routemap angezeigt werden, positioniert. Achtung wenn eine Position eines Bildes verändern wird muss diese Änderung auch im JavaScript erfolgen sonst wird das Bild beim Schließen nach der Zoom Funktion, die Zoom Funktion ändert die Positionsparameter, nicht mehr an der richtigen Position angezeigt.

Datei scrollbar.css

Die Scrollbar wird in der Datei „scrollbar.css“ redesinget, diese css wurde von Fabian Leuenberger mit seinem Einverständnis übernommen und leicht angepasst.

JavaScript

jQuery

jQuery ist eine JavaScript-Bibliothek, die Webentwicklern die Arbeit mit JavaScript erleichtern soll. Sie überbrückt technische Unterschiede zwischen verschiedenen Browsern und ermöglicht komfortablen und kompakten Code, der im direkten Vergleich zu JavaScript häufig sehr viel kürzer ist. jQuery wird für das Verschwinden und Auftauchen der Bilder verwendet.

Bilder Animationen

Um die HTML Elemente besser anzusprechen und durch zu iterieren wurde ein Array erstellt mit allen HTML Elementen. *(JavaScript Code 3.1)*

```
// Holt das HTML Element
var ImagesDay1 = [
    document.getElementById("posOfImg1D1"),
    document.getElementById("posOfImg2D1"),
    document.getElementById("posOfImg3D1"),
    document.getElementById("posOfImg4D1"),
    document.getElementById("posOfImg5D1"),
    document.getElementById("posOfImg6D1"),
    document.getElementById("posOfImg7D1"),
    document.getElementById("posOfImg8D1"),
    document.getElementById("posOfImg9D1"),
    document.getElementById("posOfImg10D1")
]
```

JavaScript Code 3.1

Da die Funktion „zoomImageD1“ *(JavaScript Code 3.3)* die Position des Bildes verändert müssen die Standartwerdet der Positionen der Bilder abgespeichert werden. Jedes Mal, wenn ein Pin auf der Route darübergefahren wird setzt es bei allen Bildern die Standartwerte. *(JavaScript Code 3.2)*

```
// Default Werte der Positionen der Bilder
var defaultValuesOfImgDay1 = [
    ["20%", "70%"], //Position des "posOfImg1"
    ["45%", "70%"], //Position des "posOfImg2"
    ["52%", "65%"], //Position des "posOfImg3"
    ["50%", "62%"], //Position des "posOfImg4"
    ["46%", "55%"], //Position des "posOfImg5"
    ["25%", "42%"], //Position des "posOfImg6"
    ["22%", "38%"], //Position des "posOfImg7"
    ["22%", "36%"], //Position des "posOfImg8"
    ["20%", "33%"], //Position des "posOfImg9"
    ["75%", "20%"]  //Position des "posOfImg10"
]
```

JavaScript Code 3.2

Die Funktion „zoomImageD1“ *(JavaScript Code 3.3)* ist zuständig dafür das das Bild wenn es angeklickt wir in der Maximalen grösse angezeigt wird. Auch verändert es die Position des Bildes, das wiederum in der Funktion „showImageD1“ *(JavaScript Code 3.4)* rückgängig gemacht werden muss. Die Standardposition bekommt es vom Array „defaultValuesOfImgDay1“ *(JavaScript Code 3.2)*

```
// Wird aufgerufen wenn auf das Bild gecklickt wird
function zoomImageD1(numberOfElement){
    ImagesDay1[numberOfElement].style.width = "50%";
    ImagesDay1[numberOfElement].style.top = "50%";
    ImagesDay1[numberOfElement].style.left = "50%";
    ImagesDay1[numberOfElement].style.zIndex = 1001;
}
```

JavaScript Code 3.3

Die mit Abstand größte Funktion ist die „showImageD1“ *(JavaScript Code 3.4)* sie ist dafür zuständig, dass die Bilder auf der Route langsam auftauchen und wieder langsam verschwinden. Das Funktioniert so:

Die Funktion wir aufgerufen wenn über Pins gefahren wird. Das HTML übergibt der Funktion ein Parameter in Form einer Zahl, in der Variable „numberOfElement“. Diese Zahl bestimmt auf welchem Bild eine Animation stattfindet. Die Funktion iteriert jedes Bild, so kann viel Code gespart werden. Bei jedem Schleifen Durchgang wird überprüft, ob die For-Schiefe am richtigen Bild ist. Wenn es am richtigen Bild angekommen ist, kommt jQuery ins Spiel, jQuery frägt als erste ab, ob das HTML Dokument schon geladen ist, was meistens der Fall ist. Wen es geladen ist führt es die Funktion „.show“ aus, dies Funktion ist von jQuery geschrieben. Diese Funktion ist dafür zuständig das das Bild innerhalb von 0.6 Sekunden angezeigt wird. So wird das Bild angezeigt.

Wenn aber die For-Schleife an eine Bild ist, auf das nicht gefahren wurde, wird es mit Hilfe von jQuery verschwinden, aber nur, wie auch vorher, nur wenn das HTML Dokument geladen wurde. Wenn das Bild verschwinden, bekommt es die Standartwerte der Position, wie schon erklärt.

```
// Funkton benutzt jQuery für die Bilder Animation

function showImageD1(numberOfElement){
    // Iteriert durch alle Bilder
    for(let i = 0; i < ImagesDay1.length; i++){

        if(i == numberOfElement){
            // JQerry animiert das Aufrufen der Bilder dauert 600ms
            $(document).ready(function(){
                $(ImagesDay1[i]).show(600);
            });
        }

        else{ // Alle Elemente, die nicht berührt werden

            // Alle Elemente die noch Angezeigt werden
            if(ImagesDay1[i].style.display == "inline" ||
            ImagesDay1[i].style.display == "inline-block"){

                // JQerry animiert, das Verschwinden der Bilder dauert 600 ms
                $(document).ready(function(){

                    $(ImagesDay1[i]).hide(600, function(){
                        // Stellt den Default-Wert ein nach dem Verschwinden der
                        // Bilder
                        ImagesDay1[i].style.width = "25%";
                        ImagesDay1[i].style.zIndex = 0;
                        ImagesDay1[i].style.top = defaultValuesOfImgDay1[i][0]
                        ImagesDay1[i].style.left = defaultValuesOfImgDay1[i][1]
                    });});}}}}
```

JavaScript Code 3.4

Autoload nach Scrollen

Nicht nur die Bilder Animationen brauchen jQuery auch das automatisch nachladen der Bilder wird mit jQuery realisiert. Das Automatisch nachladen der Bilder wurde hinzugefügt um den „Lade-Traffic“ beim Aufrufen der Webseite zu minimieren.

jQuery frägt als erste ab ob auf der Webseite gescrollt wird wenn das der Fall ist, führ jQuery alle HTML-Tag’s mit „img“ eine Funktion aus. Wenn das HTML-Tag auch noch das Attribut „data-src“ hat und in der Nähe des unter Bildschirm Rands ist wird es geladen. Das Laden der Bilder funktioniert so, jQuery holst sich die Adresse die im „data-src“ vorhanden ist Speicher sie ab und fügt sie in das Attribut „src“ hinein. Sobald das HTML in dem Attribut „src“ eine Adresse zu einem Bild hat, lädt es das Bild in die Webseite hinein. Schlussendlich wird das „data-src“ Attribut gelöscht. Das die Bilder kein „src“ Attribut haben, ergibt das Fehler in der HTML Validierung. *(JavaScript Code 3.5)*

```
// Wenn gescrolt wird
$(window).scroll(function() {

    // Holt sich alle HTML-Tag's mit 'img'
    $.each($('img'), function() {

        // Alle img werden beim Ersten scrollen geladen
        if ( $(this).attr('data-src') &&
        $(this).offset().top < ($(window).scrollTop() + $(window).height()) ) {

            // Holt sich die gespeicherte Source von 'data-src',
            // in 'data-src' liegt der Link zum Bild
            var source = $(this).data('src');

            // Überschreibt did leeren HTML-Tag's 'src' mit den 'data-src' Tag's
            $(this).attr('src', source);

            // Löscht die 'data-src' Tag's
            $(this).removeAttr('data-src');
        }
    })
})
```

JavaScript Code 3.5

Navigationsbar

Damit die Navigationsbar beim herunter scrollen verschwindet braucht es ein JavaScript Funktion. Mit Hilfe des wertes von „window.pageYOffset“ kann die obere Position der Webseite herausgefunden werden, dies wird in die variable „prewScrollpos“ abgespeichert. Wenn auf der Webseite gescrollt wird, wird die Funktion „window.onscroll“ ausgeführt. In dieser Funktion wird die vorherige Position des ober Bildschirmrande mit der neuen Bildschirmposition verglichen wenn die vorige Position grösser ist als die Aktuelle wurde nach oben gescrollt. Damit wird die Navigationsbar angezeigt. Sie wird angezeigt in dem der „style.top“ auf 0px eingestellt wird. Wenn aber nach unten gescrollt wird, wird die Position der Navigationsbar außerhalb der Webseite angezeigt.

(JavaScript Code 3.6)

```
// Navigationsbar nach unten Scrollen lässt die Navigationsbar verschwinden
// nach oben Scrollen lässt die Navigationsbar anzeigen

// Gibt die Position des Y-Offest an
var prevScrollpos = window.pageYOffset;
// Wenn gescrollt wird
window.onscroll = function() {
    // Fragt die neue Y-Offest-Position nach dem Scrollen ab
    var currentScrollPos = window.pageYOffset;
    // Wenn die aktuelle Position unter den Y-Offest der Webseite liegt
    // soll die Navigationsbar am oberen Bild Rand angezeigt werden
    if (prevScrollpos > currentScrollPos) {
        // Lässt die Navigatonsbar erscheine
        document.getElementById("navbar").style.top = "0px";
    }else{
        // Lässt die Navigatonsbar verschwinden
        document.getElementById("navbar").style.top = "-200px";
    }
    // Schreib die aktuelle Position in die vorige Position, um Flackern zu
verbinden
    prevScrollpos = currentScrollPos;
}
```

JavaScript Code 3.6

Ladezeit Analyse

Insgesamt muss für die Webseite zu erstellen 6.1MB an Daten geladen werden. Jedoch fällt das nicht am Anfange an. Beim Aufrufen der Webseite wird 2.1MB geladen, nach dem ersten Scrollen fallen weiter 4MB an. *(Abbildung 6)* Das wird mit Hilfe des „Autoload“ erreicht. *(css Code 3.5)* Das Laden braucht 1.4 s für die Basis-Webseite, darauf folg noch ca. 0.2 s für die restlichen Bilder und am Schluss noch für Google Maps.

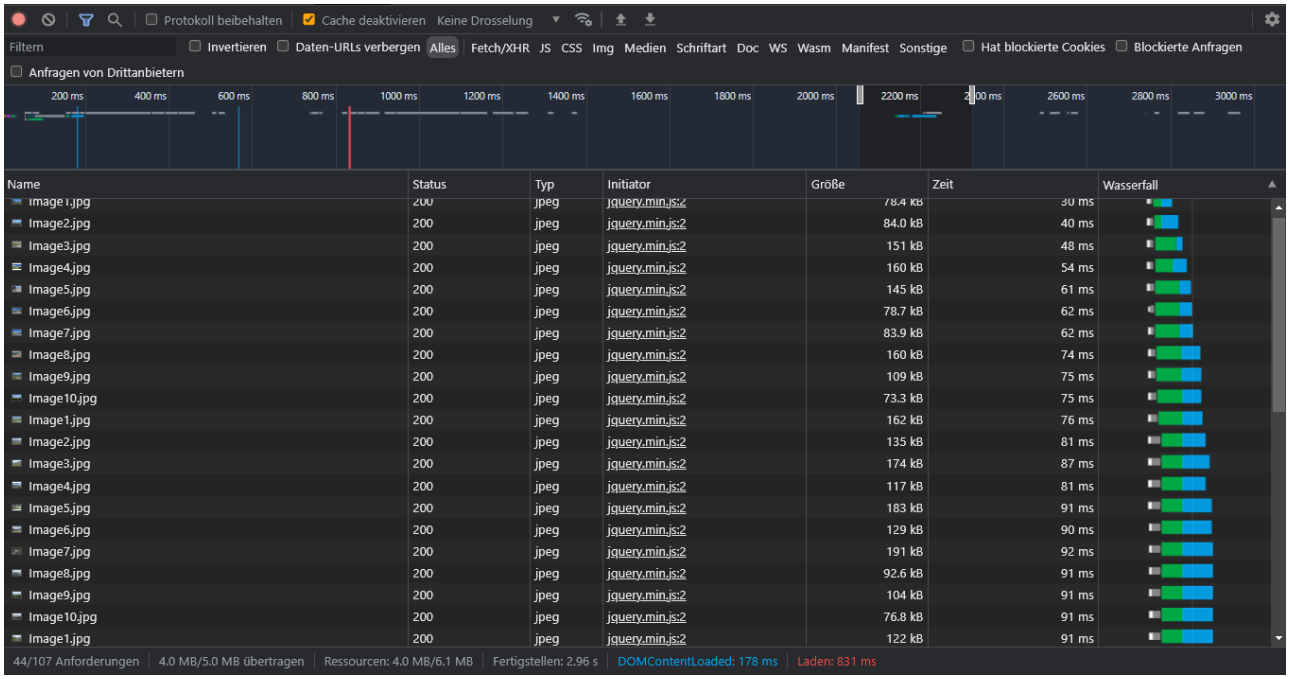

Abbildung 6 Ladezeit Analyse

Kopieret und Übernommene Elemente

Direkt übernommen ist die „Autoload nach Scrollen“, link zum Code: *https://stackoverflow.com/questions/5117421/how-to-load-images-dynamically-or-lazily-when-users-scrolls-them-into-view*

Die Funktion für das Verschwinden der Navigationsbar wurde übernommen und die Parameter wurden so angepasst das es für die Webseite passt. Link zum Code: *https://www.w3schools.com/howto/howto_js_navbar_hide_scroll.asp*

Die Funktion für das Animieren der Bilder auf der Routemap wurde selbst erstellt bis auf die jQuery Funktionen, die sind aus der jQuery Dokumentation. Link zu jQuery: *https://jquery.com/*

Bezüglich css und HTML wurde die Navigationsbar übernommen und stark angepasst. Link zu Code: *https://www.w3schools.com/howto/howto_css_dropdown_navbar.asp*

Wie schon erwähnt wurde auch die Google Map einbinudng direkt übernommen. Link zu Tool: *https://www.checkpoll.de/*

Schlusswort

Die Aufgabe eine eigene Webseite zu erstellen hat mir persönlich sehr viel Spaß bereitet. Ich wollte schon länger mal eine eigne Webseite erstellen jedoch haben ich den Zeitaufwand stark unterschätz und konnte längst nicht alle Funktionen in die Webseite einbauen. Insgesamt habe ich für den Code 44h gebraucht (angaben aus CodeTime, Plugin von vs Code). Dennoch habe ich jetzt eine gute grundlange, um an der Webseite weiterzuarbeiten.