

Quicknotes AB151-06 – 07

Icons von Font Awesome

FontAwesome ist eine Library von Icons, welche entweder heruntergeladen oder als Kit in einem HTML-Dokument eingebunden werden kann. In der Instagram Applikation wurde Font Awesome mit Hilfe des package-installer yarn installiert. So kann innerhalb der Applikation auf die Icons zugegriffen werden. Es wurden passende Icons ausgewählt und bei den Posts angewendet und mit Hilfe weiterer CSS-Klassen wurden die ausgewählten Icons weiter angepasst.

Die Icons werden in HTML als `<i>`-Tag eingebunden, dessen Klasse der Abkürzung des Icons entspricht. Die genauen Tags können direkt von der Website copy-pasted werden.

Beispiel für das Kommentar-Icon: `<i class="far fa-comment"></i>`

Icons von Font Awesome vs Core-Sprite

Ich persönlich präferiere FontAwesome, da man dort keine exakten Pixel-Werte eingeben muss, was umständlich ist. Ausserdem ist die Erweiterung eines Core Sprites ebenfalls aufwendiger als das HTML-Tag zu copy-pasten.

Vorteile Icons von Font Awesome: Font Awesome ist flexibler, die Icons können mehrfarbig und mit anderen individuell bestimmten Eigenschaften dargestellt werden. Die einzelnen Icons sind sehr schnell geladen und brauchen nicht sehr viel Speicherplatz. Einfache freie Positionierung, die Icons können problemlos an der richtigen Stelle eingefügt werden.

Nachteile: Limitierte Auswahl.

Vorteile Core-Sprites: Die Core-Sprites sind meistens kreativer gestaltet. Man hat Zugriff auf alle Icons, die auf einem vorhandenen Bild sind, was beliebig angepasst werden kann. (Mit dem entsprechenden Aufwand)

Nachteile: Für die Core-Sprites, muss eine gesamte Pixel-Grafik gespeichert und geladen werden, was zu längeren Ladezeiten und grösseren Daten führen kann. Es braucht mehr Aufwand, um in den CSS-Klassen die richtige Stelle des Core-Sprites auszuschneiden. Die Core-Sprites nicht im Nachhinein nicht einfach anzupassen.

Lösung zum Anzeigen der Likes:

Da man maximal zwei User anzeigen soll, welche den jeweiligen Post geliked haben, eignet sich eine simple if-else Anweisung zum Anzeigen der Usernamen. Man hätte dies auch mit einem etwas schöneren switch lösen können. Beispielsweise würde man bei zwei Likes den Namen des letzten Users, dem der Post gefiel, anzeigen lassen, gefolgt vom neusten User. Diese Logik scheint auf dem ersten Blick verwirrend zu sein, aber dies hatte ich absichtlich so implementiert, da ich im Post Model keine Änderungen vornehmen wollte. (Bisher wurden die Likes in Posts nach Erstelldatum absteigend sortiert, damit immer der neuste Like angezeigt wurde)

```
<% if likes.size == 1 %>
    <strong><%= likes.first.user.name %></strong> likes this
<% elsif likes.size == 2 %>
    <strong><%= likes.first.user.name %></strong> and <strong><%= likes.second.user.name %></strong> like this
<% elsif likes.size == 3 %>
    <strong><%= likes.first.user.name %></strong>, <strong><%= likes.second.user.name %></strong> and <strong><%= likes.third.user.name %></strong> like this
<% elsif likes.size > 3 %>
    <strong><%= likes.second_to_last.user.name %></strong>, <strong><%= likes.last.user.name %></strong> and <strong><%= likes.size - 2 %></strong> others like this
<% end %>
```

AJAX (Like-Funktion)

Ajax (auch AJAX; Akronym von englisch Asynchronous JavaScript and XML) bezeichnet ein Konzept der asynchronen Datenübertragung zwischen einem Browser und dem Server. Dieses ermöglicht es, HTTP-Anfragen durchzuführen, während eine HTML-Seite angezeigt wird, und die Seite zu verändern, ohne sie komplett neu zu laden.

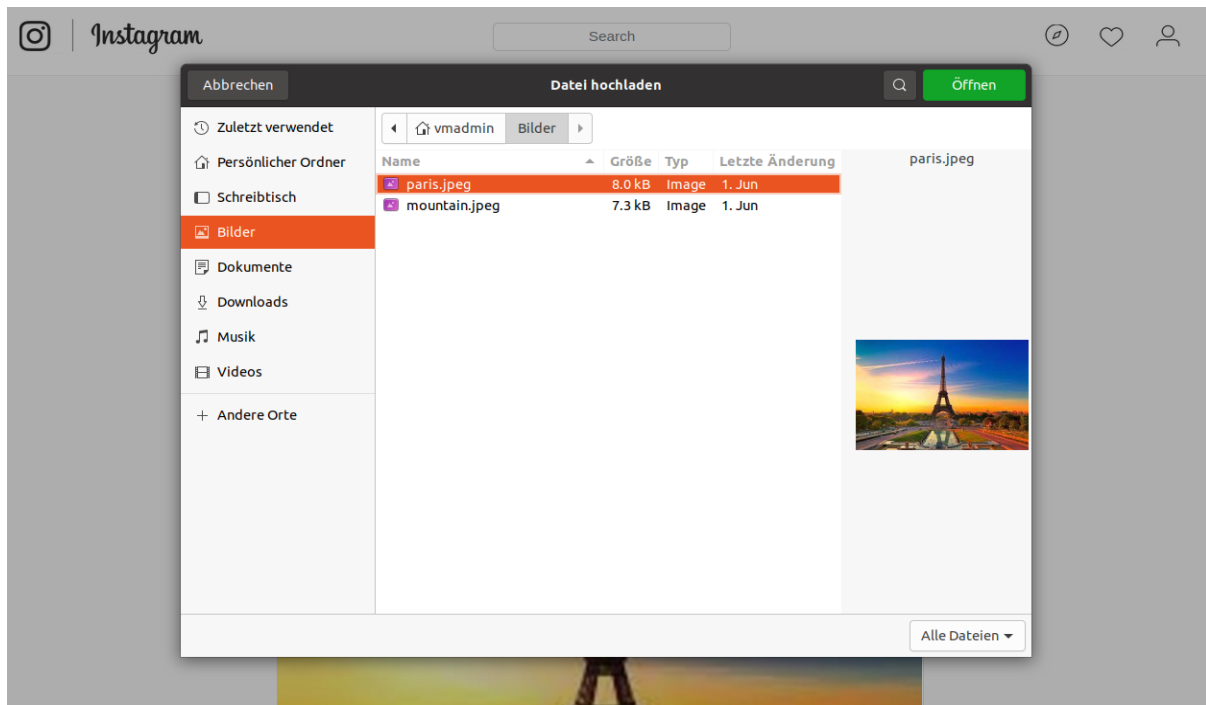
```
$('#like-icon-post-<%= @post.id.to_s %>').
  html('<%= link_to "", like_path(@like), method: :delete, remote: true, class: "fas fa-heart fa-2x icon-red icon-spacer" %>');
$('#like-text-post-<%= @post.id.to_s %>').
  html('<%= j render "posts/like_text", {likes: @post.likes} %>');
console.log("create like success");
```

Vorteile: Es können auf der Webseite dynamische Änderungen durchgeführt werden, ohne dass die jeweiligen Seiten neu geladen werden müssen.


Nachteile: Die AJAX Funktionen sind nicht mit allem Browser gleich kompatibel. Auf älteren Browser oder wenn im Browser Java-Scrip ausgeschaltet wird, kann es auf der Webseite zu Problemen führen.

Screenshots




Fenster, das sich öffnet, um Bilder für den Post auszuwählen:




Liste mit allen Posts:

 | **Instagram**

Search

Create Post



 Say something about this...


Upload



Select pictures


Browse

Post

 **M18** 





Melweezy and M18 like this

M18 Paris2

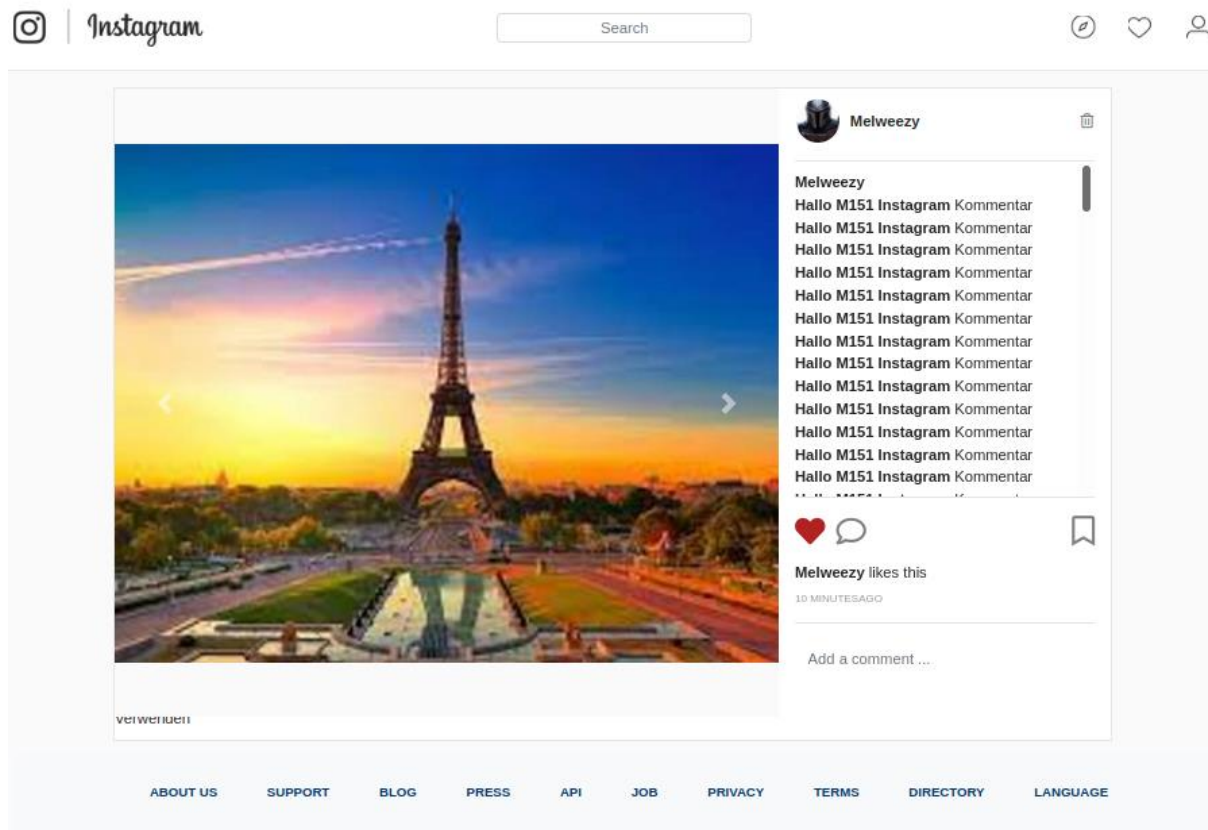
10 DAYS AGO

Add a comment

Fortsetzung:



Detail-Ansicht eines Posts (Post mit mehreren Bildern):



Reflexion

Verhältnis gewusst/nicht gewusst:

Ich kannte FontAwesome bereits zuvor und habe es auch schon in einigen Web-Projekten verwendet. Ansonsten waren mir die verwendeten Technologien fremd.

Vorgehen:

Ich habe die beiden Arbeitsblätter auf die beiden Tage verteilt gelöst. Ich bin dabei den Anweisungen gefolgt und habe bei Interesse, zum Beispiel bei FontAwesome, weiter recherchiert.

Schwierigkeiten:

Während dem Bearbeiten der beiden Arbeitsblätter hatte ich keine Schwierigkeiten. Mein einziges Problem, welches ich hatte, war dass mir ein grober Rechtschreibfehler beim Erstellen des Modells für die Likes unterlief, welcher dazu führte, dass ich auf den letzten Commit zurückkehren musste.

Was habe ich nicht verstanden:

Es gab keine Dinge, die ich nicht mit Googeln oder Fragen verstehen konnte.

Was kann ich nächstes Mal besser machen: Ich hätte mehr Zeit in zusätzliche Recherche stecken können.

Abschliessende Reflexion

Wichtigste neue Technologien:

Ich fand das Gem devise, welches wir zur Authentifizierung verwendet haben sehr spannend, da es eine Komplettlösung ist, welche ermöglicht, schnell eine ziemlich sichere Authentifizierung zu haben. Das Bootstrap-Karussell war ebenfalls interessant, weil es so simpel war, das Aussehen der echten Instagram-Seite zu kopieren, was sehr erleuchtend war. Ausserdem fand ich Figaro interessant, da es eine Möglichkeit bietet, sensitive Informationen im Source-Code zu verstecken.

Vorgehen insgesamt:

Ich bin teilweise sehr schnell vorangegangen, weshalb ich einige der Technologien nicht wirklich verstanden habe, dies lag aber auch daran, dass die Menge an Aufgaben sehr schwierig in dieser Zeit zu bewältigen war. Deshalb habe ich viel einfach kopiert, ohne den Inhalt vollumfänglich zu verstehen. Ausserdem habe ich teilweise zu schnell aufgegeben und lange Zeit mit einem kleinen Problem gekämpft, welches einfach zu lösen gewesen wäre, wenn ich die Aufgabenstellung noch einmal durchgelesen hätte oder auf die nächste Seite gewechselt wäre. Trotzdem habe ich interessante Technologien entdecken können, welche ich auch in Zukunft verwenden werde.

Schwierigkeiten generell:

Die grösste Schwierigkeit war es die Arbeitsblätter richtig durchzulesen, ohne etwas zu übersehen, sonst war der Instagram-Klon nicht besonders schwierig zu implementieren.

Korrektheit der Applikation:

Die Applikation weicht bestimmt in gewissen Aspekten von der Vorgabe ab. Dabei sind das viele kleine Dinge, wie Abstände zwischen Containern und andere kleine Design-Unterschiede. Ich denke die Applikation, bzw. das Dahinter funktioniert korrekt.

Abweichungen von der Vorgabe:

Von der Vorgabe bin ich nie abgewichen.

Was mache ich nächstes Mal besser:

Ich habe einige Dinge mitgenommen im Bereich Sicherheit und Authentifizierung, welche ich bei meiner nächsten Web-Applikation bestimmt mitnehmen kann. Ich kann keine direkte Verbesserung beschreiben, da dieses Projekt nach einer sehr genauen Vorgabe aufgebaut war, welche nach meiner Ansicht ein «Best-Practices-Projekt» war.

Leider reichte es mir zeitlich nicht einige Zusatzaufgaben zu lösen. Nächstes Mal würde ich mir mehr Mühe geben, um beispielsweise noch eine Kommentarfunktion einzubauen.