Assignment 2

DHBW Mosbach, Frontend-Entwicklung, Herr Slezak, HSS 2020, Alischa Thomas

Table of Contents

[Projektvorhaben 1](#_Toc39868148)

[Vorkenntnisse und Vorbereitung 1](#_Toc39868149)

[Dokumente und Klassen 3](#_Toc39868150)

[Dokument- und Klassenstruktur 3](#_Toc39868151)

[Dokumentenfunktionalität 3](#_Toc39868152)

[Klassenfunktionalität 3](#_Toc39868153)

[Probleme und Hindernisse 3](#_Toc39868154)

[Persönliche Reflexion 3](#_Toc39868155)

[Fazit 3](#_Toc39868156)

[Literaturverzeichnis 3](#_Toc39868157)

# Projektvorhaben

Ziel des Projektes ist die Entwicklung einer funktionalen Typescript-Applikation, die im Browser durch die Kompilierung des verwendeten JavaScript über eine *index.html* verwendet werden kann. Die Art der Applikation liegt dabei im Entscheidungsfreiraum der Autorin. Diese entschied sich für ein Registrierungsformular, das einen *Usernamegenerator*, sowie einen *Password-Checker*. Diese Applikation *form-validator* wurde aus vom Dozierenden empfohlenen Liste mit Projekten ausgewählt(Traversy, 2020/2020). Dieses wurde lediglich zur Ideenfindung genutzt. Es wurde sich allerdings nicht an dem darin verwendeten Code orientiert, sondern ein eigener Ansatz frei von dem Beispielprojekt entwickelt. Das Vorhaben wurde außerdem mit dem Dozierenden abgeklärt und von diesem bestätigt.   
Der Usernamegenerator aus dem zuvor eingegebenen Uservor- und nachnamen inklusive einer zufällgi erzeugten Zahl einen Usernamen. Der Password-Checker überprüft zwei eingegebene Passwörter auf Similarität. Die Überprüfung wird durch zwei Aktionen der NutzerInnen ausgeführt: Verlassen des Feldes, sowie Klicken des Registrierungsbuttons. Dieser Button führt vor dem Absenden der Formulars die Passwortüberprüfung statt. Ist diese erfolgreich wird kurz vor dem Absenden ein entsprechender Text auf der NutzerInnenoberfläche ausgegegeben und das Formular wird abgesendet. Ist die Überprüfung nicht erfolgreich verliert der Button seine Funktion Daten zu übermitteln und teilt der NutzerInnen mit, dass sie die Passworteingabe korrekt wiederholen sollten um die Registrierung abzuschließen.  
Kriterien für die Entwicklung der App sind die Verwendung von Typescript (*folgend TS*), was die Anwendung typescirpttypischer Features wie z.B. Typisierung mit einschließt (*Angaben in Moodle*). Aus den in der Vorlesung genannten Kriterien geht hervor, dass Interfaces oder Klassen verwendet werden sollen. Des Weiteren sollen die verschiedenen TS-Dokumente mit einem Modulebundler gebundled werden.

# Vorkenntnisse und Vorbereitung

Die Vorkenntnisse zum Thema TS belaufen sich auf die theoretischen Vorlesungsinhalte, die zuvor vom Dozierenden vermittelt wurde. Da TS auf der Logik von Javascirpt und Java aufbaut sind zumindest Grundkenntnisse vorhaben. Jedoch müssen die Kenntnisse über entsprechend um die spezielle TS-Syntax und TS-features erweitert werden. Dazu wurde hauptsächlich ein sehr ausführliches Videotutorial mit zusätzlichen Übungen verwendet (*TypeScript Course for Beginners 2020*, n.d.). Dadurch konnte das theoretische Wissen um einen ersten Überblick in der praktischen Anwendung erweitert werden. Dies erleichterte den Einstieg in das praktische Aufsetzen des Projektes immens und ist sehr weiterzuempfehlen für TS-EinsteigerInnen.  
Um einen ersten Überblick über das Vorhaben zu erhalten wird im ersten Schritt eine größtenteils funktionslose HTML-Datei inklusive zugehörigem CSS-Dokument erstellt. Die Kenntnisse im grundlegenden Umgang mit HTML und CSS sind an diesem Punkt des Studiums bereits verfestigt. Trotzdem bedurfte es einiger kleiner Nachrechercher in bezug auf den Aufbau eines Formulars mittels Formular- und Input-tags (*Html - Background Text in Input Type Text*, n.d. ) (*Change Input Placeholder Color*, n.d.)(*HTML Input Required Attribute*, n.d.) Damit werden die Bestanteile des Projektes in ihren Grundzügen gelegt und im nächsten Schritt kann ihnen eine Funktion mit Hilfe von TS zugeordnet werden.   
Damit diese Funktionen auch abgerufen werden können wird der TypeScript-Compiler im Projekt installiert. Darüber hinaus wird das Projekt mit einem *lite-server*, ein zur freien Verfügung stehender Server für Entwicklungsvorhaben, verbunden und gleichzeitig eine automatische Synchronisierung zwischen diesem und den Dateien des Projektes eingebaut. Dazu werden die entsprechenden Anpassungen in der package.json Datei im TS-Projekt gemacht. Dies kostet im ersten Schritt Zeit, erspart aber das händische Kompilieren und Aufrufen im browser bei jeder Änderung während des Projektes (*TypeScript Course for Beginners 2020*, n.d.).  
Um mit Modulen arbeiten zu können und diese miteiannder bei der Kompilierung zu verbinden wird außerdem der Modulebundler rollup.js installiert und eingebunden. Anschließend wird seine Konfiguration in der rollup.config.js-Datei so angepasst, dass bei jeder Kompilierung Datei index.ts zu einem kombinierten bundle.js-Datei wird. Diese kann dann vom Browser ausgelesen und somit die Funktion ausgesteuert werden. Des Weiteren werden in der ts.config-Datei die Root- und Sourceordner für die Kompilierung angegeben, sowie der Version des im Projekt angewandten ECMAScript auf ES6 eingestellt.  
Mit diesen Grundeinstellungen als Basis können im nächsten Schritt die Funktionalitäten der App-elemente mit TS implementiert werden.

Ansehen Projekte bei <https://github.com/bradtraversy/vanillawebprojects>

<https://github.com/MysterieDev/TypeScriptAssignment>

<https://developer.mozilla.org/en-US/docs/Learn/Forms/Form_validation>

<https://rollupjs.org/guide/en/>

<https://www.learnwithjason.dev/blog/learn-rollup-js/>

<https://medium.com/@djoepramono/how-to-validate-javascript-object-better-with-typescript-e43314d97f9c>

wie binde ich Module zusammen (import/export)

<https://www.typescriptlang.org/docs/handbook/modules.html>

Typen: <https://www.typescriptlang.org/docs/handbook/basic-types.html#number>

* Entscheidung für Form-Validator
* Rückmeldung von Dozent: usätzliche features password validator + username generator
* Warum: Realitätsnahe, funktionale Nutzung, die ich wahrscheinlich wiederverwenden kann
* Motivation: relativ gestyltes Produkt, steigert meine Motivation etwas zu schaffen.

Plan:   
Formular

* Name
* Vorname
* Username (generated aus name/vorname+zahl)
* Password
* Passwordcheck
* Registrierungsbutton

Anfangs: schwierigkeiten zu verstehen wo ich dort nun typescript unterbrigen kann weil man das auch über normale html-pattern und normale code snippets in css lösen könnte. Bei so einem kleinen projekt scheinen klassen etwas überflüssig, aber okay.

05.05: Meeting Malte Weiersbach: hat mit mir den code angesehen und geschaut wie man ts dort einbringen könnte.

Einbinden Live-Server: automatische Browsersynchronisierung

Aufbau Usernamegenerator: Input Vorname + Input Nachname + Randomnumber  
  
Problems: understanding how function and variables work together and how I can grasp the input of input fields and show my results to an HTML-input field.

Only anable letters in input field würde man eig mit HTML machen: <input onkeyup = „lettersonly(this)”>.

Lifeserver + configuration in meiner IDE +. Damit verbundene änderungen in den config dateien.

# Dokumente und Klassen

## Dokument- und Klassenstruktur

Klassenstruktur: Begründung weshalb so granulare Klassen + mit Rücksprache mit Malte: macht man in so kleinen Projekten nicht. Würde über html laufen.

<https://stackoverflow.com/questions/12930049/how-do-i-import-other-typescript-files>

<https://www.typescriptlang.org/docs/handbook/modules.html>

## Dokumentenfunktionalität

<https://www.typescriptlang.org/docs/handbook/modules.html>

Verbdinung HTML-Dokument mit Klassen (ts): <https://stackoverflow.com/questions/7674194/how-to-get-innerhtml-of-this-element-in-javascript>

<https://developer.mozilla.org/en-US/docs/Web/API/HTMLInputElement>

## Klassenfunktionalität

<https://www.tutorialspoint.com/typescript/typescript_variables.htm>

<https://stackoverflow.com/questions/34110766/how-to-show-hide-multiple-divs-based-on-input-value>

Aufbau Klassen: Konstruktoren etc.

<https://ultimatecourses.com/blog/typescript-classes-and-constructors>

<https://developer.mozilla.org/en-US/docs/Web/API/HTMLInputElement>

<https://www.typescriptlang.org/docs/handbook/functions.html>

<https://www.typescriptlang.org/docs/handbook/modules.html>

welche events gibt es? <https://www.w3schools.com/jsref/dom_obj_event.asp>

<https://www.w3schools.com/jsref/event_oninput.asp>

MethodsPasswordCheck

<https://stackoverflow.com/questions/14702190/to-show-error-message-without-alert-box-in-java-script>

<https://stackoverflow.com/questions/38639604/how-do-i-activate-deactivate-css-on-button-click>

<https://stackoverflow.com/questions/21070101/show-hide-div-using-javascript>

ApplyPasswordCheck

<https://stackoverflow.com/questions/16117290/display-the-result-of-a-javascript-function-in-a-div-element/16117312>

<https://stackoverflow.com/questions/48100152/hide-elements-using-typescript/48100271>

<https://developer.mozilla.org/en-US/docs/Web/API/HTMLElement/hidden>

<https://stackoverflow.com/questions/24834595/hide-html-element-using-javascript>

wie wird submittet: damit dozent auch sachen sehen kann etc:

<https://developer.mozilla.org/en-US/docs/Web/API/HTMLFormElement/submit_event>

MethodsUsernameGenerator

Inspiration:

<https://stackoverflow.com/questions/1497481/javascript-password-generator>

<https://medium.com/@josephcardillo/using-math-random-in-javascript-c49eff920b11>

ApplyUsernameGenerator

<https://www.w3schools.com/jsref/prop_text_disabled.asp>

# Probleme und Hindernisse

Zu Beginn nicht klar wie TS und HTML interagieren, da Komponenten von Dozierendenbeispiel verwirrend. Jetzt klar, dass TS quasi nur die Funktionen Modularisiert.

Konnte node modules nicht finden.

<https://intellij-support.jetbrains.com/hc/en-us/community/posts/207297985-Can-t-see-excluded-folders>

<https://stackoverflow.com/questions/26107125/cannot-read-property-addeventlistener-of-null>

<https://stackoverflow.com/questions/12930049/how-do-i-import-other-typescript-files>

<https://stackoverflow.com/questions/16117290/display-the-result-of-a-javascript-function-in-a-div-element/16117312>

<https://developer.mozilla.org/en-US/docs/Web/API/HTMLInputElement>

<https://stackoverflow.com/questions/52325814/why-we-are-using-htmlinputelement-in-typescript>

Zu Beginn: Probleme mit Klassen erstellen: <https://ultimatecourses.com/blog/typescript-classes-and-constructors>

Verständnis Variablen:

<https://www.tutorialspoint.com/typescript/typescript_variables.htm>

Erkenntnis mit private und public etc.

<https://stackoverflow.com/questions/34110766/how-to-show-hide-multiple-divs-based-on-input-value>

# Persönliche Reflexion

Der häugiste Satz, den ich während dieses Assignments gehört habe war „ das ist eigentlich JAvascript, nur typisiert“. Was auch stimmt, bis es zu den 100 Ausnahmen kommt, die es in Typescript gibt…

# Fazit

# Literaturverzeichnis

Interfaces

<https://www.logicbig.com/tutorials/misc/typescript/type-assertions.html>

<https://www.youtube.com/watch?v=cYBDtNnjmlA>

Backgroundbild: <https://stock.adobe.com/de/search/images?filters%5Bcontent_type%3Aphoto%5D=1&filters%5Bcontent_type%3Aillustration%5D=1&filters%5Bcontent_type%3Azip_vector%5D=1&filters%5Bcontent_type%3Avideo%5D=0&filters%5Bcontent_type%3Atemplate%5D=0&filters%5Bcontent_type%3A3d%5D=0&filters%5Bcontent_type%3Aimage%5D=1&filters%5Binclude_stock_enterprise%5D=0&filters%5Bis_editorial%5D=0&order=relevance&price%5B%24%5D=1&safe_search=1&serie_id=302616645&search_page=1&search_type=see-more&get_facets=0&asset_id=302616645>