3D Portfolio Gallery

DIPLOMARBEIT

verfasst im Rahmen der

Reife- und Diplomprüfung

an der

Höheren Abteilung für IT - Medientechnik

Eingereicht von:
Ema Halilovic
Lorenz Litzlbauer
Fabian Maar

Betreuer:
Patricia Engleitner
Natascha Rammelmüller

Leonding, April 2022

Ich erkläre an Eides statt, dass ich die vorliegende Diplomarbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt bzw. die wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe.

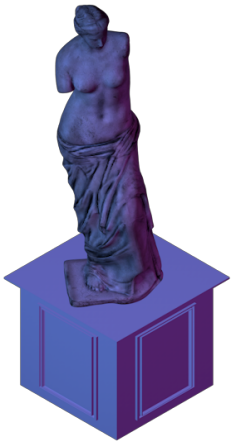
Die Arbeit wurde bisher in gleicher oder ähnlicher Weise keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Die vorliegende Diplomarbeit ist mit dem elektronisch übermittelten Textdokument identisch.

Leonding, April 2022 S. Schwammal & S. Schwammal

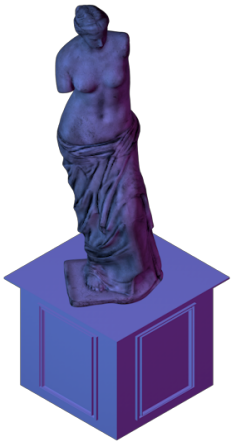
Abstract

3D Portfolio Gallery is a web application that was developed by Ema Halilovic, Lorenz Litzlbauer and Fabian Maar as part of their diploma thesis. 3D Portfolio Gallery wants to enable designers to share their design portfolio with the world in an innovative way. The designers can use the 3D Portfolio Gallery to create a 3D portfolio from their own media (film, photo or 3D data), which interested parties can view on the website with various types of interaction in 3D space. For this purpose, Angular is used in the frontend for the SPA (single-page application) logic, ThreeJs for the 3D display and technologies such as Quarkus, Maven, JPA, Panach and Hibernate ORM are used for the server. To ensure user security, we use JWT (JsonWebToken)



Zusammenfassung

3D Portfolio Gallery ist eine Webapplikation, die von Ema Halilovic, Lorenz Litzlbauer und Fabian Maar im Rahmen der Diplomarbeit entwickelt wurde. 3D Portfolio Gallery will Designer*innen ermöglichen, ihr Design-Portfolio auf eine innovative Art und Weise mit der Welt zu teilen. Die Designer*innen können mithilfe von 3D Portfolio Gallery aus eigenen Medien (Film, Foto oder 3D - Daten) ein 3D Portfolios erstellen, welche sich Interessierte mit verschiedenen Interaktionsarten im 3D Raum auf der Webseite anschauen können. Dafür werden im Frontend Angular für die SPA (Single-Page-Applikation) - Logik benutzt, für die 3D-Darstellung ThreeJs und für den Server werden Technologien wie Quarkus, Maven, JPA, Panach und Hibernate ORM verwendet. Um die Sicherheit der User zu gewährleisten, verwenden wir JWT (JsonWebToken)



Inhaltsverzeichnis

1 Einleitung **1**

2 Umfeldanalyse **2**

3 Aufgabendifferenzierung **3**

3.1 Lorenz 3

3.2 Fabian 3

3.3 Ema 3

4 Technologien **4**

4.1 Angular 4

4.2 Webstorm 5

4.3 ThreeJs [L] 6

4.4 Cinema 4D 9

5 Umsetzung der Interaktions-Seite **10**

5.1 Border-Collision 10

6 Umsetzung **12**

6.1 Plannung 14

6.2 Design 14

7 Zusammenfassung **15**

Literaturverzeichnis **VI**

Abbildungsverzeichnis **VII**

Tabellenverzeichnis **VIII**

Quellcodeverzeichnis **IX**

1 Einleitung

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Ich bin toll einzigartig und schön.

2 Umfeldanalyse

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. Citing [**?**] properly.

Was ist eine GUID? Eine GUID kollidiert nicht gerne.

Kabellose Technologien sind in abgelegenen Gebieten wichtig [**?**].

3 Aufgabendifferenzierung

3.1 Lorenz

3.2 Fabian

3.3 Ema

4 Technologien

4.1 Angular

4.1.1 Allgemeines [M]

Angular ist ein Framework für Webapplikation, das auf der Programmiersprache Typescript basiert. Es ist eines der renommiertesten Frameworks zur Front-End-Entwicklung und wird als Open-Source-Software zur Verfügung gestellt. Besonders bietet sich Angular für Single-Page-Webanwendung an, da es ein komponentenbasiertes Framework ist. Das bedeutet, der Code ist wiederverwendbar und verkapselt. Komplexe Logiken werden auf ihre Grundelemente reduziert und beeinflussen sich nicht gegenseitig.

4.1.2 Dependency [L]

Um eine nachvollziehbare Auswahl zu treffen, muss verstanden werden, wie Angular funktioniert. In den folgenden Unterabschnitten wird sich mit den Technologien auf die Angular aufgebaut ist, auseinandergesetzt.

RxJS

ReactiveX ist eine Library für das Erstellen von asynchronen und Event-basierenden Programmen, dafür benutzt es *observable sequences*. Die Library erweitert so, dass *Observer-Design-Pattern* mit verschiedenen neuen Operatoren und händelt dabei "Lowlevel"-Funktionen wie Threading, Synchronisation, Thread-Sicherheit und das nicht-blockieren von I/O vergleiche [**?**]

RxJS ist eine Implementierung von ReactiveX für die Programmiersprache Javascript. Angular verwendet RxJS für reaktive Programmierart.

Webpack

Die Hauptfunktion von Webpack ist es, viele verschiedenen Daten zu einem Paket für eine JavaScript Applikation zusammenzufassen, dabei optimiert es die Dateigröße,

indem es mithilfe von einem selbst generierten *dependency-graph* die Abhängigkeiten der Applikation überprüft und nur notwendige Teile für die JavaScript-Applikation der Daten nimmt. Bei der Zusammenfassung werden Sprachen, die der Webbrowser nicht Interpretieren kann wie Typescript, Sass uvm. in die für den Webbrowser verständlichen Gegenstücke gewandelt. Webpack hat aber auch viele andere Features. Vergleiche [**?**] Angular benutzt Webpack um TypeScript in JavaScript und Sass bzw. Scss in Scss zu wandeln, um beim Bauen des Projektes alle Module in ein einziges zusammenzufassen und um die Applikation bei der Entwicklung zu *Live-Reloading* zu unterstützen. Dabei wird bei einer Änderung im Code die gesamte Applikation geupdatet und neu gestartet.

4.1.3 Vorteile/Auswahlkriterien [M]

Dieses komponentenbasierte Programmieren war eines der vielen Gründe warum Angular die beste Option ist. Im Abschnitt 5 //TODO wird nochmals deutlich, wie dieser Aspekt genutzt wird. Ein weiterer Grund, ist die einfache Einbindung von Libraries und deren Übersichtlichkeit, die durch ngModules ermöglicht werden. Die ständige Erweiterung und die Unterstützung von Third-Party-Libraries, die unter anderem für die 3D-Darstellung verwendet werden, wird ebenfalls von diesen ngModulen ermöglicht. Beispiele wie wir dieses ngModule aufgesetzt haben befinden sich im Abschnitt Landing-Page aufsetzen //TODO VERLINKEN. Auch aufgrund von persönlichen Erfahrungen im Unterricht und in anderen Projekten war Angular die beste Auswahlmöglichkeit.

4.2 Webstorm

Webstorm ist eine Entwicklungsumgebung vom Unternehmen JetBrains, die sich auf die Programmiersprache JavaScript spezialisiert hat. Ebenfalls unterstützt es besonders Frameworks wie Angular. Das Webstorm besonders auf Angular optimiert wurde, zeigt sich durch viele Features, die das Entwickeln von Angular-Projekten erleichtern. So macht es Webstorm möglich, mit nur wenigen Mausklicks eine neue Angular-Dependency oder Komponente zu erstellen. Auch wird die Entwicklungszeit durch intelligente Code-Vervollständigung, Code-Formatierung, einfache Navigation und viele weitere hilfreiche Features deutlich verkürzt.

4.3 ThreeJs [L]

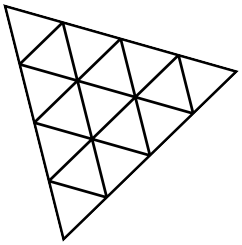


Abbildung 1: ThreeJs Logo

ThreeJs ist eine JavaScript Library für die Darstellung von 3D-Grafiken im Web. Für die 3D-Darstellung nutzt ThreeJs meistens WebGL (mehr dazu im nächsten Abschnitt ThreeJs Dependency), ein low-level Framework. WebGl hat eine hohe Komplexität. ThreeJs bietet eine Abstraktion, bei der es viele Sachen wie die 3D-Szene, Lichter, Schatten, Materialien, Texturen und 3D-Mathe händelt, um die 3D-Darstellungen im Web einfacher zu gestalten. In ThreeJs werden Geometrie, Objekte und Materialien verbunden, um ein 3D-Objekt zu erstellen. Dabei kann die Struktur einer Szene der Abbildung 2 ähneln. Vergleiche [**?**, ThreeJs fundamentals]

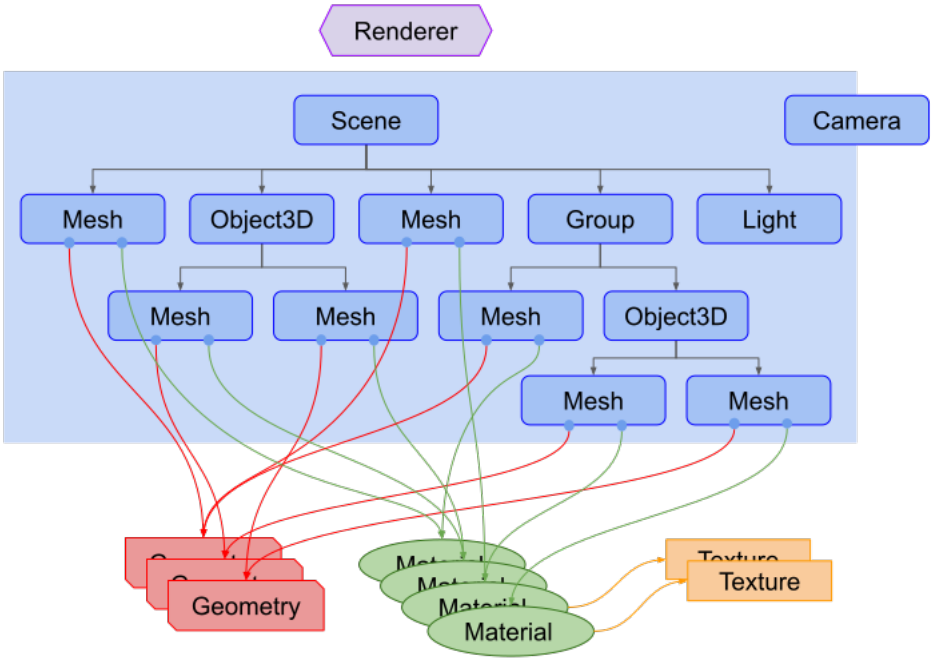


Abbildung 2: Die Struktur von ThreeJs

4.3.1 Dependency

Um einen Großteil der 3D-Darstellungen zu rendern, benutzt ThreeJs WebGL.

WebGL

Abbildung 3: ThreeJs Logo

WebGl ist eine lizensfreie Api, die dafür benutzt wird 3D-Grafiken im Web darzustellen. Sie basiert auf OpenGL

ES 2.0 und benutzt auch dieselbe shading language GLSL.
WebGL ist eine Low-Level-API, das bedeutet, dass schon sehr einfache Projekte einen hohen Programmieranteil haben. Vergleiche[?, WebGl Getting Started]

4.3.2 Auswahlkriterien

Es gab verschiedene Auswahlkriterien für die 3D-Web-API.

- Effizienzen der 3D Engine (Hardware Acceleration, RAM-Auslastung)
- Benutzerfreundlichkeit (Programmerexperience)
- Zusätzliche Features
- Die Features des Projektes müssen damit umsetzbar sein
  1. Das Laden von 3D-Modellen aus 3D-Dateinen und aus dem Internet
  2. Video-Texturen support

Alternativen 3D Web Apis

Es gibt viele Technologien, die 3D-Grafiken im Web ermöglichen, wie Three.js, Babylon.js, A-Frame, X3DOM und WebGL

A-Frame A-Frame wird von der Morzilla Foundation als OpenSource-Projekt entwickelt. Die 3D-Szene wird durch eine deklarative Sprache mit XML-Syntax definiert. Über die WebVR-API bietet die Libray auch die Möglichkeit, die 3D-Szenen durch eine VR-Brille zu erfahren. Bei der 3D-Darstellung setzt A-Frame auf ThreeJs. [?, A-Frame Wikipedia]

A-Frame hat ähnliche Funktionen und Leistung im Vergleich zu ThreeJs, da es ja schlussendlich darauf aufbaut. A-Frame sticht bei der VR support hervor (ThreeJs hätte auch eine Unterstützung dafür, diese ist aber schwieriger einzubinden), doch ist das kein unbedingt nötiges Feature. ThreeJs ist ja bereits eine Abstraktion von WebGL. Für das Projekt wird keine weiter Abstraktion gebraucht.

WebGL Im Vorherigen Kapitel wurde sich bereits mit WebGL befasst. 4.3.1 Um nur eine einfache 3D-Szene in WebGL darzustellen, muss sehr viel Code geschrieben werden. Denn WebGL übernimmt keine Low-Level-Funktion. Es müssen Matrixrechnungen für die Transformationen von 3D-Objekt Manuel und Vertex buffers, die die Daten der

Vertex Positionen, Normaldaten, Farben und Texturen, selbst programmiert werden und das in einer eigenen Programmiersprache. Deshalb war WebGL keine Option für das Projekt.

Angular Three

Angular Three ist ein Open Source Projekt von Matt DesLauriers. Es zielt darauf ab die Vorteile von Angular und ThreeJs zu kombinieren. Dabei verbindet es das Prinzip der Komponenten von Angular mit der 3D Darstellung von ThreeJs.

Listing 1: Angular Three - Komponentenbasiertes 3D Scenen in HTML

```
<ngt-canvas>
    <ngt-ambient-light intensity="0.5"></ngt-ambient-light>
    <ngt-spot-light [position]="10" angle="0.15" penumbra="1"></ngt-spot-light>
    <ngt-point-light [position]="-10"></ngt-point-light>

    <app-cube [position]="[1.2, 0, 0]"></app-cube>
    <app-cube [position]="[-1.2, 0, 0]"></app-cube>

    <ngt-soba-orbit-controls></ngt-soba-orbit-controls>
</ngt-canvas>
```

Listing 2: Angular Three - App Cube

```
<ngt-mesh
  (beforeRender)="onCubeBeforeRender($event)"
  (click)="active = !active"
  (pointerover)="hovered = true"
  (pointerout)="hovered = false"
  [scale]="active ? 1.5 : 1"
  [position]="position"
>
  <ngt-box-geometry></ngt-box-geometry>
  <ngt-mesh-standard-material [color]="hovered ? 'turquoise' :
      'tomato'"></ngt-mesh-standard-material>
</ngt-mesh>
```

Ein Vorteil von Angular Three ist, dass durch nur wenige Zeilen Code 1 eine 3D-Szene mit Lichtern und Orientierungsfunktionen erstellt werden kann und die Business-Logik, App-Cube 2 durch die Verwendung einer Komponente ausgelagert werden kann.

Ein weiterer Vorteil von Angular Three ist die ausführliche Dokumentation mit Codebeispielen. (link) Angular Three Dokumentation https://angular-three.netlify.app/docs/getting-started/overview

Wegen der vielen Vorteile hohe Benutzerfreundlichkeit, ähnliche 3D-Leistung zu ThreeJs (Angular Three basiert auf ThreeJs, welches selbst die WebGL-Renderengine benutzt) und wegen der Verbindung von Angular und ThreeJs Features bietet sich die Liberay für das Projekt an.

Mithilfe eines Prototypen (mehr dazu im Abschnitt fortlaufendes Prototyping) wurde getestet, ob Angular Three alle Features, die für das Projekt benötigt werden, unterstüt-

zen kann, die für das Projekt nötig sind. Dabei wurde festgestellt, dass sich die Liberay Angular für das Projekt nicht eignet. Denn beim Prototyping sind Fehler aufgetreten. Beim erneuten Laden von 3D-Objekten aus derselben Datei haben sich keine Daten mehr im Objekt befunden. ThreeJs Module funktionierten Teilweise nicht, wie die FristPersonControls.

Weil Angular Three nicht die für das Projekt aufgestellten Anforderungen erfüllt hat, wurde der Prototyp zu ThreeJs migriert, weil sich die Syntax ähnelt und dadurch nur wenig Aufwand bei der Migrierung entstand. Diese 3D-Library wurde schlussendlich auch im Projekt verwendet.

4.4 Cinema 4D

Cinema 4D ist ein professionelles Programm vom Unternehmen Maxon zur 3D-Modellierung und Animation. Zum einen ist die Software leicht zu bedienen und zum anderen bietet Maxon eine Vielzahl an Tutorials und Anleitungen. Auch aufgrund von persönlichen Erfahrungen im Unterricht wurde das Programm zur Erstellung von diversen 3D-Modellen benutzt. Schon bereits in der Vergangenheit modellierte Werke konnten sich zu Nutze gemacht und teilweise auch im 3D-Raum verwendet werden. Ein weiterer wichtiger Aspekt ist, dass Cinema 4D den Export von GLTF-Files unterstützt. GLTF ist das Dateiformat, das genutzt wird, um ein 3-Modell in der Three.js Szene zu laden. Da ähnliche Programme wie zum Beispiel Blender diese Exportmöglichkeit nicht anbieten, ist Cinema 4D die beste Wahl für die 3 Modellierung.

5 Umsetzung der Interaktions-Seite

5.1 Border-Collision

Um den Benutzern und Benutzerinnen ein möglichst realistisches Gefühl für die Ausstellung zu geben, ist es wichtig, den Ausstellungsraum auch so zu gestalten. Daher ist der Raum durch Wände abgegrenzt, wodurch es nicht mehr möglich ist, sich über den Raum hinaus zu bewegen. Um eine Kollision mit der Wand zu berechnen, gibt es durch die Three.js Bibliothek einige Möglichkeiten.

5.1.1 Erster Ansatz

Eine davon ist, eine Bounding Box zu erstellen. Dabei unterscheidet man zwischen zwei verschiedenen Arten. Zum einen verwendet man die Axis Aligned Bounding Box, um eine Box rund um das 3D-Objekt zu erstellen, die sich nicht an die Rotation des Objekts anpasst. Die zweite Variante ist die Oriented Bounding Box, die im Endeffekt gleich funktioniert, sich aber darin unterscheidet, dass sie sich an die Achsen des Objekts anpasst. Da sich die Bounding Boxen jedoch über das ganze 3D-Objekt erstrecken, wird eine Kollision direkt berechnet, nachdem der*die Benutzer*in den Raum betritt. Da sich die Bounding Boxen nicht an jede einzelne Wand anpassen konnten, musste ein anderer Lösungsweg gefunden werden.

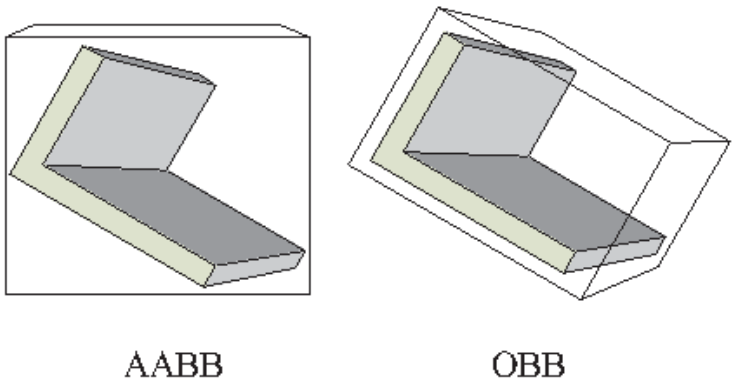


Abbildung 4: Der Unterschied zwischen AABB und OBB

5.1.2 Zweiter Ansatz

Um eine Veränderung der Position des*der Benutzers*in festzustellen, muss die Veränderung der Kameraposition evaluiert werden. Dabei wird beim initialisieren der Kamera eine Kopie von ihr erstellt. In der Animate-Funktion wird eine Veränderung der Kamera überprüft, indem die Positionen der Kamera mit der Kopie verglichen werden. Die Kamera nimmt dabei immer eine neue Position ein, wenn sich der*die Benutzer*in im Raum bewegt, während die Kopie dabei die alte Position der Kamera einnimmt. Jedesmal wenn eine Veränderung geschieht, wird überprüft, ob die Kamera mit der Wand kollidiert. Dies geschieht, indem ein Raycast mit den Positionen der Kamera und der Kopie initialisiert wird.

5.1.3 Far und Near

Die Attribute Far und Near werden dafür verwendet, um die Objekte, die im Ray liegen, einzugrenzen. Dabei können die Werte nicht negativ sein und der Far-Wert muss größer als der Near-Wert sein. Um die Kollision erst direkt am Ursprungsort, im Falle der Kamera, des Rays zu berechnen, wird der Far-Wert auf 100 gesetzt.

Nachdem der Raycast korrekt initialisiert und angewandt wurde, muss bei einer Berührung mit der Wand nur noch richtig damit umgegangen werden. Dabei wird die Bewegung des Benutzers gestoppt. Um diese auch wieder zu starten, muss sich der*die Benutzer*in weg von der Wand begeben. Überprüft wird dies nach jeder Benutzer*inneneingabe mit einem Event-Listener. Falls nach dieser keine Berührung mehr mit einer Wand besteht, wird die Bewegung fortgesetzt und der*die Besucher*in kann sich wieder frei im Raum bewegen.

6 Umsetzung

Siehe tolle Daten in Tab. 1.

Siehe und staune in Abb. 5. Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.

Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Donec odio elit, dictum in, hendrerit sit amet, egestas sed, leo. Praesent feugiat sapien aliquet odio. Integer vitae justo. Aliquam vestibulum fringilla lorem. Sed neque lectus, consectetuer at, consectetuer sed, eleifend ac, lectus. Nulla facilisi. Pellentesque eget lectus. Proin eu metus. Sed porttitor. In hac habitasse platea dictumst. Suspendisse eu lectus. Ut mi mi, lacinia sit amet, placerat et, mollis vitae, dui. Sed ante tellus, tristique ut, iaculis eu, malesuada ac, dui. Mauris nibh leo, facilisis non, adipiscing quis, ultrices a, dui.

	Regular Customers	**Random Customers**
Age	20-40	>60
Education	university	high school

Tabelle 1: Ein paar tabellarische Daten

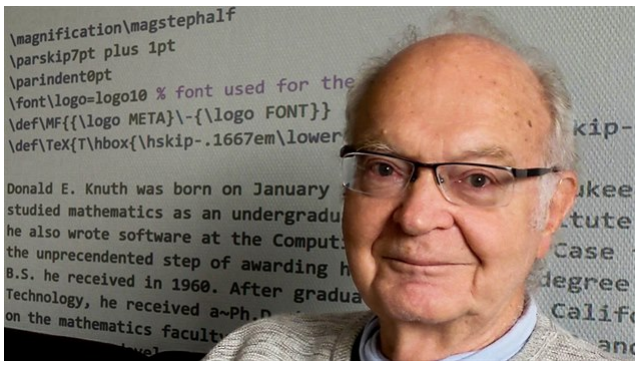


Abbildung 5: Don Knuth – CS Allfather

Morbi luctus, wisi viverra faucibus pretium, nibh est placerat odio, nec commodo wisi enim eget quam. Quisque libero justo, consectetuer a, feugiat vitae, porttitor eu, libero. Suspendisse sed mauris vitae elit sollicitudin malesuada. Maecenas ultricies eros sit amet ante. Ut venenatis velit. Maecenas sed mi eget dui varius euismod. Phasellus aliquet volutpat odio. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Pellentesque sit amet pede ac sem eleifend consectetuer. Nullam elementum, urna vel imperdiet sodales, elit ipsum pharetra ligula, ac pretium ante justo a nulla. Curabitur tristique arcu eu metus. Vestibulum lectus. Proin mauris. Proin eu nunc eu urna hendrerit faucibus. Aliquam auctor, pede consequat laoreet varius, eros tellus scelerisque quam, pellentesque hendrerit ipsum dolor sed augue. Nulla nec lacus. Dann betrachte den Code in Listing 3.

Listing 3: Some code

```
# Program to find the sum of all numbers stored in a list (the not-Pythonic-way)

# List of numbers
numbers = [6, 5, 3, 8, 4, 2, 5, 4, 11]

# variable to store the sum
sum = 0

# iterate over the list
for val in numbers:
    sum = sum+val

print("The sum is", sum)
```

6.1 Plannung

6.1.1 fortlaufendes Prototyping

6.2 Design

6.2.1 Userexperience [L]

In keinem Softwareprojekt darf nicht auf die Benutzer vergessen werden. Feature und Design müssen immer mit dem Gedanken, wie Hilft das dem*der Kunden*in oder der Zielgruppe weiter, entwickelt werden. Das Schlagwort hier ist die Userexperience (oder auch Nutzungserlebnis).

In den folgenden Kapitel wird das Thema User Experience und wie dieses Thema im Projekt umgesetzt wurde.

UX: Userexperience

7 Zusammenfassung

Aufzählungen:

- Itemize Level 1
  - Itemize Level 2
    - Itemize Level 3 (vermeiden)

1. Enumerate Level 1
   a. Enumerate Level 2
      i. Enumerate Level 3 (vermeiden)

Desc Level 1

Desc Level 2 (vermeiden)

Desc Level 3 (vermeiden)

Literaturverzeichnis

Abbildungsverzeichnis

1 ThreeJs Logo 6
2 Die Struktur von ThreeJs 6
3 ThreeJs Logo 6
4 Der Unterschied zwischen AABB und OBB 10
5 Don Knuth – CS Allfather 13

Tabellenverzeichnis

1 Ein paar tabellarische Daten 12

Quellcodeverzeichnis

1 Angular Three - Komponentenbasiertes 3D Scenen in HTML 8
2 Angular Three - App Cube 8
3 Some code 13

Anhang