Diplomarbeit

Höhere Technische Bundeslehranstalt Leonding
Abteilung für Informatik

Leo-Avatar

Eingereicht von:	**Robin Mair, 5chif**
	Lukas Riedl, 5chif
Datum:	**4. April 2018**
Betreuer:	**Dipl.-Ing. Mag. Robert Stöttinger**
Projektpartner:	**Mag. Dr. Thomas Stütz**

Declaration of Academic Honesty

Hereby, I declare that I have composed the presented paper independently on my own and without any other resources than the ones indicated. All thoughts taken directly or indirectly from external sources are properly denoted as such.
This paper has neither been previously submitted to another authority nor has it been published yet.

Leonding, 4. April 2018

Robin Mair, Lukas Riedl

Eidesstattliche Erklärung

Hiermit erkläre ich an Eides statt, dass ich die vorgelegte Diplomarbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Gedanken, die aus fremden Quellen direkt oder indirekt übernommen wurden, sind als solche gekennzeichnet.
Die Arbeit wurde bisher in gleicher oder ähnlicher Weise keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Leonding, am 4. April 2018

Robin Mair, Lukas Riedl

Zusammenfassung

Im Zuge der Diplomarbeit von Lukas Riedl und Robin Mair wird ein Avatar erstellt, dessen Aufgabe darin besteht, verschiedenste Fragen, welche während des Schultages an der HTL Leonding auftreten, zu beantworten. Das Ziel des Avatars ist es, folgende Fragen abzudecken:

- Welches Fach unterrichtet Professor Musterprofessor?
- Wann sind die nächsten Ferien?
- In welche Klasse geht Mustermann Max?
- Wie heißt der Direktor der HTL Leonding?
- Wie heißt der Abteilungsvorstand der Informatik/Elektronik?
- In welcher Klasse unterrichtet Professor Musterprofessor in der XY Stunde?

Als Sprachorgan des Hologramms dient Amazon Echo, womit die Kommunikation und die Beantwortung der Fragen ermöglicht wird. Da Echo nicht über das benötigte Wissen verfügt, wird eine "Funktion" entwickelt, welche über die benötigten Informationen verfügt und diese je nach Frage korrekt wiedergibt. Um das Hologramm darzustellen, wird eine Pyramide aus Plexiglas verwendet, die mit einem Fernseher bestrahlt wird.

Abstract

As part of the diploma thesis of Lukas Riedl and Robin Mair, a hologram is created, whose task consists of answering various questions, which occur during the school day at the HTL Leonding. The goal of the avatar is to cover the following questions:

- Which subject does Professor Musterprofessor teach?
- When are the next holidays?
- In which class is Mustermann Max?
- What is the name of the principal of HTL Leonding?
- What is the name of the department head of computer science / electronics?
- In which class does Professor Musterprofessor teach in the XY unit?

Amazon Echo serves as the organ of the hologram, enabling communication and answering questions. Since Alexa does not have the knowledge she needs, a "function" is developed, which has the required information and reproduces it correctly depending on the question. To illustrate the hologram, a pyramid of Plexiglas is used, which is irradiated with a television.

Danksagungen

Es ist uns ein großes Anliegen, uns bei den Personen zu bedanken, die diese Diplomarbeit ermöglicht haben:

- Bei unserem Betreuer Dipl.-Ing. Mag. Robert Stöttinger, der uns stets motivierte weiterzumachen.
- Bei Mag. Dr. Thomas Stütz, der uns oft technische Unterstützung bereitstellte und uns bei allen Fragen zur Seite stand.
- Bei unseren Mitschülern, die uns des Öfteren mit technischem Beistand halfen.
- Bei der HTL Leonding, für die Bereitstellung der benötigten technischen Mittel.

Inhaltsverzeichnis

Kapitel 1

Pflichtenheft

1.1 Ausgangslage

Die HTL Leonding, eine moderne Schule mit Schwerpunkt Informatik, bietet ein breites Ausbildungsspektrum im Bereich der Informatik und Elektronik an. Im Rahmen der Ausbildung werden neueste Technologien vermittelt. Die HTL Leonding möchte Besuchern ihr Ausbildungsspektrums anschaulich präsentieren.

1.2 Zielsetzung

Der Avatar dient Besuchern zum Einholen von Informationen bezüglich Veranstaltungen, Rauminformationen usw.
Die HTL Leonding kann mit dem Avatar die Breite ihres Ausbildungsprogrammes präsentieren.

1.3 Funktionale Anforderungen

Der Benutzer geht zu einer aufgebauten Pyramide in der ein Avatar steht. Den Avatar kann man von drei verschiedenen Perspektiven betrachten, dadurch wirkt er wie ein Hologramm. Der Benutzer startet das Hologramm durch das Schlagwort “erwache“ und so beginnt die Interaktion mit dem Avatar. Nun kann man Fragen zu Lehrern, Stundenplänen usw. stellen. Der Avatar antwortet mittels Amazon Echo und bewegt sich entsprechend dazu.

1.4 UseCase Diagramm

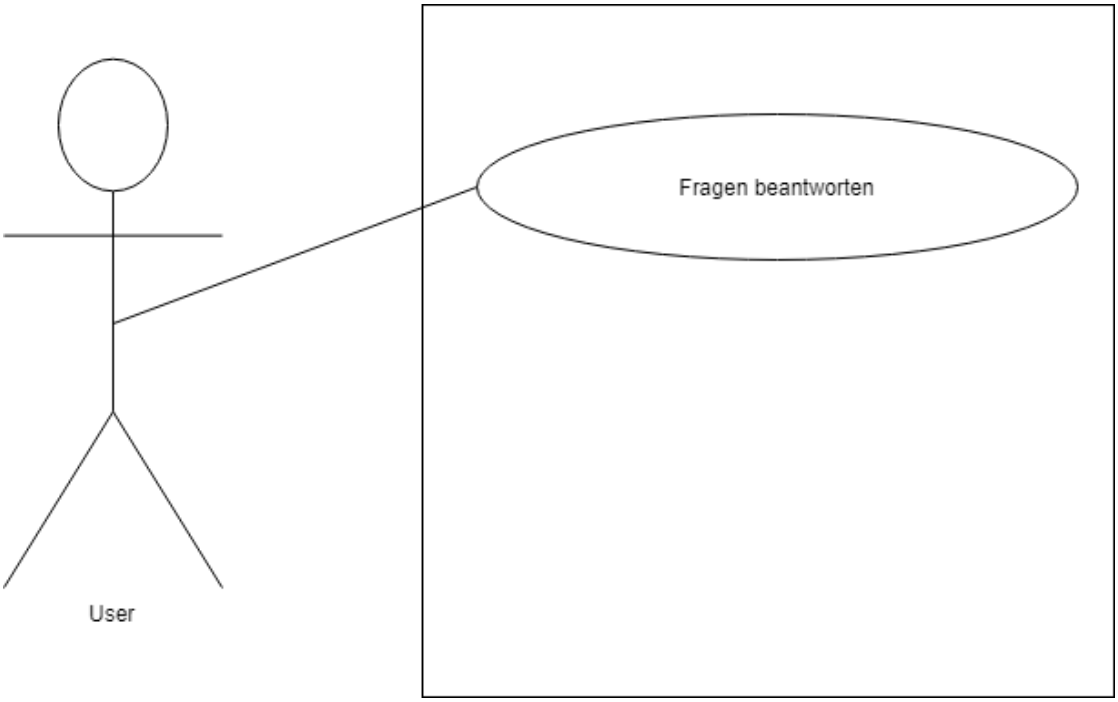


Abbildung 1.1: UseCase Diagramm

1.5 Systemarchitektur

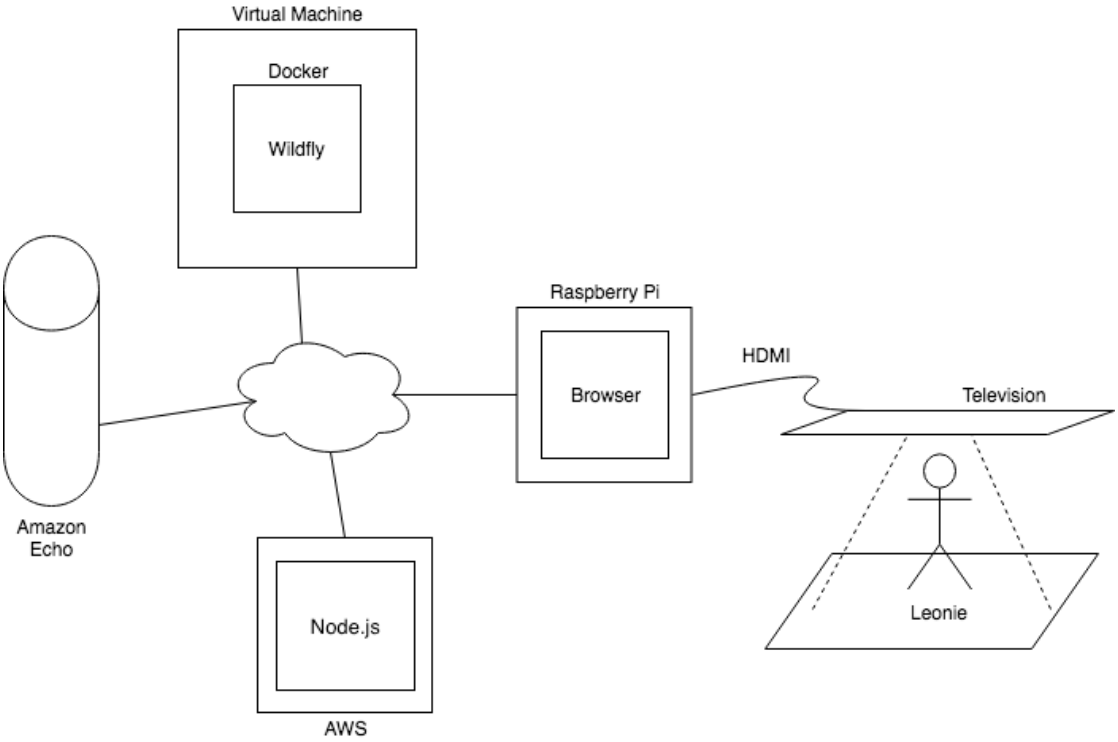


Abbildung 1.2: Systemarchitektur

1.6 Deployment Diagramm

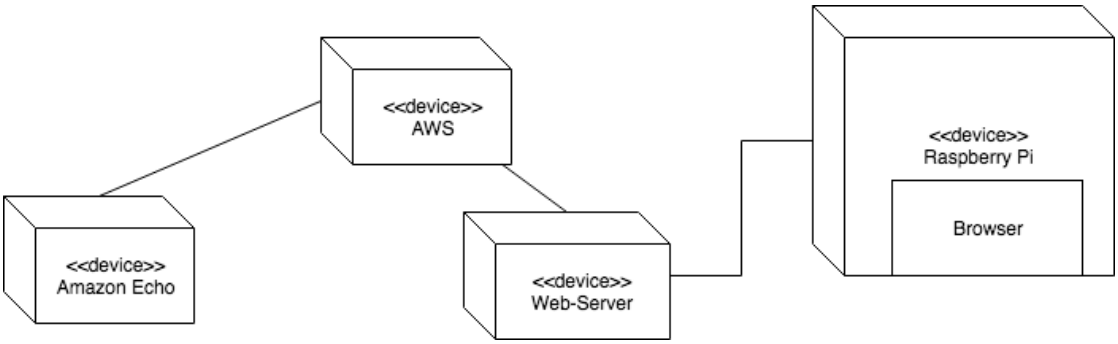


Abbildung 1.3: Deployment Diagramm

Kapitel 2

Erstellung des Avatars

Um den Avatar erfolgreich als Hologramm darzustellen, reicht eine einfache 2D Zeichnung nicht aus, weswegen ein 3D Modell notwendig war. Es gibt im Internet viele fertige 3D Modelle, jedoch ist die Kreativität dadurch stark eingeschränkt, da es zum Beispiel nicht einfach so möglich ist, die Kleidung zu ändern. Wegen diesen Gründen wurde ein eigener Avatar designt.

2.1 MakeHuman

Um diese Aufgabe zu erledigen, wurde das Programm MakeHuman verwendet. Dieses Programm zeichnet sich durch seine einfache Bedienbarkeit aus, wodurch es möglich ist, ohne jegliche Vorkenntisse in diesem Fachgebiet, eine 3D Figur zu erstellen. Das war ein ausschlaggebender Grund warum die Wahl auf MakeHuman fiel. Ein weiterer Vorteil an MakeHuman ist die Community, welche über ein informationsreiches und aktives Forum verfügt. Tritt etwa ein Fehler oder eine Frage zum Programm auf, gibt es bereits viele gelöste Probleme. MakeHuman baut stark auf die Kreativität seiner Benutzer auf, wodurch es bereits etliche Erweiterungen wie etwa neue Knochenmodelle oder Kleidungsstücke frei zum Herunterladen gibt, welche den bereits umfangreichen Baukasten des Basisprogramms erweitern.

2.1.1 Modellierung

Bei MakeHuman wird mit einer blanken Figur begonnen. Der Menüpunkt “Modelling“ beinhaltet alle Einstellungsmöglichkeiten für das Modell, welche auf verschiedenste Unterpunkte aufgeteilt werden. Dadurch gestaltet sich das Bearbeiten des Modells trotz der Vielfältigkeit als übersichtlich.

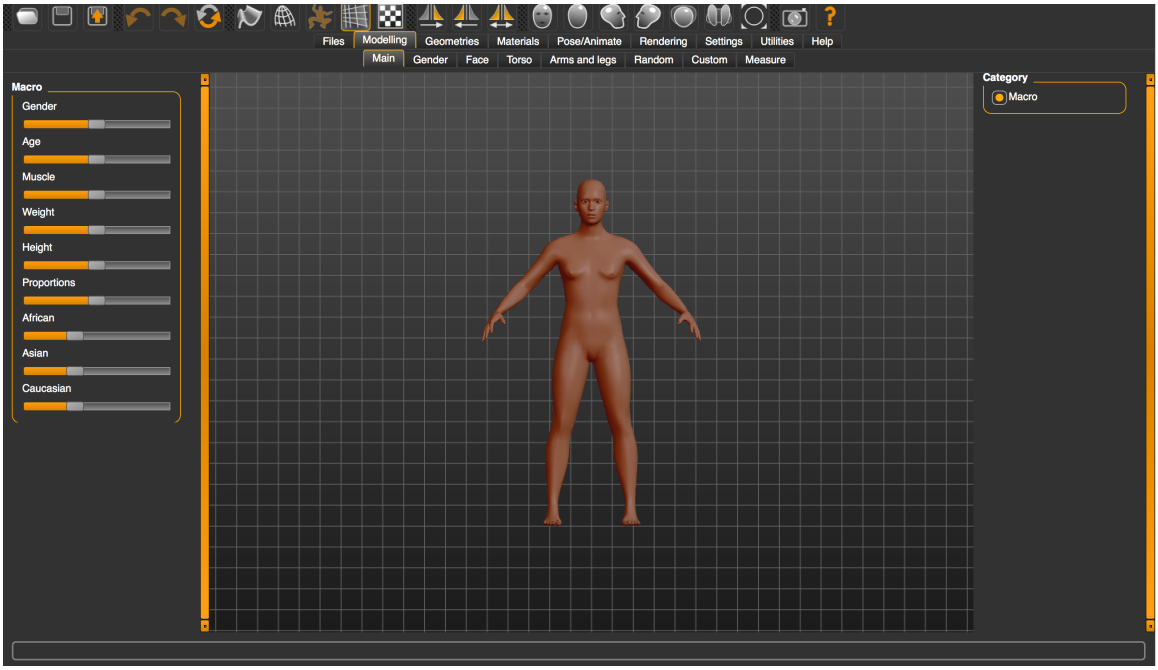


Abbildung 2.1: Die blanke Figur

Zu Beginn werden unter dem Punkt “Main“ die groben Eigenschaften - wie etwa das Geschlecht, Alter, Größe, Gewicht, etc - festgelegt. Das Ändern einer Eigenschaft erfolgt über die Schieberegler und ist selbsterklärend. Wird zum Beispiel der Schieberegler für das Geschlecht ganz nach links geschoben, so bekommt das Modell eine weibliche Grundform, wird der Regler nun ganz nach rechts verschoben bekommt das Modell eine eher männlich ausgerichtete Basis.

Nachdem die grobe Form der Figur festgelegt wurde, können in den folgenden Menüpunkten die Details entschieden werden, wobei kaum eine Körperstelle nicht abänderbar ist. Dabei zieht sich das System der Schieberegler durch alle Menüs.

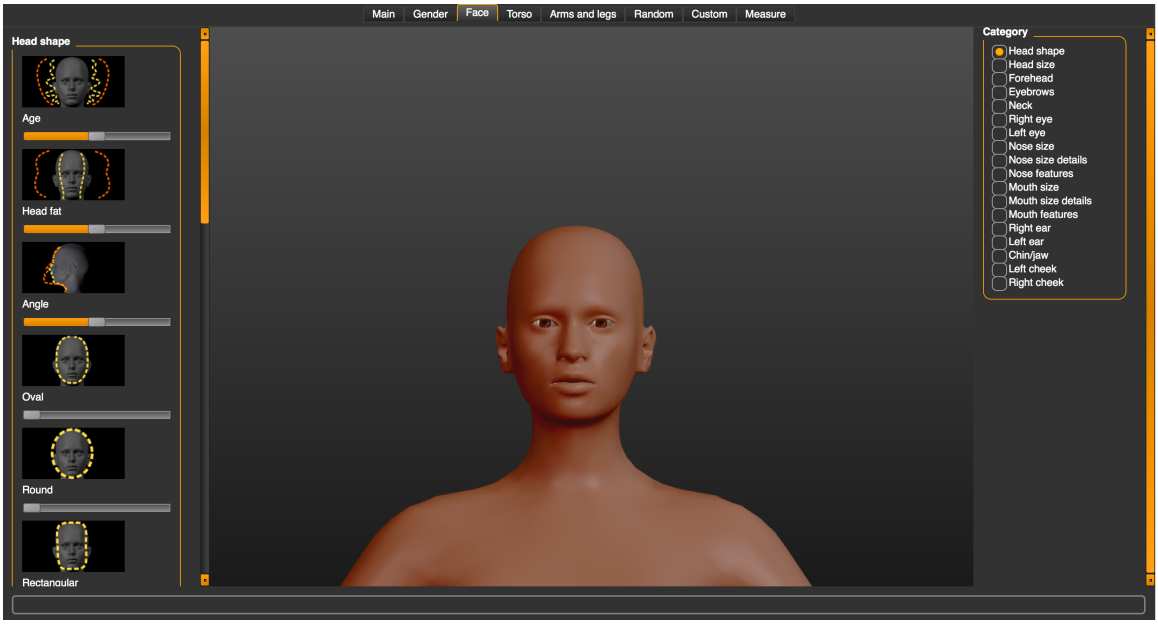


Abbildung 2.2: Gesichtsoptionen

In der linken Box befinden sich Schieberegler für die Einstellungsmöglichkeiten. Dessen Auswirkungen auf das Modell werden durch Bilder und Texte erläutert.
In der rechten Box wird entschieden, welche Region des Körpers angesprochen wird, wodurch sich die Schieberegler an die ausgewählte Region anpassen. Dies ist äußerst nützlich, da dadurch die extreme Menge an Einstellungen strukturierter wird, wodurch ein schnelles Arbeiten und unkompliziertes Ändern im Nachhinein ermöglicht wird. Sobald es ins Detail geht, wird der Nachteil an solch einem System deutlich. Denn dadurch wird es schnell sehr aufwändig wenn ein symmetrischer Körper modelliert werden soll, da die linke und die rechte Seite einzeln eingestellt werden müssen. Wenn eine 100-prozentige Symmetrie erreicht werden soll, macht sich dieses Problem bereits bei nur einer veränderten Einstellung bemerkbar, da es sehr unpräzise ist, wenn sich die Einstellung nur durch einen Schieberegler definieren lässt und nicht etwa auch durch eine Zahl.

2.1.2 Knochen

Damit die Figur später animiert werden kann, benötigt das Modell ein Skelett. Je nach dem wie die Figur anschließend animiert wird, werden unterschiedlichste Knochenmodelle benötigt. Diese werden von MakeHuman zur Verfügung gestellt. Soll zum Beispiel auch das Gesicht im Detail animierbar sein, wie es beim HTL-Leonding-Avatar der Fall ist, so ist das “Default“ Knochenmodell zu empfehlen.

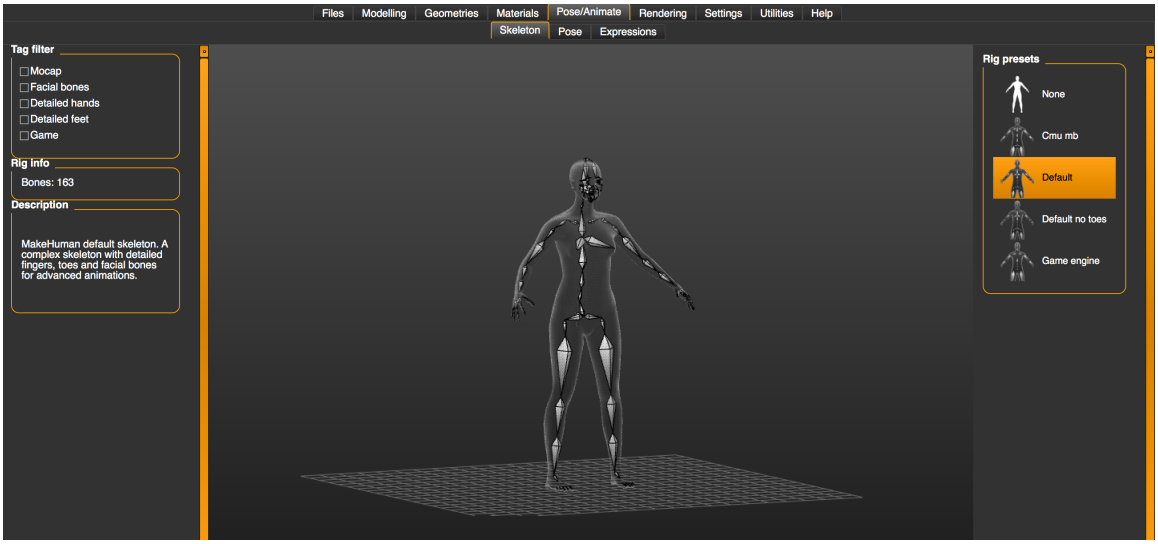
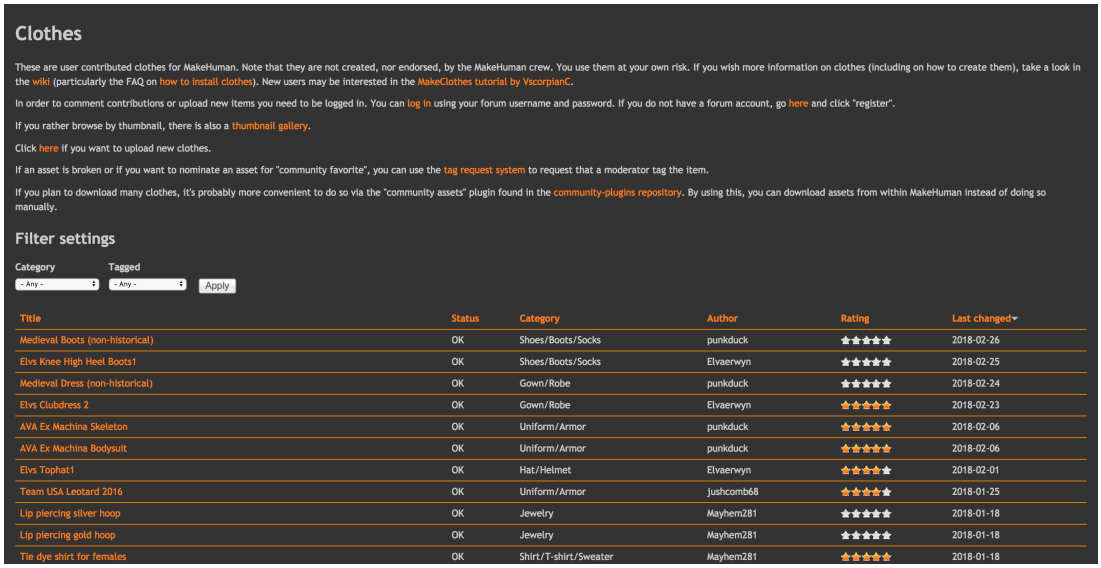


Abbildung 2.3: Darstellung der Knochenstruktur

2.1.3 Design

Nach dem Modellieren kann der Avatar nun je nach persönlichem Geschmack designt werden. Dazu stellt MakeHuman bereits ein paar "Basics", die jedoch den Anforderungen des HTL-Leonding-Avatars nicht entsprachen. In so einem Fall profitiert man von der Community welche hinter MakeHuman steckt. Auf deren Website gibt es Kleidungsstücke, Frisuren und vieles mehr zum Herunterladen.



Clothes

These are user contributed clothes for MakeHuman. Note that they are not created, nor endorsed, by the MakeHuman crew. You use them at your own risk. If you wish more information on clothes (including on how to create them), take a look in the wiki (particularly the FAQ on how to install clothes). New users may be interested in the MakeClothes tutorial by VscorpianC.

In order to comment contributions or upload new items you need to be logged in. You can log in using your forum username and password. If you do not have a forum account, go here and click "register".

If you rather browse by thumbnail, there is also a thumbnail gallery.

Click here if you want to upload new clothes.

If an asset is broken or if you want to nominate an asset for "community favorite", you can use the tag request system to request that a moderator tag the item.

If you plan to download many clothes, it's probably more convenient to do so via the "community assets" plugin found in the community-plugins repository. By using this, you can download assets from within MakeHuman instead of doing so manually.

Filter settings

Category - Any - Tagged - Any - Apply

Title	Status	Category	Author	Rating	Last changed
Medieval Boots (non-historical)	OK	Shoes/Boots/Socks	punkduck		2018-02-26
Elvs Knee High Heel Boots1	OK	Shoes/Boots/Socks	Elvaerwyn		2018-02-25
Medieval Dress (non-historical)	OK	Gown/Robe	punkduck		2018-02-24
Elvs Clubdress 2	OK	Gown/Robe	Elvaerwyn		2018-02-23
AVA Ex Machina Skeleton	OK	Uniform/Armor	punkduck		2018-02-06
AVA Ex Machina Bodysuit	OK	Uniform/Armor	punkduck		2018-02-06
Elvs Tophat1	OK	Hat/Helmet	Elvaerwyn		2018-02-01
Team USA Leotard 2016	OK	Uniform/Armor	jushcomb68		2018-01-25
Lip piercing silver hoop	OK	Jewelry	Mayhem281		2018-01-18
Lip piercing gold hoop	OK	Jewelry	Mayhem281		2018-01-18
Tie dye shirt for females	OK	Shirt/T-shirt/Sweater	Mayhem281		2018-01-18

Abbildung 2.4: Communityseite für weitere Kleidung

Das Installieren dieser Erweiterungen ist sehr simpel. Man muss lediglich zu dem “Data“ Ordner von MakeHuman navigieren, welcher sich bei einem Mac unter “/Documents/MakeHuman/v1/“ befindet. Im “Data“ Verzeichnis sind nun die Ordner wo die Communityerweiterungen reinkopiert werden.

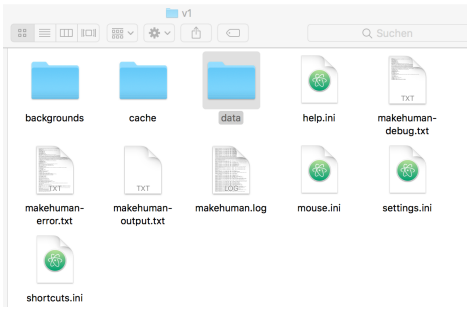


Abbildung 2.5: Data Ordner

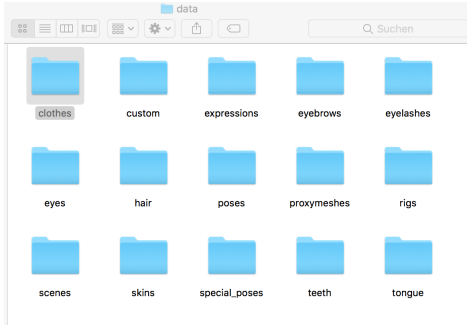


Abbildung 2.6: Ordner für die Erweiterungen

Nachdem man, wie hier zum Beispiel im “Clothes“ Ordner, einen neuen Ordner für die Erweiterung angelegt hat, kopiert man nun die Dateien, welche man sich auf der Community Homepage heruntergeladen hat, hinein.

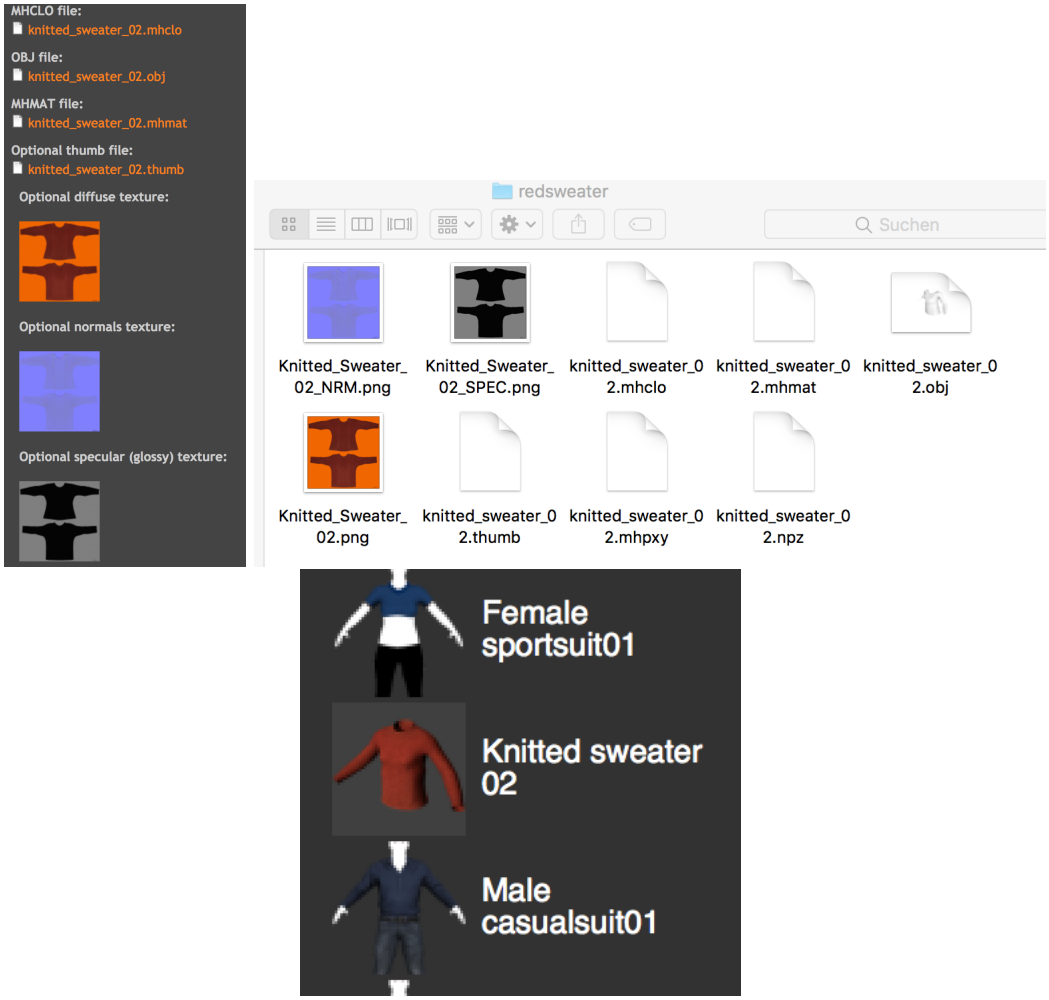


Abbildung 2.7: Beispiel mit einem roten Sweater

2.1.4 Exportieren

Ist das Ergebnis des Modells zufriedenstellend, muss die 3D-Figur noch exportiert werden.
Hier gibt es nun wieder etliche Einstellungen vorzunehmen, welche sich dann auch darauf auswirken, wie man anschließend mit dem Modell in Blender arbeiten muss. Wir empfehlen als Mesh-Format MakeHuman Exchange (mhx2) zu verwenden, da man bei diesem Format viele Vorteile und Funktionen in Blender genießen kann, welche man bei den anderen Mesh-Formaten nicht hat. Um allerdings das Modell als .mhx2 Datei exportieren zu können, muss das dazu benötigte MakeHuman Plugin installiert sein. Hierzu wird wieder die Community Seite von MakeHuman benötigt. Befindet man sich auf der Seite findet man im “Downloads“ Menü einen Link für eine Zip Datei.

Abbildung 2.8: Export Fenster

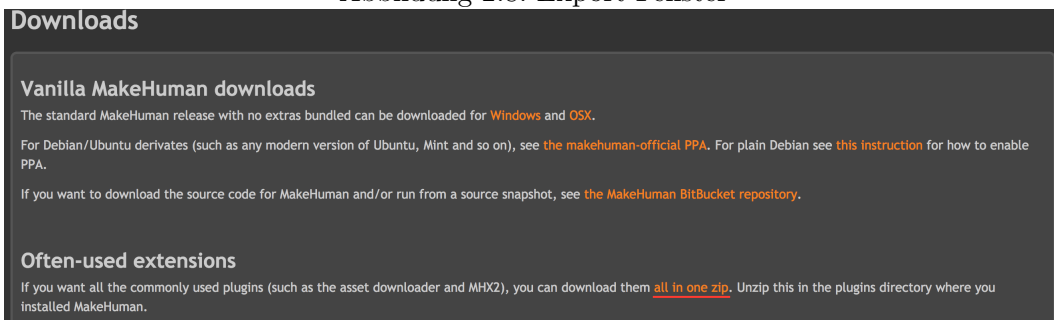


Abbildung 2.9: Downloadseite

In dieser Datei findet man einen Ordner mit dem Namen 9_export_mhx2. Nun geht man in den Programmordner von MakeHuman und sucht sich den “Plugins“ Ordner. In den “Plugins“ Ordner fügt man nun den 9_export_mhx2 Ordner ein. Jetzt MakeHuman einfach neu starten, davor allerdings kontrollieren ob man die 3D-Figur gespeichert hat. Nun wieder zum Exportmenü gehen und als Mesh-Format MakeHuman Exchange auswählen. Hier macht sich bereits der erste Vorteil des MHX2-Formates erkennbar, welcher es ermöglicht, beim Exportieren die “Expressions“, welche bereits vorgefertigte Gesichtsausdrücke mitliefern, und “Poses“, welche vorgefertigte Körperhaltungen beinhalten, mit exportiert werden können. Diese können später in Blender sehr hilfreich für das Animieren sein.

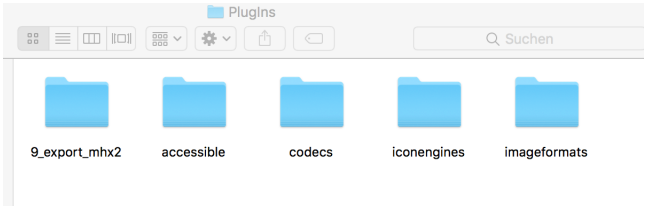


Abbildung 2.10: Plugins Ordner mit der heruntergeladenen Erweiterung

Kapitel 3

Animieren des Avatars

Nachdem die 3D-Figur erstellt wurde, muss diese nun animiert werden.

3.1 Blender

Wegen folgenden Gründen wurde das Programm Blender verwendet:

- kostenlos
- viele Tutorials
- schnell erlernbar
- vollständige Kompatibilität mit dem MakeHuman Modell

3.1.1 MakeHuman Plugin

Um das Importieren der MHX2 Datei zu ermöglichen, muss das dafür benötigte Plugin installiert werden. Dazu lädt man es von der Downloadseite der Community-Homepage herrunter. In diesem Ordner befindet sich allerdings nicht nur das Plugin zum Importieren, sondern auch noch MakeClothes, MakeTarget und MakeWalk.

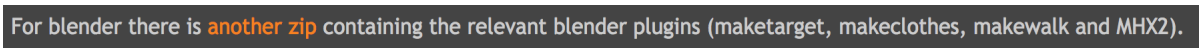


Abbildung 3.1: Download für das Plugin

Als nächstes geht man zum “Addons“ Ordner welcher im Blender Programmordner zu finden ist. In diesen Ordner fügt man nun die Ordner der Plugins ein.

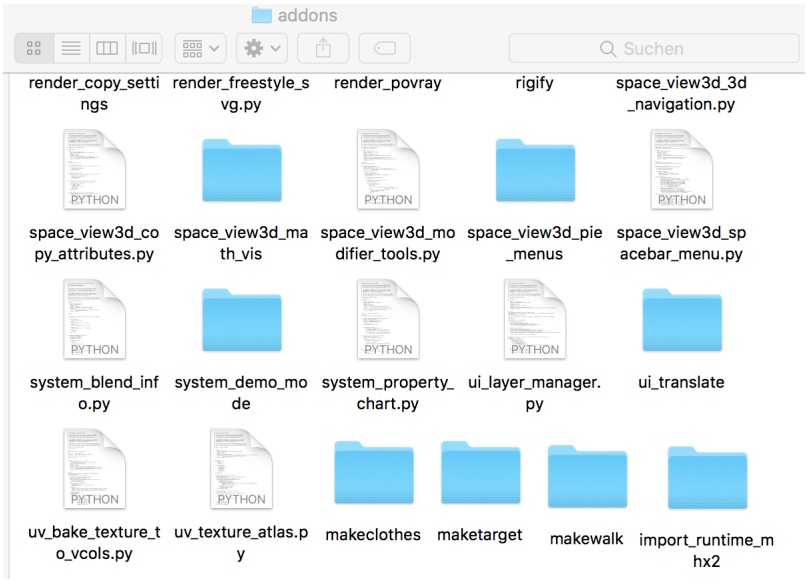


Abbildung 3.2: Plugins im Addons Ordner

Allerdings sind die Plugins noch nicht aktiviert. Hierzu geht man in Blender zu File ->User Preferences ->Add-ons. In diesem Fenster sucht man sich nun die Plugins die man aktivieren möchte, setzt bei ihnen das Häkchen und speichert die neuen Einstellungen.

Abbildung 3.3: Häkchen bei den Addons setzten

3.1.2 Importieren des Avatars

Hat man nun das Import-Plugin installiert muss man zum erfolgreichen Importieren noch eine kleine Änderung in den User Preferences vornehmen. Dazu öffnet man wieder das Preferences Fenster, klickt auf das “File“ Menü und setzt das Häkchen bei “Auto Run Python Scripts“.

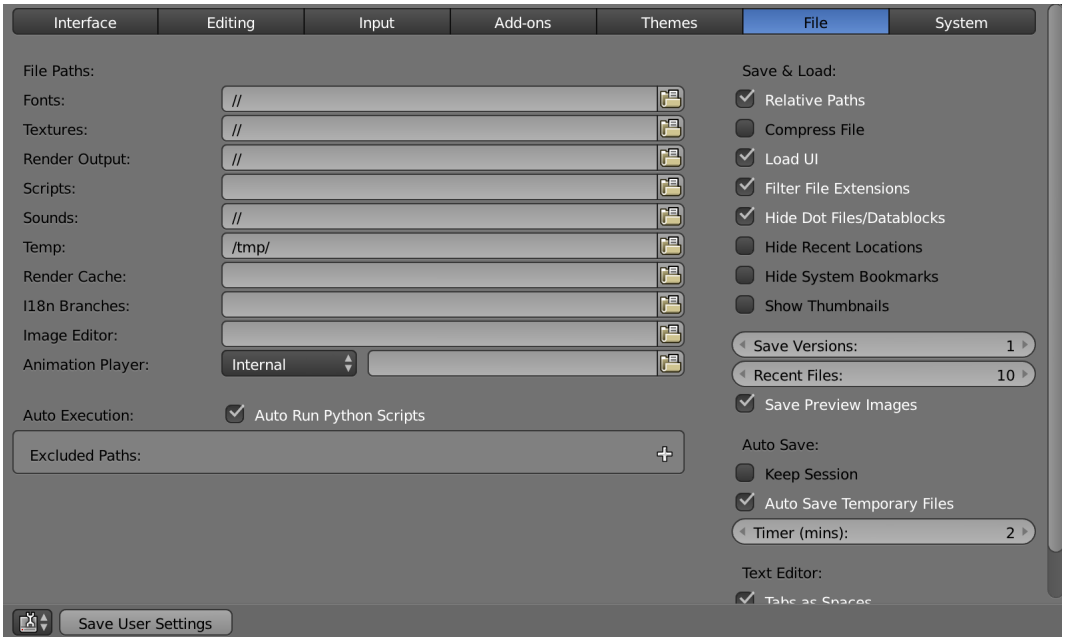


Abbildung 3.4: “Auto Run Python Scripts“ aktivieren

Nachdem alle benötigten Einstellungen getroffen wurden, kann nun auf der linken Seite unter dem “MHX2 Runtime“ Tab über den “Import MHX2“ Button die Figur importiert werden, die zuvor in MakeHuman als MHX2 Datei exportiert wurde.

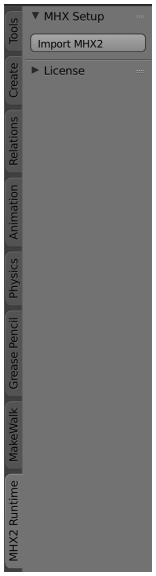


Abbildung 3.5: Import Button

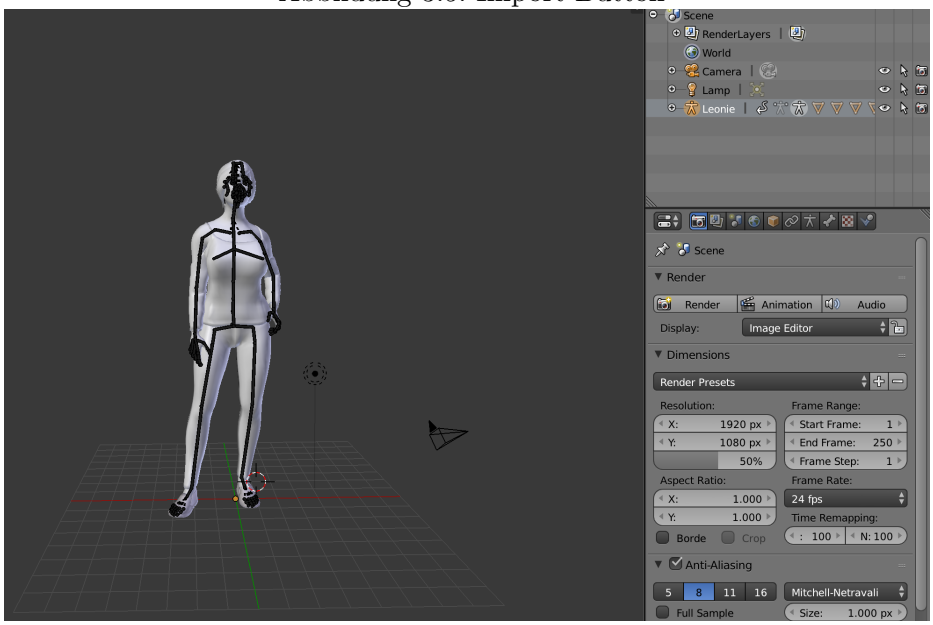


Abbildung 3.6: Die 3D Figur in Blender

3.1.3 Animation

Um eine Animation aufzunehmen muss zuerst der Record-Button aktiviert werden. Dadurch wird bei jeder Positionsveränderung eines Knochen automatisch ein Keyframe gesetzt, wodurch sich der Vorgang um einiges komfortabler gestaltet als wenn jeder Keyframe manuell gesetzt werden muss.

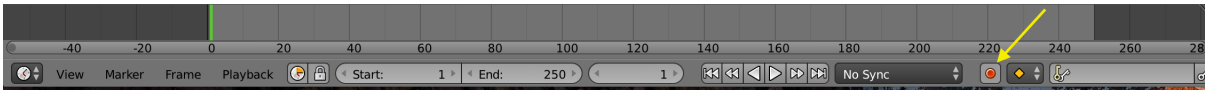


Abbildung 3.7: Record Button

Allerdings **muss** der erste Keyframe manuell gesetzt werden, da für die Ausgangspostition noch keiner vorhanden ist. Dazu muss in dem Fenster rechts neben dem Record-Button “Whole Character“ ausgewählt und anschließend rechts auf das Schlüsselsymbol geklickt werden.

Abbildung 3.8: Keyframe setzen

Jetzt geht man ein paar Frames in der Timeline weiter. Je länger der Abstand von 2 Keyframes, desto langsamer wird die Animation von der letzten Position zur nächsten Position. Dann wählt man über einen Rechtsklick einen beliebigen Knochen aus, den man bewegen möchte. Hat man sich für einen Knochen entschieden, drückt man auf der Tastatur “R“ um den Rotationsmodus zu aktivieren. Wenn man sich in diesem Modus befindet kann man durch das Bewegen der Maus den Knochen in die verschiedensten Richtungen drehen. Hierbei sollte man allerdings beachten, dass man den Knochen nicht so dreht, dass sich die Kleidung stark verdreht, da es sonst sehr unschön aussehen kann. Um den Knochen kontrollierter zu drehen kann man im Rotationsmodus zusätzlich über die X/Y/Z Taste die Achse fixieren um die sich der Knochen rotieren soll. Ist man mit der neuen Knochenausrichtung zufrieden bestägt man dessen Position mit einem linken Mausklick. Nun kann man sehen, dass in der Timeline automatisch ein neuer Keyframe gesetzt wurde. Über den Playbutton in der Leiste unter der Timeline kann man sich nun seine selbst erstellte Animation ansehen und überprüfen. Wenn man will, dass sich bis zu einem Keyframe mehrere Knochen verändern, wiederholt man dieses Vorgehen mit den anderen Knochen ohne die Position auf der Timeline zu ändern.

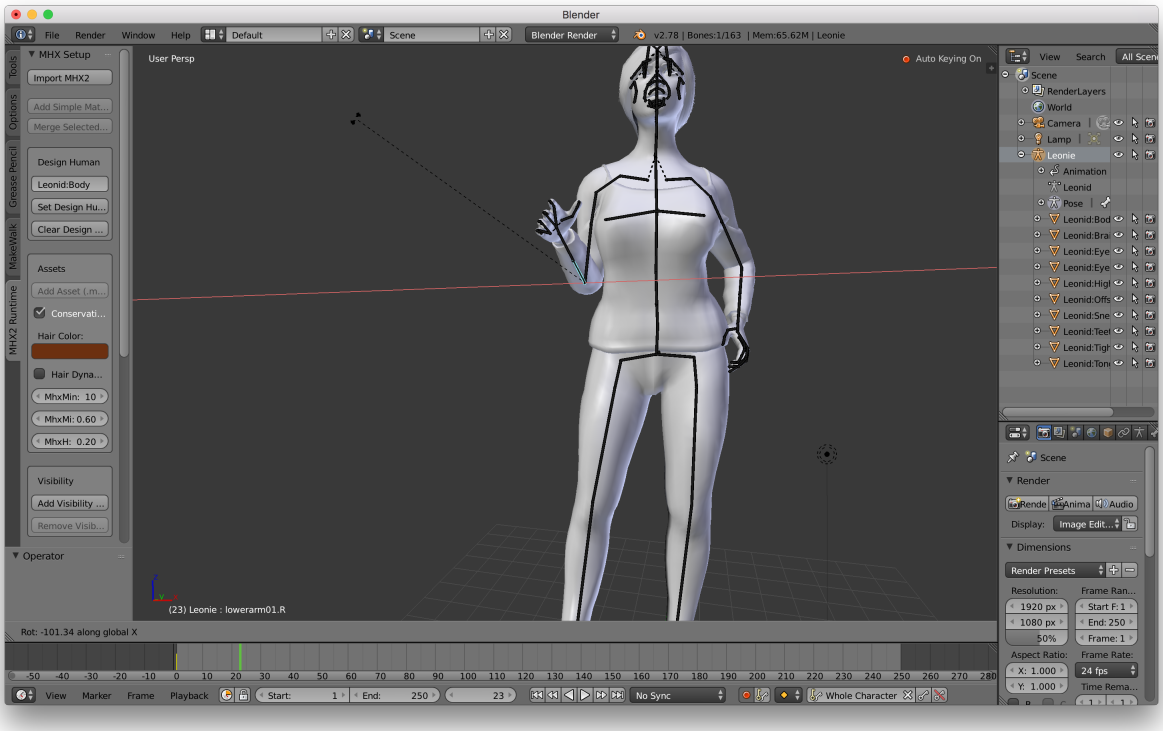


Abbildung 3.9: Rotationsmodus mit Fixierung durch die X Taste

Abbildung 3.10: Keyframes durch die gelben Striche ersichtlich

3.1.4 MHX2 Runtime

Der “MHX2 Runtime“ Tab beinhaltet hilfreiche Presets, welche das Animieren allgemein erleichtern. Darin befinden sich die “Poses“ und “Expressions“ welche beim exportieren des 3D Modells in MakeHuman mitgespeichert wurden.

Aufsteh Animation

Soll der Avatar aufstehen, so müssen viele einzelne Knochen rotiert und neu ausgerichtet werden. Dank der MHX2 Runtime geht das deutlich leichter, denn durch die mitgelieferten “Poses“ ergibt sich eine große Vorauswahl an Ausgangskörperhaltungen. Eine davon nennt sich “sit01“ bei der der Avatar eine Sitzpostion einnimmt. Diese Ausgangsposition kann nun sofort als Startkeyframe der Animation verwendet werden oder als Basis,

welche je nach Wunsch noch angepasst werden kann. Dadurch kann viel Zeit gespart werden, den Avatar so auszurichten, dass es einigermaßen nach einer gemütlichen Sitzhaltung aussieht.

Abbildung 3.11: Die mitgelieferten Poses

Avatar sprechen lassen

Das Animieren des Gesichts und, in diesem Fall, der Lippen funktioniert etwas anders als bei den restlichen Körperstellen. Die Gesichtszüge lassen sich nicht durch Rotation anders platzieren, sondern durch Skalierung, welche man nach dem Selektieren des Knochens durch die “S“ Taste aktiviert. Verkleinert man nun den ausgewählten Knochen, zum Beispiel den der Unterlippe, so öffnet sich der Mund.

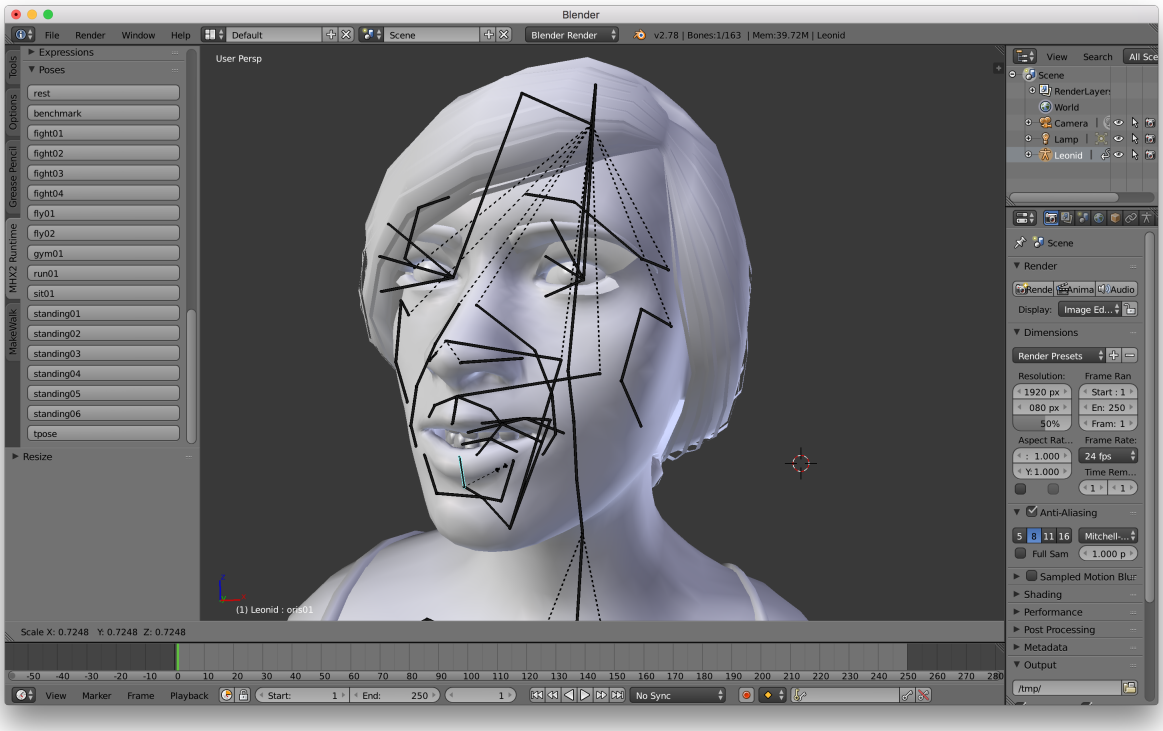


Abbildung 3.12: Skaliermodus

Allerdings ist diese Art der Lippenbewegen sehr mühsam, vorallem wenn es eine etwas längere Sprechanimation werden soll. Dafür bietet die MHX2 Runtime auch ein nützliches Tool, welches man unter “Face Units“ im Tab der Runtime findet.

Abbildung 3.13: Face Units Einstellmöglichkeiten

Darin findet man etliche Parameter für die Gesichtszüge, welche das Bewegen dieser um ein vielfaches erleichtert. Für die selbe Lippenbewegung wie vorhin beschrieben zu erzielen, muss lediglich der Parameter “lowerLipDown“ verändert werden und schon bewegt sich die Unterlippe. Dadurch lässt sich nun viel einfacher und schneller eine natürliche Lippenbewegung darstellen.

3.1.5 Darstellung als Hologramm

Da jedoch mit nur einer 3D-Figur kein Hologramm entsteht, musste eine Lösung erarbeitet werden, wie das Darstellen von drei Figuren zur selben Zeit am besten realisierbar ist. Anstatt die eine Figur einzeln zu rendern und anschließend in einem Videoprogramm die benötigte Formation zu erstellen, wurde sich dafür entschieden, in Blender das Modell drei mal zu importieren. Nach dem Importieren von zwei weiteren Kopien der Figur müssen die Objekte richtig ausgerichtet werden. Hier gab es zwei Optionen:

- alle drei Figuren schauen nach vorne, wobei bei dieser Variante das fertige Hologramm dann sehr plastisch wirkt, da keine Tiefe entsteht
- man rotiert die drei Figuren jeweils um 90 Grad, um beim Hologramm den Effekt zu haben, dass sich das Modell im Raum befindet. Diese Variante wurde anschließend umgesetzt.

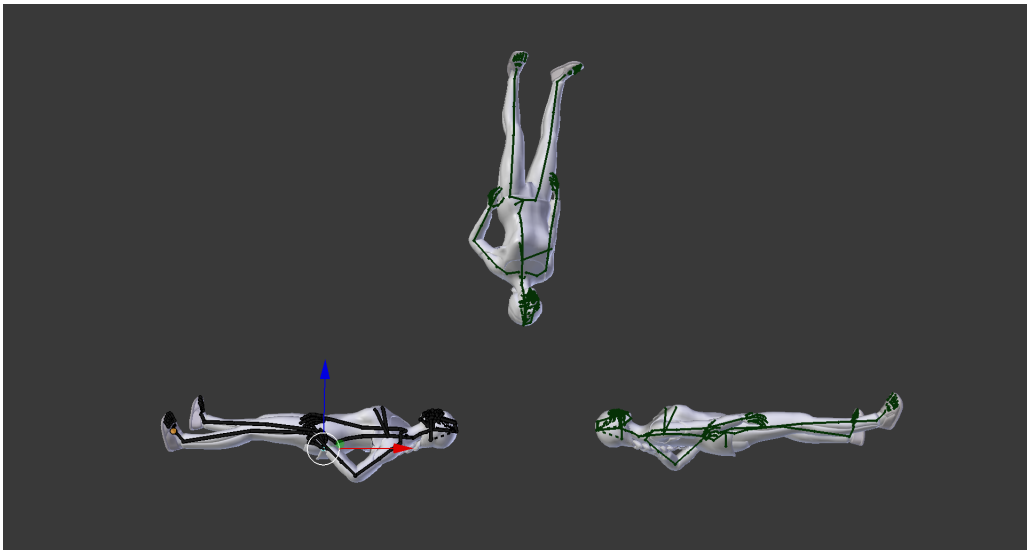


Abbildung 3.14: Formation für den Hologrammeffekt mit Rotation der Figuren

Da die Keyframes für die Animation allerdings nur an einer Figur erstellt werden, muss dieselbe Animation auch auf die zwei weiteren Figuren übertragen werden. Jedoch müssen nicht alle Knochenpositionen mühevoll einzeln auf die anderen Figuren übertragen werden, sondern es kann die ganze Animation kopiert und bei den anderen Modellen eingefügt werden. Dazu wird als erstes von der Modellansicht auf die Dope Sheet Ansicht gewechselt und vom Dope Sheet zum Action Editor gewechselt.

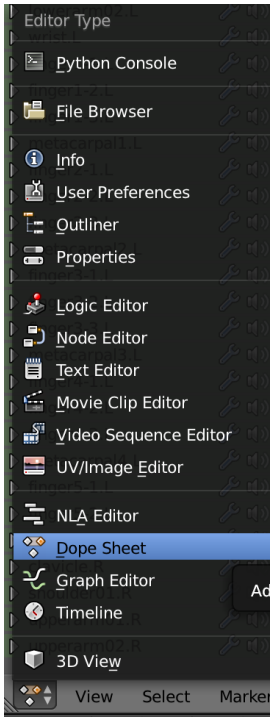


Abbildung 3.15: Dope Sheet

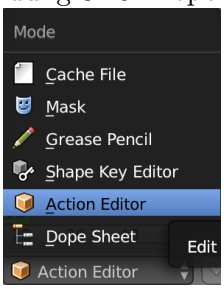


Abbildung 3.16: Auswahl des Action Editors

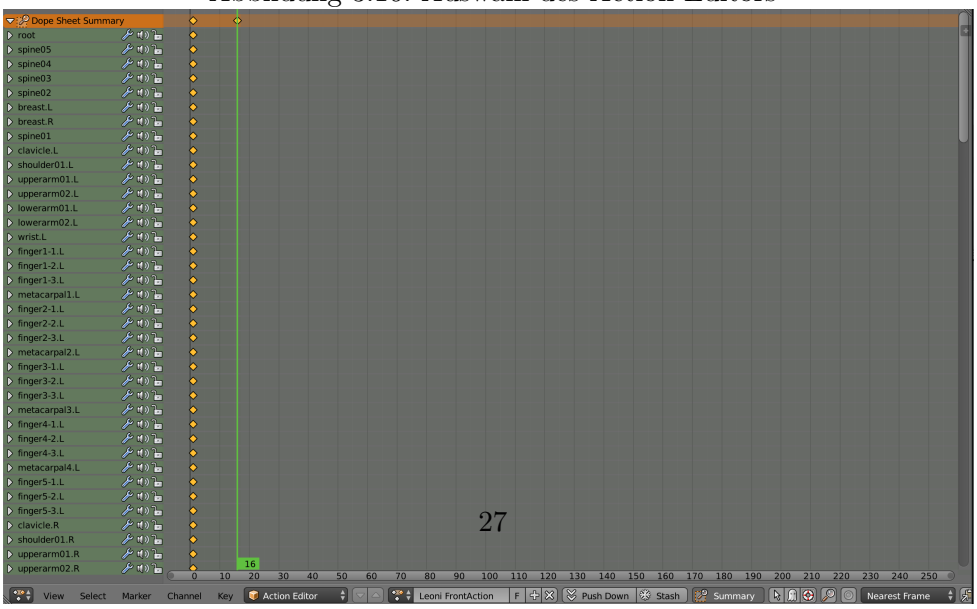


Abbildung 3.17: Action Editor

Dann wird in der Objektübersicht eine der Figuren, die noch keine Animation haben, ausgewählt. Dieser Figur wird nun die Animation der ersten Figur zugewiesen. Dieses Verfahren wiederholt sich bei der dritten Figur. Da nun alle Figuren auf die selbe Animation zugreifen und nicht drei Kopien der Animation erstellt wurden, muss dieser Vorgang bei einer Änderung der Animation nicht wiederholt werden.

Abbildung 3.18: Objektübersicht

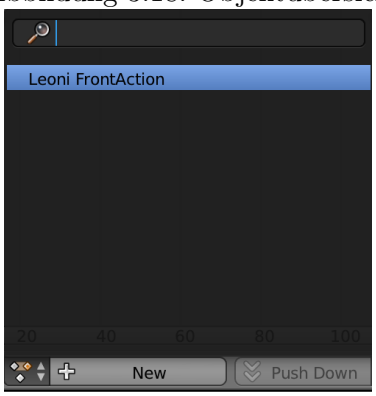


Abbildung 3.19: Animation festlegen

3.1.6 Rendering

Nach dem die Animation den Wünschen entspricht, müssen noch die richtigen Einstellungen für das Rendern der Animation getroffen werden. Neben den Standardeinstellungen für das mp4 Format kann viel mit der Auflösung, der Framerate, etc. gespielt werden. Als Auflösung wurde sich für Full HD (1920x1080) entschieden, da mit dieser Auflösung die Videos am besten auf dem Raspberry Pi laufen. Das Ändern der Framerate hat zwei Auswirkungen:

- wie flüssig die Animation dem Betrachter vorkommt

- wie lang die Animation ist. So kann etwa, wenn die Animation etwas zu kurz geraten ist, mit einer etwas niedrigeren Framerate die Animation länger gemacht werden, jedoch hat dies Einbußen zur Folge, wenn es um die Flüssigkeit der Animation geht.

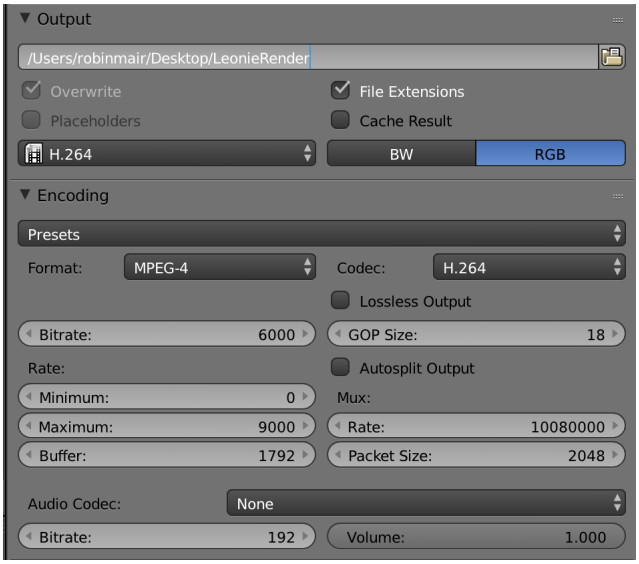


Abbildung 3.20: Einstellungen für das mp4 Format

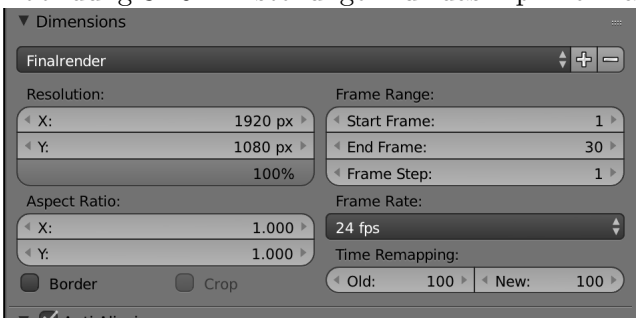


Abbildung 3.21: Einstellungen für die Auflösung und Framerate

Kapitel 4

Implementierung eines Alexa Skills

4.1 Erstellung eines Skills

Ein Skill ist eine Erweiterung für Amazon Echo. Diesen kann man sich im Store downloaden oder selbst erstellen. Für die Erstellung eines Solchen, wird ein Amazon Developer Account benötigt. Dieser ermöglicht den Zugriff auf die Developer Console, in der es einen eigenen Bereich für die Entwicklung eines Alexa Skills gibt. In diesem Bereich kann der Entwickler zwischen dem Alexa Skills Kit(ASK) und dem Alexa Voice Service(AVS) wählen. Mit dem ASK wird das Erstellen eines Skills ermöglicht. Mit dem AVS können Alexas Fähigkeiten an ein anderes Gerät, welches ein Mikrofon und einen Lautsprecher besitzt, übertragen werden. Bei der einfachen Erstellung eines Skills wird das Alexa Skills Kit verwendet. [12]

- **1.Allgemeine Informationen zum Skill angeben**
  Als "Skill Type" wird "Custom Interaction Model" gewählt, welches die Standardkonfiguration von Alexa als Client ist. Das "Smart Home Skill API" ist für die Entwicklung von IOT Sprachinterfaces zuständig. Das "Flash Briefing Skill API" wird dazu verwendet um zum Beispiel RSS-Inhalte in Form von News bereitzustellen.

  Die Sprache wird auf Deutsch eingestellt. Das Feld "Name" beschreibt, unter welcher Bezeichnung der Skill im Skill Store zu finden ist. Mit dem "Invocation Name" wird der Name festgelegt, mit dem der Benutzer den Skill startet. Wenn "erwache" angegeben ist, wird der Skill mit "Alexa erwache" oder mit "Alexa starte erwache" aktiviert. Die Einstellung "Audio Player", legt fest ob sogenannte Intents eine Audio-Player-Steuerung berücksichtigen sollen. Also ob in der Funktion eine Audio Datei abgespielt werden soll. Die Einstellung "Video App" wird verwendet, wenn zum Beispiel Videos am Echo Show angezeigt werden sollen."Render Template" dient für das Anzeigen von Inhalten auf dem Echo Show oder Echo Spot. Da weder eine Audio-Datei abgespielt, noch etwas angezeigt wird, werden diese auf "No" gesetzt. [12]

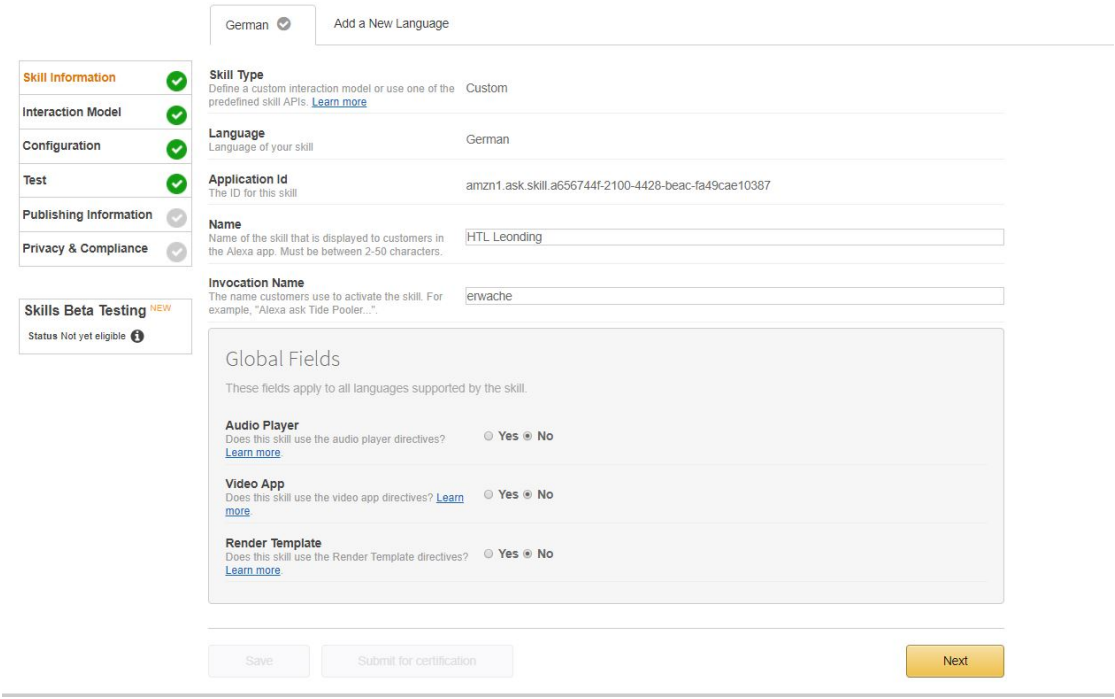


Abbildung 4.1: Amazon Developer Console

- **2.Das Interaction Model**
  Hier wird das Schema des Alexa Skills festgelegt.Also welche Satztypen beziehungsweise Intentionen es im Skill gibt. Zusätzlich wird noch eingestellt, welche variablen Teile, sogenannte Slots, diese haben können. Die Unterscheidung bei der Kommunikation mit Amazon Echo erfolgt zwischen “One-Shot-Models“ und “Dialog-Models“. Bei Ersterem, wird die Funktion mit dem Start des Skills ausgeführt. Danach wird er automatisch beendet. Bei dem “Dialog-Model“, erfolgt zuerst der Start des Skills und anschließend die Ausführung seiner Funktion. Im Dialog-Modus bleibt der Skill solange geöffnet, bis er explizit geschlossen wird. In diesem Modus ist es nicht notwendig erneut den Wortlaut “Alexa“ für Anweisungen zu benutzen. In unserem Skill verwenden wir das “Dialog-Schema“. Das Schema wird in einem JSON-Format übergeben. Dieses besteht aus einem Array mit verschiedenen Intents, jeder hat einen Namen/Identifier und kann sogenannte Slots besitzen. Ein Intent ist ein Identifier für eine Funktion des Skills. Nehmen wir als Beispiel einen selbst erstellten FerienIntent. Dieser findet die nächsten Ferien herraus und liefert sie zurück. Mit den Custom Slot Types, werden eigene Datentypen definiert. Bei der Abbildung sieht man zum Beispiel “LIST OF TEACHERS“. Darin stehen alle verfügbaren Lehrer, die mit dem Skill abgedeckt werden. Der Slot Type kann von Slots benutzt werden und ist somit der variable Teil eines Satzes, der in den “Sample Utterances“ definiert wird. [12]

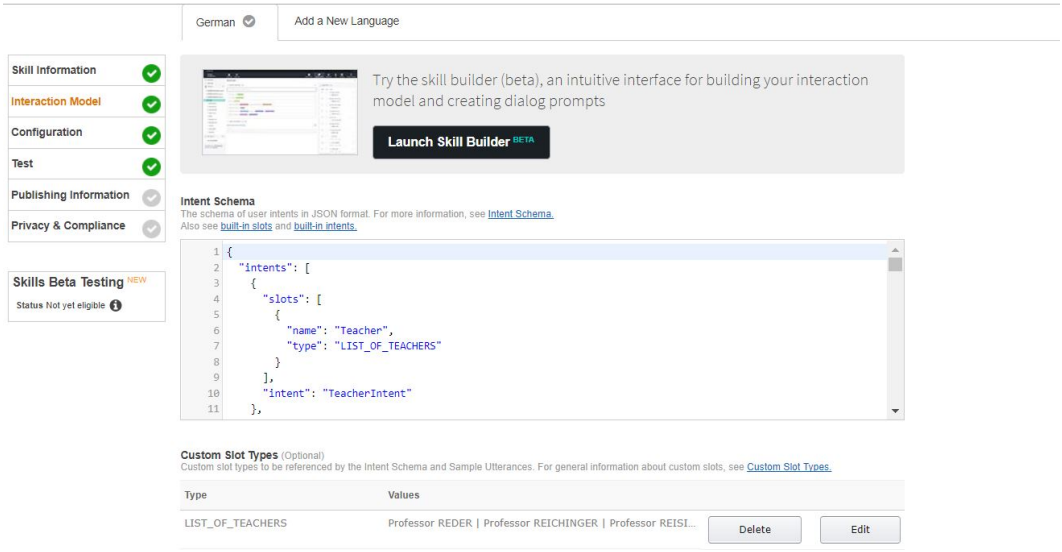


Abbildung 4.2: Interaction Model

Der Intent mit dem Namen “TeacherIntent“, hat den Slot “Teacher“. Des Weiteren gibt es auch vorgefertigte Intents von Amazon. Dazu gehören der “ AMAZON.HelpIntent“ oder der “AMAZON.StopIntent“, welche einerseits dazu benutzt werden, um Alexa nach Hilfe zu fragen oder andererseits um den Skill zu beenden. Der Bereich “Sample Utterances“ definiert jene Sätze, die von Alexa verstanden werden sollen. Dabei wird der Identifier des jeweiligen Intents davor geschrieben. [12] In der geschwungenen Klammer steht der jeweilige Slot.

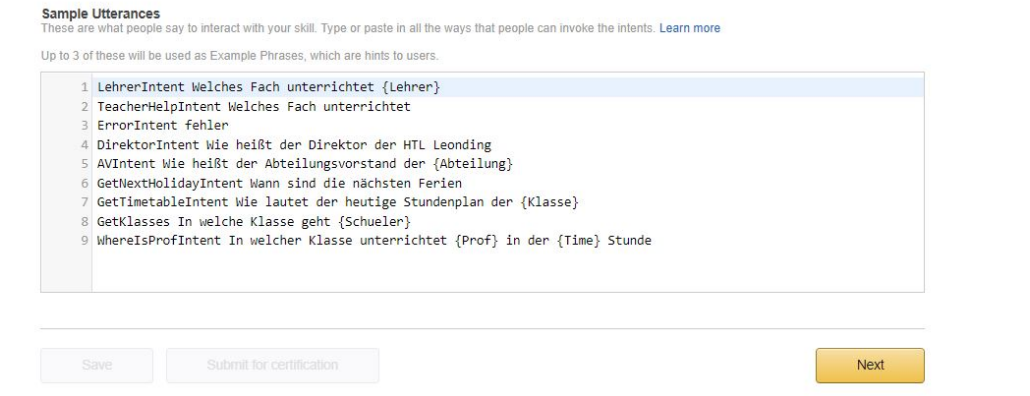


Abbildung 4.3: Sample Utterances

- **3.Coding in Amazon Web Services**
  Wir verwenden die Hosting Plattform AWS Lambda und als Programmiersprache Javascript, welche Node.js als Laufzeitumgebung verwendet. AWS Lambda funktioniert so, dass ein Request gesendet wird, dann wird der Code ausgeführt und liefert ein Ergebnis zurück. Mit AWS Lambda funktioniert die Zusammenarbeit mit Alexa problemlos und ist sehr effizient. Des Weiteren muss man sich nicht um SSL-Zertifikate oder Ähnliches kümmern. [12]

  Nach dem Erstellen einer Funktion, gelangt man zu diesem Fenster, wo unsere Funktion implementiert wird:

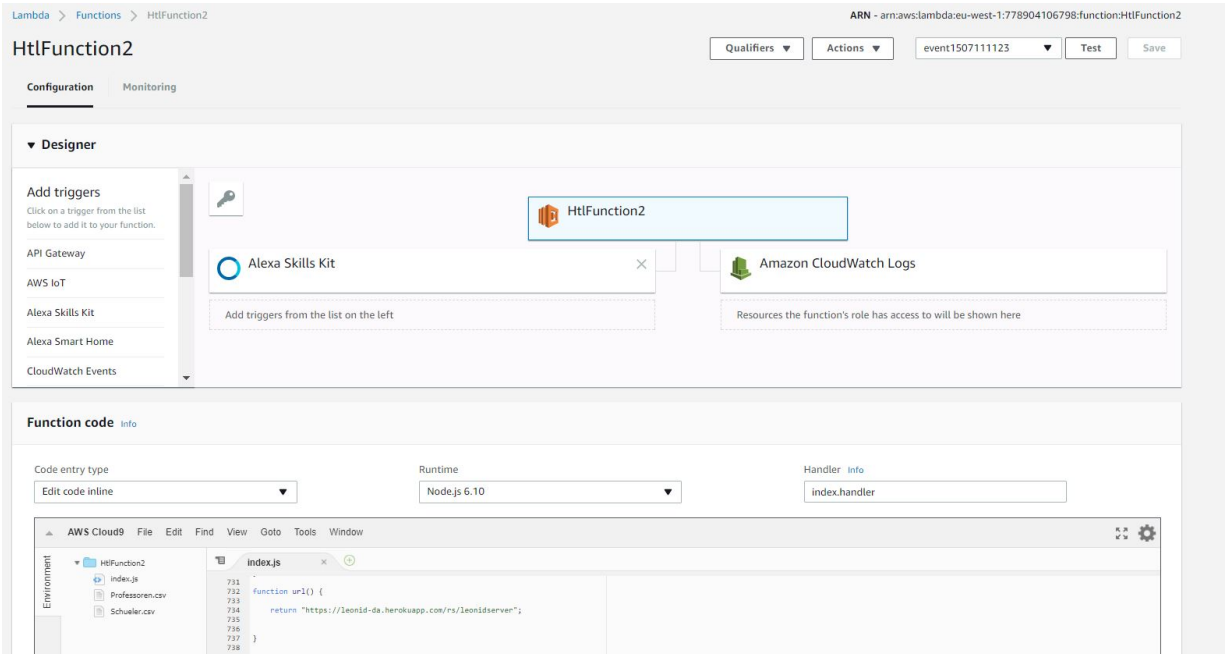


Abbildung 4.4: Lambda Function in Amazon Web Services

In unserem Code wird zuerst überprüft um welchen Request es sich handelt. In unserem Fall gibt es:

- LaunchRequest
- IntentRequest
- SessionEndedRequest.

Der **LaunchRequest** wird aufgerufen, wenn der Skill gestartet wird, also in diesem Beispiel, wenn „Alexa erwache“ gesagt wird.

Der **SessionEndedRequest** wird aufgerufen, wenn der Skill beendet wird. Zum Beispiel mit “Stop“.

Der **IntentRequest** wird in allen anderen Fällen aufgerufen, wenn dazu ein passender Intent erstellt wurde. Wenn dieser aufgerufen wird, werden wir in die Methode “onIntent()“ weitergeleitet.

```
exports.handler = function (event, context) {
    try {
        console.log("event.session.application.applicationId=" + event.session.application.applicationId);


        if (event.session.new) {
            onSessionStarted({requestId: event.request.requestId}, event.session);
        }

        if (event.request.type === "LaunchRequest") {
            onLaunch(event.request,
                event.session,
                function callback(sessionAttributes, speechletResponse) {
                    context.succeed(buildResponse(sessionAttributes, speechletResponse));
                });
        } else if (event.request.type === "IntentRequest") {
            onIntent(event.request,
                event.session,
                function callback(sessionAttributes, speechletResponse) {
                    context.succeed(buildResponse(sessionAttributes, speechletResponse));
                });
        } else if (event.request.type === "SessionEndedRequest") {
            onSessionEnded(event.request, event.session);
            context.succeed();
        }
    } catch (e) {
        context.fail("Exception: " + e);
    }
};
```

Abbildung 4.5: Node.js Code

In der Methode “onIntent()“ wird der “intentName“ mit den verschiedenen erstellten “Intents“ verglichen. Wird der passende gefunden, so wird die passende Methode dazu aufgerufen. Wird davon ausgegangen, dass es ein “LehrerIntent“ ist, so wird die Methode “handleTeacherIntent()“ aufgerufen.

```
function onIntent(intentRequest, session, callback) {

    var intent = intentRequest.intent;
    var intentName = intentRequest.intent.name;



    if (intentName === "TeacherIntent" || intentName === "LehrerIntent") {
        handleTeacherIntent(intent, session, callback);
    } else if (intentName === "AMAZON.YesIntent") {
```

Abbildung 4.6: Node.js Code

In der “handleTeacherIntent()“ Methode, wird aus dem Parameter “intent“, der Name des Professors ausgelesen. Das bedeutet, wenn Alexa gefragt wird: “Welches Fach unterrichtet Professor XY“, bekommen wir so “Professor XY“ oder nur “XY“. Das kommt darauf an, wie “LIST OF TEACHERS“ in den SlotTypes definiert wurde.

```
var lehrer = intent.slots.Lehrer.value.toLowerCase();
```

Abbildung 4.7: Node.js Code

Nachdem der Code verarbeitet worden ist, muss Alexa's Antwort in korrekter Form zurückgeben werden. Dazu werden verschieden Variablen verwendet:

```
var speechOutput = "Professor "+ teacher  + " unterrichtet " + arr[1];


var reprompt = "Möchten sie noch etwas wissen?";

var header = "HTL Leonding Teacher";

var shouldEndSession = false;

var sessionAttributes = {
    "speechOutput": speechOutput,
    "repromptText": reprompt

}

callback(sessionAttributes, buildSpeechletResponse(header, speechOutput, reprompt, shouldEndSession));
```

Abbildung 4.8: Node.js Code

- **speechoutput**: Das ist der Text, mit dem Alexa antworten soll.
- **reprompt**: Das ist der Text, mit dem Alexa antwortet, wenn für eine gewisse Zeit nach ihrer Antwort nichts gefragt wird.
- **header**: Diese Variable beinhaltet den Text, der später in der Alexa-App aufscheint, wenn die Funktion benutzt wurde.
- **shouldEndSession**: Diese Variable ist dafür zuständig, dass Alexa weiß, ob sie nach der Antwort den Skill beenden soll oder nicht.
- **sessionAttributes**: Mit dieser Variable wird aus der speechOutput-Variable und der reprompt-Variable ein JSON-Objekt zusammengestellt.
- **callback**: Hier wird die Methode "buildSpeechletResponse()" aufgerufen.

```
function buildSpeechletResponse(title, output, repromptText, shouldEndSession) {
    return {
        outputSpeech: {
            type: "PlainText",
            text: output
        },
        card: {
            type: "Simple",
            title: title,
            content: output
        },
        reprompt: {
            outputSpeech: {
                type: "PlainText",
                text: repromptText
            }
        },
        shouldEndSession: shouldEndSession
    };
}
```

Abbildung 4.9: Node.js Code

Die Methode buildSpeechletResponse() erstellt aus den übergebenen Parametern ein JSON-Objekt, das für Alexa lesbar ist.

Wenn unsere Logik ausgeführt wurde, wird die Methode buildResponse() aufgerufen, in der das endgültige JSON erstellt und zurückgeliefert wird:

```
function buildResponse(sessionAttributes, speechletResponse) {
    return {
        version: "1.0",
        sessionAttributes: sessionAttributes,
        response: speechletResponse
    };
}
```

Abbildung 4.10: Node.js Code

Zurück in der Alexa-Developer-Konsole wird nun unter Configuration der Service Endpoint Type gewählt. Dieser ist AWS Lambda. Des Weiteren wird Europa selektiert und unser Lambda-Identifier wird eingefügt. Die Verbindung vom Alexa Skill zum Backend ist nun hergestellt. [12]

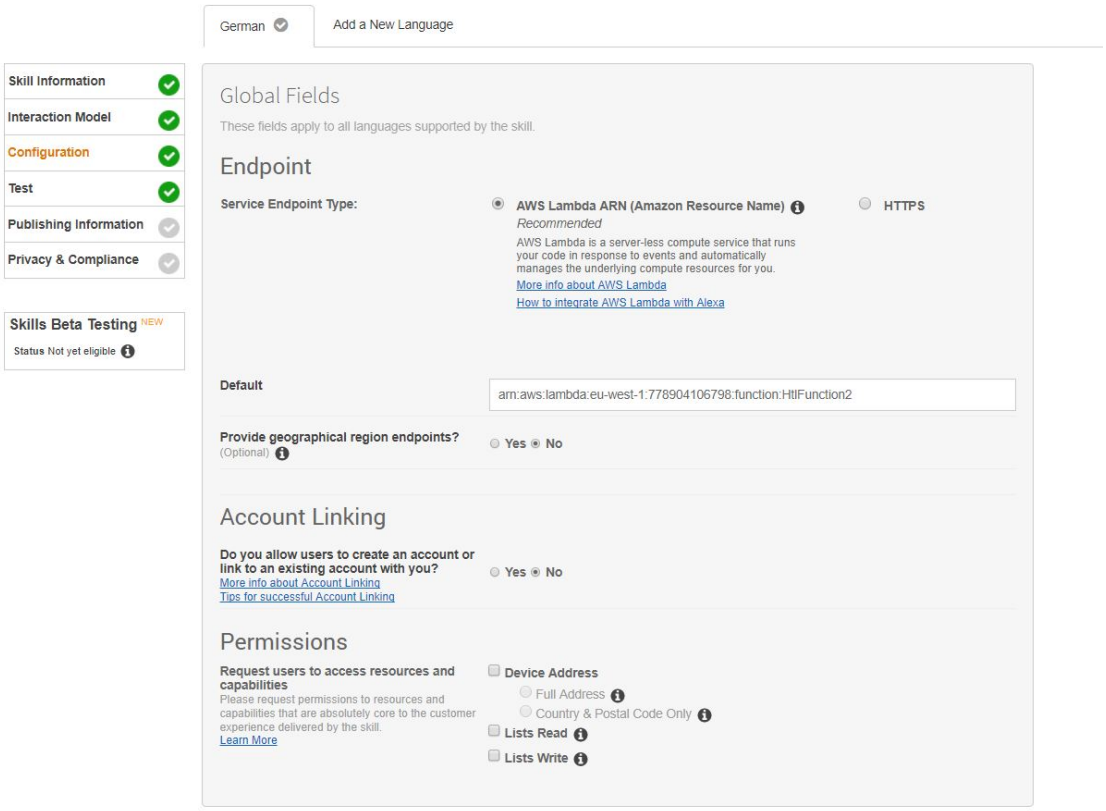


Abbildung 4.11: Amazon Developer Configuration

- **4.Testen eines Skills**
  Um einen Skill zu testen gibt es 3 Möglichkeiten:
  - **Direkt am Amazon Echo**
  - **In der Amazon Developer Console**
    In der Amazon Developer Console, gibt es ein Fenster welches "Test" heißt. Hier kann man seinen Skill durch das Eingeben der Sätze, welche Alexa beantworten soll, testen. Wenn ein Satz eingegeben wird, sieht man auf der linken Seite, den Request in einem JSON-Format. Auf der rechten Seite erhält man die Antwort im JSON-Format. Und darunter wird der Satz, mit dem Alexa antwortet ausgegeben.

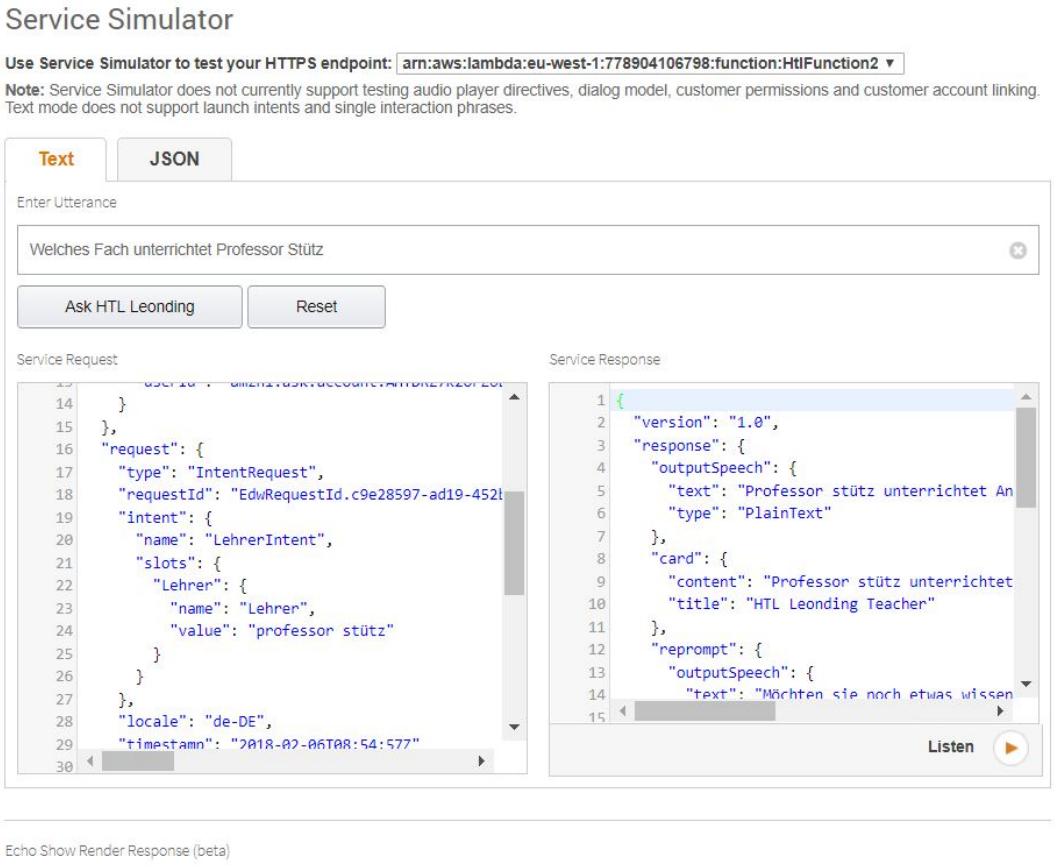


Abbildung 4.12: Amazon Developer Test-Fenster

– **In Amazon Web Services** Des Weiteren kann man auch in Amazon Web Services testen. Dazu wird der Request vom Test Fenster kopiert. Hier kann man bis zu 10 verschiedene Testfälle anlegen. Das funktioniert mit Configure Test Events:

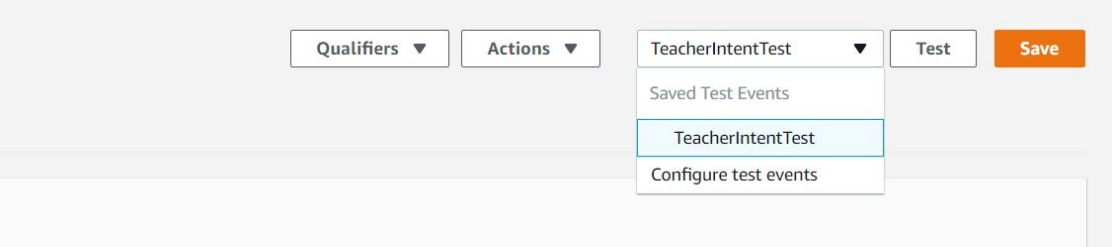


Abbildung 4.13: Configure Test Events

In diesem Fenster wird jetzt unser zuvor kopierter Request eingefügt.

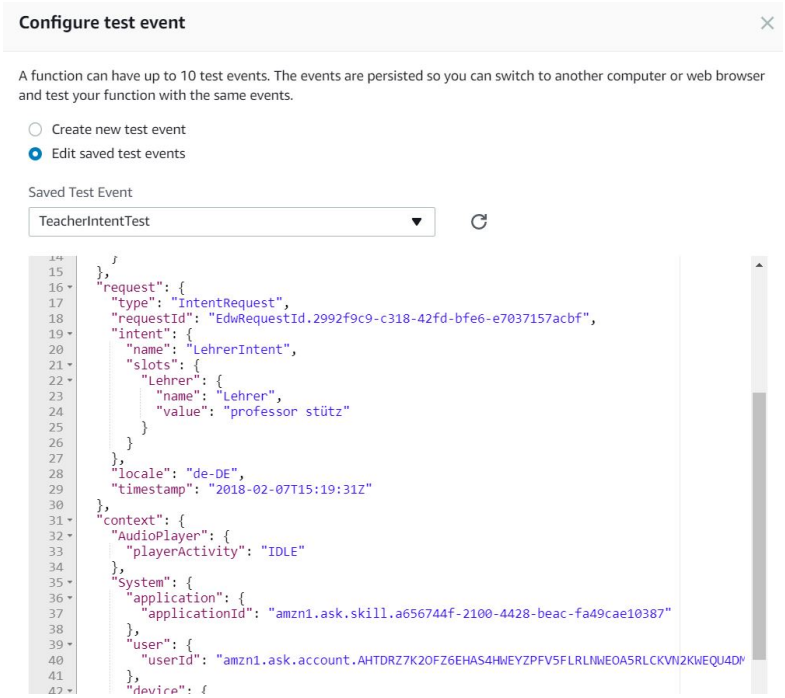


Abbildung 4.14: Configure Test Event

Durch Klicken auf den Test-Button, wird der Code getestet.

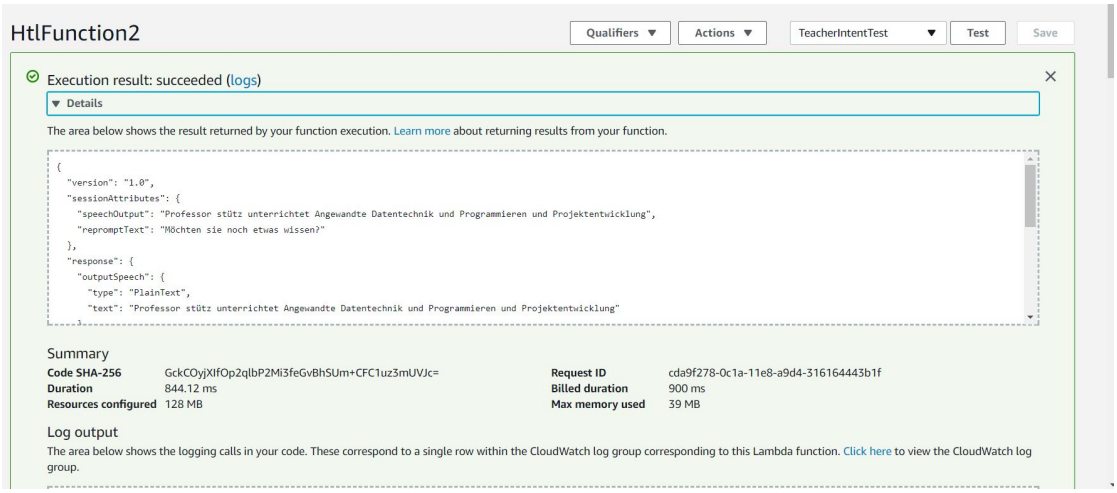


Abbildung 4.15: Test Ergebnisse

Kapitel 5

WebUntis-Daten abrufen

WebUntis ist eine Plattform für Schüler und Lehrer. Mit dieser können Stundenpläne abgerufen, Abwesenheiten angezeigt und Tests eingetragen werden. Diese Informationen können mittels App oder Website abgerufen werden. Oder wie in unserem Fall über die Rest-Schnittstelle von Web Untis. Der verwendete URL lautet: “https://mese.webuntis.com/WebUntis/jsonrpc.do.“ Der Content Type wird auf “application/json“ eingestellt und der Request Type ist “POST“.

5.1 Authentifizierung

Bevor Daten abgerufen werden können, wird eine Authentifizierung benötigt. Der URL wird mit dem Parameter “?school=htbla linz leonding“ erweitert. Das folgende JSON-Object gibt man im Body mit:

```
{
        "id": "OWNID",
        "method": "authenticate",
        "params": {"user": "OWNUSER", "password" : "OWNPASSWORD", "client":"CLIENT"},
        "jsonrpc": "2.0"
}
```

Abbildung 5.1: WebUntis Authentification

- **id:** Ist ein eigens gewählter Identifier, der die gesamte Session mitgegeben wird.
- **method:** Es gibt verschiedene Methoden, die verwendet werden können. Entweder wie hier “authenticate“, oder des Weiteren gibt es Methoden wie “getTeachers“ oder “getTimetable“.
- **params:** Die Parameter sind je nach Methodenart unterschiedlich. Hier sind es der WebUntis-User + Passwort, die man mitgeben muss.

5.1.1 Response der Authentifzierung

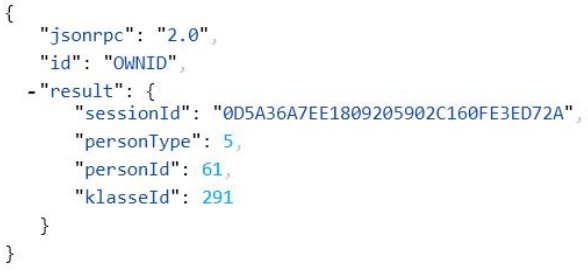


Abbildung 5.2: Authentifizierung Response

Die Rückmeldung der Authentifizierung gibt eine “sessionId“ aus. Diese wird in den nächsten Request im URL mitgegeben. Der “personType“ von fünf, bedeutet, dass sich ein Schüler authentifiziert hat. Die “personId“ ist die Id des Schülers und die “klasseId“ ist die Id der Klasse des jeweiligen Schülers.

5.2 Daten abrufen

Nachdem die Authentifizierung erfolgreich war, kann man nun Daten abrufen.

5.2.1 Professoren abrufen

Über folgendes JSON-Object werden die gesamten Professoren der Schule abgerufen:

```
{
        "id": "OWNID",
        "method": "getTeachers",
        "params": {},
        "jsonrpc": "2.0"
}
```

Abbildung 5.3: JSON-Object getTeachers

5.2.2 Stundenplan abrufen

Es gibt fünf verschiedene Arten von Stundenplänen, die abgerufen werden können.

- Für eine Klasse
- Für einen Lehrer
- Für einen Unterrichtsgegenstand
- Für einen Raum
- Für einen Schüler

Um das auszuwählen wird im Body der jeweilige "Type" und die Id mitgegeben. Wenn man den Stundenplan eines Lehrers benötigt, könnte das JSON-Object wie folgt aussehen.

```
{
        "id": "OWNID",
        "method": "getTimetable",
        "params": {"type": 2, "id" : 10},
        "jsonrpc": "2.0"
}
```

Abbildung 5.4: JSON-Object getTimetable

Kapitel 6

Aufbau

Für den Aufbau des Hologramms wurde ein Regal verwendet. In diesem ist die Pyramide aufgebaut, welche von oben durch einen Fernseher bestrahlt wird. Neben der Pyramide befindet sich der Amazon Echo Dot. Damit das Hologramm besser erkennbar ist, wurde hinter der Pyramide eine schwarze Fläche angebracht. Der Fernseher ist mittles HDMI Kabel mit dem Raspberry Pi verbunden, auf welchem die Website läuft.

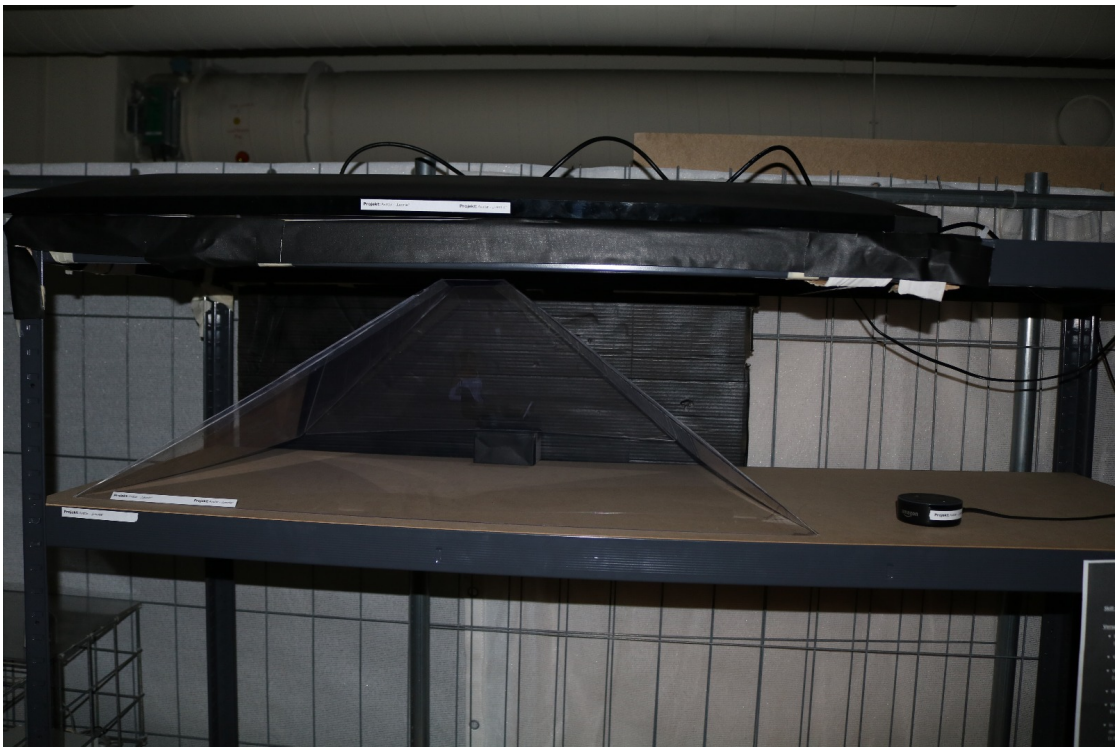


Abbildung 6.1: Aufbau

6.1 Hologrammtechnik

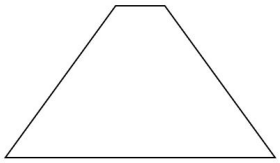


Abbildung 6.2: Eine Seite der Pyramide

Für die Darstellung des Hologramms, wird eine Pyramide aus Plexiglas verwendet. Diese besteht aus vier Plexiglasscheiben. Jede Scheibe hat die Form eines Trapezes. Die beiden unteren Ecken stehen im 45 Grad Winkel. Die Scheiben werden im 45 Grad Winkel miteinander verklebt. Durch die Bestrahlung mit dem TV-Gerät, wirkt es so, als ob die Figur im Raum stehen würde. Um das Hologramm so gut wie möglich sichtbar zu machen, wurde der Fernseher auf die hellste Bildeinstellung gestellt.

6.2 Hologramm

Abbildung 6.3: Leonie von Vorne

Abbildung 6.4: Leonie Seitlich

Kapitel 7

Verwendete Technologien

7.1 Amazon Echo

Amazon Echo ist ein persönlicher Assistent, der auf Internet basiert und eine Lautsprecherfunktion besitzt. [7] Alexa besteht aus einem Zwei-Wege-Lautsprecher mit Subwoofer. Zudem besitzt es sieben Mikrofone und kann somit auch leise Befehle erfassen. [5]

Abbildung 7.1: Amazon Echo [17]

7.1.1 Sprachsteuerung

Das Gerät funktioniert so, dass es durch bestimmte Signalwörter aktiviert wird. Solche wären: Echo, Alexa, Computer oder Amazon. Es kann auch durch Drücken auf die Aktionstaste aktiviert werden. Die nach der Aktivierung gesprochenen Befehle, werden danach über das Internet zum Hersteller übertragen, wo sie dann umgesetzt werden. Das Gerät muss immer über eine gültige Internetverbindung verfügen. Wenn Alexa ein sogenanntes Signalwort wahrgenommen hat, wird das über die farbige Leuchte oberhalb des Gerätes angezeigt. [7]

7.1.2 Skills

Für Alexa werden viele verschiedene Skills angeboten, also Anwendungen, die dazu dienen Alexa um weitere Funktionen zu erweitern. Amazon Echo besitzt ohne jegliche Erweiterung nur beschränkte Fähigkeiten. Dazu gehört das Abspielen von Musik, die Auskunft über die Wetterprognose oder das Bestellen bei Amazon. Weitere Funktionen werden durch sogenannte Skills hinzugefügt. Mit den Erweiterungen durch Skills kann man dann zum Beispiel Smarte-Geräte steuern, oder den neuesten Fahrplan der Straßenbahn anzeigen lassen. Diese Funktionen kann man durch die Alexa-App oder im Amazon Skill-Store aktivieren. [11]

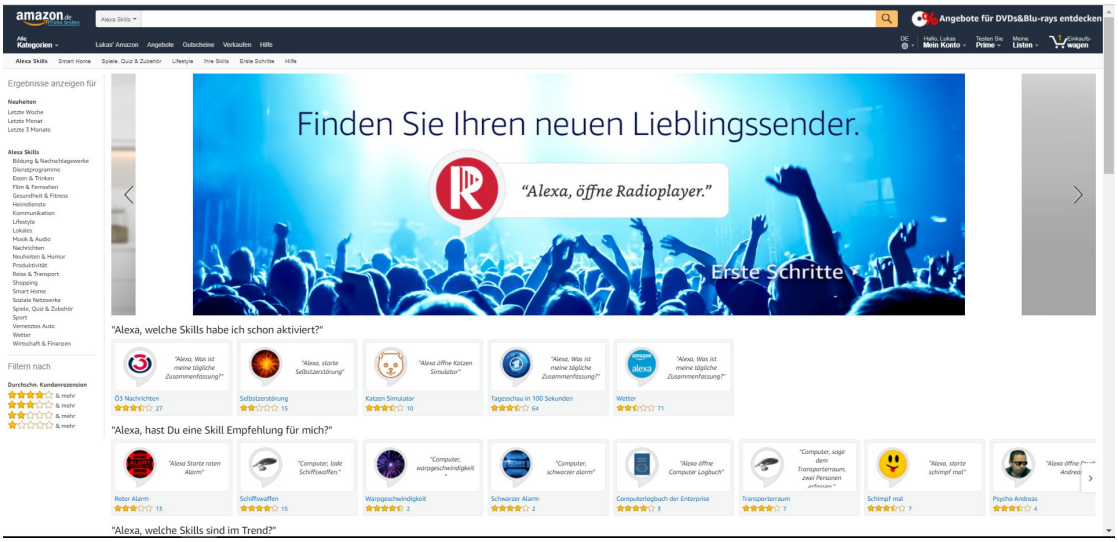


Abbildung 7.2: Alexa Skills Shop

7.2 Amazon Developer Services

Amazon Developer Services ist eine Ansammlung von Tools und Services, mit denen ein Entwickler Applikationen builden, testen und im Amazon Appstore verkaufen kann. Es beinhaltet auch ein Development Kit für Android oder IOS, Fire Tablets, Fire TV und Alexa Geräten. [13]

7.2.1 Amazon Web Services

Amazon Web Services ist eine Plattform für Cloud-Services. Hier werden Rechenleistung, Datenbankspeicherung und viele weitere Funktionen geboten. [6]

7.2.2 Alexa Skills Kit

Das Alexa Skills Kit ist ein Software Development Kit, mit dem Entwickler Skills erstellen können. Es besteht aus verschiedenen Tools, Codebeispielen, API's und Dokumentationen. [14]

7.3 Git

Git ist ein open source Versionsverwaltungsprogramm, das zum Verwalten von Versionen verwendet wird. Konkret heißt dies, dass von Dateien Versionen erzeugt werden können, die dann sinnvoll verwaltet werden. Bei Versionsverwaltungssystemen werden zusätzlich zu den reinen Änderungen auch noch weitere Informationen zu einer Version mitgespeichert. Dazu gehören der Autor, die Uhrzeit und ein Änderungstext. Durch die gesammelten Dateien lässt sich schnell und einfach eine Änderungshistorie anzeigen und verwalten.
Man kann einfach zwischen den einzelnen Versionen hin und her wechseln.
Es gibt grundsätzlich drei verschiedene Konzepte zur Versionsverwaltung:[1]

Abbildung 7.3: Git Logo [21]

7.3.1 Lokale Versionsverwaltung

Die lokale Versionsverwaltung wird eher selten verwendet, da sie nur lokal arbeitet und oft auch nur einzelne Dateien versioniert. [1]

7.3.2 Zentrale Versionsverwaltung

Das Hauptmerkmal einer zentralen Versionsverwaltung ist, dass das Repository auf einem zentralen Server liegt. Ein Repository ist ein "Lager", in dem die Daten liegen. Nutzer eines Repositorys arbeiten dabei lokal mit einer Arbeitskopie der im Repository vorhandenen Dateien. Wenn man sich eine alte Version anschauen möchte, werden die Daten stets vom Server heruntergeladen. [1]

7.3.3 Verteilte Versionsverwaltung

Zu den verteilten Versionsverwaltungssystemen gehört Git. Der Unterschied zu zentralen Versionsverwaltungssystemen ist, dass hier jeder Nutzer nicht nur eine Arbeitskopie besitzt, sondern das gesamte Repository. [1]

7.3.4 Arbeiten mit Git

Ein Git-Repository besteht aus mehreren verschiedenen Bestandteilen. Ein Repository besitzt diverse Branches, die wiederum aus mehreren Commits bestehen. [1]

- **Was ist ein Commit?**

  Ein Commit stellt in einem Repository eine Änderung dar. Dazu gehört der Zeitpunkt des Commits, der Autor, eine Änderungsnotiz und die Änderung selbst. Die Änderungen können groß und klein sein. Man kann entweder völlig neue Dateien dem Repository hinzufügen oder auch nur kleine Sachen in der Datei ändern. Auch das Löschen von Zeilen oder Dateien kann in einem Commit erfasst werden. [1]

- **Was ist ein Branch?**

  In diesem Fall lässt sich das Wort Branch am besten mit "Zweig" übersetzen. Mit einem Branch ist es möglich, den aktuellen Stand der Dateien abzuzweigen und daran weiter zu arbeiten. Es wird quasi eine Kopie erzeugt, auf der weitere Commits getätigt werden können, ohne einen anderen Entwicklungsbranch zu berühren. In der Praxis werden meist in der Softwareentwicklung in einem Branch eigene Features entwickelt. [1]

- **Was ist Mergen?**

  Da man in der Praxis in mehreren Branches entwickelt, muss man, wenn die Features fertig implementiert sind, die Änderungen natürlich wieder in den Hauptentwicklungsbranch integrieren. In größeren Projekten bleibt der Haupt-Branch, nicht ohne Änderung, das heißt die beiden Branches müssen vereinigt werden, also "gemerged". Es kann auch vorkommen, dass dabei Konflikte auftreten. Zum Beispiel wenn 2 verschiedene Benutzer etwas an der gleichen Zeile geändert haben. Solche Probleme muss der Benutzer dann händisch lösen. [1]

7.3.5 Was ist GitHub?

Git-Repositorys werden optimalerweise auf einem Server gelagert. Wer sich aber die Mühe ersparen möchte sich einen eigenen Server zu konfigurieren, der verwendet GitHub. GitHub ist ein Hosting Service, der den Austausch zwischen mehreren Teilnehmern eines Projektes vereinfacht. [1]

7.4 Heroku

Heroku ist eine Plattform welche zu den Vertretern der PaaS (Platform as a Service) zählt. Es bietet die Möglichkeit Anwendungen von den verschiedensten Sprachen in die Cloud zu deployen. Dort kann man sie laufen lassen und administrieren. Beispiele für die verfügbaren Sprachen sind Node.js, Java, Python. Eine Heroku Anwendung ist eine Sammlung aus Quellcode, Ressourcedateien und Build-Skripten. [2]

Abbildung 7.4: Heroku Logo [18]

7.4.1 Was benötigt man für das Deployment?

- Den Code der Applikation
- Ein Dependency File (z.B. in Java eine pom.xml)
- Ein Procfile(Es definiert welcher Teil der Anwendung, wie gestartet wird)

[2]
So könnte ein Procfile aussehen:

```
web: java -Dswarm.http.port=8080 -jar target/at/htl/leonid/leonidServer
/1.0-SNAPSHOT/leonidServer-1.0-20171002.102249-1-swarm.jar
```

Abbildung 7.5: Procfile

7.4.2 Wie deployed man auf Heroku?

Damit man eine Anwendung zum Laufen bringt, muss diese zunächst auf Heroku übertragen werden. Hierzu gibt es verschiedene Möglichkeiten: Git, GitHub oder Dropbox. Nachdem die Anwendung auf GitHub gepusht wurde, wird mit einfachen Befehlen Heroku darin initialisiert. [2]

```
heroku create
```

Abbildung 7.6: heroku create

Wenn Heroku den Code der Applikation erhalten hat, wird der Build initialisiert. Die passiert über einen GitHub Webhook, der nach jeden neuen push auf den Heroku Remote Master Branch der Applikation greift. [2]

```
git push heroku master
```

Abbildung 7.7: push heroku master

Um den Build der Applikation zu öffnen, wird folgender Befehl verwendet:

```
heroku open
```

Abbildung 7.8: heroku open

7.5 Texmaker

Die Ausarbeitung der Diplomarbeit erfolgte in Texmaker. Dieses Programm dient zur Erstellung und Editierung von LateX-Dateien. Texmaker bringt viele Vorteile mit sich:

- sehr schlank und schnell
- Ist für Windows, Unix, macOS verfügbar

[15]

7.5.1 LateX

LateX beschreibt man am besten mit einem Textvearbeitungsprogramm. Meistens wird es für die Erstellung von Diplomarbeiten oder Büchern verwendet. Für die Erstellung von alltäglichen Dokumenten wird es allerdings nicht verwendet. [16]

7.6 Wildfly

Der Wildfly Application Server ist ein Plattform unabhängiger Server, auf dem JavaEE Applikationen laufen gelassen werden können. Da dieser Service kostenlos ist, eignet sich dieser sehr gut für die selbst entwickelte JavaEE Applikation. Der Server läuft anschließend in einem Docker Container, welcher wiederum auf einem öffentlich zugänglichen Server läuft, damit der AlexaSkill ebenfalls auf den Java Server zugreifen kann. [3]

7.7 Docker

Durch Docker ist es möglich, leicht, unkompliziert und schnell mehrere Server auf einer Maschine laufen zu lassen. Wo zuvor noch etliche Virtual Machines aufgesetzt werden mussten, was starke Auswirkungen auf die Performance hatte, da es sich dabei immer um vollwertige Betriebsysteme handelte, werden diese nun durch Docker Container ersetzt. Jeder Container beschäftigt eine Anwendung. Ein weiterer Vorteil vom Container System ist, dass sie auf einem Image basieren, welches immer wieder aufs neue verwendet werden kann. Zum Beispiel wenn mehrmals eine Instanz einer Anwendung laufen soll. Beim System mit den Virtual Machines musste in so einem Fall immer wieder eine komplette neue VM aufgesetzt werden.

7.7.1 Dockerfile

```
FROM jboss/wildfly

RUN /opt/jboss/wildfly/bin/add-user.sh admin passme --silent

COPY target/leonidserver.war /opt/jboss/wildfly/standalone/deployments/leonidserver.war
```

Abbildung 7.9: Dockerfile

Ein Dockerfile hat einen bestimmten Aufbau:

- FROM
  mit dieser Anweisung wird ein bereits vorbereitetes Image von Dockerhub runtergeladen
- RUN
  mit dieser Anweisung können Skripts ausgeführt werden, wie etwa das add-user Skript
- COPY
  mit dieser Anweisung können Dateien in den Docker Container kopiert werden, wie etwa das .war File einer JavaEE Applikation

7.7.2 Erstellen und Ausführen eines Containers

- Wurde ein Dockerfile erstellt, muss “docker build -t NameFürDasImage“ im Ordner in dem sich das Dockerfile befindet ausgeführt werden, um aus dem Dockerfile ein Image zu erstellen.
- Anschließend wird mit “docker run -d -p 8080:8080 –name <containerName><imageName>“ der neuen Container gestartet. Mit -d läuft der Container im Hintergrund und mit -p wird angegeben, welche Ports verwendet werden sollen.

7.8 Raspberry Pi

Der Raspberry Pi ist ein Minirechner, der von der britischen Raspberry Pi Foundation entwickelt wurde. Das auf nur einer Platine untergebrachte Gerät wurde dafür entwickelt, jungen Menschen das Programmieren zu erleichtern und ihnen einen Einstieg über die Computerhardware zu vermitteln. [8]

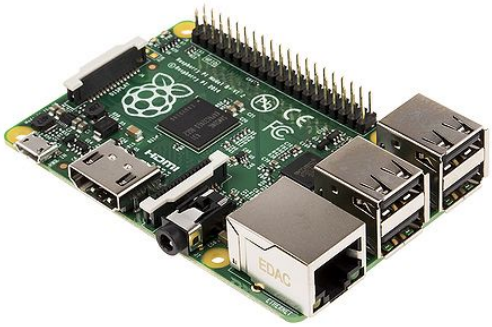


Abbildung 7.10: Raspberry Pi [20]

7.8.1 Grundlegendes

Der Raspberry Pi besitzt die ungefähre Größe einer Kreditkarte und liegt in einem Preissegment zwischen fünf und vierzig Euro. Seine grundsätzliche Ausstattung besteht aus einem ARM-Prozessor, einem Grafikprozessor und Arbeitspeicher mit unterschiedlichen Größen. [8]
Er verfügt über mehrere Schnittstellen wie Ethernet, USB oder 21 GPIO-Pins. Damit können weitere Geräte mit dem Pi verbunden werden. Weiters besitzt er eine HDMI-Schnittstelle, mit der sich ganz einfach grafische Inhalte über einen Bildschirm darstellen lassen. Das auf einer Speicherkarte installierte Betriebssystem wird in einen SD-/MMC-Karten-Slot eingeführt. [8]
Es gibt auch noch verschiedene Erweiterungsmodule,welche drahtlose Verbindungen per Bluetooth oder WLAN ermöglichen. [8]

7.8.2 Betrieb

Die betriebsnotwendige Energie liefert ein externer Adapter, der über Micro-USB verbunden ist. Bildschirm und Tastatur werden via HDMI beziehungsweise via USB angeschlossen. Um den Raspberry Pi zu starten, bieten sich zum Beispiel unterschiedliche Linux- oder Windows-Versionen an. Als vorteilhaft gilt die Linux-Distribution Raspian, welche auf Debian aufbaut. [8]

7.8.3 Erweiterungsmöglichkeiten

Der Raspberry Pi erlaubt zahlreiche Erweiterungen. Jene um WLAN-Sticks, Festplatten oder sämtliche USB-Devices sind sehr einfach über die USB-Schnittstelle durchführbar. Möglichkeiten zur Erweiterung sind ebenfalls über die an die GPIO-Pfostenleisten anschließbaren Erweiterungsboards möglich. [8]

7.8.4 Einstellungen am Raspberry PI

- **Ubuntu Mate**
  Am Raspberry PI wurde das Betriebssystem Ubuntu Mate installiert.
- **Firefox**
  Als Browser verwenden wir den Mozilla Firefox Browser. Wir haben uns für diesen entschieden, weil er die Animationen am schnellsten und am flüssigsten abgespielt hat.
  - Der Browser wurde in den Autostart Ordner gelegt.
  - Es wird automatisch die richtige Website geöffnet
  - Der Browser wird beim Öffnen sofort in den Fullscreen-Modus versetzt

7.9 Node.js

Die auf Chromes V8 JavaScript-Engine basierende Laufzeitumgebung, die in JavaScript realisiert wurde, besitzt ein event-basiertes, blockierungsfreies I/O-Modell, durch welches sie effizient und schlank ist. [10]

Abbildung 7.11: Node.js Logo [19]

7.9.1 Eigenschaften

In Node.js kommt das Single-Threading zur Verwendung. Dadurch ist es nicht möglich mehrere Befehle parallel auszuführen. Allerdings erleichtert es die Entwicklung, da nicht so hohe Fähigkeiten benötigt werden. Im Gegensatz dazu wird beim Multi-Threading auf mehrere Threads gesetzt. Hierbei gilt es auf Thread-Safety, also dass keine parallelen Threads auf ein gleiches Programmteil zugreifen, zu achten. Damit beim Single-Threading das Blockieren eines Prozesses verhindert wird, verwendet Node.js ein ausgelagertes Input/Output-System. Da in der Bearbeitungszeit einer Anfrage der Prozess blockiert ist, findet eine Auslagerung des I/O-Systems statt, wo es asynchron ausgeführt wird. Das Zentrum der asynchronen Abarbeitung stellt der Event Loop dar. Er ist für die Annahme von Anfragen für I/O und Callbacks zuständig. Für die Weiterverarbeitung ist ein externer Prozess verantwortlich. Nach der Fertigstellung des I/O-Prozesses erfolgt eine Rückmeldung des Ergebnisses an den Event Loop. Dieser ruft den vorherig festgelegten Callback mit den Resultaten auf. Die Bearbeitung nimmt ihren gewöhnlichen Lauf. Die Verbindung des extern gelagertens I/O-Systems mit Event Loop und der Single-Thread-Technologie führt zu einer sehr leistungsfähigen Lösung. [9]

Kapitel 8

JavaEE Applikation

Um die Kommunikation zwischen der Website worauf die Animationen laufen und dem Alexa Skill zu gewährleisten, wurde sich für eine JavaEE Applikation entschieden. Diese Applikation wird anschließend auf einem WildFly Application Server bereitgestellt, welcher in einem Docker Container auf einer Virtual Machine läuft. Zum Entwickeln wurde als Entwicklungsumgebung die IntelliJ IDEA verwendet.

8.1 Maven

Maven ist ein Build-Tool für Java, womit die Einstellungen für die Applikation in der einer XML Datei vorgenommen werden können. Diese wird “pom.xml“ genannt. Darin können dann viele Einstellungen getroffen werden, wie etwa die Dependencies, welche bereits vorgefertigte Pakete sind, die bestimmte Probleme bereits gelöst haben, damit nicht alles selber ausprogrammieren werden muss oder die Art des Packaging, zum Beispiel als .war File oder .jar File.

Abbildung 8.1: pom.xml

8.2 REST Service

8.2.1 Was ist ein REST Service

Durch einen REST Service ist es möglich, schnell und einfach eine Schnittstelle zu erstellen, über welche Daten ausgetauscht werden können. Deswegen wurde als Kommunikationsmittel zwischen dem JavaEE Server und dem Alexa Skill ein REST Service eingesetzt. Dieser wird dazu verwendet, dass sobald ein Benutzer Alexa eine Frage stellt und diese dann antwortet, die dafür vorgesehene Animation aktiviert wird. Das geschieht über Requests, deren Namen selbstsprechend für dessen Funktion sind, wie etwa:

- GET
  dient dazu, Daten anzufordern
- POST
  dient dazu, Daten in einem Textformat, wie etwa JSON, an einen Server zu senden
- PUT
  wird hauptsächlich dafür verwendet, einen bereits vorhandenen Datensatz zu überarbeiten
- DELETE
  wird hauptsächlich nur dann verwendet, wenn mit einer Datenbank gearbeitet wird und ein Datensatz/mehrere Datensätze zu löschen sind

8.2.2 Aufbau in JavaEE und Implementierung

Um einen REST Service zu realisieren, benötigt man mindestens zwei Klassen:

- **RestConfig**
  in dieser Klasse wird der Pfad für den REST Service festgelegt und die Klasse mit "Application" erweitert.

```java
package at.htl.leonidServer;

import javax.ws.rs.ApplicationPath;
import javax.ws.rs.core.Application;

@ApplicationPath("rs")
public class RestConfig extends Application{
}
```

Abbildung 8.2: RestConfig Klasse

- **Endpoint**
  in dieser Klasse wird zuerst der weitere Pfad festgelegt, über welchen dann in der URL die Methoden auffindbar sind, danach werden die benötigten Requestmethoden erstellt. Diese werden mit einer @ Annotation gekennzeichnet, welche gleich lautet wie die Art von Request, den die Methode erledigen soll (zum Beispiel @GET).

```java
@Path("leonidserver")
public class Endpoint {
```

Abbildung 8.3: Festlegen vom Pfad

```java
@GET
@Path("animation")
@Produces({MediaType.APPLICATION_JSON})
public Response activateAnimation(@QueryParam("name") String name)
{
    videoname = name + ".mp4";
    alexaResponse = true;
    return Response.ok().entity("Animation " + name + " activated").build();
}
```

Abbildung 8.4: Beispiel für eine Requestmethode

8.3 Websockets

Durch Websockets wird eine Kommunikationsverbindung zwischen Website und Server hergestellt. Über diese Verbindung kann sowohl der Server an die Website Informationen senden, aber auch die Website an den Server.

8.3.1 Implementierung des Websocket Server

Für die Implementierung wird nur eine Klasse benötigt.

```
package at.htl.leonidServer;

import javax.websocket.Session;
import javax.websocket.OnMessage;
import javax.websocket.OnOpen;
import javax.websocket.server.ServerEndpoint;
import java.io.IOException;

@ServerEndpoint("/ws")
public class SocketController{
    private Session session;

    @OnOpen
    public void onCreateSession(Session session) { this.session = session; }

    @OnMessage
    public void onRequest(String message){
        try {
            if(Endpoint.alexaResponse == true){
                Endpoint.alexaResponse = false;
                this.session.getBasicRemote().sendText(Endpoint.videoname);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Abbildung 8.5: SocketController

- durch die @ServerEndpoint Annotation wird der Pfad für den Zugriff auf den Websocket festgelegt.
- in der Methode mit der @OnOpen Annotation werden all die Sachen erledigt, die ab dem Start des Websockets benötigt werden
- in der Methode mit der @OnMessage Annotation wird festgelegt, wie der Websocket reagieren soll, wenn der Server benachrichtigt wird.

Kapitel 9

Website

Für die Wiedergabe der Animationen gab es verschiedenste Lösungsansätze:

- ein Programm schreiben, welches lokal auf dem Rechner läuft
- eine Website entwickeln. Das hat den Vorteil, dass keine Abhängigkeit darin besteht, dass das Programm für das Hologramm am PC installiert ist. Außerdem ist es dadurch möglich, das Hologramm nicht nur am PC wiederzugeben, sondern auch auf Tablets, Handys, etc.

Wegen der klaren Vorteile einer Website, wurde dieser Weg eingeschlagen.

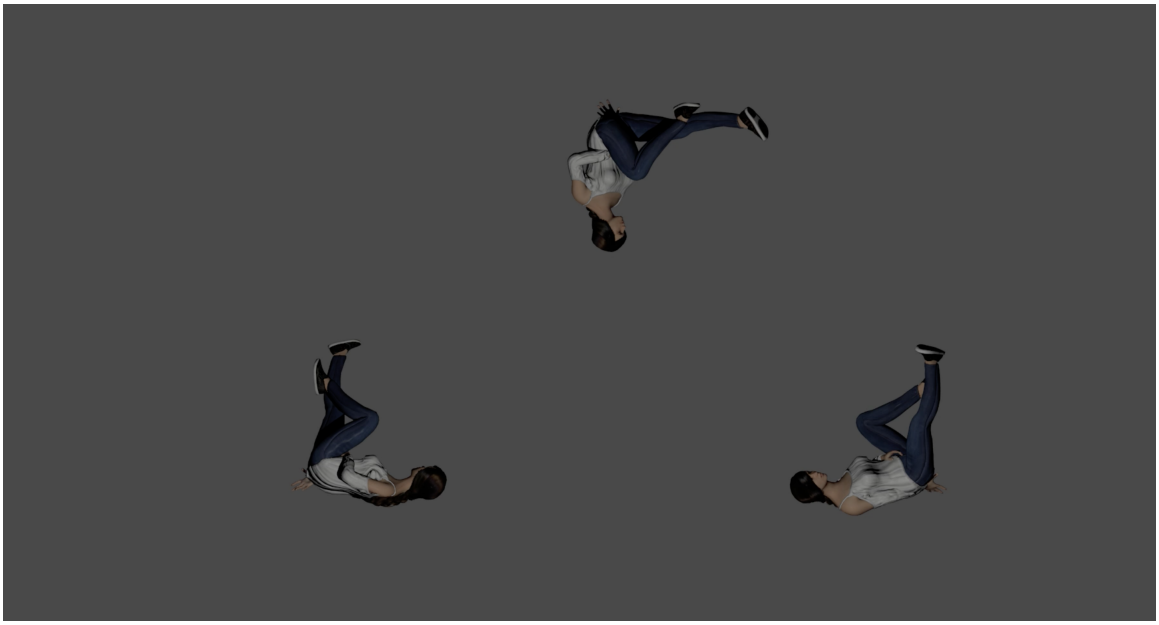


Abbildung 9.1: Website mit dem Avatar

9.1 Websocket Client

```
<script>
    var myPlayer = videojs('my-player');
    //var url = 'ws://localhost:8080/leonidserver/ws';
    var url = 'wss://leonid-da.herokuapp.com/ws';
    var ws = new WebSocket(url);
    setInterval(sendMessage, 5);
    function sendMessage(){...}
    ws.onconnect = function(e) {
        console.log("connected");
    };
    ws.onerror = function (error) {
        console.log('WebSocket Error ' + error);
    };
    ws.onclose = function(event){
        console.log("Remote host closed or refused WebSocket connection");
        console.log(event);
    };
    ws.onmessage = function(message) {
        myPlayer.src({type: 'video/mp4', src: message.data});
        console.log(message.data);
        myPlayer.load();
        myPlayer.play();
    };
</script>
```

Abbildung 9.2: JavaScript für die Websocket Verbindung

Damit die Website mit dem Server kommunizieren kann, musste noch mittels JavaScript die Client Seite der Websocket Verbindung implementiert werden.

- URL des Websockets
  die URL für die Verbindung zum Websocket ergibt sich durch den Aufbau des Servers. Wenn der Server über ein SSL Zertifikat verfügt, muss “wss:“ verwendet werden, falls nicht wird “ws:“ verwendet[4]. Darauf folgt dann die IP Adresse vom Server. Der restliche Pfad hängt davon ab, wie der Pfad auf dem Server konfiguriert ist.

- ws.onconnect
  Ähnlich zur @OnOpen Methode auf der Server Seite, können hier Vorbereitungen getroffen werden, die benötigt werden, sobald eine Verbindung besteht.

- ws.onerror
  In dieser Funktion kann auf Fehler, die bei der Benutzung des Websockets entstehen können, reagiert werden.

- ws.onclose
  Falls die Verbindung unterbrochen, beendet oder gar nicht entsteht, kann in dieser Funktion dementsprechen darauf reagiert werden.

- ws.onmessage
  Diese Funktion ist das Gegenstück zur @OnMessage Methode auf der Server Seite.

Sie dient dazu, auf eine Benachrichtung durch den Server zu reagieren. In unserem Fall kommt immer dann eine Benachrichtigung, wenn Alexa eine Frage verarbeitet hat und über den REST Service den Server benachrichtigt hat. Dieser sendet dann eine Meldung an die Website. Im Parameter “message“ steht, welche Animation verlangt wird. Dann wird die Animation, welche abgespielt werden soll, im Videoplayer gesetzt und gespielt.

9.2 Video.js Framework

```
<video id="my-player">
    <source src="LeonidStandUp.mp4" type="video/mp4">
</video>
```

Abbildung 9.3: Videoplayer in HTML einbinden

Anfangs wurden die Animationen als Gifs gespeichert und als solche abgespielt. Da aber, sobald keine gute Internetverbindung vorhanden war, das Abspielen der Animation sehr ruckelig wurde, musste eine Alternative gefunden werden. Die Wahl fiel auf das Video.js Framework, welches einen einfach benutzbaren Videoplayer bietet. Dieser Videoplayer spielt erst dann die Animation ab, wenn sie fertig geladen ist, wodurch ruckelige Wiedergaben nicht mehr auftreten, was in diesem Fall sehr wichtig ist, da dadurch Verzögerungen entstanden, was zur Folge hatte, dass die Synchronisation mit der Alexa Sprachausgabe nicht mehr passte. Jedoch kann es auch mit diesem Framework passieren, dass im Falle einer schlechten Verbindung die Videos länger laden. Wenn dieser Fall eintritt, entsteht trotzdem noch eine asynchrone Wiedergabe, jedoch nicht mehr ruckelig. Die Website wurde durch diesen Wechsel allgemein schneller und nutzt die Internetverbindung deutlich besser. Mit dem “video“-Tag wird eine Player Instanz erstellt, dem man eine ID gibt, damit dieser später in JavaScript ansprechbar ist.

```
var myPlayer = videojs('my-player');
```

Abbildung 9.4: Über diese Variable wird im Script der Videoplayer angesprochen

Durch die myPlayer.src Funktion kann mittels Code das abzuspielende Video geändert werden, anschließend wird durch myPlayer.load das neue Video geladen und zuletzt mit myPlayer.play wiedergegeben.

Kapitel 10

Ablauf

Durch folgendes Diagramm wird der Ablauf bildlich dargestellt:

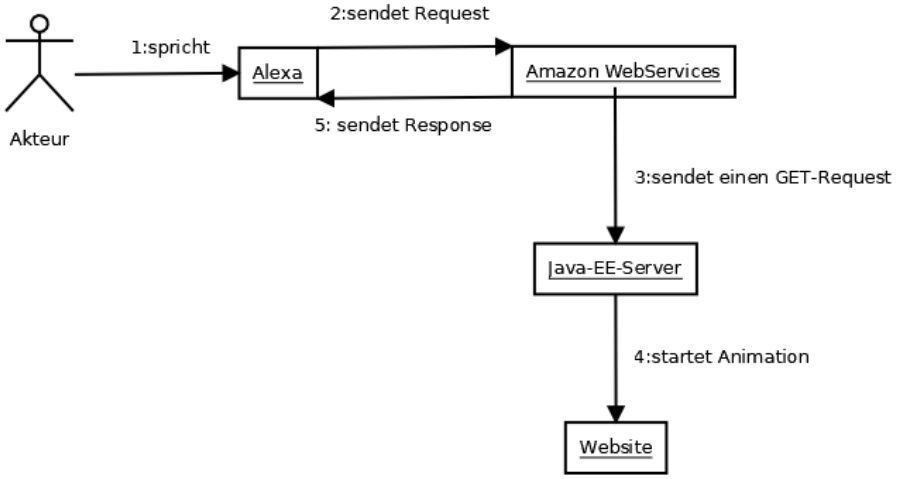


Abbildung 10.1: Kommunikationsdiagramm

- Benutzer erteilt Kommando
- Alexa verarbeitet Kommando und sendet es an AWS
- AWS startet Animation und liefert Antwort zurück
- Alexa antwortet

Kapitel 11

Welche Probleme sind aufgetreten?

Mehrere Probleme sind im Laufe der Diplomarbeit aufgetreten. In diesem Abschnitt wird beschrieben welche Probleme aufgetreten sind und wie sie gelöst wurden.

11.1 Probleme mit Amazon Echo

Bei der Arbeit mit Amazon Echo sind mehrere Probleme aufgetreten. Diese waren, dass verschiedene Wörter nicht richtig verstanden worden sind oder das die Aussprache fehlerhaft war. Es gab 3 verschiedene Varianten:

- Es wurde ein komplett falsches Wort verstanden
- Das richtige Wort wurde verstanden, jedoch waren die einzelnen Buchstaben durch Leerzeichen und Punkte getrennt
- Eine fehlerhafte Aussprache

Wenn das falsche Wort verstanden worden ist, war die einzige Lösung es nochmal zu versuchen, aber diesmal deutlicher zu sprechen.

Wenn das richtige Wort verstanden wurde, aber Punkte/Leerzeichen dazwischen waren, war die Lösung, dass im Code eine Funktion eingebaut wurde, die die Leerzeichen und Punkte entfernt.

Wenn es sich um eine falsche Aussprache gehandelt hat, war das Prüfen der einzelnen Wörter vor der Aussprache die Lösung. Mit Hilfe einer Trennung durch Leerzeichen, wurde das Wort so präperiert, dass Amazon Echo es richtig aussprach.

11.2 Probleme mit dem Raspberry Pi

Auch mit dem Raspberry Pi trat ein Problem auf.

- Der Firefox Browser stürzte ständig ab

Zuerst mussten wir herausfinden, warum der Browser ständig abstürzte. Die Lösung war durch die Hilfe des Internets schnell gefunden. Der Fehler wurde durch das automatische Updaten auf eine neuere Version verursacht. Die simple Lösung war das Deaktivieren der Auto-Updates.

11.3 Animation als Gif

Zu Beginn wurden die Animationen als GIFs abgespeichert. Bei einer stabilen und nicht all zu langsamen Internetverbindung stellte dies kein Problem dar. War dies jedoch nicht gegeben, fingen die Animationen zu ruckeln an, was daran lag, wie GIFs geladen und wiedergegeben werden. Durch das Umstellen auf die Wiedergabe mittels Videos wurde dieses Problem gelöst.

Kapitel 12

Zusammenfassung

Zum Abschluss der Arbeit können wir sagen, dass wir viele verschiedene Technologien entdeckt und gelernt haben. Zum Beispiel das Erstellen von Animationen in Blender oder das Erstellen von Skills in der Amazon Developer Console. Wir haben die Erkenntnis erlangt, dass Alexa doch nicht alles richtig versteht, aber dafür das Erstellen der Skills gut gemacht ist.

12.1 Zukunftspläne

Das Projekt wird an die HTL Leonding übergeben, wo es hoffentlich übernommen und weitergeführt wird. Der Plan ist es, Amazon Echo zu ersetzen. Es soll ein neuronales Netz gebaut werden, so dass eine natürliche Konversation mit dem Avatar möglich gemacht wird. Des Weiteren wird eine schöne Box gebaut, in der die Pyramide steht.

Literaturverzeichnis

[1] Verfasser: Sujeevan Vijayakumaran
Titel: Was ist Git?
https://svij.org/blog/2015/01/05/was-ist-git/
Abgerufen am 12.01.2018

[2] Verfasser: Webmaster
Titel: Eine etwas technischere Betrachtung von Heroku – über Procfiles, Dynos und Slugs
https://www.logicline.de/blog/2015/11/heroku_technical_view/
Abgerufen am 12.01.2018.

[3] Verfasser: unbekannt
Titel: Wildfly
https://de.wikipedia.org/wiki/WildFly
Abgerufen am 26.03.2018

[4] Verfasser: Al1
Titel: Difference between ws and wss?
https://stackoverflow.com/a/46559376
Abgerufen am 28.02.2018

[5] Verfasser: Dr. Volker Zota, Sven Hansen
Titel: Hallo,Echo?
https://www.heise.de/ct/ausgabe/2016-24-Sprachgesteuerte-Lautsprecher-Amazon-Echo-und-Echo-Dot-3459936.html
Abgerufen am 15.01.2018

[6] Verfasser: unbekannt
Titel: Cloud Computing mit Amazon Web Services
https://aws.amazon.com/de/what-is-aws/
Abgerufen am 29.03.2018

[7] Verfasser: unbekannt
Titel: Amazon Echo
https://de.wikipedia.org/wiki/Amazon_Echo
Abgerufen am 20.02.18.

[8] Verfasser: Stephan Augsten
Titel: Was ist der Raspberry Pi?
https://www.dev-insider.de/was-ist-der-raspberry-pi-a-598750/
Abgerufen am 20.02.18.

[9] Verfasser: Sebastian Springer
Titel: Node.js: Das JavaScript-Framework im Überblick
https://t3n.de/magazin/serverseitige-javascript-entwicklung-nodejs-einsatz-231152/
Abgerufen am 22.02.18.

[10] Verfasser: unbekannt
Titel: Node.js
https://nodejs.org/de/
Abgerufen am 22.02.18.

[11] Verfasser: Selim Baykara
Titel: Amazon Alexa: Skills/Apps für Echo - Anleitung und Überblick
http://www.giga.de/audio/amazon-echo/specials/amazon-alexa-skills-fuer-echo-ueberblick-und-anleitung/
Abgerufen am 22.02.18.

[12] Verfasser: Marcel Naujeck
Titel: Alexa Skill: Die ersten Schritte
https://entwickler.de/online/iot/alexa-skill-579756370.html
Abgerufen am 22.02.18.

[13] Verfasser: Margaret Rouse
Titel: Amazon Developer Services
http://searchaws.techtarget.com/definition/Amazon-Developer-Services
Abgerufen am 22.02.18.

[14] Verfasser: Margaret Rouse
Titel: Alexa Skills Kit
http://searchaws.techtarget.com/definition/Alexa-Skills-Kit
Abgerufen am 22.02.18.

[15] Verfasser: zoore
Titel: Texmaker
https://wiki.ubuntuusers.de/Texmaker/
Abgerufen am 26.02.18.

[16] Verfasser: unbekannt
Titel: LaTeX-Kompendium
https://de.wikibooks.org/wiki/LaTeX-Kompendium
Abgerufen am 26.02.18.

[17] Bild von Amazon Echo
https://www.amazon.de/Echo-Plus-Mit-Integriertem-Smart-Home-Hub
-Schwarz-Inklusive-Philips-Hue-Lampe-/dp/B075RLRGDM

[18] Heroku Logo
https://logos-download.com/9855-heroku-logo-download.html

[19] Node.js Logo
https://blog.novatrend.ch/2017/06/19/node-js/

[20] Bild vom Raspberry Pi
https://de.rs-online.com/web/p/entwicklungskits-prozessor-mikrocontroller/8111284/

[21] Git Logo
https://commons.wikimedia.org/wiki/File:Git-logo.svg

Abbildungsverzeichnis

Project Log Book

Datum	Teilnehmer	Bezeichnung
7.7.2017	Robin Mair, Lukas Riedl	Projektantrag eingereicht
9.7.2017	Lukas Riedl	ersten Basisskill erstellt
9.7.2017	Robin Mair	Avatar Prototyp
12.7.2017	Robin Mair, Lukas Riedl	Projekttreffen und zeigen der Ergebnisse
27.9.2017	Robin Mair, Lukas Riedl	Verbindung zwischen Avatar und Amazon Echo hergestellt
23.10.2017	Robin Mair, Lukas Riedl	Bau einer neuen Pyramide
3.12.2017	Robin Mair	Umstellung von GIFs auf Videos
15.12.2017	Lukas Riedl	Erweiterung der Fragemöglichkeiten des Skills
10.1.2018	Lukas Riedl	Daten über WebUntis abrufen
12.1.2018	Robin Mair	Finalisierung des Avatars
26.1.2018	Robin Mair, Lukas Riedl	Vorstellung am Tag der offenen Tür

Kapitel 13

Einteilung der Kapitel

Kapitel	**Autor**
Pflichtenheft	Robin Mair, Lukas Riedl
Erstellung des Avatars	Robin Mair
Animieren des Avatars	Robin Mair
Implementierung eines Alexa Skills	Lukas Riedl
WebUntis-Daten abrufen	Lukas Riedl
Aufbau	Lukas Riedl
Verwendete Technologien/Amazon Echo	Lukas Riedl
Verwendete Technologien/Amazon Developer Services	Lukas Riedl
Verwendete Technologien/Git	Lukas Riedl
Verwendete Technologien/Heroku	Lukas Riedl
Verwendete Technologien/Texmaker	Lukas Riedl
Verwendete Technologien/Wildfly	Robin Mair
Verwendete Technologien/Docker	Robin Mair
Verwendete Technologien/Raspberry Pi	Lukas Riedl
Verwendete Technologien/Node.js	Lukas Riedl
JavaEE Applikation	Robin Mair
Website	Robin Mair
Ablauf	Lukas Riedl
Welche Probleme sind aufgetreten?	Robin Mair, Lukas Riedl
Zusammenfassung	Robin Mair,Lukas Riedl