

# **Medienprojekt I**

## **Thema: Videochop**

Michael Duve, Felix Maulwurf, Angelina Staech

29.11.2014

# Inhaltsverzeichnis

<b>1 Projektteilnehmer</b>	<b>3</b>
1.1 Michael Duve . . . . .	3
1.2 Felix Maulwurf . . . . .	3
1.3 Angelina Staeck . . . . .	3
<b>2 Projektbeschreibung</b>	<b>4</b>
2.1 Idee . . . . .	4
2.2 Zukunft . . . . .	4
2.3 Probleme . . . . .	4
<b>3 Mockup</b>	<b>5</b>
<b>4 Testcases</b>	<b>6</b>
4.1 Canvas . . . . .	6
4.2 Drag-Drop . . . . .	7
4.3 Video Export . . . . .	7
4.4 Format Test . . . . .	7
4.5 Video Controls . . . . .	8
<b>5 Module</b>	<b>9</b>
5.1 Video Item . . . . .	9
5.2 Video Item Loader . . . . .	10
5.3 Video List . . . . .	10
5.4 Video Preview . . . . .	11
5.5 Video Controls . . . . .	11
5.6 Video Timeline . . . . .	11
<b>6 Zeitaufwand</b>	<b>12</b>
<b>7 Quellen</b>	<b>13</b>

# **1 Projektteilnehmer**

## **1.1 Michael Duve**



**Medieninformatik Fachsemester 4**

HTML5, CSS3, JS, PHP, MySQL, Webapps iOS,  
Grundkenntnisse Server, Java, min. Python Photoshop, Illustrator, InDesign  
Repositories (Git, Mercurial), JIRA, Documentation

## **1.2 Felix Maulwurf**



**Medieninformatik Fachsemester 4**

HTML5, CSS3, JS, PHP, MySQL, Java,  
Photoshop, Illustrator, After Effects, 3DSMax,  
Cinema4D, Office, Repositories (Git), LaTeX

## **1.3 Angelina Staech**



**Medieninformatik Fachsemester 4**

HTML5, CSS3, JS, PHP,  
Java, SQL, Illustrator, Photoshop

## **2 Projektbeschreibung**

VideoChop ist eine Webanwendung basierend auf JavaScript, die dem Anwender ein einfaches und intuitives „Videoschnittstudio“ zur Verfügung stellt. VideoChop soll es ermöglichen Videos vom Desktop direkt in den Browser zu ziehen. Es steht dem User frei, ob er ein oder mehrere Videos schneiden oder zusammenfügen möchte. VideoChop wird eine übersichtliche und schlichte Oberfläche bieten, die auch Anfänger nicht überfordert. Die Videos werden in einer Zeitleiste organisiert und beschnitten. Wer mal eben schnell ein Video schneiden will, um Vor- oder Abspann oder etwaige Längen zu entfernen, braucht keine Speicher fressende und komplizierte bedienende Videobearbeitung. Der Nutzer kann sich sein Werk vor dem Abspeichern im Browser als Vorschau ansehen.

### **2.1 Idee**

Die Idee ist es, dass VideoChop dem Nutzer eine Plattform zur Verfügung stellt, die keinerlei Daten serverbasiert speichert. Das Projekt basiert auf JavaScript und nutzt somit die Rechenressourcen des Nutzers. Die Belastung für den Server sind damit minimal. VideoChop soll einfach, intuitiv und schnell sein. JavaScript ist sehr mächtig, allerdings sind zu Projektbeginn die Möglichkeiten und Umsetzbarkeiten noch nicht geklärt.

### **2.2 Zukunft**

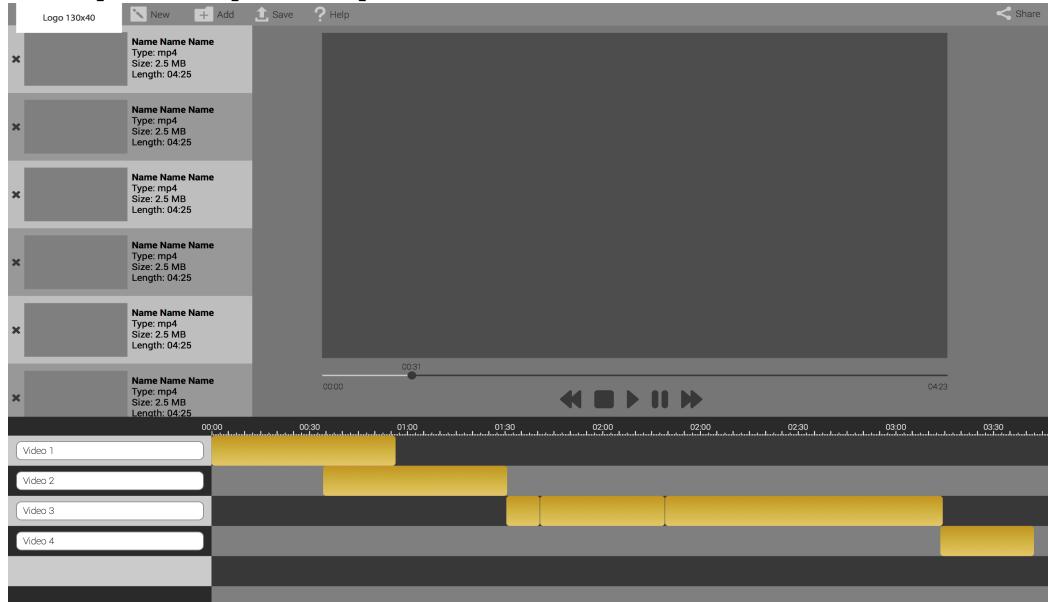
Zukünftig sollen weitere Funktionalitäten hinzukommen.

### **2.3 Probleme**

Das Verbinden mehrerer Videos zu einem mit Hilfe von JavaScript ist sehr problematisch.

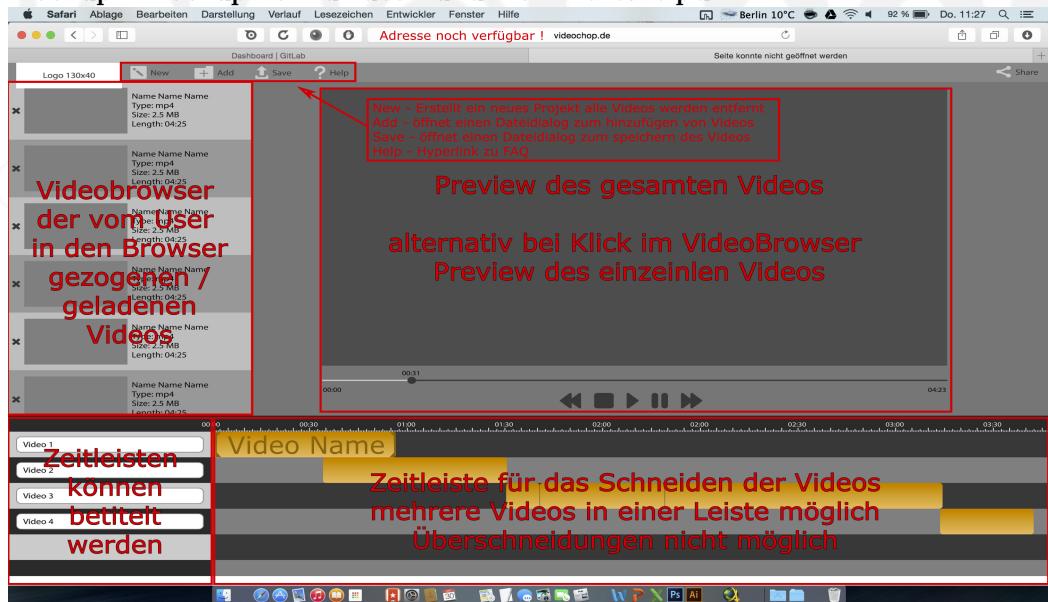
### 3 Mockup

mockup - mockup-vc-MD.pdf



Das Mockup ist nur eine Orientierung für die finale Website.

mockup - mockup-vc-md-Ideen-und-Kommentare.pdf



Es wurde in Gruppenarbeit erstellt.

## 4 Testcases

Für jede Funktion haben wir Testcases erstellt. Die Testcases werden jeweils nach Bedarf erweitert. Das bedeutet, dass im weiteren Verlauf der Entwicklung neue Testcases hinzukommen können. Die Testcases sind spezifisch und nicht modular programmiert. Aus den Testcases haben wir unsere Module abgeleitet.

### 4.1 Canvas

#### testcases - canvas - index.html

Allgemeiner Funktionsstest des Canvas Objekts.

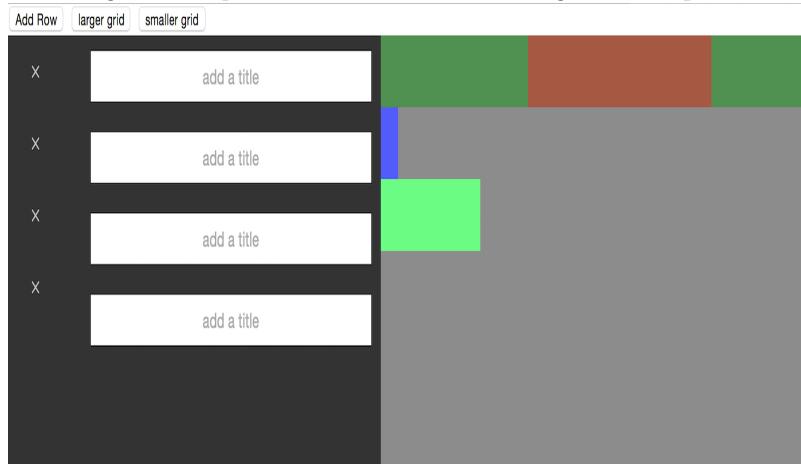


Abilden eines Videos auf einem Canvas Objekt.

## 4.2 Drag-Drop

**testcases - drag-drop - index.html**

Der Drag and Drop Testcase dient als Grundlage für die spätere Timeline.



Aus diesem Testcase wurde das Modul Video Timeline abgeleitet.

## 4.3 Video Export

**testcases - export\_videos - index.html**

## 4.4 Format Test

**testcases - format-test - index.html**

## 4.5 Video Controls

### testcases - video-controls - index.html

Der Testcase Video Controls sollte die später benötigten Funktionen zum kontrollieren des Videos über JavaScript simulieren.

Folgende Funktionen wurden geschrieben und getestet :



- Button
  - Play/Pause
  - Stop
  - Faster / Normal Speed
  - Backward
  - Mute
  - Fullscreen
  - Loop
- Slides
  - Volume
  - Current Time
- Timeupdate
  - Display Current Time
  - Display Duration Time

Das Crossbrowser Testing hat ergeben, dass einige Funktionen noch einmal überarbeitet werden müssen.

Aus diesem Testcase wurde das Modul Video Controls abgeleitet.

## 5 Module

### 5.1 Video Item

testcases - module\_video\_item - index.html

videochop - js - modules - videoItem.js

Jedes vom Nutzer in den Browser gezogene Video wird durch ein VideoItem repräsentiert. Es speichert alle relevanten Daten. Die Daten werden beim in den Browser ziehen vom Modul VideoItemLoader ausgelesen und in ein neues VideoItem geschrieben.

```
function VideoItem(settings) {
    this.settings = {
        video: null,
        name: "TEST",
        length: 100,
        start: 0,
        end: 100,
        size: 5500,
        resolution: {
            width: 3840,
            height: 2160
        },
        thumbnail: null
    };

    // if settings were not set by initializing, fill with default settings
    $.extend(this.settings, settings || {});
}

this.initialize();
```

Das Videoitem bekommt die Eigenschaften Video-URL, Name, Länge, Startpunkt, Endpunkt, Größe, Auflösung und Thumbnail übergeben oder es werden die Default-Settings gesetzt. In der Funktion getMarkUp() werden die HTML-Felder in der index.html dann mit den Werten des VideoItems befüllt, um es in der VideoItemList entsprechend anzuzeigen.

```
getMarkUp: function () {
    return '<li class="file" id="video-item-' + this.id + '">' +
        '<div class="file-delete icon_close"></div>' +
        '' +
        '<div class="file-info">' +
        '<p>Name: <span class="file-name">' + this.settings.name + '</span></p>' +
        '<p>Duration: <span class="file-duration">' + this.timeFormat() + '</span></p>' +
        '<p>Size: <span class="file-size">' + this.sizeFormat() + '</span></p>' +
        '</div></li>';
},
```

Die Größe und die Länge des Videos werden formatiert in den Funktionen timeFormat() und sizeFormat().

## 5.2 Video Item Loader

**testcases - module\_video\_item\_loader - index.html**  
**videochop - js - modules - videoItemLoader.js**

Mit diesem Modul wird nach dem Drag and Drop eines Videos vom Desktop in den Browser hinein ein neues VideoItem erstellt und zurückgegeben.

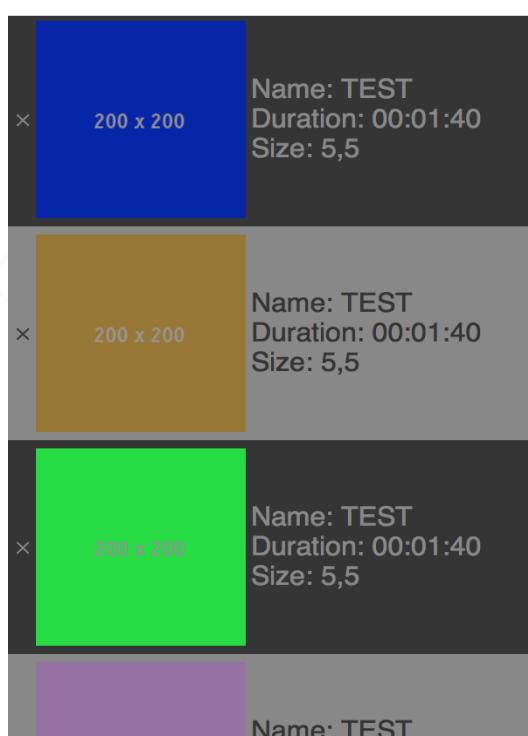
Es gibt die Attribute data, extension, name, prettySize, size und type welche zunächst auf Default-Werte gesetzt sind. Wird ein neuer VideoItemLoader instanziert, muss die Funktion add() aufgerufen werden, welche aus dem übergebenen Video eine Video-URL erzeugt. In der Funktion wird dann die Funktion loadMetaData aufgerufen und bekommt diese URL übergeben. In der Funktion wird ein Video-HTML-Element erzeugt und die URL wird diesem hinzugefügt. Anschließend wird ein neues VideoItem erzeugt und zurückgegeben, welchem die Eigenschaften des aktuellen Videos übergeben werden.

## 5.3 Video List

**testcases - module\_video\_list - index.html**  
**videochop - js - modules - videoList.js**

Das VideoList Modul stellt den Container für die VideoItems zur Verfügung.

click to add a video -> simulates a new drop



## **5.4 Video Preview**

Dieses Modul wurde noch nicht programmiert.

Stellt den Container für die Video Preview zur Verfügung.

## **5.5 Video Controls**

Dieses Modul wurde noch nicht programmiert.

Stellt alle Funktionen zum kontrollieren der Videos zur Verfügung.

## **5.6 Video Timeline**

Dieses Modul wurde noch nicht programmiert.

Das Video Timeline Modul stellt sämtliche Funktionen für die spätere Video Timeline zur Verfügung.



## 6 Zeitaufwand

Aufgabe	Zeitaufwand
Projektkonzept	Zeitaufwand
Einrichten Entwicklungsumgebung	Zeitaufwand
Installation & Konfiguration IDE	Zeitaufwand
HTML-Struktur	Zeitaufwand
JS-Module Template	Zeitaufwand
Coding Conventions	Zeitaufwand
Test-Case: Canvas	Zeitaufwand
Mockup	Zeitaufwand
Test-Case: Drag&Drop	Zeitaufwand
Test-Case: Video-Formate	Zeitaufwand
Test-Case: Video Controls	Zeitaufwand
Wiki: Lizenz prüfen	Zeitaufwand
Logo	Zeitaufwand
Mockup zu Layout	Zeitaufwand
Test-Case: Video Controls Part II	Zeitaufwand
Content: Impressum	Zeitaufwand
HTML-Gerüst bauen	Zeitaufwand
Test-Case: Drag&Drop Part II	Zeitaufwand
Video Controls: Cross Browser Test	Zeitaufwand
Dokumentation: IDE	Zeitaufwand
Test-Case: Drag&Drop Part III	Zeitaufwand
Module: VideoItem	Zeitaufwand
Module: VideoList	Zeitaufwand
Test-Case: Export Videos	Zeitaufwand
Test-Case: Video-Converter	Zeitaufwand
Module: VideoItemLoader	Zeitaufwand
Präsentation des Projektstandes	Zeitaufwand
Projektbeschreibung	Zeitaufwand
HTML-Gerüst: Timeline	Zeitaufwand
Test-Case: Video Part III	Zeitaufwand
Test-Case: Canvas Part II	Zeitaufwand
Module: Video Preview	Zeitaufwand
FFMPEG	Zeitaufwand

## **7 Quellen**

**videoChop**  
slice.compose.online