Projektarbeit:

JavaScript anstelle Java-Applet

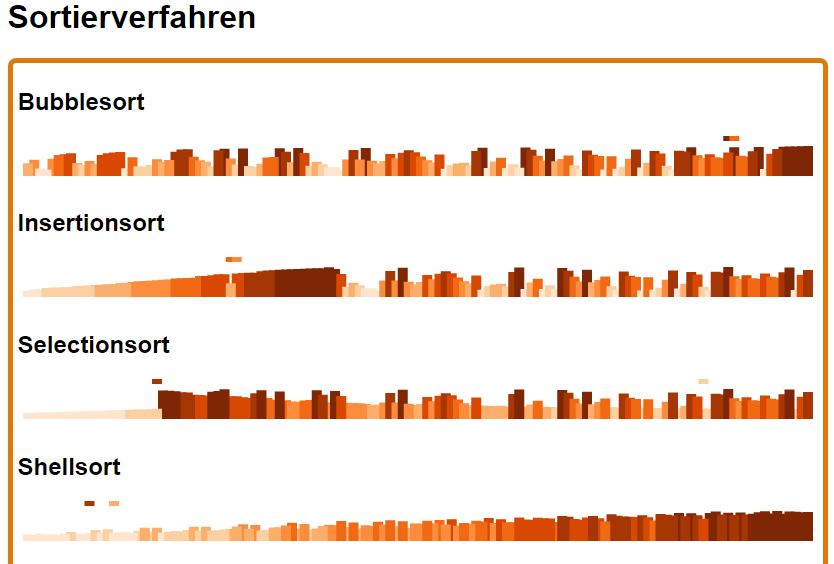
Von:

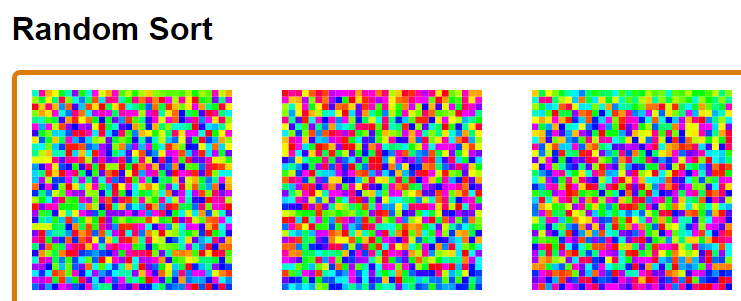
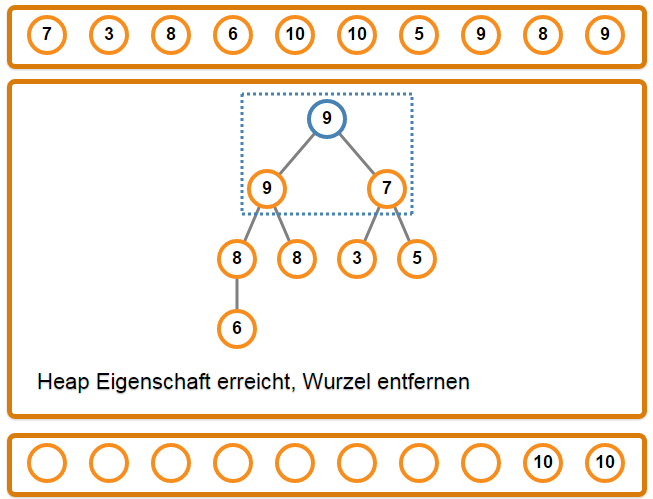
Peter Steensen 550020

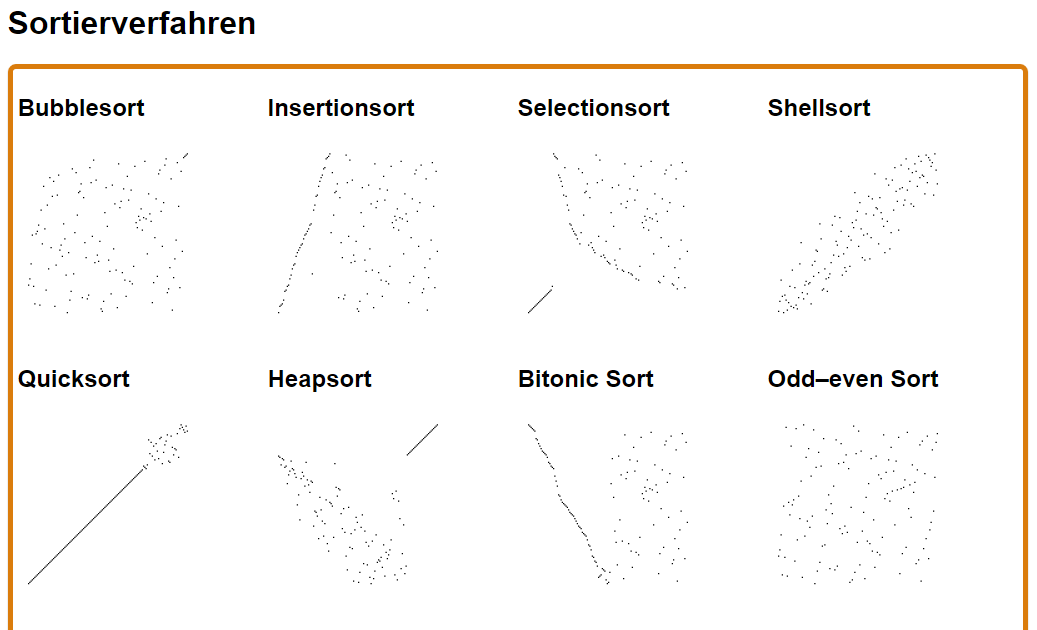
Denny Lüttschwager 343709

Betreuer:

Prof. Dr. Hans Werner Lang









Inhaltsverzeichnis

[Projekt Thema 4](#_Toc431244647)

[JavaScript anstelle Java Applet 4](#_Toc431244648)

[Einleitung 4](#_Toc431244649)

[Erste Überlegungen 5](#_Toc431244650)

[Java Applets 5](#_Toc431244651)

[Fehlermeldungen in den Verschiedenen Browsern 5](#_Toc431244652)

[JavaScript 6](#_Toc431244653)

[Scalable Vector Graphics als Anzeige der entsprechenden Elemente mittels D3.js 6](#_Toc431244654)

[D3.js 6](#_Toc431244655)

[Umsetzung 6](#_Toc431244656)

[Sortierverfahren in JavaScript 6](#_Toc431244657)

[Aufbau der Sortierklassen 7](#_Toc431244658)

[Anzeigeklassen 7](#_Toc431244659)

[Klassische Anzeige 8](#_Toc431244660)

[Neue Anzeige 9](#_Toc431244661)

[Benutzersortierungen 10](#_Toc431244662)

[Heapsort mit Baum Darstellung 10](#_Toc431244663)

[Random Sort Sortierung der Farbigen Bilder 12](#_Toc431244664)

[Usability 13](#_Toc431244665)

[Projektaufteilung und zeitliche Umsetzung 14](#_Toc431244666)

[Hilfsprogramme 14](#_Toc431244667)

[Versionierung mittels Git 14](#_Toc431244668)

[Fazit 14](#_Toc431244669)

[Fazit von Peter Steensen 14](#_Toc431244670)

[Fazit von Denny Lüttschwager 14](#_Toc431244671)

[Nicht umgesetzt 15](#_Toc431244672)

[Automaten 15](#_Toc431244673)

[Erweiterbar 15](#_Toc431244674)

[Anhang 15](#_Toc431244675)

[Abbildung Verzeichnis 15](#_Toc431244676)

[Quellcode 15](#_Toc431244677)

# Projekt Thema

## JavaScript anstelle Java Applet

Auf meinen Webseiten verwende ich eine ganze Reihe von Java Applets, um Abläufe in Algorithmen oder die Funktionsweise von Automaten zu demonstrieren, nach dem Prinzip "learning by doing".

Diese Applets funktionieren oft nicht mehr problemlos. Browser verweigern die Ausführung der Applets, weil die Sicherheitseinstellungen dies nicht zulassen oder angeblich die Java-Version veraltet ist (auch wenn sie gerade ein Jahr alt ist). Hinweise, wie dem Problem abzuhelfen ist, werden nicht gegeben. Benutzer klicken in dieser Situation sofort weg.

Daher möchte ich nach und nach die Java-Applets durch entsprechende JavaScript-Applikationen ersetzen. Ich biete daher als Projekt an, vorhandene Java-Applets in JavaScript zu übersetzen oder auch neue JavaScript-Applikationen zu erstellen.

Dabei könnte auch ein JavaScript-Framework entstehen, dass es ermöglicht, Applikationen dieser Art in einfacher Weise zu erstellen.

Betreuer: Prof. Dr. Lang

## Einleitung

Im Laufe des dritten Semesters wurden mehrere Sortierverfahren in der Veranstaltung Algorithmen behandelt und durch die grafische Visualisierung, der Abläufe gefestigt.

Durch die Visualisierung der Sortierungen wurde den Studierenden der Unterschied, der unterschiedlichen Verfahren, anschaulich verdeutlicht, sodass man die Vorgehensweise und zeitlichen zusammenhänge verinnerlichte.

# Erste Überlegungen

Bevor mit dem Projekt angefangen werden konnte musst zunächst die Überprüfungen ermittelt werden warum Java Applet nicht mehr sinnvoll ist und aus welchen Grund JavaScript genommen werden sollte.

## Java Applets

Ein Java Applet ist eine spezielle Java Anwendung, der es über ein Plugin möglich ist, im Browser zu laufen. Java Applets sind seit der Version ersten Version von Java verfügbar und haben gerade in den Anfangszeiten vom Web viele Nutzer gefunden. Eine Java Anwendung liegt im Java Byte Code vor und deswegen unabhängig vom verwendeten Prozessor und kann überall verwendet werden wo die Java VM verfügbar ist.

### Fehlermeldungen in den Verschiedenen Browsern

Die Visualisierung der aktiven Sortierverfahren der Veranstaltung Algorithmen wurde mittels Java Applets realisiert. In den heutigen Internet-Browser, wie beispielsweise Google Chrome, Firefox oder der Standardinternet-Explorer, werden Java Applets nur mit viel Aufwand ausgeführt, da sie wegen Sicherheitslücken blockiert werden.

|  |  |  |
| --- | --- | --- |
| Google Chrome  Version 45.0.2454.85 m | Firefox  Version 39.0 | Internet Explorer 11  Version 11.0.9600.18015 |
|  |  | Java has to be enabled to view this sorting contest |

Abbildung - Fehlermeldungen in den Verschiedenen Browsern

Aus diesem Grund werden während des Projektes die vorhandenen Java Applets in JavaScript neu geschrieben, sodass der Besucher der Internetseiten, keine Fehlermeldungen aufgrund der Java Applets Blockierung bekommt. Dadurch wird den Besucher erhebliche Einstellungsarbeit abgenommen um seine PC-Sicherheitsregeln für Java Anwendungen anzupassen und somit keiner der einzelnen Internetadresse aufwendig in eine Ausnahmeliste zu pflegen.

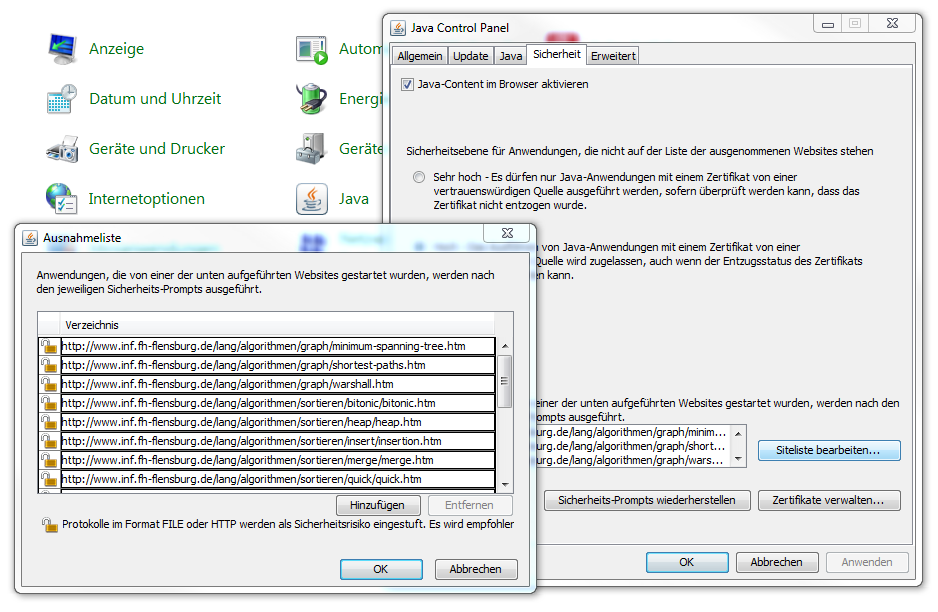


Abbildung - Einstellungen für Java Applets

## JavaScript

Javascript oder auch ECMAScript wie es im Offiziellen Sprachstandard heißt, ist eine clientseitige Scriptsprache im Internet. Jeder moderne Browser verfügt über eine Implementierung die es ermöglicht ohne Plugins oder sonstige externe Programme Websiten dynamisch zu verändern. Durch moderne Techniken wie zb. Just in Time(Jit) – Kompilierung läuft es auch auf schwächeren Prozessoren sehr schnell und kann dadurch für aufwendige Berechnungen in Betracht gezogen werden. Dank der großen Verbreitung von Javascript gibt es eine Vielzahl verschiedener Bibliotheken für so ziemlich jeden Anwendungsfall. Eine dieser Bibliotheken ist D3.js die für Visualisierungen jeder Art verwendet werden kann.

### Scalable Vector Graphics als Anzeige der entsprechenden Elemente mittels D3.js

Im ersten Semester im Fach „Webdesign“ bei Prof. Dr. W. Tepper wurden das erste Mal die Scalable Vector Graphics (SVG) eingeführt. Da das Zeichnen durch das SVG-Element eine aufwendige Arbeit ist, wird in diesem Projekt D3.js als Unterstützung für die Verarbeitung der Daten verwendet. Mittels D3.js werden die entsprechenden Daten in den geschriebenen Funktionsschleifen analysiert und grafisch in SVG-Format angezeigt. Diese komplexen SVG-Zeichnungen im Browserfenster beinhalten je nach Anwendungszweck Linien, Kreise oder Texte, die mit Funktionen wie „OnClick“ oder anderen Funktionen verknüpft werden können.

### D3.js

D3.js ist eine JavaScript Library, die es ermöglicht, Datensätze im Web zu visualisieren. Es können damit sehr leicht SVG-Grafiken erstellt und die DOM-Struktur durch Datensätze manipuliert werden. Dafür nutzt D3.js die aktuellen HTML5, CSS3 und SVG-Webstandards.

D3 steht für Data Driven Document und wurde 2011 das erste Mal veröffentlicht. Die Hauptentwickler dieses Frameworks sind Michael Bostock, Jeffrey Heer und Vadim Ogievetsky.

Michael Bostock stellt auf den Seiten <http://d3js.org/> und <http://bost.ocks.org/mike/> viele Beispielanwendungen mit Projekten und den dazugehörigen Codes vor zur Verfügung.

# 

# Umsetzung

## Sortierverfahren in JavaScript

Die Umsetzung der Sortierverfahren von Java in JavaScript war an einigen Stellen aufwendig. Dazu gehörten zum Beispiel die Code Zeilen:

|  |  |  |
| --- | --- | --- |
| Java | JavaScript | Beschreibung |
| int x=a[lo+(hi-lo)/2]; | var x = Math.floor(a[lo+(hi-lo)/2]); | Javascript rundet nicht ab, da es dynamisch Typisiertist , dh. die Variable ändert seinen Typ, wenn man versucht einen anderen Datentypen hinein zu schreiben. |
| Rekursion |  | Rekursion ist in Javascript möglich aber im Usersort hat es große Probleme bereitet, weshalb auf andere Techniken zurückgegriffen wurde um die Rekursion zu vermeiden. |
|  |  |  |
|  |  |  |

Ansonsten konnten die Sortierablaufe eins zu eins aus den Quellcodebeispielen von der Seite <http://www.iti.fh-flensburg.de/lang/algorithmen/sortieren/>, die durch Prof. Dr. Hans Werner Lang betrieben wird, übernommen werden. Für jedes Sortierverfahren wurde eine Klasse in JavaScript angelegt.

## Aufbau der Sortierklassen

Es existieren 11 Sortierklassen, wovon jede ein anderes Verfahren oder eine andere Vorgehensweise besitzt, dennoch sind sie nach der gleichen Struktur aufgebaut. Sie besitzen private und public Variablen und Funktionen, die von dem verschieden Auswertungsmechanismen genutzt werden. Zu den Auswertungsmechanismen gehören Die Neue und Alte Anzeige Sortierung und die Benutzer Sortierung.

Zu den public Funktionen die vom Auswertungsmechanismen genutzt werden gehören „sortAll(array)“, „updateMode(pMode)“ und „onClick(pSortInfo,pOrginal)“.

Die Funktion „sortAll(array)“ setzt das zu Sortierende Array welches über die Parameter übergeben wird und Startet die Private Funktion die für die jeweilige Klasse unterschiedlich definiert werden. Das Ergebnis aus der Sortierung wird in der private Variable „\_Actions“ welches ein Array ist gespeichert und an die Auswertungsmechanismen zurückgegeben. Die in „\_Actions“ enthaltenen Daten, sind Befehle die vom jeweilgen Sortierverfaren inzugefügt werden und um Auswertungsmechanismen Schritt für Schritt ausgefewertet werden. Die Befehle lauten wie Folgt „swap“, „partition“ und „done“. Der Befehl „swap“ führt einer Vertauschung durch, „partition“ zeigt die Stelle des Array die gerade überprüft wird und „done“ gibt an das das Array Sortiert ist.

Die Funktionen „updateMode(pMode)“ und „onClick(pSortInfo,pOrginal)“ werden zusammen mit den Public Variablen „LeftStart“ und „RightStart“ für die Nutzung der Benutzer Sortierung genutzt. Die Funktion „updateMode(pMode)“ ist in diesem Zusammenhang ausschließlich für die Beschreibung des nächsten Schritts da und gibt einen String zurück abhängig von dem Parameter „pMode“. Die zweite Funktion „onClick(pSortInfo,pOrginal)“ verarbeitet das Klicken auf den entsprechenden Kreisen der jeweiligen Parameter Variablen, in einem größeren „Switch Case“. Durch die Verarbeitung wird das anzuzeigende Array vertauscht und die entsprechend Fokussierten Kreise hervorgehoben.

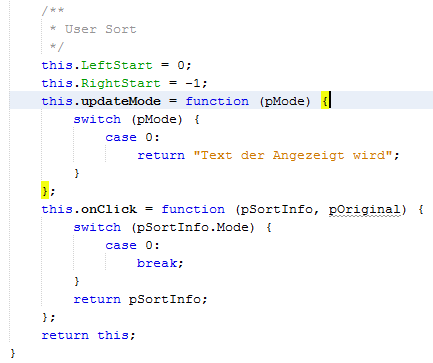
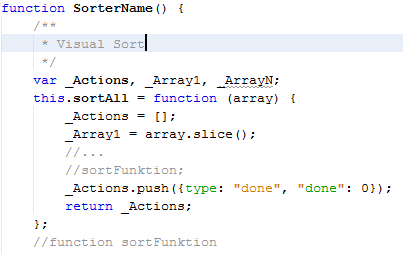


Abbildung - Struktur der Sortierklassen

## Anzeigeklassen

Zum reinen Visualisieren wurden die altbekannte Schwarz-Weiß-Sortierung im Graph und eine neue Sortieranzeige mittels Balken verwendet, die im optimalen Fall aufwärts sortiert werden. Außerdem wurden Usersort Klassen erstellt, die vom Benutzer aktiv genutzt werden kann, um somit die Sortierung zu verstehen. Das Sortierverfahren „RandomSort“ aus der Veranstaltung „Algorithmen“ war der Anreiz für dieses Projekt. Aus diesem Grund wurde auch einen Ansatz der Farbsortierung umgesetzt.

### Klassische Anzeige

Eine klar ersichtliche Verdeutlichung der Sortierverfahren erfolgt durch die zeitgleiche Ansicht der acht unterschiedlichen Sortierverfahren, in dem Fach Algorithmen bei Prof. Dr. Lang im dritten Semester.

Durch die Visualisierung wurden die Zusammenhänge der Array-Durchläufe und Vertauschung der Einträge erst richtig bewusst. So brauchte das Sortierverfahren Bubblesort gegenüber Quicksort deutlich länger. So wurden in den unten stehenden Bilder der unterschied verdeutlich vom unsortierten Array in Grafischer Darstellung und dem fast Sortierten Array einigen Sekunden Später.

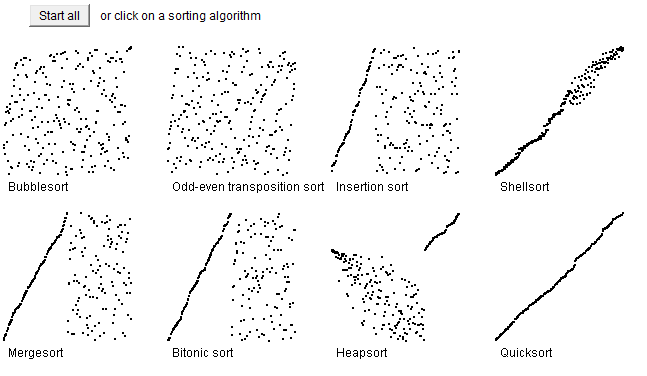
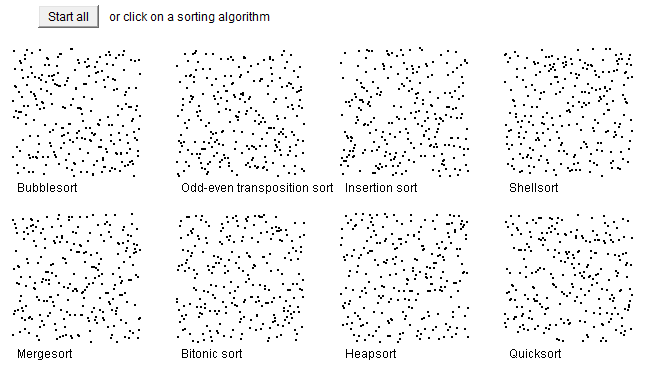


Abbildung - Sortieranzeige mittels Java Applets

Die neue Visualisierung wird wie alle anzeigenden Sortierungen mittels D3.Js realisiert. Für die Verarbeitung und Ausgabe wird die JavaScript Datei „animate-sort-old.js“ genutzt, die Sortierung übernimmt die jeweilige Sortierklasse. Bei der neuen Version der klassischen Sortierverfahrensanzeige werden nun 11 verschiedene Sortierungen angezeigt. Die Schnelligkeit der Verfahren hängt von der Leistung des anzuzeigenden PC, Laptop, Tablett oder Smartphones ab. Ist dafür aber Betriebssystem unabhängig, da JavaScript in jedem Browser zur Laufzeit genutzt wird.

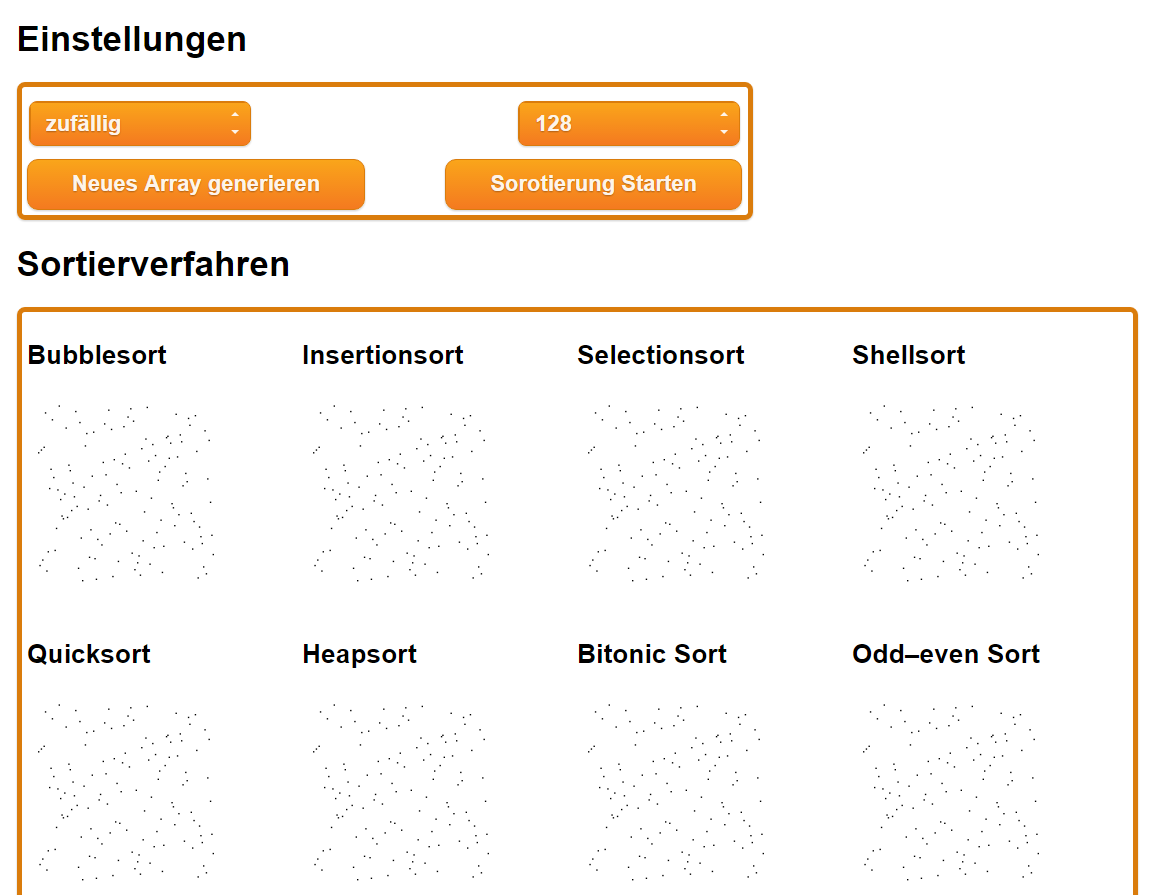


Abbildung - Alter Sortieranzeige mittels D3.js

Der Unterschied zwischen alt und neu liegt in der Nutzung des zu sortierenden Arrays. In der alten Visulisierung wird immer eine Array mit Zufallszahlen generiert werden, welches nur von einem Sortierverfahren genutzt wird. In der neuen Variante können verschiedene Arrays verwendet werden. Das Array kann somit aufsteigend oder absteigend vorsortiert sein oder mit Zufallszahlen in der entsprechenden Größe oder in einem Wertebereich von1 bis 100 genutzt werden. Das Array wird dann von allen genutzt, so zieht man, wie sich welches Sortierverfahren mit dem entsprechenden Array-Typ am besten eignet.

Das so sortierende Array wird als Kopie an das entsprechende Sortierverfahren übergeben und dann mit den entsprechenden Visualisierungsbefehlen an die Ausgabe zurückgegeben. Der definierte Befehl ist „swap“, mit dem das unsortierte, angezeigte Array nun in einzelnen Schritten sortiert werden kann. Es ist zu bedenken, das übergebende Arrays bereits gelöst wurden und das die visuelle Sortierung eine Verzögerung dessen Sortierung ist. Wenn nun erneut auf Sortieren geklickt wird, wird zunächst das Neue Array Sortiert und dann die Befehle der Sortierung auf die Anzeige angewendet.

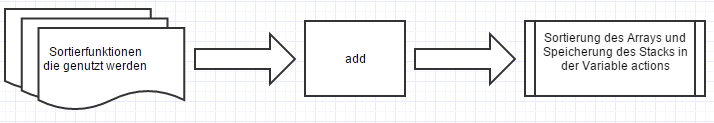


Abbildung - Ablaufdiagramm der add Funktion

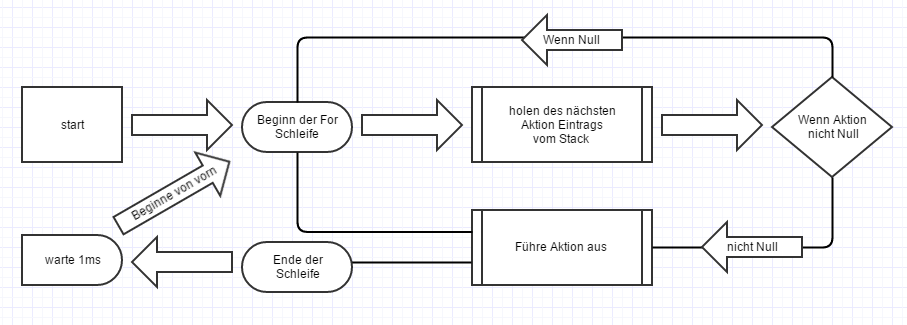


Abbildung - Ablaufdiagramm der Start Funktion

### Neue Anzeige

Die neue Anzeigevariante stellt die gleichen 11 verschiedenen Sortierverfahren dar. Der Unterschied zur klassischen Anzeige liegt zum einem in der Form mittels Balken, die farblich aufsteigend sind und zum anderen mit der Anzeige der zu überprüfenden Position mittels des Befehles „partition“, der grafisch angezeigt, welche Arraystellen mit einander verglichen und getauscht werden. Durch diese neue Art der Sortierungsvariante wird noch einmal die Sortierung des Arrays verdeutlicht. So wird nicht nur der Tausch angezeigt, sondern auch der Grund der Vertauschung. Außerdem Besitz die Anzeige den Befehl „done“ der angibt das Das Array Sortiert ist und keine extra Markierungen mehr angezeigt werden müssen.

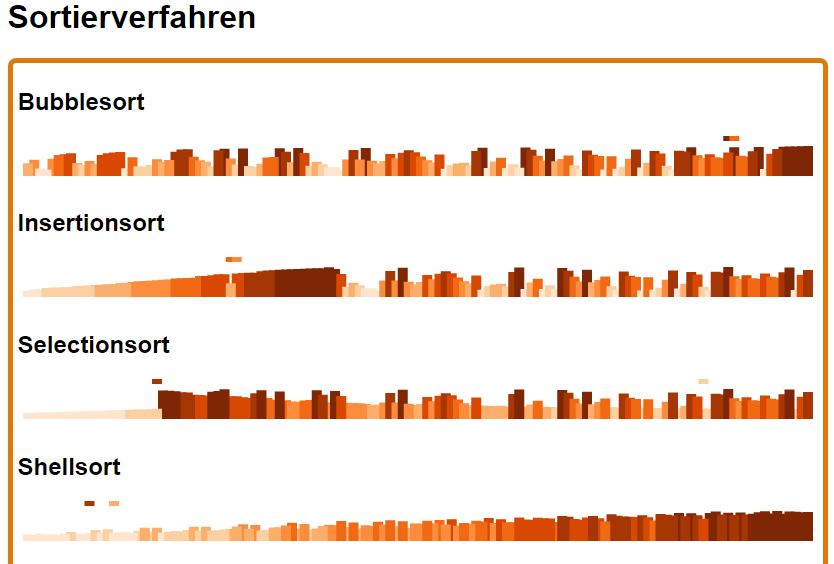


Abbildung - NEUE SORTIERANZEIGE MITTELS D3.JS

## Benutzersortierungen

Zu der reinen Anzeige von Sortierverfahren gibt es auch noch die Sortierung in der der Benutzer der Seite aktiv bei der Sortierung überlegen muss um das unsortierte Array zu sortieren. Dazu gehört zum einen die Sortiervariante Heapsort die zu verstehen ist wie ein umgedrehter Baum und mehrere Sortierverfahren die durch klicken der entsprechenden Kreise, die farblich markiert und beschrieben sind, zu sortieren.



Die Usersort Anzeige besteht im wesentliche aus 3 Teilen. Der erste stellt das zu sortierenden Array dar. Die zu vergleichenden Elemente werden durch eine grüne und rote Umrandung dargestellt. Je nach Algorithmus wird hier ein zweites Array dargestellt wenn es zb. notwendig ist Elemente irgendwo zwischen zu speichern. Der 2. Teil besteht aus der Anweisung die für den User relevant ist. Der letzte Teil besteht nur aus einem Button der dem User die Möglichkeit gibt ein neues Array zu erzeugen. Anhand von MergeSorter wird jetzt hier der generelle Aufbau von einer Usersort Implementierung dargestellt.

In der init-Methode der Klasse werden Hilfs-Variablen gespeichert, die notwendig sind um den aktuellen Zustand des Algorithmus in den onClick Events dar zu stellen. Init wird auch aufgerufen wenn ein neues Array erzeugt wird, von daher ist es wichtig die Variablen hier nochmal korrekt zu initialisieren, da es sonst zu einem fehlerhaften Zustand kommen kann. Mergesort lässt sich für den User erkennbar in 3 Schritte unterteilen. Das wäre einmal das umkopieren des Arrays in ein Hilfsarray. Danach wird dieses unterteilt , worauf anschließend die zwei Hälften sortieren wieder zusammengeführt werden. Der letzte Schritt stellt dann das löschen des Hilfsarray dar wenn das Array fertig sortiert ist oder aber das erneute zurückkopieren in das Hilfsarray wenn das noch nicht der Fall ist. Erkannt wird das an der Rekursionstiefe.

Rekursion wurde hier bewußt vermieden um den Quellcode übersichtlich zu halten. Aufgrund dieser Beschränkung eignet sich die Usersort Implementierung nur für Arraygrößen die einer Zweierpotenz entsprechen. Da man von solch einer Größe ausgeht, kann man die Größe der zwei Hälften immer mit 2 multiplizieren.

Der erste Switch-Case Fall stellt den Anfang dar wo das gesamte Array in das Hilfsarray kopiert wird. Die Variable pSortInfo.ArrayUp[] stellt hierbei das obere und pSortInfo.Array das untere Array dar. Wenn in einem der Array-Variablen eine -1 steht, wird diese vom Framework nicht dargestellt. Zusätzlich wird das Hilfs-Array in 2. Häften unterteilt. Im 2. Switch-Case wurde die Logik des Verfahren implementiert. Die auf logischer Ebene oberen beiden If-Anweisungen Prüfen auf welche Umrandung geklickt wurde, grün oder rot. Danach wird geschaut ob es zu einer Vertauschung kommt, also ob der User auf das richtige Element geklickt hat. Ist dies der Fall wird das aktuelle Element von dem oberen Array in das untere kopiert und die grüne Umrandung entweder um eins nach rechts verschoben wenn noch nicht das Ende des Teilarrays erreicht oder aber gelöscht wenn das Ende des Teilarray erreicht ist. Zusätzliche wird geschaut welche Position der Algorithmus bei den beiden Hälften gerade hat. Sollte hier bei beiden das Ende erreicht werden wird die Größe der zwei Hälften mit 2 Multipliziert. Ist dieser Wert gleich der Länge des zu sortierende Arrays ist der Algorithmus fertig und wechselt zu den letzten Switch-Case. Ist dies nicht der Fall werden die beiden Positionen der Arrayhälften auf 0 gesetzt und es wird zu dem ersten Switch-Case gewechselt.

### Heapsort mit Baum Darstellung

Auf der SortPage03.html wird ein Heapsort Baum dargestellt, der von einem Benutzer oder in drei Phasen vom dem Browser gelöst wird kann.

Die Anzeige ist in vier Bereiche unterteilt: der erste Bereich zeigt eine Ausgabe des zu nutzenden Array, welches unsortiert ist. Im zweiten Bereich befindet sich der Baum, der zunächst erzeugt und dann in mehreren Schritt gelöst wird. Beim Lösen des Baumes wird nach und nach der Stamm entfernt und in den dritten Bereich geschrieben, um hier das gelöste Array auszugeben. Im letzten Bereich befinden sich die Buttons.

Es gibt vier Buttons: der erste ist für die Erzeugung eines neuen Arrays. Damit der Benutzer den Baum von vorne erstellen kann und anschließend zu lösen. Die restlichen drei Buttons dienen als Hilfe, um den Baum durch den Browser automatisch lösen zulassen. Der erste erstellt nur den Baum, der zweite stellte eine Heap-Eigenschaft für den aktuellen Baum her und der dritte löst den Baum von jeder Startphase aus, unabhängig davon ob diese gerade erstellt wurde oder sich mitten in einer Sortierung befindet. Wenn der dritte Button mehrmals hintereinander gedrückt wird führt das nicht zum Fehler, sondern beschleunigt die Lösung des Baums.

Der Benutzer wird während des Lösens des Baumes zusätzlich mit Ausgaben zum Status oder Fehlern, die er gemacht hat, unterstützt.

Um den Baum zu lösen, wird in vier Schritten vorgegangen. Als erstes muss der Baum erstellt werden (Schritt eins). Dafür wird auf die letzte Astgabelung geklickt, sodass an jeder Astgabelung zwei Blätter hängen, bis das gesamte Array im Baum verarbeitet ist. Im zweiten Schritt wird dem erzeugenden Baum die Heap-Eigenschaft gegeben. Dazu werden vom Stamm ausgehend bis ins letzte Blatt die Variablenzahlen absteigend sortiert. Dies geschieht durch die Untersuchung der jeweiligen Astgabelung und deren nächste Blätter. Beim dritten Schritt wird nach Erlangung der Heap-Eigenschaft der Stamm mittels Anklicken in das sortierte Ausgabe-Array geschrieben und das letzte Blatt an die Stelle des Stamms verschoben. Im letzten und vierten Schritt werden die Schritte zwei und drei so oft wiederholt, bis der gesamte Baum abgearbeitet wurde und das sortierte Ausgabe-Array gefüllt ist.

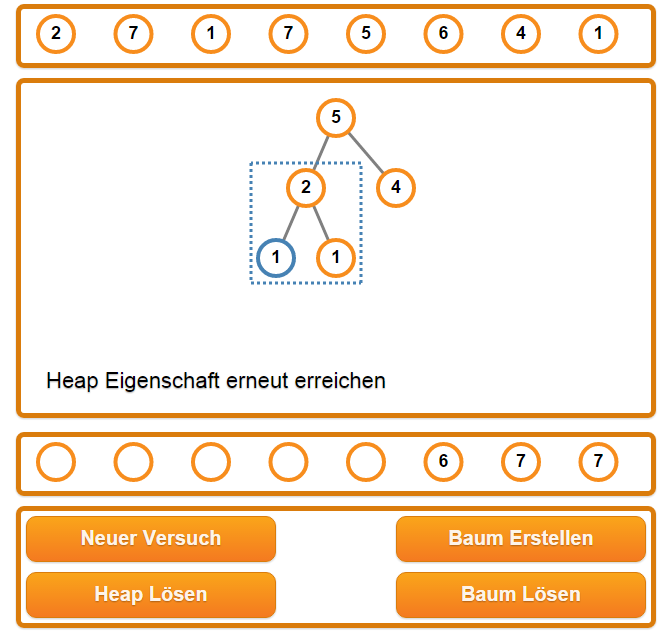
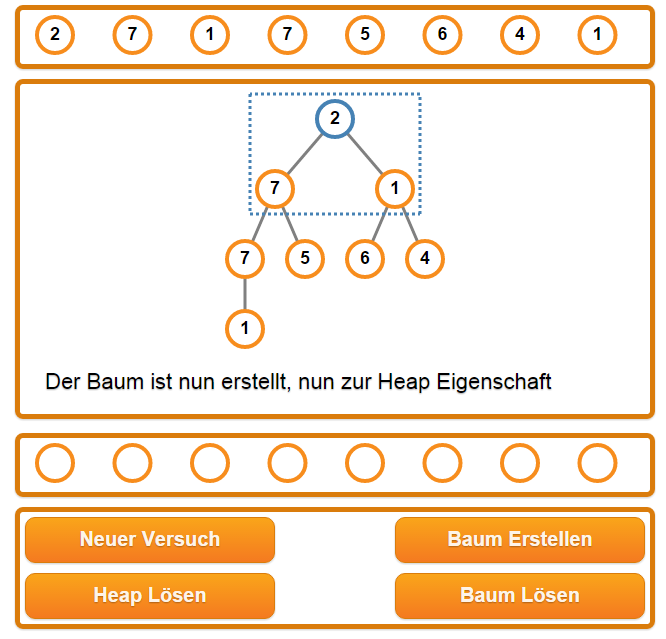
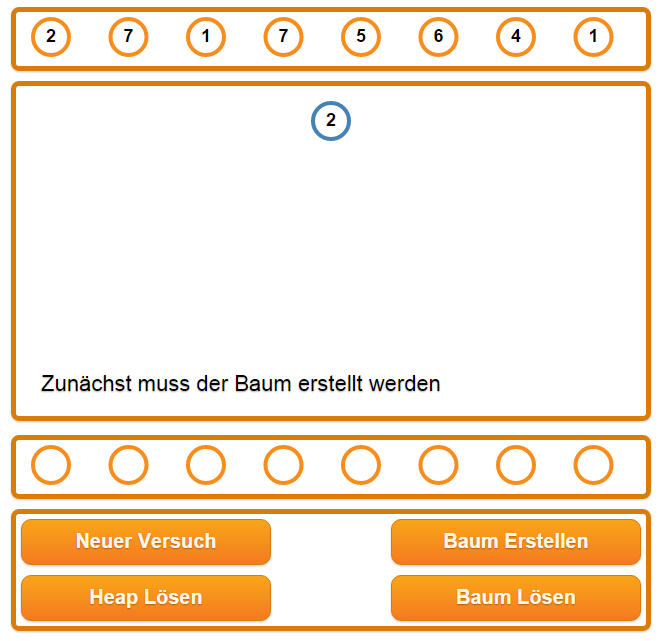


Abbildung - Ablauf der Heapsort Sortierung

Bei der Sortierung des Baum wird ein rekursives Array umgesetzt, welches die gleiche Struktur wie der Baum besitzt. Für die Nutzung des rekursiven Array wurde dieses aus einem Beispielprojekt aus D3.js umgeschrieben. In dem Beispielprojekt wurden mehrere Variablen einen Knotenpunkt zugeordnet, um diese anschließend in einer Baumstruktur auszugeben. Für die Umsetzung einer Heap-Sortierung werden zunächst eine entsprechende Sortierung sowie mehrere Überprüfungen benötigt, die die Anzahl der Blätter und deren Werte an einen Knotenpunkt kontrollieren und die Heap-Eigenschaft feststellen.

Um den Benutzer eine anschauliche Darstellung des Algorithmus zugeben, wurde eine automatisierte Lösungsabfolge programmiert. Diese ist zur besseren Illustration zeitverzögert, durch mehrmaliges Betätigen des Lösungsbuttons lässt sich der Lösungsvorgang allerdings beschleunigen. Dabei sei gesagt, dass der eigentliche Code nicht schneller durchlaufen wird, sondern ein zweiter Algorithmus gestartet wird. Dieser sucht die gleichen Stellen ab, überspringt dabei aber schon gelöste Stellen. Daher geht die Lösung schneller von statten.

## Random Sort Sortierung der Farbigen Bilder

Im letzten und fünften Bereich wurde eine grafische Präsentation von Random Sort umgesetzt. Hier zeigten sich die Grenzen von JavaScript, wenn eine Sortierung mittels animierter Verzögerung vorgenommen wird.

Die Ausgabe besteht aus einem großen Viereck, welches in viele kleine, farbige Vierecke unterteilt ist. Die Anzahl dieser lassen sich auf der Seite einstellen und stellen ein RGB Farbmuster dar, dessen Anordnung zufällig ist. Die Sortierung erfolgt dann wahllos.

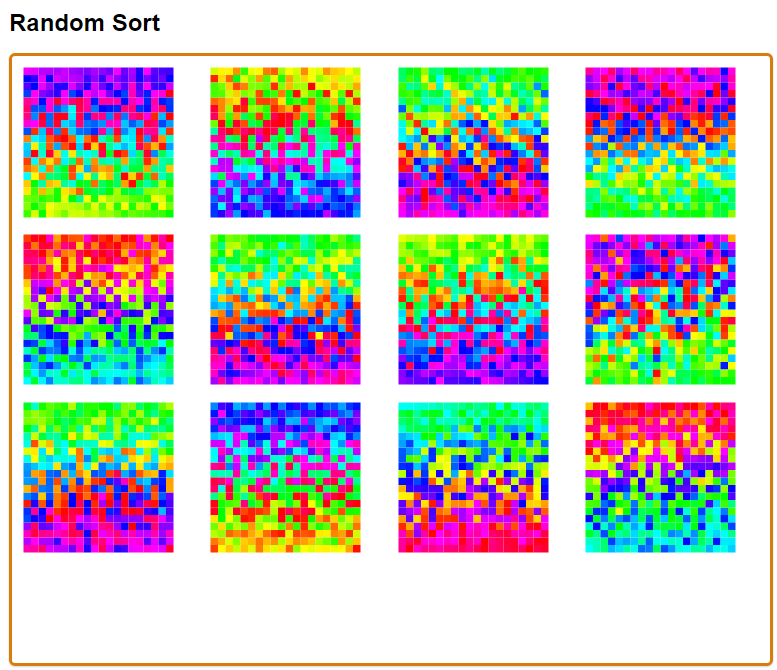


Abbildung – Random Sort

Die Vorgabe bei der Umsetzung von Random sort bildete das Java Applet von der eingangserwähnten Internetseite von Herrn Prof. Dr. Lang. Bei dem Sortierungsverfahren wird ein zufälliges, viereckiges Element herausgenommen und nacheinander werden die Rot-Grün-Blau (RGB) -Farbwerte mit denen der Nachbareinheiten verglichen.

Bei der Umsetzung in diesem Projekt wird in einer Schleife zufällig das Sortierelement ausgesucht, in dem per Zufall die Spalten- und Reihenzahl ermittelt wird und dann eine fest definierte Vergleichssortierung folgt. Die Vergleichsoptionen bestehen aus den roten, grünen und blauen Farbanteilen der Elemente. Der nächste Vergleich ist wie folgt aufgebaut:

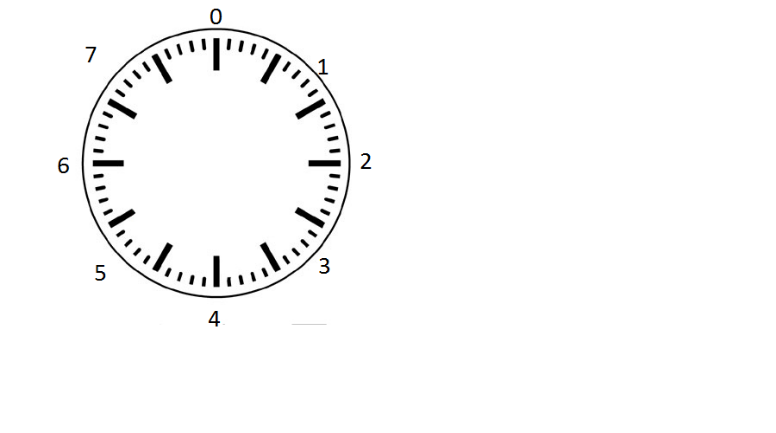
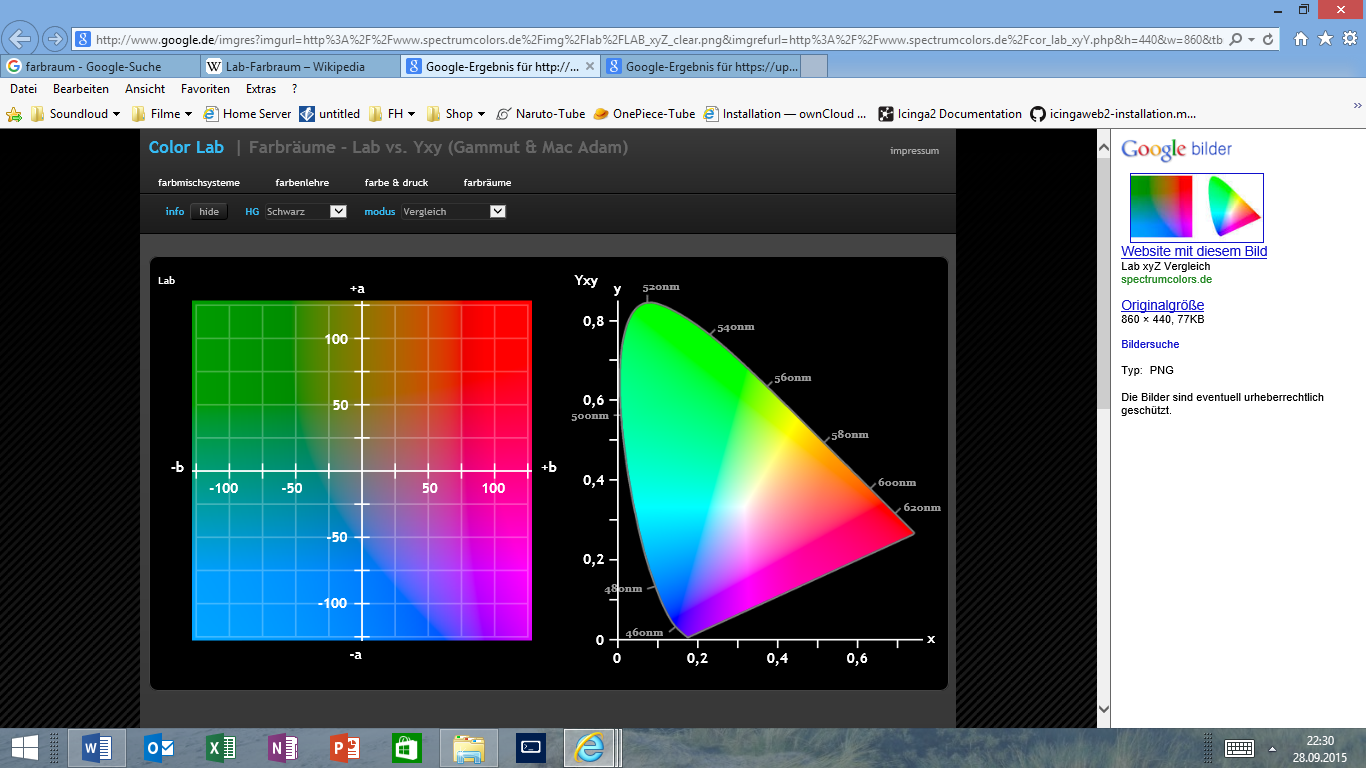


Abbildung 11 - Auswahl der nächsten Position

Da die Performance der JavaScript-Intervalle nicht zufriedenstellend war, wurde der D3.js-Timer verwendet. Durch diesen Timer, der auch gestoppt werden kann, läuft das Sortierverfahren deutlich flüssiger. Für die Erzeugung der Farben, die im Viereck zusehen sind, gab es zunächst den Ansatz mit dem LAB-Farbraum. Der LAB Farbraum stellt alle wahrnehmbaren Farben da und besteht aus der Helligkeit (Luminanz) den Farben für A zwischen Grün und Rot und für B zwischen Blau und Gelb. Da dieses in einem Viereck dargestellt wird, ist einer Unterteilung in kleinere Vierecke optimal möglich. Der Unterschied von LAB- zum RGB-Farbschema liegt in der grafischen Darstellung. Beim LAB-Farbraum beruht auf einen x-y-Koordinatensystem, während der RGB-Farbraum dreidimensional dargestellt wird.



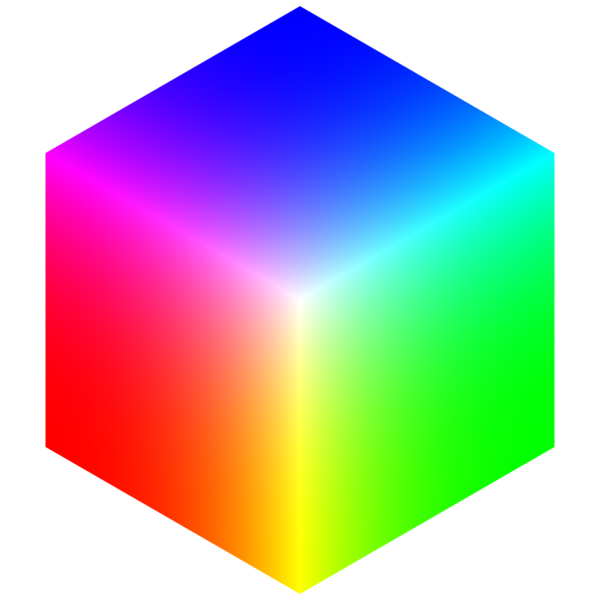


Abbildung 12 - LAB und RGB Fabraum

Der Nachteil dieses Ansatzes mit dem LAB-Farbschemas ist die zusätzliche Konvertierung in dem RGB-Farbraum. Es ist möglich diese Konvertierung durch direkte Verwendung des RGB-Farbraums zu umgehen. Der RGB-Anteil setzt sich aus den x-y-Koordinaten der Positionen der Vierecke zusammen. Die Erzeugung erfolgt mit einer Schleife, die je nach Position den Rot-, Grün- und Blau Wert setzt. Dadurch wird das gesamte Farbschema in seinen Grundfarben erzeugt. Mit dieser Vorgehensweise wird das Array erzeugt, welches anschließend per Zufall umsortiert wird. Daran anschließend erfolgt die Ausführung von Random Sort.

In Abb. (SortPage05.html) werden 12 verschieden erzeugte Vierecke dargestellt. Die entsprechende Sortierung wird in der Datei Mergsort.js definiert. Und en Sortierungsvorgang zu starten stehen den Benutzer zwei Wege zur Verfügung: entweder wird auf einen Startbutton oder auf ein einzelnes Viereck geklickt. Beim Betätigen des Startbuttons wird die Sortierung bei allen Vierecken gestartet. Durch einen Stoppbutton lässt sich die Sortierung aller Vierecke beenden.

# Usability

Um die Bedienung übersichtlich und intuitiv zu gestalten, wurden extra große Buttons sowie Auswahlbereiche, die durch einem entsprechend farblichen Rahmen zusammengehalten werden, geschaffen. Diese Gruppierung soll dem User helfen, die einzelnen Bereiche der Seite besser wahrnehmen zu können. Ebenso ist die Benutzung des Programms auf Smartphones sowie Tablets gewährleistet. Die Größe der Kreise wurde so gewählt, dass der Mauszeiger mindestens zweimal hinpasst (Abb. SortPage03 und SortPage04). So ist auch eine Bedienung mit Touch problemlos möglich (Bilder auf dem Tablett (iPad) Smartphone (iPhone, Android Handy)

In den alten Java Applets werden häufig die Farben in rot und grün gehalten. Diese sind aber für Menschen, die an einer Rot-Grün-Schwäche leiden, schwer erkennbar. Daher wurden die Farben orange und blau eingesetzt. Die orangene Farbe symbolisiert dabei Fachhochschul Farbe Orange und das blau bildet als Komplementärfarbe einen guten Gegenkontrast dazu. Somit sehen die Java Script Anwendungen jung und modern aus.

Um die Webseite modern erscheinen zu lassen, wurden keine Standard Design Buttons oder Dropdown Buttons verwendet, sondern mittels Cascading Style Sheets (css) optimierte Formen mit den entsprechenden Farben.

# Projektaufteilung und zeitliche Umsetzung

Zu Beginn des Projektes wurden zunächst mehrere Testprogramme geschrieben, die mittels D3.js realisiert wurden. Diese Testprogramme entwickelten sich dann im Laufe des Projektes in die nun fertigen Sortier- und Anzeigeklassen weiter.

# Hilfsprogramme

## Versionierung mittels Git

Das gesamte Projekt wurde mittels Git versioniert, sodass daran parallel werden konnte. Dazu wurde bei Github je ein Account angelegt und das Projekt unter dem Namen “AI\_Alogo” erstellt. Die falsche Schreibung wurde dabei extra beabsichtigt, da man als normaler Nutzer keine privaten Projekte anlegen kann, sondern nur öffentliche. Somit haben wir uns für einen Namen entschieden, der sich an Algorithmen orientiert, aber bei der Suche in GitHub nicht direkt als Treffer erscheint.

Mit Github lassen sich Issues anlegen. Issues werden genutzt, um neue Features oder Bug zu dokumentieren, die im Code neu oder verbessert wurden

# Fazit

## Fazit von Peter Steensen

Die Umsetzung der Visualisierung war für mich genau das was mich am Programmieren so stark antreibt. Die Daten Erzeugung ist für mich eher irrelevant, so ist es auch bei mir auf der Arbeit. Zu meinen Aufgaben zählen das Reine Visualisieren und auswerten der gewonnen Daten.

Die Umsetzung mit D3.js hat mich in meinem Lern Level wieder weiter gebracht, alleine das nutzen der so umfassenden Beispiele von D3.js regen zu neuen Projekten an. Zum Beispiel die Karten Nutzung in D3.js ist so vielzeitig verwendbar das man mit ein wenig Aufwand alles damit machen kann, so leicht entsprechende Daten mit anzeigen oder schon bereiste Länder hervorheben.

## Fazit von Denny Lüttschwager

Die Konvertierung der Algorithmen nach Javascript erwieß sich als sehr leicht und war schnell umgesetzt. Auch die grafische Darstellung der Sortierungen mittels D3.js bereitete keine großen Probleme. Das User Sort war durch die Interaktiven Elemente eine größere Herausforderung und bereitete bei manchen Sortierungen große Probleme. Gerade Algorithmen die als Hauptmerkmal Rekursion benutzen ließen sich nur schwer umsetzen da man sich die aktuellen Parameter der Rekursionstiefe merken und Werte für spätere Rekursionen zwischenspeichern mußte.

## Nicht umgesetzt

### Automaten

Ein großer Teil auf der seit inf.fh-flensburg.de/lang befasst sich mit den Automaten und der Auswertung dieser. Zu Beginn war es ein Punkt auf der ToDo Liste, der aber im Laufe des Projektes nicht umgesetzt wurde. Die nicht Umsetzung wurde zusammen mit Herrn Lang abgesprochen um sich mehr auf die Sortierverfahren zu konzentrieren.

## Erweiterbarkeit

Die restlichen Algorithmen sollten keine Elemente mehr beinhalten die Probleme verursachen können und sich deshalb sehr leicht umsetzen lassen . Die Grafische Visualisierung ließe sich vlt. noch Verbessern wenn man eine dritte Farbe hinzufügt. Hier sollte man aber Vorsicht walten lassen da man den User schnell mit zu vielen Farben überfordern kann. Die generelle Struktur der Klassen ist sehr konsistent und lässt sich dadurch sehr gut für zukünftige Sachen erweitern.

# Anhang

## Abbildung Verzeichnis

[Abbildung 1 - Fehlermeldungen in den Verschiedenen Browsern 5](#_Toc431244238)

[Abbildung 2 - Einstellungen für Java Applets 5](#_Toc431244239)

[Abbildung 3 - Struktur der Sortierklassen 7](#_Toc431244240)

[Abbildung 4 - Sortieranzeige mittels Java Applets 8](#_Toc431244241)

[Abbildung 5 - Alter Sortieranzeige mittels D3.js 8](#_Toc431244242)

[Abbildung 6 - Ablaufdiagramm der add Funktion 9](#_Toc431244243)

[Abbildung 7 - Ablaufdiagramm der Start Funktion 9](#_Toc431244244)

[Abbildung 8 - NEUE SORTIERANZEIGE MITTELS D3.JS 10](#_Toc431244245)

[Abbildung 9 - Ablauf der Heapsort Sortierung 11](#_Toc431244246)

[Abbildung 10 - Auswahl der nächsten Position 12](#_Toc431244247)

[Abbildung 11 - LAB und RGB Fabraum 13](#_Toc431244248)

## Quellcode