Projektarbeit:

JavaScript anstelle Java-Applet

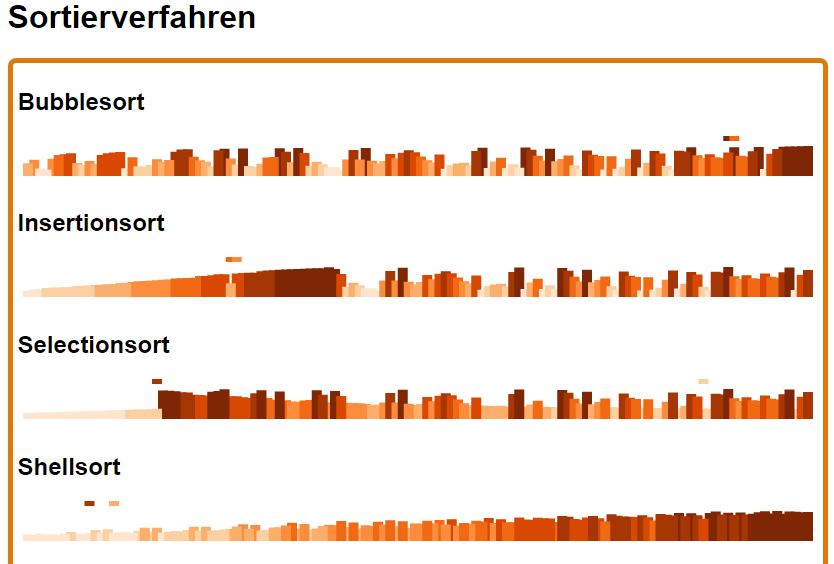
Von:

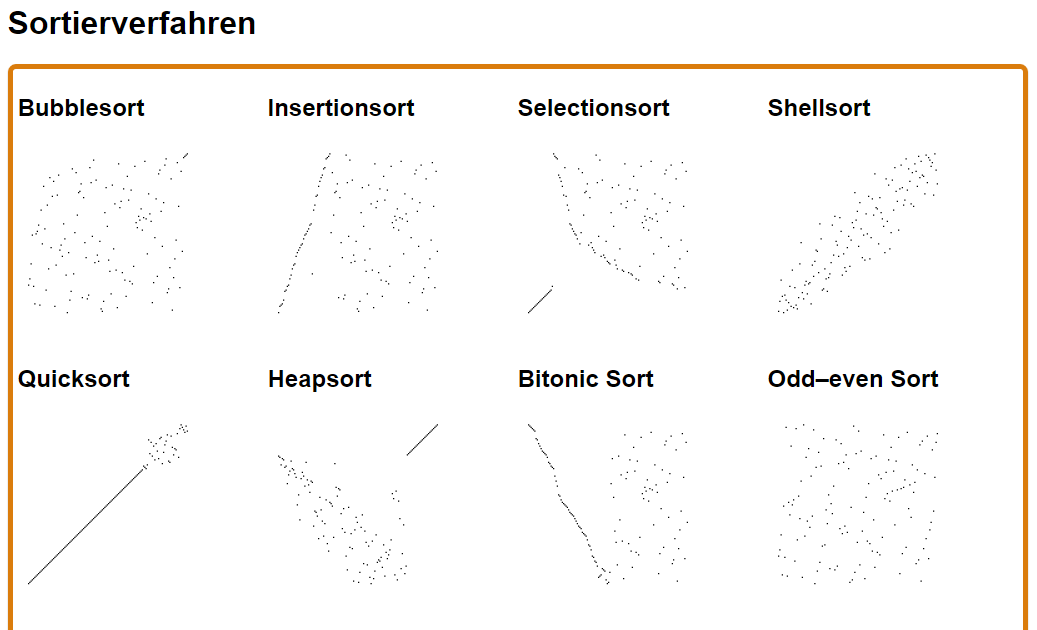
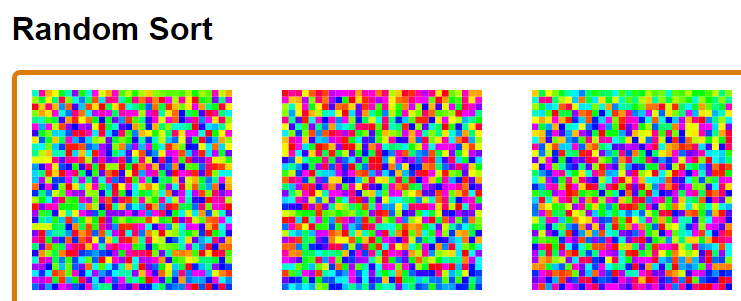
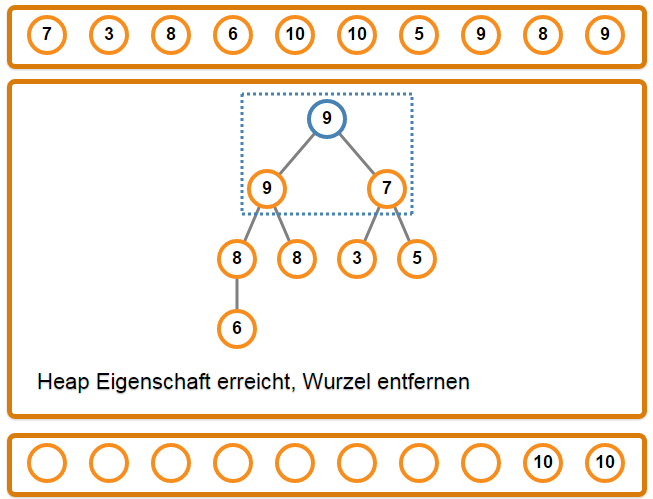
Peter Steensen 550020

Denny Lüttschwager 343709

Betreuer:

Prof. Dr. Hans Werner Lang





Inhalstverzeichnis

[JavaScript anstelle Java-Applet](#h.36muc7fkh0vk)

[Einleitung](#h.7cknzfj9wafi)

[Möglich Umsetzunggen verfahren](#h.dbnym5t9g6uk)

[Eigendes JavaScript](#h.h2x45dhobj1p)

[D3js](#h.h2x45dhobj1p)

[Umsetzung](#h.rphklaehsluc)

[Anzeige Klassen](#h.v0am3j8dxdrn)

[Klassische Anzeige](#h.1jpgbetxc5t0)

[Neue Anzeige](#h.3nvx9p1mnvts)

[Sortier Klassen](#h.3iz7que9bwu3)

[Quick Sort](#h.4z97q919jt1t)

[Insercen Sort](#h.4z97q919jt1t)

[Fazit](#h.2pq6kgx5uddt)

[Fazit von Peter Steensen](#h.pato71s2ut1a)

[Fazit von Denny Lüttschwager](#h.pato71s2ut1a)

### 

### 

### 

# 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: Lang

# Einleitung

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

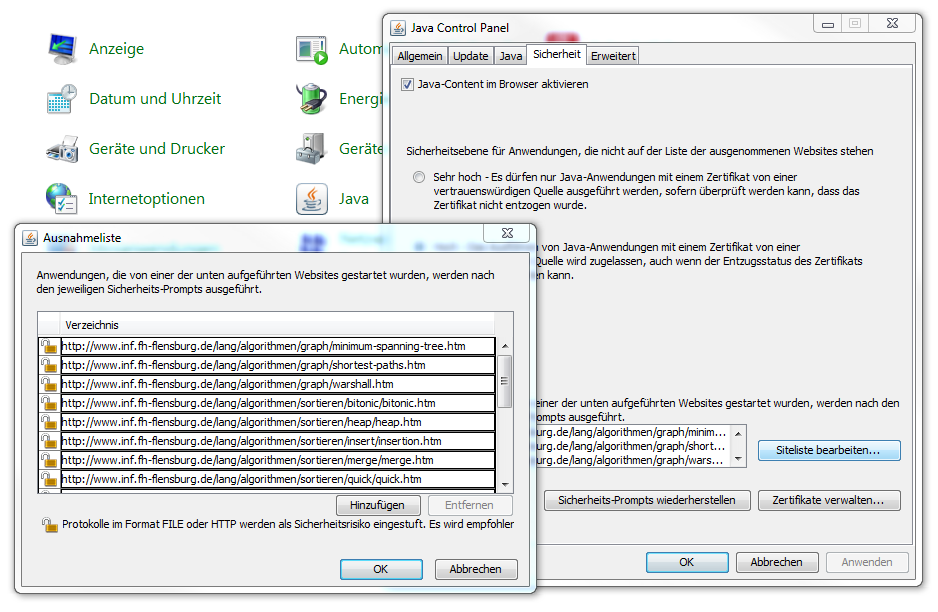
Durch die Visualisierung der Sortierungen der unterschiedlichen Verfahren wurde den Studierenden der Unterschied anschaulich verdeutlicht und dies geschah durch den graphischen und zeitlichen Ablauf.

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 als Sicherheitslücken blockiert werden.

## Fehlermeldungen in den Verschiedenen Browsern

|  |  |  |
| --- | --- | --- |
| 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 |

Aus diesem Grund werden während des Projektes die vorhandenen Java Applets in JavaScript um, so dass der Besucher der Internetseiten, die Fehlermeldungen aufgrund der Java Applets aufzeigen, nicht aufwendig seine PC-Sicherheitsregeln für Java Anwendungen anpassen und auch nicht jede einzelne Internetadresse aufwendig in eine Ausnahmeliste pflegen muss (siehe Bild).



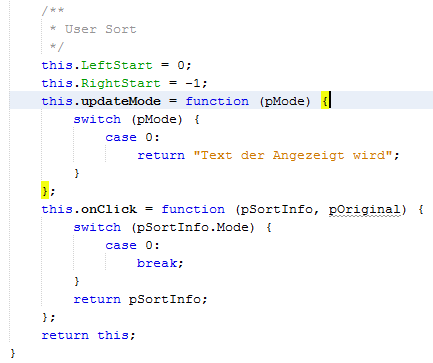
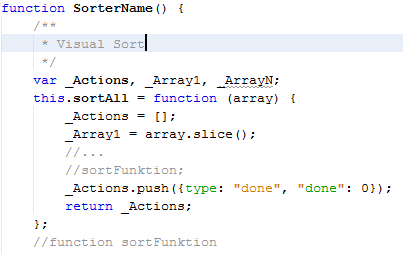
# 

# Sortierverfahren in JavaScript

Die Umsetzung der Sortierverfahren von Java in JavaScript war an einigen Stellen aufwendig. 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 ~~Verfahrens~~ 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.



# 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

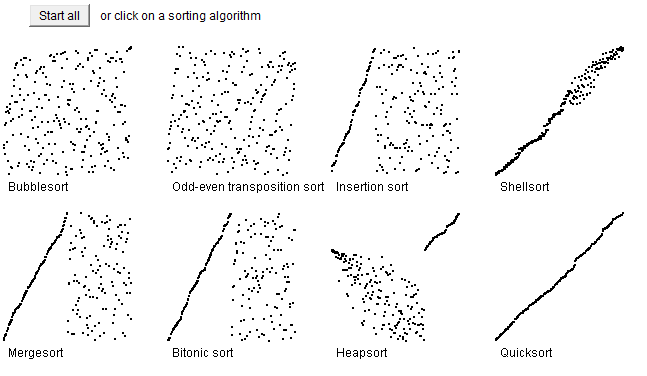
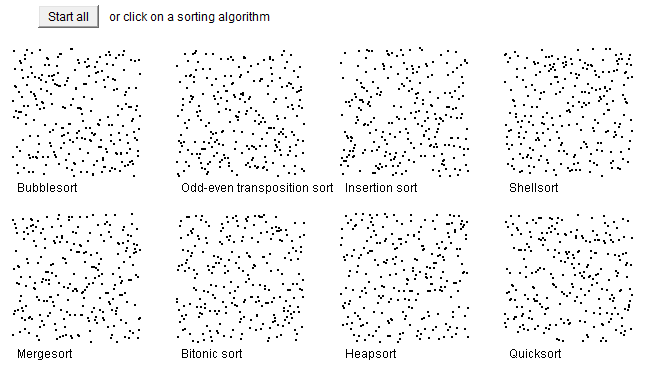
## Anzeigeklassem

Zum reinem Visualisieren wurde die altbekannte Schwarz-Weiß-Sortierung im Graph und eine neue Sortieranzeige mittels Balken verwendet, die im optimalen Fall aufwärts sortiert werden. Außerdem wurden Visualisierungklassen 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 verwendet.

### Klassische Anzeige

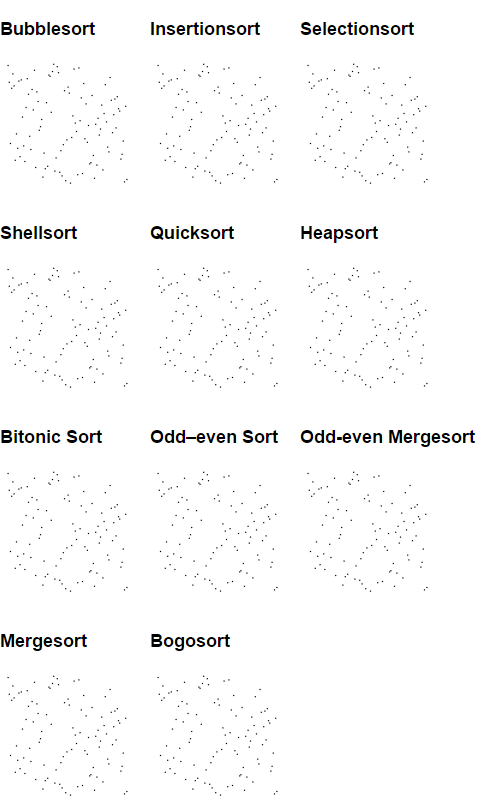
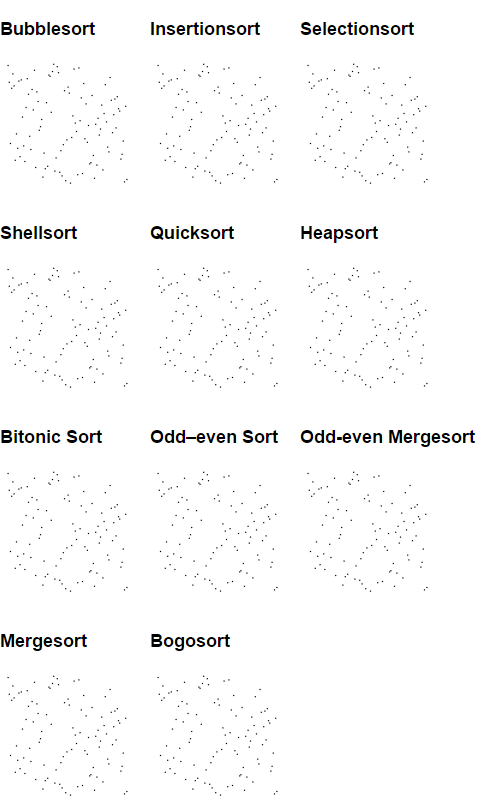
Eine klar ersichtliche Verdeutlichung der Sortierverfahren erfolgt durch die zeitgleiche Ansicht der acht unterschiedlichen Sortierverfahren.

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.

[[1]](#footnote-1)

Die neue Visualisierung wird wie alle anzeigenden Sortierungen mittels D3.Js realisiert.

Für die Verarbeitung und Ausgabe wird die JavaScript Datei XXX 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.



Der Unterschied zwischen alt und neu liegt in der Nutzung des zu sortierenden Arrays. In der alten Visiulisierung wird immer eine Array mit Zufallszahlen gefühlt, 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 bis XXX 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. Die definierten Befehle sind xxx und xxx, mit denen 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.

Bild mittels des Ablaufes der Vorgehensweise des Ablaufdiagramms der Klassischen Sortierung

### 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 des XX Pilote XX, 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 Tauschung.

Um diese Art der Visualisierung zu gewährleisten, benötigt es einen weiteren PiloXX-Befehl und einen abschließenden „ENDE“-Befehl.

**Benutzersortierung eins**

Heapsort mit Baum Darstellung (SortPage03)

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

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 beginnt sich der Baum vor vorne zu erstellen und zu lösen. Die restlichen drei Buttons dienen als Hilfe, um den Baum durch den Browser automatisch lösen zulassen. Der erste der drei restlichen Button 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.

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 das Auswählen der jeweiligen Astgabelung und deren nächste Blätter (siehe Bild). 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.

Im letzten und vierten Schritt werden die Schritte zwei uns drei so oft wiederholt, bis der gesamte Baum abgearbeitet wurde und das sortierte Ausgabe-Array gefüllt ist.

**Die Umsetzung des JavaScript Codes**

Die grafische Sortierung wurde mittels D3.js und einem rekursiven 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. Im 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.

Zu Beginn des Projektes wurde zunächst nur ein Baum erstellt und sortiert. Nach dem wir den ersten Schritt erledigt hatten bauten wir mehrere Fehlerüberprüfungen ein um aus dessen Fehlerkollisionen die entsprechenden Infoausgaben anzuzeigen. Für eine leichte Bedienbarkeit wurden die ausgebenden Sortierergebnisse mit Kreise umrandet, die Farben angepasst und entsprechend große Buttons eingebaut.

Um den User 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.

Im Code sind sieben Abschnitte vorhanden, in denen ausschließlich Funktionen definiert sind, die für die Ausgabe verwendet werde. Diese sind für das statische und animierte Zeichnen sowie für verändernde Zeichnungen zuständig.

Random Sort Farbige Bilder (SortPage05)

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.

Aufbau des Bereichs

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

**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:

Bild (UHR 1 2 3 4 5 6)

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 Hue-saturation-value (HSV)-Farbbaum. Da dieses in einem Viereck dargestellt wird, ist einer Unterteilung in kleinere Vierecke optimal möglich. Der Unterschied von HSV- zum RGB-Farbschema liegt in der grafischen Darstellung. Beim HSV-Farbraum beruht auf einen x-y-Koordinatensystem, während der RGB-Farbraum dreidimensional dargestellt wird.

Bild HSV und RGB

Der Nachteil dieses Ansatzes mit dem HSV-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 Blauwert 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.

Für die Erzeugung der Farben wird die D3.js Color-Klasse genutzt, da diese eine Vielzahl an Möglichkeiten bietet, wie beispielsweise die Nutzung und Konvertierung von verschiedenen Farbschemas.

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 Stopbutton lässt sich die Sortierung aller Vierecke beenden.

Zusätzlich besteht die Möglichkeit die Anzahl der kleinen Vierecke in einem großen Viereck durch eine Auswahlliste zu bestimmen.

**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 Visualiseren 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

ghfhgcff

## Nicht umgesetzt

### Automaten

Ein großer teil auf der seit inf.fh-flensburg.de/lang befasst sich mit den Automaten und der auswertug 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.

## Erweiterbar

Labyrinth

1. http://www.inf.fh-flensburg.de/lang/algorithmen/sortieren/sortcontest/sortcontest.htm [↑](#footnote-ref-1)