Hochschule Stralsund
University of Applied Sciences

Johannes Wehden
Matrikel-Nr. 16085
WS 18/19
5. Semester
Wirtschaftsinformatik

Vorlesung: Creative Programming
Dozent: Prof. Dr. rer. nat. Thomas Wengerek

Hausarbeit

p5js-Projekt „Bubble-Painting“

Einleitung

Ziel dieser Hausarbeit ist es, ein eigenständiges JavaScript-Projekt mit p5js umzusetzen, welches die folgenden Kriterien erfüllt:

1. Das Lesen und Extrahieren externer Daten bzw. Quellen (Datei, URL)
2. Animationen bzw. Motion mit Hilfe der Draw-Funktion erzeugen
3. Interaktion und Navigation durch Maus- bzw. Tasten-Events
4. Interaktive Kontrolle des dargestellten Detailgrads (z.B. Skalierung)
5. Verwendung von Koordinationstransformationen, benutzerdefinierter Shapes oder Easing-Effekten (bei interaktiver Manipulation)
6. Beschriftung
7. Interaktive Informationen (z.B. zusätzliche Informationen bei Mouseover)
8. Verwendung von Farben zur Informationskodierung
9. Codestrukturierung (Funktionen, auch als Klassen bzw. Objekte)
10. Ausführliche Kommentierung des Quellcodes

Dabei ist die kreative Designentscheidung nicht beschränkt, sofern bestimmte Entscheidungen und Vorgehensweisen begründet werden können.

Vorstellung der Projekt-Idee

Das Projekt „Bubble-Painting“ ist ein vollständig funktionierendes, jedoch simpel gehaltenes Spiel, bei dem der Nutzer so schnell wie möglich zufällig positionierte und animierte Kreise durch Anklicken einfärben muss. Dabei wird die zum Durchspielen aller Level benötigte Zeit gestoppt, wodurch ein Highscore-Vergleich möglich ist.

Die Idee zu diesem Projekt ergab sich, als ich an meinem ursprünglichen Plan, einem Sternzeichen-Rechner mit Horoskop-Ausgabe, arbeitete. Da ich mir jedoch nicht sicher war, ob ich dabei alle geforderten Kriterien erfüllen könnte, entschied ich mich für einen Neuanfang.

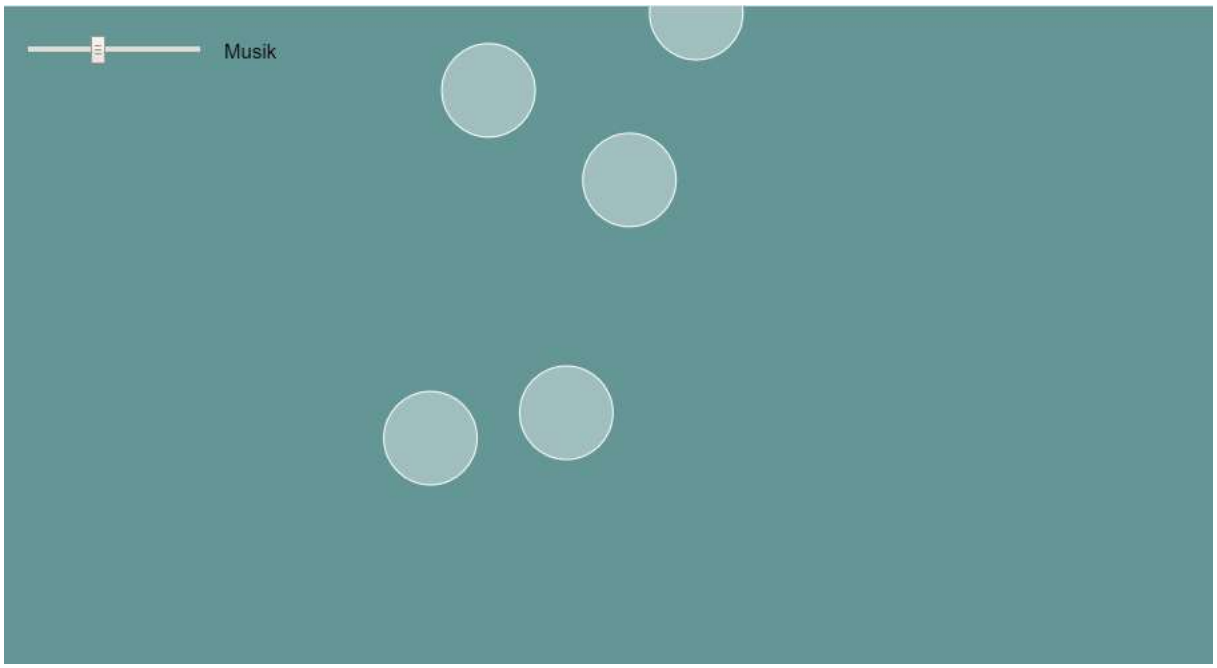
Ablauf von „Bubble-Painting“

Wenn man die index.html bei lokal laufendem Server (z.B. atom-live-server) aufruft, gelangt man direkt zum Startbildschirm. Hier befindet sich ein relativ selbsterklärendes Menü mit den Optionen durch das Drücken der Enter-Taste ([ENTER]) das Spiel zu starten oder mittels der I-Taste ([i]) weitere Informationen anzuzeigen. Zudem wird, wie auch dauerhaft im weiteren Verlauf, ein Regler zur Einstellung der Musiklautstärke angezeigt. Mit der Initialisierung des Spiels wird eine Musikdatei („song.mp3“) geladen und in Dauerschleife abgespielt.

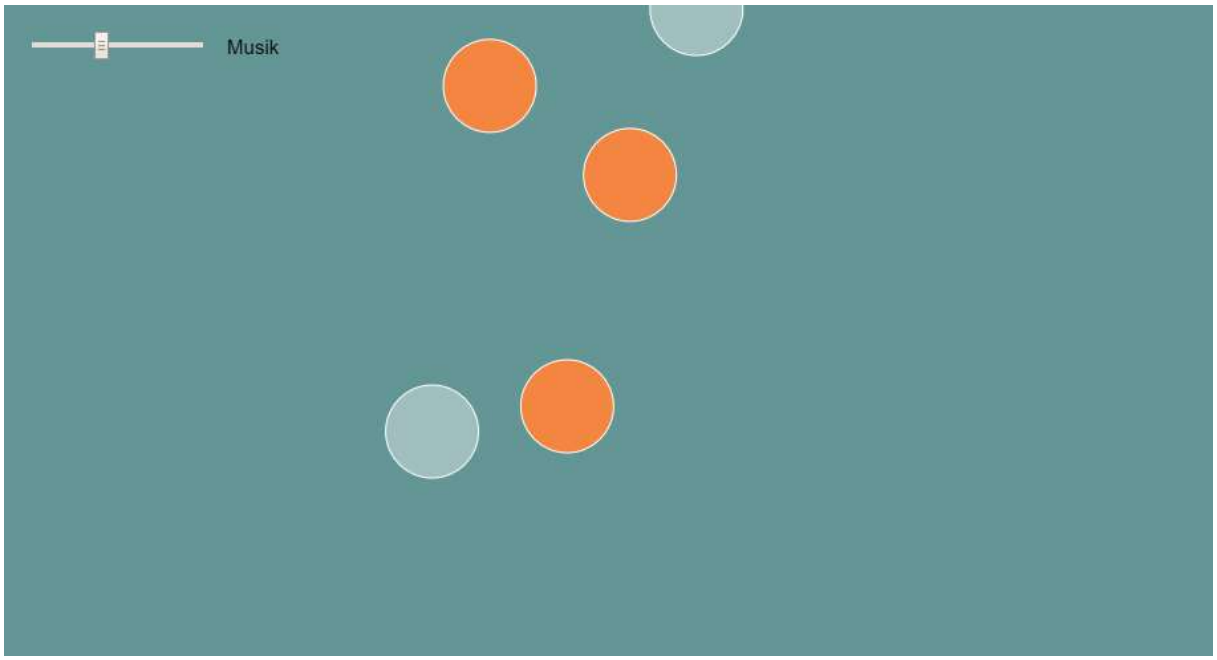
Durch das Drücken der Taste „i“ werden weitere Informationen ein- bzw. ausgeblendet, denen man eine kurze Spielanleitung und Levelanzahl, sowie eine weitere Tastendruck-Option entnehmen kann. Wenn man nämlich zu irgendeinem Zeitpunkt die Escape-Taste ([ESC]) drück, wird das Spiel neu geladen.

Wenn man nun „Enter“ drückt, startet das Spiel und das erste Level wird geladen. Das besondere hierbei ist, dass die Level in einer externen Datei (level.txt) definiert sind, was bedeutet, dass sie jederzeit modifiziert werden können, sofern man das richtige Schema beachtet. In jedem Level wird eine bestimmte Anzahl von Kreisen („Bubbles“) erzeugt, zufällig positioniert und nach definiertem Schema animiert.

Zugleich mit dem Spielstart wird zudem eine Stoppuhr gestartet, damit am Ende die benötigte Spielzeit ausgewertet werden kann.



Durch das Klicken auf eine Bubble wird diese dauerhaft eingefärbt.



Ziel ist es, in jedem Level alle Bubbles einzufärben. Sobald alle Bubbles eines Levels angeklickt wurden, ist das Level geschafft und das nächste beginnt. Sobald das letzte Level abgeschlossen ist, endet das Spiel, die Stoppuhr wird angehalten und der Endbildschirm erscheint.

Hier kann man nun sehen, wie lange man zum Durchspielen gebraucht hat und wie viele Bubbles dabei eingefärbt wurden. Außerdem wird die Spieldauer mit dem vorherigen Highscore verglichen und dieser gegebenenfalls aktualisiert. Da es in JavaScript nicht ohne weiteres nicht möglich ist, lokale Dateien zu überschreiben, ist eine langfristige Highscore-Speicherung nicht eingebaut, was zur Folge hat, dass die erzielte Bestzeit mit jedem Neuladen der Seite gelöscht wird.

Da man aber mit Hilfe der Escape-Taste ([ESC]) das Spiel neustarten kann, ohne die Seite neu zu laden, ist zumindest ein Session-Highscore-Vergleich möglich.

Sobald man einen Highscore erstellt hat, wird im Infobereich des Startmenüs zudem eine Aufforderung zum Schlagen der eigenen Bestzeit angezeigt.

Durch Änderungen in der „level.txt“ Datei (Zeilen hinzufügen, löschen und verändern) kann man sich Selbst vor immer neue Herausforderungen stellen.

Umsetzung der Kriterien

Die nummerierten Unterpunkte dieses Abschnitts beziehen sich jeweils auf die gleich nummerierten Kriterien in der Einleitung.

1. (externe Daten)

Wie bereits erwähnt, verwendet das Projekt zwei externe Daten, welche sich beide im Ordner „data“ befinden. Zum einen ein Lied namens „song.mp3“, welches dauerhaft abgespielt wird und zum anderen eine Textdatei („level.txt“), in der die Spiellevel kodiert sind. Beide Dateien werden im *preLoad* eingelesen, damit sie beim Ausführen der *setup*-Funktion bereits verwendet werden können.

2. (Animationen)

Die *draw*-Funktion ist in diesem Projekt wie ein „gameloop“. In der *draw*-Funktion wird *runLevel* aufgerufen, welche wiederum positionsverändernde Methoden und die *display*-Funktion der einzelnen Bubbles aufruft, was dann eine ständige Neuzeichnung der Bubbles und somit einen Animationseffekt bewirkt. Dank der *draw*-Funktion laufen die Animationen also überhaupt erst.

3. (Interaktion)

Im Startmenü bewirkt die Enter-Taste den Spielstart und „i“ togglet Informationen. Die Taste Escape lässt das Spiel zudem jederzeit neustarten. Der dauerhaft angezeigt Slider verändert die Musiklautstärke und das Klicken mit der Maus auf Bubbles ist sogar das Kernprinzip des Spiels.

4. (interaktive Kontrolle)

Der Slider kann die Lautstärke der Musik kontrollieren.

5. (Shapes)

Alle Bubbles in diesem Spiel sind Shapes. Sie werden in bubbles.js in *this.display* als Kreis (Ellipse) erzeugt.

6. (Beschriftung)

Der ganze Start- und Endbildschirm besteht aus Text. Das Slider-Label „Musik“ ist sogar dauerhaft eingeblendet.

7. (interaktive Informationen)

Im Startmenü des Spiels kann man mittels der Taste „i“ weitere Informationen anzeigen bzw. ausblenden (togglen). Die Levelanzahl ergibt sich hierbei aus der Zeilenanzahl der „level.txt“-Datei.

8. (farbige Informationskodierung)

Das einfärben von Bubbles kann wohl als farbige Informationskodierung verstanden werden, da dem User so mitgeteilt wird, welche Bubbles bereits angeklickt wurden. (siehe *this.clicked* in bubbles.js)

9. (Codestrukturierung)

Der Quellcode ist in mehrere Funktionen unterteilt. Damit die *draw*-Funktion beispielsweise nicht unübersichtlich wird, habe ich „Unterfunktionen“ wie *startScreen*, *endScreen* und *controllMusic* eingebaut. Da diese in der *draw*-Methode aufgerufen werden, werden sie ebenfalls kontinuierlich ausgeführt. Die Methode *Bubble*(x, y, size) fungiert zudem als Klasse, von der (in der Funktion *nextLevel*) Instanzen erzeugt werden. Diese Instanzen sind die Bubbles.

10. (Quellcode-Kommentare)

Der ganze Quellcode ist mit Kommentaren versehen. Zudem sollten sprechende Variablen- und Funktionsnamen das Verständnis erleichtern.

Levelanpassung

Wenn man das Spiel durch Leveländerungen anpassen möchte, muss man nur den Inhalt der „level.txt“ Datei anpassen. Hierbei muss jedoch stets das vorgegebene Schema beibehalten werden, da der Code keine integrierte Fehlerprüfung eingebaut hat und daher bei Schemaabweichungen nicht funktioniert. Die Struktur lautet wie folgt:

<num, size, q_left, q_right, q_top, q_down, b_x, b_y, r_min, r_max, r_power>

ID	Variable	Beschreibung
0	num	Anzahl der Bubbles (Kreise) die erzeugt werden
1	size	Größe der Bubbles
2	q_left	Stärke, die eine Bubble (bei quake) nach links variieren kann
3	q_right	Stärke, die eine Bubble (bei quake) nach rechts variieren kann
4	q_top	Stärke, die eine Bubble (bei quake) nach oben variieren kann
5	q_down	Stärke, die eine Bubble (bei quake) nach unten variieren kann
6	b_x	Geschwindigkeit, mit der sich eine Bubble nach rechts bewegt (bounce)
7	b_y	Geschwindigkeit, mit der sich eine Bubble nach unten bewegt (bounce)
8	r_min	Kleinste Größe, die eine Bubble annehmen kann (rezise)
9	r_max	Größte Größe, die eine Bubble annehmen kann (rezise)
10	r_power	Geschwindigkeit, mit der eine Bubble ihre Größe ändert (rezise)

Eine mögliche Level-Definition sieht z.B. so aus: *8,60,6,9,8,3,0,0,40,90,1*

Weitere Hinweise

Eigenen Tests zufolge funktioniert dieses Projekt nur bei den Browsern „Google Chrome“ und „Edge“. Außerdem kann es sein, dass die Musik beim ersten Laden des Projekts nicht richtig eingelesen wird, obwohl *preLoad* dieses Problem beheben sollte. Leider konnte ich keinen Grund hierfür finden. Ein Neuladen der Seite mittels der F5-Taste sollte jedoch helfen.

Bubble-Painting könnte zukünftig noch durch ansprechende Animationen beim Spielstart und -ende, sowie erweitertem Farbwechsel der Bubbles und des Hintergrundes erweitert werden.