Programmieren für Nicht-Programmierer

Eine Einführung mit JavaScript

Oliver Zeigermann / http://zeigermann.eu

Druck Version

Web-Version unter: **http://zeigermann.de/non-dev**

Aufgabe 0: Kennenlernen der Programmierumgebung

1. Verbinde dich mit dem Internet
2. Öffne die Seite **http://jsfiddle.net/**
3. Gib im Kasten mit der Beschriftung JavaScript die folgende Anweisung ein

```
alert("Hallo, Leute");
```

4. Starte das Programm mit dem `Run`-Knopf
5. Vergewissere dich, dass ein kleines Fenster mit dem Text "Hallo, Leute" angezeigt wird

JavaScript

- Wir betrachten die Programmiersprache JavaScript
- Diese Programmiersprache läuft auf jedem Computer, der einen Webbrowser enthält
- Ihr könnt eigene HTML-Seiten mit JavaScript mit Leben füllen
- JavaScript ist weltweit einer der am weitest verbreiteten Programmiersprachen
- Wir benutzen den Browser, um eigenen Programme zu schreiben, wir müssen nichts installieren
- **http://jsfiddle.net/**

Anweisungen

- Programme in JavaScript bestehen häufig aus Anweisungen
- Mit Anweisungen teilt ihr dem Computer mit, was er tun soll
- Ihr müsst dem Computer Anweisungen in einer bestimmten Form geben, damit er sie versteht
- Der Computer ist sehr kleinlich

Beispiele

- Der Computer soll 10 plus 10 ausrechnen

```
10 + 10;
```

- Am Ende jeder Anweisung steht ein Semikolon
- Jeder Anweisung sollte in einer eigenen Zeile stehen
- Ihr könnt auch mit Texten arbeiten, diese müssen dann in Anführungszeichen stehen

```
"Olli " + " hat " + "Kaffeedurst";
```

Beispiele

- Der Computer soll 10 plus 10 nicht nur ausrechnen, sondern auch ausgeben
- Wir rufen die Funktion `alert` auf und übergeben ihr als Parameter, was sie ausgeben soll

```
alert(10 + 10);
```

- oder

```
alert("Olli " + " hat " + "Kaffeedurst");
```

- Diese Ausgaben erscheinen dann in einem Fenster

Funktionsaufrufe

- Damit richtig etwas passiert, müsst ihr Funktionen aufrufen
- Funktionen haben einen Namen und erwarten eine Reihe von Parametern
- Die Parameter geben der Funktion Informationen darüber, was im einzelnen passieren soll
- Ihr ruft eine Funktion auf, indem ihr den Namen der Funktion angebt und in runden Klammern danach die Parameter
- Mehrere Parameter werden mit Komma von einander getrennt

Variablen

- Wie in der Mathematik gibt es in der Programmierung Variablen
- Variablen können unterschiedliche Werte annehmen und sind veränderbar
- Ihr greift auf Variablen über ihre Namen zu
- Variablen können z.B. Texte, Zahlen oder Objekte enthalten

Beispiele

- Um eine neue Variable einzuführen, schreibt ihr das Wort `var` vor den Namen der Variablen
- Die Variable `laenge` hat den Wert 10

```
var laenge = 10;
```

- Ihr könnt den Wert einer Variablen auch ändern

```
laenge = 20;
```

- Ihr könnt auf den Wert einer Variablen jederzeit wieder zugreifen, z.B. um ihn einer Funktion als Parameter zu übergeben

```
alert(laenge);
```

Beispiele

- Ihr habt über die Variable `document` Zugriff auf das Objekt für die HTML-Seite
- Dieses Objekt ist von vorn herein verfügbar
- Das Objekt `document` hat z.B. einen Titel, an diesen kommt ihr über einen Punkt

```
alert(document.title);
```

- Über den Punkt kommt ihr an alle Eigenschaften eines Objekts, auch an seine Funktionen
- Über die Funktion `getElementById` kommt ihr an alle Teile eurer HTML-Seite

```
<h1 id="titel">Hallo, Leute</h1>
alert(document.getElementById("titel").innerHTML);
```

Objekte

- In der Programmierung gibt es Konstrukte, die Dinge der realen Welt abbilden
- Diese Konstrukte heißen Objekte
- Objekte beinhalten Eigenschaften und Funktionen
- Beispiele für Objekte: Gesicht, HTML-Seite, Leinwand
- Objekte könnt ihr in Variablen speichern
- Manche Objekte sind von vorn herein in Variablen gespeichert
- Andere Objekte könnt euch durch Aufrufe aus einem dieser Objekte geben lassen

Leinwand

- Ihr könnt in eure HTML-Seiten eine Leinwand einbinden

```
<canvas id="myCanvas" width="600px" height="400px"></canvas>
```

- An diese Leinwand kommt ihr als Objekt über seine `id` heran

```
var canvas = document.getElementById("myCanvas");
```

- Bevor es los geht, müssen wir noch alles einrichten für ein Zeichnen in 2D

```
var context = canvas.getContext('2d');
```

Zeichnen auf der Leinwand

- Über den `context` könnt ihr Funktionen aufrufen, die Dinge auf der Leinwand darstellen, z.B. einen Kreis

```
context.arc(centerX, centerY, 100, 0, 2 * Math.PI);
```

- Ihr könnt auch Eigenschaften der Darstellung verändern, z.B. die Breite des Strichs

```
context.lineWidth = 4;
```

- Oder die Farbe als ein kombinierter RGB-Farbwert wie in einer HTML-Seite

```
context.strokeStyle = '#000000';
```

Aufgabe 1: JavaScript kennen lernen

Start: **http://jsfiddle.net/DJCordhose/H2EZN/**

Musterlösung: **http://jsfiddle.net/DJCordhose/H2EZN/1/**

Stoff: Canvas, Variablen, Objekte, Funktionsaufrufe und Felder

Kommentare

- Manchmal möchte man zusätzliche Informationen zu seinem Programm geben
- Diese Information könnt ihr als einen Kommentar schreiben
- Kommentare beginnen mit `//`, z.B.

```
// Dies ist ein Kommentar innerhalb eines Programms
```

Schritte

1. Öffne die Seite **http://jsfiddle.net/DJCordhose/H2EZN/** in deinem Browser
2. Vergewissere dich, dass unten rechts ein paar geometrische Objekte dargestellt werden
3. Mache einige Experimente in denen du diese Darstellung verändert. Versuche Änderungen von
   - Farbe
   - Strichdicke
   - Position
   - Radius von Kreisen und Scheiben und
   - Anzahl der Objekte
4. Verändere die Darstellung so, dass ein Gesicht dargestellt wird. Dieses sollte zumindest zwei Augen enthalten

Eigene Funktionen

- Funktionen fassen mehrere Zeilen Programmiercode zusammen
- Funktionen werden über ihren Namen definiert und können eine Reihe von Parametern erwarten
- Die Parameter werden über eine Liste von Namen definiert
- Innerhalb der Funktion verhalten sich die übergebenen Parameter wie Variablen

Fall-Unterscheidung

- Manchmal will man anhand einer bestimmten Bedingung mal das eine Stück Code, mal ein anderes ausführen
- Dazu gibt es in JavaScript eine Fallunterscheidung
- Bei dieser kann man ein Stück Code angeben, wenn eine Bedingung erfüllt ist
- ... und einen anderen, wenn die Bedingung nicht erfüllt ist
- Fallunterscheidungen könnt ihr verschachteln

Beispiele

- Ihr beginnt eine Funktionsdefinition mit dem englischen Wort `function`
- Dann kommen Name, Klammern und eine Liste von Namen, die die Parameter angeben
- Die Programmierzeilen, die die Funktion zusammenfasst, stehen in geschweiften Klammern `{ ... }`

```
function hallosagen(name) {
    alert("Hallo " + name);
}
```

- nun könnt ihr die Funktion aufrufen und es erscheint ein Fenster mit dem Text "Hallo, Olli"

```
hallosagen("Olli");
```

Beispiele #1

- Ihr beginnt eine Funktionsdefinition mit dem englischen Wort `if`
- Dann kommt in Klammern die Bedingung
- Die Programmierzeilen, die dann ausgeführt werden sollen, stehen in geschweiften Klammern `{ ... }`

```
if (name == 'Olli') {
    alert("Hallo, Meister!");
}
```

- Mit ==, <, >, <=, >= vergleicht ihr Werte

Beispiele #2

- Mit `else` könnt ihr angeben, was passieren soll, wenn die Bedingung nicht erfüllt ist

```
if (name == 'Olli') {
    alert("Hallo, Meister!");
} else {
    alert("Hallo!");
}
```

- Ob eine Variable oder ein Parameter überhaupt vorhanden sind könnt ihr ohne Vergleich prüfen

```
if (name) {
    alert("Name ist da!");
} else {
    alert("Huch, wo ist der Name?!");
}
```

Schritte

1. Speichere deine Lösung aus Aufgabe 1 (Update drücken) oder beginne mit der Musterlösung
2. Führe eine Funktion “kreis” mit den Parametern “x”, ”y”, “radius” und “gefuellt” ein. Diese soll an der Position “x”, “y” entweder einen Kreis oder eine Scheibe mit dem Radius “radius” malen. Wenn der Parameter “gefuellt” gesetzt ist, soll eine Scheibe gemalt werden.
3. Male deine Augen ausschließlich mit Aufrufen dieser Funktion. Das Gesicht sollte am Ende dieser Übung genau so aussehen wie am Anfang.

Aufgabe 2: Programm-Code zusammenfassen und wiederverwenden

Start: **http://jsfiddle.net/DJCordhose/H2EZN/1/**

Musterlösung: **http://jsfiddle.net/DJCordhose/H2EZN/2/**

Stoff: Fallunterscheidungen und eigenen Funktionen

Vielleicht ist dir schon aufgefallen, dass der Programm-Code für dein Gesicht häufige Wiederholungen enthält. Dies wollen wir zusammenfassen und wiederverwenden.

Benutzer-Eingaben

- Euer Programm wird interessanter, wenn es auf Eingaben reagiert
- Eingaben können z.B. über die Tastatur oder über die Maus passieren
- In JavaScript reagiert ihr auf Eingaben indem ihr eine Funktion angebt
- Diese wird aufgerufen, sobald eine Eingabe passiert
- Eingaben laufen immer über HTML-Elemente, z.B. Leinwände oder Textfelder

Beispiel Textfeld

- Zuerst einmal braucht ihr das HTML-Element, dessen Eingabe ihr abfragen wollt, z.B.

```
<input type="text" id="name">
```

- Dies bekommt ihr am einfachsten als JavaScript-Objekt über seine `id` mit `document.getElementById`

```
var textfeld = document.getElementById("name");
```

- Diesem Objekt teilt ihr nun mit, welche Funktion bei einer Änderung der Eingabe ausgeführt werden soll

```
function gibEingabeAus () {
    alert(textfeld.value);
}
textfeld.onchange = gibEingabeAus;
```

Komplettes Beispiel

http://jsfiddle.net/DJCordhose/8XjWh/

Aufgabe 3: Mausbewegungen nutzen

Start: **http://jsfiddle.net/DJCordhose/H2EZN/2/**

Musterlösung: **http://jsfiddle.net/DJCordhose/H2EZN/7/**

Stoff: Funktionen als Variablen, Reaktion auf Benutzereingaben

Schritte

1. Schreibe eine Funktion, die dein komplettes Gesicht zeichnet. Übergib dabei centerX und centerY als Parameter und verwende den bereits bestehenden Code.
2. Schreibe eine zusätzliche Funktion, die die komplette Leinwand löscht. Z.B. so

```
context.clearRect(0, 0, canvas.width, canvas.height);
```

3. Verfolge nun die Mauspositionen auf deiner Leinwand und stelle das Gesicht immer an der Position der Maus da. Lösche dazu zuerst die Leinwand und zeichne dann das Gesicht neu.

```
function beiMausbewegung (event) {
    loescheLeinwand();
    zeichneGesicht(event.clientX, event.clientY);
}
canvas.onmousemove = beiMausbewegung;
```

Schleifen

- Manchmal will man ein Stück Code mehrere Male ausführen
- Dazu gibt es in JavaScript ein Schleifen-Konstrukt
- In diesem zählt ihr über eine Variable die Schleifen-Durchläufe mit
- Mit einer Bedingung wie bei der Fallunterscheidung könnt ihr entscheiden, ob es einen weiteren Durchlauf geben soll

Beispiele

- Ihr beginnt eine Schleife mit dem englischen Wort `for`
- Dann kommen in Klammern, mit Semikolon mehrere Teile
  1. Das Anlegen einer neuen Variablen
  2. Die Bedingung für einen weiteren Schleifendurchlauf
  3. Das Hochzählen der Variablen
- Die Programmierzeilen, die bei jedem Schleifendurchlauf ausgeführt werden sollen, stehen in geschweiften Klammern `{ ... }`

```
for (var i = 0; i < 5; i = i + 1) {
    alert("Durchlauf: " + i);
}
```

Aufgabe 4: Eigene Augenbewegungen umsetzen

Start: **http://jsfiddle.net/DJCordhose/H2EZN/3/**

Beispiel Pupille: **http://jsfiddle.net/DJCordhose/H2EZN/4/**

Beispiel 1000-Äugler:
http://jsfiddle.net/DJCordhose/H2EZN/6/

Überlege dir eine eigene Augenbewegung und setze sie um.