TECHNISCHE UNIVERSITÄT WIEN
Vienna | Austria

Programmieren eines kleinen Spieles mit Hilfe von KI

Luis & Thomas

15. Mai 2025

Die Geschichte von HTML, CSS und JavaScript

- **HTML**: 1991 von Tim Berners-Lee entwickelt, strukturiert Webseiten. HTML5 (2014) brachte Multimedia- und Mobil-Support.
- **CSS**: 1996 eingeführt, trennt Design von Inhalt. CSS3 ermöglicht Animationen und moderne Layouts.
- **JavaScript**: 1995 von Brendan Eich entwickelt, macht Webseiten interaktiv. ECMAScript 6 (2015) brachte moderne Features.
- **Bedeutung**: Basis des modernen Webs, Grundlage für Frameworks wie React und Vue.

Grundstruktur eines HTML-Dokuments

- **DOCTYPE**: Definiert den HTML-Typ (<!DOCTYPE html>).
- <**html**>... </**html**>: Umschließt das gesamte Dokument.
- <**head**>... </**head**>: Enthält Metadaten (Titel, CSS, Skripte).
- <**body**>... </**body**>: Enthält den sichtbaren Inhalt.

Ein minimaler HTML-Code

```
<!DOCTYPE html>
<html>
<head>
    <title>Meine erste Seite</title>
</head>
<body>
    <h1>Hallo Welt!</h1>
</body>
</html>
```

Wichtige HTML-Tags

- `<h1>` bis `<h6>` – Überschriften in verschiedenen Größen.
- `<p>` – Absatz für normalen Text.
- `<a href="">` – Link zu einer anderen Seite.
- `<img src="" alt="">` – Bild einfügen.
- `<ul>` und `<ol>` – Ungeordnete und geordnete Listen.
- `<li>` – Listeneintrag in einer Liste.
- `<table>`, `<tr>`, `<td>` – Tabelle mit Zeilen und Zellen.
- `<div>` und `<span>` – Gruppierung von Elementen.

Ein einfaches CSS-Beispiel

```css
body {
    font-family: Arial, sans-serif;
    background-color: #f0f0f0;
    color: #333;
}

h1 {
    color: blue;
    text-align: center;
}

p {
    font-size: 16px;
    line-height: 1.5;
}
```

IDs und Klassen in CSS

IDs und Klassen stylen

- **ID-Selektor:** IDs werden verwendet, um einzelne HTML-Elemente eindeutig zu identifizieren. Sie werden mit `#idname` in CSS angesprochen. Beispiel: `#haupttitel` ändert die Farbe oder Schriftgröße eines bestimmten Elements.
- **Klassen-Selektor:** Klassen ermöglichen es, mehrere Elemente mit denselben Stileigenschaften zu versehen. Sie werden mit `.klassenname` angesprochen. Beispiel: `.text` kann für mehrere Absätze dieselbe Farbe oder Schriftgröße festlegen.

Farben in HTML (RGB-System)

RGB-Farben in HTML und CSS

- Farben werden im **RGB-Modell** definiert: Rot (R), Grün (G) und Blau (B).
- Wertebereich: **0 bis 255** (je höher der Wert, desto intensiver die Farbe).
- Beispiel für CSS-Farben mit `rgb(r, g, b)`:

`color: rgb(255, 0, 0);` → Rot
`color: rgb(0, 255, 0);` → Grün
`color: rgb(0, 0, 255);` → Blau

- Alternative Hexadezimale Schreibweise: `#RRGGBB`, z. B. `#FF0000` für Rot.
- Transparenz mit Alpha-Kanal: `rgba(r, g, b, a)`, wobei a von 0 (transparent) bis 1 (sichtbar) reicht.

JavaScript: Variablen

Eine Variable speichert einen Wert, den man später im Code wiederverwenden oder ändern kann.

```
let zahl = 42;  // Eine Variable mit let
const PI = 3.14; // Eine Konstante mit const
var name = "Alice"; // Aeltere Schreibweise mit var

console.log(zahl);  // Gibt 42 aus
console.log(PI);    // Gibt 3.14 aus
console.log(name);  // Gibt "Alice" aus
```

JavaScript: Schleifen

```
1  // For-Schleife
2  for (let i = 0; i < 5; i++) {
3      console.log("Durchlauf: " + i);
4  }
5  // While-Schleife
6  let j = 0;
7  while (j < 5) {
8      console.log("While: " + j);
9      j++;
10 }
11 // Do-While-Schleife (mindestens einmal ausfuehren)
12 let k = 0;
13 do {
14     console.log("Do-While: " + k);
15     k++;
16 } while (k < 5);
```

JavaScript: Verzweigungen

```
let zahl = 10;

if (zahl > 0) {
    console.log("Die Zahl ist positiv");
} else if (zahl < 0) {
    console.log("Die Zahl ist negativ");
} else {
    console.log("Die Zahl ist null");
}
```

JavaScript: Operatoren

```
let sum = 10 + 5;    // Addition -> 15
let diff = 10 - 5;   // Subtraktion -> 5
let prod = 10 * 5;   // Multiplikation -> 50
let div = 10 / 5;    // Division -> 2
let mod = 10 % 3;    // Modulo -> 1

// Vergleichsoperatoren (geben true oder false zurueck)
console.log(10 > 5);    // true
console.log(10 == "10"); // true (lockerer Vergleich)
console.log(10 === "10"); // false (strikter Vergleich)

// Logische Operatoren
console.log(true && false); // false (UND)
console.log(true || false); // true (ODER)
console.log(!true); // false (NICHT)
```

JavaScript: Funktionen

```javascript
function sagHallo(name) {
    return "Hallo, " + name + "!";
}

let gruss = sagHallo("Alice");
console.log(gruss);  // Gibt "Hallo, Alice!" aus

// Anonyme Funktion
let addiere = function(a, b) {
    return a + b;
};

console.log(addiere(3, 4));  // Gibt 7 aus

// Arrow Function (ES6)
const multipliziere = (a, b) => a * b;
console.log(multipliziere(5, 2));  // Gibt 10 aus
```

JavaScript: Events

Events sind Aktionen oder Ereignisse, die im Browser passieren, z. B. wenn ein Nutzer auf einen Button klickt, eine Taste drückt oder die Maus bewegt. JavaScript kann auf solche Ereignisse reagieren und entsprechend Code ausführen.

```
1 <button id="meinButton">Klick mich!</button>
2
3 <script>
4 document.getElementById("meinButton").addEventListener("click",
      function() {
5     alert("Button wurde geklickt!");
6 });
7 </script>
```

JavaScript: DOM-Manipulation

Das Document Object Model (DOM) stellt die Struktur einer HTML-Seite als Baum dar, in dem jedes Element ein Knoten ist. JavaScript kann diesen Baum verändern, um Inhalte dynamisch anzupassen, Elemente hinzuzufügen oder zu entfernen.

```
<p id="text">Alter Text</p>
<button onclick="aendereText()">Aendere Text</button>

<script>
function aendereText() {
    document.getElementById("text").innerHTML = "Neuer Text!";
}
</script>
```

Hands-on: Programmieren eines Spiels

1. **Live-Coding:**
   - Wir programmieren gemeinsam ein kleines Spiel mit HTML, CSS und JavaScript.
   - Einführung in Spiellogik, Events und Interaktivität.
2. **Eigenes Spiel entwickeln:**
   - Nutzt Duck.AI, um euer eigenes Spiel zu erstellen.
   - Alternativ könnt ihr Scratch verwenden – mit Hilfe von Duck.AI und anderen Tutorials.
   - Ziel: Ein funktionierendes, spielbares Mini-Game!
3. **Präsentation:**
   - Stellt euer Spiel am Ende der Session vor.
   - Zeigt, was ihr gebaut habt, und erklärt kurz eure Ideen.