Webentwicklung Grundkurs Teil 1

Description

Spieleentwickler haben mit Webentwicklung mehr gemein als man annehmen möchte. Sei es die eigene HP zum Spiel, eine Online-Highscoreliste, die Einbettung des eigenen HTML5-Exports oder ein eigenes Browser-Spiel. Diese Serie geht auf die Basics der Webentwicklung ein und zeigt, wie HTML, CSS, JavaScript und PHP ineinandergreifen. Wer einen leicht verständlichen Einstieg in das Thema sucht, ist hier genau richtig.

Die Kurs-Reihe

Nichts geht über Grundlagen. Das merkt man spätestens dann, wenn man in fremden Skripten herumwerkelt und keine Ahnung hat, was man da eigentlich tut. Als Hobby-Spieleentwickler passiert das ziemlich oft. Sei es ein fremdes Template für die HP, eine fremde Erweiterung für das eigene Spiel oder irgendwelche PlugIns, die man für das CMS (Content-Management-System) noch schnell anpassen will.

Worum es in dieser Reihe geht, sind die absoluten Basics mit einigen Fallbeispielen. Sie sollen ein Grundverständnis für die Thematik bieten und Lust auf mehr machen. Ich habe versucht, die einzelnen Lektionen straff zu halten, was aber nicht immer gelingt. Manchmal muss man einfach mehr erklären, damit die Thematik verstanden werden kann. Das Leitmotiv lautet aber: Lieber mehrere kurze Kurse als wenige lange.

Begleitet werden die Tutorials von Interviews mit Profis, die entweder selbst Browserspiele entwickeln oder sich mit der Thematik seit vielen Jahren befassen.

Die Qual der Wahl

Für die Webentwicklung gibt es zahlreiche Sprachen und Entwicklungsumgebungen. Der Grundkurs geht dabei auf die Klassiker ein:

- HTML

- CSS
- JavaScript
- PHP

Wer das Zusammenspiel versteht und für die beiden Programmiersprachen JavaScript und PHP ein Grundverständnis entwickelt, kann schon sehr viel reißen. Hilfreich sind natürlich Vorkenntnisse in GML, C#, C++ oder Java, aber keine Grundvoraussetzung.

Entwicklungsumgebung

Für die Webentwicklung reichen im Prinzip ein Texteditor und ein Browser. Wer PHP schreibt, braucht noch einen Webserver, damit der Code interpretiert werden kann. Hierfür eignet sich lokal XAMPP, ansonsten kann man auch auf einem gemieteten Server online arbeiten. Da PHP am Ende der Kursreihe kommt, kann darauf vorerst verzichtet werden.

Als Editor für den Einstieg kann ich Notepad++ empfehlen. Einmal eingerichtet lassen sich damit kleine und mittelgroße Projekte gut realisieren. Wer mehr Ambitionen hat, kann sich Visual Studio Code installieren. Beide Umgebungen reichen für den Kurs voll aus.

Im ersten Teil liegt das Hauptaugenmerk darauf, das Zusammenspiel der einzelnen Bestandteile zu verstehen. Ab dem zweiten Teil arbeiten wir an konkreten Beispielen.

CMS

Die meisten Webseiten wie auch Content-Management-Systeme (CMS) bestehen aus HTML, CSS, JavaScript und PHP. Was es damit auf sich hat, beschreibe ich gleich.

Ein CMS ist, vereinfacht gesagt, ein System, um Webseiten aufzubauen und Inhalte bereit zu stellen. Das derzeit meist genutzte ist WordPress. Rund jede dritte Webseite wird mittlerweile mit WordPress erstellt. Ursprünglich wurde es als Blog-System konzipiert. Aufgrund der offenen Architektur und zehntausenden von Erweiterungen und Templates hat es sich zu einem allumfassenden Werkzeug gemausert.

Ebenfalls sehr bekannt ist Joomla. Verglichen mit WordPress hat es seine Vor- und Nachteile. Es ist bei größeren Seiten etwas übersichtlicher und bietet etwas mehr Möglichkeiten. Dafür ist der Einstieg bei WordPress einfacher und viele PlugIns wirken „ausgereifter”.

Wer eine eigene HP erstellen möchte, sollte sich gut überlegen, ob er alles selbst schreibt oder ein CMS verwendet. Und wenn ein CMS, welches.

Die Basis

Wir wollen im Kurs aber nicht lernen, wie man ein CMS installiert und einrichtet, sondern die Grundlagen verstehen. Fangen wir also damit an, was die kryptischen Abkürzungen bedeuten.

HTML

Die Abkürzung steht für *Hypertext Markup Language*. Sie wurde 1992 spezifiziert und war eigentlich dafür gedacht, einen Standard zu schaffen, mit dem wissenschaftliche Dokumente über Nationen hinweg ausgetauscht werden können. Es entstand also quasi mit dem, was wir heute als Internet bezeichnen.

HTML ist eine Auszeichnungssprache, keine Programmiersprache! Das bedeutet, dass HTML lediglich den Inhalt darstellt und ggf. dem Browser ein paar Informationen über die Webseite verrät. Das betrifft den Seitentitel, das Icon und Informationen für Suchmaschinen wie eine Seitenbeschreibung und Schlagwörter.

CSS

CSS steht für *Cascading Style Sheets*. Es ist eine sog. Stylesheet-Sprache. CSS entstand aus der Notwendigkeit heraus, HTML und Stil der Seite zu trennen. Das hat sehr viele Vorteile für die Entwicklung von Webseiten. Im HTML definiert man, wo der entsprechende Inhalt der Seite steht, in CSS wird definiert, wie der Inhalt aussieht. So kann man sehr leicht das Aussehen einer Seite ändern, ohne dass man jede HTML-Seite anpassen muss.

In CSS werden u. a. Schriften, Schriftgrößen, Farben, das Aussehen von Tabellen und viele andere Dinge definiert. Hat man mehrere solcher Dateien für eine HP, könnte man den Besucher mit einem Mausklick selbst entscheiden lassen, welche Darstellung er bevorzugt.

JavaScript

HTML und CSS bieten bereits viele Möglichkeiten, aber erst durch JavaScript wird eine richtige Interaktion möglich. Bspw. Spiele im Browser. Dank des modernen HTML5 und CSS3 braucht man nicht mehr für jede Kleinigkeit JS (die gängige Abkürzung für JavaScript), aber für zahlreiche Anwendungsbeispiele kommt man kaum drum herum.

JS hat aus verschiedenen Gründen keinen sehr guten Ruf. Die Programmiersprache (genauer gesagt ist es eine Client basierte Skriptsprache) hat ihre Eigenheiten, von denen ein paar mittlerweile ausgemerzt wurden. Außerdem hat sie den Ruf, dass böswillige Seiten mittels JS ziemlich viel Schindluder treiben. Mittlerweile werden die Möglichkeiten von JS vom Browser ziemlich limitiert, dennoch setzen viele User sog. No-Skript PlugIns im Browser ein, welche alle JavaScripte automatisch blockieren.

Deshalb gilt: JS sollte nur wohl bedacht eingesetzt werden. Bei den meisten Spielen kommt man nicht drum herum, aber man braucht nicht für jeden albernen Button-Effekt JS. Manches lässt sich bereits mit CSS realisieren.

Client-Seitig bedeutet übrigens, dass der Code direkt beim User ausgeführt wird. Es ist somit abhängig vom Browser, ob alles funktioniert oder nicht. Das hat Vor- und Nachteile. Ein Vorteil ist, dass man die Skripte an den Seitenbesucher sendet und er nicht mehr laufend nachladen muss. Für Anwendungen und Spiele ist das perfekt. Auf der anderen Seite bekommt der User damit natürlich auch den ganzen

Code und damit ggf. auch das Knowhow.

Dadurch wird vielleicht auch klar, was *Skriptsprache* bedeutet. Das sind Programmiersprachen, deren Programmcode nicht kompiliert wird. Er ist frei einsehbar und wird zur Laufzeit interpretiert. Bei JavaScript geht also jedes Mal der Browser über den Code und führt ihn aus. Abgesehen davon, dass jeder in den Code Einblick hat, ist diese Methode relativ langsam.

PHP

PHP steht für *PHP: Hypertext Preprocessor*. Sie ist ebenfalls eine Skriptsprache, im Gegensatz zu JS aber Serverbasiert. Das heißt: PHP Skripte werden auf dem Server ausgeführt, dem Benutzer wird lediglich, wenn überhaupt, das Resultat mitgeteilt.

Ein wesentlicher Unterschied zwischen JS und PHP ist, dass man bei PHP immer eine Seite aufrufen, also dem Server „Bescheid geben" muss, damit etwas passiert. Das bedeutet: In Sachen Interaktivität ist PHP nicht gerade die erste Wahl.

Dennoch ist es perfekt für dynamische Webseiten, was zunächst nach einem Widerspruch klingt. Warum das keiner ist, schauen wir uns im zweiten Teil genauer an.

PHP eignet sich auch hervorragend um Daten zu verarbeiten. Bspw. bei Abfragen von Datenbanken, Verarbeitung von hochgeladenen Dateien, automatisierte Wartungsarbeiten und vieles mehr. Was PHP liefert ist, verglichen mit JS, für den Endanwender meist nicht das, was man als „cool" bezeichnen würde, aber es ist ein solides Arbeitspferd, das viele gute Dienste leistet.

Übrigens, um die Info vorweg zu nehmen: In PHP kann man prozedural und objektorientiert programmieren und dies sogar gemischt. JavaScript kann ebenfalls objektorientiert genutzt werden, ist aber eine sog. klassenlose Sprache.

Ausblick

Im zweiten Teil bauen wir uns ein Gerüst um zu sehen, wie die vier Elemente einer modernen Webseite zusammenspielen. Spätestens hier wird deutlich, wie die Aufgaben verteilt werden und warum das alles so *kompliziert* sein muss.

Überblick Webdev-Serie

Webentwicklung Grundkurs Teil 1 – Einstieg
Webentwicklung Grundkurs Teil 2 – Aufbau von Webseiten
Webentwicklung Grundkurs Teil 3 – Datei- und Ordnerstrukturen
Webentwicklung Grundkurs Teil 4 – Einstieg in JavaScript
Webentwicklung Grundkurs Teil 5 – Datenverarbeitung und Formulare mit JavaScript

Überblick Interviews

Interview mit Magnus Reiß – Webgamers
Interview mit Wolfgang Scheidle – Tischtennis Manager
Interview mit Warg – Drifting Souls II

Date Created
8. November 2019
Author
sven