JavaScript Grundlagen

Ralph Steyer

4. Ausgabe, März 2020

ISBN 978-3-86249-957-1

JAVS_2020

Plus

HERDT

1

Informationen zu diesem Buch

1.1 Voraussetzungen und Ziele

Zielgruppe

Die Zielgruppe dieses Buches sind Webseitenersteller und Webdesigner, die JavaScript einsetzen möchten. Dabei spielt es keine Rolle, ob diese z. B. bereits professionelle Webseiten für ein Internetportal oder nur für die eigene Homepage erstellt haben. Aber auch für Umsteiger aus anderen Programmiersprachen ist diese Unterlage geeignet.

Empfohlene Vorkenntnisse

Für das Verstehen dieses Buches werden gute Kenntnisse in HTML vorausgesetzt. Auf die Bedeutung von HTML-Tags wird nicht eingegangen. Erfahrungen in anderen Programmier- oder Skriptsprachen sind nicht notwendig.

Lernziele

In diesem Buch lernen Sie den Funktionsumfang von JavaScript kennen. Dabei werden Sie eigene Skripte erstellen und lernen, auf Ereignisse innerhalb des Browsers zu reagieren. Ebenso werden Sie Formulareingaben auswerten und Informationen in Cookies oder auch dem sogenannten Local Storage speichern und auslesen. Sie erhalten einen Einblick in den Umgang mit Ajax und in das jQuery-Framework sowie die Neuerungen von JavaScript, die unter dem Bezeichner „HTML5“ versteckt sind.

Hinweise zu Soft- und Hardware

Sofern nicht anders vermerkt, wird im Folgenden davon ausgegangen, dass Sie einen oder mehrere der unten genannten Browser in aktuellen Versionen verwenden. Beachten Sie, dass die meisten Browser mittlerweile einen äußerst kurzen Versionszyklus einhalten und in einem kurzen Zeitraum mehrere Versionen erscheinen, die sich oft nur geringfügig unterscheiden.

Name	Erhältlich unter der Internetadresse:
Internet Explorer	*https://www.microsoft.com/de-de/download/internet-explorer.aspx*
Edge	*https://www.microsoft.com/en-us/edge*
Mozilla Firefox	*https://mozilla-firefox.de.uptodown.com*
Opera	*https://www.opera.com/de*
Apple Safari	*https://www.apple.com/de/safari/*
Google Chrome	*https://www.google.com/chrome/*

Spezielle Browser für Windows XP (alte Internet Explorer) werden in dem Buch nicht mehr berücksichtigt und von Microsoft wurde der Support eingestellt. Mittlerweile werden alte Versionen des Internet Explorers in der Praxis auch kaum noch eingesetzt.

1.2 Inhaltliche Konventionen

Hervorhebungen im Text

Damit Sie bestimmte Elemente auf einen Blick erkennen und zuordnen können, werden diese im Text durch eine besondere Schreibweise hervorgehoben. So werden Datei-und Ordnernamen, Hyperlinks und Bezeichnungen für Menüs bzw. Menüpunkte und Programmelemente wie Register oder Schaltflächen immer *kursiv* geschrieben und wichtige Begriffe **fett** hervorgehoben.

`Courier New`	kennzeichnet Programmtext, Programmnamen, Funktionsnamen, Variablennamen, Datentypen, Operatoren etc.
`Courier New Kursiv`	kennzeichnet Zeichenfolgen, die vom Anwendungsprogramm ausgegeben oder ins Programm eingegeben werden.
[ ]	Bei Darstellungen der Syntax einer Programmiersprache kennzeichnen eckige Klammern optionale Angaben.
/	Bei Darstellungen der Syntax einer Programmiersprache werden alternative Elemente durch einen Schrägstrich voneinander getrennt.

Sprachliche Konventionen

In den abgebildeten Beispielen wird aufgrund der verwendeten Dokumenttyp-Deklaration auf die exakte Schreibweise der öffnenden und schließenden Tags verzichtet. Darum werden z. B. `<p>`-Tags ohne das schließende `</p>`-Tag verwendet.

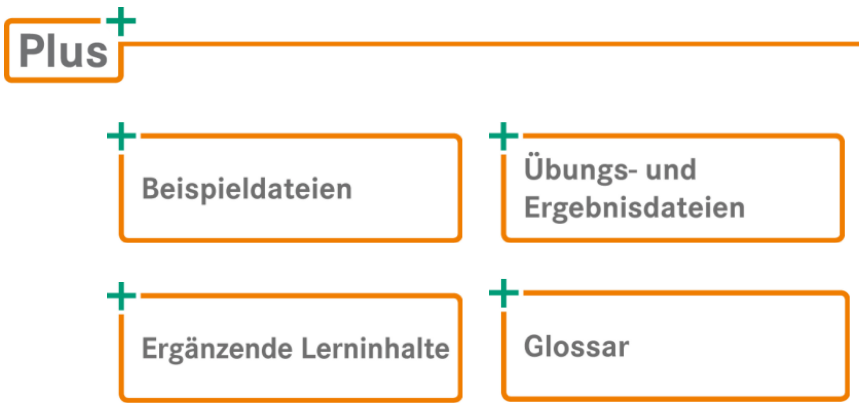
1.3 Bevor Sie beginnen …

HERDT BuchPlus - unser Konzept:

Problemlos einsteigen - Effizient lernen - Zielgerichtet nachschlagen

(weitere Infos unter *www.herdt.com/BuchPlus*)

Nutzen Sie dabei unsere maßgeschneiderten, im Internet frei verfügbaren Medien:



- Rufen Sie im Browser die Internetadresse *www.herdt.com* auf.

In den Übungsdateien wird über die Meta-Angabe `charset` die Zeichencodierung UTF-8 verwendet. Damit ist es nicht mehr notwendig, dass Sie z. B. für deutsche Umlaute die Zeichenreferenz angeben. Dazu wurden die Übungsdateien im Unicode-Format abgespeichert und müssen daher auch in einem Unicode-fähigen Texteditor geöffnet werden, um sie verändern zu können. Der aktuelle Windows Editor *notepad* unterstützt auch das Unicode-Textformat.

2

Einführung in JavaScript

2.1 Entstehung von JavaScript

Die Sprache JavaScript wurde 1995 von der Firma Netscape Communications in Kooperation mit der Firma Sun Microsystems ins Leben gerufen. Der ursprüngliche Name war LiveScript. Aus Marketinggründen wurde ein ähnlicher Name wie für das Sun-Produkt Java gewählt, um eine gemeinsame Vermarktungsstrategie zu gewährleisten. Eingeführt wurde JavaScript im damaligen Browser Netscape Navigator 2.

Im Zuge der Standardisierung von JavaScript arbeitete die Firma Netscape später mit der ECMA (**E**uropean **C**omputer **M**anufacturers **A**ssociation) zusammen. Die daraus resultierende Spezifikation wurde ECMA-262 mit weiteren Unterversionen (Editionen) genannt. Die offizielle Version von JavaScript heißt deshalb ECMAScript.

Nach dem Ende der Firma Netscape Communications und der Einstellung des Browsers Netscape Navigator übernahm die Mozilla Foundation (Browser Firefox) die Weiterentwicklung der Skriptsprache.

2.2 Grundlagen zu JavaScript

Wie Sie JavaScript einsetzen können

JavaScript wurde entwickelt, um die statischen Inhalte von Webseiten mit dynamischen Inhalten und Interaktionen zu erweitern. Dies sollte letztendlich dazu dienen, Webseiten attraktiver zu gestalten und den Nutzern eine verbesserte Funktionalität zu bieten. Heutzutage erlangt JavaScript in Zusammenhang mit HTML5 und CSS3 und den modernen **Rich Internet Applications** (RIAs) immer mehr an Bedeutung beim Erstellen von applikationsähnlichen Programmen. JavaScript wird z. B. für die folgenden Anwendungsfälle genutzt:

- Bevor Sie die Inhalte von Formularen abschicken, können Sie über JavaScript die korrekte Eingabe der Daten sicherstellen. So können Sie beispielsweise prüfen, ob eine Postleitzahl nur aus Zahlen und im Falle von Deutschland aus genau fünf Ziffern besteht. Diese Prüfung erfolgt ohne Kommunikation mit dem Webserver. Dieser wird dadurch entlastet, und der Benutzer wird schneller auf Fehler hingewiesen.

- Sie können vollständige Anwendungen entwickeln, die in einer Webseite direkt beim Nutzer ausgeführt werden. Das können kleine Anwendungen wie z. B. ein Taschenrechner oder Währungsumrechner sein. Aber auch größere Anwendungen wie interaktive Kalender, Portale etc. sind mit JavaScript möglich.
- Es ist die Darstellung von dynamischen Inhalten möglich, ohne dass dazu Skripts auf dem Webserver oder spezielle Erweiterungen in dem Browser notwendig sind. So können Sie beispielsweise nach der Eingabe von Messdaten ein Diagramm anzeigen oder je nach eingegebenen Werten dynamisch weitere Formularelemente anzeigen bzw. verbergen.
- Sie können Daten zwischen Browser und Webserver austauschen, ohne dass Sie die Webseite im Browser neu laden müssen. Ein bekannter Einsatz ist hierbei die Anzeige von Suchergebnissen, während Sie noch die Suchanfrage eingeben.
- Sie können Daten auf dem Client speichern.
- Sie können den Ort des Besuchers einer Webseite bestimmen und ihm damit lokalisierte Informationen anzeigen.
- Sie können zahlreiche Daten des Clientrechners auswerten und damit Seiten und Inhalte an die Gegebenheiten beim Besucher anpassen.

Was ist JavaScript?

JavaScript ist eine Skriptsprache, mit der Sie vollwertig programmieren können, im Gegensatz zur Auszeichnungssprache HTML. Da JavaScript mit Objekten arbeitet, wird auch von einer objektbasierten oder (mit Abstrichen) objektorientierten Sprache gesprochen. Da ebenso funktionales sowie prozedurales Programmieren möglich ist, kann JavaScript sehr flexibel eingesetzt werden.

Die Verwendung von JavaScript ist im Gegensatz zu vielen anderen Programmiersprachen recht einfach zu erlernen, da sie weniger Sprachelemente besitzt und viele Dinge automatisch erledigt. Dennoch nimmt die Einarbeitung einige Zeit in Anspruch, da Sie nicht nur die Sprache, sondern auch das Einsatzgebiet kennenlernen müssen. Gerade die Einschätzung der sehr unterschiedlichen Verhaltensweisen von Browsern erfordert Erfahrung, und bei richtig komplexen Applikationen können auch JavaScripts sehr umfangreich werden. In der letzten Zeit wird JavaScript auch immer professioneller in großen Projekten eingesetzt und löst zum Teil mächtige Konkurrenztechnologien wie Java oder C# in geeigneten Umgebungen ab. Viele Programmierer mit Erfahrung in C#, Java oder C/C++ eignen sich deshalb JavaScript-Kenntnisse an.

JavaScript einsetzen

JavaScript kann im Browser (clientseitig – beim Benutzer) und in einigen Webservern (serverseitig) eingesetzt werden. Diese Unterscheidung ist für den Benutzer einer Webseite nicht relevant, da er beim Umgang mit JavaScript-Anwendungen nur mit clientseitigem JavaScript in Berührung kommt. JavaScript wird überwiegend in Browsern genutzt, gewinnt aber auch auf dem Server aktuell rasant an Bedeutung.

Die Implementierung von JavaScript erfolgt clientseitig direkt im Browser, indem ein Skript in eine Webseite eingebunden wird. JavaScript-Anwendungen können clientseitig nicht außerhalb, sondern nur innerhalb eines Browsers bzw. einer entsprechenden Ausführungsumgebung ausgeführt werden. Diese Abschirmung vom Rest des Computersystems wird auch als Sandbox bezeichnet.

Da Browser unter verschiedenen Betriebssystemen zum Einsatz kommen, wird JavaScript auch als plattformübergreifende Sprache bezeichnet. Der Vorteil für Sie ist dabei, dass ein JavaScript-Programm theoretisch unter verschiedenen Browsern und Betriebssystemen gleichermaßen läuft. Dies ist aufgrund der unterschiedlichen JavaScript-Versionen und Implementierungen im Browser allerdings oft nur für einfache Skripts der Fall.

Wie wird JavaScript in eine Webseite eingebunden?

Mit JavaScript können Sie auf HTML-Elemente zugreifen bzw. HTML-Code erzeugen. Der JavaScript-Code wird dazu in das HTML-Dokument eingefügt oder über eine separate Datei eingebunden. Da sich der JavaScript-Code in einem für Menschen lesbaren Zustand befindet, muss er vom Browser erst interpretiert werden (Prüfung der syntaktischen Korrektheit, Übersetzung, Ausführung).

JavaScript wird in den Quelltext eines HTML-Dokuments eingebunden und gesondert gekennzeichnet. Die Einbindung erfolgt über das Tag `<script>`, das mit einem Ende-Tag abgeschlossen werden muss. Bei der Darstellung des HTML-Dokuments kann der Browser die relevanten Stellen zwischen den beiden `<script>`-Tags als JavaScript identifizieren und sie entsprechend verarbeiten. Browser, die kein JavaScript verstehen oder bei denen JavaScript deaktiviert ist, ignorieren diese Anweisungen.

Browser, die kein JavaScript verstehen, wurden meist vor 1995 entwickelt und werden nicht mehr verwendet.

```
<script type="text/javascript">
  <!-- JavaScript-Anweisungen -->
</script>
```

Das Attribut `type` kann entfallen, denn Browser werden als Vorgabe immer JavaScript verwenden. Sie geben mit dem Attribut bei einer vollständigen Notation allgemein den sogenannten MIME-Type (**M**ultipurpose **I**nternet **M**ail **E**xtensions) an. Mit HTML5 wird das Attribut offiziell nicht mehr gefordert. Die Beibehaltung des Attributs ist aber empfehlenswert, um deutlich zu kennzeichnen, dass man mit JavaScript arbeitet.

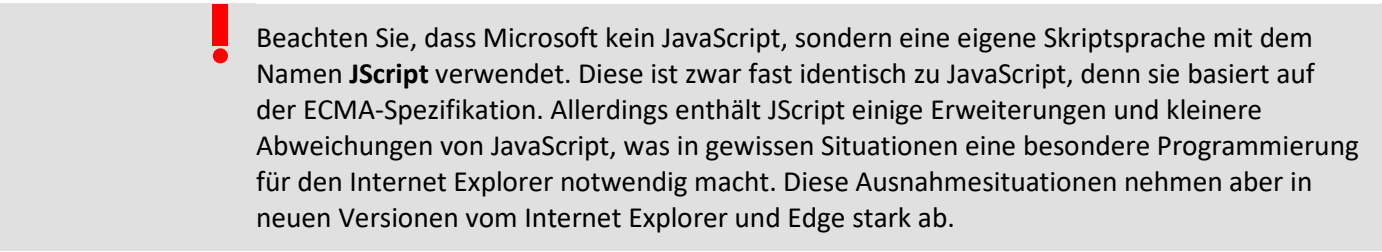
Sicherheit

Das Ausführen von JavaScript-Code kann vom Anwender in den Einstellungen des Browsers deaktiviert werden. Früher haben einige Benutzer JavaScript auch wegen Sicherheitslücken durch fehlerhafte Implementierungen in den Browsern ausgeschaltet. Die Gefahr durch JavaScript-Anwendungen ist jedoch gering, denn JavaScript-Anwendungen laufen im Browser in einer abgeriegelten Umgebung ab, in einer sogenannten Sandbox. Sie können nicht auf Dateien des lokalen Rechners zugreifen (Ausnahme – dafür explizit vorgesehene Dateien wie der lokale Speicher) und auch keine Benutzerdaten abfragen, die nicht auch über den Browser übermittelt werden können. Zudem werden die JavaScript-Implementierungen in den aktuellen Browsern immer besser und sicherer. Mittlerweile haben fast alle Anwender JavaScript aktiviert, denn moderne Webseiten sind ohne JavaScript meist nicht mehr zu verwenden, und Browser haben JavaScript in der Standardeinstellung aktiviert. Die Browserhersteller verstecken zudem die Möglichkeiten zur Deaktivierung von JavaScript so, dass ein typischer Anwender sie gar nicht findet.

2.3 JavaScript-Versionen

Die Sprache JavaScript wird von der Mozilla Foundation immer wieder erweitert, wobei die **praxisrelevante** Entwicklung im Grunde seit dem Jahr 2000 stagniert. Die Neuerungen aller späteren Vorschläge der Mozilla Foundation wurden gar nicht oder nicht einheitlich von den unterschiedlichen Browsern interpretiert.

Die Weiterentwicklung von JavaScript wird allerdings durch HTML5 vorangetrieben, und viele der proprietären JavaScript-Neuerungen der Mozilla Foundation, die seit dem Jahr 2000 für Firefox und seine Verwandten eingeführt wurden, halten über diesen „Marketingumweg" Einzug in andere moderne Webbrowser. Auch haben populäre JavaScript-Erweiterungen wie Ajax jenseits der „offiziellen" JavaScript-Versionen immer wieder für eine Weiterentwicklung der JavaScript-Features gesorgt.



Beachten Sie, dass Microsoft kein JavaScript, sondern eine eigene Skriptsprache mit dem Namen **JScript** verwendet. Diese ist zwar fast identisch zu JavaScript, denn sie basiert auf der ECMA-Spezifikation. Allerdings enthält JScript einige Erweiterungen und kleinere Abweichungen von JavaScript, was in gewissen Situationen eine besondere Programmierung für den Internet Explorer notwendig macht. Diese Ausnahmesituationen nehmen aber in neuen Versionen vom Internet Explorer und Edge stark ab.

Die jeweiligen Versionen von JavaScript werden von der Mozilla Foundation mit einer fortlaufenden Nummerierung gekennzeichnet und starteten im März 1996 mit der Version 1.0. Die derzeit letzte offizielle Versionsnummer 1.8.5 stammt vom Juli 2010 und ist mit ECMAScript 1.8.1 kompatibel. Die zum Zeitpunkt der Bucherstellung aktuellste Version ECMAScript 2017 stammt von Juni 2018 und wird von den meisten Browsern nicht vollständig unterstützt. Aber das soll keinesfalls bedeuten, dass sich im Umfeld von JavaScript nichts tut. Vereinheitlichungen in Browsern, Optimierungen der Ausführungsumgebungen und vor allen Dingen die Unterstützung zusätzlicher Features erfolgen permanent.

JavaScript-Versionen angeben

In modernen Webseiten gibt man normalerweise die verwendete JavaScript-Version nicht mehr an, da alle modernen Browser in etwa die gleichen Features unterstützen. Im Prinzip kann man bei der Einbindung von JavaScript jedoch eine bestimmte JavaScript-Version angeben. Damit können Sie inkompatible Browser von dem verwendeten Skript-Bereich fernhalten. Dazu können Sie die jeweilige Version mit dem Attribut `type` im `<script>`-Tag angeben. Unterstützt ein Browser nur eine niedrigere Versionsnummer, soll der entsprechende JavaScript-Code nicht ausgeführt werden. Andernfalls würden die neuen und damit unbekannten JavaScript-Befehle einen Fehler produzieren. Aber diese Vorsicht ist aus genannten Gründen in der Praxis mittlerweile unbegründet.

```
<script type="text/javascript; version=1.6">
  <!-- JavaScript-Anweisungen -->
</script>
```

Diese Versionsangabe im `<script>`-Tag über einen speziellen Wert im Attribut `type` ist nicht standardisiert und wird zurzeit nur von Mozilla Firefox und dessen Verwandten ausgewertet. Früher hat man bei der Überprüfung der maximal möglichen JavaScript-Version stattdessen das eigentlich veraltete `language`-Attribut verwendet. Die gleichzeitige Verwendung von `type` und `language` muss unterbleiben, da `language` immer ignoriert wird, wenn Sie gleichzeitig auch `type` notieren. Aber die Angabe der Version sollte wie gesagt unterbleiben, wenn es nicht einen unabdingbaren Grund dafür gibt.

2.4 JavaScript, JScript, VBScript und Java

Da Netscape die Rechte am Namen JavaScript besitzt und es zu Beginn keine Standardisierung gab, entwickelte Microsoft eine eigene Skriptsprache mit dem Namen JScript als JavaScript-Clone.

Alternativ wurde mit VBScript (**V**isual **B**asic **S**cript) eine weitere Skriptsprache im Internet Explorer bereitgestellt, die normalerweise in den Microsoft-Office-Produkten zum Einsatz kommt und auf der Sprache Visual Basic basiert. Sie wird nur vom Internet Explorer unterstützt und hat im Web keine Bedeutung, besonders wegen der großen Sicherheitsprobleme bei VBScript.

Häufig kommt es aufgrund der Ähnlichkeit der Namen JavaScript und Java zu Verwechslungen. JavaScript wurde von Netscape und Sun gemeinsam vorgestellt. Sun hatte den Namen Java geschützt, und so wählten die Firmen den Namen JavaScript. Die Gemeinsamkeiten zwischen den Sprachen Java und JavaScript sind syntaktisch groß, aber vom Konzept gibt es starke Abweichungen. Während mit JavaScript Skripte erstellt werden, die in der Regel im Browser (und immer mehr auch in einem Webserver) interpretiert werden, können mit Java beliebige Anwendungen entwickelt werden, die sowohl im Browser, aber auch auf Servern, Desktop-Rechnern, Smartphones oder beliebigen elektronischen Geräten (DVD-Player, Fernseher, Waschmaschinen etc.) laufen.

2.5 JavaScript-Aktivierung im Browser testen

Sie können davon ausgehen, dass JavaScript grundsätzlich bei den Anwendern aktiviert ist (Voreinstellung aller Browser) und die meisten Anwender JavaScript nicht deaktivieren. Dennoch können Sie nicht zu 100 % sicher sein, dass JavaScript aktiviert ist. Verwenden Sie das Skript im folgenden Beispiel, um die Aktivierung von JavaScript zu prüfen. Erstellen Sie ein HTML-Dokument, das eine erste einfache JavaScript-Anweisung ausführt. Die Erläuterung der genauen Funktionsweise ist in diesem Moment nicht von Interesse. Sie erfolgt im weiteren Verlauf des Buches.

Beispiel: *kap02/test.html*

Geben Sie den folgenden Quellcode in einem Texteditor ein, und speichern Sie ihn in der Datei *test.html*. Nach dem Öffnen des HTML-Dokuments in Ihrem Browser sollte sich ein Pop-up-Fenster öffnen.

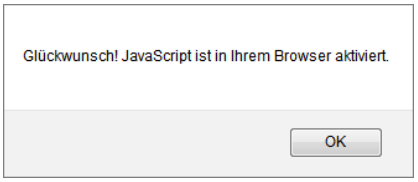
```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Der erste JavaScript-Test</title>
  </head>
  <body>
    <script type="text/javascript">
      alert('Glückwunsch! JavaScript ist in Ihrem Browser
             aktiviert.');
   </script>
  </body>
</html>
```

① Das Grundgerüst eines HTML-Dokuments wird eingeleitet.

② Dem Browser wird über die Anweisung `<script type="text/javascript">` mitgeteilt, dass die Anweisungen der verwendeten Skriptsprache als JavaScript zu interpretieren sind.

③ Die JavaScript-Funktion `alert()` zeigt ein Meldungsfenster an. Als Text wird die angegebene Zeichenkette eingeblendet.

④ Mit dem schließenden Tag `</script>` wird der JavaScript-Teil beendet.



JavaScript-Meldung im Mozilla Firefox

2.6 Nützliche Webseiten

https://developer.mozilla.org/en/Javascript	Informationen zu JavaScript
https://wiki.selfhtml.org/wiki/JavaScript	Einführung in JavaScript (SelfHTML, deutsch)
http://www.jswelt.de/	Umfangreiche JavaScript-Sammlung

Ergänzende Lerninhalte: *Informationen im Internet.pdf*

2.7 Übung

Theoriefragen zu JavaScript

Übungsdatei: -- **Ergebnisdatei:** *kap02/uebung1-3.pdf*

1. Wo werden JavaScript-Anweisungen ausgeführt?
2. Erklären Sie die Unterschiede der (Skript-)Sprachen VBScript, JScript, Java und JavaScript.
3. Zählen Sie einige der möglichen Einsatzgebiete von JavaScript auf.

3

Grundlegende Sprachelemente

3.1 JavaScript in HTML verwenden

JavaScript direkt einbinden

Über das `<script>`-Tag können Sie JavaScript-Code in ein HTML-Dokument einfügen. Der JavaScript-Bereich wird durch Angabe des `</script>`-Tags wieder beendet.

```
<script type="text/javascript">
  <!-- Hier werden die JavaScript-Anweisungen eingefügt -->
</script>
```

Der JavaScript-Code kann sowohl im `head`-Bereich (engl.: head = Kopf) als auch im `body`-Bereich (engl.: body = Körper, Rumpf) einer HTML-Datei eingefügt werden. Beim Laden eines HTML-Dokuments werden die JavaScript-Anweisungen sequenziell abgearbeitet. In der Regel werden Skripts im `<head>`-Tag eingefügt, um oft verwendete Funktionen und globale Variablen anzulegen (zu deklarieren). Skripts im `<body>`-Tag werden z. B. dazu genutzt, eine Ausgabe in das HTML-Dokument durchzuführen.

Innerhalb des `<script>`-Tags wird über das Attribut `type` der MIME-Typ der Skriptsprache angegeben, wobei diese Angabe mit HTML5 nicht mehr gefordert wird. Die Verwendung des alternativen Attributs `language` ist laut Standard grundsätzlich veraltet.

- Durch die Angabe des optionalen Attributs `defer` (ohne Wert) im `<script>`-Tag teilen Sie dem Browser mit, dass der JavaScript-Code keine Ausgabe im HTML-Dokument erzeugt. Diese Angabe ist rein informell und dient dazu, dass der Browser dadurch das HTML-Dokument schneller darstellen kann, da der JavaScript-Code nicht sofort analysiert (geparst) werden muss, z. B. `<script type="..." defer>`.
- Wenn Sie über JavaScript Ausgaben im HTML-Dokument vornehmen, wird dieser Code erneut interpretiert und die Webseite neu aufgebaut. Auf diese Weise können Sie über JavaScript beispielsweise JavaScript-Code erzeugen, der ebenfalls interpretiert und ausgeführt wird. Für das Erzeugen umfangreicher dynamischer Webseiten ist dies eine gängige Praxis.

In einer Webseite kann es mehrere Skriptbereiche geben. Diese bilden einen gemeinsamen **Namensraum**. Namensraum bedeutet, dass Elemente, die Sie in einem Skriptbereich deklariert haben, auch in den anderen (nachfolgenden) Skriptbereichen verfügbar sind. Ebenso müssen Bezeichner in einem Namensraum eindeutig sein. Das gilt unabhängig von der Art der Einbindung.

Einbinden als externe Dateien

JavaScript-Code können Sie auch als separate Dateien einbinden. Dies ist z. B. nützlich, wenn Sie Funktionen in mehreren HTML-Dokumenten verwenden möchten. Durch die Deklaration der Funktionen in einer Datei steht Ihnen eine zentrale Stelle zur Verfügung, an der Sie Quellcode anpassen können, ohne alle entsprechenden HTML-Dokumente bearbeiten zu müssen.

```
<script type="text/javascript" src="URL"></script>
```

Eigene JavaScript-Dateien werden meist in einem separaten Unterverzeichnis gespeichert. In der Praxis haben sich Verzeichnisstrukturen wie *lib/js* (lib steht für Library-Bibliothek), *lib/scripts* oder nur *scripts* oder *js* etabliert.

Die (ausschließliche) Verwendung externer JavaScript-Dateien ist bei modernen Webseiten der Regelfall. Nur damit können Sie die Trennung von Struktur und Funktionalität erreichen, was bei komplexeren Webseiten unabdingbar ist. Beachten Sie, dass in den Beispielen im Buch **rein aus didaktischen Gründen** in der Regel mit internen Skript-Containern gearbeitet wird. Damit lassen sich Zusammenhänge leichter erklären. In der Praxis können Sie diese internen Skript-Container jederzeit in externe Skripte auslagern.

Über das Attribut `src` beim `<script>`-Tag geben Sie die URL zur Datei an, in der sich der JavaScript-Code befindet. Wie bei der Definition von Hyperlinks können Sie relative und absolute Pfadangaben verwenden. JavaScript-Dateien besitzen standardmäßig die Endung **.js*, wobei das aber nicht zwingend ist.

Eine JavaScript-Datei kann im `head`-Bereich oder `body`-Bereich eines HTML-Dokuments eingebunden werden. Vorzugsweise sollten Sie diese jedoch im `head`-Bereich einbinden, wenn in der JavaScript-Datei **Deklarationen** (das bedeutet die Einführung von neuen Elementen) durchgeführt werden. Erzeugen Sie jedoch direkt Ausgaben in der JavaScript-Datei, sollte die Datei in den `body`-Bereich eingebunden werden, da Sie im `head`-Bereich einer Webseite keine Ausgaben erzeugen können.

Beispiel: *kap03/javascript.js*

In der Datei befindet sich lediglich eine Ausgabeanweisung, die einen Text in ein HTML-Dokument schreibt.

```
document.write('Ich befinde mich in einer externen Datei.');
```

Beim Erstellen von JavaScript-Code in externen Dateien notieren Sie nur die JavaScript-Anweisungen.

Beispiel: *kap03/javascript-extern.html*

Binden Sie die JavaScript-Datei *javascript.js* im `body`-Bereich des HTML-Dokuments ein. Dadurch wird die Anweisung in der Datei ausgeführt und die betreffende Zeichenkette ausgegeben.

```
...
<head>
  <meta charset="utf-8">
  <title>JavaScript-Test</title>
</head>
<body>
  <script type="text/javascript" src="javascript.js"></script>
...
</body>
```

Verwendung in Funktionen und HTML-Tags

Zur Reaktion auf Ereignisse und bei der Verwendung in HTML-Tags können Sie JavaScript-Code auch ohne Angabe des `<script>`-Tags angeben. Dazu gibt es zwei Möglichkeiten:

- den Eventhandler und
- die Inline-Referenz.

Reaktion auf Ereignisse mit HTML-Eventhandlern

In HTML gibt es sogenannte **Eventhandler**, um auf gewisse Ereignisse zu reagieren, die bei bestimmten Elementen der Webseite auftreten. Das Ereignis `onclick` wird beispielsweise ausgelöst, wenn der Benutzer auf eine Schaltfläche eines Formulars klickt. In doppelten Anführungszeichen wird der JavaScript-Code angegeben, der in diesem Fall ausgeführt werden soll.

```
<button type="button" onclick="alert('Sie haben mich gedrückt');">
 Klicken Sie auf mich
</button>
```

Aufruf einer JavaScript-Funktion mit einer Inline-Referenz

Bei einer **Inline-Referenz** wird das Ziel eines Hyperlinks durch eine JavaScript-Funktion definiert. In diesem Fall muss der Text `javascript:` vor den Funktionsnamen gesetzt werden, da die JavaScript-Funktion die statische Angabe der URL ersetzt.

```
<a href="javascript:history.back()">Zurück</a>
```

Beide Varianten (Inline-Referenz und HTML-Eventhandler) zum Aufruf von JavaScript direkt bei HTML-Elementen sind veraltet und sollten in modernen Webseiten nicht mehr verwendet werden. Sie vermischen Struktur und Funktionalität, sind schlecht wartbar und lesbar und vom Umfang der Möglichkeiten sehr eingeschränkt. In alten Skripten werden Sie diese Aufrufe jedoch noch finden. Im Kapitel 9 lernen Sie, wie Sie heutzutage auf Ereignisse reagieren.