3 Ein kurzer Überblick über Webanwendungen

Dieses Buch soll Ihnen zeigen, wie Sie moderne Webanwendungen schreiben und dabei anstelle von Plugins nur die nativ im Webbrowser bereitgestellten Tools und Technologien verwenden. Es konzentriert sich dabei ausschließlich auf die Sprachen und Bibliotheken, die von den neuesten Versionen der folgenden Webbrowser unterstützt werden:

Unterstützte Browserversionen

- Chrome
- Firefox
- Internet Explorer
- Safari
- Opera

Viele Beispiele in diesem Buch funktionieren in älteren Browsern nicht. Oft gibt es dann auch keine Workarounds. Wenn Sie eine Webanwendung schreiben, lautet die erste Frage ganz klar: »Welche Browser und welche Versionen davon möchte oder muss ich unterstützen?«

Kompromisse

Hier müssen Sie auf jeden Fall Kompromisse eingehen:

Je mehr Browser und Browserversionen Sie unterstützen, desto mehr Anwender können Ihre Webanwendung nutzen. Vergessen Sie nicht: Manche Nutzer, vor allem in Unternehmen, können sich ihren Browser oder die Browser-Version nicht selbst aussuchen.

Je mehr Browser und Browserversionen Sie unterstützen, desto mehr Einschränkungen gibt es hinsichtlich der Verfügbarkeit und Kompatibilität von APIs. Weiter hinten in diesem Buch zeige ich Ihnen die sogenannten »Polyfills«, mit denen Sie die von einem bestimmten Browser nicht unterstützten Funktionen »upgraden« können. Dies funktioniert jedoch nicht immer.

Automatische Aktualisierungen

Alle gängigen Browser bieten mittlerweile automatische Aktualisierungen. Zwar können diese deaktiviert werden, aber dennoch ist es keine bloße Vermutung mehr, dass die meisten Benutzer mit der neuesten Version ihres Lieblingsbrowsers unterwegs sind – zumindest abseits ihres Arbeitsplatzes.

Ausnahme: Internet Explorer

Die wichtigste Ausnahme ist der Internet Explorer. Die Versionen 10 und 11 gibt es für ältere Windows-Versionen nicht. Daher haben viele Nutzer noch ältere Versionen des Internet Explorers. Die meisten Beispiele in diesem Buch funktionieren im Internet Explorer 9, manche auch im Internet Explorer 8. Die Unterstützung des Internet Explorers 6 und 7 ist jedoch ein eher schwieriges Unterfangen.

caniuse.com

Die Website *http://caniuse.com* ist eine unschätzbar wertvolle Ressource zum Funktionsumfang der verschiedenen Browser und Browser-Versionen.

3.1 Was ist eine Webanwendung?

Was ist eine Webanwendung und wie unterscheidet sie sich von einer Website? Das ist eine durchaus berechtigte Frage. Selbst die Einführung zur HTML5-Spezifikation spricht vom »nicht klar umrissenen Bereich der Webanwendungen«.

Eigenschaften der Beispiel-Webanwendung

Es gibt keine endgültige Antwort auf diese Frage; die in diesem Buch entwickelte Webanwendung hat jedoch die folgenden Eigenschaften:

- Für die Benutzeroberfläche wird ein Webbrowser verwendet.
- Der Nutzer kann Aktionen ausführen und Daten manipulieren, ohne die Seite neu laden zu müssen.
- Die Anwendung ist interaktiv und reagiert sofort auf Benutzereingaben.
- Sie speichert für den Benutzer Daten, entweder auf dem Client oder dem Server.
- Wenn die Anwendung auf einen Webserver zugreifen muss, werden asynchrone (A)JAX-Aufrufe verwendet.
- Asynchrone APIs werden gegenüber synchronen APIs bevorzugt. Manche Webanwendungen sind auch dann verfügbar und einsatzfähig, wenn der Benutzer nicht mit dem Internet verbunden ist.

Hinweis

In diesem Buch wird der Unterschied zwischen einer asynchronen und einer synchronen API immer wieder wichtig. Sie werden hierzu zahlreiche Beispiele erhalten. Der wichtigste Unterschied ist:

Eine synchrone API wartet auf eine Rückmeldung. Bis diese eingetroffen ist, werden alle anderen Prozesse der Anwendung blockiert.

Anders die asynchrone API: Hier können weitere Prozesse ausgeführt werden, bis eine Benachrichtigung über die Rückmeldung vorliegt.

Die HTML5-Spezifikation schlägt für Webanwendungen außerdem auch die folgenden Merkmale vor:

Weitere Merkmale von Webanwendungen

- Sie werden gelegentlich genutzt oder aber regelmäßig, jedoch von verschiedenen Orten aus.
- Sie benötigen nur eine geringe CPU-Leistung.

Diese Aussagen sind sicherlich in gewisser Weise richtig. Webanwendungen zur Textverarbeitung sind zum Beispiel deutlich im Vorteil, wenn Dokumente von mehreren Personen an unterschiedlichen Orten bearbeitet werden sollen, selbst wenn sie vielleicht nicht ganz so benutzerfreundlich und leistungsstark sind wie ein richtiges Textverarbeitungsprogramm.

In diesem Kapitel stelle ich Ihnen kurz die Sprachen vor, die wir in diesem Buch zur Entwicklung der Beispiel-Webanwendung verwenden.

3.2 HTML5

Begriffsbestimmung

Der Begriff »HTML5« kann recht verwirrend sein.

HTML5 beinhaltet die Spezifikation einer Markup-Sprache zur Entwicklung von Dokumenten, die im Webbrowser dargestellt werden sollen. Als Auszeichnungssprache erweitert HTML5 frühere Versionen von HTML und XHTML, vereinfacht sie gleichzeitig aber auch.

Als Erweiterung von HTML bietet HTML5 dem Webentwickler eine Reihe von neuen Tags. Viele davon sollen die Beschreibung des Inhalts in HTML verbessern und erleichtern. So gibt es in HTML5 beispielsweise `header`- und `footer`-Elemente. Nach Anwendung dieser Tags sehen die Seiten weder anders aus als zuvor, noch verhalten sie sich anders. Damit sind sie eine der weniger interessanten Neuerungen von HTML5. Wir werden uns in diesem Buch noch mit manchen dieser neuen Tags beschäftigen.

Audio- und Videounterstützung

Weiterhin enthält HTML5 neue Elemente zur Unterstützung von Audio und Video und einen Canvas zur Darstellung von 2D-Formen und Bitmaps. Genau diese Features von HTML5 haben viel Aufmerksamkeit bekommen, vor allem weil HTML5 dadurch zur direkten Konkurrenz von Adobe Flash geworden ist. Apple hat den Einsatz von Adobe Flash auf bestimmten Geräten unterbunden – stattdessen sollen die Websites die entsprechenden HTML5-Funktionen verwenden, da diese standardkonform sind und ohne Plugins auskommen. Diese Multimedia-Funktionalität von HTML5 bleibt in diesem Buch weitgehend außen vor, da sie in der Regel für Web-Anwendungen nicht relevant ist. Man sollte jedoch wissen, dass es sie gibt.

Strenge Regeln kontraproduktiv

Im Rahmen der Vereinfachung insbesondere von XHTML wurde erkannt, dass die strenge Umsetzung der HTML-Standards in früheren Versionen nicht nur unnötig war (nicht regelkonforme Seiten konnten im Browser dennoch angezeigt werden), sondern auch kontraproduktiv (weil es keinen Standard für den Umgang mit aus irgendwelchen Gründen ungültigen Seiten gab, kochte jeder Browseranbieter sein eigenes Süppchen). Die HTML5-Spezifikation liefert den Herstellern jedoch detaillierte Vorgaben, wie ihre Browser ein einheitliches Dokumentobjektmodell aus den vorhandenen Daten erzeugen sollten. In weiten Teilen definiert die HTML-Spezifikation genau diese Regeln – und auch diese würden den Rahmen des Buches sprengen.

Hinweis

Keine Sorge, wenn Sie nicht mit dem Document Object Model (DOM) vertraut sind – es wird noch erläutert. Auch mit XHTML müssen Sie sich nicht auskennen; es ist weitestgehend überholt.

Formularkomponenten

HTML5 verbessert auch die HTML-Formularkomponenten. Es liefert nicht nur neue Feldtypen (zum Beispiel zur Datums- und Farbwahl), sondern auch zusätzliche Attribute für bestehende Eingabefelder. HTML5 ermöglicht zudem die native Validierung von Formularkomponenten.

Standards für APIs

Neben der Markup-Sprache und verschiedenen Formularkomponenten bietet HTML5 eine Reihe von Standards für die APIs, die vom Webbrowser bereitgestellt werden. Diese APIs sind breit gefächert: Das Spektrum reicht von der Offline-Speicherung von Daten und Inhalten über das Lesen von Dateien bis hin zu Hintergrundprozessen, Server-Sent Events und vielem mehr. Genau diese Merkmale von HTML5 machen den Webbrowser erst zu einer echten Plattform für die Anwendungsentwicklung. Für die Beispiel-Webanwendung in diesem Buch verwenden wir viele der neuen APIs.

HTML5 immer noch Arbeitsentwurf

Der HTML5-Standardisierungsprozess ist schon an sich sehr interessant. Viele Standards orientieren sich an bereits bestehenden Browserfunktionen. So wurde beispielsweise die Technologie hinter AJAX (das XMLHttpRequest-Objekt) zunächst als proprietäre Funktion für den Internet Explorer entwickelt. Andere Browser bauten diese Funktion dann nach. Als sie in allen gängigen Browsern unterstützt wurde, wurde sie vom W3C standardisiert. Offiziell gilt HTML5 immer noch als Arbeitsentwurf.

Hinweis

Das World Wide Web Consortium (W3C) ist die Hauptstandardisierungsorganisation für das World Wide Web. Die Webseite finden Sie unter *http://www.w3.org*. Die HTML5-Spezifikation finden Sie unter *http://www.w3.org/TR/html5*.

W3C und WHATWG

Eigentlich wird die HTML5-Spezifikation von zwei verschiedenen Organisationen entwickelt, dem W3C und dem WHATWG. Beide bieten die Standards unter ihren eigenen Lizenzen an. Die Verdienste des WHATWG um HTML5 sind in Wirklichkeit sehr viel höher einzuschätzen als die des W3C. Das W3C wollte zunächst nicht auf HTML5 setzen, sondern XML-basierte Standards weiterentwickeln. Irgendwann erkannte das W3C, dass es aufs falsche Pferd gesetzt hatte und beteiligte sich aktiv an der Entwicklung von HTML5.

Erwähnenswert ist auch, dass die Versionen der Spezifikationen bei W3C und WHATWG nicht komplett übereinstimmen. Vom Standpunkt eines Anwendungsentwicklers aus ist dies weitgehend irrelevant. Unter Programmierern kursiert das Sprichwort: »Code gewinnt immer.« Dies gilt auch für HTML5: Die Spezifikation ist weitgehend uninteressant; es sind die Browser-Implementierungen, die zählen.

Web-SQL-API

Oft wird ein Standard von einem ganz bestimmten Browseranbieter vorangetrieben, was gelegentlich in eine Sackgasse führt. Ein Beispiel ist die Web-SQL-API. In anderen Fällen wird eine Spezifikation vorangetrieben, obwohl sie keine breite Unterstützung findet und daher eine ungewisse Zukunft hat (das gilt beispielsweise für die File-Writer- und die File-System-API). Im Idealfall unterstützen aber die gängigen Browser die API gemäß dem anerkannten Standard.

Lebendiger Standard

Wichtig ist bei HTML5 weiterhin, dass es sich um einen lebendigen Standard handelt. In den folgenden Kapiteln werden Sie sehen, dass im HTML5-Dokumententyp keine Version angegeben wird: Es handelt sich einfach um HTML, das sich gemeinsam mit dem Standard weiterentwickeln wird, und die Browser werden diese Weiterentwicklung übernehmen.

3.3 JavaScript

Native Unterstützung in fast allen Browsern

Die dynamischen und interaktiven Merkmale einer Webanwendung erfordern JavaScript. JavaScript ist die einzige Sprache, die von fast allen vorhandenen Browsern nativ unterstützt wird. Sie tauchte erstmals im Jahr 1995 in einer frühen Version des Browsers Netscape Navigators auf und wurde schnell vom Internet Explorer übernommen.

ECMAScript Version 5

Zunächst möchte ich eine Kleinigkeit hinsichtlich der Terminologie klären: Der Sprachkern von JavaScript wurde als ECMAScript standardisiert. Wenn es in diesem Buch um JavaScript geht, ist damit technisch ECMAScript Version 5 gemeint (die neueste Version – EC6 – befindet sich momentan noch im Entwicklungsstadium).

JavaScript wurde nach der Programmiersprache Java benannt. Dies zielte aber nicht in erster Linie auf eine Ähnlichkeit zwischen den beiden Sprachen ab, sondern man wollte auf den Bekanntheitsgrad von Java aufsetzen. In Wirklichkeit unterscheidet sich JavaScript ganz deutlich von Java, und zwar aus den folgenden Gründen:

Dynamische Typisierung

Anders als Java mit seiner statischen Typisierung unterstützt JavaScript die dynamische Typisierung. Das heißt, dass Sie in JavaScript eine Variable ohne Deklaration ihres Typs verwenden können. Der Typ wird erst zur Laufzeit ermittelt.

First-Class-Funktionen

JavaScript verfügt über First-Class-Funktionen: Es ist möglich, einer Variablen eine Funktion zuzuweisen oder sie als Parameter an eine andere Funktion zu übergeben. Das scheint keine große Sache zu sein – in Wirklichkeit eröffnen sich dem Software-Entwickler damit sehr viele Möglichkeiten und er kann Anwendungen mittels funktionaler Programmierung entwickeln. Diese Techniken werden im Detail in den folgenden Kapiteln erläutert.

Prototyping-Techniken

Zwar gibt es in JavaScript Klassen; die Umsetzung ist jedoch ein wenig verwirrend. Ich empfehle daher, Klassen so weit wie möglich zu vermeiden und Objekte mithilfe von Prototyping-Techniken zu erstellen.

Dieses Buch soll nicht als Handbuch zu allen Funktionen der JavaScript-Sprache dienen. Stattdessen zeigt es die grundlegenden Ansätze, die Sie als Software-Entwickler nutzen können.

Wenn Sie bisher noch keine Zeit hatten, sich mit JavaScript zu beschäftigen, und vor allem wenn Sie bisher nur mit statisch typisierten Sprachen programmiert haben, wird die Eleganz und Flexibilität der JavaScript-Sytax Sie wahrscheinlich beeindrucken.

3.4 jQuery

JavaScript-Bibliothek

jQuery ist eine JavaScript-Bibliothek, die die Entwicklung von JavaScript-Anwendungen im Webbrowser vereinfacht.

Wegen des dokumentenzentrierten Charakters von Webseiten wird JavaScript normalerweise für die Auswahl von Elementen im Dokument (die interne Darstellung eines Dokuments im Browser wird DOM, Document Object Model, genannt), für die Manipulation dieser Elemente oder für die Reaktion auf durch sie ausgelöste Ereignisse verwendet. Diese Merkmale unterstützt JavaScript durch die Document Object Model-API, die auch in der HTML5-Spezifikation enthalten ist. Im Wesentlichen stellt jQuery eine elegante Schnittstelle zur DOM-API dar.

Sizzle

Den Kern von jQuery bildet die Selektor-Engine »Sizzle«. jQuery empfängt Auswahlkriterien auf der Grundlage von CSS-Selektoren und liefert eine Reihe von Elementen aus dem Dokument zurück, die diese Kriterien erfüllen. Auf die so ausgewählten Elemente können Sie dann eine Vielzahl von Funktionen anwenden. So lassen sich verschiedene Operationen durchführen oder Event-Listener hinzufügen.

Obwohl die Möglichkeiten von jQuery die von JavaScript oder der nativen DOM-API keineswegs übersteigen, erfreut es sich aus verschiedenen Gründen enormer Beliebtheit:

Gründe für die Beliebtheit von jQuery

- Sie müssen sich nicht mehr mit den Macken der verschiedenen Browser herumärgern.
- jQuery bietet eine umfangreiche und prägnante Syntax, die von den meisten als eine große Verbesserung gegenüber der Document Object Model-API angesehen wird.
- Sie können sehr einfach benutzerdefinierte Plugins für jQuery schreiben und es so an bestimmte Anforderungen anpassen.
- Es gibt zahlreiche Open-Source-Plugins für jQuery, unter anderem ein beliebtes UI-Toolkit namens jQuery UI.

jQuery hat durchaus Konkurrenz, zum Beispiel von Dojo Toolkit und Prototype. Jedoch hat jQuery auf dem Markt eine kritische Masse erreicht und ist inzwischen beinahe zum Quasistandard für Webanwendungen geworden.

3.5 Cascading Style Sheets

Cascading Style Sheets (CSS) stellen eine Stylesheet-Sprache für HTML dar. Fast alle Gestaltungselemente, die in HTML noch aus der Zeit vor CSS verblieben waren, wurden aus HTML5 gestrichen. Die gesamte Präsentation sollte nun vollständig mit CSS erfolgen.

Trennung von Inhalt und Gestaltung

CSS ermöglicht die Trennung von Inhalt und Gestaltung. Beide können unabhängig voneinander geändert werden. Der Inhalt der Seite soll in HTML erzeugt werden, die Darstellung der Seite in CSS.

Das bedeutet auch, dass ein und derselbe Inhalt für verschiedene Geräte (z. B. Smartphones) wiederverwendet werden kann, indem einfach ein neues Stylesheet verwendet wird.

CSS enthält eine Reihe von Eigenschaften, die beschreiben, wie die Elemente in einem Dokument gestaltet werden sollen, sobald bestimmte Regeln darauf zutreffen, und wie diese Eigenschaften miteinander in Wechselwirkung treten sollen. Die Stile, die auf Elemente angewandt werden können, sind unbeschreiblich vielfältig und wurden in CSS3 noch erheblich erweitert. Die zuletzt genannte Spezifikation läuft größtenteils parallel zu HTML5.

Als Software-Entwickler braucht man normalerweise nicht alle CSS-Funktionen bis ins Detail zu kennen. Wichtig ist es jedoch, die Grundlagen zu verstehen – sonst drohen Frustration und Ärger. Genau wie die übrigen Sprachen in diesem Buch ist CSS nicht so einfach, wie es scheinen mag. Wer sich mit der Entwicklung von Webanwendungen beschäftigt, braucht zwingend solide Grundlagenkenntnisse.

Bei der Entwicklung der Beispiel-Webanwendung in diesem Buch werde ich CSS weitgehend ignorieren. Anhang A enthält jedoch eine ausführliche Einführung – schlagen Sie dort einfach bei Bedarf nach.

2 Über dieses Buch

2.1 Was Sie brauchen

Für dieses Buch sollten Sie etwas Erfahrung als Softwareentwickler und möglichst auch einige HTML-Grundkenntnisse mitbringen.

Keine Schritt-für-Schritt-Anleitungen

Sie finden hier keine Schritt-für-Schritt-Anleitungen zu den Grundlagen von HTML oder JavaScript. Auch ohne Vorkenntnisse in diesen Sprachen werden Sie beim Schreiben Ihrer eigenen Webanwendung das wesentliche Basiswissen erlangen. Natürlich geschieht das dann nicht mehr unbedingt in der Reihenfolge einer klassischen Einführung.

Nützliche Ressourcen

Wenn Sie zuvor noch nicht mit HTML oder JavaScript gearbeitet haben, könnte sich ein Blick auf die Grundsyntax dieser Sprachen (etwa Schleifen, bedingte Anweisungen usw.) lohnen, bevor Sie weitermachen. Websites wie *https://developer.mozilla.org/de/* oder *https://docs.webplatform.org/* bieten Einführungen in die Grundlagen von HTML und JavaScript.

Benötigte Werkzeuge

Die Übungen in diesem Buch können Sie auf jedem Computer durchführen, der Ihnen Zugriff auf Folgendes gewährt:

- **Einen Texteditor zum Schreiben von Code.** Notepad++ (*http://notepad-plus-plus.org*) ist eine gute Wahl für Windows, für Mac-Rechner empfiehlt sich zum Beispiel Text Wrangler (*http://www.barebones.com/products/textwrangler*). Für Linux-Systeme gibt es Emacs. Sie können natürlich auch eine Integrierte Entwicklungsumgebung (IDE) wie Eclipse nutzen.
- **Chrome- oder Firefox-Webbrowser.** Wenn Sie Firefox wählen, müssen Sie für eine vollständige Sammlung von Entwicklerwerkzeugen außerdem das Firebug-Plugin installieren. Die Beispiele und Screenshots basieren auf Chrome; alles Gezeigte funktioniert aber auch mit Firefox, Internet Explorer 10 oder Safari. Aus meiner Sicht bietet Chrome inzwischen unter allen Browsern die besten Entwicklerwerkzeuge. Daher empfehle ich jedem Neueinsteiger unbedingt diesen Browser.

- **Einen Webserver.** Diesen brauchen Sie erst im späteren Verlauf des Buchs. Dort finden Sie auch ein Kapitel, das die Installation und die Verwendung des Mongoose-Webservers erläutert. Es steht Ihnen frei, auch einen anderen Webserver zu verwenden, beschrieben wird hier aber nur Mongoose.

Beispiele zum Download

Alle Beispiele im Buch finden Sie auf dieser Website:
http://www.dpunkt.de/leseproben/5160/Beispielcode.zip

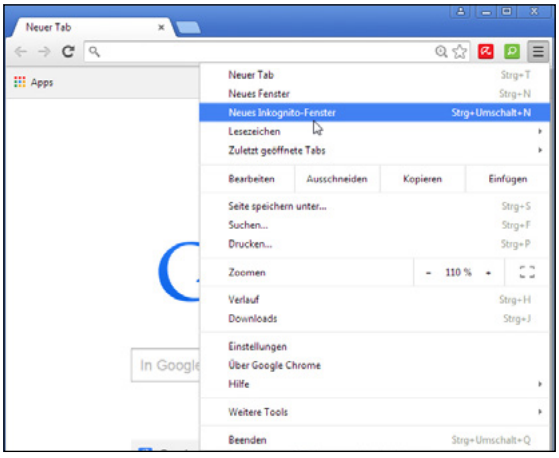
Für jedes Buchkapitel (mit entsprechenden Beispielen) finden Sie eine ZIP-Datei mit den Ressourcen für die Webanwendung, so wie sie am Ende des Kapitels abgedruckt sind.

Wie bereits erwähnt, führt Sie das Buch durch den Erstellungsprozess einer Webanwendung. In der ersten Hälfte können Sie die Webanwendung direkt von Ihrem lokalen Dateisystem in den Browser laden. Ein Webserver ist nicht unbedingt erforderlich.

Alle Webbrowser können HTML-Dateien direkt vom Dateisystem anzeigen: Ziehen Sie die HTML-Datei dazu einfach in Ihren Browser oder verwenden Sie die Menüoption *Datei > Datei öffnen*.

Inkognito-Modus

Leider nehmen die Browser Ressourcen wie JavaScript- und CSS-Dateien häufig in den Cache-Speicher auf, wenn eine Website direkt vom lokalen Dateisystem geladen wird. In Chrome können Sie das umgehen, indem Sie ein neues Fenster im Inkognito-Modus öffnen:



In diesem Modus werden die geladenen Ressourcen nicht in den Browsercache geladen.

Cache deaktivieren

In Chrome können Sie bei geöffneten Entwicklungswerkzeugen den Cache ganz einfach deaktivieren: Klicken Sie dazu einfach auf das Zahnradsymbol und wählen Sie *Disable cache (while DevTools is open)*.

lokaler Webserver

Wenn Ihnen das zu umständlich ist, können Sie Ihre Seiten auch gleich auf einen lokalen Webserver legen. In Kapitel 8 finden Sie eine detaillierte Anleitung zur Installation des Mongoose-Webservers auf Ihrem Computer. Von einem Webserver bezogene Seiten werden teilweise trotzdem noch im Browsercache abgelegt. Es gibt jedoch eine Funktion, mit der Sie die aktuelle Seite neu laden und dabei den Cache überschreiben können. Dann werden die Ressourcen garantiert neu geladen:

Unter OS X drücken Sie dazu die Tastenkombination ⌘+⇧+R, unter Windows Strg+F5.

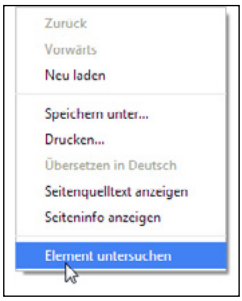
Entwicklerwerkzeuge im Browser

Viele Beispiele in diesem Buch, besonders die aus den vorderen Kapiteln, können direkt in einem JavaScript-Interpreter ausgeführt werden. Alle wichtigen Browser bieten inzwischen Entwicklerwerkzeuge, zu denen auch ein JavaScript-Interpreter gehört.

In Chrome verwenden Sie im geöffneten Browserfenster einfach die folgende Tastenkombination, um zum Interpreter zu gelangen:

- ⌘+⌥+I unter OS X
- F12 oder Strg+⇧+I unter Windows

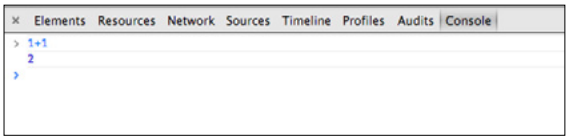
Alternativ können Sie auch irgendwo auf der Webseite einen Rechtsklick ausführen und dann im Kontextmenü *Element untersuchen* auswählen:



Konsole öffnen

Sobald die Entwicklerwerkzeuge angezeigt werden, klicken Sie auf das Register *Console* bzw. *Konsole*.

Zum Beweis, dass es sich um einen echten JavaScript-Interpreter handelt, geben Sie an der Eingabeaufforderung einfach 1+1 ein und drücken Sie die ↵-Taste:



Ein Aspekt der Browser hat in den letzten Jahren beträchtliche Fortschritte gemacht: die mitgelieferten Entwicklerwerkzeuge. Die Hersteller haben erkannt, dass sie einen direkten Nutzen daraus ziehen, wenn die Entwickler ihren Browser verwenden. Darum locken sie die Entwickler mit den angebotenen Werkzeugen inzwischen aktiv an. In diesem Buch erhalten Sie eine Einführung in viele Funktionen der Chrome-Entwicklerwerkzeuge. Es lohnt sich aber, die gebotenen Möglichkeiten selbst auszukundschaften.

2.2 Konventionen

Code wird in diesem Buch in Proportionalschrift dargestellt:

```
Dies ist ein Code
```

Code im JavaScript-Interpreter

Wird Code im JavaScript-Interpreter ausgeführt, steht vor der Eingabe ein >-Zeichen. Wenn Sie die Befehle selbst eintippen, lassen Sie dieses Zeichen weg. Außerdem trennt eine Leerzeile Eingabe und Ausgabe voneinander:

```
> 1 + 1

2
```

Leerzeilen

Werden zwei Befehle gleichzeitig gezeigt, trenne ich sie durch eine weitere Leerzeile zwischen der Ausgabe des ersten Befehls und der Eingabe des zweiten Befehls.

```
> 1 + 1

2

> 2 + 2

4
```

Ist die Ausgabe zum Verständnis des Erklärten unwichtig, habe ich sie manchmal weggelassen.

2.3 Rückmeldungen

Ich möchte sehr gerne Ihre Meinungen (positiv und negativ) zu diesem Buch hören. Bitte schreiben Sie mir eine E-Mail an dane@cisdal.com.