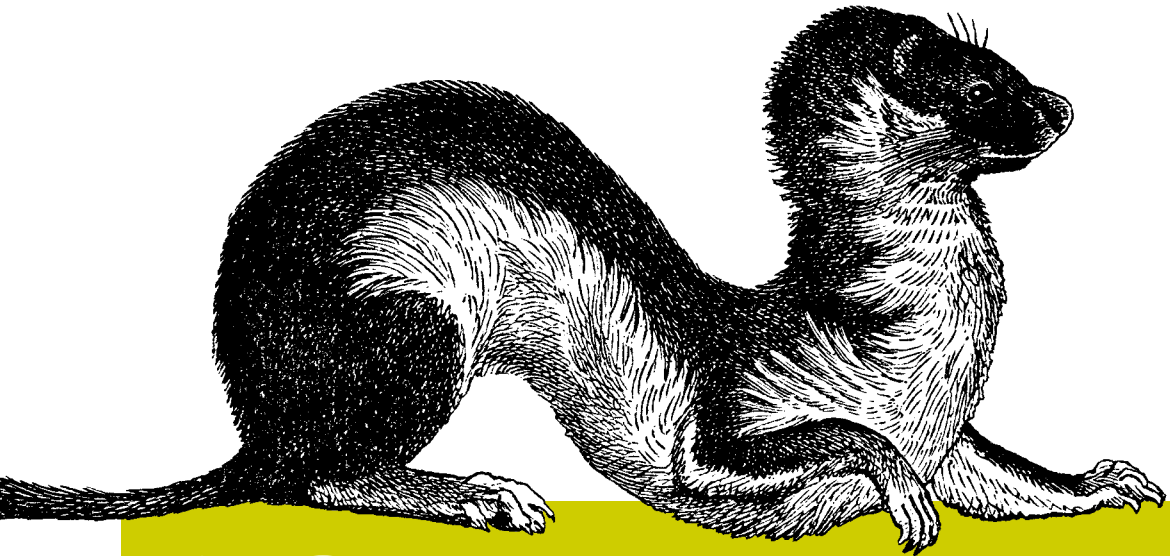


*Lösungen für jQuery-Entwickler*



# jQuery Kochbuch

**O'REILLY®**

*jQuery Community Experten*  
*Deutsche Übersetzung von Thomas Demmig*

<b>Vorwort</b> .....	<b>XIII</b>
<b>Beteiligte</b> .....	<b>XV</b>
<b>Einleitung</b> .....	<b>XIX</b>
<b>1 Grundlagen von jQuery</b> .....	<b>1</b>
1.1 Einbinden der jQuery-Bibliothek in eine HTML-Seite .....	9
1.2 Ausführen von jQuery/JavaScript-Code nach dem Laden des DOM, aber noch vor dem vollständigen Laden der Seite .....	11
1.3 Selektieren von DOM-Elementen mit Selektoren und der jQuery-Funktion	13
1.4 Selektieren von DOM-Elementen in einem bestimmten Kontext .....	15
1.5 Ein Wrapper-Set mit DOM-Elementen filtern .....	16
1.6 Abhängige Elemente im aktuell selektierten Wrapper-Set finden .....	18
1.7 Vor einer destruktiven Änderung zur vorherigen Selektion zurückkehren	19
1.8 Die vorherige Selektion mit der aktuellen Selektion vereinigen .....	21
1.9 Das DOM basierend auf dem aktuellen Kontext durchlaufen, um ein neues Set mit DOM-Elementen zu erhalten .....	22
1.10 DOM-Elemente erstellen, bearbeiten und einfügen .....	23
1.11 Entfernen von DOM-Elementen .....	25
1.12 DOM-Elemente ersetzen .....	26
1.13 DOM-Elemente klonen .....	27
1.14 Attribute von DOM-Elementen lesen, setzen und entfernen .....	30
1.15 HTML-Inhalte lesen und setzen .....	31
1.16 Text-Inhalte lesen und setzen .....	32
1.17 Den \$-Alias verwenden, ohne globale Konflikte zu erzeugen .....	33

<b>2</b>	<b>Elemente mit jQuery selektieren</b> . . . . .	<b>35</b>
2.1	Nur Kind-Elemente selektieren. . . . .	36
2.2	Bestimmte Geschwister-Elemente selektieren. . . . .	38
2.3	Elemente über die Index-Reihenfolge selektieren . . . . .	39
2.4	Aktuell animierte Elemente selektieren . . . . .	41
2.5	Elemente anhand ihres Inhalts selektieren . . . . .	42
2.6	Elemente über eine negative Selektion selektieren . . . . .	43
2.7	Elemente anhand ihrer Sichtbarkeit selektieren . . . . .	44
2.8	Elemente anhand von Attributen selektieren . . . . .	45
2.9	Form-Elemente anhand des Typs selektieren . . . . .	46
2.10	Elemente mit bestimmten Eigenschaften selektieren. . . . .	47
2.11	Den Kontext-Parameter verwenden . . . . .	49
2.12	Einen eigenen Filter-Selektor erstellen. . . . .	50
<b>3</b>	<b>Fortgeschrittene Techniken</b> . . . . .	<b>53</b>
3.1	Ein Set mit selektierten Ergebnissen durchlaufen . . . . .	53
3.2	Das Selektions-Set auf ein bestimmtes Element reduzieren . . . . .	56
3.3	Ein selektiertes jQuery-Objekt in ein reines DOM-Objekt konvertieren . . . . .	59
3.4	Den Index eines Elements in einer Selektion ermitteln . . . . .	62
3.5	Aus einem bestehenden Array ein Array mit ausgewählten Werten erstellen . . . . .	64
3.6	Eine Aktion auf einer Untermenge des selektierten Sets ausführen. . . . .	66
3.7	Konfigurieren von jQuery, so dass es nicht mit anderen Bibliotheken kollidiert. . . . .	69
3.8	Funktionalität durch Plugins hinzufügen. . . . .	71
3.9	Die verwendete Selektion bestimmen . . . . .	74
<b>4</b>	<b>jQuery-Tools</b> . . . . .	<b>77</b>
4.1	Features mit jQuery.support erkennen . . . . .	77
4.2	Mit jQuery.each über Arrays und Objekte iterieren . . . . .	79
4.3	Arrays mit jQuery.grep filtern . . . . .	80
4.4	Über Array-Elemente mit jQuery.map iterieren und sie verändern. . . . .	81
4.5	Zwei Arrays durch jQuery.merge kombinieren . . . . .	81
4.6	Doppelte Array-Einträge mit jQuery.unique ausfiltern . . . . .	82
4.7	Callback-Funktionen mit jQuery.isFunction testen . . . . .	83
4.8	Whitespace aus Strings oder Form-Werten mit jQuery.trim entfernen. . . . .	84
4.9	Objekte und Daten per jQuery.data an DOM-Elemente anhängen . . . . .	84
4.10	Objekte durch jQuery.extend erweitern . . . . .	86

<b>5</b>	<b>Schneller, Einfacher, Spaßiger</b>	<b>89</b>
5.1	Das ist nicht jQuery, sondern JavaScript!	89
5.2	Was ist an \$(this) falsch?	90
5.3	Überflüssige Wiederholungen vermeiden	93
5.4	Ihre verketteten jQuery-Methoden formatieren.	95
5.5	Code aus anderen Bibliotheken übernehmen	96
5.6	Einen eigenen Iterator schreiben	99
5.7	Ein Attribut umschalten	101
5.8	Performance-Killer finden	103
5.9	Ihre jQuery-Objekte puffern	108
5.10	Schnellere Selektoren schreiben.	109
5.11	Tabellen schneller laden	111
5.12	Schleifen programmieren	114
5.13	Name Lookups verringern	117
5.14	Das DOM mit .innerHTML schneller aktualisieren.	120
5.15	Debuggen? Sprengen Sie die Ketten	121
5.16	Ist das ein Bug von jQuery?.	123
5.17	In jQuery debuggen	124
5.18	Weniger Server-Anfragen erzeugen	126
5.19	Zurückhaltendes JavaScript schreiben	129
5.20	jQuery für die progressive Verbesserung nutzen	131
5.21	Machen Sie Ihre Seiten barrierefrei.	133
<b>6</b>	<b>Dimensionen</b>	<b>137</b>
6.1	Die Dimensionen von Window und Document ermitteln	137
6.2	Ermitteln der Dimensionen eines Elements.	138
6.3	Den Offset eines Elements ermitteln	141
6.4	Ein Element in den sichtbaren Bereich scrollen.	143
6.5	Ermitteln, ob sich ein Element im sichtbaren Bereich befindet	145
6.6	Zentrieren eines Elements im sichtbaren Bereich.	148
6.7	Absolute Positionierung eines Elements an seiner aktuellen Position	149
6.8	Ein Element relativ zu einem anderen Element positionieren	149
6.9	Stylesheets abhängig von der Browser-Breite wechseln	151
<b>7</b>	<b>Effekte</b>	<b>153</b>
7.1	Elemente per Sliding und Fading ein- und ausblenden	155
7.2	Elemente durch ein Sliding Up anzeigen.	158
7.3	Erzeugen eines horizontalen Akkordeons	160
7.4	Elemente gleichzeitig Sliden und Faden	163

7.5	Sequenzielle Effekte anwenden . . . . .	164
7.6	Erkennen, ob Elemente aktuell animiert werden . . . . .	166
7.7	Animationen stoppen und zurücksetzen . . . . .	167
7.8	Eigene Easing-Methoden für Effekte nutzen . . . . .	169
7.9	Alle Effekte deaktivieren . . . . .	170
7.10	Aufwändigere Effekte mit jQuery UI erzeugen . . . . .	171
<b>8</b>	<b>Events . . . . .</b>	<b>173</b>
8.1	Einen Handler mit vielen Events verbinden . . . . .	174
8.2	Eine Handler-Funktion mit anderen Daten wiederverwenden . . . . .	175
8.3	Ein ganzes Set mit Eventhandlern entfernen . . . . .	177
8.4	Eventhandler auslösen . . . . .	178
8.5	Dynamische Daten an Eventhandler übergeben . . . . .	179
8.6	Sofortiger Zugriff auf ein Element (noch vor document.ready) . . . . .	182
8.7	Die Ausführungsschleife für Handler stoppen . . . . .	185
8.8	Beim Verwenden von event.target das richtige Element erhalten . . . . .	187
8.9	Mehrere parallele hover()-Animationen vermeiden . . . . .	189
8.10	Eventhandler für neu hinzugefügte Elemente nutzbar machen . . . . .	191
<b>9</b>	<b>Events für Fortgeschrittene . . . . .</b>	<b>195</b>
9.1	jQuery nutzen, wenn es dynamisch geladen wird . . . . .	195
9.2	Das globale Auslösen von Events beschleunigen . . . . .	196
9.3	Eigene Events erstellen . . . . .	199
9.4	Eventhandler stellen benötigte Daten bereit . . . . .	202
9.5	Event-gesteuerte Plugins erstellen . . . . .	205
9.6	Benachrichtigungen erhalten, wenn jQuery-Methoden aufgerufen werden . . . . .	209
9.7	Objekt-Methoden als Event Listener nutzen . . . . .	213
<b>10</b>	<b>HTML-Forms durch eigenen Code verbessern . . . . .</b>	<b>217</b>
10.1	Ein Texteingabefeld beim Laden der Seite fokussieren . . . . .	218
10.2	Form-Elemente aktivieren und deaktivieren . . . . .	219
10.3	Automatisch Radio Buttons auswählen . . . . .	222
10.4	(De)selektieren aller Checkboxes durch Links . . . . .	224
10.5	(De)selektieren aller Checkboxes über einen einzelnen Umschalter . . . . .	225
10.6	Auswahl-Optionen hinzufügen und entfernen . . . . .	227
10.7	Abhängig von der Anzahl der Zeichen ins nächste Feld springen . . . . .	228
10.8	Anzahl der verbleibenden Zeichen anzeigen . . . . .	230
10.9	Texteingabefelder auf bestimmte Zeichen beschränken . . . . .	233

10.10	Eine Form mit Ajax abschicken . . . . .	234
10.11	Forms überprüfen . . . . .	236
<b>11</b>	<b>Verbesserungen von HTML-Forms durch Plugins . . . . .</b>	<b>243</b>
11.1	Forms überprüfen . . . . .	244
11.2	Eingabemasken für Felder erstellen . . . . .	254
11.3	Textfelder automatisch vervollständigen . . . . .	256
11.4	Einen Wertebereich selektieren . . . . .	257
11.5	Einen Wert eingeben, der innerhalb bestimmter Grenzen liegt. . . . .	260
11.6	Dateien im Hintergrund hochladen . . . . .	262
11.7	Die Länge von Texteingabefeldern begrenzen. . . . .	264
11.8	Label oberhalb von Eingabefeldern anzeigen . . . . .	265
11.9	Ein Eingabeelement mit seinem Inhalt wachsen lassen . . . . .	266
11.10	Ein Datum wählen . . . . .	267
<b>12</b>	<b>jQuery-Plugins . . . . .</b>	<b>271</b>
12.1	Wo finden Sie jQuery-Plugins? . . . . .	271
12.2	Wann sollten Sie ein jQuery-Plugin schreiben? . . . . .	273
12.3	Schreiben Sie Ihr erstes jQuery-Plugin . . . . .	275
12.4	Ihrem Plugin Optionen mitgeben . . . . .	277
12.5	Die Kurzform \$ in Ihrem Plugin verwenden . . . . .	278
12.6	Private Funktionen in Ihr Plugin aufnehmen . . . . .	280
12.7	Das Metadata-Plugin unterstützen. . . . .	282
12.8	Ihrem Plugin eine statische Funktion hinzufügen . . . . .	284
12.9	Unit Tests für Ihr Plugin mit QUnit . . . . .	286
<b>13</b>	<b>Selbst geschriebene Benutzerschnittstellen . . . . .</b>	<b>289</b>
13.1	Eigene Tooltips erstellen. . . . .	291
13.2	In einem Baum navigieren. . . . .	295
13.3	Ein Akkordeon aufziehen . . . . .	298
13.4	Registerkarten in einem Dokument . . . . .	303
13.5	Ein einfaches modales Fenster anzeigen . . . . .	306
13.6	Dropdown-Menüs erstellen. . . . .	313
13.7	Bilder zyklisch einblenden. . . . .	315
13.8	Sliding Panels. . . . .	320
<b>14</b>	<b>Benutzerschnittstellen mit jQuery UI . . . . .</b>	<b>325</b>
14.1	Die komplette jQuery UI-Suite einbinden. . . . .	327
14.2	Ein oder zwei einzelne jQuery UI-Plugins einbinden . . . . .	328

14.3	Ein jQuery UI-Plugin mit den Standard-Optionen initialisieren. . . . .	329
14.4	Ein jQuery UI-Plugin mit eigenen Optionen initialisieren . . . . .	330
14.5	Eigene jQuery UI-Plugin-Standardwerte erstellen. . . . .	331
14.6	Optionen für jQuery UI-Plugins lesen und setzen . . . . .	333
14.7	Plugin-Methoden von jQuery UI aufrufen . . . . .	334
14.8	Mit Events von jQuery UI-Plugins umgehen . . . . .	335
14.9	Ein jQuery UI-Plugin zerstören. . . . .	337
14.10	Einen Musikplayer mit jQuery UI erstellen . . . . .	337
<b>15</b>	<b>Themes in jQuery UI . . . . .</b>	<b>351</b>
15.1	Themes für jQuery UI-Widgets mit ThemeRoller erstellen . . . . .	356
15.2	Layout und Theme-Styles von jQuery UI überschreiben . . . . .	370
15.3	Ein Theme auf Komponenten anwenden, die nicht zum jQuery UI gehören	381
15.4	Mehrere Themes auf einer einzelnen Seite verwenden . . . . .	391
15.5	Anhang: Weitere Informationen zu CSS . . . . .	400
<b>16</b>	<b>jQuery, Ajax, Datenformate: HTML, XML, JSON, JSONP . . . . .</b>	<b>401</b>
16.1	jQuery und Ajax . . . . .	401
16.2	Ajax auf der gesamten Site verwenden . . . . .	404
16.3	Einfache Ajax-Anwendung mit Rückmeldungen an den Benutzer . . . . .	406
16.4	Ajax-Hilfsfunktionen und Datentypen . . . . .	410
16.5	HTML-Fragmente mit jQuery einsetzen . . . . .	412
16.6	XML-Code in ein DOM konvertieren . . . . .	413
16.7	JSON erzeugen . . . . .	414
16.8	JSON parsen. . . . .	415
16.9	jQuery und JSONP verwenden. . . . .	416
<b>17</b>	<b>jQuery in großen Projekten verwenden . . . . .</b>	<b>419</b>
17.1	Auf dem Client speichern. . . . .	419
17.2	Den Anwendungs-Status für eine einzelne Session speichern . . . . .	422
17.3	Den Anwendungs-Status über eine Session hinaus speichern . . . . .	424
17.4	Eine JavaScript Template Engine nutzen . . . . .	425
17.5	Ajax-Anfragen queuen. . . . .	428
17.6	Ajax und der Zurück-Button . . . . .	430
17.7	JavaScript am Seitenende unterbringen . . . . .	432
<b>18</b>	<b>Unit Tests . . . . .</b>	<b>435</b>
18.1	Unit Tests automatisieren . . . . .	435
18.2	Ergebnisse sicherstellen . . . . .	437

18.3	Synchrone Callbacks testen. . . . .	438
18.4	Asynchrone Callbacks testen. . . . .	439
18.5	Benutzeraktionen testen . . . . .	440
18.6	Tests atomar halten . . . . .	441
18.7	Tests gruppieren . . . . .	443
18.8	Durchzuführende Tests auswählen . . . . .	444
<b>Index</b>	. . . . .	<b>446</b>



---

# Grundlagen von jQuery

*Cody Lindley*

## 1.0 Einleitung

Da Sie zu einem Kochbuch über jQuery gegriffen haben, gehen die Autoren dieses Buches davon aus, dass Sie eine vage Idee davon haben, was jQuery genau ist und was es tut. Kochbücher sind im Allgemeinen für ein Publikum geschrieben, das schon gewisse Grundlagen im Bereich des betreffenden Themas besitzt, die ausgebaut werden sollen. Daher wird das Format Rezept – Lösung – Diskussion dazu genutzt, Ihnen schnell Lösungen für häufiger vorkommende Probleme anbieten zu können. Sind Sie aber ein jQuery-Anfänger, dann werfen Sie das Buch nicht gleich wütend gegen eine Wand, während Sie wilde Verwünschungen gegen uns aussprechen. Wir haben dieses Kapitel extra für Sie geschrieben.

Falls Sie ein wenig Auffrischung oder überhaupt eine Starthilfe benötigen, weil Sie über jQuery nicht viel oder gar nichts wissen, vermittelt Ihnen dieses erste Kapitel (die weiteren Kapitel gehen davon aus, dass Sie die Grundlagen beherrschen) einen Einstieg in jQuery. Haben Sie noch so gar kein Wissen über JavaScript und das DOM, dann sollten Sie sich vielleicht überlegen, ob es überhaupt sinnvoll ist, mit jQuery arbeiten zu wollen, *ohne* die Sprache JavaScript und ihr Zusammenwirken mit dem DOM zumindest in Grundzügen zu kennen. In diesem Fall empfehle ich ein Buch über DOM und JavaScript, bevor Sie sich wieder mit jQuery beschäftigen. Mein Tipp ist da *JavaScript - Das umfassende Referenzwerk* (<http://www.oreilly.de/catalog/jscript5ger>) von David Flanagan (O'Reilly) als Einführung. Aber lassen Sie sich nicht davon abhalten, jQuery noch vor dem DOM und JavaScript zu erlernen. Viele haben sich gerade auf diesem Weg Wissen über diese Technologien angeeignet. Es mag zwar nicht der ideale Weg sein, aber er kann funktionieren.

So, dann wollen wir uns mal eine formale Definition von jQuery und eine kurze Beschreibung ihrer Funktionalität anschauen:

jQuery ist eine Open Source JavaScript-Bibliothek, die die Interaktionen zwischen einem HTML-Dokument oder genauer dem Document Object Model (DOM) und JavaScript vereinfacht.

Oder einfacher gesagt – und für die klassischen JavaScript-Hacker – jQuery macht die Arbeit mit Dynamic HTML (DHTML) total einfach. Insbesondere erleichtert jQuery das Durchlaufen und Bearbeiten des HTML-Dokuments, den Umgang mit Browser-Events, DOM-Animationen, Ajax-Interaktionen und JavaScript-Entwicklung für verschiedene Browser.

Mit der formalen Beschreibung von jQuery im Hinterkopf wollen wir uns nun anschauen, warum Sie jQuery eventuell nutzen sollten.

## Warum jQuery?

Es mag ein bisschen seltsam erscheinen, die Vorzüge von jQuery in diesem Kochbuch aufzuzählen, denn vermutlich kennen Sie sie bereits.

Ich renne also vielleicht offene Türen ein, aber trotzdem sollten wir uns kurz anschauen, warum sich ein Entwickler für jQuery entscheiden könnte. Ich möchte damit auch Ihr Grundlagenwissen über jQuery verbessern, indem ich erst das »Warum« erkläre, um dann zum »Wie« zu kommen.

Ich will jQuery nicht mit der Konkurrenz vergleichen, um die Bedeutung von jQuery zu unterstreichen. Denn ich glaube nicht, dass es einen direkten Konkurrenten gibt. Zudem bin ich der Meinung, dass die einzige Bibliothek, die sowohl die Bedürfnisse der Designer als auch die der Programmierer erfüllt, jQuery ist. In diesem Kontext steht jQuery einfach für sich alleine.

Ich gehe fest davon aus, dass all die bekannten JavaScript-Bibliotheken und -Frameworks ihre Nische und ihren Wert haben. Es wäre verrückt, alle miteinander vergleichen zu wollen, aber es wird doch immer wieder gemacht. Ich war sogar selbst dran beteiligt. Aber nachdem ich mir viele Gedanken dazu gemacht habe, glaube ich ehrlich daran, dass alle JavaScript-Bibliotheken ihren Zweck haben. Es hängt nur mehr davon ab, wer sie nutzt und wie sie genutzt werden, als davon, was sie wirklich können. Zudem scheint es so, dass die kleinen Unterschiede zwischen JavaScript-Bibliotheken in Bezug auf die größeren Ziele der JavaScript-Entwicklung oft unwichtig sind. Ohne jetzt noch philosophischer zu werden, will ich einfach mal eine Liste von Eigenschaften präsentieren, die jQuery zum Vorteil gereichen:

- Es ist Open Source und das Projekt ist unter einer MIT- und einer GNU General Public License (GPL)-Lizenz lizenziert. Es ist frei, und das sogar mehrfach!
- Es ist klein (18 KB minifiziert) und gezippt (114 KB unkomprimiert).
- Es ist unglaublich beliebt, hat also viele Anwender und ausreichend Beitragende als Entwickler und Botschafter.
- Es normalisiert die Unterschiede zwischen Web-Browsern, so dass Sie es nicht tun müssen.
- Es hat einen kleinen Footprint mit einer einfachen, aber doch klugen Plugin-Architektur.
- Es gibt ein umfangreiches Repository mit Plugins (<http://plugins.jquery.com/>), das seit der Veröffentlichung von jQuery immer weiter gewachsen ist.

- Die API ist vollständig dokumentiert – sogar mit Code-Beispielen, was in der Welt der JavaScript-Bibliotheken ein echter Luxus ist. Nun, eigentlich war es jahrelang schon ein Luxus, dass es überhaupt eine Dokumentation gibt.
- Es ist freundlich, denn es stellt Möglichkeiten bereit, Konflikte mit anderen JavaScript-Bibliotheken zu vermeiden.
- Die Unterstützung durch die Community ist wirklich hilfreich. Es gibt eine Reihe von Mailing-Listen, IRC-Channels und unglaublich viele Tutorials, Artikel und Blog-Posts.
- Es wird offen entwickelt. Jeder kann zu Bugfixes, Verbesserungen und der Entwicklung an sich beitragen.
- Die Entwicklung geht stetig weiter, das Entwicklungs-Team hat keine Scheu, neue Releases zu veröffentlichen.
- Die Nutzung durch große Firmen sorgt für Langlebigkeit und Stabilität (zum Beispiel Microsoft, Dell, Bank of America, Digg, CBS, Netflix).
- Es berücksichtigt Spezifikationen des W3C, bevor es die Browser tun. Zum Beispiel unterstützt jQuery einen Großteil der CSS3-Selektoren.
- Es ist für die Entwicklung auf modernen Browsern getestet und optimiert (Chrome 1, Chrome Nightly, IE 6, IE 7, IE 8, Opera 9.6, Safari 3.2, WebKit Nightly, Firefox 2, Firefox 3, Firefox Nightly).
- Sowohl Designer als auch Programmierer können sehr viel aus jQuery herausholen.
- Seine Eleganz, die Methodologien und die Philosophie beim Ändern der Art, wie in JavaScript geschrieben wird, werden langsam selbst ein Standard. Überlegen Sie nur einmal, wie viele andere Lösungen die Selektions- und Auswahl-Muster übernommen haben.
- Der nicht erklärliche Nebeneffekt des guten Gefühls beim Programmieren ist ansteckend und lässt sich auch nicht vermeiden. Selbst Kritiker finden Teile von jQuery sehr gut.
- Die Dokumentation gibt es in unterschiedlichsten Facetten (zum Beispiel durch den API Browser, Dashboard Apps und Cheat Sheets), einschließlich eines Offline-API-Browsers (AIR Application).
- Es ist absichtlich so gestaltet, dass ordentliche JavaScript-Codingpraktiken gefördert werden.
- Es ist im Kern eine JavaScript-Bibliothek geblieben (statt eines Frameworks), stellt aber gleichzeitig ein Partner-Projekt für UI-Widgets und die Anwendungs-Entwicklung bereit (jQuery UI).
- Die Lernkurve ist nicht allzu steil, weil jQuery auf Konzepten aufbaut, die die meisten Entwickler und Designer schon kennen (zum Beispiel CSS und HTML).

Meiner Meinung nach ist es keine einzelne Eigenschaft, sondern die Kombination aller dieser Punkte, die jQuery von den anderen Lösungen abhebt, Das gesamte jQuery-Paket ist als JavaScript-Tool einfach unerreich.

## Die Philosophie hinter jQuery

Die Philosophie von jQuery ist: »Schreibe weniger, schaffe mehr.« Diese Philosophie kann dabei in drei Konzepte aufgebrochen werden:

- Elemente finden (über CSS-Selektoren), mit denen dann etwas getan wird (über jQuery-Methoden)
- Mehrere jQuery-Methoden für ein Set aus Elementen verketteten
- Den jQuery-Wrapper und die implizite Iteration nutzen

Wenn Sie Ihren eigenen jQuery-Code schreiben oder die in diesem Buch gefundenen Rezepte aufpeppen wollen, ist es unabdingbar, diese drei Konzepte im Detail verstanden zu haben. Lassen Sie uns diese unter die Lupe nehmen.

### Elemente finden, mit denen dann etwas getan wird

Genauer gesagt – eine Menge von Elementen im DOM finden, mit der dann etwas getan wird. Lassen Sie uns zum Beispiel ein Szenario anschauen, bei dem Sie ein `<div>` vor dem Anwender verbergen, neue Inhalte dort hineinladen, ein Attribut des `<div>` ändern und schließlich das verborgene `<div>` wieder sichtbar machen wollen.

Diese Kombination an Aufgaben, umgewandelt in jQuery-Code, würde in etwa so aussehen:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
<head>
<script type="text/JavaScript"
src="http://ajax.googleapis.com/ajax/libs/jquery/1.3.2/jquery.min.js"></script>
</head>
<body>
<div>alter Inhalt</div>
<script>

//alle divs auf der Seite verbergen
jQuery('div').hide();

//Text in allen divs aktualisieren
jQuery('div').text('neuer Inhalt');

//Klassen-Attribut mit dem Wert updatedContent bei allen divs hinzufügen
jQuery('div').addClass("updatedContent");

//alle divs auf der Seite anzeigen
jQuery('div').show();

</script>
</body>
</html>
```

Lassen Sie uns diese vier jQuery-Anweisungen durchgehen:

- Die `<div>`-Elemente auf der Seite verbergen, so dass sie der Anwender nicht mehr sieht.
- Den Text in den verborgenen `<div>` durch neuen Text (neuer Inhalt) ersetzen.
- Das `<div>`-Element durch ein neues Attribut (`class`) mit einem Wert (`updatedContent`) ergänzen.
- Das `<div>`-Elemente auf der Seite anzeigen, so dass es der Benutzer wieder zu sehen bekommt.

Wenn Ihnen der jQuery-Code jetzt noch etwas mystisch vorkommt, ist das schon in Ordnung. Wir werden uns im ersten Rezept dieses Kapitels die Grundlagen anschauen. Was Sie aber aus diesem Code-Beispiel mitnehmen sollten, ist das jQuery-Konzept »Elemente finden, mit denen etwas getan wird«. In diesem Beispiel haben wir alle `<div>`-Elemente in der HTML-Seite mit Hilfe der jQuery-Funktion (`jQuery()`) gefunden und dann jQuery-Methoden genutzt, um mit dieser Menge etwas zu tun (zum Beispiel `hide()`, `text()`, `addClass()`, `show()`).

## Verketteten

jQuery ist so aufgebaut, dass Sie dessen Methoden verketteten können. Warum sollten Sie nicht zum Beispiel ein Element finden und dann nacheinander Operationen für dieses anwenden können? Unser vorheriges Codebeispiel, das Konzept »Elemente finden, mit denen etwas getan wird« könnte durch das Verketteten zu einer einzigen JavaScript-Anweisung kombiniert werden.

Dieser Code:

```
//alle divs auf der Seite verbergen
jQuery('div').hide();

//Text in allen divs aktualisieren
jQuery('div').text('neuer Inhalt');

//Klassen-Attribut mit dem Wert updatedContent bei allen divs hinzufügen
jQuery('div').addClass("updatedContent");

//alle divs auf der Seite anzeigen
jQuery('div').show();
```

kann durch Verketteten auch so aussehen:

```
jQuery('div').hide().text('neuer Inhalt').addClass("updatedContent").show();
```

oder mit Einrückung und Zeilenumbruch:

```
jQuery('div')
  .hide()
  .text('neuer Inhalt')
  .addClass("updatedContent")
  .show();
```

## Symbole

- \$ (Dollar-Zeichen)
  - Abkürzung 71
  - als Variable 280
  - Präfix 109
  - Variable 69
- \$ (Funktion) 279
- \$ (Kurzform)
  - Verwenden in jQuery-Plugins 278
- \$(jQuery()) (Funktion) 22, 91
- \$.Alias
  - Verwenden ohne globale Konflikte 33
- > (größer als)
  - Kombinator für direkt abhängige Elemente 36
- <script>-Tags 128
- <select>-Element 239
- <table>-Element 114
- <textarea>-Element 230
- <tr> (Element) 114
- + (Plus-Zeichen)
  - Kombinator für benachbarte Geschwister 38
- . (Punkt)
  - Operatoren 96
- .context (Eigenschaft) 74
- .innerHTML (Eigenschaft) 120
- .js-Dateien
  - Minifizieren 127
- .load(url) (Methode) 410
- .selector (Eigenschaft) 74
- .toggleCheck() (Plugin) 101
- .unbind() 241
- ~ (Tilde)
  - Kombinator für allgemeine Geschwister 39

## A

- abhängige Elemente
  - Finden im Wrapper-Set 18
- Abhängigkeiten 249

- Abkürzungen
  - Ajax 410
  - Ajax-Anforderungen 410
- Abschicken
  - Forms mit Ajax 234
- absolute Positionierung
  - Elemente 149
- Abstand
  - Animations-Methoden 153
- addClass 172
- Ajax
  - Anfragen queuen 428
  - auf der gesamten Site verwenden 404
  - Events 405
  - Forms abschicken 234
  - Hilfsfunktionen und Datentypen 410
  - Rückmeldungen 406
  - und jQuery 401
  - Zurück-Button 430
- ajax() (Methode) 402
- Ajax-Events 173
- ajaxForm (Methode) 263
- ajaxQueue() (Funktion) 429
- ajaxSetup() (Methode) 404
- ajaxSync() (Funktion) 430
- Akkordeon
  - Aufziehen 298
  - horizontales Akkordeon 160
- Aktionen
  - Benutzeraktionen testen 440
  - Durchführen auf einer Untermenge des selektierten Sets 66
- Aktivieren
  - Form-Elemente 219
- Aktualisieren
  - DOM 120
- alert()-Fenster 11
- andSelf() (Methode) 21
- Anfragen
  - Ajax-Anfragen queuen 428

- Anhängen
  - Objekte und Daten an DOM-Elemente per data() 84
- animate (Methode) 154, 164, 167, 170
- :animated (Filter) 42
- :animated (Selektor) 168
- Animation
  - Erkennen, ob Elemente aktuell animiert werden 166
  - Geschwindigkeiten 154
  - mehrere parallele hover()-Animationen 189
  - Selektieren von Elementen, die aktuell animiert sind 41
  - Stoppen und Zurücksetzen 167
- anklickbare Elemente
  - Styles 369
- Anwendungs-Status
  - Speichern 422
- Anzeigen
  - Anzahl der verbleibenden Zeichen 230
  - Label oberhalb von Eingabefeldern anzeigen 265
  - modale Fenster 306
- APIs
  - DOM Storage API 423, 425
  - Organisation 7
- appendTo() (Methode) 24–24, 28, 227
- arguments keyword 166
- ARIA (Accessible Rich Internet Applications)
  - Semantik zu Widgets hinzufügen 133
- Arrays
  - Ausfiltern doppelter Einträge mit unique() 82
  - Filtern mit grep() 80
  - Iterieren 79
  - Iterieren über und Verändern von Einträgen mit map() 81
  - Kombinieren mit merge() 81
  - mit ausgewählten Werten aus bestehendem Array erstellen 64
- asynchrone Callbacks
  - Unit Tests 439
- attr() (Methode) 30, 219, 228
- attrDelta (Parameter) 170
- Attribute
  - DOM-Elemente 30
  - Liste 78
  - Selektieren von Elementen 45
  - Umschalten 101
- attrStart (Parameter) 170
- Aufrufe
  - jQuery-Methoden 209
- Aufrufen
  - Plugin-Methoden von jQuery UI 334
- Aufziehen
  - Akkordeon 298
- Ausdrücke 250
- Ausführen
  - JavaScript-Code 11
- Auslösen
  - globale Events 196
- Auswahl-Optionen hinzufügen und entfernen 227
- Auswählen
  - Radio Buttons 222
- autocomplete (Widget) 256

**B**

- Barrierefreiheit
  - Webseiten 133
- Bearbeiten von DOM-Elementen 23
- beforeSend (Callback) 409
- Benachrichtigungen
  - jQuery-Methoden 209
- Benutzer-Events 173
- Benutzeraktionen
  - Unit Tests 440
- Benutzeroberfläche jQuery UI 22
- Bibliotheken
  - Code übernehmen 96
  - JavaScript 2
  - Konflikte 69
  - Namensraum-Konflikte 70
- Bilder
  - zyklisch einblenden 315
- Bilder zyklisch einblenden 315
- bind() 175, 213, 241, 335
- bind() (Methode) 93, 229
- Binden
  - Slider 258
- blur (Event) 234
- blur() (Methode) 223
- boxModel (Attribut) 78
- broadcast() (Funktion) 212
- Browser
  - Storage 421
  - Stylesheets abhängig von der Breite wechseln 151
- Browser-Events 173
- :button (Filter) 47

- Buttons
  - Ajax und der Zurück-Button 430
  - Auswählen von Radio Buttons 222
- C**
- Callbacks
  - asynchrone Callbacks testen 439
  - synchrone Callbacks testen 438
  - Testen mit isFunction() 83
- Captcha
  - Form-Design 244
- change (Event) 93
- change() (Event) 226
- :checkbox (Filter) 47
- Checkboxes
  - Selektieren oder Deselektieren über einen einzelnen Umschalter 225
  - Selektieren oder Deselektieren von Checkboxes durch Links 224
- children() (Methode) 37, 95
- clean() (Methode) 120
- clearForm (Methode) 263
- click-Handler 77
- Click-Listener 312
- clone() (Methode) 27
- Code
  - aus anderen Bibliotheken 96
- complete (Option) 403
- :contains (Filter) 42
- createElement() (Methode) 24
- CSS
  - Eigenschaften 149
  - Informationen 400
  - jQuery UI CSS 366
  - Pseudoklassen 39
  - visibility (Eigenschaft) 44
- cssFloat (Attribut) 78
- Cycle-Plugin 72
- D**
- :data (Filter) 51
- data() (Methode)
  - Objekte und Daten an DOM-Elemente anhängen 84
- dataType 402
- Dateien
  - im Hintergrund hochladen 262
- Daten
  - Anhängen an DOM-Elemente per data() 84
  - dynamische Daten 179
  - Eventhandler 202
  - Wiederverwenden von Handlern 175
- Datentypen
  - Ajax 410
- datepicker (Plugin) 268
- Datum
  - Wählen 267
- Deaktivieren
  - Effekte 170
  - Form-Elemente 219
- Debuggen
  - Debugger verwenden 126
  - jQuery 124
  - Verkettungen 121
- Debugging
  - jQuery-Code 123
  - Skript- und CSS-Referenzen auf Plugin-Dateien als Debugging-Hilfen 329
- Definieren von Methoden 276
- Deselektieren von Checkboxes
  - durch Links 224
  - mit einem einzelnen Umschalter 225
- Deserialisierung
  - JSON 415
- destruktive Änderungen
  - Rückkehr zur vorherigen Selektion 19
- Die jQuery-Bibliothek in eine HTML-Seite einbinden 9
- Dimensionen
  - absolute Positionierung von Elementen 149
  - Elemente 138
  - Elemente in den sichtbaren Bereich scrollen 143
  - Ermitteln, ob Elemente sichtbar sind 145
  - Offset von Elementen 141
  - relative Positionierung von Elementen 149
  - Stylesheets abhängig von der Browser-Breite wechseln 151
  - Window und Document 137
  - Zentrieren von Elementen im sichtbaren Bereich 148
- display:block (Css-Deklaration) 236
- display:none (CSS-Deklaration) 236
- div.errorMessage (Selektor) 236
- div.showErrorMessage (Selektor) 236
- Document
  - Dimensionen 137
- document.ready() (Funktion) 290
- document.ready() (Methode) 218



- document.write() 290
- Dokumentation
  - jQuery 89
  - jQuery-API 7
- Dokumente
  - Registerkarten 303
- Dollar (\$)
  - als Variable 280
- Dollar-Zeichen (\$)
  - Abkürzung 71
  - Präfix 109
  - Variable 69
- DOM
  - Aktualisieren 120
  - Durchlaufen 22
  - Finden einer Menge von Elementen 4
  - Konvertieren aus XML 413
- DOM durchlaufen 35
- DOM Storage API 423, 425
- DOM-Elemente
  - Attribute 30
  - Entfernen 25
  - Ersetzen 26
  - Erstellen, Bearbeiten und Einfügen 23
  - Filtern eines Wrapper-Sets 16
  - Klonen 27
  - Selektieren 13
  - Speicherlecks 197
- DOM-Objekte
  - Konvertieren aus jQuery-Objekten 59
- Doppelte Array-Einträge
  - Ausfiltern mit unique() 82
- drag (Plugin) 202
- drop (Plugin) 202
- Dropdown-Menüs
  - Erstellen 313
- duration (Parameter) 170
- Durchlaufen des DOM 22
- Durchlaufen eines Sets mit Ergebnissen 53
- dynamische Daten
  - Übergeben an Eventhandler 179

**E**

- each() (Callback) 118
- each() (Methode) 54, 99, 115, 302
  - Iterieren über Arrays und Objekte 79
- Easing-Methoden
  - eigene 169
- effect() (Methode) 172
- Effekte 153
  - Animation 154, 166
  - Deaktivieren 170
  - eigene Easing-Methoden 169
  - Fading 155
  - horizontales Akkordeon 160
  - jQuery UI 171, 326
  - sequenzielle Effekte 164
  - Sliding 155, 163
- eigene Iteratoren
  - Schreiben 99
- Einfügen
  - DOM-Elemente 23
- Eingabe
  - Datum wählen 267
  - Eingabeelement mit seinem Inhalt wachsen lassen 266
  - Eingabemasken erstellen 254
  - Label oberhalb von Eingabefeldern anzeigen 265
  - Länge von Texteingabefeldern begrenzen 264
  - Texteingabefeld beim Laden der Seite 218
  - Texteingabefelder auf bestimmte Zeichen beschränken 233
- Eingabemasken
  - Erstellen 254
- Eingeben
  - Werte in Grenzen 260
- elapsed (Parameter) 169
- elastic (Plugin) 267
- Element 4
- Elemente 35
  - abhängige Elemente 18
  - absolute Positionierung 149
  - Animation 166
  - Dimensionen ermitteln 138
  - eigener Filter-Selektor 50
  - Elemente beim Verwenden von event.target erhalten 187
  - Eventhandlers 191
  - Fading 155, 163
  - in den sichtbaren Bereich scrollen 143
  - Kontext-Parameter 49
  - Offsets 141
  - relative Positionierung 149
  - Selektieren anhand des Inhalts 42
  - Selektieren anhand ihrer Sichtbarkeit 44
  - Selektieren anhand von Attributen 45
  - Selektieren mit bestimmten Eigenschaften 47
  - Selektieren über die Index-Reihenfolge 39

- Selektieren über eine negative Selektion 43
- Selektieren von Elementen, die aktuell animiert sind 41
- Selektieren von Form-Elementen anhand des Typs 46
- Selektieren von Geschwister-Elementen 38
- Selektieren von Kind-Elementen 36
- Sichtbar 145
- sichtbarer Bereich 148
- Sliding 155, 163
- Zugriff 182
- ElementReady (Plugin) 184
- end() (Methode) 19, 95
- engine.ready() (Callback) 420
- Engpässe
  - Finden 103
- Entfernen
  - Attribute von DOM-Elementen 30
  - Auswahl-Optionen 227
  - DOM-Elemente 25
  - Überflüssige Wiederholungen 93
  - Whitespace aus Strings oder Form-Werten mit trim() 84
- :eq (Filter) 40
- eq() (Methode) 56, 226
- equals() 438
- ereignisorientierte Programmierung 173
- error (Callback-Methode) 402
- error (Methode) 403
- errorElement 251
- errorLabelContainer 252
- errorPlacement 251
- Ersetzen von DOM-Elementen 26
- Erstellen
  - DOM-Elemente 23
  - Events 199
  - Filter-Selektoren 50
  - JSON 414
  - Musikplayer mit jQuery UI 337
  - Tooltips 291
- Erweitern
  - Objekte mit extend() 86
- esc() (Funktion) 119
- eval() (Funktion) 105, 416
- :even (Filter) 40
- Event Bubbling 188
- Event-Delegation 183, 191
- Event-Objekt
  - eigenes 336
- event.special 200, 212
- event.specialAll 200
- event.target
  - Elemente erhalten 187
- Eventhandler
  - neue Elemente 191
- Events 173
  - Ajax 405
  - Ausführungsschleife für Handler stoppen 185
  - Auslösen von Eventhandlern 178
  - Benachrichtigungen beim Aufruf von jQuery-Methoden 209
  - Binden an Handler 174
  - dynamische Daten an Eventhandler übergeben 179
  - Elemente mit event.target erhalten 187
  - Entfernen von vielen Eventhandlern 177
  - Erstellen 199
  - Event-gesteuerte Plugins 205
  - Eventhandler stellen Daten bereit 202
  - Eventhandler und neue Elemente 191
  - globale Events und Ajax-Architektur 405
  - globales Auslösen von Events 196
  - Handler-Funktionen wiederverwenden 175
  - jQuery dynamisch laden 195
  - mehrere parallele hover()-Animationen 189
  - mit Events von jQuery UI-Plugins umgehen 335
  - Mutation Events 209
  - Objekt-Methoden als Event Listener 213
  - Zugriff 182
- expect() 439
- extend() (Methode)
  - Erweitern von Objekten 86
- extend() method 86

## F

- fadeIn() (Methode) 168, 316
- fadeOut() (Methode) 168
- fadeTo() (Methode) 157, 168
- Fading 155
- Farbe
  - durch Themes ändern 380
- Fehlermeldungen 250
- Felder
  - automatisch vervollständigen 256
  - Eingabemasken erstellen 254
  - Label oberhalb von Eingabefeldern anzeigen 265
- Fenster
  - modale Fenster anzeigen 306

- :file (Filter) 47
- Filter
  - :animated 42
  - :button 47
  - :checkbox 47
  - :contains 42
  - :data 51
  - :eq 40
  - :even 40
  - :file 47
  - :has 42
  - :hidden 44, 47
  - :image 47
  - Liste mit Form-Filtern 46
  - :lt 40
  - :not 43
  - :odd 40
  - :password 47
  - :radio 47
  - :reset 47
  - :submit 47
  - :text 47
  - :visible 44
- filter() (Methode) 16, 19, 44, 46–48
- Filter-Selektoren
  - Erstellen 50
- Filtern
  - Arrays mit grep() 80
  - Wrapper-Sets mit DOM-Elementen 16
- find() (Methode) 16, 18, 24, 75
- Finden
  - Engpässe 103
  - jQuery-Plugins 271
- Finden abhängiger Elemente in Wrapper-Sets 18
- Firebug
  - Timing-Problem 105
- fn (Objekt) 98
  - Definieren von Plugins 276
- focus() (Methode) 222, 229
- focusin (Plugin) 202
- focusout (Plugin) 202
- for...in-Schleife 116
- form (Plugin) 263
- form-Elemente
  - Selektieren von input-Elementen 15
- Form-Elemente
  - Aktivieren und Deaktivieren 219
  - Selektieren anhand des Typs 46
- Formatieren
  - verkettete jQuery-Methoden 95
- Forms HTML Forms 22
- Forms überprüfen 236
- fraction (Parameter) 169
- Framework-Klassen 353, 382
- Function() 105
- Funktionen 83
  - Anhängen an das jQuery-Objekt 276
  - private Funktionen in jQuery-Plugins 280
  - statische Funktionen in jQuery-Plugins 284
  - Wiederverwenden von Eventhandlern 175

## G

- Geschwister-Elemente
  - Selektieren 38
- get() (Methode) 59, 411
- getJSON() (Methode) 418
- getTime() (Methode) 104
- GitHub 272
- globale Events
  - Ajax-Architektur 405
  - Auslösen 196
- globale Konflikte
  - Verwenden des \$-Alias 33
- Google
  - minifizierte Version von jQuery 10
  - Themes in der jQuery UI ThemeRoller-Galerie 327
- Google Code 272
- Google maps 261
- grep() (Methode)
  - Filtern von Arrays 80
- größer als (>)
  - Kombinator für direkt abhängige Elemente 36

## H

- Handler
  - Ausführungsschleife stoppen 185
  - Auslösen von Eventhandlern 178
  - Binden an Events 174
  - dynamische Daten an Eventhandler übergeben 179
  - Entfernen vieler Eventhandler 177
  - Eventhandler stellen Daten bereit 202
  - Eventhandler und neue Elemente 191
  - Wiederverwenden 175
- :has (Filter) 42
- height (Methode) 137
- Herausziehen
  - Selektoren 109

- :hidden (Filter) 44, 47
  - hide() (Methode) 153, 157
  - history (Plugin) 431
  - historyLoad() (Callback) 431
  - Hochladen
    - Dateien im Hintergrund 262
  - horizontales Akkordeon
    - Erzeugen 160
  - hover()-Animationen
    - mehrere parallele Animationen 189
  - href (Attribut) 78
  - hrefNormalized (Attribut) 78
  - HTML
    - Beziehung zu JavaScript 217
  - html() (Methode) 31, 120, 125
  - HTML-Forms 217, 243
    - Abhängig von der Anzahl der Zeichen ins nächste Feld springen 228
    - Abschicken mit Ajax 234
    - Anzahl der verbleibenden Zeichen anzeigen 230
    - Auswahl-Optionen hinzufügen und entfernen 227
    - Auswählen von Radio Buttons 222
    - Dateien im Hintergrund hochladen 262
    - Datum wählen 267
    - Deaktivieren und Aktivieren von Form-Elementen 219
    - Eingabeelement mit seinem Inhalt wachsen lassen 266
    - Eingabemasken 254
    - Label oberhalb von Eingabefeldern anzeigen 265
    - Länge von Texteingabefeldern begrenzen 264
    - Selektieren oder Deselektieren von Check-boxen durch Links 224
    - Selektieren oder Deselektieren von Check-boxen über einen einzelnen Umschalter 225
    - Selektieren von Form-Elementen anhand des Typs 46
    - Texteingabefeld beim Laden der Seite 218
    - Texteingabefelder auf bestimmte Zeichen beschränken 233
    - Textfelder automatisch vervollständigen 256
    - Überprüfen 236, 244
    - Werte in Grenzen eingeben 260
    - Wertebereiche selektieren 257
  - HTML-Fragmente 412
  - HTML-Inhalt
    - Lesen und Setzen 31
  - HTML-Seiten
    - Einbinden der jQuery-Bibliothek 9
  - HTML5 Media Element API 338
  - htmlSerialize (Attribut) 78
- ## I
- IE (Internet Explorer)
    - href (Attribut) 78
    - vertikale CSS-Rahmen rendern (Fehler) 315
  - if/else (Anweisung) 90
  - :image (Filter) 47
  - index() (Methode) 63
  - Index-Reihenfolge
    - Selektieren von Elementen 39
  - Indexe
    - Ermitteln für ein Element in einer Selektion 62
  - Inhalt
    - HTML-Inhalt 31
    - Selektieren von Elementen 42
    - Text-Inhalt 32
  - innerHeight (Methode) 139
  - innerHTML (Eigenschaft) 32
  - innerWidth (Methode) 139
  - input:text (Selektor) 221
  - Interaktionen
    - jQuery UI 325
  - Interface (Paket) 325
  - Internet Explore IE 22
  - invalidHandler (Callback) 253
  - is() (Methode) 42
  - isFunction() (Methode) 83
    - Testen von Callback-Funktionen 83
  - Iterieren
    - Array-Elemente mit map() 81
- ## J
- JavaScript
    - am Ende einer Seite 432
    - Beziehung zu HTML 217
    - Bibliotheken 2
    - Packen 128
    - Profiler 104
    - und jQuery 89
    - zurückhaltendes JavaScript schreiben 129
  - JavaScript Object Notation JSON 22

- JavaScript Template Engine
  - Anzeigen von JSON-Daten 425
- JavaScript-Code
  - Ausführen 11
- jQuery 2
  - AP-Organisation 7
  - Definition 1
  - dynamisch laden 195
  - Philosophie 4
  - und JavaScript 89
- jQuery Plugin Repository 272
- jQuery UI 289, 325, 351
  - Akkordeon 298
  - Bilder zyklisch einblenden 315
  - Dropdown-Menüs erstellen 313
  - Effekte 171
  - eigene Tooltips erstellen 291
  - Einbinden der kompletten jQuery UI-Suite 327
  - Einbinden einzelner jQuery UI-Plugins 328
  - initializing jQuery UI plugins 329
  - jQuery UI-Plugins zerstören 337
  - Layout und Theme-Styles von jQuery UI überschreiben 370
  - mehrere Themes auf einer einzelnen Seite verwenden 391
  - mit Events von jQuery UI-Plugins umgehen 335
  - modale Fenster anzeigen 306
  - Musikplayer mit jQuery UI 337
  - Navigieren in einem Baum 295
  - Optionen lesen und setzen 333
  - Plugin-Methoden von jQuery UI aufrufen 334
  - Registerkarten in einem Dokument 303
  - Sliding Panels 320
  - Standardwerte erstellen 331
  - Themes auf andere Komponenten anwenden 381
  - Themes für jQuery-Widgets mit ThemeRoller erstellen 356
- jQuery UI CSS
  - Versionen 366
- jQuery()-Funktion 13
- jQuery-Funktion 23
  - Selektieren von DOM-Elementen 13
- jQuery-Objekte
  - Konvertieren in DOM-Objekte 59
  - Puffern 108
- jQuery-Plugin s 271
- jQuery-Plugins 325
  - \$ (Kurzform) 278
  - Finden 271
  - Initialisieren 329
  - jQuery UI-Plugins einbinden 328
  - jQuery UI-Plugins zerstören 337
  - jQuery UI-Standardwerte erstellen 331
  - Metadata-Plugin 282
  - mit Events von jQuery UI-Plugins umgehen 335
  - Musikplayer mit jQuery UI 337
  - Optionen für jQuery UI-Plugins lesen und setzen 333
  - Optionen mitgeben 277
  - Plugin-Methoden von jQuery UI aufrufen 334
  - Plugins mit QUnit testen 286
  - private Funktionen 280
  - Schreiben 273
  - statische Funktionen 284
- jQuery.Event 202, 212
- JSON (JavaScript Object Notation)
  - Anwendung 416
  - Erstellen 414
  - JavaScript Template Engine 425
  - Parsen 415
- JSONP 417
- jStore (Plugin) 420
- jStore.ready() (Callback) 420

**K**

- keydown (Event) 230, 234
- keyup (Event) 93, 230, 234
- Kind-Elemente
  - Selektieren 36
  - verschachtelte, ungeordnete Listen 295
- Klassen
  - Kategorien 352
- Klassennamen-Selektoren
  - Performance 109
- Klonen
  - DOM-Elemente 27
- Kombinator für allgemeine Geschwister (~) 39
- Kombinator für benachbarte Geschwister (+) 38
- Kombinator für direkt abhängige Elemente (>) 36
- Kombinieren von Arrays mit merge() 81
- Konfiguration
  - Konflikte mit anderen Bibliotheken 69

- Konflikte
  - globale 33
  - mit anderen Bibliotheken 69
- Kontext
  - Durchlaufen des DOM 22
  - Selektieren von DOM-Elementen 15
- Kontext-Parameter
  - Selektieren von Elementen 49
- Konvertieren von jQuery-Objekten in DOM-Objekte 59
- Kurzform
  - Browser- und Ajax-Events 173
- L**
- Label
  - oberhalb von Eingabefeldern anzeigen 265
- Laden
  - jQuery dynamisch 195
  - Plugins 127
  - Seiten 218
  - Tabellen 111
- laden
  - Webseiten 12
- Länge
  - Länge von Texteingabefeldern begrenzen 264
- Layout
  - Layout und Theme-Styles von jQuery UI überschreiben 370
- leadingWhitespace (Attribut) 78
- Lesen
  - Attribute von DOM-Elementen 30
  - HTML-Inhalt 31
  - Optionen für jQuery UI-Plugins 333
  - Text-Inhalt 32
- linear (Funktion) 169
- Links
  - Selektieren oder Deselektieren von Check-boxen durch Links 224
- live() 421
- live() (Methode) 192
- LiveQuery (Plugin) 184
- Lizenz von Plugins 274
- Lokalisierung 251, 269
- Lookups
  - Name Lookups 117
- :lt (Filter) 40

- M**
- Makros 177
- map() (Methode) 66, 212
  - Iterieren über und Verändern von Array-Elementen 81
- Masked Input (Plugin) 254
- maxlength (Plugin) 264
- Menüs
  - Dropdown-Menüs erstellen 313
- merge() (Methode)
  - Kombinieren von zwei Arrays 81
- metadata (Plugin) 248, 266
- Metadata-Plugin 282
- Methoden
  - Basis-Methoden in jQuery UI-Plugins 334
  - Benachrichtigungen beim Aufruf von jQuery-Methoden 209
  - Definieren 276
  - Objekt-Methoden als Event Listener 213
  - Plugin-Methoden von jQuery UI 334
  - Plugins 247
- Minifizierung
  - .js-Dateien 127
  - Größe des JavaScript-Codes und Anzahl der HTTP-Anfragen verringern 432
  - minifizierte Version von jQuery 124
  - minifizierter Code 10
- mit Events von jQuery UI-Plugins umgehen 335
- Mithilfe
  - Plugin-Entwicklung 275
- modale Fenster
  - Anzeigen 306
- module() (Funktion) 443
- mouseover 167
- mousewheel (Plugin) 202
- Musikplayer
  - mit jQuery UI 337
- Mutation Events 209
- N**
- Nachkommen
  - versus Kind-Element 37
- Name Lookups
  - verringern 117
- Namensraum
  - Eventhandler auslösen 178
  - jQuery und andere Bibliotheken 70
- Namensräume
  - Plugins 177
- Navigieren in einem Baum 295

next() (Methode) 39  
nextAll() (Methode) 39  
noCloneEvent (Attribut) 78  
noConflict() (Methode) 69  
:not (Filter) 43  
not() (Methode) 43

## O

### Objekte

Anhängen an DOM-Elemente per data() 84  
DOM-Objekte 59  
Erweitern mit extend() 86  
Iterieren 79  
jQuery-Objekte 59, 108  
Objekt-Methoden als Event Listener 213  
:odd (Filter) 40  
offset (Methode) 141, 143  
offsetParent (Methode) 141  
Offsets  
  Elemente 141  
ok() 437  
one() 175  
opacity (Attribut) 78  
Optimierung  
  Selektoren 36  
option (Methode) 335  
Optionen  
  Metadata-Plugin 283  
  Übergeben an Plugins 277  
Optionen an Plugins übergeben 277  
outerHeight (Methode) 139  
outerWidth (Methode) 139

## P

### Packen

JavaScript 128

### Panels

Sliding Panels in jQuery UI 320

:password (Filter) 47

### Performance 89

Aktualisieren des DOM 120  
Animations-Geschwindigkeiten 154  
Attribute 101  
Barrierefreiheit 133  
Code aus anderen Bibliotheken 96  
Debuggen von jQuery 124  
Debuggen von jQuery-Code 123  
eigene Iteratoren 99  
Engpässe 103

Event Propagation 409  
globale Events auslösen 196  
jQuery-Objekte 108  
Name Lookups 117  
Progressive Verbesserung 131  
Schleifen 114  
Selektoren 109  
Server-Anfragen 126  
Tabellen 111  
Überflüssige Wiederholungen 93  
verkettete jQuery-Methoden 95  
Verkettungen debuggen 121  
zurückhaltendes JavaScript schreiben 129  
Persistieren von Daten  
  Web-Browser 419  
Philosophie von jQuery 4  
Player  
  Musikplayer mit jQuery UI 337  
Plugins 243, 325  
  \$ (Kurzform) 278  
  Dateien im Hintergrund hochladen 262  
  Datum wählen 267  
  Eingabeelement mit seinem Inhalt wachsen  
    lassen 266  
  Eingabemasken erstellen 254  
  einzelne jQuery UI-Plugins einbinden 328  
  Event-gesteuert 205  
  event.special 202  
  Finden 271  
  Forms überprüfen 244  
  Hinzufügen von Funktionalität 71  
  jQuery UI-Plugins initialisieren 329  
  jQuery UI-Plugins zerstören 337  
  jQuery UI-Standardwerte 331  
  jQuery-Plugins schreiben 273  
  Label oberhalb von Eingabefeldern anzeigen  
    265  
  Laden 127  
  Länge von Texteingabefeldern begrenzen 264  
  Metadata-Plugin 282  
  mit Events von jQuery UI-Plugins umgehen  
    335  
  Musikplayer mit jQuery UI 337  
  Namensräume 177  
  Optionen für jQuery UI-Plugins lesen und  
    setzen 333  
  Optionen mitgeben 277  
  Plugin-Methoden von jQuery UI aufrufen  
    334

- Plugins mit QUnit testen 286
- Polling 184
- private Funktionen 280
- statische Funktionen 284
- stopImmediatePropagation() 187
- Textfelder automatisch vervollständigen 256
- Werte in Grenzen eingeben 260
- Wertebereiche selektieren 257
- Plus-Zeichen (+)
  - Kombinator für benachbarte Geschwister 38
- Polling 184
- position (Methode) 141
- Positionierung
  - absolute Positionierung 149
  - relative Positionierung 149
- post() (Methode) 411
- preventDefault() (Methode) 204
- private Funktionen
  - jQuery-Funktionen 280
- Profiler
  - Engpässe finden 104
- Programmieren
  - Schleifen 114
- Progressive Verbesserung
  - jQuery 131
- Pseudoklassen Filter 39
- Puffern von jQuery-Objekten 108
- Punkt (.)
  - Operatoren 96
- Pure Templating Engine 425

## Q

- Queuen von Ajax-Anfragen 428
- Quirks-Mode 153
- QUnit 436
  - jQuery-Plugins testen 286

## R

- :radio (Filter) 47
- Radio Buttons 256
  - Auswählen 222
- radioClass() (Methode) 96
- Rahmen
  - Animations-Methoden 153
  - unterer Rahmen des Widget-Headers 380
- ready() (Funktion) 195
- ready() (Methode) 11
- Rebinding 191

- Registerkarten
  - in einem Dokument 303
- relative Positionierung
  - Elemente 149
- remove() (Methode) 25–25, 228
- removeAttr() (Methode) 30, 219
- removeClass 172
- replaceAll() (Methode) 27
- replaceWith() (Methode) 26
- :reset (Filter) 47
- resizable (Plugin) 267
- reverseEach() (Methode) 101
- Rückmeldungen
  - Ajax 406

## S

- same() 438
- Schleifen
  - Programmieren 114
- Schreiben
  - eigene Iteratoren 99
  - jQuery-Plugins 273
  - Selektoren 109
- Schriftgröße 159
- scriptEval (Attribut) 79
- Scrollen
  - Elemente in den sichtbaren Bereich 143
- scrollLeft (Methode) 145
- scrollTo (Plugin) 148
- scrollTop (Methode) 143
- Seiten
  - Barrierefreiheit 133
  - JavaScript am Ende einer Seite 432
  - Laden 12
  - Mehrere Themes auf einer einzelnen Seite verwenden 391
  - Texteingabefeld beim Laden der Seite 218
- selector.focus() (Methode) 218
- selects 256
- Selektieren
  - Checkboxes durch Links 224
  - Checkboxes über einen einzelnen Umschalter 225
  - Wertebereiche 257
- Selektieren von Elementen 35
  - aktuell animierte 41
  - anhand des Inhalts 42
  - anhand ihrer Sichtbarkeit 44
  - anhand von Attributen 45
  - DOM-Elemente 13



- eigener Filter-Selektor 50
- Form-Elemente anhand des Typs 46
- Geschwister-Elemente 38
- Index-Reihenfolge 39
- Kind-Elemente 36
- Kontext-Parameter 49
  - mit bestimmten Eigenschaften 47
  - negative Selektion 43
- Selektionen
  - Bestimmen der verwendeten Selektion 74
  - Rückkehr vor einer destruktiven Änderung 19
  - vereinigen mit der vorherigen 21
- Selektoren 15
  - Aufbauen 35
  - Herausziehen 109
  - Kind-Elemente 295
  - Reduzieren des Sets auf ein bestimmtes Element 56
  - Schreiben 109
- Serialisierung
  - JSON 415
- Server-Anfragen
  - Verringern 126
- Sessions
  - Anwendungs-Status speichern 422
- setData event 85
- setInterval 232
- Sets
  - Durchführen von Aktionen auf einer Unter-  
menge des selektierten Sets 66
  - Durchlaufen 53
  - Reduzieren auf ein bestimmtes Element 56
- setTimeout() (Funktion) 90, 99, 101, 123
- setVisibility() (Funktion) 94
- Setzen
  - Attribute von DOM-Elementen 30
  - HTML-Inhalt 31
  - Optionen für jQuery UI-Plugins 333
  - Text-Inhalt 32
- show() (Methode) 125, 157
- siblings() (Methode) 38, 95, 223
- Sicherheit
  - Ajax 417
- Sichtbarkeit
  - Selektieren von Elementen 44
  - visibility (Eigenschaft) 44
- Skripten
  - Vergleich zu Bibliotheks-Widgets 352
- sleep() (Funktion) 101
- slice() (Methode) 66
- slide (Callback) 258
- slideDown() (Methode) 302
- slider (Widget) 257
- slider() (Funktion) 342
- Sliders
  - Binden 258
- slideToggle 159
- slideUp() (Methode) 153, 158, 302
- Sliding
  - Elemente 155, 163
  - Panels in jQuery UI 320
- slowEach() (Methode) 100
- SourceForge 273
- Speicherlecks
  - DOM-Elemente 197
- Speichern auf dem Client 419
- Speichern des Anwendungs-Status 422
- split() (Methode) 90
- Standard
  - Plugins 278
- Standardwerte
  - jQuery UI-Plugins 329, 331
- start (Event) 336
- start() 439
- statische Funktionen
  - jQuery-Plugins 284
- Status
  - Anwendungs-Status speichern 422
- stop() 439
- stop() (Methode) 168, 189
- stopImmediatePropagation() (Methode) 185
- Stoppen
  - Animationen 167
  - Ausführungsschleife für Handler 185
- Storage
  - Browser 421
- Strings
  - Entfernen von Whitespace mit trim() 84
- style (Attribut) 79
- Styles
  - anklickbare Elemente 369
  - Layout und Theme-Styles von jQuery UI  
überschreiben 370
  - Themes für jQuery-Widgets mit ThemeRoller  
erstellen 356
- Stylesheets
  - Wechseln 151
- :submit (Filter) 47
- submitHandler 253
- success (Callback-Methode) 402

- Support
  - für plugins 274
  - support-Objekt 78
  - swing (Funktion) 169
  - switch(){} (Anweisung) 240
  - synchrone Callbacks
    - Unit Tests 438
- T**
- Tabellen
  - Laden 111
- tbody (Attribut) 79
- Testen 435
  - Callback-Funktionen mit isFunction() 83
  - Debugger 126
  - jQuery-Plugins mit QUnit 286
  - Selektions-Testseiten 110
- Tests gruppieren
  - Unit Tests 443
- Text
  - Länge von Texteingabefeldern begrenzen 264
  - Texteingabefeld beim Laden der Seite 218
  - Texteingabefelder auf bestimmte Zeichen beschränken 233
- :text (Filter) 47
- text() (Methode) 32, 228
- Text-Inhalt
  - Lesen und Setzen 32
- textarea
  - Größe 266
- Textarea
  - Größe 264
- Textfelder
  - automatisch vervollständigen 256
- ThemeRoller 353
  - referencing multiple themes on a single page 391
  - Themes für jQuery-Widgets erstellen 356
- Themes 351
  - Layout und Theme-Styles von jQuery UI überschreiben 370
  - Mehrere Themes auf einer einzelnen Seite verwenden 391
  - Themes auf andere Komponenten anwenden 381
  - Themes für jQuery-Widgets mit ThemeRoller erstellen 356
- ThickBox 307
- this() 90
- Tilde (~)
  - Kombinator für allgemeine Geschwister 39
- time() (Funktion) 105, 117
- toggleAttr() (Methode) 102
- toggleCheck() (Methode) 102
- toggleClass 172
- Tools
  - data() 84
  - each() 79
  - extend() 86
  - grep() 80
  - isFunction() 83
  - map() 81
  - merge() 81
  - support() 77
  - trim() 84
  - unique() 82
- Tooltips
  - Erstellen 291
- trigger() (Methode) 94, 179, 196, 212, 440
- triggerHandler() 440
- triggerHandler() (Funktion) 204
- triggerHandler() (Methode) 212
- trim() (Methode) 84
  - Entfernen von Whitespace aus Strings oder Form-Werten 84
- Twitter
  - Zeichenmenge 264
- Typ
  - Selektieren von Form-Elementen 46
- type (Parameter) 411
- typeof (Operator) 83
- U**
- Überladen
  - jQuery-Events 212
- Überprüfung auf Client-Seite 244
- UI jQuery UI 22
- ui (Argument) 336
- Umschalten
  - Attribute 101
- unbind() 175
- unique() (Function)
  - Ausfiltern doppelter Array-Einträge 82
- unique() (Funktion) 83
- Unit Tests 435
  - asynchrone Callbacks 439
  - atomar 441
  - Benutzeraktionen 440
  - Ergebnisse sicherstellen 437

- synchrone Callbacks 438
- Tests auswählen 444
- Tests gruppieren 443
- Unit tests
  - automatisierte Unit Tests 435
- Untermengen
  - Durchführen von Aktionen 66
- URL 402

## V

- valueOf() (Methode) 104
- Verändern von Array-Elementen mit map() 81
- Vererbung
  - Kind-Elemente in Framework-Klassen 382
- Verketteten 5
  - Definition 20
  - DOM-Elemente 27
  - durchlaufende Methoden 23
- verkettete jQuery-Methoden
  - Formatieren 95
- Verkettungen
  - Debuggen 121
- verschachtelte Daten
  - Anzeigen 295
- Verzeichnisstruktur
  - Theme-Ordner 398
- Viewport
  - Höhe und Breite im Browser 307
- :visible (Filter) 44

## W

- watermark (Plugin) 265
- Web-Seiten Seiten 22
- Wechselschichten
  - Stylesheets 151
- Werte
  - Bereiche selektieren 257
  - Entfernen von Whitespace mit trim() 84
  - in Grenzen eingeben 260
- Whitespace
  - Entfernen aus Strings oder Form-Werten 84

- Widget-spezifische Klassen 352
- Widgets
  - ARIA 133
  - jQuery UI 326
  - Themes für jQuery-Widgets mit ThemeRoller erstellen 356
  - unterer Rahmen des Headers 380
  - Widgets aus Bibliotheken verglichen mit dem Einrichten von Skripten 352

- width (Methode) 137

- Window

- Dimensionen 137

- window.onload (Event) 11

- Wrapper-Set

- Finden abhängiger Elemente 18

- Wrapper-Sets 6

- DOM-Elemente 16

## X

- XML

- Konvertieren in ein DOM 413

## Z

- Zeichen

- Texteingabefelder auf bestimmte Zeichen beschränken 233

- Zeichenanzahl

- Anzeige 230

- Autotabbing 228

- Zentrieren

- Elemente im sichtbaren Bereich 148

- Zerstören

- jQuery UI-Plugins 337

- Zugriff auf Elemente 182

- Zurück-Button

- Ajax 430

- Zurücksetzen von Animationen 167

- Zusammenfassen/Minifizieren

- Tools 128

- zyklisch Einblenden

- Bilder 315