

# Inhaltsverzeichnis

	<b>Vorwort</b> .....	13
	<b>Einleitung</b> .....	17
<b>Teil I</b>	<b>Trainingsvorbereitungen</b> .....	<b>27</b>
<b>1</b>	<b>Werden Sie ein JavaScript-Ninja</b> .....	<b>29</b>
1.1	Die verwendeten JavaScript-Bibliotheken .....	29
1.2	JavaScript verstehen .....	31
1.3	Überlegungen zur browserübergreifenden Programmierung .....	33
1.4	Best Practices .....	37
1.4.1	Best Practice: Assertionen .....	38
1.4.2	Best Practice: Performance-Analysen .....	38
1.5	Zusammenfassung .....	39
<b>2</b>	<b>Aufrüsten mit Assertionen und Debugging</b> .....	<b>41</b>
2.1	Code-Debugging .....	42
2.1.1	Logging .....	42
2.1.2	Breakpoints .....	44
2.2	Testerstellung .....	46
2.3	Test-Frameworks .....	48
2.3.1	JUnit .....	51
2.3.2	YUI Test .....	51
2.3.3	JUnit .....	51
2.3.4	Neuere Unit-Test-Frameworks .....	52
2.4	Die Grundpfeiler einer Testsuite .....	52
2.4.1	Assertionen .....	52
2.4.2	Testgruppen .....	53
2.4.3	Asynchrones Testen .....	55
2.5	Zusammenfassung .....	57

<b>Teil II</b>	<b>Basistraining für angehende JavaScript-Ninjas</b>	<b>59</b>
3	Funktionen – Ein JavaScript-Kernkonzept	61
3.1	Worin besteht der funktionale Unterschied?	62
3.1.1	Warum ist der funktionale Charakter von JavaScript so bedeutsam?	64
3.1.2	Sortieren mit einer Vergleichsfunktion (Comparator)	70
3.2	Deklarationen	73
3.2.1	Funktionen und Scopes	77
3.3	Funktionsaufrufe	82
3.3.1	Von Argumenten zu Funktionsparametern	82
3.3.2	Aufruf als Funktion	84
3.3.3	Aufruf als Methode	85
3.3.4	Aufruf als Konstruktor	88
3.3.5	Aufruf mit den Methoden apply() und call()	91
3.4	Zusammenfassung	96
4	Der Umgang mit Funktionen	99
4.1	Anonyme Funktionen	99
4.2	Rekursion	102
4.2.1	Rekursion in benannten Funktionen	102
4.2.2	Rekursion mit Methoden	104
4.2.3	Das Problem mit der »gestohlenen« Referenz	105
4.2.4	Benannte Inline-Funktionen	108
4.2.5	Die callee-Eigenschaft	110
4.3	Programmierspaß mit Funktionen als Objekten	111
4.3.1	Funktionen speichern	112
4.3.2	Selbstmemoisierende Funktionen	114
4.4	Argumentlisten variabler Länge	119
4.4.1	Argumentlisten variabler Länge mit apply()	119
4.4.2	Funktionsüberladung	121
4.5	Auf Funktionen prüfen	131
4.6	Zusammenfassung	133
5	Der Umgang mit Closures	135
5.1	Die Funktionsweise von Closures	135
5.2	Closures verwenden	140
5.2.1	Private Variablen	140
5.2.2	Callbacks und Timer	143

5.3	Funktionskontexte binden .....	147
5.4	Partielle Funktionsanwendung .....	151
5.5	Funktionsverhalten überschreiben .....	155
5.5.1	Memoisierung .....	155
5.5.2	Funktions-Wrapping .....	159
5.6	Direkte Funktionen .....	161
5.6.1	Temporärer Scope und private Variablen .....	163
5.6.2	Schleifen .....	167
5.6.3	Library-Wrapping .....	169
5.7	Zusammenfassung .....	171
6	<b>Objektorientierung und Prototypen</b> .....	173
6.1	Instanziierung und Prototypen .....	173
6.1.1	Objektinstanziierung .....	174
6.1.2	Objektypisierung mittels Konstruktoren .....	182
6.1.3	Vererbung und die Prototypkette .....	184
6.1.4	HTML-DOM-Prototypen .....	189
6.2	Häufige Fehler .....	191
6.2.1	Erweiterungsobjekt .....	191
6.2.2	Number-Erweiterung .....	193
6.2.3	Subclassing nativer Objekte .....	195
6.2.4	Instanziierungsprobleme .....	197
6.3	Klassenähnlichen Code schreiben .....	201
6.3.1	Serialisierbarkeit von Funktionen prüfen .....	204
6.3.2	Unterklassen initialisieren .....	206
6.3.3	super-Methoden erhalten .....	207
6.4	Zusammenfassung .....	209
7	<b>Programmentwicklung mit regulären Ausdrücken</b> .....	211
7.1	Die Vorzüge der regulären Ausdrücke .....	212
7.2	Reguläre Ausdrücke – Übersicht .....	213
7.2.1	Was sind reguläre Ausdrücke? .....	214
7.2.2	Terme und Operatoren .....	215
7.3	Reguläre Ausdrücke kompilieren .....	221
7.4	Übereinstimmende Teilbereiche »capturen« .....	223
7.4.1	Einfache Captures durchführen .....	224
7.4.2	Übereinstimmungssuche mit globalen Ausdrücken .....	225
7.4.3	Captures referenzieren .....	227
7.4.4	Nicht zu erfassende Gruppen .....	228

7.5	Ersetzen mit Funktionen. ....	230
7.6	Generelle Probleme mit regulären Ausdrücken lösen .....	233
7.6.1	Strings kürzen. ....	233
7.6.2	Zeilenumbrüche finden .....	235
7.6.3	Unicode .....	236
7.6.4	Maskierungszeichen. ....	237
7.7	Zusammenfassung .....	238
8	Der Umgang mit Threads und Timern .....	239
8.1	Die Arbeitsweise von Timern und Threads .....	240
8.1.1	Timer einrichten und deaktivieren .....	240
8.1.2	Timer-Ausführung innerhalb des Ausführungs-Threads ...	241
8.1.3	Unterschiede zwischen Timeouts und Intervallen .....	244
8.2	Minimale Timer-Verzögerung und Zuverlässigkeit .....	245
8.3	Der Umgang mit rechenintensiven Operationen .....	249
8.4	Zentrale Timer-Steuerung .....	253
8.5	Asynchrones Testen. ....	256
8.6	Zusammenfassung .....	258
<b>Teil III Ninja-Training .....</b>		<b>259</b>
9	Ninja-Alchemie: Codeauswertung zur Laufzeit. ....	261
9.1	Mechanismen zur Codeauswertung. ....	261
9.1.1	Auswertung per eval()-Methode. ....	262
9.1.2	Auswertung per Funktionskonstruktor .....	265
9.1.3	Auswertung mit Timern .....	266
9.1.4	Auswertung im globalen Scope .....	266
9.1.5	Sichere Codeauswertung .....	270
9.2	Funktionen dekompilieren .....	271
9.3	Codeauswertung in Aktion .....	274
9.3.1	JSON konvertieren .....	274
9.3.2	Importieren von Code mit Namensraum .....	276
9.3.3	JavaScript komprimieren und verschleiern .....	277
9.3.4	Dynamisches Umschreiben von Code .....	280
9.3.5	Aspektororientierte Skript-Tags .....	281
9.3.6	Metasprachen und DSLs .....	283
9.4	Zusammenfassung .....	286

10	<b>Die with-Anweisung</b> .....	289
10.1	<b>Arbeitsweise der with-Anweisung</b> .....	290
10.1.1	Referenzieren von Eigenschaften innerhalb eines with-Scopes .....	290
10.1.2	Zuweisungen innerhalb eines with-Scopes .....	292
10.1.3	Überlegungen zur Ausführungsgeschwindigkeit .....	294
10.2	<b>Praxisorientierte Beispiele</b> .....	296
10.3	<b>Importieren von Code mit Namensraum</b> .....	298
10.4	<b>Testen</b> .....	299
10.5	<b>Templating mit with</b> .....	300
10.6	<b>Zusammenfassung</b> .....	303
II	<b>Cross-Browser-Strategien</b> .....	305
II.1	<b>Auswahl der Browser</b> .....	305
II.2	<b>Die fünf wichtigsten Belange bei der Entwicklung</b> .....	307
II.2.1	Browser-Bugs und Unterschiede zwischen den Browsern ...	309
II.2.2	Browser-Bug-Fixes .....	309
II.2.3	Koexistenz mit Auszeichnungssprachen und externem Code	311
II.2.4	Fehlende Features .....	317
II.2.5	Regressionen .....	319
II.3	<b>Strategien zur Implementierung</b> .....	321
II.3.1	Gefahrlose Cross-Browser-Korrekturen .....	321
II.3.2	Objekterkennung .....	323
II.3.3	Feature-Simulation .....	325
II.3.4	Nicht überprüfbare Browserprobleme .....	328
II.4	<b>Anzahl der Annahmen reduzieren</b> .....	331
II.5	<b>Zusammenfassung</b> .....	332
12	<b>Attribute, Eigenschaften und CSS</b> .....	335
12.1	<b>Attribute und Eigenschaften</b> .....	337
12.1.1	Cross-Browser-Namensgebung .....	338
12.1.2	Beschränkungen bei der Namensgebung .....	339
12.1.3	Unterschiede zwischen XML und HTML .....	340
12.1.4	Verhalten benutzerdefinierter Attribute .....	341
12.1.5	Überlegungen zur Geschwindigkeit .....	341
12.2	<b>Attribute und Cross-Browser-Probleme</b> .....	345
12.2.1	Automatische Ergänzung von Namen und IDs im DOM ...	345
12.2.2	URL-Normalisierung .....	347
12.2.3	Das style-Attribut .....	349

12.2.4	Das type-Attribut . . . . .	349
12.2.5	Das tabIndex-Problem . . . . .	351
12.2.6	Knotennamen . . . . .	351
12.3	Stilattribute . . . . .	352
12.3.1	Wo sind die Stile? . . . . .	352
12.3.2	Namensgebung bei Stileigenschaften . . . . .	355
12.3.3	Die Stileigenschaft float . . . . .	357
12.3.4	Konvertierung von Pixelwerten . . . . .	357
12.3.5	Vermessung von Höhen und Breiten . . . . .	358
12.3.6	Transparenzen durchschauen . . . . .	363
12.3.7	Farben . . . . .	366
12.4	Abfrage berechneter Stile . . . . .	369
12.5	Zusammenfassung . . . . .	373
 <b>Teil IV Meister-Training . . . . .</b>		<b>375</b>
13	<b>Der Umgang mit Ereignissen . . . . .</b>	<b>377</b>
13.1	Anbindung von Event-Handlern und Lösen dieser Bindung . . . . .	378
13.2	Das Event-Objekt . . . . .	383
13.3	Verwaltung mehrerer Handler . . . . .	387
13.3.1	Zusammengehörige Informationen zentral speichern . . . . .	387
13.3.2	Verwaltung von Event-Handlern . . . . .	391
13.4	Ereignisse auslösen . . . . .	401
13.4.1	Benutzerdefinierte Ereignisse . . . . .	403
13.5	Bubbling und Delegation . . . . .	408
13.5.1	Ereignisse an übergeordnete Elemente delegieren . . . . .	408
13.5.2	Umgehen von Browserdefiziten . . . . .	409
13.6	Das ready-Ereignis . . . . .	420
13.7	Zusammenfassung . . . . .	422
14	<b>Manipulation des DOMs . . . . .</b>	<b>425</b>
14.1	<b>HTML-Injektion ins DOM . . . . .</b>	<b>426</b>
14.1.1	Konvertierung von HTML-Code in eine DOM-Struktur . . . . .	427
14.1.2	Einfügen ins Dokument . . . . .	431
14.1.3	Skriptausführung . . . . .	433
14.2	Klonen von Elementen . . . . .	435
14.3	Entfernen von Elementen . . . . .	437

14.4	Textinhalte .....	439
	14.4.1 Textzuweisungen .....	440
	14.4.2 Auslesen des Textinhalts .....	441
14.5	Zusammenfassung .....	442
15	<b>CSS-Selector-Engines</b> .....	443
15.1	Die W3C-Selektoren-API. ....	445
15.2	XPath zur Suche von Elementen verwenden .....	448
15.3	Reine DOM-Implementierung. ....	450
	15.3.1 Parsen des Selektors. ....	453
	15.3.2 Elemente suchen .....	454
	15.3.3 Filtern .....	455
	15.3.4 Rekursion und Zusammensetzen. ....	456
	15.3.5 Bottom-Up-Selector-Engine. ....	457
15.4	Zusammenfassung .....	459
	<b>Stichwortverzeichnis</b> .....	461