

Dieser Artikel ist ein Teil des Extra-Kapitels »Features, Hacks, Gadgets« aus dem »Praxisbuch Mac OS X Tiger«. Das Kapitel wurde in einzelne Artikel aufgeteilt, die in sich abgeschlossen sind und einzeln unter <http://www.galileodesign.de/959> und <http://www.galileocomputing.de/963> für Leser des Praxisbuchs und der zweiten Auflage von »UNIX für Mac OS X-Anwender« zum Download zur Verfügung stehen. Vorschläge und Anregungen für weitere Themen werden gerne entgegen genommen. Diese PDF-Datei wurde für den Ausdruck optimiert und verwendet das Papierformat DIN A4.



3 Safari für Webworker

Safari hat sich nach seiner Einführung 2003 schnell als der Standard-Browser unter Mac OS X etabliert. Neben den Funktionen, die das Abrufen von Webseiten komfortabel gestalten, verfügt Safari noch über einige weitere Fähigkeiten, die insbesondere für Web-Designer sehr nützlich sein können. Da Safari auch in Form des Web Kits als Grundlage für das Dashboard dient, lassen sich einige der hier beschriebenen Funktionen auch für eigene Widgets und deren Programmierung verwenden.



3.1 Texte in Safari bearbeiten

Text im Layout | Wenn man eine Webseite gestaltet, dann kann es bei einigen Seiten notwendig sein, den Text direkt in das Layout zu platzieren. Zum Beispiel bei der Startseite, die ja den Besucher schnell einfangen und für die Seite interessieren muss, lohnt es sich darauf zu achten, dass die Länge der Sätze und auch der Wörter perfekt mit den Grafiken harmonisiert. In diesem Fall ist die Eingabe des Textes im Quellcode eher unkomfortabel, da nach jeder Änderung des Textes die Datei gespeichert und im Browser neu geladen werden muss. Safari bietet zwei Möglichkeiten, mit denen Texte direkt im Browser eingegeben werden können.

X Open Source

Bei Safari handelt es sich nicht um eine vollständige Eigenentwicklung von Apple. Vielmehr bedienen sich die Programmierer des Programms Konqueror, dem Standard-Browser der insbesondere unter Linux sehr verbreiteten grafischen Oberfläche KDE.



Abbildung 1: Auch die Formatierung von Text ist teilweise möglich.

Das Attribut contenteditable | Die erste Möglichkeit, von der in Abbildung 1 Gebrauch gemacht wurde, besteht in dem Attribut `contenteditable`. Der Text in Absätzen, Überschriften und Blöcken, die Sie mit diesem Attribut auszeichnen,

können in Safari direkt im Fenster bearbeitet werden. Das Attribut wird ohne weitere Parameter direkt an das jeweilige HTML-Tag angehängt. Beispielsweise könnten Sie durch die Eingabe

```
<p contenteditable>Dieser Absatz kann bearbeitet werden.</p>
```

einen ganzen Absatz zur Manipulation durch den Surfer freigeben. Beachten Sie hierbei, dass Safari zwar die Eingaben und Änderungen des Surfers im Fenster anzeigt, den eigentlichen Quelltext aber nicht ändert.

Gesamtes Dokument bearbeiten | Während sich das Attribut `contenteditable` nur auf das ausgezeichnete HTML-Tag und dessen Inhalte bezieht, ist es auch möglich, den Text des gesamten Dokumentes zur Bearbeitung durch den Surfer freizugeben. Hierzu dient die JavaScript-Anweisung `document.designMode`, die Sie im `<head>`-Bereich des Dokumentes nutzen können. Mit dem Aufruf

```
<script>
document.designMode="On";
</script>
```

geben Sie den gesamten Inhalt des Dokuments zur Bearbeitung frei. Es ist dem Surfer in Safari nun möglich, sämtliche Texte zu bearbeiten, zu löschen und teilweise zu formatieren.

Dashboard | Neben der akkuraten Anpassung von Texten lassen sich diese beiden Funktionen natürlich auch für die Entwicklung eigener Widgets auf dem Dashboard nutzen. So verwendet beispielsweise das Widget **Notizzettel** diese Funktionen, um den Text der Notiz auf der Vorderseite einzugeben.

3.2 Das Debug-Menü

Hilfen für Entwickler | Bei der Entwicklung eines Programms stoßen die Programmierer immer wieder auf Fehler und Probleme, deren Ursache nicht sofort einsichtig ist. Um sich die Arbeit ein wenig zu erleichtern, bauen viele Entwickler in ihre Programme einige spezielle Funktionen ein, die sie bei der Suche nach den Ursachen unterstützen und die Sammlung von Informationen erleichtern. Auch in Safari sind solche Funktionen in Form eines speziellen Menüs **Debug** enthalten, das aber dem »normalen« Anwender nicht angezeigt wird. Am Terminal können Sie mit der Eingabe

```
defaults write com.apple.Safari IncludeDebugMenu 1
```

die Voreinstellungen von Safari so manipulieren, dass Ihnen nach einem Neustart das **Debug**-Menü rechts vom **Hilfe**-Menü angezeigt wird. (Um das Menü wieder zu deaktivieren, ersetzen Sie in diesem Aufruf 1 durch 0 und starten das Programm erneut.)

In dem nun verfügbaren Menü (siehe Abbildung 2) finden sich mit den Optionen **Start/Stop Profiling With Sample**, **Use Shark for Profiling**, **Always Check for World Leaks**, ... einige Funktionen, die in erster Linie für Entwickler wichtig sind. Diese dienen dazu, die Performance des Programms zu analysieren und zu optimieren.

Tastenkürzel | Über den Menüpunkt **Keyboard and Mouse Shortcuts** erhalten Sie in einem neuen Fenster eine Liste aller in Safari verfügbaren Tastenkürzel.

User Agent | Im Untermenü **User Agent** können Sie einstellen, mit welcher Browser-Kennung sich Safari gegenüber dem Webserver ausgeben soll. Es kommt hin und wieder vor, dass Web-Designer diese Kennung überprüfen und alle Anwender, die beispielsweise nicht den Internet Explorer von Microsoft nutzen, den Aufruf der Webseite verweigern. Hiermit können Sie dem Webserver »vorgaukeln«, dass Sie den Internet Explorer verwenden. Im Untermenü **Supported Protocols** ist es möglich, Safari die Kommunikation über einige Protokolle zu untersagen. Wenn Sie das Protokoll FTP deaktivieren, dann verweigert das Programm die Kommunikation mit

X Adresszeile

Es ist auch möglich, die gerade im aktuellen Fenster angezeigte Webseite zu »überarbeiten«.

Rufen Sie eine beliebige Webseite auf und geben Sie anschließend in der Adresszeile des Browsers anstelle des URL den Befehl

```
javascript:document.designMode="On";
```

in der Adresszeile an. Die geladene Webseite kann nun von Ihnen in Safari bearbeitet werden. Mit der Eingabe

```
javascript:document.designMode="Off";
```

deaktivieren Sie den Modus zur Bearbeitung.

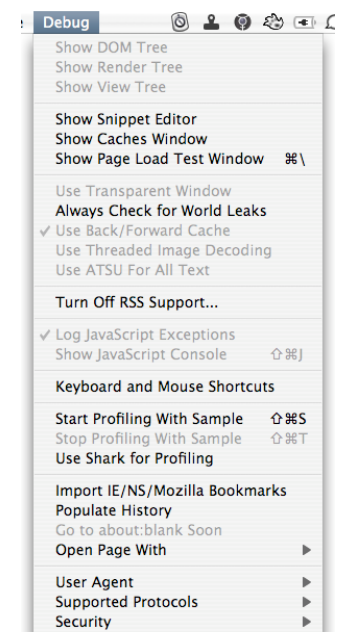


Abbildung 2: Das Debug-Menü enthält einige nützliche Funktionen.

einem FTP-Server. Bei einem Klick auf einen Link `ftp://ftp.server.tld/...` passiert nun nichts mehr.

Document Object Model | HTML-Dateien beinhalten einen hierarchischen Aufbau, insbesondere wenn das Layout nicht mit der veralteten Technik der Tabellen sondern mit Hilfe von `<div>`-Bereichen realisiert wird. Um hier Fehler aufzuspüren, ist eine Anzeige der Hierarchie nützlich. Anhand dieser Hierarchie ist leicht ersichtlich, welcher Bereich in einem anderen eingebettet wurde und gegebenenfalls im StyleSheet die Eigenschaften des übergeordneten erbt. Über den Menüpunkt **Show DOM Tree** können Sie ein Fenster aufrufen, in dem links die Hierarchie der verwendeten HTML-Tags angezeigt wird und rechts der Quellcode des gerade ausgewählten Bereichs.

JavaScript | Verwenden Sie JavaScript um ein Widget zu programmieren, dann wird Ihnen die spezielle Konsole von Safari bei der Suche nach Fehlern eine große Hilfe sein. Sie können diese Konsole – die ähnlich wie die über das gleichnamige Dienstprogramm einsehbaren Protokolle Fehlermeldungen enthält – über den Menüpunkt **Show JavaScript Console** aufrufen. In diesem Fenster werden Ihnen die Fehlermeldungen angezeigt, die bei der Auswertung und Abarbeitung etwaiger Skripten in JavaScript aufgetreten werden. Neben dem URL des Skriptes werden Sie auch über die Art des Fehlers und die Zeile, in der er aufgetreten ist, informiert.

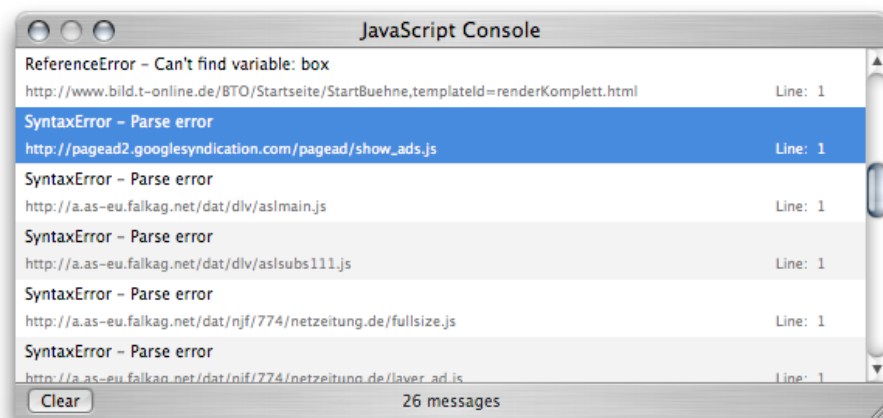
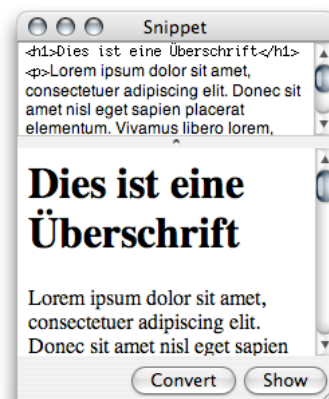


Abbildung 3: Die JavaScript Console enthält die Fehlermeldungen bei der Abarbeitung von Skripten.

3.3 Zeichnen auf dem Dashboard: `<canvas>`

Zeichenfläche | Die Arbeit mit den standardisierten HTML-Tags mag für viele aber sicher nicht alle möglichen Widgets ausreichen. Spätestens wenn Sie Daten grafisch in Form eines Graphen auswerten möchten, werden Sie an die Grenzen der Möglichkeiten stoßen. Eine mögliche Lösung, die Ihnen auch den Zugriff auf einige Funktionalitäten der Grafik-Engine Quartz ermöglicht, ist die Verwendung einer Zeichenfläche, die Ihnen Safari und auch das Dashboard in Form des HTML-Tags `<canvas>` zur Verfügung stellen. Hiermit können Sie einen rechteckigen Bereich auf der Webseite als Zeichenfläche markieren. Innerhalb dieses Bereichs können Sie eine Reihe von grafischen Operationen – vom Zeichnen einer Linie über eine quadratische Füllung bis hin einer Kurve mit Schattenwurf – ausführen.

`<canvas>` definieren | Im folgenden Listing wird im `<body>`-Bereich mit der Angabe `<canvas width="500" height="200" id="flaeche" style="border: 1px black dotted"/>` eine 500 Bildpunkte breite, 200 Bildpunkte hohe Fläche definiert. Dieser wird zur Identifikation der Name `flaeche` zugewiesen. Schließlich wird mit Hilfe der `style`-Anweisung noch ein gepunkteter Rahmen definiert. Dieses so als Zeichenfläche definiertes Rechteck kann nun über JavaScript angesprochen und »bemalt« werden. Unterhalb der Zeichenfläche finden sich zwei Buttons, die die beiden Funktionen `zeichnen()` und `loeschen()` innerhalb des Skriptes auslösen.



X Schnipsel bearbeiten

Zur schnellen Bearbeitung eines kleinen Code-Schnipsels steht Ihnen der Menüpunkt **Show Snippet Editor** zur Verfügung. Dieser öffnet ein Fenster, in dem Sie oben den HTML-Quelltext eingeben können. Mit der Schaltfläche **Show** wird Ihnen das Ergebnis des Quellcodes in der unteren Hälfte angezeigt.

X RSS deaktivieren

Mit dem Menüpunkt **Turn Off RSS Support ...** können Sie die Unterstützung von RSS-Feeds in Safari deaktivieren. In diesem Fall erscheint, wenn ein gültiger RSS-Feed gefunden wurde, in der Adresszeile der blaue RSS-Button nicht mehr.

<canvas> ansprechen | Damit Sie innerhalb von JavaScript die Zeichenfläche ansprechen können, müssen Sie zwei Variablen definieren. Dies geschieht innerhalb der beiden Funktionen mit den Zeilen

```
var canvas = document.getElementById("flaeche");
var context = canvas.getContext("2d");
```

Mit der ersten Zeile wird das als `flaeche` bezeichnete `<canvas>`-Objekt angesprochen und in der Variable `canvas` gespeichert. Der zweite Schritt besteht darin, eine weitere Variable zu definieren, die den zweidimensionalen Zeichenraum übernimmt. Die so definierte Variable `context` dient dann im folgenden Code zur Ansprache der Zeichenfläche.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
  <head>
  </head>
  <body>
  <script>
    function zeichnen () {
      var canvas = document.getElementById("flaeche");
      var context = canvas.getContext("2d");
      context.clearRect(0, 0, 500, 200);
      context.strokeStyle="000000";
      context.fillStyle="CCCCCC";
      context.shadowBlur="10";
      context.fillRect(200, 70, 150, 100);
      context.shadowBlur="0";
      context.beginPath();
      context.moveTo(200,70);
      context.lineTo(275,10);
      context.stroke();
      context.moveTo(275,10);
      context.lineTo(350,70);
      context.stroke();
    }
    function loeschen () {
      var canvas = document.getElementById("flaeche");
      var context = canvas.getContext("2d");
      context.fillStyle="FFFFFF";
      context.fillRect(0, 0, 500, 200);
    }
  </script>
  <canvas width="500" height="200" id="flaeche"
    style="border: 1px black dotted"/>
  <input type="submit" onClick="zeichnen()"
    value="Gebildezeichnen">
  <input type="submit" onClick="loeschen()"
    value="Gebilde loeschen">
</body>
</html>
```

Listing 1: Eine einfache Zeichnung auf dem Dashboard

Stile definieren | Wenn das JavaScript abgearbeitet wird, dann werden die einzelnen Befehle innerhalb der jeweiligen Funktion nacheinander auf die Zeichenfläche angewandt. Sie müssen im Prinzip zwischen zwei Arten von Befehlen unterscheiden: Der erste Typ nimmt die Änderungen vor und zeichnet zum Beispiel eine Linie oder ein Quadrat. Der zweite Typ trifft die Einstellungen für die Elemente, die gezeichnet werden. In der Funktion `zeichnen ()` wird mit der Anweisung `context.strokeStyle="000000";` festgelegt, dass die Farbe der Linie (stroke) dem RGB-Wert 000000 entspricht, die im Folgenden zu zeichnende Linie also

X Dokumentation

Die in diesem kurzen Abschnitt angesprochenen Zeichenfunktionen reißen die umfangreichen Möglichkeiten von `<canvas>` noch nicht einmal an. Neben den Strichen und Quadraten können Sie auch Kurven verwenden, Grafiken als Füllmuster verwenden, den Schattenwurf im Detail konfigurieren und noch einiges mehr. Sie finden in der Entwickler-Dokumentation (Reference-Library) im Bereich **Documentation** • **Apple Applications** • **Safari** • **Safari JavaScript Reference** eine detaillierte Beschreibung aller Funktionen und Fähigkeiten von `<canvas>`.

X Schatten: blur

Neben der Breite des Schattens können Sie über die Anweisung `.shadowColor` auch die Farbe des Schattens festlegen. Den Versatz nach links / rechts und oben / unten legen Sie mit den Anweisungen `shadowOffsetX` und `shadowOffsetY` fest.

schwarz ist. Ebenso wird mit der Anweisung `context.fillStyle="CCCCC";` festgelegt, dass Objekte mit einer Fläche weiß gefüllt werden sollen. Schließlich dient die Anweisung `context.shadowBlur="10";` dazu, dass die nun zu zeichnenden Objekte mit einem 10 Pixel breiten Schatten unterlegt werden. Dies wird im Verlauf der Funktion mit der Anweisung `context.shadowBlur="0";` wieder aufgehoben.

Rechteck zeichnen | Nachdem die Vorgaben für das zu zeichnende Quadrat getroffen wurden, wird dies mit der Anweisung `context.fillRect(200, 70, 150, 100);` realisiert. Die Anweisung `fillRect` hat die Aufgabe, einen rechteckigen Bereich zu zeichnen. In den Klammern wird mit den ersten beiden Zahlen die X- und Y-Koordinate des linken oberen Punktes des Quadrats festgelegt. In diesem Beispiel wird die linke obere Ecke 200 Pixel nach rechts eingerückt und hält vom oberen Rand der Zeichenfläche einen Abstand von 70 Pixeln. Die letzten beiden Zahlen geben die Breite (150) und die Höhe (100) vor.

Pfade verwenden | Auf dem canvas können Sie Pfade definieren, die als Parameter für eine Reihe von Operationen dienen können. Um eine Linie zu zeichnen, müssen Sie zuerst einen Pfad mit einem Start- und Endpunkt definieren. Diesen Pfad können Sie dann mit einer Linie füllen und auf diese Weise eine Linie realisieren. In Listing 1 wird innerhalb der Funktion `zeichnen()` mit der Anweisung `context.beginPath();` ein neuer Pfad erstellt. Der Startpunkt des Pfades wird mit der Anweisung `context.moveTo(200, 70);` auf den Punkt mit einem X-Wert 200 und einem Y-Wert von 70 festgelegt. Er entspricht also der linken oberen Ecke des zuvor gezeichneten Quadrats. Nun wird ausgehend von diesem Startpunkt mit `context.lineTo(275, 10);` eine Linie in diesem Pfad zu dem Punkt mit dem X-Wert 275 und dem Y-Wert 10 gezogen. Diese Linie innerhalb des Pfades ist zunächst fiktiv und nicht sichtbar. Zu einem sichtbaren Strich wird Sie mit der Anweisung `context.stroke();`, die diese fiktive Linie zeichnet. Die folgenden Befehle dienen dazu, eine zweite Linie zu zeichnen, die vom Endpunkt der ersten Linie zur rechten oberen Ecke des Quadrates reicht. Auf diese Weise wurde also eine Art Dach gezeichnet (siehe Abbildung 3).

3.4 RSS-Feeds anpassen

Eigene Farben und Bilder | Mit der Einführung von 10.4 ist Safari auch in der Lage, RSS-Feeds anzuzeigen. Hierbei werden die Überschriften und Zusammenfassungen neuer Inhalte von einigen Webseiten in einer speziellen XML-Datei zusammengefasst, die sich innerhalb der RSS-Anzeige in komprimierter Form (siehe Abbildung 4) anzeigen lassen.



Abbildung 4: Safari ist mit der Version 10.4 auch in der Lage, als RSS-Reader zu fungieren.

Zur Anzeige dieser RSS-Feeds setzt Safari wiederum auf Web-Standards. Genau genommen handelt es sich bei der Anzeige in Abbildung 4 um eine normale Webseite, in die Safari die Inhalte des RSS-Feeds an passender Stelle eingefügt hat. Diese Webseite dient als eine Vorlage. Sie finden im Ordner `/System/Library/PrivateFrame-`

Xfunction loeschen()

Die in diesem Beispiel verwendete Funktion `loeschen()` hat einfach nur die Aufgabe, die gesamte Zeichenfläche mit weißer Farbe zu übermalen und somit optisch in den Ursprungszustand zu versetzen.

WARNUNG

Sie sollten bei der Anpassung der CSS-Dateien für die Darstellung der RSS-Feeds immer eine Sicherheitskopie des gesamten Ordners vorhalten. Es ist gut möglich, dass Änderungen zu unerwarteten Ergebnissen und auch Abstürzen führen können. Ob der Name **Themes** des übergeordneten Ordners auf eine zukünftige Fähigkeit von Safari hindeutet, steht nicht sicher fest.

works/SyndicationUI.framework/Versions/A/Resources/Themes/default.syndt-heme/ Contents/Resources eine Reihe von Dateien, die das Aussehen der einzelnen Elemente bestimmen. So dient die Datei **Article.syntmpl** in diesem Ordner als Vorlage für die Anzeige der einzelnen Artikel. Die für das Aussehen (Farben, Schriftarten, ...) zentrale Datei ist **Articles.css**. Wenn Sie sich gut auf die Arbeit mit HTML, JavaScript und Cascading StyleSheets verstehen, können Sie diese Dateien nach Ihrem Geschmack anpassen und auch die Grafiken austauschen. Beachten Sie hierbei allerdings, dass diese Manipulation vergleichsweise tief in das Betriebssystem eingreift.

3.5 Der Zeit voraus: das WebKit

Nightly Builds Die Arbeit an dem Kern von Safari, dem für die Anzeige von HTML-Dateien zuständigen WebKit, ist mittlerweile Bestandteil der unter <http://www.opendarwin.org> zusammengefassten Open Source Projekte. Der Quellcode des WebKits ist mittlerweile wie auch schon bei seinem Stammvater Konqueror Open Source und steht allen Interessierten offen. Ein Resultat dieser Bestrebungen sind auch die Nightly Builds, die tagesaktuell den Stand der Entwicklungen und Änderungen im Quellcode des WebKit in einem ausführbaren Programm zusammenfassen. Wenn Sie sich für die zukünftigen Entwicklungen und Neuerungen von Safari interessieren, dann können Sie unter <http://webkit.opendarwin.org> sich eine solchen Nightly Build herunterladen. Dieser enthält in der Regel ein mit WebKit bezeichnetes Programm, das Safari im Aussehen und der Bedienung weitgehend entspricht. Indes wird bei diesem WebKit die Berechnung von HTML-Seiten mit dem aktuellen Stand des WebKit vorgenommen und alle Änderungen und Neuerungen sind hier (testweise) bereits aktiv. So ist das WebKit bereits in der Lage, ohne eine zusätzliche Erweiterung von Adobe Scalable Vector Graphics (SVG) anzuzeigen. Eine weitere, für Web-Designer sehr nützliche Funktion, ist der **Web Inspector**. Diesen können Sie über das Kontextmenü aufrufen und er zeigt Ihnen Details zu dem ausgewählten HTML-Abschnitt in einem schwebenden Fenster (siehe Abbildung 5) an.

X Surfin' Safari

Wenn Sie sich für die Kommentare, Anmerkungen und Ergebnisse eines bei Apple angestellten Entwicklers interessieren, dann ist das Weblog von Dave Hyatt sicherlich einen Blick wert. Hyatt ist einer der Hauptentwickler von Safari und berichtet unter <http://webkit.opendarwin.org> in seinem eigenen Weblog über seine Erfahrungen und den aktuellen Stand der Entwicklungen.

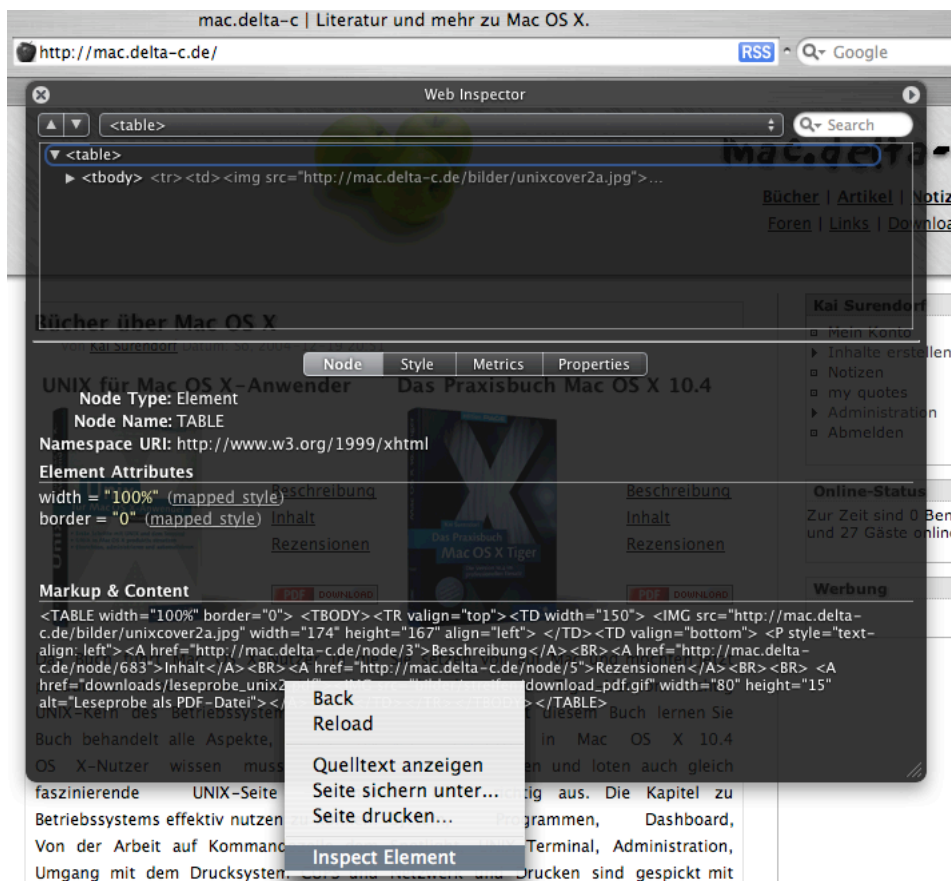


Abbildung 5: Der Inspector erleichtert die Analyse von HTML-Code.