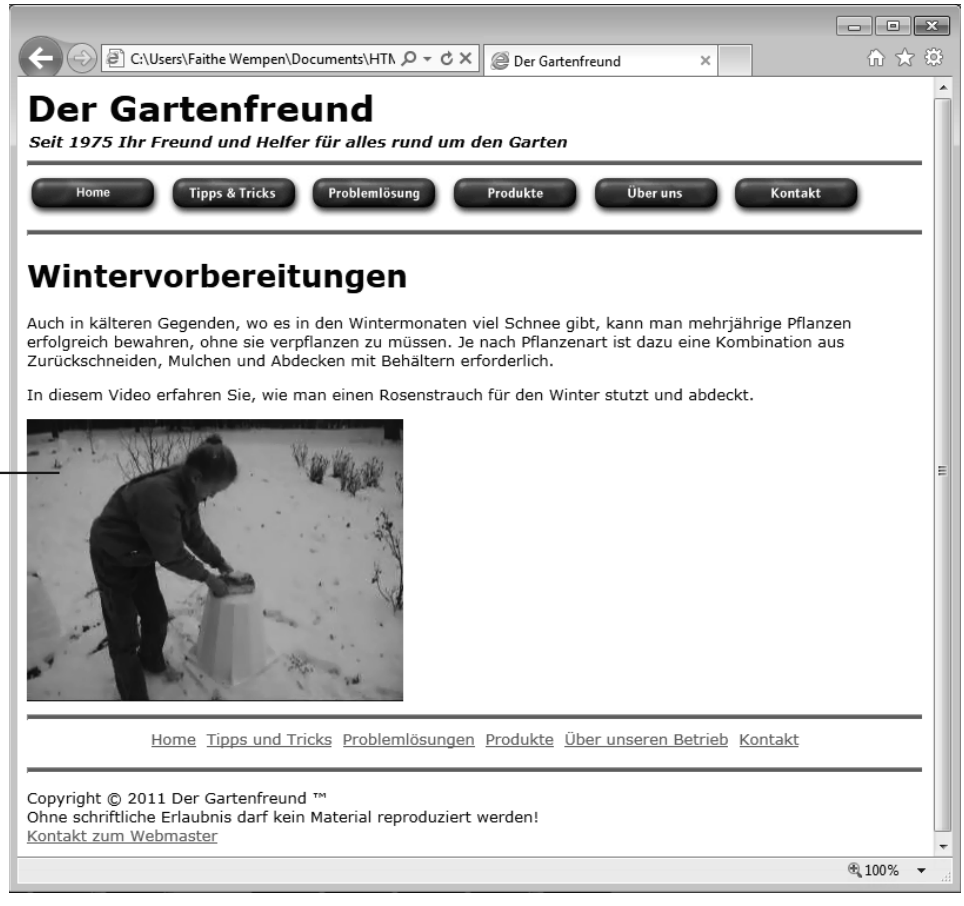
Kapitel 15

Audio und Video einbinden

In diesem Kapitel:

Faithe Wempen: HTML 5 - Schritt für Schritt, Microsoft Press 2011 (ISBN 978-3-86645-550-4)

Der Inhalt auf einen Blick



Videos abspielen

Seite 302

In diesem Kapitel erfahren Sie, wie Sie

- mit den neuen <audio>- und <video>-Tags in HTML5 arbeiten
- Multimediadateien abspielen sowie Formate und Codecs wählen
- das <video>-Tag einsetzen
- das <audio>-Tag einsetzen

Es ist nicht ganz so leicht wie anderes im Web, wenn man über das Internet Video- und Audiodateien abspielen will. Das liegt an der großen Vielzahl von Formaten, die über konkurrierende Anbieter und Open Source-Gruppen verfügbar sind. Diese Formate werden in unterschiedlichem Maße von den modernen Browsern unterstützt und von älteren Browser manchmal gar nicht. Zusammen machen es diese Faktoren zu einer Herausforderung, wenn man Audio- und Videoinhalte anbieten will, die für alle Besucher gleichermaßen gut abspielbar sind.

Die nun in HTML5 integrierten <audio>- und <video>-Tags vereinfachen dies ein wenig. Im Laufe der Zeit wird es einfacher, solche Multimediadateien abzuspielen, weil neuere Browser die Tags besser unterstützen und die Benutzer im Allgemeinen ältere Browser aktualisieren. Allerdings wird es noch eine Weile erforderlich sein, dass Sie Ihre Multimediadateien in mehreren Formaten kodieren und anbieten.

Siehe auch Brauchen Sie nur einen schnellen Überblick über die Themen dieses Kapitels? Schauen Sie sich die Zusammenfassung am Ende dieses Kapitels an.

Übungsdateien Bevor Sie mit den zu diesem Kapitel gehörigen Übungsdateien arbeiten können, müssen Sie sie von der zu diesem Buch gehörigen Website herunterladen und installieren. Im Abschnitt »Der Einsatz der Übungsdateien« in der Einleitung dieses Buches erfahren Sie alles Wesentliche darüber.

Was bringt HTML5 bei Audio und Video Neues?

Wenn Entwickler und Designer auf ihren Webseiten Video und Audio abspielen wollen, bedienen sie sich traditionell meist der Flash-Technologie von Adobe. Sites wie YouTube (*www.youtube.de*) betten Videos in eine Flash-Datei ein. Somit muss der Benutzer den Adobe Flash Player auf seinem System installiert haben.

Die HTML5-Spezifikation führt nun eine Alternative ein: das Standard-Tag <video>, mit dem Videoinhalte abgespielt werden können. Doch auch das <video>-Tag benötigt eine Videodatei und dass die Benutzer auf ihren Computern einen entsprechenden Player installiert haben.

Audioclips werden auf Internetseiten traditionell mit dem <object>- oder <embed>-Tag bereitgestellt. Bei HTML5 gibt es dafür das <audio>-Tag.

Bei Erscheinen dieses Buches werden <audio>- und <video>-Tags in den Browsern nur begrenzt unterstützt. Was die Dinge noch verkompliziert: Man muss abhängig davon, was der Browser des Benutzers abspielen kann, verschiedene Formate für die Videos vorhalten.

In diesem Kapitel lernen Sie die neuen <audio>- und <video>-Tags kennen und bekommen Informationen darüber, wie mit den Schwierigkeiten der Video-Kompatibilität umzugehen ist. Bevor wir hier näher einsteigen, sollten Sie wissen, dass diese beiden neuen Tags mit Stand dieser Niederschrift nur von folgenden Browsern unterstützt wird:

- Internet Explorer 9+
- Firefox 3.5+
- Safari 3+
- Google Chrome 3+
- Opera 10.5+
- iPhone 1+
- Android 2+

Browser, die diese Tags nicht unterstützen, ignorieren sie, aber wenn Sie wollen, dass Ihre Audio- oder Videoinhalte von möglichst vielen Browsern (alt und neu) abgespielt werden können, müssen Sie das auch ohne diese Tags hinbekommen können. Dieses Kapitel stellt die alten und auch die neuen Methoden vor.

Multimedia-Grundlagen von HTML

Bevor wir uns im Detail mit dem Erstellen von Internetseiten mit Multimedia-Inhalten beschäftigen, sollten Sie ein paar grundlegende Dinge darüber wissen, wie HTML5 – und auch frühere Versionen von HTML – Audio- und Videoclips präsentiert.

Am häufigsten werden Multimedia-Inhalte im Internet bereitgestellt, indem in der Webseite eine Audio- oder Videodatei *eingebettet* wird. Dann wird sie innerhalb der Seite selbst abgespielt, wenn der Besucher auf eine Schaltfläche klickt. Damit das funktioniert, müssen die Besucher mit einem Browser auf die Site kommen, der den bereitgestellten Typ Audio- oder Videodatei unterstützt, oder ein Plugin (ein Hilfsprogramm) herunterladen und installieren, um die Fähigkeiten ihres Browsers zu erweitern, damit er diese Datei abspielen kann. Falls Ihr Zielpublikum den Microsoft Internet Explorer 5.5 und höher nutzt, können Sie dafür mit dem <object>-Tag arbeiten, anderenfalls nehmen Sie das <embed>-Tag. Falls Ihre Besucher einen HTML5-konformen Browser haben, können Sie dafür auch die neuen <audio>- und <video>-Tags nehmen.

Als Alternative können Sie einen *Link* zu einem Audio- oder Videoclip setzen, damit er in einer externen Anwendung (z.B. dem Microsoft Windows Media Player) abgespielt wird, wenn der Besucher auf diesen Hyperlink klickt. Damit das funktioniert, braucht der Besucher eine externe Anwendung, die die von Ihnen angebotene Art Audio- oder Videodatei unterstützt, oder er muss extra ein Programm herunterladen und installieren. Diese Technik funktioniert in allen Browsern, was ein Vorteil ist. Nehmen Sie für die Verknüpfung das <a>-Tag wie für andere Hyperlinks auch. Das sieht dann beispielsweise so aus:

```
<a href="meinsong.mp3">Meinen Song abspielen!</a>
```

Dieses Kapitel konzentriert sich hauptsächlich auf die Einbettung von Multimedia-Präsentationen.

Multimedia-Formate und -Container

Die Ausführungen von Multimedia im Internet müssen damit beginnen, die verschiedenen Formate zu erläutern. Wenn man über Videodateien spricht, meint man meistens Dateien mit der Erweiterung *.avi*, *.mp4* oder *.mkv*. Diese Erweiterungen verweisen einfach auf das Containerformat der Videodatei selbst, zeigen aber nicht an, in welchem Format das Video kodiert wurde.

Es gibt eine Reihe weit verbreiteter Containerformate wie Ogg (*.ogv*), Flash Video (*.flv* oder *.f4v*), das bereits erwähnte Audio Video Interleave (*.avi*), MPEG-4 Part 14 (*.mp4*), Matroska (*.mkv*) und noch viele andere. Eine Übersicht der verschiedenen Containerformate finden Sie unter *http://de.wikipedia.org/wiki/Containerformat*.

Zusätzlich enthalten Videodateien fast immer Audiospuren. Die Containerdatei enthält sowohl die Video- als auch die Audiokomponenten.

Es gibt auch ein neues Format namens WebM, das ähnlich ist wie Matroska. WebM ist ein Open Source Video Containerformat, das wahrscheinlich bald recht beliebt sein wird, da es von Google unterstützt wird. WebM soll exklusiv mit dem Videocodec VP8 und dem Vorbis-Audiocodec eingesetzt werden (mehr über Codecs im nächsten Abschnitt).

Mit Codecs Video und Audio dekodieren

Wenn ein Produzent (also die Person oder Organisation, die die Audio- oder Videodatei zur Verfügung stellt) Multimedia-Inhalte kodiert, wählt er das Format, in dem die Datei kodiert wird. Wer dieses Video anschauen oder die Audiodatei anhören will, muss auf seinem Computer die entsprechende Kodierungssoftware (den so genannten *Codec*) installiert haben, um die Datei abspielen zu können.

Sie treffen in diesem Kapitel und auch anderen Veröffentlichungen über Video und Audio auf das Wort Codec. Das Wort selbst ist eine Abkürzung von *encode/decode* (oder abhängig davon, wen Sie fragen, auch *decode/encode*), also Kodieren/Dekodieren. Der Codec bezieht sich darauf, wie die Audio- oder Videodatei kodiert oder formatiert wurde. Zum Dekodieren einer Audio- oder Videodatei muss der Computer einen Algorithmus nutzen, um die kodierten Audio- bzw. Videoinhalte in eine für Menschen verständliche Form zu dechiffrieren.

Jetzt kommt der Browser ins Spiel: Er muss wie z.B. der Internet Explorer das Format unterstützen oder braucht ein Plugin, damit er die Audio- bzw. Videodatei abspielen kann. Zum Glück werden heute alle üblichen Formate und Codes entweder nativ unterstützt oder in Form eines Plugins für die bekannten Browser abspielbar gemacht. In dem Maße, wie die neueren Browser HTML5 immer mehr unterstützen, wird der Bedarf an Plugins von Drittanbietern (zumindest für Audio und Video) hoffentlich immer mehr eine Sache der Vergangenheit.

So wie es eine ganze Reihe von Containerformaten gibt, gibt es auch mehrere übliche Formate zur Videokodierung. Zu den bekanntesten Formaten gehören H.264, VP8, DivX, Theora usw. Wenn Sie vorhaben, im Internet viel mit Video anzubieten, müssen Sie sich wahrscheinlich mit mehreren unterschiedlichen Formaten und Containern vertraut machen, um das größtmögliche Zielpublikum zu erreichen.

Genauso wie bei Video ist fürs Abspielen von Audiodateien über einen Computer oder ein Handheld-Gerät wie ein SmartPhone ein Codec erforderlich, mit dem die Datei gelesen und abgespielt wird. Zwei bekannte Formate sind MPEG-4 Audio Layer 3, das Sie wahrscheinlich als MP3 kennen, und AAC, das meist von Apple verwendet wird. Ein weiteres Format ist Vorbis, das häufig in Ogg-Containern eingesetzt wird.

Viele der Videoformate unterstützen *Profile*, bei denen es sich im Wesentlichen um die Parameter handelt, die zur Kodierung des Videos verwendet wurden. Ein High Profile H.264-Video bietet beispielsweise eine höhere Qualität, aber dafür ist die Datei weitaus größer – zu groß, um allgemein im Internet eingesetzt zu werden. Für den Augenblick reicht es zu wissen, dass verschiedene Profile existieren und für unterschiedliche Anwendungen passen.

Die Wahl des Formats

Hört sich das alles ziemlich komplex an? Stimmt genau. Es ist nicht nur recht kompliziert, unter all den vielen Formaten überhaupt eine Auswahl zu treffen, und dann ist noch nicht mal garantiert, dass alle Besucher dieses Format überhaupt abspielen können. Auf höherer Ebene ist Audio leichter als Video, also werden Sie im Grunde Ihre Energie in die Arbeit mit Videoformaten stecken.

Wie wählen Sie nun das richtige Format aus? Die Antwort lautet, dass nicht bloß eins, sondern drei oder vier zu wählen sind. Das Ziel ist letztendlich, einem möglichst großen Personenkreis das Video verfügbar zu machen. Als Konsequenz müssen Sie also eine Videodatei in verschiedene Formate konvertieren, damit auch wirklich möglichst viele von Ihrem Angebot profitieren.

Tabelle 15.1 zeigt die drei wichtigsten Container, die neben Flash noch zum Einsatz kommen.

Container	Videocodec	Audiocodec
Ogg	Theora	Vorbis
mp4	H.264	AAC
WebM	VP8	Vorbis

Tabelle 15.1 Bekannte Videoformate im Internet

Momentan unterstützt Microsoft Internet Explorer 9 zwar das `<video>`-Tag, aber nur das Format H.264. Frühere Versionen von Internet Explorer unterstützen es nicht, aber keine Sorge: Sie erfahren gleich, wie diese Einschränkung zu umgehen ist.

Mozilla Firefox Version 3.5 und höher unterstützt die Container WebM und Ogg. Safari liest H.264 Video und AAC Audio in einem mp4-Container. Opera arbeitet sowohl mit WebM als auch mit Ogg. Der Ogg-Container wird mit großer Wahrscheinlichkeit Theora-Video und Vorbis-Audio unterstützen.

Dateigröße und Qualität

Bei einem Videoclip meint das Wort »Größe« zweierlei: die Dateigröße und die Darstellungsgröße (also die Zahl der vertikal bzw. horizontal dargestellten Pixel). Wie Sie sicher erwartet haben, hängen diese beiden Faktoren zusammen: Je größer das Darstellungsfenster des Clips, desto größer auch die Datei. Ein Clip auf einer Webseite muss nicht den gesamten Bildschirm ausfüllen; meist reicht ein sechs bis zehn Zentimeter großes Fenster aus.

Die Darstellungsgröße ist allerdings nicht der einzige bestimmende Faktor der Dateigröße. Manche Formate führen zu kleineren Dateien als andere, weil sie mit unterschiedlich starker Komprimierung arbeiten. Ein Videoclip wird mit einem bestimmten *Komprimierungsalgorithmus* verkleinert. Diese Gruppe von mathematischen Formeln wird verwendet, um zur ressourcenschonenderen Speicherung überflüssigen Platz im Clip zu entfernen. Um einen komprimierten Clip abzuspielen, muss der Computer über einen entsprechenden Codec verfügen.

HINWEIS Ein Kompressionsalgorithmus arbeitet so, dass er die Wiederholung von Zeichen oder bestimmten Mustern in der Datendatei erkennt und sie durch kompaktere Codes ersetzt. Ein Algorithmus würde z. B. diese Folge von Nullen 00000000000000000000 durch 20*0 ersetzen.

Außerdem unterscheiden sich Videoclips noch bezüglich der Frames pro Sekunde (fps), also der pro Sekunde dargestellten Bilder. Mehr Frames bedeuten ein flüssigeres Abspielen und größere Dateien. Ein VHS-Videoband wird mit 30 fps aufgenommen, aber für die Nutzung im Internet reicht eine Framerate von 15 fps aus, weil das zu einer deutlich kleineren Datei führt. Sie können bei der Aufnahme des Clips die Anzahl der Frames pro Sekunde einstellen oder mit dem Programm eines Drittanbieters die Frames in einem bereits aufgezeichneten Clip reduzieren.

Wenn ein Audioclip digitalisiert (also in ein digitales Format verwandelt) wird, wird pro Sekunde eine Serie von Audio-»Schnappschüssen« erstellt. Diese Schnappschüsse nennt man *Samples* (etwa: *Abtastwerte*). Höhere *Sampling-Raten* (also die Zahl der Samples pro Sekunde) führen zu einer besseren Genauigkeit bei der Klangwiedergabe, aber auf Kosten von größeren Dateien. Sampling-Raten für Audioclips werden in Kilohertz angegeben, also beispielsweise 11 KHz, 22 KHz oder 44 KHz.

HINWEIS »Kilo« steht für Tausend. Ein Clip mit 11 KHz enthält also schätzungsweise 11.000 Samples pro Sekunde.

Audioclips haben auch unterschiedliche *Sampling-Raten*. Damit ist gemeint, mit wie vielen Bits jedes Sample beschrieben wird. Übliche Sampling-Auflösungen sind 8 Bit, 16 Bit und 32 Bit. Je mehr Bits gesampelt werden, desto größer wird die Datei.

Audioclips werden entweder in Stereo oder Mono aufgenommen (damit ist die Zahl der Audiokanäle in der Aufnahme gemeint). Bei Mono wird nur ein Kanal abgespielt, aber in beiden Lautsprechern. Stereo verwendet zwei Kanäle, die dann im linken bzw. rechten Lautsprecher wiedergegeben werden. Stereoclips sind etwa doppelt so groß wie Monoclips.

Bei der Aufzeichnung von Audioclips können Sie meist zwischen verschiedenen Sampling-Raten und Auflösungen wählen. Hier folgt eine Auflistung üblicher Einstellungskombinationen:

Einstellung	Qualität
8 KHz, 8 Bit, Mono	Telefonqualität
22 MHz, 16 Bit, Stereo	Radioqualität
44 KHz, 16 Bit, Stereo	CD-Qualität

Video kodieren

Nun haben Sie einen besseren Überblick über die Wiedergabe von Audio und Video im Internet, aber fragen sich gewiss, wie Sie Ihre Lieblingsvideos aus dem Urlaub in diese drei Formate (vier, wenn man Flash mitzählt) konvertieren. Sie können gleich mit den für diese Übung bereitgestellten Clips arbeiten, aber eigene Videos müssen erst noch entsprechend vorbereiten.

So komplex wie die Wiedergabe ist auch die Kodierung. Meist wird bei der Konvertierung von Videos verschiedene Software kombiniert. Die Software Handbrake ist beispielsweise sehr bekannt für die Videokonvertierung von H.264 ins AAC-Format zur Wiedergabe auf Apple-Geräten und auch für die Konvertierung von Videos im Internet.

Wenn man ein Video mit Vorbis-Audio in Ogg Theora konvertieren will, geht das anhand verschiedener Softwarepakete wie ffmpeg2theora, VLC Media Player, Firefogg (ein Plugin für Firefox) und anderen. Firefogg, ffmpeg und einige andere können auch ins WebM-Format konvertieren.

TIPP Wenn Sie mit Firefox arbeiten (oder Videos kodieren wollen), geht das mit VLC ganz einfach und effektiv. Aber üben Sie sich schon mal in Geduld. Die Videokonvertierung in verschiedene Formate ist ein langwieriger Prozess. Beim Schreiben dieses Kapitels habe ich mit VLC alle Konvertierungen vorgenommen.

Wenn Ihr Ziel ist, die Videos Ihrer Site möglichst breit verfügbar zu machen, sollten Sie die Videodateien in jedes der drei Formate und außerdem in Flash konvertieren. So sind Sie auf der sicheren Seite, da bei den neuen Browsern der Support für die neuen `<audio>`- und `<video>`-Tags integriert ist, und die älteren Browser nehmen für die Videos einfach Flash.

Videoclips einbetten

Nun haben Sie bisher eine ganze Menge Hintergrundmaterial über etwas erfahren, das doch scheinbar ganz einfach sein sollte! Und dabei haben wir hier nur die Oberfläche angekratzt. Dieser Abschnitt zeigt, wie man mit dem `<video>`-Tag ein Video auf einer Seite platziert und bei Bedarf auch noch auf Flash zurückgreifen kann.

Das <video>-Tag

Im Grund sieht das `<video>`-Tag so aus:

```
<video src="meinvideo.ogv"></video>
```

Es gibt mehrere Attribute und verschiedene Möglichkeiten, das `<video>`-Tag zu nutzen, um es für Ihre Bedürfnisse und die Ihres Publikums zu konfigurieren. Mehrere Attribute helfen Ihnen dabei, und zwar:

- `autoplay`
- `controls`
- `height`
- `loop`
- `preload`
- `width`

Es ist nicht überraschend, dass die Attribute `width` und `height` für Höhe und Breite der Videodarstellung im Fenster stehen:

```
<video src="meinvideo.ogv" width="320" height="240"></video>
```

Das Attribut `controls` legt fest, ob Standardelemente für die Wiedergabesteuerung im Browser sichtbar sein sollen. Das ist praktisch, und ich kann es nur empfehlen. Wenn Sie auf dieses Attribut verzichten, kann der Besucher das Video nicht noch einmal anschauen, ohne die gesamte Seite neu zu laden. Sehr lästig! Hier ein Beispiel des `controls`-Attributs:

```
<video src="meinvideo.ogv" controls></video>
```

Dem `preload`-Attribut entnimmt der Browser, dass er sofort mit dem Download des Videos beginnen soll, sobald er auf dieses Element trifft. Wenn das Video das zentrale Thema der Seite ist und wahrscheinlich die meisten oder alle Besucher es anschauen wollen, dann sollten Sie die `preload`-Option wählen. Falls dieses Element hingegen nur ein kleiner Teil der Seite ist und sich kaum einer das Video anschaut, dann ist das Vorabladen des Videos nur Bandbreitenverschwendung. Hier ist das `preload`-Attribut in Aktion:

```
<video src="meinvideo.ogv" preload></video>
```

Mit dem Attribut `loop` startet der Browser das Video nach dem Ende sofort erneut:

```
<video src="meinvideo.ogv" loop></video>
```

Schließlich wird das Video mit dem Attribut `autoplay` automatisch gestartet, wenn die Seite geladen wird. Vom Standpunkt der Benutzerfreundlichkeit ist das im Allgemeinen eine ganz schlechte Idee. Die meisten Benutzer wollen die Kontrolle über das Video haben; es soll abgespielt werden, wenn sie sich darauf konzentrieren können. Und auch wenn das Attribut `autoplay` aktiviert ist, könnten Ihre Besucher diese Option im Browser deaktiviert haben. Aus diesem Grund und wegen der Benutzerfreundlichkeit empfehle ich, dieses Attribut nicht einzusetzen – mit einer Ausnahme: Wenn Sie das Attribut `controls` nicht einbauen, müssen Sie `autoplay` integrieren; anderenfalls wird das Video nicht abgespielt und die Besucher können es gar nicht starten. Hier ist ein Beispiel für `autoplay`:

```
<video src="meinvideo.ogv" autoplay></video>
```

In einem echten Videoelement sieht das wie folgt aus:

```
<video src="meinvideo.ogv" width="320" height="240" controls></video>
```

Die bisherigen Beispiele funktionieren alle gut, wenn Ihre Besucher einen Browser wie Firefox 3.5 oder Opera 10.5 oder höher nutzen. Doch was ist, wenn jemand mit Internet Explorer zu Ihnen surft? In diesem Fall muss das Video so kodiert sein, dass es in Internet Explorer abgespielt werden kann. Mit dem `<video>`-Tag können Sie über das `source`-Element mehr als eine Quelle angeben. So verknüpfen Sie mehrere Versionen eines Videos. Sie können dem Browser anhand des `type`-Attributs auch ein wenig mehr über die verknüpfte Videodatei verraten. Nach dem vorigen Beispiel sähe ein `<video>`-Tag, das den Ogg-Container sowie ein H.264-Video in einem mp4-Container und einen WebM-Container enthält, so aus:

```
<video width="320" height="240" controls>
<source src="meinvideo.mp4" type="video/mp4">
<source src="meinvideo.ogv" type="video/ogg">
<source src="meinvideo.webm" type="video/webm">
</video>
```

Zusätzlich kann ein optionaler Codec-Bereich des type-Attributs dem Browser auch verraten, mit welchen Audio- und Videoabschnitten die Datei arbeitet. Näheres über den Einsatz der Codec-Option würde aber den Rahmen dieses Buches sprengen.

Mit diesen beiden Optionen sind nun der Internet Explorer 9 und Safari versorgt (dank des mp4-Containers), Firefox ebenso wie Google Chrome (dank des Ogg-Containers) und auch andere Browser (dank des WebM-Containers).

Das <embed>-Tag – Ihr persönlicher Plan B

Aber was passiert, wenn auf Ihrer Site jemand mit einem älteren Browser erscheint, der kein HTML5 »spricht«? In diesem Fall kann er sich über das <video>-Tag das Video nicht anschauen. Zum Glück ignorieren ältere Browser das <video>-Tag einfach, also verursacht dessen Anwesenheit keine Störungen. Allerdings müssen Sie sich nun für solche Leute überlegen, wie sie an das Video kommen.

Sie werden merken, dass bei jenen, die den Internet Explorer nutzen, meist Adobe Flash installiert ist. Also halten Sie auf Ihrer Seite einfach auch eine Flash-Version des Videos vor. Schreiben Sie einfach mithilfe des <embed>-Tags ein zusätzliches Element in die Seite. Adobe Flash kann ein mit H.264 kodiertes Video mit AAC-Audio abspielen, und somit braucht das Video nicht noch in ein weiteres Format konvertiert zu werden. Hier folgt ein Beispiel:

```
<embed src="meinvideo.mp4" type="application/x-shockwave-flash" width="320" height="240"
allowscriptaccess="always" allowfullscreen="true">
```

Videoclips auf einer Webseite integrieren

Nach diesem theoretischen Exkurs machen wir uns nun an die praktische Umsetzung.

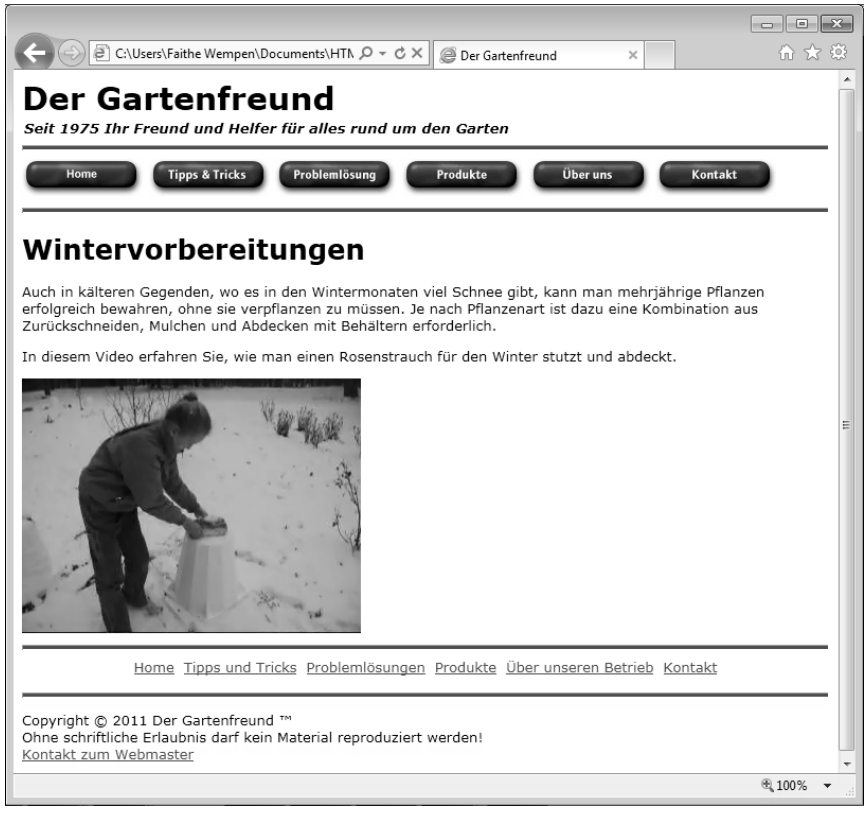
In dieser Übung betten Sie ein Video mit dem <video>-Tag in eine HTML-Seite ein und bieten über das <a>-Tag eine alternative Version als Download-Verknüpfung an. Sie erfahren auch, wie man den Clip über das <embed>-Tag einfügt.

Vorbereitung Arbeiten Sie bei dieser Aufgabe mit den Dateien *winter.html*, *myvideo.mp4*, *myvideo.webm* und *myvideo.ogv* im Ordner der Übungsdateien. Diese Dateien befinden sich im Ordner *Dokumente\Microsoft Press\HTML5 SBS\15Video\AddVideo*.

1. Öffnen Sie die Datei *winter.html* in Editor und Internet Explorer 9 (oder einem anderen HTML5-konformen Browser).
2. Im *#main*-Bereich tragen Sie direkt vor dem schließenden -Tag den Code fürs Einfügen des Videos ein.

```
<p>In diesem Video erfahren Sie, wie man einen Rosenstrauch für den Winter stutzt und abdeckt.</p>
<video width="320" height="240" autoplay controls>
<source src="myvideo.mp4">
<source src="myvideo.webm">
<source src="myvideo.ogv">
</video>
</div>
```

3. Aktualisieren Sie den Internet Explorer, um sich auf der Webseite den Clip anzuschauen.



Sie sehen das Video, das automatisch starten sollte. Falls nicht, ist Ihr Browser möglicherweise nicht HTML5-konform.

4. Kehren Sie zu Editor zurück. Direkt vor dem schließenden `</video>`-Tag schreiben Sie das `<embed>`-Tag, damit der Clip mit Flash abgespielt werden kann.

```
<p> In diesem Video erfahren Sie, wie man einen Rosenstrauch für den Winter stutzt und abdeckt.</p>
<video width="320" height="240" autoplay controls>
<source src="myvideo.mp4">
<source src="myvideo.webm">
source src="myvideo.ogv">
<embed src="myvideo.mp4" type="application/x-shockwave-flash" width="320" height="240"
allowscriptaccess="always" allowfullscreen="true">
</video>
</div>
```

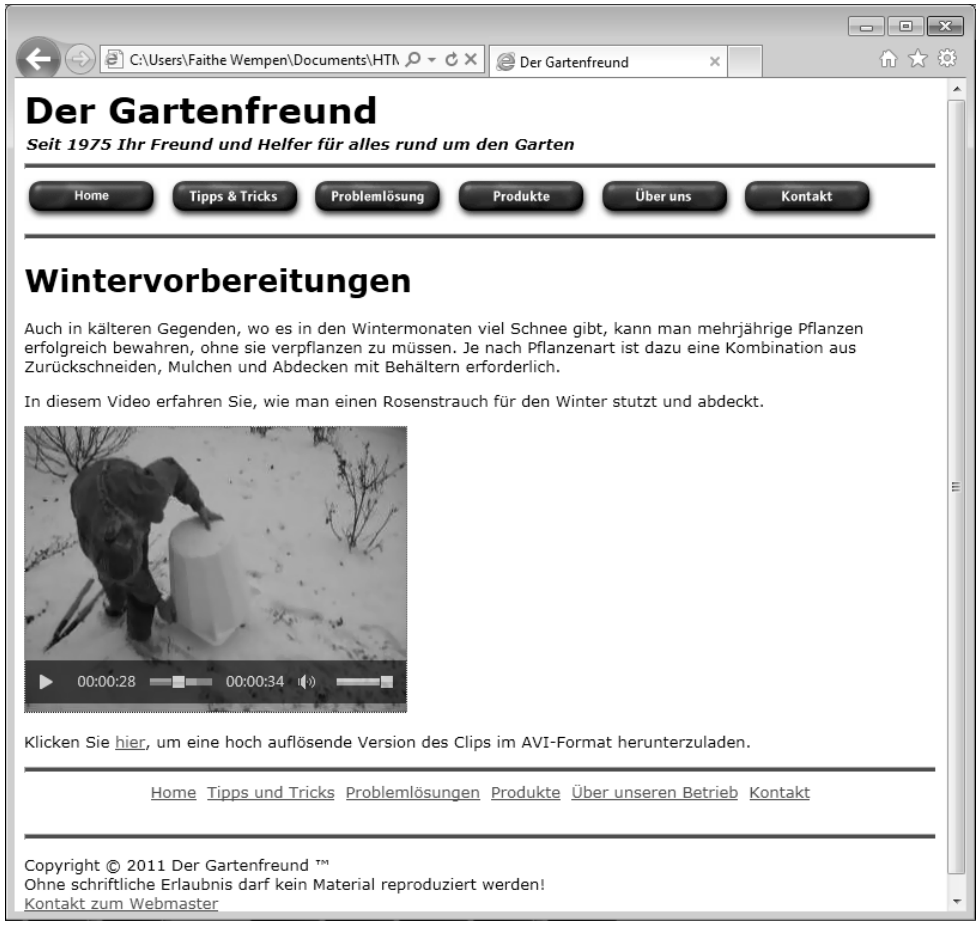
5. Geben Sie nach dem <embed>-Tag Folgendes ein:

```
<p>Klicken Sie hier, um eine hoch auflösende Version des Clips im AVI-Format herunterzuladen.</p>
```

6. Verwandeln Sie das Wort *hier* in einen Hyperlink, der mit der Datei *myvideo.avi* verknüpft ist.

```
<p>Klicken Sie <a href="myvideo.avi">hier</a>, um eine hoch auflösende Version des Clips im AVI-
Format herunterzuladen.</p>
```

7. Speichern Sie Ihre Arbeit in Editor und aktualisieren Sie den Internet Explorer, um die Änderungen zu sehen.



HINWEIS Falls im Browser-Fenster eine Sicherheitswarnung erscheint, müssen Sie diese vielleicht auf einer Schaltfläche bestätigen, damit das Flash-Video abgespielt werden kann.

Aufräumen Schließen Sie das Fenster von Editor und Internet Explorer.

Audiodateien auf einer Webseite integrieren

Die gute Nachricht ist, dass Sie durch all die Beschäftigung mit den Infos über Videos in diesem Kapitel bereits das Meiste wissen, was für das Bereitstellen von Audiodateien auf einer Webseite nötig ist. Die schlechte lautet allerdings, dass Sie auch bei Audio auf die gleichen Probleme mit Formatierung und Kodierung treffen, die bei Video im Internet so lästig sind – außer dass die Audioprobleme noch etwas schlimmer sind. In diesem Abschnitt untersuchen wir das `<audio>`-Tag und dessen Alternativen.

Audiodateien per <audio>-Tag abspielen

Vielleicht nehmen Sie an, dass Audiodateien auf einer Webseite einfacher abzuspielen sind als Videos, was aber leider größtenteils nicht zutrifft. Sie müssen sich immer noch um die unterschiedlichen Vorlieben der Browser kümmern und die Audiodateien in verschiedene Formate kodieren. Außerdem brauchen Ihre Besucher meistenteils immer noch spezielle Audio-Plugins. Nach dieser Vorrede nun die Feststellung, dass das `<audio>`-Tag in HTML5 neu ist, und damit – vorausgesetzt, die Browser-Hersteller kommen zu irgendeiner Übereinkunft (was in etwa so wahrscheinlich ist wie mein Hauptgewinn in der Lotterie) – sollte die Audiowiedergabe im Internet einfacher werden.

Wie `<video>` unterstützt auch das `<audio>`-Tag mehrere Quellen. Da es kein übliches Format gibt, müssen Sie die Audiodateien mehrfach kodieren, um eine möglichst große Abdeckung für Ihr Publikum zu erzielen. Wie das `<video>`-Tag unterstützt auch `<audio>` Attribute wie `controls`, `autoplay`, `loop` und `preload`. Somit ist die Syntax für `<audio>` im Prinzip die Gleiche wie beim `<video>`-Tag.

TIPP Zur Audiokonvertierung gibt es zahlreiche Anwendungen. Wie bei den Videokonvertierungen habe ich bei der Arbeit an diesem Kapitel für die Audiodateien VLC genutzt. VLC bekommen Sie unter *http://www.videolan.org/vlc/*.

Besonders erfolgreich ist die gleichzeitige Verwendung der Formate MP3 und Ogg. Zumindest eines dieser Formate wird von Firefox, Google Chrome, Safari, Opera und Internet Explorer 9 unterstützt. Wie beim Video müssen Sie Ihren Audio-Stream in ein Flash-Objekt einbetten, damit auch ältere Versionen von Internet Explorer damit klarkommen.

Hier folgt ein Beispiel mit dem `<audio>`-Tag für zwei Dateien, die mithilfe des `<source>`-Elements aufgerufen werden, das Sie bereits aus dem Videoabschnitt dieses Kapitels kennen.

```
<audio controls>
<source src="meinaudio.mp3"></source>
<source src="meinaudio.ogg"></source>
</audio>
```

Audiodateien in älteren Browsern abspielen

Wie bei Video brauchen ältere Browser das `<embed>`-Tag, um Audio abspielen zu können. Wird es für Audio eingesetzt, schreiben Sie normalerweise die beiden Attribute `src` und `autostart` mit hinein. `src` konfiguriert die Quelle der Audiodatei, und `autostart` steuert, ob der Audioclip beim Laden der Seite automatisch starten soll. Das HTML für unser Beispiel von eben mit dem `<embed>`-Tag sieht so aus:

```
<audio autoplay loop>
<source src="meinaudio.mp3">
<source src="meinaudio.ogg">
<embed src="meinaudio.mp3">
</audio>
```

Standardmäßig werden über `<embed>` integrierte Inhalte automatisch abgespielt. Wird das nicht gewünscht, schreiben Sie ins Attribut `autostart="false"`.

```
<embed src="meinaudio.mp3" autostart="false">
```

HINWEIS Auch beim Einsatz von `<embed>` für Audio muss der Besucher immer noch Software auf seinem Rechner haben, die die übermittelte Dateiart auch abspielen kann.

Ein weiteres, häufig mit `<embed>` genutztes Attribut ist `loop`. Dieses Attribut startet den Audioclip nach dem Ende erneut, wenn es auf `true` oder `infinite` gesetzt wird. Es kann auch auf `false` gesetzt werden, damit der Audioclip nur einmal abgespielt wird. Meist soll der Audioclip aber nur einmal abgespielt werden, und so ist es egal, ob Sie das Attribut weglassen oder auf `false` setzen.

Audioclips auf einer Webseite integrieren

Nun bauen wir einen Audioclip auf unserer Webseite ein. In dieser Übung fügen Sie eine Audiodatei in einer HTML5-Seite ein.

Vorbereitung Arbeiten Sie bei dieser Aufgabe mit den Dateien *index.htm*, *myaudio.mp3* und *myaudio.ogg* im Ordner der Übungsdateien. Diese Übungsdateien befinden sich im Ordner *Dokumente\Microsoft Press\HTML5 SBS\15Video\AddAudio*.

1. Öffnen Sie die Datei *index.html*.
2. Direkt über dem schließenden `</div>`-Tag des `#main`-Bereichs schreiben Sie die Codes für den Audioclip hinein.

```
<audio autoplay loop>
<source src="myaudio.mp3">
<source src="myaudio.ogg">
</audio>
</div>
</p>
```

3. Vor dem schließenden `</audio>`-Tag fügen Sie ein `<embed>`-Tag ein, das den Clip in einem Nicht-HTML5-konformen Browser abspielt.

```
<audio autoplay loop>
<source src="myaudio.mp3">
<source src="myaudio.ogg">
<embed src="myaudio.mp3">
</audio>
</div>
```

4. Öffnen Sie den Internet Explorer 9 oder höher und rufen Sie die Seite auf.

   Die Audiodatei sollte automatisch starten und dann nach Ende wieder an den Anfang gesetzt werden.

Aufräumen Schließen Sie die Fenster von Editor und Internet Explorer.

Zusammenfassung

- Internetseiten bekommen Audio und Video, indem Audio- und Videodateien in mehreren Formaten bereitgestellt werden, damit sich Ihre Besucher die Multimedia-Inhalte egal in welchem Browser anzeigen lassen können
- Es ist wichtig, die verschiedenen Container und Codecs zu kennen, die für Video und Audio möglich sind, und wie sie von den verschiedenen Browsern unterstützt werden
- HTML5 stellt die neuen `<video>`- und `<audio>`-Tags vor, mit denen Multimedia-Inhalte auf Webseiten integriert werden
- Von älteren Browsern werden sie nicht unterstützt, und deswegen müssen Videos auch in solchen Formaten wie Flash angeboten werden
- Mit dem `<embed>`-Tag binden Sie Audio- und Videoinhalte in einem Format ein, das auch Nicht-HTML5-konforme Browser verstehen

Faithe Wempen: HTML 5 - Schritt für Schritt, Microsoft Press 2011 (ISBN 978-3-86645-550-4)