LÖWENSTARK ONLINE MARKETING

HTML Verweise

Stand: 15.08.2022

Verweise mit HTML
Anker und Links

Für Seiten mit längerem Inhalt stellt HTML ein nützliches Feature bereit: Links zu dateiinternen **Ankern**. Viele Webseiten enthalten beispielsweise neben der Seitenüberschrift eine ganze Reihe von Unterüberschriften, weil es der umfangreiche
Inhalt erfordert. Dabei ist es für Leser angenehm, am Beginn der Seite gleich anklickbare Links auf die Unterüberschriften angeboten zu bekommen. Die Links helfen bei der Orientierung, was auf der aktuellen Seite zu finden ist, und ermöglichen außerdem
den gezielten Sprung zu einer bestimmten Unterüberschrift.

Zunächst das Beispiel eines Links zu einem Anker innerhalb des aktuellen HTML-Dokuments, also innerhalb der aktuell im Browser angezeigten Webseite:

```
<a href="#spieltag_12">Ergebnisse des 12. Spieltags</a>
```

Info Anker werden durch das Gatterzeichen (#) adressiert. Unmittelbar hinter dem Gatterzeichen folgt der Name des Ankers, zu dem bei Anklicken des Links gesprungen werden soll. Innerhalb des Dokuments muss natürlich ein entsprechender
Anker definiert sein. Zum Definieren von Ankern wird ebenfalls das a-Element benutzt, jedoch mit einem `name`-Attribut anstelle des `href`-Attributs. Die HTML-Umgebung des Verweisziels aus dem obigen Beispiel könnte
so aussehen:

```
<h3><a name="spieltag_12">Ergebnisse des 12. Spieltags</a></h3>
```

Info Der Anker wird also innerhalb einer Überschrift platziert. Das schließende könnte auch weiter vorne notiert werden, sogar unmittelbar hinter dem Start-Tag. Doch ein solcher „leerer“ Anker wird von diversen Browsern
aus unerfindlichen Gründen nicht erkannt, weshalb empfohlen wird, **Anker-Elemente** stets mit Inhalt zu versehen. Ankernamen können Sie frei vergeben. Folgendes ist jedoch zu beachten:

- Ankernamen müssen dokumentweit eindeutig sein.
- Das `name`-Attribut teilt mit dem `id`-Attribut den gleichen Namensraum. Falls Sie also Elemente mit `id`-Namen versehen, müssen alle `id`-Namen und Wertzuweisungen an das `name`-Attribut unterschiedlich
  sein.
- Groß-/Kleinschreibung wird unterschieden. Die Namen WS_2004 und ws_2004 sind unterschiedlich.

- Das name-Attribut darf auch HTML-Entities wie ü enthalten.
- Das erste Zeichen muss ein Buchstabe A-Z oder a-z sein.
- Nachfolgende Zeichen dürfen Buchstaben, Ziffern, der Unterstrich, der Bindestrich, der Doppelpunkt oder der Punkt sein. Im Hinblick auf andere Verwendungen, etwa mit JavaScript, ist jedoch dringend zu empfehlen, kein anderes Sonderzeichen als den Unterstrich zu verwenden.

Es ist übrigens erlaubt, beide Attribute name= und href= in einem a-Element zu notieren. Damit ist das Element zugleich ein Hyperlink und auch ein Anker, der anderen Hyperlinks als Verweisziel dienen kann. Es ist auch möglich, Anker in anderen Dokumenten anzuspringen. Dabei ist es unerheblich, wie das Dokument selbst adressiert wird, ob mit einer vollständigen URI-Angabe oder lokal relativ oder absolut. Am Ende wird einfach das Gatterzeichen, gefolgt vom Ankernamen, notiert.

```
<a href="http://www.w3.org/TR/html401/#minitoc">Inhalt der HTML 4.01 Spezifikation</a>
```

Links auf beliebige Inhaltstypen

Die meisten **HTML-Links** führen sicherlich zu anderen HTML-basierten Seiten. Es gibt aber keine Vorschrift dafür, dass ein Verweisziel von diesem Dateityp sein muss. Sie können ebenso gut Links auf PDF-Dokumente, Excel-Tabellen, ZIP-Archive, Grafik- oder Multimediadateien setzen.

Info Der Link ist also nicht das Problem. Das Problem besteht allein darin, was beim Anwender passiert, wenn er auf einen solchen Link klickt. Dazu wird der zugehörige Mime-Type bestimmt. Wird das Linkziel über einen Webserver angefordert (HTTP-Umgebung), entscheidet die Kommunikation zwischen Webserver und Browser darüber, welches der korrekte Mime-Type ist. Existiert keine HTTP-Umgebung, entscheidet der Browser allein, und zwar auf Grund ermittelbarer Daten wie der Dateiendung. Außerdem besteht noch die Möglichkeit, als HTML-Autor die Entscheidung zu beeinflussen, indem man im Link den gewünschten Mime-Type explizit angibt. Dazu kann im <a>-Tag das Attribut type= notiert werden. Die obigen Beispiele nochmals, doch diesmal mit expliziter Typangabe:

```
<a href="http://www.html-info.eu/html.zip"type="application/zip">Download HTML</a>
<a href="/bilder/sonnenuntergang-2.tif" type="image/tiff">Sonnenuntergang (1200 x 800 Pixel)</a>
```

In der Konfiguration eines Browsers ist eine Liste mit ihm bekannten Mime-Typen gespeichert.

Für die einzelnen Mime-Typen ist auch gespeichert, wie der Browser mit Inhalten dieses Typs verfahren soll. Außerdem ist eingestellt, was der Browser mit Daten tun soll, deren Mime-Type er nicht kennt. Da diese Einstellungen vom Endanwender ganz oder
teilweise bearbeitbar sind oder – wie beim MS Internet Explorer der Fall – eng mit den Dateiverknüpfungsinformationen des Betriebssystems verzahnt sind, ist nicht vorhersehbar, wie der Browser reagiert. Hat ein Anwender beispielsweise MS Office oder OpenOffice
installiert, könnte sein Browser, falls er ein Dokument mit dem Mime-Type *application/msexcel* (also eine Excel-Datei) erhält, direkt die betreffenden Office-Programme (MS Excel bzw. OpenOffice Calc) mit den heruntergeladenen Daten aufrufen, eleganterweise
sogar über Schnittstellen wie OLE direkt im Browserfenster. Ist bei einem Anwender jedoch kein Officepaket installiert, das mit Dateien dieses Typs etwas anfangen kann, wird der Browser dem Anwender vermutlich in einem Dialogfenster anbieten, das Dokument
herunterzuladen und als Datei unter einem frei wählbaren Namen abzuspeichern. Bei anderen Dateitypen wie etwa Archivdateien (ZIP, gzip, RAR, auch selbstentpackend als EXE) wünscht sich ein Linkanbieter vielleicht, dass der Browser bei Anklicken eines
entsprechenden Links einen Download-Dialog anbietet. Doch auch das ist nicht sicher. Es kann durchaus sein, dass sich bei einem Anwender z.B. gleich ein externes Programm wie WinZip öffnet. Eine Kontrolle hierüber haben Sie als „Linksetzer“ nicht, und
das ist auch gut so.

Features für E-Mail-Verweise

Links auf E-Mail-Adressen erfreuen sich großer Beliebtheit. Kein Wunder: Ein Klick und der Anwender kann z.B. dem Anbieter einer Website sofort eine Mail an eine von diesem bestimmte Mail-Adresse schreiben. In **HTML** sieht ein einfacher
Link dieser Art so aus:

```
<p>E-Mail an <a href="mailto:info@html-info.eu">Darko Pipic, info@html-info.eu</a></p>
```

Info Die Wertzuweisung an das `href`-Attribut besteht in diesem Fall aus dem Schema `mailto:`, gefolgt von der E-Mail-Adresse. Eine Garantie darauf, dass solche Links tatsächlich den gewünschten Effekt haben, gibt es jedoch
nicht. Der Browser muss wissen, welche lokale Software die Default-Software zum Managen von E-Mails ist. Außerdem muss ihm bekannt sein, wie er das Mail-Programm so aufrufen kann, dass sich darin sofort ein Fenster zum Schreiben einer neuen Mail öffnet.
Am einfachsten haben es hierbei natürlich so genannte Browser-Suiten wie die Mozilla-Suite oder die Opera-Suite, die über ein eigenes Mail-Modul verfügen. Bei separaten Mail-Programmen, besonders dann, wenn sie nicht Outlook oder Outlook Express heißen,
ist die Gefahr, dass E-Mail-Verweise nicht funktionieren, dagegen deutlich größer.

Angesichts der genannten Gründe ist es zweckmäßig, im Verweistext eines E-Mail-Links

stets die Mail-Adresse noch mal explizit anzugeben. So können auch Anwender, bei denen der Link nicht funktioniert, den angegebenen Mail-Kontakt nutzen. In der oben notierten Form öffnet sich also beim Anwender nach Anklicken des Links ein Fenster zum
Eingeben einer neuen E-Mail. Das Adressatenfeld ist vorbelegt mit der im **Link** angegebenen Mail-Adresse. Da ist es verständlich, dass schon früh der Wunsch aufkam, die Felder für das Mail-Subject, Kopienempfänger oder sogar den Mail-Inhalt
mit Text vorzubelegen. Eine entsprechende Syntax im erlaubten Rahmen der URIs hat sich dabei etabliert. Die meisten modernen Browser interpretieren die entsprechende Syntax. Dennoch ist keinesfalls garantiert, dass die Schnittstellenkommunikation mit
dem Mail-Programm so klappt, dass alle Felder wie gewünscht vorbelegt werden. Ein einfaches Beispiel mit Vorbelegung des `subject`-Felds:

```
<a href="mailto:info@html-info.eu?subject=Feedback%20zur%20Webseite">Mail an
Darko Pipic, info@html-info.eu</a>
```

Info Die optionalen Feldvorbelegungen werden innerhalb des `mailto`-URIs als Anfrageteil, also getrennt durch ein Fragezeichen, hinter der Mail-Adresse notiert. Wie in GET-Strings üblich, wird jedes gewünschte Feld wie ein
Parameter behandelt. Parameter folgen dem Schema `Name=Wert`. Mehrere Parameter-Wert-Kombinationen werden durch ein &-Zeichen voneinander getrennt: `Name=Wert&Name=Wert&Name=Wert`. Ein Beispiel für mehrere Angaben zur
Feldvorbelegung:

```
<a href="mailto:info@html-info.eu?cc=darko.pipic@html-
info.eu&subject=Feedback%20zur%20Webseite">Mail an Darko Pipic, info@html-
info.eu</a>
```

Info In diesem Beispiel werden das `subject`-Feld (Parameter subject) und das `cc`-Feld (Parameter cc) mit entsprechenden Inhalten vorbelegt. Folgende Parameter werden in der Regel erkannt:

- cc (sichtbarer Kopienempfänger)
- bcc (unsichtbarer Kopienempfänger)
- subject (Betreff)
- body (Nachrichtentext)

Beachten Sie, dass Sie **Sonderzeichen** maskieren müssen. Wenn Sie den Nachrichtentext vorbelegen möchten, können Sie darin Zeilenumbrüche durch die Sonderzeichenmaskierung `%0A` erzeugen, z.B.:

```
<a href="mailto:info@html-info.eu?cc=darko.pipic@html-
info.eu&subject=Feedback%20zur%20Webseite&body=Lieber%20Autor,
%0A%0AIch%20habe%Ihre%20Webseite%20besucht">Mail an Darko Pipic, info@html-
info.eu</a>
```

Allgemeines zu projektexternen Verweisen

Aus technischer Sicht stellen projektexterne **Verweise** kein großes Problem dar. Diese Möglichkeiten sind beim `href`-Attribut des <a>-Tags erlaubt. Dennoch sollten Sie einiges mehr über projektexterne Verweise wissen:

> Im Normalfall dürfen Sie ungefragt Verweise auf fremde Web-Angebote setzen.

Sie brauchen also keine E-Mail an den Anbieter mit Bitte um Genehmigung zu schreiben, wenn Sie auf sein Angebot einen Link setzen wollen. Jeder Anbieter, der mit seinem Webprojekt online geht, muss sich im Klaren darüber sein, dass er Teil eines weltweiten
Hypertext-Systems ist, in dem er nicht allein ist. Wenn er das nicht akzeptieren kann, ist er im Web fehl am Platz und hat das falsche Medium gewählt. Die Grundregeln des Web werden von den Ideen des Web bestimmt, nicht von den hermetischen Zwangsvorstellungen
einiger Zuspätgekommener. Es gibt jedoch Ausnahmen von der Regel. Wenn Sie beispielsweise selbst ein sehr stark frequentiertes Web-Angebot haben und auf dessen Einstiegsseite einen Verweis auf die Homepage eines kleinen, unbekannten Anbieters setzen,
dann sollten Sie ihn vorher fragen. Der Grund: Durch die vielen zu erwartenden Besucher, die über Ihren Verweis auf das fremde Angebot finden, wird dort plötzlich sehr viel Traffic (Besucherverkehr und Datenübertragung) erzeugt. Viele Anbieter haben bei
ihrem Provider eine Volumenbegrenzung, und wenn diese überschritten wird, entstehen den Anbietern unkalkulierbare Kosten. Auch könnte es sein, dass der fremde Server nicht sehr belastungsfähig ist und durch die vielen plötzlichen Besucher zusammenbricht.
Eine andere Ausnahme ist, wenn Sie den Verweis in einem negativen Kontext setzen. Wenn Sie also auf einen Anbieter verweisen, nur weil sie ihn auf Ihren eigenen Seiten heftig kritisieren, dann ist es sicherer, sich dort eine Bestätigung einzuholen, dass
ein solcher Verweis gebilligt wird. Juristisch sicher ist eine solche Bestätigung aber nur, wenn sie mit Unterschrift in dokumentenechter Form von einer dafür zuständigen Person geleistet wird.

> Eine Sache, gegen die man sich im Sinne der Zivllcourage wehren sollte,

ist der Versuch von Anbietern, juristisch gegen so genannte **Deeplinks** (Verweise in die Tiefe eines fremden Angebots) vorzugehen. Weltweiter Hypertext im Web bedeutet, dass auf vorhandene URIs verwiesen werden darf. Schließlich kann sich
auch jeder Anwender auf jeden URI ein Lesezeichen (Bookmark, Favoriten) in seinem Browser setzen. Das kann auch eine Datei tief in einem fremden Web-Angebot sein. Es gibt von Anbieterseite aus technische Möglichkeiten, Deeplinks auf eine allgemeinere
Seite umzulenken. Wenn er also nicht will, dass Deeplinks auf sein Angebot gesetzt werden, dann soll er von diesen technischen Möglichkeiten Gebrauch machen.

Beispiele für projektexterne Verweise

Die einzige Bedingung, die beim Anwender erfullt sein muss, damit er projektexterne Verweise ausfuhren kann, ist eine bestehende Internet-Verbindung.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<title>Text des Titels</title>
</head>
<body>
<h1>Besuchen Sie doch mal...</h1>
<p><a href="http://www.html-info.eu/"> ein paar Kurse &uuml;ber die hier
vermittelten inhalte</a>
<br>
<a href="http://www.html-info.eu/text-html/verweise.html/"> <i>HTML-Info</i>
&uuml;ber externe Verweise</a>
<br>
<a
href="http://www.html-info.eu/text-html/verweise/item/interne-verweise.html/">
interne Verweise</a>
<br>
<a
href="http://www.html-info.eu/text-html/verweise/item/definieren-und-gestalten.h
tml/"> Verweise Definieren und Gestalten</a>
<br>
<a href="http://www.html-info.eu/text-html/verweise.html/"> Info &uuml;ber
Verweise</a>
<br>
<a href="http://www.html-info.eu/"> Info &uuml;ber HTML</a></p>
</body>
</html>
```

Info Webadressen beginnen mit *http://*, manchmal auch mit *https://* (Letzteres sind Server, bei denen die Datenübertragung von und zum Browser verschlüsselt stattfindet, z.B. bei Internet-Banking).

Andere Internet-Protokolle können Sie ebenfalls adressieren,

beispielsweise FTP-Adressen mit *ftp://*, Adressen auf Gopher-Servern mit *gopher://* oder Telnet-Adressen mit *telnet://*. Auch Newsgroups im Usenet können Sie adressieren, nämlich mit *news:* (ohne die beiden sonst charakteristischen

Schrägstriche). Bei anderen als *http*-Adressen kommt es darauf an, wie der Webbrowser damit umgeht. Die modernen Browser beherrschen meistens FTP und Gopher und stellen entsprechende Adressen in ihrem Anzeigefenster dar. Bei Protokollen, die der
Browser nicht unterstützt, versucht er, auf dem Rechner des Anwenders ein Programm auszuführen, das fur das entsprechende Internet-Protokoll zuständig ist. Bei Telnet wird beispielsweise ein auf dem Rechner installierter Telnet-Client aufgerufen, und
bei Verweisen auf Newsgroups ein Newsreader oder das Newsreader-Modul eines Mailprogramms. Bei Newsgroups muss jedoch ein News-Server im Newsreader des Anwenders eingerichtet sein, der die adressierte Newsgroup anbietet. Auch E-Mail-Verweise sind möglich.
Viele Adressen bestehen nur aus dem Namen einer „WWW-Domain“, etwa http://www.html-info.eu/. Trotzdem führt der **Verweis** auf eine konkrete HTML-Datei. Das liegt daran, dass es bei vielen Webservern einen so genannten Default-Dateinamen
gibt – meistens index.htm, index.html oder welcome.htm bzw. welcome.html Das Projekt muss natürlich auch eine entsprechende Datei besitzen. Im Verweis braucht die HTML-Datei aber nicht mit angegeben zu werden.

Viele solcher Adressen werden immer wieder ohne abschließenden Schrägstrich angegeben,

etwa http://www.html-info.eu. Es ist jedoch sauberer, wenn Sie noch den Schrägstrich dahinter setzen. Nur so kann der Webbrowser bereits am Verweis erkennen, dass es sich um ein Verzeichnis handelt, in dem eine Default-Datei steht, deren Namen
der Webserver zur Verfiigungstellt. Es ist deshalb besser zu notieren: http://www.html-info.eu/. Noch wichtiger ist es, bei Unterverzeichnissen einen abschließenden Schrägstrich zu notieren. Zwar klappt es auch, wenn Sie eine Adresse wie http://www.html-info.eu/text-html/verweisenotieren. Doch dann findet unnötig viel Kommunikation zwischen Browser und Server statt, denn intern fordert der Browser vom Server im Beispiel erst mal eine Datei namens training, was den Server zunächst zu einer Fehlermeldung veranlasst,
da diese Datei nicht existiert. Erst im zweiten Schritt wird erkannt, dass es sich um den Namen eines Verzeichnisses handelt. Notieren Sie deshalb immer Angaben wie
http://www.html-info.eu/text-html/verweise/, also mit abschließendem Schrägstrich.
Es gibt neben den angesprochenen Internet-Protokollen auch noch andere, etwas wais oder irc. Ferner gibt es das „Un-Protokoll“ file, über das sich lokale Rechner und Netzwerkadressen absolut adressieren lassen, mit Angaben
wie `<a href="file://localhost/"a>...</a>`. Dies wird von moderneren Browsern jedoch aus Sicherheitsgründen nicht mehr oder nur noch eingeschränkt unterstützt und ist auch nicht besonders sinnvoll, da es heute wirklich kein Problem
mehr ist, für lokale Zwecke einen Webserver lokal einzurichten.

Hyperlink-Optimierung

HTML stellt verschiedene Attribute zur Verfügung, deren Aufgabe darin besteht, die Funktionalität von **Links** zu verbessern. Zunächst einmal können Sie in einem a-Element die beiden Attribute `rel=` (logischer Bezug)

und `rev=` (logischer Rückbezug) notieren. Dabei sind die gleichen Wertzuweisungen möglich wie in Zusammenhang mit dem `link`-Element.

```
<a href="../index.htm" rel="chapter">Kapitel 7</a>
```

Mögliche Angaben für das `rel`– und das `rev`-Attribut sind z.B. `contents` (Inhaltsverzeichnis), `chapter` (Kapitel), `section` (Abschnitt), `subsection` (Unterabschnitt), `index`(Stichwortverzeichnis), `next` (Seite vor), `prev` (Seite zurück) und `start` (Startseite). Welche Auswirkungen eine solche Notation im Browser hat, bleibt der Browser-Software überlassen. Firefox beispielsweise behandelt `rel`– und `rev`-Angaben in Hyperlinks genauso wie notierte `link`-Elemente, d.h., er übernimmt die Bezugsangaben in die als Plug-In erhältliche Navigationsleiste für logische Linkbezüge.

Eine andere Hilfe – sowohl für den Browser als auch für den Anwender

sind nähere Angaben zum Verweisziel. Dazu werden die Attribute `hreflang=` und `charset=` angeboten. Bei `hreflang=` können Sie ein Sprachenkürzel gemäß ISO 3166-1 angeben, also z.B. de für deutschsprachig, en für englischsprachig
oder ar für arabischsprachig. Bei `charset` können Sie dem Browser bereits mitteilen, welchen Zeichensatz das Verweisziel benutzt, z.B. ISO-8859-1 oder einen anderen Zeichensatz gemäß der Liste der IANA-Organisation.

```
Der arabische Sender <a href="http://www.aljazeera.net/" hreflang="ar"
charset="windows-1256">Al Dschasira</a>
```

Die Angabe zum `charset`-Attribut deckt sich mit der entsprechenden Angabe aus dem **HTML-Quelltext** der verlinkten Seite:

```
<meta http-equiv="Content-Type" content="text/html; charset=windows-1256">
```

Beim `hreflang`-Attribut wird das Sprachenkürzel für „arabisch“ angegeben. Welchen Nutzen ein Browser aus diesen Informationen zieht, bleibt ebenfalls ihm überlassen. Bei den heutigen Browsern haben diese Angaben jedenfalls keine sichtbare
Wirkung, obwohl solche Dinge wie das Anzeigen eines Flaggensymbols beim Überfahren des Links mit der Maus denkbar wäre.

Eindeutig auf den Anwender zielen dagegen die möglichen Attribute

`tabindex=` und `accesskey=` ab. Beide Attribute sollten, wenn verwendet, bei allen oder zumindest allen wichtigen Links einer Seite notiert werden, da ihre Werte einen logischen Zusammenhang bilden. Drückt ein Anwender beim Anzeigen
der Seite wiederholt die Tabulatortaste, kann er auf diese Weise nacheinander auf die in der Seite enthaltenen Links positionieren. Mit der (`Enter`)-Taste kann er den Verweis dann „anklicken“. Das Feature ist für Anwender gedacht, die keine
Maus benutzen können oder wollen. Normalerweise wird zunächst auf den ersten Link der Seite positioniert, dann auf den zweiten usw. Durch Angabe des `tabindex`-Attributs in den Links einer Seite können Sie die Reihenfolge selbst bestimmen.
Bei `tabindex=` können Sie als Wert Zahlen zwischen 0 und 32767 angeben. Auf den Link mit der niedrigsten `tabindex`-Nummer wird zuerst positioniert, dann auf den mit der zweitniedrigsten usw. Das `accesskey`-Attribut
erlaubt dagegen das direkte „Anklicken“, also Ausführen eines Links über einen **Hotkey**. Wenn Sie beispielsweise `accesskey="c"` notieren, dann bieten die meisten Browser eine entsprechende Hotkey-Funktionalität an, um den Link
auszuführen. Als Wert sollten also Buchstaben oder auch Ziffern zugewiesen werden. Die Umsetzung im Browser kann jedoch unterschiedlich aussehen, wie die folgende Tabelle belegt:

Plattform / Browser	Umsetzung des accesskey-Attributs
Windows / Internet Explorer	(Alt) + (accesskey) + (Enter)
Windows / Mozilla	(Alt) + (accesskey)
Windows / Opera	(a) + (Esc) + (accesskey)
Macintosh / Internet Explorer	(Ctrl)+ (accesskey)+ (Enter)
Macintosh / Safari	(Ctrl) + (accesskey)
Macintosh / Mozilla	(Ctrl)+ (accesskey)
Macintosh / Opera	(a) + (Esc) + (accesskey)

Leider kennen die meisten Anwender die entsprechenden Tastaturkombinationen bei ihren Browsern gar nicht. Jene, die sie kennen und nutzen, werden dem Entwickler der Webseiten jedoch die Verwendbarkeit bei wichtigen Links danken.

Hyperlinks definieren und gestalten

Bislang haben wir in diesem Kapitel typische Elemente zur Textstrukturierung und die Möglichkeiten zu deren Gestaltung kennen gelernt. Das „HT“ in HTML steht aber für **Hypertext**. Ein Kern-Feature von HTML ist nämlich die konsequente Umsetzung
des von HTML- und WWW-Vordenker Tim Berners Lee entwickelten Schemas zur Adressierung beliebiger Inhalte im Internet und in lokalen Host-Umgebungen.

> Es ist nicht schwer, in HTML „einfach mal“ ein paar Links zu notieren.

Um Hyperlinks in allen Fällen korrekt einzusetzen, sind jedoch Kenntnisse über den Aufbau von URIs erforderlich. In diesem Abschnitt lernen Sie, wie Sie Quellen richtig adressieren, aber auch, welche zusätzliche Möglichkeiten Sie dem Anwender in Zusammenhang
mit Hyperlinks anbieten können und wie Sie Links optisch ansprechend gestalten.

URIs und Links in HTML

Möglicherweise ist Ihnen das Akronym URL (Uniform Resource Locator – zu Deutsch: einheitlicher Quellenorter) geläufiger als URI (Universal Resource Identifier – zu Deutsch: universelle Quellenbezeichnung). Noch unbekannter dürfte
URN (Uniform Resource Name – zu Deutsch: einheitlicher Quellenname) sein. Um zu verstehen, was ein URI ist, muss man jedoch alle drei Akronyme im Zusammenhang sehen. Ein URN sieht aus wie eine typische „Internetadresse“, also z.B. http://www.example.org/ von-mir-persoenlich-erfunden/.
Es handelt sich jedoch nicht um eine existierende Adresse, sondern um einen „universell eindeutigen Namen“, der das Schema der Internetadressierung benutzt, weil sich damit leicht universell eindeutige Namen erstellen lassen.

> Ein URL ist eine existierende Internetadresse

wie http://www.mut.de/. Insofern ist es verständlich, dass vielfach von URLs oder URL-Adressen die Rede ist. URI ist der Überbegriff für URL und URN. Im Falle existierender Internetadressen bedeutet URI und URL also durchaus das Gleiche. Wegen
der URNs jedoch, die vor allem in der XML-Welt eine gewisse Bedeutung haben, spricht die **HTML-Spezifikation** vorzugsweise von URIs. Für den Aufbau von URIs gilt folgendes Format:

```
<Schema>://[<Benutzer>[:<Passwort>]@]<Server>[:<Port>]/<Pfad,...> ?<Anfrage>
```

HTML-Notation von Hyperlinks

Wenn Sie in HTML Ressourcen auf einem anderen Server im Internet adressieren wollen, müssen Sie auf

jeden Fall einen hinreichend vollständigen URI angeben. Entsprechende Links in HTML könnten etwa so aussehen wie im folgenden Quelltextausschnitt:

```
<h1>Links für HTML-Kenner</h1>
<ul>
<li><a href="http://www.w3.org/TR/xhtml11/">XHTML 1.1 Recommendation</a></li>
<li><a href="http://www.w3.org/TR/xhtml1/">XHTML 1.0 Recommendation</a></li>
<li><a href="http://www.w3.org/TR/html401/">HTML 4.01 Recommendation</a></li>
</ul>
```

HTML-Links haben immer den gleichen Aufbau:

```
<a href="Zieladresse">Anklickbarer Verweisinhalt</a>
```

Das a-Element ist für Links zuständig. Es kann in Block- und anderen Inline-Elementen vorkommen und selbst neben Text auch andere Inline-Elemente (mit Ausnahme anderer a-Elemente) enthalten. Erlaubt sind also z.B. **HTML-Konstrukte**wie diese:

```
<h2>Spezial-Domains wie <a href="http://www.example.org/"> example.org</a></h2>
Wikipedia hat auf alles eine <a href="
http://de.wikipedia.org/wiki/Antwort"><em>Antwort</em></a>
<a href="http://de.news.yahoo.com/"><img
src="http://eur.i1.yimg.com/eur.yimg.com/i/de/ne/nws6b.gif" alt="Yahoo
Nachrichten"></a>
```

Info Im ersten Beispiel steckt der Link in einer h2-Überschrift. Entsprechend dem Vererbungsprinzip übernimmt der Link bei der Darstellung z.B. die Schriftgröße der Überschrift. Im zweiten Beispiel enthält der Linktext selbst ein anderes
Inline-Element (em). Das Element wird wie üblich dargestellt, übernimmt jedoch die Darstellungsform für Hyperlinks.

Zu den Inline-Elementen gehört auch das

```
img-Element, das zum Referenzieren von Grafiken in HTML dient. Als Elementinhalt
eines Hyperlinks notiert, wird die Grafik selbst zum anklickbaren Link. Den
Rahmen, den der Browser um eine auf diese Weise anklickbare Grafik zieht,
bekommen
Sie weg, indem Sie im <img>-Tag den Rahmen unterdrücken - am besten über die
CSS-Syntax style="border:none". Das Konstrukt <a...><img...></a> ist in der
```

```
Praxis übrigens
extrem verbreitet - so basieren z.B. anklickbare Werbebanner darauf oder Logos,
die bei Anklicken z.B. zur Startseite der Webpräsenz führen.
```

Links auf dynamische Inhalte

Hinter zahlreichen Webseiten steckt kein statischer Inhalt, wie z.B. eine fest abgespeicherte HTML-Datei, sondern ein Script oder Programm (z.B. ein PHP-Script), das variable HTML-Inhalte erzeugt und an den aufrufenden Browser zurücksendet. Dazu erwartet
ein solches Script meist entsprechenden Input. In vielen Fällen steht dieser Input direkt im URI, und zwar in den so genannten GET-Parametern, eingeleitet durch ein Fragezeichen hinter der Pfadangabe des Scripts. In einem **HTML-Link**können Sie als Verweisziel auch solche URIs angeben.

```
<a href="
http://www.google.de/search?hl=de&q=Meteoriten&btnG=Suche&meta=">Google-Suche
nach Meteoriten</a>
```

Info Am einfachsten bringen Sie einen solchen URI in den eigenen HTML-Quelltext, indem Sie im Browser die Zielsite aufrufen (im Beispiel also Google) und dort den gewünschten Zustand herstellen (im Beispiel also nach Meteoriten suchen). Wenn die
Seite mit Suchtreffern angezeigt wird, kopieren Sie den Inhalt der Adresszeile des Browsers in die Zwischenablage. Den kopierten Inhalt fügen Sie dann im Editor als Zuweisung an das `href`-Attribut ein.

> Testen Sie Links auf solche URIs jedoch aus und überprüfen Sie,

ob Sie tatsächlich die gewünschten Inhalte erhalten. Denn in vielen dynamischen URIs werden auch Parameter wie Session-IDs mitgeschleppt, die sich nicht für einen Link eignen. Probieren Sie gegebenenfalls auch mal, solche Parameter-Wert-Kombinationen
wegzulassen.

Linktypen und Darstellung von Links

Es mag erstaunen, dass alle Links in HTML über das gleiche HTML-Element nach dem gleichen Schema notiert werden. Denn auf modernen Webseiten gibt es offensichtlich ganz unterschiedliche Typen von Links, z.B. anklickbare Werbebanner oder Schaltflächen
in einer Site-internen Navigationsleiste und Hyperlinks im normalen Fließtext. Der Vorteil dieser Lösung ist auf jeden Fall, dass Sie nur ein Verlinkungsschema kennen müssen, das auf alle Fälle passt. Der Nachteil ist, dass Sie die unterschiedliche Bedeutung
der Links nur durch optisch unterschiedliche Gestaltung oder Platzierung auf der Webseite kenntlich machen können. Wir werden in diesem Abschnitt zunächst noch näher auf die möglichen Fälle und Besonderheiten bei der Adressierung von Verweiszielen eingehen.

Hinweise zu Links auf fremde Quellen

Durch Hyperlinks in Verbindung mit dem Internet und dem Adressierungsschema der URIs ist im Prinzip alles mit allem anbieterübergreifend vernetzbar. Deshalb sind **Hyperlinks** in HTML gewissermaßen der Schlüssel zur Wunschvorstellung der
Informationsgesellschaft, die immer mehr zum Paradigma unseres Zeitalters wird. Doch leider ist die Realität nicht so einfach. Ein Link ist schnell gesetzt, kann jedoch ebenso schnell zivil- oder gar strafrechtliche Folgen haben. Außerdem gibt es gewisse
Anstandsregeln. Die folgenden Hinweise sollten Sie auf jeden Fall beachten:

Kostenpflichtige Links:

Manche Anbieter erlauben Links nur gegen Entgelt, z. B. einige Anbieter von Karten- oder Straßenplanmaterial. Wenn Sie Links auf solche URIs ohne Kenntnis des Anbieters setzen (z.B. weil Sie auf diese Weise Ihren Besuchern eine Anfahrtsskizze präsentieren
wollen), ist das genauso wie Schwarzfahren mit der U-Bahn oder Parken auf einem kostenpflichtigen Parkplatz ohne Parkschein. Das Vergehen kann vom Anbieter durch Einschalten eines Anwalts oder gerichtlich verfolgt werden.

Links auf illegale Inhalte:

Im Internet tummeln sich etliche Angebote mit Quellen, die zwar als URI adressierbar sind, deren Bereitstellung jedoch eine Straftat darstellt. Das betrifft z.B. illegale pornografische Inhalte wie Kinderpornos oder Downloadangebote von Vollversionen
registrierungspflichtiger Software. Wenn Sie Links zu solchen Quellen setzen, können Sie im Rahmen der Mitstörerhaftung bei Entdecken juristisch genauso belangt werden wie der Anbieter selbst. Dabei können je nach Schwere des Falls mehrjährige Haftstrafen
drohen!

Deeplinks:

Einige Anbieter wünschen keine direkten Links auf Unterseiten ihres Angebots, sondern nur auf die Startseite. Manchmal steckt technisches Unvermögen dahinter (z.B. weil der Anbieter Frames einsetzt und direkt adressierte Unterseiten plötzlich ohne Navigation,
Werbebanner und Logo angezeigt werden). In anderen Fällen ist es dem Anbieter einfach unangenehm, wenn ein Besucher zuerst sein "Kellerregal" zu sehen bekommt und nicht das protzige Eingangsportal. Es gibt zwar kein Gesetz, das Deeplinks verbietet, doch
ein Anbieter, der explizit keine solchen Links wünscht, könnte auf jeden Fall einen Anwalt einschalten und ein Gerichtsverfahren anstrengen. Bei Deeplinks auf fremde Webangebote empfiehlt es sich deshalb, dort nach publizierten Geschäftsbedingungen (AGB),
nach einem Impressum oder nach einer anderen Seite mit rechtlichen Hinweisen zu suchen, um sich über eventuell unerwünschte Deeplinks zu informieren. Der Rest ist eigenes Risiko. Es gibt sogar gute Gründe, sich über ein Deeplink-Verbot hinwegzusetzen,
nämlich die Zivilcourage, die nötig ist, damit die Linkfreiheit im Netz nicht den Zwangsvorstellungen von einzelnen Anbietern geopfert wird.

Links zu kleineren Sites:

Wenn Sie selbst eine von Besuchern sehr stark frequentierte Seite anbieten und darauf einen Link auf ein fremdes Angebot setzen, ist damit zu rechnen, dass durch den Link viele Besucher zu dem fremden Angebot surfen. Die meisten Anbieter müssen bei ihren
Hosting-Providern jedoch für Traffic ab einer bestimmten Grenze mehr bezahlen - das Prinzip ist ähnlich wie beim Stromverbrauch. Durch Ihren **Link** können Sie dem fremden Anbieter also zusätzliche und in der Höhe unkontrollierbare Kosten
verursachen, indem Sie gewissermaßen seine Lichter auf seine eigenen Kosten dauernd ein- und ausschalten. Wer ein Webangebot unterhält, muss zwar stets damit rechnen, dass Besucherzahlen und Traffic auch mal stark anschwellen können. Doch es gehört zum
guten Ton, sich in entsprechenden Fällen vor dem Setzen eines Links mit dem Fremdanbieter zu verständigen und dessen Einwilligung einzuholen.

> Wenn Sie diese Punkte beachten, brauchen Sie jedoch auch nicht übervorsichtig zu sein.

Sie brauchen beispielsweise keine Mail mit Lesebestätigung und Wichtig-Flag an Google zu schicken, wenn Sie beabsichtigen, auf ein Suchergebnis von deren Suchmaschine zu verlinken.

Link zum Thema Links auf fremde Quellen:

"Linkhaftung: Gesetzgeberische Untätigkeit schafft endlich Klarheit": Telepolis-Artikel

Links auf lokale Quellen

Unter einer "lokalen Quelle" verstehen wir **Links**, deren Ziel ohne Angabe eines Protokolls und Servers adressierbar ist. Aus dem Format eines URIs werden in diesem Fall nur der Pfad und gegebenenfalls der Anfrage-Teil benötigt.

Relative Adressierung

Der einfachste Fall ist der, dass sich das Verweisziel im gleichen Verzeichnis befindet wie das HTML-Dokument mit dem Verweis. Angenommen, der Verweis befindet sich in einer Datei namens index.html und führt zu einer Datei namens impressum.html im gleichen Verzeichnis. Im HTML-Link genügt dann folgende Adressierung:

```
<a href="impressum.html">Impressum</a>
```

Info Wenn sich das Verweisziel in einem Unter- oder einem Unterunterverzeichnis relativ zum verlinkenden Dokument befindet, sind Angaben dieser Form möglich:

```
<a href="team/julia.html">Julia Anders</a>
```

```
<a href="projekte/ws2004/uebersicht.html">Projektübersicht Wintersemester
2004/2005</a>
```

Info Das Trennzeichen für Verzeichnisse in Pfadangaben ist dabei stets ein einfacher Schrägstrich (/). Das gilt auch dann, falls Sie unter MS Windows arbeiten, wo das entsprechende Trennzeichen normalerweise der Backslash (\)
ist. Falls sich das **Verweisziel** eine Verzeichnisebene oberhalb befindet oder in einem Verzeichnis unterhalb einer übergeordneten Verzeichnisebene, sind Adressierungen der folgenden Art möglich:

```
<a href="../impressum.html">Impressum</a>
<a href="../../projekte/ws2004/uebersicht.html">Projektübersicht Wintersemester
2004/2005</a>
```

Info Eine Verzeichnisebene oberhalb wird durch die Zeichenfolge `../` notiert. Eine Angabe wie im zweiten Beispiel bedeutet: zwei Verzeichnisebenen höher und dort im Unterverzeichnis projekte das Unterverzeichnis ws2004 und dort die Datei uebersicht.html. Das aktuelle Verzeichnis kann übrigens durch die Zeichenfolge `./` explizit angesprochen werden. Eine `href`-Wertzuweisung wie `./impressum.html` ist identisch mit impressum.html.

Absolute Adressierung

Alle zuvor angeführten Beispiele verwenden eine relative Adressierung ihres Verweisziels. Ebenso möglich ist jedoch eine absolute Adressierung wie in den nachfolgenden Beispielen:

```
<a href="/team/julia.html">Julia Anders</a>
<a href="/impressum.html">Impressum</a>
```

Info Eine absolute Adressierung beginnt mit einem Schrägstrich. Dahinter folgt der vollständige Pfad zur gewünschten Quelle. Unklar bleibt dabei allerdings der Bezugspunkt. Auf einem Windows-PC bedeutet eine solche **Adressierung**:
ausgehend vom Wurzelverzeichnis des aktuellen Laufwerks. Auf einem Unix-basierten Betriebssystem bedeutet die Adressierung: ausgehend vom Wurzelverzeichnis des aktuellen Dateisystems (darunter können auch externe Gerätelaufwerke, Netzwerkpfade usw.
fallen, sofern sie entsprechend gemountet wurden). Befindet sich das HTML-Dokument mit dem Link dagegen in einer HTTP-Umgebung, erkennbar daran, dass die Adresszeile im Browser, der das Dokument anzeigt, mit *http://...* beginnt, so bedeutet
die absolute Adressierung noch etwas anderes. Der Bezugspunkt ist in diesem Fall das im Webserver konfigurierbare document-root-Verzeichnis (Wurzelverzeichnis für Webseiten), und zwar das document-root-Verzeichnis des "implizierten"
Hosts. Das kann z.B. auch ein virtueller Host sein, eine Subdomain usw.

Bei absoluter Adressierung sollten Sie deshalb wissen,

was Sie tun. Falls Sie z.B. in Ihrer lokalen Entwicklungsumgebung einen Webserver fahren und gegebenenfalls die Virtual-Host-Konfiguration mit derjenigen der Produktivumgebung auf dem öffentlichen Webserver abgleichen, werden absolute Pfadangaben bei
lokalen HTTP-Aufrufen ebenso aufgelöst wie auf dem öffentlichen Webserver. Falls Sie ohne lokale HTTP-Umgebung Seiten erstellen, die auf einen öffentlichen Webserver hochgeladen werden sollen, ist es besser, mit relativer Adressierung zu arbeiten.
Und wenn Sie Seiten etwa für eine HTML-basierte CD/ DVD-ROM-Produktion erstellen, ist relative Adressierung oberste Pflicht bei der Link-Adressierung.

Sonderzeichen und URL-Angaben

URIs dürfen eigentlich nur Zeichen aus dem **ASCII-Zeichensatz** enthalten. Wenn Sie beispielsweise eine Datei münchen.html genannt haben und einen Link darauf setzen, dürfen Sie den Dateinamen nicht so verlinken, wie er lautet.

Falsch ist also:

```
<a href="http://meinserver/münchen.html">München</a>
```

Richtig ist dagegen:

```
<a href="http://meinserver/m&uuml;nchen.html">München</a>
```

Bei allen Zeichen, die einen Unicode-Zeichenwert höher als 127 haben, sollten Sie in URI-Angaben so verfahren. Betroffene Zeichen werden durch ein Prozentzeichen eingeleitet, gefolgt vom Zeichenwert in Hexadezimalschreibweise. Doch nicht alle Zeichen
des ASCII-Zeichensatzes dürfen in URIs innerhalb von Server-, Verzeichnis- oder Dateinamen oder im Anfrageteil (GET-String) eingegeben werden. URI-eigene Sonderzeichen und Leerraumzeichen müssen ebenfalls hexadezimal maskiert werden.
Die folgende Tabelle listet die betroffenen Zeichen und ihre korrekte Notation auf:

Zeichen	Notation	Zeichen	Notation
?	%3F	Leerzeichen	%20
=	%3D	Wagenrücklauf	%0D

Zeichen	Notation	Zeichen	Notation
%	%25 oder %%	Zeilenvorschub	%0A
:	%3B	Tabulator	%09
/	%2F	&	%26

Optische Gestaltung von Hyperlinks mit CSS

In den meisten Browsern kann der Anwender die Farbe für Links zu bereits besuchten Seiten und zu noch nicht besuchten Seiten einstellen, und oft auch, ob Links unterstrichen dargestellt werden sollen oder ob sie beim Überfahren oder Anklicken mit
der Maus durch optische Änderung reagieren sollen. Per Voreinstellung werden Links zu noch nicht besuchten Seiten in grafischen Browsern meist blau und unterstrichen angezeigt und Links zu bereits besuchten Seiten lila oder dunkelblau und unterstrichen.

Da es sich dabei um "Zustände" handelt und nicht um HTML-Elemente, ist der CSS-Zugriff darauf durch so genannte Pseudoelemente möglich. Speziell für die Gestaltung von **Hyperlinks** in ihren verschiedenen Zuständen stehen die Pseudoformate
`:link`, `:visited`, `:hover` und `:active` zur Verfügung. Ihren Einsatz zeigt das folgende Beispiel. Im zentralen `style`-Bereich im Dateikopf oder in einer separaten, referenzierten CSS-Datei
stehen folgende Definitionen:

```
a:link {
text-decoration: none;
font-weight: bold;
color: #E00000;
}
a:visited {
text-decoration: none;
font-weight: bold;
color: #E08080;
}
a:hover {
text-decoration: none;
font-weight: bold;
background-color: #FFFFA0;
}
a:active {
text-decoration: none;
font-weight: bold;
background-color: #A0FFFF;
```

```
}
```

Der Selektor `a:link` bedeutet "alle a-Elemente (also Hyperlinks) zu noch nicht besuchten Zielen", `a:visited` "alle Links zu bereits besuchten Zielen", `a:hover` "Links, auf die der Anwender positioniert" und `a:active` "Links, die der Anwender anklickt". Bei der Gestaltung, also der Wahl der **CSS-Formateigenschaften**, können Sie Ihrer Phantasie freien Lauf lassen. Im obigen Beispiel haben wir festgelegt, dass die Links nicht
unterstrichen dargestellt werden (`text-decoration:none`), dass sie stattdessen fett dargestellt werden (`font-weight:bold`), dass Links zu noch nicht besuchten Zielen in kräftigem Rot erscheinen (`color:#E00000`),
während Links zu besuchten Zielen blassrot angezeigt werden (`color:#E08080`). Links, auf die der Anwender gerade positioniert, egal ob mit der Maus oder mit der Tastatur, erhalten eine hellgelbe Hintergrundfarbe (`background-color:#FFFFA0`)
und beim Anklicken eine hellblaue (`background-color:#A0FFFF`). Vor allem das Nichtunterstreichen von Links (`text-decoration:none`) ist ein immer wieder nachgefragtes Feature. Mit etwas aufwändigeren Angaben können Sie auch
komplette Navigationsleisten inklusive Mouseover-Effekt allein mit HTML und CSS erstellen, d.h. ohne Rückgriff auf Grafiken und JavaScript, wie es früher üblich war.