ITSysAdminFw
WebSK

FüUstgSBw

Lehrgruppe C / IX. Inspektion

DRAFT

HF Weidinger
L Siegerth

Inhaltsverzeichnis

1 Einführung 4

2 HTML 8

2.1 Grundlagen HTML . . . 9
2.2 Inhalte in HTML . . . 13
2.2.1 Paragraphen . . . 14
2.2.2 Textformatierung . . . 16
2.2.3 Überschriften . . . 17
2.2.4 Listen . . . 18
2.2.5 Bilder . . . 20
2.2.6 Links . . . 21
2.2.7 Tabellen . . . 24
2.2.8 Formulare . . . 30
2.2.9 Externe Inhalte . . . 33
2.3 Struktur in HTML . . . 35
2.3.1 Blockelemente vs. Inline Elemente . . . 35
2.3.2 Semantische Strukturierung . . . 37

3 CSS 38

3.1 CSS - Basics . . . 42
3.1.1 Schriften . . . 44
3.1.2 Farben . . . 46
3.1.3 Ränder . . . 47
3.1.4 Breite und Höhe . . . 49
3.1.5 Hintergründe . . . 51
3.1.6 Links . . . 53
3.1.7 Außen und Innenabstände . . . 54
3.1.8 das CSS Box Modell . . . 55
3.1.9 Einbinden von mehreren CSS Dateien . . . 60
3.2 Layouting with CSS . . . 60
3.2.1 Navigationsmenüs . . . 65

3.3 Advanced CSS Techniques . . . 68
3.3.1 text-shadow . . . 68
3.3.2 box-shadow . . . 70
3.3.3 CSS - only Slider . . . 70
3.3.4 Bouncing Balls with CSS . . . 71
3.3.5 border-radius . . . 71
4 Webseiten 73
4.1 Dynamische Webseiten . . . 75
5 WEBSERVER 78
5.1 HTTP und das Web . . . 78
5.1.1 Webserver und Protokolle . . . 78
5.2 Apache . . . 81
5.2.1 Einfuehrung und Installation . . . 81
5.2.2 Konfiguration . . . 84
5.2.3 Sicherheit . . . 88
5.2.4 Indexes . . . 94
5.2.5 SSI . . . 100
5.2.6 CGI . . . 103
5.2.7 PERL mit mod_perl.so . . . 107
5.2.8 Apache und PHP . . . 109
5.2.9 Die Log-Dateien des Apache . . . 111
5.2.10 Virtual Hosting . . . 114
5.2.11 Secure Socket Layer (SSL) . . . 123
5.3 Tomcat . . . 126
5.3.1 Der Apache Tomcat - Ein Applikationsserver . . . 127
5.3.2 Architektur des Tomcat . . . 127
5.3.3 Installation . . . 129
5.3.4 Konfiguration des Tomcat . . . 131
5.3.5 Anwendungen bereitstellen . . . 132
5.3.6 Virtual Hosting . . . 137
5.3.7 HTTPD to Tomcat interconnect . . . 138
5.4 Internet Information Services . . . 141
5.4.1 Herkunft und Installation der IIS . . . 141
5.4.2 Datensicherung . . . 145
5.4.3 Remoteverwaltung . . . 147
5.4.4 Bereitstellen von Webseiten . . . 147
5.4.5 Authentifizierung und Autorisierung . . . 152
6 Coding Guidelines 158
6.1 HTML . . . 158
6.2 CSS . . . 160

1 Einführung

ITSysAdminFwWebSK

Ziele

Der Lehrgang ITSysAdminFwWebSK soll Ihnen die Grundlagen der Webentwicklung näher bringen. Die Inhalte des Lehrgangs gliedern sich wie folgt:

Webdesign

- HTML
- CSS
- Entwicklung von Webseiten

Webserver

- Apache
- Tomcat
- IIS

Der Lehrgang wird Sie in den ersten beiden Kapiteln in die Programmierung mit der Markup-Language HTML vertraut machen. HTML ist die Sprache mit der Webseiten geschrieben werden. Das folgende Kapitel CSS wird Ihnen beibringen wie Sie mittels der Sprache CSS ihre erstellten HTML-Seiten optisch ansprechend gestalten können.
Diese beiden Kapitel sollen Sie dazu befähigen selbstständig Webauftritte mit mehrere Seiten in einem ansprechenden Design zu erstellen. Das Kapitel Entwicklung von Webseiten schließt die beiden Kapitel ab. Es wird dort ein komplettes Projekt umgesetzt werden.

Das zweite große Bereich behandelt das Thema Einrichtung und Betreuung von Webservern. Sie sollen hier die Einrichtung und Administration von einfachen Webservern lernen. Sie sollten danach auch in der Lage sein, einen Webserver an speziellere Bedürfnisse wie der Installation von serverseitigen Erweiterungen anzupassen und zu betreuen.

Das vorliegende Script soll Ihnen helfen, die im Unterricht erlernten Techniken im Selbststudium zu vertiefen. Es wurde versucht, dieses Script sehr umfangreich zu erstellen und es sind daher auch viele Aufgaben zur eigenen Bearbeitung hinterlegt. Es wird jedoch darauf hingewiesen, dass dieses Script den Unterricht und die aktive Mitarbeit an den Unterrichten nicht ersetzen soll und kann.

Der Besuch dieses Lehrgang setzt einige Grunderfahrungen voraus. Eine der Vorraussetzung zum Besuch dieses Lehrgangs ist es, den Lehrgang ITFwInfoVarbBw erfolgreich abgeschlossen zu haben. Es wird daher nicht auf die folgenden Themen explizit eingangen, jedoch Kenntnisse vorausgesetzt:

Vorraussetzungen

- Sicherer Umgang mit verschiedenen Betriebsystemen
- Kenntnisse im Bereich Netzwerktechnik, speziell:
  - FTP
  - HTTP
  - TCP/IP
  - DNS
- Kenntnisse in der Betreuung von Windows Server

The history of the Internet

Warum überhaupt HTML, CSS etc... Ich kann doch auch in C++ programmieren

Der größte Vorteil in der Programmierung mit HTML, CSS, und co. ist, dass Sie absolut Plattform unabhängig sind.
Eine Webseite, die in validem HTML mit validem CSS geschrieben ist, wird auf jedem Gerät gleich dargestellt. Egal ob Sie einen PC mit Windows, einen Mac mit OSX, ein Android mit Chrome oder ein iPad mit Safari nutzen, Sie haben immer exakt das gleiche Ergebnis. Sofern Sie sich an die Standards halten. Der eigentlich noch bessere Vorteil, ist, dass sie keinerlei Software auf dem Client installieren müssen. Ein Browser ist immer von Anfang an vorhanden. Selbst komplexeste Anwendungen, die zusätzlich zu HTML noch Datenbanken und Programmiersprachen wie Ruby, Pearl, Python oder PHP nutzen, laufen alle serverseitig und übermitteln Ihnen nur reines HTML.

Benötigte Software

Für das Erstellen Ihrer ersten eigenen Webseiten brauchen Sie gänzlich wenig Software. Theoretisch reicht Ihnen ein einfacher Editor und ein Browser, um Ihre Ergebnisse zu betrachten

Browser

Wir empfehlen Ihre erstellten Seiten immer auf mehreren Browsern zu testen. Die wohl wichtigsten Browser sind nachgehend aufgelistet

chrome	Google Chrome
Safari	Apple Safari
mozilla Firefox	Firefox
OPERA software	Opera

FTP-Client

Zur Übertragung der HTML - Seiten auf den Server wird ein FTP-Client benötigt. FileZilla ist OpenSource und daher unsere erste Wahl.

	FileZilla FTP Client

Editor

Es würde Notepad als Editor für die Erstellung komplexester Anwendungen reichen. Die Software Notepad++ unterstütz jedoch Syntax Highlighting in vielen verschiedenen Programmiersprachen und ist nebenbei auch noch kostenlos.

	Notepad++ Editor

2 HTML

Websites, Homepages, Webpages, etc....

Die oft gehörten Begriffe

- Website,
- Homepage,
- Webpage,
- Internetseite,
- HTML-Seite und
- Webauftritt

stellen im groben Sinne Synonyme dar.
Im normalen Sprachgebrauch kann und wird zwischen diesen Begriffen nicht unterschieden. Im engeren Sinne lassen sich jedoch kleinere Unterschiede feststellen.

Beispiel:

Ein **Webauftritt** kann aus mehreren einzelnen **HTML**-Seiten bestehen.

Beispiel 2:

Korrekt betrachtet ist eine **Homepage** nur eine **einzelne Seite**. Erst wenn es **mehrere Pages** sind, kann von einer **Site** gesprochen werden.

Da umgangsprachlich diese Abgrenzungen jedoch nicht berücksichtigt werden, ist dies relativ irrelevant.

Was ist HTML

HTML ist die Auszeichnungssprache für die Inhalte einer Webseite. Mittels **HTML** wird die Strukturierung von Inhalten wie Texten, Bilder, Hyperlinks oder anderen Ressourcen vorgenommen.

HTML-Dokumente ".html" oder ".htm" bilden die Grundlage des World Wide Web **WWW**. Es ist möglich in HTML nicht nur die sichtbaren Inhalte zu strukturieren, sondern auch zusätzliche Informationen, wie zum Beispiel den Autor, das Datum oder eine Beschreibung der Seite zu hinterlegen.

Die Definitionen, wie die Sprache aufgebaut ist und auch die Weiterentwicklung unterliegt dem World Wide Web Consortium **W3C**.

Derzeit ist die aktuelle Version **HTML5**

Das Acronym **HTML** steht für **H**yperText **M**arkup **L**anguage

2.1 Grundlagen HTML

Versionen von HTML

Die Standards für HTML werden durch das World Wide Web Consortium **W3C** entwickelt und veröffentlicht.

Version	Erscheindungsdatum	Merkmale
HTML	03.11.1992	Initial Release
HTML2	11.1995	Formulare werden eingeführt
HTML3.2	01.1997	Tabellen, Textfluss um Bilder, Einbindung von Applets
HTML4	12.1997	Stylesheets, Skripte und Frames
XHTML 1.1	2001	Eine Neuformulierung von HTML 4.01 mit Hilfe von **XML**
HTML 5	04.2009	

Grundbegriffe für HTML

Tag

Was ist nun ein HTML-TAG? - Als HTML-TAG wird ein einzelner HTML-"Befehl" bezeichnet.
Dabei kommt der Begriff TAG aus dem Englischen und hat dort die Bedeutung von "Etikett, Anhänger, Aufkleber, Marke, Kennzeichnung, Auszeichner". Am besten lässt es sich mit dem Begriff "Kennzeichnung" veranschaulichen. Man kennzeichnet Bereiche, dass diese in bestimmter Weise angezeigt werden sollen. Dabei kann man dann definieren: hier beginnt der Bereich und dort hört er wieder auf.

Ein Tag ist ein Markup für einen bestimmten Bereich. Es gibt 3 Arten von Tags:

- öffnende Tags - `<div>`
- schließende Tags - `</div>`
- selbstschließende Tags - `<img/>`

Element

Ein Bereich der von 2 zusammengehörigen Tags umschlossen wird oder eine selbstschließender Tag ist, nennt man Element.

Ein Element besteht somit fast immer aus den 3 Bereichen:

- `<öffnender Tag>`
- Inhalt des Elements
- `</schließender Tag>`

Attribut

Ein Attribut kann einem Element zugewiesen werden. Es können einem Element mehrere verschiedene Eigenschaften (Attribute) zugewiesen werden. Ein Element muss aber keine Attribute besitzen.

Die Zuweisung von Attributen an Elemente erfolgt innerhalb des öffnenden Tags.
`<p class="class_name">`

HTML - Struktur

Eine HTML Datei besteht grundsätzlich aus 3 Bereichen:

1. der Dokumenttypdeklaration **(Doctype)** ganz am Anfang der Datei, die die verwendete Dokumenttypdefinition **DTD** angibt, z. B. **HTML 4.01 Strict**,
2. dem HTML-Kopf **(HEAD)**, der hauptsächlich technische oder dokumentarische Informationen enthält, die üblicherweise nicht im Anzeigebereich des Browsers dargestellt werden,
3. und dem HTML-Körper **(BODY)**, der jene Informationen enthält, die gewöhnlicherweise im Anzeigebereich des Browsers zu sehen sind.

Damit gelangen wir zu folgender Struktur:

Listing 2.1: HTML - Struktur

```html
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
<html>

    <head>
    </head>

    <body>
    </body>

</html>
```

HTML - Doctype

Der DOCTYPE in HTML dient nur dem Zweck, den Rechnern, die diese Seite öffnen, eine Vorinformation zu geben. Der Computer kann sich dann direkt darauf einstellen, welche Tags folgen können und wie er diese interpretieren muss. Ein normaler Webbrowser wie Google Chrome, Firefox, Safari oder auch Opera (**NEIN**, der Internet Explorer **IE** ist kein **normaler** Browser) ignoriert diese Angabe gänzlich. Diese Angaben haben erst eine Relevanz, wenn z.B. automatisierte Parse-Roboter (Crawler) die Seite öffnen. Ein Crawler ist ein Programm, dass eine Webseite abruft und die ausgelesenen Inhalte verarbeitet. Im Großteil der Fälle dient der Abruf der Seiten der Indizierung der Inhalte für schnellere Suchanfragen.

Um diesen Vorgang noch zu verbessern, gibt es neue Techniken und Möglichkeiten um die semantischen Angaben einer Seite durch viele Zusatzinformationen zu optimieren. Dies wird erreicht durch zusätzlich neue Tags, die für den menschlichen Betrachter nicht sichtbar sind.

Werden zusätzliche Sprachen also in HTML eingebunden, müssen diese im DOCTYPE deklariert werden, so dass die Maschine weiß, wie Sie damit umzugehen hat.

Beispielhaft verschiedene DOCTYPES wären:

DOCTYPE	
`<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"> <"http://www.w3.org/TR/html4/loose.dtd">`	HTML 4.01
`<!doctype html>`	HTML5
`<?xml version="1.0" encoding="UTF-8"?>/span> >`	XHTML + rdfa

Relevant ist in diesem Lehrgang lediglich der DOCTYPE `<!doctype html>`. Dieser DOCTYPE gibt an, dass das folgende Dokument in HTML5 geschrieben wurde

HTML - HEAD

Der `<head>` - Bereich einer HTML-Datei dient der Übergabe von zusätzlichen Daten, die nicht direkt zum anzuzeigenden Inhalt gehören. Diese Informationen können zum Beispiel externe Ressourcen wie CSS - oder Javascript-Dateien sein. Oder es können Informationen über den Author, das Erstellungsdatum, oder eine Beschreibung sein.

Mögliche Elemente im `<head>`

Element	Bedeutung	Anwendung
title	TITEL	Der in den Tabs eines Browsers angezeigte Titel der Seite
meta	ZUSATZINFORMATIONEN	z.B. über den Author der Seite oder das Veröffentlichungsdatum
base	BASIS-URI oder -Frame	
link	Verknüpfung zu einer Ressource	z.B. externe Stylesheets
script	CODE einer anderen Programmiersprache	hauptsächlich für die Einbindung von Javascript genutzt
style	Layout - Eigenschaften	
object	externe Dateien	

Das folgende Listing zeigt einen beispielhaften `<head>` eines üblichen HTML - Dokuments

Listing 2.2: Mögliche Elemente im head

HTML

```
<head>


    <meta charset="utf-8">
    <meta name="description" content="Development Framework">
    <meta name='author' content='Kevin Siegerth'>


    <title>ITSysAdminFwWebSK</title>


    <link rel="stylesheet" media="screen" href="/css/style.css" >

    <script type="text/javascript" src="etc/markItUp.js"></script>

    <style type="text/css" media="print,screen"><!--
        h1 {
            color:green;
        }
    --></style>

</head>
```

HTML - BODY

Der <body> eines HTML - Dokuments beinhaltet die Daten, die im Browser angezeigt werden sollen. Er beinhaltet somit alle Elemente wie

- Links,
- Tabellen,
- Bilder,
- Texte,
- etc...

Das folgende Listing zeigt einen Beispielhaften Body für eine einfache <HTML> Seite.

Listing 2.3: HTML - BODY

```html
<body>

<h1>Willkommen auf meiner Webseite</h1>

<h2>Hier eine Liste</h2>

<ul>
    <li>Links</li>
    <li>Tabellen</li>
    <li>Bilder</li>
    <li>Texte</li>
    <li>etc...</li>
</ul>

<h2>Und hier noch eine kleine Tabelle</h2>

<table>
    <tr>
        <td>Eintrag</td>
        <td>1</td>
    </tr>
    <tr>
        <td>Eintrag</td>
        <td>2</td>
    </tr>
</table>

</body>
```

Auftrag

Erstellen Sie Ihre erste eigene HTML-Seite

Ihre erste Seite soll bereits folgende Merkmale aufweisen können:

- korrekte DOCTYPE Definition
- einen Header mit Informationen zum Author und eine aussagekräftige Beschreibung der Seite
- einen Body-Bereich
- einen anzuzeigenden Text: "Hello World!"

Laden Sie diese Datei anschließend auf den für Sie bereitgestellten Webserver und betrachten Sie ihr Ergebnis!

2.2 Inhalte in HTML

TAGS in HTML

Wie bereits erwähnt werden Elemente in **HTML** parallel zu **XML** meist durch einen öffnenden und schließenden Tag abgegrenzt. In HTML nennen sich diese Begrenzer **tag**.

öffnender HTML Tag

```
<span>
```

schließender HTML Tag

```
</span>
```

daraus ergibt sich die folgende Syntax für z.B. eine Überschrift
`<h1>` Überschrift `</h1>`

Listing 2.4: schließender HTML Tag

```
<h1>Inhalt</h1>

<span>Inhalt</span>

<div>Inhalt</div>
```

Selbstschließende Tags

HTML kennt zu den normalen Tags, die einen Inhalt einschließen, auch selbstschließende Tags. Es handelt sich bei diesen Tags entweder um Tags für die Einbindung von externen Ressourcen oder Funktionselemente wie Buttons.

Ein selbstschließender Tag - wie der Name schon sagt - schließt sich selbst d.h. er benötigt keinen eigenen schließenden Tag. Ein Beispiel hierfür ist der Tag für die Ausgabe eines Bilder oder ein Button zum Abschicken eines Formulars. Der Tag

wird innerhalb des öffnenden Tags direkt wieder mit einem Slash "/" geschlossen
`<img />`

Seit **HTML5** ist es nicht mehr nötig die selbstschließenden Tags auch explizit mit dem Slash zu schließen, da durch die richtige Umsetzung der Spezifikation der nachfolgende Inhalt automatisch wieder ausgeschlossen wird. Auf Grund der Einheitlichkeit oder auch Umsetzungsprobleme von den leider immer noch zu stark vertetenen alten Browsern sollte der Tag trotzdem explizit geschlossen werden - **HTML5** stellt trotzdem auch alles richtig dar.

Listing 2.5: Selbstschließende Tags

```html
<input type="text" />
<img src="baerchen.jpg" />
```

2.2.1 Paragraphen

Paragraphen und Zeilenumbrüche

Paragraphen dienen wie überall dazu, einen längeren Text in logische Abschnitte zu unterteilen. Genauso wie bei Word, sollte hierzu **NICHT** der einfache Zeilenumbruch [WORD: SHIFT-ENTER] genutzt werden. Sondern es sollte dafür ein neuer Paragraph erstellt werden [WORD: ENTER].
In HTML wird dies durch den Tag `<p>` realisiert. Der einfache Zeilenumbruch ist der selbstschließende Tag `<br/>`

Beispiel

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus lacinia nunc eu quam vestibulum pretium. Aliquam ullamcorper turpis ut nisi imperdiet dignissim. Quisque lacinia suscipit justo, ut mattis nisl dignissim ac. Morbi eget nibh eu neque auctor laoreet sed id leo. Suspendisse potenti. Maecenas sed sapien purus. Ut molestie pharetra justo sed hendrerit. Proin id odio vitae diam ullamcorper rhoncus. Nulla facilisi.

Nam non nisl nisl. Vivamus ac pulvinar erat. In hac habitasse platea dictumst. Praesent tincidunt, neque eu pellentesque luctus, risus tortor porttitor sapien, et molestie nibh est sit amet neque. Integer sollicitudin lacus hendrerit risus dictum nec egestas ligula condimentum. In hac habitasse platea dictumst. Praesent ornare magna eu nisl ullamcorper et tincidunt dolor dapibus. Phasellus viverra elit ac nisi semper sed feugiat purus interdum. Nulla facilisi. Phasellus ornare tincidunt facilisis. Sed fringilla, dui adipiscing blandit rhoncus, mi dui mollis erat, sit amet facilisis risus ipsum id ipsum. Mauris et blandit felis. Mauris ligula nunc, cursus et ultricies quis, pharetra in arcu.

Donec suscipit mi sed erat tempus tempus. Duis a magna sapien. Aenean elit nibh, dignissim dignissim vestibulum eu, accumsan quis nibh.
Sed ut nisi eget neque volutpat iaculis vel eu ante.
Sed placerat, augue vitae fringilla placerat, metus neque viverra odio, in laoreet nunc nibh euismod orci. Aliquam eu purus dolor.
Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Nam eu ullamcorper dui.

Listing 2.6: Beispiel

```html
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus lacinia nunc eu quam vestibulum pre      Aliqu

<p>Nam non nisl nisl. Vivamus ac pulvinar erat. In hac habitasse platea dictumst. Praesent tincidunt, neque

<p>Donec suscipit mi sed erat tempus tempus. Duis a magna sapien. Aenean elit nibh, dignissim dignissim vestibulu
Sed ut nisi eget neque volutpat iaculis vel eu ante.<br />
Sed placerat, augue vitae fringilla placerat, metus neque viverra odio, in laoreet nunc nibh euismod orci. Aliquam
Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Nam eu ullamcorper d
```

HTML ignoriert Umbrüche im Code

Zeilenumbrüche sollten nur innerhalb von Elementen genutzt werden, und nicht für das Layout "missbraucht" werden.

Beispiel:

Dies ist ein einzelner Paragraph. Dies ist der zweite Satz des Paragraphs. Dies ist der dritte Satz des Paragraphs

Listing 2.7: Beispiel:

```
<p>Dies ist ein einzelner Paragraph.
Dies ist der zweite Satz des Paragraphs. Dies ist der dritte Satz des Paragraphs</p>
```

HTML

Beispiele für Zeilenumbrüche:

Dies ist ein einzelner Paragraph.
Dies ist der zweite Satz des Paragraphs.
Dies ist der dritte Satz des Paragraphs

Listing 2.8: Beispiele für Zeilenumbrüche:

```
<p>Dies ist ein einzelner Paragraph.<br />Dies ist der zweite Satz des Paragraphs.<br />Dies ist der dr
```

HTML

Zusammenfassung

- <p> Ein **Paragraph** in HTML entspricht dem Tastenkommando "**ENTER**" in Word.
-
 Ein **Zeilenumbruch** in HTML entsprichet dem Tastenkommando "**SHIFT-ENTER**" in Word.

Erstellen Sie mehrere verschiedene Paragraphen. Machen Sie sich vertraut mit den Unterschieden zwischen <br /> und <p>

2.2.2 Textformatierung

Einfache Formatierungen

HTML ist nicht dafür gedacht, Text optisch zu formatieren. Jedoch haben manche Element von Haus aus eine optische Hinterlegung. Diese ist jedoch noch aus den Anfängen von HTML übrig geblieben. Eine Formatierung eines Textes erfolgt ausschließlich über CSS. Die im Folgenden kurz vorgestellten Formen der Formatierung mit den Mitteln von HTML dienen der semantischen Hinterlegung von Text.

This text is bold

This text is italic

`This is computer output`

~~Dieser Text ist durchgestrichen~~

Listing 2.9: Einfache Formatierungen

```html
<p><strong>This text is bold</strong></p>

<p><em>This text is italic</em></p>

<p><pre>This is computer output</pre></p>

<p><del>Dieser Text ist durchgestrichen</del></p>
```

b vs. strong

In den meisten normalen Browsern werden die beiden TAGs `<b>` und `<strong>` identisch dargestellt.
Sie dürfen jedoch nicht als komplementär angesehen werden.

Unterschiede:

`<b>`

Der Tag b dient lediglich der optische fetteren Darstellung eines Ausdrucks. Inhaltlich (semantisch) hat dieser Ausdruck nicht mehr Bedeutung als normaler Text.

`<strong>`

Der Tag strong hat zusätzlich zur optischen Gestaltung eine semantische Bedeutung: Ein Ausdruck der mit strong markiert wurde ist als wichtiger zu betrachten, als normaler Text.

Warum sollte mich das interessieren wenn beide Tags exakt gleich dargestellt werden?

Beispiel:

Ein Screenreader (Gerät, dass den Inhalt einer Homepage vorlesen kann - wird benutzt von Menschen mit Sehbehinderungen) würde bei dem Tag strong diesen Ausdruck auch betont vorlesen. Dies ist bei dem Tag `<b>` nicht der Fall.

em vs. i

Parallel zu den tags `<b>` und `<strong>` verhalten sich auch die Tags `<em>` und `<i>`. `<em>` hat eine semantische Bedeutung und `<i>` besitzt lediglich die optischen Eigenschaften für den Browser.

2.2.3 Überschriften

Headings - Überschriften

Überschriften dienen in **HTML** der Strukturierung der Inhalte. HTML kennt 6 Stufen von Überschriften. Je niedriger die Nummer, desto wichtiger (besser) ist die Überschrift. [Schulnotensystem]

Eine Überschrift `<h1>` stellt somit immer die wichtigste Überschrift auf einer Seite dar.

> Eine `<h1>` - Überschrift sollte nur einmalig auf einer Seite vertreten sein. Als Überschrift für genau den Artikel, der auf der Seite zu finden ist.

Eine Suchmaschine wie zum Beispiel **Google** bewertet die Wichtigkeit des Textes einer Überschrift nach der Klassifizierung mit dem jeweiligen `<hX>` Tag.

Beispiel der standardmäßigen Darstellung von Überschriften in einem Browser

Heading 1
Heading 2
Heading 3
Heading 4
Heading 5
Heading 6

Listing 2.10: Heading 4

```html
<h1>Heading 1</h1>
<h2>Heading 2</h2>
<h3>Heading 3</h3>
<h4>Heading 4</h4>
<h5>Heading 5</h5>
<h6>Heading 6</h6>
```

HTML

Hierarchische Strukturierung

Erstellen Sie nur mittels Überschriften eine HTML-Seite, die die folgende Struktur logisch abbildet:

- HTML
  - Grundlagen
  - Inhalte
    - Tabellen
    - Listen
    - Formulare
      - Radio-Buttons
      - Checkboxes
      - Input
      - Textarea
    - Überschriften
  - Struktur
    - Aside
    - Header
    - Footer
    - Navigation

2.2.4 Listen

Nummerieren, Definieren, oder einfach nur so....

HTML kennt **3** verschiedene Möglichkeiten Listen zu erstellen:

- unsortierte Listen
- sortierte Listen
- Definitionslisten

Unsortierte Listen - unordered lists [ul]

Die einfachste Form einer Liste ist die Aufzählung.
In HTML wird die unsortierte Liste mit dem Element `<ul>` eingeleitet und dementsprechend mit `</ul>` wieder geschlossen.
Die einzelnen Elemente innerhalb einer Liste werden mit dem Tag `<li>` definiert.

> Gemäß den Spezifikation von HTML ist es nicht nötig, jedes einzelne `<li>` auch wieder explizit zu schließen. Um eventuelle Darstellungsprobleme im vornherein zu verhindern und auch einen einheitlichen Coding-Stil zu gewährleisten, sollten alle `<li>` auch wieder mit `</li>` geschlossen werden.

Beispiel für eine einfache Liste

- Milch
- Äpfel
- Bananen
- Joghurt

Listing 2.11: Beispiel für eine einfache Liste

```html
<ul>
    <li>Milch</li>
    <li>Äpfel</li>
    <li>Bananen</li>
    <li>Joghurt</li>
</ul>
```

Geordnete und ungeordnete Listen

Sie haben soeben gelernt wie sie eine normale Aufzählungsliste erstellen können. Jedoch haben Sie die teilweise die Anforderung innerhalbe einer Liste eine Rangordnung darzustellen - zum Beispiel eine Ergebnisliste eines Wettkampfs. Sie könnten nun manuell in Ihrer Liste jeweils vor die Elemente noch eine Zahl schreiben. Dies ist jedoch eine sehr umständlicher und auch nicht der korrekte Weg. Betrachten Sie den folgenden Code und vergleichen das Ergebnis zwischen einer "ordered" und einer "unordered" List.

ungeordnete Liste	nummerierte Liste
■ Listenelement ■ Listenelement ■ Listenelement ■ Listenelement ■ Listenelement	1. erstes Element 2. zweites Element 3. drittes Element 4. viertes Element 5. fünftes Element

Listing 2.12: Geordnete und ungeordnete Listen

```html
<ul>
    <li>Listenelement</li>
    <li>Listenelement</li>
    <li>Listenelement</li>
    <li>Listenelement</li>
    <li>Listenelement</li>
</ul>


<ol>
    <li>erstes Element</li>
    <li>zweites Element</li>
    <li>drittes Element</li>
    <li>viertes Element</li>
    <li>fünftes Element</li>
</ol>
```

Selbststudium

Aufgabe

Erstellen Sie eine Liste: Die aktuelle Weltranglisten-Ordnung der Fußball - Nationalmannschaften.
Welche Art von Liste würden Sie nutzen?

Aufgabe:

Recherchieren Sie in Eigenarbeit, wie man eine Definitionsliste erstellt.
Erstellen Sie eine Definitionsliste mit mehreren verschiedenen Dienstgraden der Bundeswehr.

Aufgabe:

Stellen Sie sich den Verzeichnisbaum Ihres Windows - Nutzerverzeichnisses vor.
Sie haben mehrere verschiedene Ordner, die auch Unterordner enthalten.
Welche Möglichkeiten sehen Sie, dies in HTML umzusetzen.
Erstellen Sie eine Liste mit 3 Ordnern und jeweils 5 Unterordnern.

2.2.5 Bilder

Bilder in HTML

Bilder werden in HTML mittels des `<img>` tag eingebunden. Der Image-Tag ist auch ein selbstschließender Tag. Die wichtigsten Attribute, die der `<img>`-Tag enthalten darf sind:

src

Hier wird der Pfad des Bildes angegeben, dass eingebunden werden soll. Die Pfadangabe ist relativ von der HTML-Datei. Alternativ kann der Pfad auch absolut angegeben werden.

alt

Das "alt"-Attribut gibt eine **alternative** Darstellung an. Falls ein User innerhalb seines CLients die Darstellung von Bildern abgeschaltet hat, wird ihm dieser Text präsentiert. Des weiteren ist diese Beschreibung relevant in Bezug auf Barrierefreiheit. Personen, die einen ScreenReader nutzen, wird an der Stelle des Bildes diese Beschreibung vorgelesen.

title

Das "title"-Attribut gibt dem Bild einen Titel. Dieser wird zum Beispiel beim MouseOver angezeigt

Listing 2.13: title

```html
<img src="./img/1342446643_Logo_512.png" alt="Das Logo von HTML5. Ein oranges Schild
im Hintergrund mit dem Buchstaben 5 als Aufdruck" title="HTML5 Logo"/>
```

HTML

Selbststudium

Aufgabe

Binden Sie mehrere verschiedene Bilder in Ihre Seiten ein.

Testen Sie dabei die Möglichkeiten der Einbindung mit relativen und auch absoluten Pfadangaben.

Vergeben Sie für jedes Bild die Attribute "alt" und "title"

2.2.6 Links

Das wichtigste im Web

Links sind der wichtigste Bestandteil des **WWW**.

Ein Link ist eine Verknüpfung zu einer Seite, Datei oder auch einem bestimmten Abschnitt innerhalb eines Dokuments. Durch Klicken eines Links erhält der Browser die Anweisung, die Ressource aufzurufen.

Sie ermöglichen die Navigation nicht nur innerhalb einer Seite - sondern auch auf andere Seite.

Ohne direkte Links zu anderen Seiten, wären Sie immer dazu gezwungen, die **URI** einer Seite einzeln im Browser einzugeben und dann ENTER zu drücken.

HTML Link Syntax

Ein Link macht den kompletten Inhalt, der innerhalb des Tags `<a>` steht, klickbar.

Es kann somit nicht nur Text, sondern auch Bilder oder ganze Container mit einem link versehen werden. Das wichtigste Attribut eines links ist "href". Die Abkürzung steht für Hyperlink Reference und gibt das Ziel an, auf welches der Link verweisen soll.

Listing 2.14: HTML Link Syntax

```html
<a href="#">Link</a>
```

HTML

Link zu einer E-Mail Adresse

Mit Hilfe der normalen Links können auch Verknüpfungen zu e-Mail Adressen erstellt werden. Funktionsweise: Ein Klick auf solch einen Link öffnet das im Betriebssystem als Standard eingestellte e-Mail-Programm und übergibt die hinterlegte Email-Adresse. Schick mir eine Mail

Listing 2.15: Link zu einer E-Mail Adresse

```html
<a href="mailto:test@example.org">Schick mir eine Mail</a>
```

Links in neuen Fenstern

Mittels der Attribute eines Links kann eingestellt werden, dass ein Link z.B. in einem neuen Fenster geöffnet werden soll. Dies geschieht durch Angabe des Attributes: "target"
Dieses Attribute ist jedoch lediglich noch ein Überbleibsel aus der Zeit, in denen HTML noch Frames genutzt wurden. Es sollte mittlerweile komplett darauf verzichtet werden, dem User vorzugeben, wie ein Link geöffnet wird. Das Attribut "target" ist mittlerweile auch aus den Spezifikationen für HTML entfernt worden. Diese Einstellung machen die Browser bzw. der User für sich selbst. Für den Fall, dass Sie jedoch explizit einen link in einem neuen Fenster (Tab) geöffnet haben möchten, stellen wir Ihnen diese "deprecated" - Lösung trotzdem vor.

Mögliche Werte für das target Attribut:

Wert	Auswirkung
_blank	neues Fenster
_self (standard)	in eigentlichen Fenster
_top	im eigentlichen Fenster (nur bei Nutzung von Frames relevant)
_parent	im Eltern-Frame (nur bei Nutzung von Frames relevant)
framename	im benannten Frame

Beispiel

Google in neuem Fenster

Listing 2.16: Beispiel

```html
<a href="http://www.google.de" target="_blank">Google in neuem Fenster</a>
```

verschiedene Beispiele für Links

Bilder als Links

Ein Klick auf den Hund öffnet das Bild in seinem Ursprung

Container als Links

Ein komplettes DIV als Link

Buttons als Links

Button

Text als Links

Dieser Text ist ein Link

Listing 2.17: Text als Links

```
<!-- IMAGE-Link -->
<a href="./img/1342449951le-pic.jpg"><img class="small" src="./img/1342449951le-pic.jpg" alt="Boo - the b

<!-- DIV-Link -->
<a href=""><div class="blackbox">Ein komplettes DIV als Link</div></a>

<!-- button-Link -->
<a href=""><button class="button">Button</button></a>

<!-- Text-Link -->
<a href="">Dieser Text ist ein Link</a>

```

Downloads

Wie stelle ich nun einen Donwload-Link bereit?

Stellen Sie sich einen klassischen Download vor: Sie klicken auf einen link, und kurze Zeit später fängt ihr Browser an, diese Datei herunter zu laden. Doch warum macht dies der Browser? Ist der Begriff Donwload ein eigenes Attribut innerhalb von HTML?
Sie können das Verhalten eines "Download" nicht steuern.

Ein Download ist die Maßnahme eines Browsers auf eine Datei, mit der er nichts anfangen kann.

MP3 Download

Früher war es einfach, z.B. seinem Kameraden eine MP3-File zukommen zu lassen. Man hat die MP3-Datei einfach auf seinen Webserver geschoben, dem Kameraden den Link geschickt und schon hat der Browser die Datei für die lokale Nutzung heruntergeladen.
Mittlerweile können die modernen Browser aber auch intern MP3's abspielen. Ihr Kamerad wird Ihren Link also anklicken und der Browser spielt die Musik direkt ab. Ihr Kamerad müsste mittlerweile explizit auswählen, dass er die MP3 speichern möchte.

PDF Download

Selbiges gilt mittlerweile auch für .pdf - Dateien

Bilder Download

Bei dem Begriff Bilder-Download denken Sie sich bestimmt: Wieso denn Bilder Download... Jeder Browser kann doch fast alle Arten von Bildern anzeigen. Mit dieser Aussage hätten Sie jedoch nur teilweise Recht. Es gibt spezielle Nur-Text-Browser. Ein Link auf ein Bild würde es in diesem Fall nicht in Ihrem Browser anzeigen, sondern einen Download starten.

Selbststudium

Aufgabe

Erstellen Sie 5 verschiedene HTML Seiten. Erzeugen Sie auf jeder Seite Links zu den jeweiligen anderen Seiten. Nutzen Sie auf jeder Seite mindestens einmal ein Bild als Link.

Erstellen Sie eine Seite, auf der Sie mehrere verschiedene Word Dokumente zum Donwload anbieten

2.2.7 Tabellen

Table

[Englisch]: a table

Das wohl erste an das man denkt, wenn man das Wort Table hört, ist wohl ein Tisch.
Doch was hat dieser Tisch mit **HTML** zu tun?

Natürlich nichts! Jedoch hat das Wort Table im Englischen 2 Bedeutungen:

- Tisch
- Tabelle

Nachdem es uns wohl im Schwerpunkt nicht darum gehen wird, möglichste viele Bilder von Tischen auf unseren Seiten unterzubringen -- außer wir sind ein Schreiner mit Online-Shop -- wenden wir uns dem Thema **Tabellen in HTML** zu.

Tabellen in HTML

Das folgende Beispiel zeigt den minimalen Aufbau einer Tabelle.
Tabellen bestehen aus Reihen und Zeilen (rows tr und columns td). Der `<tr>`-Tag leitet in diesem Beispiel eine Zeile ein. Innerhalb dieser Zeile erstellen wir mit dem `<td>`-Tag eine Zelle. Ein `<tr>` kann dabei eine beliebige Anzahl an Zellen haben. Diese werden dann innerhalb der einen Zeile von links nach rechts nebeneinander ausgegeben.

> Recherchieren Sie, wofür die Abkürzungen `<tr>`, `<th>` und `<td>` stehen könnten. Es erleichtert Ihnen auf jeden Fall die Arbeit, wenn Sie den vollen Namen zu einem Tag kennen.

Listing 2.18: Tabellen in HTML

```html
<table>
        <tr>
                <td>Zelleninhalt</td>
        </tr>
</table>
```

Sinnvolle Tabellen

Das folgende Beispiel stellt nun schon eine etwas sinnvollere Tabelle dar.

Zelleninhalt	Zelleninhalt	Zelleninhalt
Zelleninhalt	Zelleninhalt	Zelleninhalt
Zelleninhalt	Zelleninhalt	Zelleninhalt

Listing 2.19: Sinnvolle Tabellen

```html
<table>
        <tr>
                <td>Zeileninhalt</td>
                <td>Zeileninhalt</td>
                <td>Zeileninhalt</td>
        </tr>
        <tr>
                <td>Zeileninhalt</td>
                <td>Zeileninhalt</td>
                <td>Zeileninhalt</td>
        </tr>
        <tr>
                <td>Zeileninhalt</td>
                <td>Zeileninhalt</td>
                <td>Zeileninhalt</td>
        </tr>
</table>
```

Semantik in Tabellen

im folgenden Beispiel wurde unsere Tabelle um die beiden Elemente

- `<thead>` und
- `<tbody>`

ergänzt.

Diese beiden Elemente dienen dazu innerhalb der Tabelle zwei verschiedene Bereiche zu definieren:

- Den Tabellenkopf mit der Beschreibung und
- die Tabelleninhalte

Kopfzelle 1	Kopfzelle 2	Kopfzelle 3
Zelleninhalt	Zelleninhalt	Zelleninhalt
Zelleninhalt	Zelleninhalt	Zelleninhalt

Kopfzelle 1	Kopfzelle 2	Kopfzelle 3
Zelleninhalt	Zelleninhalt	Zelleninhalt

Warum sollte man in Tabellen verschiedene Bereiche definieren können?

- Überlegen Sie sich ein Beispiel aus der Praxis, in dem in einer Tabelle zwei Bereiche vorkommen.
- Ich habe mit CSS die Möglichkeit einzelne Zeilen auch bunt einzufärben, um damit Übersicht zu schaffen. Warum sollte ich also trotzdem noch mittels HTML die Kopfzeile definieren?

Listing 2.20: Warum sollte man in Tabellen verschiedene Bereiche definieren können?

HTML

```html
<table>
        <thead>
                <tr>
                        <th>Kopfzelle 1</th>
                        <th>Kopfzelle 2</th>
                        <th>Kopfzelle 3</th>
                </tr>
        </thead>
        <tbody>
                <tr>
                        <td>Zelleninhalt</td>
                        <td>Zelleninhalt</td>
                        <td>Zelleninhalt</td>
                </tr>
                <tr>
                        <td>Zelleninhalt</td>
                        <td>Zelleninhalt</td>
                        <td>Zelleninhalt</td>
                </tr>
                <tr>
                        <td>Zelleninhalt</td>
                        <td>Zelleninhalt</td>
                        <td>Zelleninhalt</td>
                </tr>
        </tbody>
</table>
```

TableEnd?

1. Finden Sie heraus ob es auch einen Bereich für Fußzeilen gibt
   Falls ja, testen Sie an selbst gewählten Beispielen, wie sich dieser verhält.
2. Denken Sie sich ein eigenes Beispiel aus, warum es solch einen Bereich innerhalb einer Tabelle geben sollte.

Experimente mit Tabellen

Mit Hilfe der Attribute "colspan" und "rowspan" ist ein möglich, dass sich einzelne Zellen über mehrere Zeilen bzw. auch Spalten erstrecken können. Das folgende Beispiel macht dies deutlich:

Kopfzelle 1	Kopfzelle 2	Kopfzelle 3			
Zelleninhalt			Zelleninhalt		
Zelleninhalt	Zelleninhalt	Zelleninhalt	Zelleninhalt		
Zelleninhalt	Zelleninhalt	Zelleninhalt	Zelleninhalt		
Zelleninhalt	Zelleninhalt	Zelleninhalt	Zelleninhalt		
	Zelleninhalt	Zelleninhalt	Zelleninhalt		
Zelleninhalt	Zelleninhalt				
Zelleninhalt			Zelleninhalt		
Zelleninhalt	Zelleninhalt	Zelleninhalt	Zelleninhalt	Zelleninhalt	Zelleninhalt
Zelleninhalt	A <table> in a <table>	Zelleninhalt	Zelleninhalt	Zelleninhalt	Zelleninhalt

- Erarbeiten Sie sich an Hand des Beispiels, wie die Nutzung von "colspan" und "rowspan" funktioniert
- Erstellen Sie selbst eine einfache Tabelle mit 3 * 3 Felder. Was passiert wenn sie dem genau in der Mitte liegenden Feld das Attribut "colspan=2" zuweisen

Listing 2.21: Experimente mit Tabellen

```html
<table class="example">
        <thead>
                <tr>
                        <th>Kopfzelle 1</th>
                        <th>Kopfzelle 2</th>
                        <th>Kopfzelle 3</th>
                </tr>
        </thead>
        <tbody>
                <tr>
                        <td colspan="3" class="green">Zelleninhalt</td>
                        <td>Zelleninhalt</td>
                </tr>
                <tr>
                        <td>Zelleninhalt</td>
                        <td>Zelleninhalt</td>
                        <td>Zelleninhalt</td>
                        <td>Zelleninhalt</td>
                </tr>
                <tr>
                        <td>Zelleninhalt</td>
                        <td>Zelleninhalt</td>
                        <td>Zelleninhalt</td>
                        <td>Zelleninhalt</td>
                </tr>
                <tr>
                        <td rowspan="2" class="green">Zelleninhalt</td>
                        <td>Zelleninhalt</td>
                        <td>Zelleninhalt</td>
                        <td>Zelleninhalt</td>
                </tr>
                <tr>
                        <td>Zelleninhalt</td>
                        <td>Zelleninhalt</td>
                        <td>Zelleninhalt</td>
                </tr>
                <tr>
                        <td>Zelleninhalt</td>
                        <td colspan="2" rowspan="2" class="green">Zelleninhalt</td>
                </tr>
                <tr>
                        <td>Zelleninhalt</td>
                        <td>Zelleninhalt</td>
                </tr>
                <tr>
                        <td>Zelleninhalt</td>
                        <td>Zelleninhalt</td>
                        <td>Zelleninhalt</td>
                        <td>Zelleninhalt</td>
                        <td>Zelleninhalt</td>
                        <td>Zelleninhalt</td>
                </tr>
                <tr>
                        <td>Zelleninhalt</td>
                        <td>
                                <table>
                                        <tr>
                                                <td>A</td>
                                                <td><span class="tag">table</span></td>
                                        </tr>
                                        <tr>
                                                <td>in</td>
                                                <td>a <span class="tag">table</span></td>
                                        </tr>
                                </table>
                        </td>
```

Selbststudium

Aufgabe

Bauen Sie nachfolgende Tabelle inhaltlich korrekt nach

laufende Nummer	Lehrgang		Teilnehmer	
	Nr.	Bezeichnung	PK	Name
1	269317	Java	123456-D-78901	Saturn
2			109876-I-54321	Venus
3			209684-N-43829	Mars
4			567483-G-78965	Pluto
5	268318	C++	keiner	

Ein paar kurze Überlegungen zu Tabellen

Wie wir in diesem Kapitel gemerkt haben, kann der Quellcode einer Tabelle leicht sehr komplex werden und man verliert schnell den Überblick, was dazu führt, das man ein `<tr>` nicht schließt ein komplettes `<td>` weg lässt. Deswegen ein paar Überlegungen zu Tabellen in HTML.

1. Nutzen Sie Tabellen nur dann, wenn Sie wirklich eine Tabelle benötigen. Überlegen Sie sich, ob nicht eine Liste reichen würden
2. Der Browser muss den Inhalt der kompletten HTML Tabelle "kennen", bevor er diese darstellen kann. Das bedeutet u.U. lange Ladezeiten, ohne dass der Leser etwas zu sehen bekommt. Dies könnte jedoch durch Aufsplitten von Riesentabellen in kleinere Einheiten umgangen werden.
3. Nuzten Sie Tabellen niemals, um das Layout einer kompletten Seite damit umsetzen zu wollen. Es kann einen sehr schnell dazu verleiten. Aber Tabellen sind dazu da, großen **Inhalt** optisch und semantisch geschickt darzustellen. Nicht für das Layout --> dafür gibt es CSS

2.2.8 Formulare

Formulare in HTML

Zu Beginn der Zeiten des Internet, war der Schwerpunkt lediglich bei der Bereitstellung von Daten. Der Austausch von Daten und Informationen wurde noch auf den herkömmlichen Transportwegen wie z.B. physikalischen Datenträgern, eMail oder FTP vollzogen. Mit der Einführung von HTML2 im Jahr 1995 wurde die Möglichkeit geschaffen mit HTML Formulare auszugeben. Anfangs wurden diese meist lediglich dazu benutzt etwas Interaktivität auf den Seiten zu schaffen. Es wurden Kontaktformulare angeboten, so dass nicht jedes mal eine angegebene Email Adresse kopiert und in sein eigenes lokales Email-Programm kopiert werden musste. Der User konnte direkt auf der Seite Kontakt mit dem Author aufnehmen.

Diese Idee entwickelte sich dahin weiter, dass es für die User auch möglich sein sollte, Kommentare zu hinterlassen, um zusammen auf einfache Weise zu diskutieren - die ersten einfachen Foren war geboren.

Aus dieser Idee heraus - die Inhalte von Webseiten zusammen zu erstellen - kamen dann die ersten Ideen für Content Management Systeme, professionelle Lösungen für Foren, Formulare zum Suchen auf den Seiten, etc.

Mittlerweile ist auf fast keiner Seite mehr nur noch statischer Inhalt angeboten, sondern der User kann überall interagieren - mit **Formularen**.
Egal ob es die Auswahl eines Hotels - das Hinterlassen eines Kommentars, das Posten eines Status auf Facebook oder die Suche nach einem Artikel auf Ebay ist - all das sind Formulare.

Ein Formular in HTML wird mittels des Tags `<form>` realisiert. Jedes Formular muss die Attribute "action" und "method" besitzen. Die beiden Attribute beschreiben, **wohin** und **wie** die Inhalte übergeben werden sollen

Mögliche Werte für die Attribute:

action

Wert	Auswirkung
" " (deprecated)	übergibt die Werte an sich selber (lädt dadurch die Seite neu)
formular.php	übergibt die Werte an die Datei formular.php

method

Wert	Auswirkung
post	überträgt die Daten im HTTP - Header
get	überträgt die Daten in der URL - z.B. http://google.de?suche=FORMULARINHALT

Ein erstes Formular

Submit

und der dazugehörige Code:

Listing 2.22: Ein erstes Formular

```html
<form action="" method="">
        <input type="text"/>
        <input type="submit"/>
</form>
```

das input element

das `<input>` Element stellt das meistgenutzte Eingabefeld innerhalb eines Formulars dar. Das `<input>` Element ist ein variables Element und kann je nach Definition durch das Attribut `<type>` ein verschiedenes Aussehen haben und verschiedene Daten aufnehmen.

Beispiele:

Wert	Auswirkung
type="text"	
type="password"	
type="search"	
type="submit"	Submit

Als weiteres wichtiges Attribut für das `<input>` Element betrachten wir das Attribut "maxlength". Durch setzen dieses Attributs ist es möglich, die maximale Eingabelänge eines Feldes zu begrenzen. "maxlength="40"" würde eine Feld zum Beispiel auf 40 Zeichen begrenzen

mehrzeilige Eingabebereiche

Wir haben bereits das Element `<input>` fuer Formulare kennen gelernt. Mit dessen Hilfe war es uns schon möglich ein Formular zu erstellen, um einfache Daten aufzunehmen. Stellen wir uns jedoch vor, wir möchten es dem Nutzer ermöglichen uns ein längeres Kommentar zu übersenden. Das Element `<input>` stellt uns nur eine einzige Zeile zur Verfügung. Dies macht es offensichtlich sehr schwer einen größeren Text einzutragen und diesen im Überblick zu behalten

Zur Lösung dieses Problems ist in HTML das Element `<textarea>` vorgesehen, welches einen beliebig großen Eingabebereich zur Verfügung stellt.

Beispiel:

Inhalt der Textarea

Listing 2.23: Beispiel:

```html
<textarea cols="75" rows="5">Inhalt der Textarea</textarea>
```

Checkboxes - Auswahlfelder

Man kann auch diesen Text klicken für die Auswahl der Checkbox ☐

Beispiel für Checkboxes

Hiermit stimmen Sie den AGB zu: ☐ Hiermit stimmen Sie der Zusendung von Newslettern zu ☐

Radioboxes - nur eine Auswahl ist möglich

Hatten Sie bei den Checkboxen noch die Auswahl, auch mehrere Elemente anzuwählen, bieten Radio-Boxes die Möglichkeit, genau eine Auswahl zu treffen.

Überlegen Sie sich 2 mögliche Beispiele, in denen Radio-Buttons sinnvoller sind, als Checkboxen

○ Mastercard

○ Visa

○ American Express

Finden Sie heraus, wie man Radio-Buttons gruppiert, so dass immer nur einer von mehreren markiert werden kann. (wie im obigen Beispiel)

Beschriftungen für die Felder

Jedes Eingabefeld kann das Attribut name besitzen. Das Attribut name gibt dem jeweiligen Eingabefeld einen **eindeutigen** Bezeichner. Dies ist speziell dann wichtig, wenn die Daten aus dem Formular später verarbeitet werden sollen. Das Script, an das das Formular gesendet wird, muss zuordnen können, welches Feld es z.B. wo in die Datenbank schreiben soll.

Des weiteren sollte jedem Formularelement eine eindeutige ID zugeordnet werden. Der Vorteil des "id" - Attributes ist, dass es uns ermöglicht eine eindeutige Beziehung zu diesem Feld zu erstellen - ein `<label>`

`<label>`

Der `<label>`-Tag dient lediglich dazu einen Text einem Formularelement zuordnen zu können. Wenn Sie einem Formular-Element mittels `<label for='ID DES FORMULARELEMENTS'>` zugeordnet haben, springen Sie bei dem Klick auf ein Label automatisch in das zugehörige Formularfeld.

Labels sollten sie speziell bei Checkboxes verwenden. Es ist einfacher den kompletten Text auch anklicken zu können, als nur das kleine Checkbox Feld

Möchten Sie mehrere Checkboxen zusammenfassen, kommen Sie mit dem `<label>`-Tag nicht weiter, das dieser nur einer einzigen id zugeordnet werden kann. Für die Gruppierung mehrer Formularelemente sieht html das `<fieldset>` mit `<legend>` als Beschreibung vor.

Ihre Haustiere Hund ☐ Katze ☐ Maus ☐

Listing 2.24: label

```html
<fieldset>

  <legend>Ihre Haustiere</legend>

  <label for="hund">Hund</label>
  <input name="haustier" type="checkbox" id="hund" value="hund" />

  <label for="katze">Katze</label>
  <input name="haustier" type="checkbox" id="katze" value="katze" />

  <label for="maus">Maus</label>
  <input name="haustier" type="checkbox" id="maus"  value="maus" />

</fieldset>
```

Aufgabe

Erstellen Sie ein einfaches Formular, dass zur Registrierung eines Nutzers auf einer Homepage dienen soll (Erstellen Sie nur den HTML Rahmen, das Formular muss noch keinerlei Funktionen besitzen):

Normalerweise findet man bei Registrierungsformularen diese verschiedenen Felder:

- Vorname
- Name
- Nutzername
- Passwort
- Passwort wiederholen
- Geburtsdatum
- Geschlecht
- Haustiere (Hund, Katze, Maus, Ente)
- Besschreibung oder Kommentar
- Newsletter (ja oder nein)

Überlegen Sie sich, welcher Typ von Eingabefeld für das jeweilige Feld von Vorteil wäre und nutzen Sie diesen. Gruppieren Sie ihr Formular in logische Abschnitte mit dem Tag `<fieldset>`.
Erarbeiten Sie sich die Funktionsweise von `<fieldset>` selbstständig.

2.2.9 Externe Inhalte

Karten, um die eigene Adresse zu visualisieren

Sie haben in HTML die Möglichkeit auch Inhalte von anderen Seiten einbinden zu können. Als bekanntes Beispiel fallen Ihnen sicherlich direkt YouTube oder GoogleMaps ein. In den meisten Fällen finden Sie direkt auf diesen Seiten ein Hinweis auf den Code, den Sie dazu nutzen müssen. Anbei ein Beispiel um seine eigene Adresse zu visualisieren.

The Creative Way

The Easy Way

Nützliche Browser-Infos

Listing 2.25: Nützliche Browser-Infos

```
<a href="http://www.wieistmeineip.de/cometo/?de"><img src="http://www.wieistmeineip.de/ip-adresse/?size
```

HTML

Aufgabe

Binden Sie auf Ihren HTML-Seiten folgende Inhalte ein:

- Ein Video von Youtube
- Eine Adresse von Google Maps
- Den IP-Banner von wieistmeineiip.de

2.3 Struktur in HTML

2.3.1 Blockelemente vs. Inline Elemente

Gruppierung von Elementen

HTML-Elemente können mit Hilfe der beiden Tags `<span>` und `<div>` zu Gruppen zusammengefasst werden. Der Unterschied zwischen diesen beiden Elementen liegt in Ihrer Darstellungsform.
Grundsätzlich haben diese beiden Elemente keinerlei Auswirkung auf das Design. Jedoch hat ein `<DIV>`-Container von Haus aus eine Breite von 100%. Ein `<span>`-Tag hingegen hat keine feste Breite, sondern gliedert sich in den fließenden Tag mit ein.

Ein `<span>` sollte also benutzt werden, wenn man nur einzelne Buchstaben oder Wörter innerhalb eines Textes markieren will; ein `<div>` hingegen, wenn man einen ganzen Block gruppieren und hervorheben will.

HTML Block Elemente

BLOCK Elemente werden normalerweise -- (kann mit CSS überschrieben werden) -- in einer neuen Linie dargestellt.
Sie haben grundsätzlich eine Breite von 100%.

Beispiele für Block Elemente:

- `<h1>`
- `<p>`
- `<ul>`
- `<table>`
- `<div>`

Inhalt des orange `<P>`

Listing 2.26: Beispiele für Block Elemente:

```
<style type="text/css">
    <!--
        .orange {
            background:orange;
        }
    -->
</style>



<p class="orange">Inhalt des orange <span class="tag">P</span></p>
```

HTML Inline Elemente

HTML Inline-Elemente werden im Gegensatz zu den Block-Elementen normalerweise innerhalb des normalen Textfluss dargestellt.

Beispiele für Inline Elemente:

- `<span>`
- `<img>`
- `<strong>`
- `<em>`
- `<a>`

Inhalt des orange `<SPAN>`

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean vitae eros eros, vel ultricies dolor. Aenean ultricies adipiscing viverra. Morbi in adipiscing quam. Proin neque lacus, lacinia placerat porta at, adipiscing ut neque. Nunc metus arcu, consectetur id lobortis in, lacinia nec augue. Ut vitae ipsum sit amet velit ornare dapibus. Curabitur eget libero vitae sem hendrerit porttitor id sed arcu. Sed condimentum ligula at nunc suscipit nec auctor tortor interdum. Proin non leo leo, sed consequat leo. Aliquam eu libero mi. Etiam non orci ac turpis porta laoreet. Ut sollicitudin porttitor mi, vel consectetur lorem lobortis ut. Nunc ipsum ipsum, faucibus eget rhoncus ac, faucibus eu magna.

Listing 2.27: Beispiele für Inline Elemente:

```
<style type="text/css">
    <!--
        .orange {
            background:orange;
        }
    -->
</style>


<span class="orange">Inhalt des orange SPAN</span>


<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean vitae eros eros, vel ultricies dolor.
Aenean ultricies adipiscing viverra. Morbi in adipiscing quam. Proin neque lacus, lacinia placerat porta at,
adipiscing ut neque. Nunc metus arcu, consectetur id lobortis in, <span class="orange">lacinia nec augue.</span>
Ut vitae ipsum sit amet velit ornare dapibus. <span class="orange">Curabitur</span> eget libero vitae sem hendrer
porttitor id sed arcu. Sed condimentum ligula at nunc suscipit nec auctor tortor interdum. Proin non leo leo, sed
consequat leo. Aliquam eu libero mi. Etiam non orci ac turpis porta laoreet. Ut sollicitudin porttitor mi, vel co
lorem lobortis ut. Nunc ipsum ipsum, faucibus eget rhoncus ac, faucibus eu magna.</p>
```

2.3.2 Semantische Strukturierung

Bekämpfung des Missbrauchs

Mit der Einführung von HTML5 wurden endlich Elemente geschaffen, um eine Seite semantisch korrekt zu strukturieren. In den Zeiten vor HTML5 wurden meist normale Container dazu missbraucht, mittels Ihrer ID zu beschreiben, was sie beinhalten. Durch die Analyse von Milliarden von Webseiten konnte ein eindeutiges Muster ermittelt werden, welche Bereiche auf einer Webseite zwingend notwendig sind. In fast allen Fällen gab es Container mit den folgenden IDs:

- #nav bzw. #navigation
- #header
- #footer
- #content
- #content_left bzw. #content_right

Da bei dieser empirischen Untersuchung derart viele Übereinstimmungen in der Struktur von Webseiten gefunden wurde, veranlasste dies die **WHATWG** (Community des W3C zur Weiterentwicklung von HTML) gewisse Elemente zu schaffen, die exakt nur diesem Zweck dienen. Durch die Nutzung dieser neuen Elemente wird versucht eine größere Vereinheitlichung zu schaffen und zusätzlich den Computern mehr Hilfe zu bieten, die Struktur einer Seite an Hand eindeutiger Objekte zu verstehen. Die wichtigsten neugeschaffenen Elemente sind:

- `<nav>`
- `<header>`
- `<footer>`
- `<article>`
- `<aside>`

Eine genauere Erklärung dieser Elemente erübrigt sich auf Grund der Eindeutigkeit der Namen.

Diese Elemente verhalten sich in der Handhabung mit **CSS** exakt wie ein Container. --> auch ältere Browser stellen diese Elemente richtig dar. (nämlich ohne optische Darstellung, so wie einen Container)

Exakte Definition der Elemente

WHATWG

Listing 2.28: Exakte Definition der Elemente

```html
<!DOCTYPE html>
<html>

<head></head>

<body>

<header></header>

<nav></nav>

<article></article>

<footer></footer>

</body>


</html>
```

3 CSS

CSS - Cascading Style Sheets

Die **Cascading Style Sheets**, kurz **CSS** genannt, sind eine deklarative Sprache für Stilvorlagen von strukurierten Elementen. Sie werden hauptsächlich in Verbindung mit HTML eingesetzt.

Der Grundgedanke ist, Inhalt und Aussehen voneinander zu trennen. Dafür gibt es mehrere Gründe. Einer der Gründe ist, dass nicht jeder Client (egal ob Mensch vor dem Computer oder reine Maschine) eine optische Aufbereitung von Informationen benötigt. Einer Maschine ist es egal, ob Text in gelb oder blau dargestellt wird. Eine Maschine liest nur das reine HTML und erkennt an Hand der korrekten Strukturierung, was eine Überschrift ist und was nur reiner Text. Unter anderem aus diesem Grund heraus, kam die Entscheidung, Inhalt und Design zu trennen. Somit müssen weniger Daten übertragen werden. In heutigen Zeiten von VDSL ist dies zwar irrelevant, jedoch gibt es noch weitere Gründe.

HTML bietet zwar unter anderem die Möglichkeit CSS direkt innerhalb des HTML Codes einzusetzen, jedoch müsste dieser bei jedem Element erneut geschrieben werden. D.h. bei einer Seite mit nur 5 Überschriften, muss 5 mal definiert werden, wie die Überschrift auszusehen hat. Es kann hierbei leicht einmal zu kleinen Tippfehlern kommen, geschweige denn von der Übersichtlichkeit des Codes. Die zweite Möglichkeit CSS zu benutzen, ist, die Stylesheets innerhalb einer HTML Datei im Kopf-Bereich zu definieren. Diese gelten dann für alle Elemente innerhalb dieser Datei. Doch eine normale Webseite besteht unglücklicherweise nicht nur aus einer Datei, sondern meist aus vielen Hundert.

Daraus folgt der logische Schritt, CSS über externe Dateien einzubinden. Es wird hierbei mindestens eine Datei erstellt, von welcher aus sich zentral das Design aller kompletten Einzelseiten steuern lässt. Die Wartbarkeit des Codes nimmt hierbei immens zu. Zusätzlich kann man auch problemlos durch Austauschen einer einzigen Datei, dem gesamten Webauftritt ein einheitlich neues Design verpasst werden.

CSS verleiht Webseiten ihr Aussehen - es hat nichts mit den eigentlichen Inhalten einer Webseite zu tun.

CSS - ZEN Garden

Der CSS - ZEN Garden, ist ein Projekt im Internet, dass sehr schnell zeigt, was nur mittels CSS alles erreicht werden kann. Die folgenden 3 Bilder unterscheiden sich lediglich in einem Punkt: dem CSS.....
Der HTML Code ist bei allen 3 Seiten identisch.

Weitere Beispiele für die Inspiration finden Sie unter
CSS - ZEN Garden

CSS Syntax

Um einem bestimmten HTML-Element CSS zuweisen zu können, haben Sie hier auch wieder mehrere Möglichkeiten. Die nachstehende Möglichkeit 1 weißt global jedem Tag mit dem Namen `<tag>`, der in einem Dokument steht seine Eigenschaften zu.

Die zweite Möglichkeit weißt seine Eigenschaften nun nur noch den Elementen zu, die explizit die Klasse "class" besitzen.

In der dritten Möglichkeit wird das CSS exakt dem einen Element mit der ID "id" zugewiesen.

CSS per Tag zuweisen

```
<tag>
```

CSS per Klasse zuweisen

```
<tag class="class">
```

CSS per ID zuweisen

```
<tag id="id">
```

Möglichkeiten der Einbindung von CSS

Nachdem wir nun gelernt haben, wie CSS Eigenschaften übergeben kann, stellt sich nun die Frage, wie schreibe ich diese Eigenschaften in mein HTML. Wie bereits erwähnt haben wir 3 verschiedene Möglichkeiten CSS einzubinden:

1. externe Dateien
2. Embedded CSS
3. Inline CSS

Anbei ein Beispiel einer HTML Datei, die alle 3 verschiedenen Varianten kombiniert.

Listing 3.1: Möglichkeiten der Einbindung von CSS

HTML

```html
<!DOCTYPE html>
<html>

<head>

    <link rel="stylesheet" type="text/css" href="style.css" />

    <style type="text/css">
        h1 {
            color:green;
        }
    </style>

</head>

<body>

    <h1>Grüne Überschrift</h1>

    <h1 style="color:yellow">Gelbe Überschrift</h1>

</body>

</html>
```

Hierarchie von CSS

Es wird stark angeraten, CSS nur mittels der externen Dateieinbindung zu nutzen. Lediglich für schnelle Testzwecke sollte inline CSS verwendet werden. Der Grund für diesen Rat, liegt in der Wartbarkeit und Entwicklung eines Web-Projekts. Falls innerhalb von CSS einem Element 2 mal das gleiche Attribut zugewiesen, jedoch mit verschiedenen values, nutzt der Browser immer das, was er als letztes gelesen hat. Da ein Browser eine HTML - Seite von oben nach unten liest, kommen wir zu folgender Hierarchie:

Inline-CSS > Embedded CSS > Externer CSS-Datei

Für den Fall, dass Sie nun die 3 verschiedenen Möglichkeiten der Einbindung genutzt haben: Sie schreiben in Ihrer externen CSS - Datei neue Werte und wundern sich warum nichts passiert? Wahrscheinlich haben Sie vergessen, dass Sie genau bei diesem Element auch Inline Code genutzt haben.

Betrachten Sie nachfolgenden Code und beantworten Sie folgende Fragen:

- In welcher Farbe wird die Überschrift H1 dargestellt?
- In welcher Farbe wird die Überschrift H2 dargestellt?
- In welcher Farbe wird die Überschrift H3 dargestellt?

Listing 3.2: Betrachten Sie nachfolgenden Code und beantworten Sie folgende Fragen:

HTML

```
<!-- DIESER BLOCK STELLT DIE EXTER EINGEBUNDENE DATEI DAR -->

    h1 {
        color:blue;
    }

    h2 {
        color:orange;
    }

    h3 {
        color:fuchsia;
    }

<!--        ENDE DER EXTERNEN EINGEBUNDENEN DATEI DAR       -->


<!DOCTYPE html>
<html>

<head>

    <link rel="stylesheet" type="text/css" href="style.css" />

    <style type="text/css">
        h1 {

        }
        h2 {
            color:green;
        }
        h3 {
            color:green;
        }
    </style>

</head>

<body>

    <h1>Überschrift 1</h1>

    <h2 style="color:yellow">Gelbe Überschrift</h2>

    <h2 style="color:green">Gelbe Überschrift</h2>

</body>

</html>
```

Maßangaben in CSS

Maß != Maß

Die wichtigsten Maßangaben in CSS

- em
- %
- px

Die beiden Maßangaben "px" für Pixel und "%" für Prozent wird jeder von Ihnen bereits schon kennen. Doch was hat es mit diesem "em" auf sich?

Bringen Sie in Erfahrung was die Maßeinheit **"em"** bedeutet und wieso sie speziell im Bereich Digital Publishing immer mehr verwendet wird.

3.1 CSS - Basics

CSS Syntax

Im nachfolgenden Beispiel sind ein paar verschiedene, einfache, CSS Definitionen aufgezeigt. Ab dem 4.ten Beispiel ("li a") wird deutlich, für was das **C** in CSS eigentlich steht - Cascading [Verschachtelung]. Der dort genutzte Selektor ist sehr üblich für CSS um nur bestimmte Elemente zu erwischen. Er wird von links nach rechts gelesen und gibt folgenden "Befehl" aus:

li a

Übersetzung

Suche nach allen `<li>` Elementen. Suche innerhalb von den gefunden Ergebnissen alle `<a>` und gib diesen meine nachstehenden Eigenschaften

Listing 3.3: li a

```
h1 {
    color: #333
}
/* ALLE h1 bekommen die Farbe #333          */



h1.orange {
    color:orange
}
/* alle h1, die auch die klasse orange habe */
/* <h1 class="orange"></h1>                 */


#content {
    border: 5px;
}
/* genau das Element mit der id content     */
/* <div id="content"></div>                 */


li a {
    color:#990000;
}
/* alle links, die innerhalb eines li stehen*/

li a.blue {
    color:blue
}
/* links, die innerhalb eines li stehen UND die Klasse blue haben */


```

Vergabe der Klassen in HTML

Wir haben nun gesehen, wie es möglich ist mit CSS exakt auf einzelne Elemente zuzugreifen. Dies geschieht durch die Vergabe von Klassen. Diese Klassen müssen jedoch erst in HTML vergeben werden, um Ihnen die Eigenschaften zuzuweisen. Das HTML Attribut "class" kann bei jedem Element eingefügt werden. Es hat innerhalb von HTML keine Auswirkungen. Es ermöglicht lediglich genau dieses Element mit CSS zu beeinflussen.

Das folgende Beispiel kombiniert schon mehrere Eigenschaften innerhalb einer CSS-Anweisung. CSS ist die Reihenfolge der Anweisungen dabei völlig egal.

Hier könnte auch Ihr Inhalt stehen

Listing 3.4: Vergabe der Klassen in HTML

```
<div class="beispielKlasse">Hier könnte auch Ihr Inhalt stehen</div>


.beispielKlasse {
        border:2px solid red;
        background: orange;
        padding:20px;
}
```

Aufgabe

Erstellen Sie einen CSS Selektor mit den folgenden Eigenschaften:

- border:5px solid green;
- font-size:17px;
- border: 3px;
- font-size:10px;
- border:yellow;

Überlegen Sie zuerst, was herauskommen sollte. Probieren Sie das Beispiel aus.

3.1.1 Schriften

Arial = Ariel ?

Schriften formatieren mit CSS

Für die Definition des Aussehens von Schriften bietet CSS viele verschiedene Möglichkeiten:

Eigenschaft			Beispiel
font-family	definiert die zu benutzende Schriftart	font-family: Arial, Helvetica, sans-serif;	
font-weight	definiert die Dicke der Schrift	font-weight: bold;	
font-style		font-style: italic;	
font-size	definiert die Schriftgröße	font-size: 15px;	

Mit diesen 4 Attributen können sie 99% aller Anforderungen an Schriften abdecken. Recherchieren Sie dennoch, ob es noch weitere gebräuchliche Attribute für Schriften gibt.

Beispiele

Eine Schrift in italic

Eine fettere Schrift

Eine größere Schrift

und schließlich noch eine andere Schriftart

Listing 3.5: Beispiele

```html
<span style="font-style:italic;">Eine Schrift in italic</span>
<span style="font-weight:bold;">Eine fettere Schrift</span>
<span style="font-size:3em;">Eine größere Schrift</span>
<span style="font-family:Times New Roman">und schließlich noch eine andere Schriftart</span>
```

Aufgaben

Erstellen Sie folgende Beispiele:

Beispiel 1 - der kleine dicke Italiener

- Schriftdicke: bold
- Schriftgröße: 9px
- Schriftstyle: kursiv
- Schriftart: Arial

Beispiel 2 - Die typische Überschrift

- Schriftdicke: bold
- Schriftgröße: 18px
- Schriftstyle: normal
- Schriftart: Times New Roman

Aufgabe 2

Erarbeiten Sie sich im Selbststudium, wie es möglich ist, eine Schrift unterstrichen darstellen zu lassen. Erweitern Sie das Beispiel 2 mit dieser neuen Eigenschaft.

Aufgabe 3

Es ist of üblich. dass Überschriften nur in Großbuchstaben dargestellt werden. Finden Sie heraus, ob es nur mittels CSS möglich ist, Schrift

- nur in Großbuchstaben,
- nur in Kleinbuchstaben,
- in Kapitälchen (jeder erste Buchstabe ist groß)

darstellen zu lassen.

Ausrichten von Schriften

Genau wie in Word ist es möglich, die Schrift links, rechts, zentriert oder auch passend ausgerichtet auszurichten.
Mit CSS setzen Sie dies mit der Eigenschaft "text-align" um.

- Erarbeiten Sie sich selbst, welche Werte das Attribut "text-align" annehmen kann.
- Erstellen Sie 4 verschiedene Absätze mit Beispieltexten und probieren die verschiedenen Werte aus

3.1.2 Farben

Farben in CSS

Sie haben in CSS mehrere verschiedenen Möglichkeiten Farben anzugeben. Die einfachste Angabe erfolgt über die englische Bezeichnung der jeweiligen Farbe. Das Problem welches hierbei jedoch sehr schnell Auftritt: Wieviele Farbnamen kennen Sie im Englischen und kennen Sie auch die jeweiligen Abstufungen?

Beispiel:

blau = blue

hellblau = lightblue

doch wie sieht es auch mit

aliceblue

oder

darkshadowblue

Finden Sie heraus wie die Bezeichnungen für 16 wichtigsten Farben lauten

Man sieht direkt, dass man sich hinter einigen Farbenbezeichnungen nur wenig vorstellen kann. Sie eignen sich somit nur für schnelle Tests, aber nicht für einen produktiven Einsatz.

CSS nutzt im Allgemeinen Hexadezimale Angaben für die Zahlen. Diese basieren auf dem RGB Farbmodell

Recherchieren Sie die Bedeutung der Kurzschreibweise von CSS - Farbcodes und erschließen Sie sich folgende Beispiele

- Was kommt heraus bei: #333
- Was kommt heraus bei: #F3A
- Was kommt heraus bei: #F3AB
- Was kommt heraus bei: #F
- Was kommt heraus bei: #FH3434
- Was kommt heraus bei: #blue
- Was kommt heraus bei: #mellongreen

Alternativ können die Farben auch dezimal angegeben werden. Dies wird praktisch jedoch nur sehr selten angewandt, da es einfach mehr zu tippen ist. Im folgenden sehen Sie ein paar Beispiele

Listing 3.6: Recherchieren Sie die Bedeutung der Kurzschreibweise von CSS - Farbcodes und erschließen Sie sich folgende Beispiele

```
color: teal;

color: #333333;
color: #9900FF;

color: rgb(255,0,0);
color: rgb(90%, 25%, 4%);
```

Was kann alles farbig werden?

Sie können mit CSS fast alle Attribute mit Farbe hinterlegen: Die für Sie gängstigen Attribute die sie mit Farbe hinterlegen werden:

- Standard element wie <p> oder <h> mittels "color:"
- Ränder entweder mit "border-color:" oder der Kurzschreibweise "border:"
- Hintergründe von Elementen mittels "background-color:" oder der Kurzschreibweise "background:"

Beispiele:

Listing 3.7: Beispiele:

```
div {
    background: #990033;
    border: #cccccc;
}

h1 {
    color:#456789;
    border-bottom: #789787;
}

span {
    border-color:orange;
    border: 1px 1px 3px 5px;
}
```

Das RGB Modell

Das RGB-Farbmodell gibt jede einzelne Farbe damit an, wieviel Prozent der 3 Grundfarben Rot, Grün oder Blau enthalten sein soll. Bei Angabe von jeweils 0% wird schwarz dargestellt. 100% jeder Farbe entspricht weiß.
Der Farbcode für CSS besteht in HTML meist aus 6 Zeichen. 2 für Rot, 2 für Grün und 2 für Blau. Die gängigen Browser unterstüzen auch die Kurzschreibweise, Farben mit 3 Zeichen anzugeben. Der Browser verdoppelt jedes Zeichen.

3.1.3 Ränder

Ränder um Elemente

Es kann jedem HTML Element ein Rand zugewiesen werden. In diesem Paragraph wird darauf eingegangen, wie man Ränder um Elemente setzen kann.
Für Ränder mittels CSS gibt es 2 verschiedene Schreibweise. Die Kurzschreibweise und die Langschreibweise.

das border attribut im detail

Beispiel:

Beispiel #1

Beispiel #2

Beispiel #3

Beispiel #4

Listing 3.8: Beispiel:

```
.beispiel1 {
    border:2px solid green;
}

.beispiel2 {
    border:4px dashed red;
}

.beispiel3 {
    border:1px dotted orange;
}

.beispiel4 {
    border-bottom:22px solid green;
    border-top:4px solid fuchsia;
    border-left:5px solid blue;
    border-right:4px solid orange;
}
```

Selbststudium

Aufgabe

Vergeben Sie an jeden Paragraphen, den Sie im Kapitel HTML erstellt haben einen Rand. Nutzen Sie die Beispiele aus dem Script.

3.1.4 Breite und Höhe

width und height

Mittes der Eigenschaft "width" ist es in CSS möglich, einem Element eine feste Breite zu geben. Als Anwendungsbeispiel kann man sich ein Bild, einen Absatz, einen Container, einen Link, eine Liste oder auch jedes andere Element vorstellen. Die Angabe der Breite sowie auch der Höhe ist eine der wichtigsten Eigenschaften von CSS.
Für die Angabe einer Breite haben Sie verschiedene Möglichkeiten der Notation:

- exakt: in **px**
- relativ: in **em**
- relativ: in **%**

Dies sind nur 3 von Notationen von vielen mehreren Möglichen. Doch diese 3 stellen die gängisten dar und sollten ausschließlich genutzt werden.

Die Breite und Bilder

Die meisten Bilder haben eine Relation der Auflösung von 4:3. Sie möchten nun ein etwas zu großes Bild auf Ihrer Seite einbinden. Das Bild hat eine Auflösung von 850px * 560px. Sie wollen es aber nur in 800px anzeigen, da es sonst nicht auf Ihre Seite passen würde. In diesem Fall wissen Sie, dass die neue Breite 800px ist. Doch ducrh welchen Wert muss die Angabe 560px ersetzt werden, ohne dass das Bild verzerrt dargestellt wird.

Aufgabe

1. Binden Sie ein beliebiges Bild auf einer Ihrer HTML - Seiten ein.
2. Geben Sie mittels CSS dem Bild eine Breite von 800px.
3. Überprüfen Sie Ihr Ergebnis. (Google Chrome mit Rechtsklick)
4. In welcher Breite wurde Ihr Bild ausgegeben? Ist dieser Wert der exakt richtig berechnete oder ist Ihr Bild verzerrt?

Bilder und deren Höhe

Diese mal sind die drei kleinen Hunde über Ihre Höhe und nicht über deren Breite definiert worden

Auftrag:

Testen Sie an Hand von Ihrem Beispiel für die Breite, wie sich die Bilder mit den Angaben von "height:" nun Verhalten.
Berechnet Ihr Browser wieder selbstständig die richtige Breite und Höhe?
Testen Sie Ihre Beispiele in mehreren verschiedenen Browsern.

Verzerrte Bilder

3.1.5 Hintergründe

Farbhintergründe

CSS erlaubt es mit dem Attribut "background" jedem Element einen Hintergrund zu geben.

background: #990045

Farbe

Bilder als Hintergründe

```
background: url('./img/wall.jpg') no-repeat;
```

url zum Bild

zusätzliche Eigenschaften

Auftrag:

- Recherchieren Sie, wie sie einen Hintergrund fest anbringen können. Es soll immer ganz oben im Browser angeheftet sein, auch wenn Sie weiter runter scrollen.
- welche Eigenschaften für die Wiederholung von Hintergründen gibt es?
- Ist es möglich, ein Hintergrundbild zu skalieren? Ihr Bild hat eine Breite von 2000px. Sie wollen es jedoch trotzdem als Hintergrund für einen 400px breiten Container nutzen. Sie wollen das Bild jedoch nicht abschneiden.
- Wie kann ich es schaffen, dass ein Bild erst ab einer bestimmten Stelle eingeblendet wird?

Hintergundbeispiele

Sie haben hier eine Sammlung an sehr schönen Hintergrundbildern für die eigene Nutzung.

Die angebotenen Hintergründe sind alle hochauflösend und sollten daher nur zum testen verwendet werden. Zur Nutzung auf Produktivsystemen sollten Sie runterskaliert / optimiert werden

Listing 3.9: Hintergundbeispiele

```
body {
    background: url('./img/1343311890.jpg') top center;
}
```

3.1.6 Links

Link Styling

Links stellen im Internet die einfachste Form der Interaktion dar. Durch Links wird erst die einfache Navigation im Internet ermöglicht. Es hat sich eingebürgert Links - also Interaktion für den User - speziell hervorzuheben. Der Nutzer soll direkt sehen: Hier geht es weiter, hier kann etwas passieren. Standardmäßig werden links deswegen schon in einer anderen Farbe und unterstrichen dargestellt.

Leider passt dieses blau jedoch keinesfalls zu meinem CI. Auch die bereits besuchten Links, die meist in lila dargestellt werden stören das Design meiner Seite erheblich. Nachdem wir mittlerweile Erfahrungen im Umgang mit CSS haben, fällt es uns sicher leicht, einen normalen Link umzugestalten.

Definieren Sie einige Links und geben diesen die folgenden Eigenschaften

- fett
- blauer Rand
- gelber Hintergrund
- feste Breite von 300px

Doch wie sieht es nun mit den bereits besuchten links aus? Oder dem Effekt, wenn Sie mit der Maus darüber fahren? Speziell für diese Anforderungen gibt es in CSS sogenannte Pseudo-Klassen.

Pseudo-Klassen sind Klassifizierungen, die einen Zustand oder eine Eigenschaft eines Elements beschreiben.

Pseudo-Klassen werden in CSS einfach direkt an den Selektor mit einem ":" angehängt.

Listing 3.10: Definieren Sie einige Links und geben diesen die folgenden Eigenschaften

```
a:hover {
    color:fuchsia;
}
/*färbt den Link beim Mouse-over (hover) lila*/
```

Weitere Pseudo-Klassen

:visited

definiert das Aussehen der bereits besuchten links [standard: lila]

:active

definiert was bei einem Klick auf das Element passieren soll

:focus

Definiert, wie ein focussiertes Element gehanhabt werden soll [speziell bei Formularen]

3.1.7 Außen und Innenabstände

Innenabstand von Elementen

Beispiele sagen oftmals mehr als tausend Worte:

padding:0px

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

padding:50px

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

padding:15px

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Wie die Beispiele verdeutlichen ermöglichen die Werte von padding, den Innenabstand von Elementen zu definieren. Es kann hierbei wieder die Kurz- und die Langschreibweise gewählt werden, um zum Beispiel links und rechts verschiedene Innenabstände zu definieren

Listing 3.11: padding:15px

```
div {
    padding: 15px;
}
/* alle 4 Seiten haben ein padding von 15px */

div {
    padding-top: 10px;
    padding-bottom: 20px;
}
/* Oben und Unten sind verschiedene Werte fuer padding definiert */
```

Außenabstand von Elementen

Die 3 folgenden bunten spans besitzen zunächsten noch die Standardwerte eines `<span>` (Inline - Elemente). Es sieht jedoch etwas komisch aus, dass diese 3 Elemente direkt nebeneinander kleben. Dem kann mit Hilfe der Eigenschaft "margin" Abhilfe geschafft werden.
Span 1 Span 2 Span 3

Den `<span>`-Elementen wird nun die Eigenschaft "margin:5px;" zugewiesen. Dieser Ausdruck definiert, dass jedes span zu seinen umgrenzenden Elementen einen Abstand von 5px einnehmen soll.
Span 1 Span 2 Span 3

Das Attribut "margin" definiert die Außenabstände von Elementen

3.1.8 das CSS Box Modell

Breite und Höhe von Elementen

Wir haben bereits gelernt wie man Innen- oder auch Außenabstände definieren kann. Doch wie wirken sich diese Pixel auf die Breite bzw. Höhe von Objekten aus?. Um dieses Verhalten zu testen, erstellen wir uns zunächste ein `<div>` und geben diesem eine feste Höhe und eine feste Breite. Wir Hinterlegen noch eine Hintergrundfarbe, so dass wir unser Ergebnis auch sehen können.

Listing 3.12: Breite und Höhe von Elementen

```
1 <div style="width:300px; height:300px; background:orange;"></div>
```

HTML

Prüfen der Umsetzung

Der Webbrowser Google Chrome hat bereits von Haus aus wunderbare Funktionen für Web-Developer mitgeliefert. Bei Nutzung des Browsers ist es möglich, sich mit einem Rechtsklick und der Auswahl "Inspect Element" sich die genauen umgesetzen CSS Angaben anzeigen zu lassen. Dies geschieht auch durch eine optische Hervorhebung, in der wir die genauen Angaben betrachten können.

Wir können im unteren Bild nun erkennen, dass unser erstelltes `<div>` exakt 300 * 300 Pixel einnimmt.

Die Breite + Innenabstand

Für die weiteren Test verpassen wir unserem `<div>` nun einen Innenabstand und einen Text, so dass wir den Abstand auch sehen können.

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet.

Prüfen Sie selbstständig mit Hilfe des Browsers Google Chrome, welche Breite das Element nun angenommen hat.

Listing 3.13: Die Breite + Innenabstand

```
<div style="width:300px; height:300px; background:orange; padding:20px;">Lorem ipsum dolor sit amet, consetetur
sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam
voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata
sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet.</div>
```

HTML

Die Breite + Innenabstand + Rand

In diesem Beispiel gehen wir nun noch weiter, und verpassen dem `<div>` nun auch noch einen einen Rand

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet.

Prüfen Sie auch für dieses Beispiel die ausgegebene Breite

Das box-model von CSS

Durch die soeben durchgeführten Tests, haben wir feststellen können, dass es doch etwas Kopfrechenleistung erfordert, eine Element trotz einer fest angegebenen Breite exakt zu definieren. Zu Grunde liegend für diese Berechnungen gibt es das so genannte **box-model** von CSS.

Die Grafik erläutert an Hand eines einfachen Beispiels, wieviel Platz Ihre jeweilige Box wirklich auf einer Seite einnehmen wird.

```
.box {
        width: 200px;
        padding:15px;
        border: 10px
}
```

Erläuterung für die CSS-Berechnungen:

1. Breite 200px

Durch die Angabe von "padding:15px" ist definiert, dass auf jeder Seite 15px padding eingerichtet werden sollen. Daraus folgt: das Element muss um 30px breiter werden (15px links + 15px rechts)

2. Breite 230px

Durch die Angabe von "border" erhalten wir erneut 20 px dazu (10px links + 10px rechts)

3. **Breite 250px**

Anwendungsbeispiel

Lage:

Sie arbeiten nun nicht mehr alleine an Ihrer Homepage, sondern haben ein Team von Redakteuren unterstellt. Diese Redakteure kennen sich leider nur ganz geringfügig mit CSS aus. Ihre Vorgaben für die gemeinsame Homepage besagen, dass keine Box breiter als 400px sein darf.
Ihre Redakteure bestehen darauf, dass Sie selber Innenabstände und auch Ränder anwenden dürfen.

Sie haben nun 2 Möglichkeiten:

1. Sie weisen jeden einzelnen Ihrer Redakteure in das soeben gelernte box-model von CSS ein und hoffen, dass jeder seine Boxen richtig berechnen kann.
2. Sie steigen etwas tiefer in die Nutzung von CSS ein und finden eine Lösung, das box-model umzudefinieren. Das Attribut "width" soll fortan nicht mehr nur die Breite des Inhalts begrenzen, sondern auch das "padding" und "border" mit einberechnen.

Hilfestellung:

- Recherchieren Sie die Eigenschaften für das attribut "box-sizing"
- Welche Eigenschaften kann "box-sizing" annehmen.
- Schaffen Sie es mit einer der Eigenschaften die oben genannte Lage zu lösen?

box-sizing: border-box

Die Lösung für die oben gegebene Lage ist die Eigenschaft "box-sizing:border:box". Mit dieser Eigenschaft wird nimmt ein Element bei Angabe einer Breite mittels "width" wirklich maximal diese Breite ein.

mit "box-sizing:border-box" können Sie eine maximale Breite inkl. des padding und der border angeben.

3.1.9 Einbinden von mehreren CSS Dateien

@import

Sie haben in CSS die Möglichkeit innerhalb einer CSS Datei, weitere CSS Dateien einzubinden. Es empfiehlt sich auch diese Möglichkeit zu nutzen. Sie können dadurch für die logischen Bereiche Ihrer Seite jeweils einzelne CSS-Dateien schreiben und können so leichter den Überblick behalten. Sie müssen ausserdem im HTML nur diese eine Datei angeben, und können diese immer wieder erweitern.

CSS bindet die Dateien in der Reihenfolge ein, wie Sie im CSS angegeben wurden. D.h. CSS Definitionen in den unteren Dateien können die Eigenschaften von den oberen Dateien überschreiben.

Erstellen Sie in einem Ordner "css" die folgenden CSS-Dateien:

1. style.css
2. layout.css
3. headings.css
4. lists.css
5. forms.css
6. content.css

Binden Sie in Ihrer HTML Datei nur die Datei style.css ein.
Importieren Sie innerhalb der Datei style.css alle anderen CSS-Dateien.

Listing 3.14: Erstellen Sie in einem Ordner css die folgenden CSS-Dateien:

```
/* Bindet eine CSS-Datei aus dem gleichen Ordner ein */

@import url("content.css");


/* Bindet eine CSS-Datei aus dem Ordner <span class="attribute">ordner</span> ein */

@import url("ordner/datei.css");

```

3.2 Layouting with CSS

Positionierung von Elementen

Wir haben im Kapitel HTML gelernt, das eine Webseite immer von oben nach unten aufgebaut und auch so vom Browser gelesen wird.
Wenn wir aber eine normale Webseite betrachten, stellen wir in den allermeisten Fällen fest, dass ein Navigationsmenue meist links oder auch rechts vom Content angesiedelt ist. Nachdem wir des Weiteren gelernt haben, diesen Aufbau **nicht** mit einer Tabelle zu generieren, kommen wir nun zu dem Thema das Layout mit CSS zu erstellen.

Aufbau einer Webseite

Ziel

Analyse

Bevor wir mit der eigentlichen Umsetzung des Layouting beginnen, analysieren wir zuerst, wo überhaupt unser Handlungsbedarf besteht. Unser angestrebtes Layout besteht aus 4 Containern. 2 Container davon (im Bild grün) dürfen ihre standardmäßige Eigenschaft - die Breite 100% - behalten. Wir haben somit nur bei den beiden noch blauen Containern etwas zu erledigen.

Die wohl einfachste Vorgehensweise ist es, die beiden Container erst mal grob in die gewünschte Form zu bekommen und danach zu positionieren. Dies bringt uns zu dem Punkt: Größe und Positionierung von Elementen.

Wie breit darf meine Webseite sein?

Wir machen einen kurzen Ausflug in die Vergangenheit:

In den ersten Zeiten des Internet gab es noch keine Handys; Laptops waren noch nicht weit verbreitet; von Tablets wagte man noch nicht einmal zu träumen. Von dem her war das einzige Medium, mit dem auf das Internet zugegriffen werden konnte, ein lokaler PC. 16:9 oder gar 20:9 Bildschirme waren damals auch noch völlig unvorstellbar. Es gab genau eine Art von Bildschirmen mit einem Verhältnis von 4:3. Die damals übliche Auflösung von Bildschirmen kann auf ca. 800*600px festgesetzt werden. Einzelne Monitore waren schon in der Lage 1024*768 px darzustellen - (auch kein großer Unterschied).
Ein Internetseite musste also lediglich auf 4:3 Bildschirmen angezeigt werden. Und Multitasking mit mehreren geöffneten Fenstern gleichzeitig musste auch noch nicht berücksichtigt werden. Der Inhalt wurde immer bildschirmfüllend angezeigt. Für den Ersteller einer Homepage war es demnach noch einfach... Inhalte erstellen und fertig.

Zurück in die Zukunft:

Wie sieht es mittlerweile aus?
Internetseiten werden aufgerufen mit winzigsten Bildschirmen von 420px * 320px Auflösung bis hin zu Monitoren die jenseits der 2560px * 1440px besitzen. Internetseiten werden aufgerufen von Handys, Tablets, Desktops, Laptops, etc.... Hier besteht nun die eigentliche Herausforderung für Sie als Designer.

> In den meisten Fällen erstellen Sie Internet-Seiten nicht primär für sich selbst, sondern wollen anderen Menschen Informationen bereit stellen. Schreiben Sie Ihre Homepage deswegen nicht so, dass Sie nur bei Ihnen gut aussieht, sondern so, dass möglichst viele Menschen mit möglichst vielen unterschiedlichen Geräten ihre Seite ohne Einschränkung nutzen können.

Dies ist leicht gesagt, aber stellt die eigentliche Herausforderung aus.
Eine Homepage, die auf Ihrem Rechner gut aussieht, schön lesbar ist und wunderbar funktioniert, kann bei dem nächsten Nutzer im schlechtesten Fall direkt zu Augenkrebs führen.

Beispiel

Sie besitzen einen Monitor mit einer Auflösung von 1024*768px. Die Inhalte ihrer Seite werden schön auf der gesamten Breite angezeigt und der Lesefluß ihrer langen Texte ist angenehm für Sie.
Ihr Hörsaalleiter besitzt einen Bildschirm mit einer Auflösung von 2560px in der Breite.... stellen Sie sich vor was passiert:

Der arme Mensch muss bei jedem ganzen Satz den er lesen will, seinen Kopf um mehrere Grad drehen, weil jeder Satz auf die gesamte Breite langgezogen wird. Es kann passieren, dass das Bild, das Sie beschreiben an der linken Seite des Bildschirms hängt, die Beschreibung jedoch ca. 30" Zoll entfernt auf der rechten Seite des Bildschirms auftaucht.

Responsive Design

Die Techniken zur Vermeidung solcher Probleme lassen sich unter dem Begriff responsive Design zusammen fassen. Sie sollen diesen Begriff nur einmal gehört haben. Wir werden darauf nicht viel weiter darauf eingehen, sondern holen uns nur den für uns wichtigsten und einfachsten Aspekt heraus: **Die Breite einer Webseite**

Die optimale Breite

Wir brauchen das Rad nicht neu zu erfinden. Es gab genug Institute, die erforscht haben, dass die optimale Breite einer Homepage derzeit 960px betragen soll. Daran werden wir uns mit möglichst wenigen Ausnahmen halten.

Fakten, die für 960px sprechen

- Bildschirme mit 1024px Breite (immer noch sehr verbreitet): Nach Abzug der Scrollbar rechts bleiben noch ca. 1000px übrig... 960 passt sehr gut
- 960 lässt sich durch extrem viele Zahlen teilen.

1 2 3 4 5 6 8 10 12 15 16 20 24 30 32 40 48 60 64 80 96 120

Praktisch um zu berechnen wie viele Bilder ich nebeneinander in welcher Größe unterbringen kann....
Wie breit mach ich eine Navigation...
...

- 960px lassen sich direkt ohne viel Bewegung der Augen auf einmal aufnehmen

Definieren der Breite

Zurück zu unserem Beispiel:

Nachdem wir uns nun auf eine Breite von 960px festegelegt haben, können wir nun unsere Navigation und unseren Content unterteilen.

```
nav + content = 960px
```

Für unser Beispiel wollen wir nun, dass unser Content genau 3 mal so breit ist, wie unsere Navigation. Dies führt zu:

```
nav = 3
content = 9
nav + content = 12
```

Wir teilen dementsprechen 960 / 12 = 80 Daraus folgt:

```
3* 80 = 240 = nav
9 * 80 = 720 = content
```

Wir haben also nun nicht einfach nur grob nach Gefühl, sondern exakt unsere beiden Werte gefunden. Diese setzen wir nun um und kommen zu folgendem Ergebnis:

article

720px

nav

240px

Nebeineinanderschachteln

Nachdem wir die Breiten nun definiert haben, fehlt uns nur noch die Eigenschaft, dass die beiden Elemente nebeneinander stehen. Erfahrungsgemäß stellt dies Kapitel die größte Herausforderung dar, und der Fehlerteufel kann sich leicht einschleichen.

Umsetzung der Überlegungen dieses Kapitels

Gesamtbreite 960px;

Als erstes bauen wir uns einen Container mit exakt 960px Breite. In diesen Container sollen dann alle Elemente rein. Der Vorteil, wenn wir alle Elemente in einem umschließenden Container erstellen, ist, dass wir dann nur an einer Stelle ansetzen müssen, wenn wir die gesamten Elemente dann später auf dem Bildschirm zentrieren möchten. Überlicherweise gibt man diesem Container eine ID. Mit dem Namen "wrapper".

Erstellen sie den Container mit der ID "wrapper". Verleihen Sie ihm eine Breite von 960px

Erstellen unserer Inhaltselemente

Erstellen Sie die 4 Elemente innerhalb unseres "div#wrapper":

- `<header>`
- `<nav>`
- `<article>`
- `<footer>`

- Vergeben Sie an Ihre 4 Elemente 4 verschiedene Farben.
- definieren Sie für die Veranschaulichung feste Höhen:
  - header:50px
  - nav:100px;
  - article:300px;
  - footer:50px;

Für eine Auswahl an Farben, die sehr gut zusammen passen, können Sie den ColorSchemeDesigner nutzen. Dies ist ein einfaches Tool, welches Ihnen direkt exakt zusammenpassende Farbcodes bereit stellt.
ColorSchemeDesigner

Umbrüche definieren

Definieren Sie nun die Umbrüche für "nav" und "article"
"float:left" und "float:right"

Betrachten Sie ihr Ergebnis! Wir haben wohl noch nicht so ganz erreicht, was wir erreichen wollten, oder?

Möchten Sie nach einem Element, dass einen "float" aufweist wieder das "normale" Verhalten, müssen sie dies durch die Eigenschaft "clear" für das nachfolgende Element ausdrücken.

Abschlussarbeiten

definieren sie das "clear" für den `<footer>`

Ergebnis

Ihr Ergebnis sollte nun in etwa so aussehen:

Listing 3.15: Ergebnis

```html
<!DOCTYPE html>
<html>

<head>
<style type="text/css">
    div.wrapper{
         width:400px;
         margin:10px auto;
    }
    header {
        background:#FF9900;
        height:50px;
    }
    nav {
        background:#FFB240;
        height:100px;
        width:100px;
        float:left;
    }
    article {
        background:#FFC773;
        height:300px;
        width:300px;
        float:right;
    }
    footer {
        clear:both;
        background:#BF8630;
        height:50px;
    }
</style>
</head>

<body>

<div class="wrapper">

    <header></header>

    <nav></nav>

    <article></article>

    <footer></footer>

</div>

</body>

</html>
```

3.2.1 Navigationsmenüs

Navigationsmenüs

Es ist üblich auf einer Seite jeweils ein horizontales und ein vertikales Navigationsmenü bereit zu stellen.

Aufgabe:

Untersuchen Sie die Seite Verwaltung der Bundeswehr auf Ihre Menüs.

Es gibt
- ein dunkelblaues horizontales Navigationsmenü (mittig über dem Inhalt),
- ein vertikales Navigationsmenü (links).

In welchem Zusammenhang stehen diese beiden Menüs? Wechselt das obere Menüs in Abhängigkeit von dem linken Menü?

Vergleichen Sie dazu auch die folgenden Seiten:

- Google Suche
- BMW M3 Coupe
- Arzneimittel.de

Lässt Sich auf diesen Seiten ein Zusammenhang erkennen?

Vertikales Navigationsmenü

Wir haben festgestellt, dass bei den meisten Internetseiten mit einem vertikalen Navigationsmenü gearbeitet wird. Wie wird solch ein Menü umgesetzt. Ein Vertikales ist im Grunde genommen eigentlich nichts anderes als eine ungeordnete Auflistung. Es liegt daher nahe und ist auch semantisch korrekt, dies mit einer Liste zu vergleichen und auf diese Art umzusetzen. Im Kapitel Inhalte in HTML haben wir bereits kennen gelernt, wie eine Liste erstellt wird. Wir kommen daher zu folgendem Code für HTML.

Listing 3.16: Vertikales Navigationsmenü

HTML

```html
<ul>
    <li><a href="Seite1.html">Seite 1</a></li>
    <li><a href="Seite2.html">Seite 2</a></li>
    <li><a href="Seite3.html">Seite 3</a></li>
    <li><a href="Seite4.html">Seite 4</a></li>
    <li><a href="Seite5.html">Seite 5</a></li>
</ul>
```

CSS für unser Menü

- Link1
- Link2
- Link3
- Link4
- Link5

Unsere soeben generierte Liste hat vom Aussehen, jedoch noch nicht viel mit dem Beispiel rechts gemeinsam. Es fehlen noch die Deklaration für das Aussehen.

Navigationsmenüs werden immer in einem `<nav>`-Container untergebracht. In den CSS-Deklaration wollen wir auch nur die Liste beeinflussen, die in einem `<nav>` steht. Diese Trennung wird durchgeführt, dass wir später noch die Möglichkeit haben, in unserem Inhaltsbereich noch normale Aufzählugslisten verwenden können, die nicht direkt wie ein Menü aussehen.

Wir kommen somit zu folgendem ersten CSS:

Listing 3.17: CSS für unser Menü

```
nav ul {

}

nav ul li {
}

nav ul li a {
}
```

Um unser angestrebtes Ergebnis zu erreichen, entfernen wir zunächste, die Aufzählungszeichen und geben unserer kompletten Liste einen Hintergrund sowie eine feste Breite

Die Angaben von "padding-left" bei `<ul>` entfernen uns die ungeswünschte Einrückung links. Die Angabe display:block bei unserem `<a>` bewirkt, dass unser Link in der gesamten Breite anklickbar ist, und nicht nur die Schrift.

Durch weitere kleine Anpassungen mit bereits bekannten Deklarationen erlangen wir schon fast unser Ergebnis.

Beachten Sie die Nutzung von "box-sizing"

Bis auf kleine farbliche Anpassungen haben wir es geschafft, mit wenig Code ein vertikales Navigationsmenü zu erstellen.

Aufgabe:

Versuchen Sie die Nutzung der CSS-Deklarationen in einem eigenen Beispiel nachzuvollziehen!
Erstellen Sie selbst ein Menü mit 5 Menüpunkten.

Horizontales Navigationsmenü

Eine horizontales Navigationsmenü stellt im grundelegenden Sinn auch nur eine Auflistung von Links dar --> Liste. Lediglich die Darstellung ist von links nach rechts. Unser horizontales Menü soll sich über unsere ganze Seite erstrecken. Wir vergeben daher die Eigenschaft "width:960px"

Wir möchten in unserem Menü 6 verschiedene Punkte bereitstellen. Errechnen Sie die richtige Breite für jeden einzelnen Link und definieren Sie diesen in einem neuem Beispiel.

Betrachten Sie ihr Ergebnis. Auf Grund welcher standardmäßig vorgebenen Eigenschaft werden Ihre Links untereinander angezeigt?

Welche Eigenschaft müssen Sie Ihren Links nun verleihen, so dass diese nebeneinander angezeigt werden?
Versuchen Sie, das folgende Menü selbstständig nachzubilden.

Link1	Link2	Link3	Link4	Link5	Link6

3.3 Advanced CSS Techniques

3.3.1 text-shadow

text-shadow

Die text-shadow - Eigenschaft wurde bereits in CSS 2.0 eingeführt. Jedoch wurde diese CSS-Eigenschaft von keinem Browser implementiert. In CSS 2.1 wurde die Textschatten-Eigenschaft entfernt und erst wieder in CSS3 in Kraft gesetzt. Daraufhin fingen fast alle Browser-Hersteller mit der Umsetzung der Text-Schatten an.

Listing 3.18: text-shadow

```
h1 {
    text-shadow: 1px 1px 5px #333;
}
```

Achtung

Im Gegensatz zu den Eingeschaften "padding" oder "margin" wird bei text-shadow an erstre Stelle der X-Versatz, also links / rechts angegeben. Bei "Margin" und "Padding" kommen in der Kurzschreibweise erst die Werte für Y

Beispiele

Selbststudium

Aufgabe

Versuchen Sie als Bild hinterlegten Beispiele nur mit CSS nachzubauen.

3.3.2 box-shadow

3.3.3 CSS - only Slider

CSS-Slider

Dieser Slider wurde nur mit reinem CSS und HTML geschrieben. Wenn Sie im Internet nach einem Slider suchen, werden Sie immer wieder die Begriffe Javascript oder jQuery lesen. Es war früher nötig, solch einen Slider mit den zusätzlichen Sprachen zu schreiben. Mittlerweile ist aber CSS so weit entwickelt, dass kein Javascript mehr benötigt wird.

Dieses Beispiel lässt sich im Script leider nicht betrachten, da es mit Animation funktioniert

3.3.4 Bouncing Balls with CSS

Bouncing Balls

3.3.5 border-radius

Listing 3.19:

```
<div style="width:200px;height:100px;background:orange;border-radius:50px;"></div>
```

PHP

4 Webseiten

Guideline für die Erstellung eines kompletten Internetauftritts

Lage:

Die Professur für Massivbau hat gehört, dass Sie den Lehrgang ITSysAdminFwWebSK besuchen und gibt Ihnen daraufhin den Auftrag für eines Ihrer Projekte einen Webauftritt zu gestalten. Das Projekt umfasst folgende Rahmenbedingungen.

> Das Layout der Seite wurde schon grob festgelegt und sollte in etwa auch so umgesetzt werden.
> Es sollen 8 verschiedene HTML-Seiten erstellt werden. Diese sollen alle das gleiche Menü auf der linken Seite erhalten. Die Inhalte werden Ihnen durch Ihren Auftraggeber bereit gestellt.
> Der Quellcode der Seiten soll gut strukturiert sein und es soll nachvollziehbar programmiert worden sein. Es ist angedacht, dass sich später eine neuer Webmaster findet, der Ihr Projekt dann übernehmen soll.

Durchführung

Nachdem die Rahmenbedingungen nun klar sind, geht es nun an die Umsetzung der Vorgaben.

Doch wie sollten Sie Vorgehen?
Die folgende Auflistung soll Ihnen einen einfachen Rahmen bieten, wie Sie die Umsetzung gut strukturiert anstellen können.

1
Erstellen der Grundstruktur
css
style.css
layout.css
headings.css
img
index.html

2
Erstellen des Layouts
<header>
<nav>
<article>
<footer>

3
Erstellen des Menü
Geschichte
Modell
Pläne
Fotos / Filme
Statik
Untersuchungen
Material

4
Kopieren der Seiten
css
style.css
layout.css
headings.css
img
index.html
geschichte.html
modell.html
plaene.html
fotosfilme.html
statik.html
untersuchungen.html
material.html

5
Erstellen der Inhalte
Bogenbrücke über die Pfreimd bei Kaltenthal
Geschichte
Bogenbrücke über die Pfreimd bei Kaltenthal
Gallerie
Bogenbrücke über die Pfreimd bei Kaltenthal
Modell
ungeordnete Liste
nummerierte Liste
Listenelement
1. erstes Element
2. zweites Element
3. drittes Element
4. viertes Element
5. fünftes Element

Erstellen der Grundstruktur

Wir empfehlen folgende Vorgehensweise:

Erstellen Sie sich ein erstes Grundgerüst mit den wichtigsten Dateien:

- Legen Sie sich in einem neuen Ordner ihre Unterordner an:
  - css
  - img
- Erstellen Sie sich ihre ersten leeren Dateien (CSS - Dateien mit klarer Struktur)
- Binden Sie ihre CSS - Dateien in ihre erste HTML Datei ein.

Erstellen des Grundlayouts

Erstellen Sie sich, wie Sie es im Kapitel "Layouting mit CSS" gelernt haben, das Grundlayout ihrer Seite

Erstellen des Menues

Erstellen Sie sich auf Ihrer ersten Seite ihr Menü

Fertigstellung des Layouts

Nachdem Sie nun die Inhalte und die Struktur, die auf jeder Seite gleich sein werden festgelegt haben, können Sie sich nun um das Design der Seite kümmern. Stellen Sie das Design für Ihre erste HTML - Seite komplett fertig.

Kopieren Ihrer ersten HTML-Seite

Kopieren Sie ihre erste nun fertige HTML so oft, wie Sie verschiedene Seiten brauchen werden.

Erstellen der Inhalte

In dem nun fast letzten Schritt, müssen Sie nun noch lediglich Ihre Inhalte auf den Seiten einpflegen.

4.1 Dynamische Webseiten

Autmatische Erstellung der Menüs mit PHP

Wie sie nun sicher festgestellt haben, kann es bei einer größeren Seite sehr sehr aufwändig werden, in einem Menü einen Punkt zu generieren. Das neue Menüelement, dass Sie erstellen möchten, müssen Sie nun langwierig auf jeder Seite einzeln reinkopieren. Es wäre doch eine schöne Möglichkeit, wenn Sie ähnlich wie in CSS, nur eine Datei haben, und diese wird überall automatisch eingebunden. Hier kommt nun PHP ins Spiel.

PHP ist eine Programmiersprache um dynamische Webseiten zu erstellen. **PHP** steht für PHP Hypertext Preprocessor.

Wir werden in diesem Kapitel nicht PHP lernen, sondern uns nur ein bis 2 Funktionen ansehen, die uns das Leben erheblich erleichtern könnnen. Auf Ihren virtuellen Maschinen ist PHP bereits fertig installiert und konfiguriert. Wie man PHP konfiguriert, lernen wir im Kapitel WEBSERVER.

Vorbereitungen

Damit Sie PHP-Code auf Ihren Webseiten nutzen können, müssen Sie lediglich den Dateinamen Ihrer HTML-Seiten in ".php" umbenennen. Der HTML Code wird dadurch nicht beeinflusst. Aber Sie haben ab sofort die Möglichkeit PHP Code einzubetten.

Aufgabe:

Benennen Sie eine HTML-Seite ihrer Wahl in ".php" um. Betrachten Sie ihr Ergebnis.

Ein erstes Snippet

Damit ihr PHP-Code auf Ihrer Hompeage ausgeführt wird, müssen Sie das nur dem Browser mitteilen, indem Sie den auszuführenden Code innerhalb des Tags `<?php ?>` schreiben.

Binden Sie auf einer Ihrer Seiten das folgende Snippet ein und testen die Ausgabe.

Dieses Beispiel funktioniert nicht lokal. Die Vorraussetzung ist ein laufender Webserver mit PHP

Listing 4.1: Ein erstes Snippet

```php
<?php
    echo "Hallo Welt";
?>
```

Das automatische Menü

Betrachten Sie die untenstehende Grafik:
Unser Ziel ist es, dass die eine Datei "menu.html" aus dem Ordner menu auf allen Seiten automatisch eingebunden wird. Wir möchten unser Menu ab sofort nur noch in dieser einen Datei ändern und es soll sich auf allen Seiten genau gleich verhalten.

PHP include

Mit der PHP Funktion "include" ist es möglich den gesamten Inhalt einer Datei einzubetten.

Aufgabe:

1. Erstellen Sie auf Ihrem Webserver den Ordner "menu"
2. Erstellen Sie in Ihrem neuen Ordner die Datei "menu.html" und fügen einen beispielhaften Inhalt wie "Hallo Welt" ein.

3. Benennen Sie jede Datei, in die Sie "menu.html" einbinden möchten von ".html" in ".php" um.
4. Kopieren Sie das untenstehende Snippet in jede ".php" - Datei an die Stelle, an der Sie Ihr Menu später haben möchten.

Prüfung

Überprüfen Sie Ihr Ergebnis. Rufen Sie Ihre Seiten im Browser auf. Ändern Sie nun den Text "Hallo Welt" in etwas anderes und Prüfen Sie, ob sich jede einzelne Seite automatisch geändert hat.

Listing 4.2: Aufgabe:

```php
<?php include('menu/menu.html'); ?>
```

Aufgabe

Sie haben nun gelernt, wie Sie eine Webseite mit etwas Dynamik ausstatten können.

Aufgabe:

Erweitern Sie Ihre Beispiel-Seiten so, dass Ihr komplettes Menu auf allen Seiten ausgegeben wird. Ersetzen Sie dafür den einfachen String "Hallo Welt" mit Ihrer kompletten ungeordneten Liste.

Achten Sie darauf, dass Ihre Dateien nun nicht mehr ".html" heißen, sondern ".php". Das "href" Attribut muss nun auf die ".php" Dateien verweisen.

5 WEBSERVER

5.1 HTTP und das Web

5.1.1 Webserver und Protokolle

Was ist ein Webserver?

Betrachtet man nur die physikalischen Komponenten des Internets, dann ist dieses Netzwerk eine gigantische Ansammlung von aktiven Komponenten (Switches, Router, usw.) und Computern, die verschiedenste Dienste in diesem Netzwerk zur Verfügung stellen (Webserver). Was aber ist ein Webserver genau?

- Was ist die Hauptaufgabe eines Webservers?
- Nennen Sie drei Protokolle, die häufig zur Kommunikation mit einem Webserver benutzt werden!
- Auf welcher Schicht im ISO/OSI-Modell ist ein Webserver einzuordnen? Begründen Sie Ihre Antwort!
- Wo, außer im Internet, kann man Webserver finden? Nennen Sie mindestens drei Möglichkeiten!

Das Hyper Text Transfer Protocol

Um Nachrichten, Webseiten oder Binärdateien mit anderen Webservern oder Browsern austauschen zu können, benötigen Webserver Protokolle, wie z. B. das "Hyper Text Transfer Protocol", kurz **HTTP**.

Benutzen Sie Wikipedia, um die folgenden Fragen zu beantworten:

- Was ist ein Protokoll (Artikel "Kommunikationsprotokoll")?
- Welche Hauptaufgabe hat das Prokoll HTTP (Artikel "HTTP")?
- Zu welcher Protokollfamilie gehört HTTP?
- Auf welchen Schichten des ISO/OSI-Modells ist HTTP beheimatet?
- Wie werden die Kommunikationseinheiten in HTTP bezeichet?
- Wie viele unterschiedliche Arten solcher Kommunikationseinheiten gibt es in HTTP? Nennen Sie auch deren Namen!
- Was ist ein "HTTP-Header", und was ist ein "Message-Body"? Welche Intformationen enthalten HTTP-Header bzw. Message-Body?
- Was bedeuten die Angaben "HTTP-Get" und "HTTP-Post"? Worin unterscheiden sich diese beiden Dinge?
- Worin liegt der Unterschied zwischen den Versionen "HTTP 1.0" und "HTTP 1.1"

URL/URI und virtueller Dateipfad

Ein **URI** ("Uniform Resource Indicator/Identifier") bezeichnet die vollständige Adresse, um beliebige Daten auf einem Server im Internet zu finden. Ein Beispiel für einen URI könnte sein:

http://www.fueustgsbw.de:80/intern/service/lehre_und_ausbildung/index.html

Benutzen Sie zur Beantwortung der folgenden Fragen den deutschen Wikipedia-Artikel "URI"!

- Wie heißen die Teile, in die sich der obige URI gliedert (Hinweis: Es sind fünf Teile)?
- Worin liegt der Unterschied zwischen einem URI und einem URL?

5.2 Apache

5.2.1 Einfuehrung und Installation

Einführung

Der Apache-Server ist eines der erfolgreichsten freien Software-Projekte in der Geschichte. Man geht derzeit davon aus, dass 54 % aller Websites von einem Apache auf verschiedensten Plattformen ausgeliefert werden. Dabei ist der Apache im Betrieb sehr zuverlässig, performant und durch seine modulare Architektur leicht erweiterbar.

Durch eine große Anzahl von Zusatzmodulen wird der Apache zur geeigneten Plattform für dynamische Inhalte, für sichere Transaktionen oder Workflow-Management. Die Flexibilität im Aufbau und die Vielzahl der unterschiedlichen Erweiterungen bringen eine hohe Komplexität bezüglich der Konfigurationsdateien mit sich, weshalb hier, in dieser Unterrichtsunterlage, nur ein Teil seiner Features und Module beleuchtet werden kann.

Es gibt verschiedene Versionen des Apache. Am häufigsten werden die Versionen 1.3, 2.0 sowie 2.2 genutzt, wobei zwischen den Versionen 1.3 und 2.0 ein großer Qualitätssprung stattfand. Bei einer Neuinstallation sollte darauf geachtet werden, immer die aktuellste Version, mit ihren aktuellsten Patches, zu nutzen, damit Sicherheitslüucken so schnell wie möglich geschlossen werden. Die derzeit aktuellste Version des Apache ist 2.4.2.

Recherchieren Sie im Internet, wo Sie...

- die aktuelle Online-Dokumentation zum Apache HTTP-Server finden.
- das aktuelle Installationspakete für Windows (inklusive OpenSSL) finden.

Speichern Sie sich diese Links als Bookmarks auf Ihrem PC!

Installation

Die Echtheit des Installationspaketes überprüfen

Laden Sie die aktuellste Version (2.4.x) des Apache HTTP-Servers für Windows, und die dazugehörige SHA1-Datei herunter! (Download here)

Mit Hilfe der SHA1-Checksumme einer Datei kann deren Echtheit verifziert werden. Für das Installationspaket wird die Checksumme "64D78A9C90E005E8F4F55F4E1C3720E856BBC005" bereitgestellt. Um eine Überprüfung dieses Pakets durchführen zu können, wird zusätzlich noch eine Software benötigt, die SHA1-Checksummen generieren kann. Eine solche Software ist z. B. "digestIT2004".

Laden Sie die Software "digestIT2004" aus dem Internet herunter (Download here), und installieren Sie sie (Complete Installation). Führen Sie jetzt eine Verifikation des Apache-Installationspaketes, gemäß der folgenden Anleitung durch!

1. Markieren und Kopieren Sie sich die SHA1-Checksumme (64D78A9C90E005E8F4F55F4E1C3720E856BBC005).
2. Klicken Sie mit der rechten Maustaste auf das Installationspaket. Es erscheint ein Kontextmenü.
3. Klicken Sie auf den Menüeintrag "digestIT 2004" mit der linken Maustaste.
4. Wählen Sie den Menüpunkt "Verify SHA1 Hash"!

Wenn alles ordnungsgemäß verlaufen ist, sollte die Meldung "Digest matches. Verification succeeded." erscheinen, und die Verifikation ist erfolgreich abgeschlossen.

Die Software installieren

Installieren Sie, nach den folgenden Vorgaben, den Apache HTTP-Server auf Ihrem Laptop!

- Network Domain: apache.org
- Server Name: localhost
- Administrator's Email Address: Admin@localhost.apache.org
- Install for all users, on port 80, as a service -- Recommended.
- Setup Type: Typical
- Installationspfad: DATENPARTITIONapachehttpd2.4.2

Ist die Installation erfolgreich verlaufen, erscheint am unteren rechten Bildschirmrand ein kleines Symbol, eine Feder mit einem grünem Pfeil.

Die Steuerkonsole (Apache Service Monitor)

Mit einem Doppelklick können Sie den Apache Service Monitor öffnen.

Welche Möglichkeiten/Funktionen bietet der Apache Service Monitor?

Der Apache in der Windows Dienstekonsole

Wenn der Apache HTTP-Server als Dienst installiert wird, ist er selbstverständlich auch in der Windows Dienstekonsole zu finden.

Unter welchem Namen ist der HTTP-Server in der Windows Dienstekonsole zu finden?

Nach einem Doppelklick auf den HTTP-Server Dienst öffnet sich dessen Eigenschaftenfenster.

Achten Sie an dieser Stelle darauf, dass der Apache automatisch, beim Systemstart, mitgestartet wird.

Das Installationsverzeichnis

Nach der Installation des HTTP-Servers finden sich in dessen Installationsverzeichnis diverse Unterverzeichnisse. Diese sind:

Welche Bedeutung haben die unten aufgeführten Unterverzeichnisse?

bin	

cgi-bin	
conf	
error	
logs	
modules	
htdocs	

5.2.2 Konfiguration

Der erste Blick in die Konfigurationsdatei

Der Apache HTTP-Server wird nicht wie viele andere Windowsprogramme mittels der Windows Registry, sondern mit Hilfe einer eigenen Textdatei konfiguriert. Üblicher weise heißt diese Datei httpd.conf. Sie kann jedoch auch einen beliebigen anderen Namen tragen und durch zusätzliche Dateien erweitert werden. Zufinden ist diese Datei im Unterverzeichnis "conf" des Installationsverzeichnisses.

Fertigen Sie jetzt eine Sicherheitskopie der Datei httpd.conf an!

Direktiven

Zur Konfiguration des Apache werden sogenannte "Direktiven" in die Datei httpd.conf eingetragen. Eine solche Direktive ist wie folgt aufgebaut:

Direktive	Argument
ServerRoot	E:/apache/httpd/2.4.2

Für die Erstellung von Direktiven gelten die folgenden Regeln:

- Nur eine Direktive pro Zeile
- Die Direktiven sind nicht Casesensitiv,d. h. ServerRoot = serverroot.
- Die Argumente der Direktiven sind Casesensitiv.
- Einrückungen können verwendet werden, um die Lesbarkeit der Konfigurationsdatei zu verbessern.
- Der Backslash () kann verwendet werden, um sehr lange Zeilen in mehrere Zeilen zu verteilen (Linebreak char).
- Bei der Angabe von Datei- und Verzeichnispfaden wird immer der Slash (/) als Trennzeichenverwendet, nicht der Backslash ().

Kommentare

Recherchieren Sie in der Onlinedokumentation des Apache HTTP-Server, wie Kommentare in die Konfigurationsdatei eingetragen werden, bzw. welche Regeln dafür gelten!

Erstellen einer ersten Konfiguration

Noch startet der Server nicht

Die Mindestkonfiguration eines Apache HTTP-Servers besteht aus der `Listen`-Direktive.

- Finden Sie heraus, welche Bedeutung und welche Syntax die `Listen`-Direktive hat!
- Legen Sie eine neue Konfigurationsdatei, unter dem Namen httpd.conf an! Der HTTP-Server soll ausschließlich auf Port 80 lauschen!
- Testen Sie, ob Ihr Apache HTTP-Server startet!
- Welche Fehlermeldung erhalten Sie von Ihrem Browser, wenn Sie die URL http://localhost eingeben?

Aktuell ist der Apache noch nicht in der Lage, Webseiten auszuliefern. Dies liegt daran, dass der Server keinerlei Information darüber hat, in welchem Verzeichnis die Dateien liegen, mit denen er arbeiten soll. Die betreffende Information erhält der HTTP-Server mit der `DocumentRoot`-Direktive. Sie nimmt eine Pfadangabe als Argument entgegen.

Standardmäßig ist das Unterverzeichnis htdocs im Apache Installationsverzeichnis als `DocumentRoot` vorgesehen.

- Fügen Sie die `DocumentRoot`-Direktive in Ihre Konfigurationsdatei ein, und verweisen Sie mit ihr auf das htdocs-Verzeichnis (z. B. E:/apache/httpd/2.4.2/htdocs)!
- Testen Sie erneut den Zugriff auf http://localhost mit Hilfe Ihres Browsers! Welche Antwort erhalten Sie jetzt?
- Versuchen Sie jetzt, mit Hilfe des Internet Explorers, auf die Adresse http://localhost/index.html zuzugreifen! Welche Antwort erhalten Sie von Ihrem Browser?
- Versuchen Sie erneut auf die Adresse http://localhost/index.html zuzugreifen, dieses mal aber mit Hilfe des FireFox! Wie unterscheiden sich die Antworten des FireFox und des Internet Explorers von einander?

Weitere wichtige Direktiven

Mit den beiden Direktiven `Listen` und `DocumentRoot` hat der Apache HTTP-Server bereits alle Informationen, die er für einen rudimentären Betrieb benötigt. Für den produktiven Betrieb eines Apache HTTP-Servers genügen diese Angaben jedoch noch nicht. Es existieren weitere Direktiven, die zur Mindestkonfiguration dieses Webservers gehören. Die folgende Auflistung zeigt einige davon.

Recherchieren Sie die Bedeutung der Direktiven in der obigen Auflistung!

Direktive	Bedeutung
`DefaultType`	
`ErrorLog`	
`LogLevel`	
`ServerAdmin`	
`ServerName`	
`ServerRoot`	

Einbinden von Modulen

Wie bereits zu Anfang erwähnt, ist der Apache HTTP-Server ein modular aufgebautes Produkt. Im Unterverzeichniss modules , des Installationsverzeichnisses, liegen mehrere Dateien, mit der Dateiendung .so. Hierbei handelt es sich um sogenannte "Shared Objects", oder auch um "Dynamic Link Libraries", die in der Programmiersprache C erstellt wurden. Jede dieser Dateien stellt eine Vielzahl an Funktionen bereit, die der Apache nutzen kann, um seine eigenen Fähigkeiten zu erweitern.

- Recherchieren Sie in der Onlinedokumentation (Dynamic Shared Objects), wie das Laden von Modulen funktioniert (Syntax)!
- Recherchieren Sie in der Dokumentation was die Direktive `DirectoryIndex index.html` bedeutet!
- Welches Modul muss geladen werden, damit die `DirectoryIndex`-Direktive funktioniert?
- Laden Sie das betreffende Modul, tragen Sie diese Direktive in Ihre Datei httpd.conf ein und prüfen Sie was passiert, wenn Sie die URL http://localhost, nach einem Webserverneustart, aufrufen!
- Was passiert, wenn Sie die folgende URL in Ihrem Browser aufrufen: http://localhost/login.php?
- Lesen Sie in der Dokumentation über die Direktive `FallbackResource` nach, und benutzten Sie diese, um im Falle eines HTTP-404-Fehlers auf eine Fehlerseite zu verweisen!
- Worin besteht der wesentliche Unterschied zwischen der `FallbackResource`-Direktive und der Direktive `ErrorDocument`?
- Konfigurieren Sie Ihren Apache jetzt so, dass er statt der `FallbackResource`-Direktive die `ErrorDocument`-Direktive nutzt, um im Falle eines HTTP-404-Fehlers die Fehlerseite zu zeigen!

Bedingtes laden von Modulen

In manchen Umgebungen kann es notwendig sein, dass ein Apache-Modul nur dann geladen werden darf, wenn eine bestimmte Bedingung erfüllt ist. Um dies zu erreichen, muss der `IfModule`-Kontainer eingesetzt werden.

Ein Kontainer besteht aus einem öffnenden Tag (z. B. `<IfModule [Bedingung]>` und einem schließenden Tag (`</IfModule>`). Beide Tags umschließen Direktiven, die nur dann ausgeführt werden, wenn der beteffende Kontainer bei einer Anfrage angesprochen wird.

Im konkreten Falle des `<IfModule>`-Kontainers heißt das, dass die in einem solchen Kontainer eingeschlossenen Direktiven nur dann ausgeführt werden, wenn [Bedingung] zutrifft. Das Beispiel auf dieser Seite zeigt den `<IfModule>`-Kontainer, zusammen mit der `DirectoryIndex`-Direktive. Diese wird nur dann ausgeführt, wenn das Modul `dir_module` geladen wurde.

Listing 5.1: Einbinden von Modulen

```
<IfModule dir_module>
  DirectoryIndex index.html
</IfModule>
```

5.2.3 Sicherheit

Directory-Kontainer

`<Directory>`-Kontainer stellen einen zentralen Sicherheitsmechanismus innerhalb der Konfiguration des Apache HTTP-Servers dar. Mit ihrer Hilfe kann der Zugriff auf Verzeichnisse, die über den Webservers erfolgen, reguliert werden. Wie bereits beim `<IfModule>`-Kontainer besprochen, gelten auch hier die Anweisungen innerhalb des Kontainers nur für das angegebene Verzeichnis.

- Erläutern Sie die Bedeutung der Anweisungen im `<Directory>`-Kontainer aus dem unten angegebenen Beispiel.
- Welches Modul muss geladen sein, damit die Direktive `Require` zur Verfügung steht?
- Laden Sie SELFHTML aus dem Internet herunter, falls dies noch nicht geschehen ist, und entpacken Sie das ZIP-File in das Verzeichnis htdocs/selfhtml.
- Erstellen Sie in der Datei httpd.conf einen `<Directory>`-Kontainer, der den Zugriff auf das Verzeichnis htdocs/selfhtml von allen jedem Rechener aus und ohne jede weitere Bedingung erlaubt.

Listing 5.2: Directory-Kontainer

```
<Directory />
    AllowOverride none
    Require all denied
</Directory>

```

Externe Konfiguration der Verzeichnissicherheit

In machen Situationen kann es notwendig sein, dass die Konfiguration der Zugriffsrechte eines Verzeichnisses nicht durch den Administrator des Webservers, sondern durch eine andere Person erfolgen muss. In so einem Fall gibt es zwei Optionen:

- Der Serveradministrator gibt der betreffenden Person Zugriff auf die Hauptkonfiguration des Webservers (httpd.conf).
- Es werden .htaccess-Dateien genutzt (man beachte den Punkt vor dem Namen).

Im Allgemeinen dürfte die zweite Option, die Nutzung der .htaccess-Dateien, die sinnvollere Alternative sein.

Unter MS-Windows ist es nicht möglich, dass ein Dateiname mit einem Punkt beginnt. Deshalb muss hier die `AccessFile`-Direktive in der Hauptkonfigurationsdatei genutzt werden, um den Namen der .htaccess-Dateien zu ändern, z. B. in htaccess (ohne Punkt)!

Benutzung von .htaccess-Dateien

.htaccess-Dateien ermöglichen es, denn Zugriff auf die Verzeichnisstruktur des Webservers auf der Basis einzelner Verzeichnisse zu regeln. Dies geschieht, in dem .htaccess-Dateien in die Verzeichnisse gelegt werden, für die Zugriffsbeschränkungen eingerichtet werden müssen.

Die Direktiven einer .htaccess-Datei gelten immer für das Verzeichnis, in dem sie sich befindet, und alle darunter liegenden Unterverzeichnisse.

.htaccess-Dateien nutzen die gleiche Syntax, wie die Hauptkonfigurationsdatei httpd.conf. So kann z. B. der `<Directory>`-Kontainer für das Verzeichnis htdocs/selfhtml durch eine entsprechende .htaccess-Datei ersetzt werden.

Innerhalb der .htaccess-Datei entfällt die Angabe des `<Directory>`-Kontainers, da der gesamte Inhalt der Datei nur für ein Verzeichnis gültig ist.

Testen Sie, ob die auf dieser Seite gezeigte .htaccess-Datei bereits Auswirkungen hat! Wenn nicht, was muss an der Hauptkonfigurationsdatei geän dert werden, um die .htaccess-Datei wirksam werden zu lassen?

Negative Effekte

Bei der Nutzung von .htaccess-Dateien treten im Wesentlichen drei negative Effekte auf:

1. Sobald mit Hilfe der `AllowOverride`-Direktive die Nutzung von .htaccess-Dateien erlaubt wird, sucht der Apache automatisch, in allen Unterverzeichnissen immer nach einer .htaccess-Datei, auch dann, wenn gar keine vorhanden ist.
2. Eine .htaccess-Datei wird bei jedem Verzeichnisszugriff erneut geladen, um etwaige Änderungen an der Verzeichniskonfiguration feststellen zu können.
3. Da die Direktiven aus den .htaccess-Dateien immer kumulativ betrachtet werden, müssen alle .htaccess-Dateien, auch die aus höheren Ebenen gelesen werden, um die Direktiven addieren zu können, und somit die effektive Konfiguration bilden zu können.

Direktiven aus verschiedenen .htaccess-Dateien werden immer in der Reihenfolge angewandt, in der sie vom Webserver vorgefunden werden. D. h. Direktiven aus .htaccess-Dateien in den unteren Verzeichnissebenen können Direktiven aus .htaccess-Dateien den höheren Verzeichnissebenen überschreiben.

Ein Beispiel zum vorangegangenen Merksatz:

Es existieren die folgenden Verzeichnisse: htdocs/selfhtml und htdocs/selfhtml/css. Beide Verzeichnisse haben eine eigene .htaccess-Datei. In der .htaccess-Datei des Verzeichnisses htdocs/selfhtml ist die Direktive `Require all granted` angegeben. Das bedeutet, dass ein Nutzer Zugriff auf beide Verzeichnisse hat, da die Direktiven aus .htaccess-Dateien auch immer für alle Unterverzeichnisse gelten.

Im Verzeichniss htdocs/selfhtml/css ist jedoch in der .htaccess-Datei die Direktive `Require all granted` gesetzt. Daraus folgt, dass der Zugriff auf das Verzeichnis htdocs/selfhtml/css für alle Nutzer gesperrt ist, weil die `Require all granted`-Direktive als zweites gefunden wird, und somit Gültigkeit hat. Nutzer können also nur auf das Verzeichnis htdocs/selfhtml zugreifen.

1. Wie verhält es sich zwischen einer ..htaccess-Datei und der Hauptkonfigurationsdatei? Kann eine .htaccess-Datei Direktiven aus der Hauptkonfigurationsdatei überschreiben, und umgekehrt?
2. Was passiert, wenn ein Benutzer, in seinem Browser, die folgende URL aufruft: http://localhost/selfhtml/htaccess oder http://localhost/selfhtml/.htaccess?

3. Wie kann verhindert werden, dass sich ein Nutzer den Inhalt einer .htaccess-Datei anzeigen lässt (Hinweis: Schlagen Sie hierfür die `FilesMatch`-Direktive nach!)

Listing 5.3: Externe Konfiguration der Verzeichnissicherheit

```
# httpd.conf
<Directory "htdocs/selfhtml">
  Require all granted
</Directory>

#.htaccess-Datei
  Require all granted
```

Zugriffskontrolle durch Autorisierung

Autorisierung mit Hilfe von Ausdrücken

Das bereits bekannte Kern-Autorisierungsmodul `authz_core_module` stellt den Grundbaustein für alle Autorisierungsmechanismen des Apache HTTP-Server dar. Neben den bereits bekannten Optionen `Require all granted` und `Require all denied` bietet es noch die Möglichkeit Umgebungsvariablen auszuwerten und abhängig vom Ergebnis einer solchen Auswertung den Zugriff zu gewähren.

Nutzen Sie die Apache Onlinedokumentation (http://httpd.apache.org/docs/2.4/expr.html), um die folgenden Aufgaben zu erledigen:

- Das Verzeichnis htdocs darf nur zwischen 07:00 Uhr und 16:00 Uhr für die Nutzer verfügbar sein.
- Legen Sie die beiden Verzeichnisse htdocs/msie und htdocs/mozilla an.
- Auf das Verzeichnis htdocs/msie dürfen nur solche Nutzer Zugriff haben, welche einen Internet Explorer als Browser benutzten (Variable HTTP_USER_AGENT).
- Auf das Verzeichnis htdocs/mozilla dürfen nur solche Nutzer Zugriff haben, welche den Browser Mozilla Firefox oder Google Chrome benutzen.
- Zugriffe auf das htdocs-Verzeichnis, die mit Hilfe veralteter Browserversionen erfolgen (MSIE 6.x oder Firefox 3.x) sollen direkt abgewiesen werden!

Zusätzlich zu dem Modul `authz_core_module` stellt der Apache noch weitere Module bereit, welche unterschiedlichste Autorisierungsmechanismen zur Verfügung stellen.

`authnz_ldap_module`	Modul zur Authentifizierung und Autorisierung mittels eines LDAP-Dienstes, z. B. Microsoft Active Directory
`authz_dbd_module`	Ermöglicht die Autorisierung mit Hilfe einer SQL-fähigen Datenbank z. B. MySQL
`authz_dbm_module`	Dieses Modul benutzt Nutzernamen und Passwörter, welche in DBM-Dateien gespeichert sind.
`authz_groupfile_module`	Liest Informationen über Gruppenzugehörigkeiten aus Klartextdateien und benutzt diese zur Autorisierung.
`authz_host_module`	Autorisierung mittels IP-Adresse oder Hostname
`authz_owner_module`	Ermöglicht die Zugriffssteuerung auf Dateien mit Hilfe der Besitzrechte.
`authz_user_module`	Ein Autorisierungsmodul, welches basierend auf dem Nutzernamen eine Autorisierung durchführt.

Autorisierung mittels IP-Adresse/Hostname

`authz_host_module` stellt die Möglichkeit bereit eine Autorisierung mittels IP-Adresse oder Hostname durchzuführen.

`authz_host_module` ist kein Ersatz für eine Firewall!

- Kommentieren Sie die Zugriffsregeln aus der letzten Übung aus!

- Konfigurieren Sie den Zugriff auf das htdocs-Verzeichnis so, das ihr eigener Client (Windows 7) keinen Zugriff mehr hat.
- Ändern Sie die Autorisierung für das htdocs-Verzeichnis so, das nur noch lokale Verbindungen zugreifen dürfen!
- Verbieten Sie jetzt alle Verbindungen aus Ihrem eigenen IP-Subnetz (10.2.11.64/26)!
- Verändern Sie die Autorisierung des htdocs-Verzeichnisses erneut, so das alle Computer, die Mitglied in der Domäne IT-TRAINING.FUS sind, zugriff haben.

Zugriffsbedingungen kombinieren

Benutzen Sie die Apache Onlinedokumentation, um herauszufinden welche Bedeutung die drei Direktiven `RequireAll`, `RequireAny` und `RequireNone` haben!

Richten Sie die folgenden Zugriffsregeln ein:

- Alle Computer, die der Domäne IT-TRAINING.FUS angehören, dürfen zwischen 7:00 Uhr und 16:00 Uhr auf das Verzeichnis htdocs zugreifen. Computer, welche dieser Domäne nicht angehören, dürfen nur zwischen 8:00 Uhr und 12:00 Uhr zugreifen.
- Alle Computer, die sich im IP-Subnetz des Webservers befinden (10.2.11.64/26) dürfen mit dem Mozilla FireFox auf das Verzeichnis htdocs/selfhtml zugreifen. Alle Computer die sich nicht in diesem IP-Subnetz befinden dürfen zusätzlich zum FireFox auch den Browser Internet Explorer nutzen!

Was bewirken die unten angegebenen Konfigurationen?

Listing 5.4: Zugriffsbedingungen kombinieren

```
# Konfiguration 1
  Require all denied
  Require local

# Konfiguration 2
  <RequireAny>
     <RequireAll>
       Require %{TIME_HOUR} -gt 7
     </RequireAll>
  </RequireAny>
  Require local
```

Abwärtskompatibilität

Im Apache HTTP-Server wurden bis zur Version 2.2 die Direktiven `Allow From` und `Deny From`, bzw. `Order`, statt der `Require`-Direktive eingesetzt. Um abwärtskompatibel zu bleiben, hat der Apache in der Version 2.4 ein Modul namens `mod_access_compat`. Dies enthält die oben genannten Direktiven, so dass diese auch weiterhin noch eingesetzt werden können.

5.2.4 Indexes

Ein einfacher Verzeichnisindex

Webseiten darzustellen ist nur eine Aufgaben, von vielen, die ein Webserver zu bewältigen hat. Eine andere, ist das bereitstellen von Dateien zum Download.

1. Erstellen Sie den Ordner htdocs/download.
2. Verschieben Sie die SELFHTML-Zip-Datei in den Downloadordner.
3. Konfigurieren Sie den Downloadordner so, dass alle Nutzer darauf zugreifen können (Zugriffsrechte haben)!
4. Laden Sie das Module mod_autoindex.so! Welche Aufgabe hat dieses Modul?
5. Testen Sie den Zugriff auf die URL: http://localhost/download!
6. Falls der Zugriff noch nicht funktionieren sollte: Welche zusätzliche Direktive müssen Sie angeben, damit eine automatische Indizierung des Downloadordners funktioniert?

Benutzten Sie jetzt eine .htaccess-Datei, um den Zugriff auf den Downloadornder zu ermöglichen! Welche Änderungen müssen dazu an der Hauptkonfigurationsdatei gemacht werden?

Ein bisschen Farbe ins Spiel bringen

Das Modul "mod_autoindex.so" bietet viele verschiedene Möglichkeiten, um einen einfachen, schmucklosen Verzeichnisindex ansprechender zu gestalten. Eine dieser Möglichkeiten ist beispielsweise das hinzufügen von Icons. Hierzu muss jedoch zu erst die Option `FancyIndexing` aktiviert werden.

Aktivieren Sie die Index-Options `FancyIndexing`! Was verändert sich dadurch sofort am Index?

Listing 5.5: Ein bisschen Farbe ins Spiel bringen

```
<Directory "htdocs/download">
  Require all granted

  # FancyIndexing aktivieren
  IndexOptions FancyIndexing

  Options      Indexes
</Directory>

```

Icons hinzufügen

Der Apache HTTP-Server kennt zwei Wege, um Dateitypen mit Icons zu verbinden:

- Durch Angabe des Dateityps
- Durch Angabe eines MIME-Types

Hierfür existieren drei Direktiven:

Direktive	Bedeutung
`AddIcon`	Verbindet verschiedene Dateitypen mit einem Icon. `AddIcon` nimmt den Namen einer Icon-Datei und eine oder mehrere Dateieindungen entgegen.
`AddIconByEncoding`	Verknüpft Dateien mit Hilfe des MIME-Types mit einem Icon. Hier wird der Name einer Icon-Datei und ein MIME-Type entgegen genommen.
`AddIconByType`	Verhält sich genau so, wie `AddIconByEncoding`

Was ist ein MIME-Type?

Da mit der HTTP-Server MIME-Typen verarbeiten kann, muss das Modul "mod_mime.so" geladen werden. Dieses Modul, wird zusammen mit einer Konfigurationsdatei für MIME-Typen geliefert. Um diese Datei zu laden, müssen die folgenden Direktiven in die Hauptkonfigurationsdatei aufgenommen werden:

Nehmen Sie die Direktiven aus eispiel{TypesConfig} in Ihre Hauptkonfigurationsdatei auf, und benutzen Sie die Direktive `AddIconByType`, um das Icon "compressed.gif" für ZIP-Dateien festzulegen! Welchen MIME-Type benötigen Sie hierfür? Testen Sie Ihr Ergebnis!

Listing 5.6: Icons hinzufügen

```
LoadModule mime_module "modules/mod_mime.so"
TypesConfig "conf/mime.types"
```

Aliase / Virtuelle Verzeichnisse

Das das Laden des Icons in der letzten Übung fehlschlug lagt daran, dass die benötigte Ressource, das Icon, außerhalb des DocumentRoot-Verzeichnisses liegt. Um Ressourcen außerhalb des DocumentRoot erreichen zu können, muss mit Aliasnamen bzw. Virtuellen Verzeichnissen gearbeitet werden.

Verzeichnisaliase ermöglichen es, Ressourcen außerhalb des DocumentRoot-Verzeichnisses einzubinden. Zuständig ist hierfür das "alias_module" (mod_alias.so).

Laden Sie das Modul mod_alias.so, und recherchieren Sie, welche Direktive für das Erstellen eines Verzeichnisaliases zust"andig ist.

Prüfen Sie, ob im Verzeichnisindex jetzt ein Icon angezeigt wird! Falls nicht, kontrollieren Sie Ihr Error-Log!

Um den Zugriff auf die Icon-Ressource zu ermöglichen, muss dem Nutzer der Zugriff auf das icons-Verzeichnis gewährt werden. Dies geschieht wiederum mit einem `<Directory>`-Kontainer.

Worin unterscheiden sich die beiden folgenden `<Directory>`-Kontainer (Hinweis: Benutzen Sie Ihr Error-Log, um den Unterschied festzustellen)?

Formatieren Sie den automatischen DirectoryIndex des Downloadordners nach den folgenden Angaben. Benutzen Sie die Onlinedokumentation zum "autoindex_module", um die benötigten Informationen zu recherchieren!

- Als Icon für das übergeordnete Verzeichnis soll dir.gif verwendet werden!
- Für *.msi-Dateien soll das Icon compressed.gif genutzt werden.
- Das Standardicon für alle unbekannten Dateitypen muss unknown.gif sein.
- Im Index sollen Verzeichnisse immer an erster Stelle aufgelistet werden, und dann erst die Dateien.
- Die Icons müssen ein Teil der Links sein.
- Dateinamen müssen in ihrer vollen Länge erhalten bleiben und dürfen nicht abgeschnitten werden.
- Die Spalte "Description" soll ausgeblendet werden.
- Die Standardsortierung für den Verzeichnisindex ist eine aufsteigende Reihenfolge.

Listing 5.7: Aliase / Virtuelle Verzeichnisse

```
<Directory "/icons">
  Require all granted
</Directory>

<Directory "E:/apache/httpd/2.4.2/icons">
  Require all granted
</Directory>

```

5.2.5 SSI

Was sind Server Side Includes?

Benutzen Sie Wikipedia, um zu klären, was Server Side Includes sind!

SSI aktivieren

- Fügen Sie die beiden Direktiven `AddType` und `AddOutputFilter` Ihrer Hauptkonfigurationsdatei hinzu! Was bedeuten sie?
- Laden Sie das Modul `include_module`!

Damit Server Side Includes in einem Verzeichnis funktionieren können, muss für dieses Verzeichnis noch die Option `Includes` festgelegt werden! Erst jetzt sind die SSI aktiviert!

Listing 5.8: SSI aktivieren

```
LoadModule include_module "modules/mod_include.so"

AddType         text/html .shtml
AddOutputFilter INCLUDES .shtml

<Directory "htdocs">
  AllowOverride none
  Options       Includes
  Order         allow,deny
  Allow From    all
</Directory>

```

Wie funktionieren Server Side Includes?

Das Beispiel auf dieser Seite soll die Funktionsweise von Server Side Includes verdeutlichen.

Erstellen Sie die im folgenden dargestellte HTML-Datei ssi.shtml im Verzeichnis htdocs/ssi!

Ein Zugriff auf die obige HTML-Datei zeigt dieses Ergebnis:

- Testen Sie den Zugriff auf ssi.shtml !
- Zusätzlich zur Option `Includes` gibt es nocht die Option `InlcudesNOEXEC` ! Worin unterscheiden sich die beiden?

Listing 5.9: Wie funktionieren Server Side Includes?

```html
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
        "http://www.w3.org/TR/html4/strict.dtd">
<html>
  <head>
    <title>Server Side Includes</title>
  </head>
  <body>
        <h1>SSI Rocks!</h1>
        <p>Heute ist der <!--#echo var="DATE_LOCAL" -->.</p>
        <!--#config timefmt="%d.%m.%Y" -->
        <p>Oder anders ausgedrückt, der <!--#echo var="DATE_LOCAL" -->.</p>
        <p>Als Browser benutzen Sie: <!--#echo var="HTTP_USER_AGENT" -->.</p>
  </body>
</html>
```

5.2.6 CGI

PERL over CGI

Das Common Gateway Interface stellt eine Schnittstelle zwischen dem Apache HTTP-Server, und externen Programmen, die den Bildschirminhalt generieren, dar. Diese externen Programme werden auch oft als CGI-Programme oder CGI-Skripte bezeichnet. Mit CGI kann auf eine sehr einfache Art und Weise dynamischer Content erzeugt werden.

Ein Zitat aus http://de.wikipedia.org/wiki/PERL

Perl ist eine freie, plattformunabhängige und interpretierte Programmiersprache (Skriptsprache), die mehrere Programmierparadigmen unterstützt.

Entworfen worden ist diese Sprache 1987, von Larry Wall, einem Linguisten. Sie sollte als Werkzeug zur Analyse und Verarbeitung von Textdateien dienen. Da sie seit vielen Jahren eine weite Verbreitung in der Welt des Internets gefunden hat, soll sie auch hier, für die Demonstration von CGI-Programmen dienen.

Im Internet existieren zwei große Distributionen von Perl:

- Strawberry Perl
- Active Perl

Bei Active Perl handelt es sich um die Perl-Variante der Firma Activestate. Es existiert in den folgenden Versionen:

- Community Edition
- Bussiness Edition
- Enterprise Edition

Lediglich die Community Edition ist als kostenlose Version erhältlich, darf jedoch nur zu Testzwecken und für den nicht kommerziellen Einsatz genutzt werden. Strawberry Perl hingegen ist eine Open Source Implementierung der Sprache Perl, die ohne Einschränkungen durch eine Lizenz genutzt werden darf. Im folgenden wird in dieser Unterrichtsunterlage Strawberry Perl zum Einsatz kommen.

Laden Sie Strawberry Perl in der aktuellsten Version aus dem Internet herunter und installieren Sie es! Als Installationsverzeichnis soll E:\perl\strawberry\5.14.2.1 dienen!

CGI mit ScriptAlias konfigurieren

Um den Apache HTTP-Server dazu überreden zukönnen das Common Gateway Interface zu aktivieren, ist es notwendig das `cgi_module` (mod_cgi.so) zu laden. Da es sich bei CGI-Skripten um ausführbare Programme handelt, sollten diese nicht mit anderen Dateien, wie z. B. HTML-Dateien vermischt, sondern in einem separaten Verzeichnis aufbewahrt werden. Um dieses Verzeichnis für Nutzer zugänglich zu machen, und gleichzeitig die Ausführung der CGI-Programme zu erlauben, wird die `ScriptAlias`-Direktive verwendet.

1. Schlagen Sie die Syntax der `ScriptAlias`-Direktive in der Onlinedokumentation nach!
2. Versehen Sie das Verzeichnis [ServerRoot]/cgi-bin mit dem ScriptAlias cgi-bin Hinweis: [ServerRoot] stellt einen Platzhalter für Ihr Serverroot-Verzeichnis, z. B. E:/apache/httpd/2.4.2, dar!
3. Welcher Schritt muss noch unternommen werden, damit CGI-Skripte aus dem Verzeichnis cgi-bin ausgeführt werden können?

Listing 5.10: CGI mit ScriptAlias konfigurieren

```
1  LoadModule cgi_module "modules/mod_cgi.so"
2
```

CGI mit einem Alias konfigurieren

Eine andere Möglichkeit CGI zu aktivieren besteht darin, dass die Option `ExecCGI`, in Zusammenhang mit einem Verzeichnisalias und einem Actionhandler genutzt wird.

Legen Sie das Verzeichnis E:/apache/httpd/2.4.2/cgi-bin_2 an!

Analog zur ersten Variante, muss statt einem ScriptAlias, ein Alias angelegt werden. Im zweiten Schritt, ist es notwendig einen Verzeichniskontainer für das cgi-bin_2-Verzeichnis anzulegen.

Zeile zwei, `AddHandler cgi-script .cgi .pl` bewirkt, dass alle Dateien, mit den Endungen .cgi oder .pl als CGI-Skripte betrachtet und ausgeführt werden. Zeile Nummer 7, `Options ExecCGI` hat zur Folge, dass die Ausführung von CGI-Skripten erlaubt wird. Durch Entfernung dieser Option kann die Ausführung von CGI-Skripten unterbunden werden, ohne das die restliche Konfiguration dazu verändert werden müsste.

Listing 5.11: CGI mit einem Alias konfigurieren

```
Alias "/cgi-bin_2" "E:/apache/httpd/2.4.2/cgi-bin_2"

<Directory "E:/apache/httpd/2.2.21/cgi-bin_2">
  #Hinzufuegen eines Actionhandlers fuer CGI
  AddHandler    cgi-script .cgi .pl

  AllowOverride None
  Require all granted

  #Die Ausfuehrung von CGI-Skripten zulassen
  Options       ExecCGI
</Directory>

```

Jetzt wird es spannend - Das erste Perl-Programm

Wenn es so etwas wie eine Tradition im Bereich der Programmierung gibt, dann die, dass das erste Programm eines jeden Programmierers die Worte "Hallo Welt!" auf dem Bildschirm ausgibt. Da diese Unterrichtsunterlage keinesfalls mit dieser Tradition brechen möchte, wird sie auch hier fortgesetzt:

- Erstellen Sie das unten abgebildete PERL-Programm und speichern Sie es als Datei, im Verzeichnis E:/apache/httpd/2.4.2/cgi-bin, mit dem Namen hallo_welt.pl ab.
- Testen Sie das Programm, in dem Sie folgende URL aufrufen:

http://localhost/cgi-bin/hallo_welt.pl

Es sollte der Schriftzug "Hallo Welt!" erscheinen!

Das Programm hallo_welt.pl ist wie folgt aufgebaut:

Die erste Zeile ist eine spezielle Kommentarzeile, zu erkennen an den Zeichen #!. Sie hat die Aufgabe, dem Browser mitzuteilen, wo sich der Kommandointerpreter, in diesem Falle die Datei perl.exe, befindet.

Die zweite Zeile benutzt das `print`-Kommando, um den Contenttype und den für die Ausgabe zu verwendenden Zeichensatz an den Browser weiterzugeben. Der Browser unterdrückt die Ausgabe dieses Textes auf dem Bildschirm, verarbeitet aber trotzdem die Informationen.

Die dritte Zeile ist die alles entscheidende. Sie sorgt für die Bildschirmausgabe von "Hallo Welt!".

Listing 5.12: Jetzt wird es spannend - Das erste Perl-Programm

```
#!"E:/perl/strawberry/5.14.2.1/perl/bin/perl.exe"
print "Content-type: text/plain; charset=iso-8859-1";
print "Hallo Welt!";

```

Die Spannung steigt - Das zweite Perl-Programm

Das zweite Versuch mit CGI ein PERL-Programm aufzurufen soll etwas komplexer gestaltet werden. Im ersten Schritt wird eine HTML-Datei erstellt, die das PERL-Programm mittels eines SUBMIT-Buttons aufruft.

Erstellen Sie die unten abgebildete HTML-Datei begruessung.html und legen Sie sie im Verzeichnis htdocs ab!

Das Beispiel enthält drei Tags, die für die Dateneingabe und -Verarbeitung zuständig sind:

<input type="text" name="name"> erzeugt ein Textfeld, in das der Nutzer einen beliebigen Freitext eintragen kann. In unserem Beispiel soll hier der Vor-/Nachname eines Benutzers eingetragen werden.

<input type="submit" value="Begrüssen"> erstellt einen Button mit der Aufschrift "Begrüßen". Hierbei handelt es sich um einen sogenannten "SUBMIT-Button", was bedeutet, dass beim Drücken des Buttons alle Daten aus dem HTML-Formular an ein Ziel übertragen werden. In diesem Falle ist das PERL-Skript hallo_formular.pl das Ziel.

<form action="cgi-bin/hallo_formular.pl" method="POST"> legt fest, dass durch das Drücken eines SUBMIT-Buttons die HTTP-Methode POST genutzt wird, um alle Formulardaten an das PERL-Skript hallo_formular.pl zu übertragen.

Erstellen Sie die Datei hallo_formular.pl und speichern Sie sie im Verzeichnis cgi-bin_2 ab. Testen Sie ob HTML-Formular und PERL-Skript funktionieren!

Listing 5.13: Die Spannung steigt - Das zweite Perl-Programm

HTML

```
#begruessung.html
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
        "http://www.w3.org/TR/html4/strict.dtd">
<html>
  <head>
    <title>Ein HTML-Formular mit PERL</title>
  </head>
  <body>
        <h1>Sei gegrüsst!</h1>
        <form action="cgi-bin/hallo_formular.pl" method="POST">
        <div>
        <p>Sag mir deinen Namen!</p>
        <input type="text" name="name">
        </div>
        <div>
          <input type="submit" value="Begrüssen">
        </div>
  </body>
</html>

#hallo_formular.pl
#!"E:/perl/strawberry/5.14.2.1/perl/bin/perl.exe"

print "Content-type: text/plain; charset=iso-8859-1

";

read (STDIN, $query, $ENV{CONTENT_LENGTH});

print ("Hallo ");
print (split((/.+=/), $query, 0));
print (", wie geht es Dir?");

```

5.2.7 PERL mit mod_perl.so

Konfigurieren von mod_perl.so

Eine zweite Variante PERL in den Apache HTTP-Server einzubinden, stellt das Modul mod_perl.so dar. Dieses Modul sorgt genauso wie mod_cgi.so für die Ausführung von PERL-Programmen, aber mit dem Vorteil, dass es nur PERL-Programme, und keine anderen Skripte/Programme zulässt. Dadurch kann verhindert werden, das "aus Versehen" andere Programme, wie z. B. PHP- oder C++ Programme ausgeführt werden.

Als erstes muss die Installation von mod_perl.so erfolgen. Diese geschieht mit dem PERL eigenen Installationsprogramm "pip" (Perl Installation Program).

1. Öffnen Sie eine Windows-Command-Shell (cmd.exe)
2. Setzen Sie, mit Hilfe des set-Kommandos, die Umgebungsvariable HTTP_PROXY auf den folgenden Wert:

   http://username:passwort@192.168.4.250:3128
3. Führen Sie das folgende PIP-Kommando aus, um zusätzliche Dateien für das Modul mod_perl.so zu installieren

   pip http://strawberryperl.com/package/kmx/mod_perl/5.12_x64/mod_perl-2.0.4-MSWin32-x86-multi-thread-5.12.par
4. Führen Sie das folgenden PIP-Kommando aus, um zusätzliche Dateien für das das Modul libapreq zu installieren:

pip http://strawberryperl.com/package/kmx/mod_perl/5.12_x86/libapreq2-2.12-MSWin32-x86-multi-thread-5.12.par

5. Kopieren Sie die folgenden Dateien in das Verzeichnis [ServerRoot]/modules:

- mod_perl.so
- mod_apreq2.so
- libapreq2.dll

- Konfigurieren Sie den Apache so, dass die folgenden beiden Module geladen werden:
  - `perl\_module` (mod_perl.so)
  - `apreq\_module` (mod_apreq2.so)
- Legen Sie das Verzeichnis [ServerRoot]/perl an.
- Kopieren Sie das PERL-Skript "hallo_formular.pl" in das Verzeichnis perl, und entfernen Sie die erste Zeile (Kommentarzeile) aus dieser Datei.
- Passen Sie die Datei begruessung.html so an, dass das PERL-Skript aus dem Verzeichnis perl genutzt wird.
- Konfigurieren Sie in Ihrer Hauptkonfigurationsdatei das Verzeichnis perl so, wie in unten angegeben!
- Legen Sie einen Alias für das Verzeichnis perl an!
- Starten Sie den Apache Webserver neu!
- Testen Sie, ob begruessung.html funktioniert!

Hier werden drei Direktiven verwendet, die für die Ausführung von PERL-Programmen verantwortlich sind. Zwei davon, sind neu.

Die Direktive `PerlHandler ModPerl::Registry` stellt die Verbindung zwischen mod_perl.so und dem PERL-Skript her, in dem ein sogenannter "Perlhandler" konstruiert wird. Dieser ermöglicht es mod_perl.so dann, das Skript aufzufinden und auszuführen.

Die zweite Direktive ist `PerlSendHeader On`. Sie sorgt dafür, dass mod_perl.so jedes PERL-Programm nach HTTP-Header-Zeilen durchsucht und diese ausführt. Im Falle des vorangegangenen HalloWelt-Beispiels ist dies die Zeile:

print "Content-type: text/plain;charset=iso-8859-1 ";

Testen Sie war passiert, wenn Sie die Direktive `PerlSendHeader` auf den Wert "Off" einstellen!

Die letzte Direktive `AddHandler perl-script .pl` ist bereits von CGI her bekannt. Sie ermöglicht es, dass alle Dateien mit der Endung .pl von Apache als ausführbare PERL-Skripte betrachtet werden.

Listing 5.14: Konfigurieren von mod_perl.so

```
<Directory "E:/apache/httpd/2.4.2/perl">
  Options        ExecCGI
  PerlHandler    ModPerl::Registry
  PerlSendHeader On
  AddHandler     perl-script .pl
  Require        all granted
</Directory>

```

5.2.8 Apache und PHP

PHP installieren und konfigurieren

Eine weitere Programmiersprache die im Verlauf der Jahre große Verbreitung im Internet gefunden hat ist PHP. Die Abkürzung PHP stelt ein rekursives Akronym(Ein Akronym ist dann rekursiv, wenn das Akronym selbst in der Bedeutung vorkommt!), mit der Bedeutung "PHP Hypertext Preprocessor" dar. Erschaffen wurde PHP im Jahr 1995 von Rasmus Lerdorf, der es später mit den beiden Softwareentwicklern Andi Gutmans und Zeev Suraski gemeinsam weiterentwickelte.

Im laufe seiner Entwicklung wurde PHP stark von den Sprachen Perl, C, C++ und Java beeinflusst, weshalb an einigen Stellen die Syntax eine hohe Ähnlichkeit mit diesen Sprachen hat. Da PHP auf fast 75 % aller Webseiten zur Erzeugung/ Darstellung von dynamischen Inhalten genutzt wird, ist es zum de facto Standard im Internet geworden. Aktuell läuft die Entwicklung an der Version 5.4.4 (Stand 14.06.2012).

Unter der Adresse http://windows.php.net/download/ können alle aktuellen PHP-Versionen für Microsoft Windows heruntergeladen werden.

- Laden Sie PHP 5.4.4 VC9 x86 Thread Safe als Zip-Datei herunter!
- Entpacken Sie die Zip-Datei in das Verzeichnis E:\PHP
- Legen Sie eine Kopie der Datei php.ini-production an und benennen Sie diese um in: php.ini!
- Laden Sie die Datei php5apache2_2.dll als Modul für Apache!
- Fügen Sie in Ihrer Hauptkonfigurationsdatei, dem Verzeichniskontainer für htdocs, die Direktive `AddHandler application/x-httpd-php .php` hinzu!
- Starten Sie den Apache neu!
- Erstellen Sie die unten angegebene PHP-Datei, und speichern Sie diese mit dem Namen index.php im Verzeichnis [ServerRoot]/htdocs!
- Rufen Sie die Datei index.php

im Browser auf!

Listing 5.15: PHP installieren und konfigurieren

```php
<?php
  phpinfo();
?>
```

PHP

PHP + MySQL + Mediawiki

Diese Sektion soll ein einfaches und doch realastisches Beispiel dafür geben, welchen Einsatzzweck PHP in der Praxis hat. Mit Hilfe des Datenbankmanagementsystems MySQL und der PHP-Software MediaWiki wird auf dem Apache Webserver ein kleines Wiki-System erzeugt. Da an dieser Stelle der Einsatz von PHP im Vordergrund steht, werden die Installationen von MySQL und MediaWiki nur rudimentär erläutert.

MySQL herunterladen und installieren

MySQL ist ein relationales Datenbankmanagementsystem, das als Grundlage für sehr viele dynamische Webauftritte dient. Ursprünglich wurde es von dem schwedischen Unternehemn MySQL AB entwickelt, dass jedoch im Jahr 2008 von Sun Microsystems aufgekauft wurde. 2010 wurde die Firma Sun Microsystems schließlich von Oracle aufgekauft.

MySQL existiert sowohl als kommerzielle Software, als auch als Open Source Produkt, in der Community Edition. Diese steht unter dem folgenden Link zum Download zur Verfügung:

http://www.mysql.de/downloads/mysql

Laden Sie MySQL, als Windows MSI-Installer von der obigen Adresse herunter und installieren Sie es mit den folgenden Angaben!

- Installationstyp: Complete
- Haken setzen bei: Launch the MySQL Instance Configuration Wizard
- Detailed Configuration
- Server Machine
- Multifunctional Database
- InnoDB Tablespace Settings: E:MySQLDataFiles
- Online Transaction Processing (OLTP)
- Haken setzen bei: Enable TCP/IP Networking (Port: 3306)
- Haken setzen bei: Enable Strict Mode
- Zeichensatz: Standard Character Set
- Haken setzen bei: Install as Windows Service (Servicename: MySQL)
- Haken setzen bei: Launch the MySQL Server automatically
- Haken setzen bei: Inlcude Bin Directory
- Haken setzen bei: Modify Security Settings (root-Passwort: password)

Nach erfolgter Installation sollte MySQL als Windows Dienst, unter dem Namen glqq MySQLgrqq laufen.

MediaWiki installieren

MediaWiki wird unter http://www.mediawiki.org/wiki/Download/de als Archivdatei, im Format *.tar.gz, zum herunterladen zur Verfügung gestellt.

- Laden Sie MediaWiki von der obigen Adresse herunter und entpacken Sie den Inhalt des Archives nach E:apachehttpd2.4.2htdocs, so dass dort das Verzeichnis mediawiki-1.x.x entsteht!
- Benennen Sie das Verzeichnis mediawiki-1.x.x in mediawiki um!

Bevor die Installation des MediaWiki gestartet werden kann, müssen noch einige Einstellungen für die Nutzung von PHP vorgenommen. PHP wird mit Hilfe der Datei php.ini konfiguriert.

Öffnen Sie die Datei E:phpphp.ini, und nehmen Sie die im folgenden beschriebenen Änderungen vor!

- Suchen Sie die Zeile ;extension_dir ="ext" und entfernen Sie das Kommentarzeichen (;)!
- Suchen Sie die Zeile ;extension=php_mysql.dll und entfernen Sie das Kommentarzeichen!

- Speichern und schließen Sie die Datei.
- Kopieren Sie die Datei E:phpphp.ini nach C:Windows

ODER

- Setzen Sie die Umgebungsvariable **PHPRC** mit dem Wert: E:php
- Starten Sie die Installation von Mediawiki, in dem Sie die Adresse http://localhost/mediawiki/index.php aufrufen!
- Klicken Sie den Link "Please set up the wiki first" an.
- Installieren Sie MediaWiki mit den folgenden Konfigurationseinstellungen:
- Sprache: de - Deutsch
- Sprache des Wikis: de - Deutsch
- Datenbankserver: localhost
- Datenbankname: my_wiki
- Datenbanktabellenpräfix:
- Name des Datenbankbenutzers: root
- Passwort des Datenbankbenutzers: password
- Haken entfernen bei: Dasselbe Konto wie während des Installationsvorgangs verwenden
- Names des Datenbankbenutzers: mediawiki
- Passwort des Datenbankbenutzers: password
- Haken setzen bei: Sofern nicht bereits vorhanden, muss nun das Konto erstellt werden
- Speicher-Engine: InnoDB
- Datenbankzeichensatz: UTF-8
- Name des Wikis: my_mediawiki
- Name des Projektnamensraums: Entspricht dem Namen des Wikis
- Administratorkonto - Name: wikiadmin
- Administratorkonto - Passwort: passwordwiki
- Administratorkonto - E-Mail-Adresse:
- Option: Nein, das Wiki soll nun installiert werden

Befolgen Sie die weiteren Anweisungen auf dem Bildschirm!

5.2.9 Die Log-Dateien des Apache

Das Error-Log

Das Error-Log

Ein ganz wesentlicher Aspekt bei Verwaltung eines Serverproduktes, wie z. B. eines Webservers ist es, Informationen über das Verhalten des Servers und über die Zugriffe auf den Server gewinnen zu können. Der Apache HTTP-Server bietet hierzu ein umfassendes Loggin an.

Das Error-Log ist die zentrale Logdatei des Apache Webservers. Hier laufen alle Fehler- und Diagnosemeldungen auf, die während des Serverbetriebs entstehen. Unter Microsoft Windows heißt die betreffende Datei standardmässig error.log und liegt im Verzeichnis [ServerRoot]/logs. Der Inhalt dieser Datei sollte in regelmässigen Zeitabständen immer wieder kontrolliert bzw. im Fehlerfalle als erstes geprüft werden.

Die Meldungen im Error-Log haben ein festes Format, das durch den Admin nicht geändert werden kann. Ein Beispiel für eine Fehlermeldung könnteso aussehen:

```
[Mon Nov 28 08:57:37 2011] [error] [client 127.0.0.1]
Premature end of script headers: login.pl
```

Der erste Teil der Meldung ist das Datum, an dem die Meldung aufgelaufen ist. Der zweite Teil ([error]) stellt den Schweregrad der Fehlermeldung dar. Der dritte Teil gibt die Adresse des Clients an, der durch einen Serverzugriff den Fehler ausgelöst hat. Als letztes kommt schließlich die eigentliche Fehlermeldung.

Konfiguriert wird das Error-Log mit Hilfe der Direktive `ErrorLog`. ErrorLog "logs/error.log"

Auch wenn die `ErrorLog`-Direktive nicht angegeben wurde erstellt der Apache eine Log-Datei namens error.log, im Verzeichnis [ServerRoot]/logs.

Access Logs

Zusätzlich zum Error-Log kennt der Apache sogenannte Access-Logs. Diese enthalten verbindungsspezifische Informationen zu den Fehlermeldungen im Error-Log. Das Format einer Access-Log Datei ist frei wählbar. Ein Access-Log kann mittels der Direktive `CustomLog` erstellt werden.

Es ist erlaubt mehrere `CustomLog`-Direktiven in der Hauptkonfigurationsdatei zu verwenden, um mehrere Access-Logs zu konfigurieren.

- Wie lautet die Syntax der Direktive `CustomLog`?
- Welche Aufgabe hat die Direktive `LogFormat`?
- Wo finden Sie in der Onlinedokumentation Informationen über die einzelnen Formatparameter der Direktive `LogFormat`?
- Was verbirgt sich hinter der Abkürzung CLF?
- Wie kann ein Format für ein bestimmtes Access-Log festgelegt werden (Syntax)?

Rotierende Log-Dateien (Log rotation)

Bei einem Webserver auf den täglich mehrere tausend Zugriffe erfolgen, kann eine Log-Datei sehr schnell anwachsen und dadurch unübersichtlich werden. Um so etwas zu vermeiden, kann der Apache seine Log-Dateien in konfigurierbaren Zeitabständen automatisch austauschen. Dieser Mechanismus wird als "Log Rotation" bezeichnet und ist sowohl für das Error-Log als auch für alle Access-Logs getrennt einstellbar.

Sowohl bei der `ErrorLog`- als auch bei der `CustomLog`-Direktive sind zwei neue Angaben hinzugekommen. Der erste Teil ist `|bin/rotatelogs.exe` . Der senkrechte Strich, am Anfang der Zeile wird als "Pipe-Zeichen" bezeichnet. Er sorgt für das Pipelining der Loging-Informationen an das Programm rotatelogs.exe , welche dann die Meldungen in die jeweilige Log-Datei schreibt, und alle n-Sekunden (hier ist n = 21600) eine neue Datei beginnt.

Unter dem Begriff "Pipelining" versteht man eine Art der Interprozesskommunikation. Das heißt zwei Prozesse tauschen Informationen untereinander aus. In diesem speziellen Fall gibt der Apache Webserver Informationen an das Programm rotatelogs.exe weiter.

Konfigurieren Sie Ihr Error-Log so, dass es alle 30 Sekunden rotiert!

Listing 5.16: Rotierende Log-Dateien (Log rotation)

```
ErrorLog  "|bin/rotatelogs.exe logs/error.log 21600"
LogFormat "%h %l %u %t "%r" %>s %b" common
CustomLog "|bin/rotatelogs.exe logs/access.log 21600" common
```

Bedingtes Logging - Das setenvif_module-Modul

Um schon während des Loggings eine Auswahl treffen zu können, welche Informationen protokolliert werden sollen, kennt der Apache die Möglichkeit, mit Hilfe des Moduls `mod_setenvif.so` Bedingungen für das Logging zu formulieren.

Recherchieren Sie mit Hilfe der Onlinedokumentation nach der Bedeutung der Anweisungen aus dem unten angegebenen Beispiel!

Listing 5.17: Bedingtes Logging - Das setenvif_module-Modul

```
LoadModule setenvif_module "module/mod_setenvif.so"
SetEnvIf   Remote_Addr "127.0.0.1" dontlog
CustomLog  "logs/access.log" env!=dontlog

```

Bedingtes Logging - LogFormat und HTTP-Fehlercodes

Die LogFormat-Direktive erlaubt es, Log-Format für bestimmte HTTP-Fehlercodes festzulegen. Im unten gezeigten Beispiel 1 wird eine Access-Log Datei namens logs/access_http400error.log erstellt, in welche nur Einträge für den HTTP-Fehlercode 404 - Not Found - gemacht werden. Beispiel 2 hingegen zeigt die Möglichkeit der Verneinung. In der Datei logs/access_referer.log werden alle Fehlercodes, außer 200, 302 und 304 geloggt.

Listing 5.18: Bedingtes Logging - LogFormat und HTTP-Fehlercodes

```
#Beispiel 1
LogFormat "%404{User-agent}i %U %a" http404
CustomLog "logs/access_http404error.log" http404

#Beispiel 2
LogFormat "%!200, 302, 304 {Referer}i" referer
CustomLog "logs/access_referer.log" referer
```

5.2.10 Virtual Hosting

IP-Basiertes Virtual Hosting

Unter dem Begriff "Virtual Hosting" versteht man die Möglichkeit auf einem einzigen physikalischen Webserver mehrere Webangebote gleichzeit bereitzustellen.

Virtual Hosting kann IP-Basiert oder Namensbezogen erfolgen. Um IP-Basiertes Virtual Hosting betreiben zu können, braucht ein Webserver mehrere IP-Adressen, bzw. eine IP-Adresse mit mehreren freien Ports. Die Reihenfolge der Schritte zur Konfiguration von V-Hosts ist:

1. Konfiguration der IP-Einstellungen am Webserver durch Sys-/Netzwerkadmin.
2. Freischalten von IP-Adresse und Port an einer evtl. vorhandenen Firewall durch den Sys-/Netzwerkadmin.
3. Konfiguration des Apache Webservers mittles des `<VirtualHost>`-Kontainers.

Der Kontainer `<VirtualHost>` wird verwendet, um in der Hauptkonfigurationsdatei des Apache Webservers einen Virtuellen Host darzustellen. Am Ende dieser Sektion ist ein Beispiel dazu angegeben.

1. Wo können Sie in der Onlinedokumentation ersehen, welche Direktiven für einen `<VirtualHost>`-Kontainer zulässig sind?
2. Warum ist hier die Direktive `ServerName` sehr wichtig? Recherchieren Sie unter: http://httpd.apache.org/docs/2.4/dns-caveats.html
3. Benutzen Sie das unten angegebene Konfigurationsbeispiel, um einen Webauftritt für SelfHTML zu erstellen! SelfHTML soll unter der URL http://localhost:8080 für alle Nutzer erreichbar sein! Welche Konfigurationsschritte sind dazu notwendig?

Für die Angabe eines Ports gibt es drei Möglichkeiten:

- Keine Angabe, z. B. 127.0.0.1: In diesem Fall wird Port 80 angenommen.
- Angabe eines Ports, z. B. 127.0.0.1:8080: Es wird der angegebene Port verwendet.
- Angabe eines *, z. B. 127.0.0.1:*: Ein so konfigurierter Virtual Host beantwortet Anfragen auf allen Ports.

Erstellen Sie einen weiteren virtuellen Host, der auf Port 80 den MediaWiki bereitstellt!

Listing 5.19: IP-Basiertes Virtual Hosting

```
<VirtualHost "127.0.0.1:8080">
  ServerAdmin    root@127.0.0.1
  DocumentRoot   "E:/apache/httpd/2.4.2/htdocs/port_8080"
  ServerName     "www.example.org"
  DirectoryIndex index.php
</VirtualHost>

```

Virtuelle Hosts richtig konfigurieren

Sobald eine größere Anzahl von Virtuellen Hosts auf einem Apache Webserver konfiguriert wird, kann es leicht passieren, dass dessen Konfigurationsdatei sehr unübersichtlich wird. Um dies zu vermeiden, bietet er die Möglichkeit einzelne Konfigurationsabschnitte, z. B. die Konfiguration eines Virtual Hosts, in eine andere Datei auszulagern und diese in die Konfiguration des Servers einzubinden.

Wenn Sie virtuelle Hosts konfigurieren, sollte der physikalische Webserver höchstens noch eine Fehlerseite bereitstellen. Alle Webauftritte werden dann durch die virtuellen Server dargestellt.

Benutzen Sie die Apache Onlinedokumentation, um zu Recherchieren wie die beiden Direktiven `Include` und `IncludeOptional` funktionieren, bzw. worin sich beide unterscheiden!

Benutzen Sie für die folgenden Aufgaben die beiden unten angegebenen Virtuellen Hosts VHost1 und VHost2!

- Speichern Sie die Konfigurationen der beiden Virtual Hosts in jeweils einer eigenen Datei (port_80.conf und port_8080.conf)
- Konfigurieren Sie Ihren Apache so, dass beide VHosts dargestellt werden. Sollte die Konfiguration von VHost 1 nicht gelesen werden können, muss der Start des Apache Servers unterbrochen werden. Bei VHost 2 ist diese Maßnahme nicht notwendig, bzw. nicht erwünscht!
- Konfigurieren Sie jetzt VHost3 und speichern Sie seine Konfiguration in der Datei generic.conf. Binden Sie diese Datei an erster Stelle ein, und zwar genauso wie die Konfiguration von VHost2.
- Testen Sie den Zugriff auf die URL http://localhost! Welcher VHost antwortet?
- Verschieben Sie VHost 3 auf den ersten Platz, so dass die Reihenfolge der VHosts 3, 1, 2 lautet und testen Sie erneut den Zugriff auf http://localhost! Welcher VHost antwortet jetzt?
- Legen Sie für die beiden virtuellen Hosts 1 und 3 explizit den Port 80 fest (`<VirtualHost "127.0.0.1:80">`)! Welche Antwort erhalten Sie, wenn Sie jetzt auf http://localhost zugreifen?
- Konfigurieren Sie einen weiteren virtuellen Host der Unter der IP-Adresse 10.3.11.xx (xx = Platznummer) die Datei begruessung.html anzeigt, und PERL-Skripte verarbeiten kann. Speichern Sie auch dessen Konfiguration in einer eigenen Datei, und binden Sie diese an dritter Stelle so ein, dass der Serverstart unterbrochen wird, falls die Datei nicht verfügbar ist.
- Konfigurieren Sie den virtual Host `<VirtualHost 10.3.11.xx>` explizit für Port 80 (`<VirtualHost 10.3.11.xx:80>`)!
- Tragen Sie die folgende Direktive in Ihre Konfiguration ein: `NameVirtualHost 10.3.11.xx:*`!
- Testen Sie den Zugriff auf die URL http://10.3.11.xx:80! Welche Antwort erhalten Sie?

Listing 5.20: Virtuelle Hosts richtig konfigurieren

```
#VHost 1 - port_80.conf
<VirtualHost "127.0.0.1">
  ServerAdmin    root@127.0.0.1
  DocumentRoot   "E:/Apache/httpd/2.4.2/htdocs/port_80"
  ServerName     "www.example.org"
</VirtualHost>

#VHost 2 - port_8080.conf
<VirtualHost "127.0.0.1:8080">
  ServerAdmin    root@127.0.0.1
  DocumentRoot   "E:/Apache/httpd/2.4.2/htdocs/port_8080"
  ServerName     "www.example.de"
  DirectoryIndex index.php
</VirtualHost>

#VHost 3 - generic.conf
<VirtualHost "127.0.0.1:*">
  ServerAdmin    root@127.0.0.1
  DocumentRoot   "E:/Apache/httpd/2.4.2/htdocs/generic"
  ServerName     "www.example.com"
</VirtualHost>

```

Name-Based Virtual Hosts konfigurieren

Moderner und daher auch von der Apache Foundation empfohlen ist das Name-Based Virtual Hosting. Bei dieser Variante werden die virtuellen Hosts nach ihrem Hostnamen unterschieden, d. h. es können sich viele V-Hosts eine einzige IP-Adresse teilen.

Für die Aktivierung des Name-Based V-Hosting, muss die `NameVirtualHost`-Direktive genutzt werden. Um z. B. einen namens-basierten virtuellen Host auf der IP-Adresse 10.2.11.120 und dem DNS-Namen selfhtml.it-training.fus zu erstellen, muss eine Konfiguration eingerichtet werden, wie sie unten abgebildet ist. Dieses Beispiel ist jedoch nur ein Auszug aus der Hauptkonfigurationsdatei. An dem `<VirtualHost>`-Kontainer hat sich nichts geändert. Lediglich die `NameVirtualHost`-Direktive ist neu hinzugekommen.

Um das Name-Based V-Hosting zu aktivieren, muss die `NameVirtualHost`-Direktive mit der IP-Adresse und dem Port konfiguriert werden, auf dem der V-Host aktiviert werden soll. Des weiteren muss jeder Name-Based V-Host Z W I N G E N D eine `ServerName`-Direktive erhalten, um von den anderen V-Hosts unterscheidbar zu bleiben!

Listing 5.21: Name-Based Virtual Hosts konfigurieren

```
Listen 10.2.11.120:80

NameVirtualHost "10.2.11.120:80"

<VirtualHost "10.2.11.120:80">
  DocumentRoot   "E:/Apache/httpd/2.4.2/htdocs/selfhtml.it-training.fus"
  ServerAdmin    root@127.0.0.1
  ServerName     "selfhtml.it-training.fus"
</VirtualHost>

```

Der Host-Server antwortet nicht mehr!

Sobald das Name-Based Virtual Hosting aktiviert wurde antwortet der Webserver selbst nicht mehr. Antworten werden nur noch von den virtuellen Servern verschickt!

Um sicherzustellen, dass immer eine Antwort von Ihrem Webserver kommt, sollte ein Default V-Host eingerichtet werden. Für den Default V-Host gelten folgende Regeln:

- Er sollte immer als erstes in der Liste der V-Hosts stehen, so dass er standardmäßig antwortet.
- Die Direktiven `DocumentRoot` und `ServerName` sollten die gleichen Argumente aufweisen, wie die gleichnamigen Direktiven im Hauptserver.

Das der Default V-Host immer als erstes in der Liste der V-Hosts stehen sollte, hängt damit zusammen, wie der Apache Webserver Client-Anfragenabarbeitet. Wenn eine Anfrage eintrifft:

1. Prüft der Apache zuerst, ob die IP-Adresse einer der IP-Adressen entspricht, die in einer `NameVirtualHost`-Direktive angegeben wurden.
2. Wenn dies der Fall ist, wird jeder einzelne `<VirtualHost>`-Kontainer durchsucht, bis einer gefunden wird, dessen `ServerName`-Direktive dem angeforderten Server entspricht.
3. Ist ein solcher Server gefunden worden, wird dessen Konfiguration verwendet. Falls nicht, wird der erste V-Host in der Liste genutzt, dessen IP-Adresse passend ist!

Wenn zwei virtual Hosts mit gleicher IP-Adresse/Port-Konfiguration auf treten, diese aber unterschiedliche Einträge in der `ServerName`-Direktive haben, wird automatisch das Name-Based VHosting aktiviert. Ein hinzufügen der `NameVirtualHost`-Direktive ist nicht mehr notwendig!

Listing 5.22: Der Host-Server antwortet nicht mehr!

```
Listen 10.2.11.120

NameVirtualHost "10.2.11.120:*"

<VirtualHost "10.2.11.120:*">
  DirectoryIndex error.html
  DocumentRoot   "E:/Apache/httpd/2.4.2/htdocs"
  ServerAdmin    root@10.2.11.120
  ServerName     "FEA11-119VD02.IT-TRAINING.FUS"
</VirtualHost>

```

Ein Server - mehrere Namen

Soll ein Name Based Virtual Host unter mehreren DNS-Namen verfügbar sein, kann die `ServerAlias`-Direktive genutzt werden, um alle gewünschten Namen anzugeben.

Alle Aliasnamen müssen auch auf einem DNS-Server eingetragen sein, damit der V-Host erreichbar ist!

Listing 5.23: Ein Server - mehrere Namen

```
NameVirtualHost "10.2.11.120:80"

<VirtualHost "10.2.11.120:80">
  DocumentRoot   "E:/Apache/httpd/2.4.2/htdocs/selfhtml.it-training.fus"
  ServerAdmin    root@10.2.11.120

  # Dieser Server hat zwei zusaetzliche Aliasnamen
  ServerAlias    selfhtml.it-training.de selfhtml.it-training.org
  ServerName     "selfhtml.it-training.fus"
</VirtualHost>

```

Umleitung - Redirect

Es gibt verschiedene Gründe, warum der Apache eine Nutzeranfrage auf eine andere URL umleiten muss. In unserem Falle geht es darum, dass der Standard V-Host alle Anfragen auf einen anderen virtuellen Host umleiten soll, so dass immer eine korrekte Antwort vom Webserver geliefert wird.

Dies geschieht am einfachsten mit der Direktive `Redirect`. Diese hat die folgende Syntax.

`Redirect [Status] URL-Pattern URL`

Sie nimmt drei Argumente entgegen:

- `Status`: Hierbei handelt es sich um eine HTTP-Statusmeldung, die dem Client übermittelt wird. Der Status kann sein:
  - `permanent`: Dies ist die HTTP-Statusmeldung 301, die dem Client mitteilt, dass eine Resource dauerhaft umgezogen ist.
  - `temp`: Die HTTP-Statusmeldung 302 teilt dem Client mit, dass eine Resource nur zeitweilig umgezogen ist.
  - `seeother`: HTTP-Statusmeldung Nummer 303 sagt aus, dass eine Resource dauerhaft durch eine andere ersetzt wurde.
  - `gone`: Mit der HTTP-Statusmeldung 410 wird dem Client mitgeteilt, dass eine Resource dauerhaft nicht mehr verfügbar ist. Dies ist die einzige Statusmeldung, die einen Fehler darstellt, so dass keine Weiterleitung mehr erfolgt. Bei dieser Statusmeldung muss das Argument URL entfallen!

  Wird das Argument `Status` ausgelassen, wir automatisch "temp" als Standard versandt.

- URL-Pattern Das URL-Pattern ist eine Zeichenkette, die als Vergleichswert dient. Sie wird mit der angeforderten URL verglichen. Wenn der Vergleich erfolgreich war, d. h. das Vergleichsmuster auf die URL passt, wird die Umleitung durchgeführt.
- URL Hierbei handelt es sich um die Resource, zu der hin umgeleitet werden soll.

Entwerfen Sie zu dem unten angegebenen VHost einen Default Virtual Host, der alle Anforderungen an diesen V-Host weiterleitet! Ersetzen Sie die IP-Adresse und den Hostnamen durch Ihre eigene IP-Adresse und Ihren Hostnamen! Hinweis: Der Slash (/) kann als URL-Pattern für jede beliebige Anforderung genutzt werden!

Eine sehr viel flexiblere und mächtigere Variante um Redirects zu bewerkstelligen, stellt das Apache Modul `mod_rewrite` dar, dass jedoch in der Summe seiner Möglichkeiten für diese Unterrichtsunterlage zu umfangreich ist.

Das Apache-Modul `mod_rewrite` ermöglicht ein regelbasiertes Umleiten von Client-Anforderungen auf eine andere URL. Es handelt sich hierbei um ein Modul mit umfassenden Möglichkeiten und hoher Flexibilität, dass sogar Abfragen auf externe Datenbanksystem unterstützt.

- Konfiguration des Hauptservers:
  - Die HTML-Dokumente des Hauptservers sollen für jeden Host, aus dem Verzeichnis E:\Apache\httpd\2.4.2\htdocs abrufbar sein!
  - Der Name des Hauptservers ist "wwwXX.it-training.fus", wobei XX Ihre zweistellige Platznummer darstellt (z. B. www01 oder www11)!
  - Der Server beantwortet Anfragen auf Port 80!
  - Das Error-Log heißt logs\error.log und muss stündlich ausgetauscht werden.
  - Fehler werden erst ab der Stufe "error" geloggt.
  - Es muss zusätzlich ein Access-Log im CLF-Format konfiguriert werden! Auch hier ist ein stündlicher Wechsel der Logs einzustellen (logs\access.log)!
  - Der Server muss mit verschiedenen MIME-Types umgehen können!
- Konfiguration des benannten virtuellen Servers "wwwXX.ssi.it-training.fus"
  - Die HTML-Dokumente dieses Servers sollen für jeden Host, aus dem Verzeichnis E:\Apache\httpd\2.4.2\htdocs\ssi abrufbar sein!
  - Dieser Server beantwortet Anfragen auf Port 80!
  - Das Error-Log heißt logs\ssi\error.log und muss stündlich ausgetauscht werden.
  - Fehler werden erst ab der Stufe "error" geloggt.
  - Es muss zusätzlich ein Access-Log im CLF-Format konfiguriert werden! Auch hier ist ein stündlicher Wechsel der Logs einzustellen (logs\ssi\access.log)!
  - Der Server muss mit verschiedenen MIME-Types umgehen können!
  - Die Standardwebsite dieses Servers ist index.shtml. Diese soll auch dann geladen werden, wenn der Nutzer keine explizite Angabe einer HTML-Seite gemacht hat!

- Sollte die Standardwebseite nicht geladen werden können, muss der Nutzer auf die Seite error.html umgeleitet werden. Benutzten Sie hierfür eine Standardfehlerseite, so wie Sie sie im Unterricht erstellt haben!
- Aktivieren Sie SSI für diesen Server! Hinweis: Die HTML-Seite index.shtml ist eine HTML-Seite mit SSI-Kommandos. Sie können an dieser Stelle entweder eine neue HTML-Seite entwerfen, oder eine aus den vorangegangenen Übungen nutzen!

- Konfiguration des benannten virtuellen Servers "wwwXX.selfhtml.it-training.fus":
  - Die HTML-Dokumente dieses Servers sollen für jeden Host, aus dem Verzeichnis E:\Apache\httpd\2.4.2\htdocs\selfhtml} abrufbar sein!
  - Dieser Server beantwortet Anfragen auf Port 80!
  - Das Error-Log heißt logs\selfhtml\error.log und muss stündlich ausgetauscht werden.
  - Fehler werden erst ab der Stufe "error" geloggt.
  - Es muss zusätzlich ein Access-Log im CLF-Format konfiguriert werden! Auch hier ist ein stündlicher Wechsel der Logs einzustellen (logs\selfhtml\access.log)!
  - Der Server muss mit verschiedenen MIME-Types umgehen können!
  - Die Standardwebsite dieses Servers ist index.htm. Diese soll auch dann geladen werden, wenn der Nutzer keine explizite Angabe einer HTML-Seite gemacht hat!
  - Sollte die Standardwebseite nicht geladen werden können, muss der Nutzer auf die Seite error.html umgeleitet werden. Benutzten Sie hierfür eine Standardfehlerseite, so wie Sie sie im Unterricht erstellt haben!
  - Dieser Server soll ebenfalls unter den Namen "wwwXX.selfhtml.it-training.org" und "wwwXX.selfhtml.it-training.de" verfügbar sein!

- Konfiguration des benannten virtuellen Servers "wwwXX.mediawiki.it-training.fus":
  - Führen Sie die Konfiguration dieses Servers mit Hilfe von .htaccess-Dateien durch!
  - Die HTML-Dokumente dieses Servers sollen für jeden Host, aus dem Verzeichnis E:\Apache\httpd\2.4.2\htdocs\mediawiki abrufbar sein!
  - Dieser Server beantwortet Anfragen auf Port 80!
  - Das Error-Log heißt logs\mediawiki\error.log und muss stündlich ausgetauscht werden.
  - Fehler werden erst ab der Stufe "error" geloggt.
  - Es muss zusätzlich ein Access-Log im CLF-Format konfiguriert werden! Auch hier ist ein stündlicher Wechsel der Logs einzustellen (logs\mediawiki\access.log)!
  - Der Server muss mit verschiedenen MIME-Types umgehen können!
  - Die Standardwebsite dieses Servers ist index.php. Diese soll auch dann geladen werden, wenn der Nutzer keine explizite Angabe einer HTML-Seite gemacht hat!
  - Sollte die Standardwebseite nicht geladen werden können, muss der Nutzer auf die Seite error.html umgeleitet werden. Benutzten Sie hierfür eine Standardfehlerseite, so wie Sie sie im Unterricht erstellt haben!
  - Der MediaWiki muss korrekt angezeigt werden!

Listing 5.24: Umleitung - Redirect

```
<VirtualHost "10.2.11.120:8080">
  Alias "/perl"      "E:/Apache/httpd/2.4.2/perl"
  DirectoryIndex hallo.html
  DocumentRoot "E:/Apache/httpd/2.4.2/htdocs"
  ServerName "FEA11-119VD02.IT-TRAINING.FUS"

  <Directory "htdocs">
    AddHandler application/x-httpd-php .php
    Options    Includes
    Require    all granted
  </Directory>

  <Directory "E:/Apache/httpd/2.4.2/perl">
    Options        ExecCGI
    PerlHandler    ModPerl::Registry
    PerlSendHeader On
    AddHandler     perl-script .pl
    Require        all granted
  </Directory>
</VirtualHost>

```

5.2.11 Secure Socket Layer (SSL)

SSL - Eine kurze Einleitung

Der Secure Socket Layer, kurz SSL sorgt dafür, dass Verbindungen verschlüsselt, und somit für Dritte unlesbar, aufgebaut werden können. Für viele Anwendungen im Internet, wie z. B. Internet Banking, ist diese Technologie unverzichtbar, um die Vertraulichkeit der Daten zu gewährleisten.

Die Onlinedokumentation des Apache HTTP-Servers bietet eine umfassende und informatiefe Einführung in das Thema "Verschlüsselung mit SSL". Sie finden dieses Dokument unter: http://httpd.apache.org/docs/2.4/ssl/ssl_intro.html

OpenSSL installieren

Da das SSL-Protokoll eine nahezu unüberschaubare Vielfalt an Möglichkeiten kennt, soll in dieser Unterrichtsunterlage nur eine rudimentäre Anleitung zur Konfiguration einer einfachen SSL-Verbindung gegeben werden.

Um eine SSL-Verbindung erstellen zu können, müssen im ersten Schritt Zertifikate erzeugt werden, welche dann unter anderem für die Verschlüsselung der Verbindung genutzt werden. Ein einfaches und kostenlose Mittel zur Erstellung von Zertifikaten ist das Tool: "OpenSSL", dessen Installation hier nun beschrieben wird.

Das Open Source Projekt "OpenSSL" stellt ein mit kommerziellen Produkten vergleichbares Tool Kit mit vollständiger Unterstützung der SSL-Technologie dar. Es fußt auf der von Eric A. Young und Tim J. Hudson entwickelten "SSLeay library" und wird unter einer Lizenz vertrieben, die mit der Lizenz des Apache Webservers vergleichbar ist. Dadurch ist es sowohl für kommerzielle als auch für nicht kommerzielle Projekte kostenfrei einsetzbar.

OpenSSL existiert für verschiedene Betriebssystem der Unix/Linux-Welt, und auch für Microsoft Windows. Der Download der Windowsversion kann von der folgenden URL erfolgen: http://slproweb.com/download/Win64OpenSSL_Light-1_0_1c.exe

Damit OpenSSL unter Windows installiert werden kann, müssen zusätzlich die "Microsoft Visual C++ 2008 Redistributables" heruntergeladen und installiert werden! Diese sind unter der folgenden URL zu finden: http://www.microsoft.com/de-de/download/details.aspx?id=15336

Laden Sie die "Microsoft Visual C++ 2008 Redistributables" und OpenSSL in der Version "Win64 OpenSSL v1.0.1c Light" herunter und installieren Sie beide Produkte! Hiweis: Installieren Sie OpenSSL in das Verzeichnis E:\ OpenSSL\ 1.0.1c\win64} und w"ahlen Sie die Option "Copy OpenSSL DLLs to the Windows system directory"!

Ein Zertifikat mit OpenSSL generieren

Da der Fokus dieser Unterrichtsunterlage auf der Konfiguration von SSL für den Apache HTTP-Server liegt, wird an dieser Stelle die Erstellung eines selbst-signierten Serverzertifikats beschrieben. Dies ist jedoch keine Variante für eine Produktivumgebung. Der betrieb eines SSL-Gesicherten Webservers im Internet erfordert weit mehr Kenntnisse über SSL, als an dieser Stelle vermittelt werden können.

1. Erstellen Sie die folgenden Arbeitsverzeichnisse:
   - E:\ssl
   - E:\ssl\keys
   - E:\ssl\certs
2. Laden Sie eine Kopie der Datei openssl.cfg von der folgenden URL herunter: http://www.dylanbeattie.net/docs/openssl.conf

   Achtung: Hierbei handelt es sich um eine öffentlich zugängliche und stark vereinfachte Version der Konfigurationsdatei für OpenSSL, die keinesfalls im Produktivbetrieb eingesetzt werden sollte!
3. Erstellen Sie die Datei E:\ssl\database.txt. Bei dieser Datei handelt es sich um eine leere Datei.
4. Erstellen Sie die Datei E:\ssl\serial.txt. Diese Datei enthält lediglich die Zeichenkette 01.
5. Erstellen Sie ein selbst-signiertes CA Zertifikat für Ihren Webserver, inklusive Private Key.

   openssl req -new -x509 -days 365 -sha1 -newkey rsa:1024 -nodes -keyout E:\ssl\keys\server.key -out E:\ssl\certs\server.cert -config E:\OpenSSL\1.0.1c\win64\bin\openssl.cfg}
6. Recherchieren Sie in der OpenSSL Onlinehilfe, was die einzelnen Parameter des Kommandos openssl req bedeuten! Sie finden die OpenSSL Onlinehilfe unter: http://www.openssl.org/docs/apps/req.html

Den Apache mit SSL konfigurieren - Der Hauptserver

Der erste und auch wichtigste Schritt bei der Konfiguration von SSL für den Apache HTTP-Server ist, dem Webserver mitzuteilen, dass er auf dem HTTPS-Port 443 arbeiten soll. Dies geschieht mit Hilfe der `Listen 443`-Direktive. Der zweite Schritt ist optional. Um x509- und pkcs7-Zertifikatsdateien korrekt zu unterstützen, sollte der Webserver deren MIME-Types kennen. Verwenden Sie hierzu die `AddType`-Direktive.

Während die ersten beiden Schritte für den Hauptserver zu konfigurieren sind, sind alle weiteren Schritte für jeden einzelnen V-Host durchzuführen.

Listing 5.25: Den Apache mit SSL konfigurieren - Der Hauptserver

```
AddType application/x-x509-ca-cert  .crt
AddType application/x-pkcs7-crl     .crl

```

Den Apache mit SSL konfigurieren - Die virtuellen Hosts

Jeder V-Host der mit SSL arbeiten soll benötigt mindestens drei Dinge:

- Die SSL-Engine muss eingeschaltet werden
- Ein SSL-Zertifikat muss vorhanden sein, und evlt. auch ein Privat Key-File.
- Die Angabe welche Verschlüsselungsverfahren (engl. CipherSuits) genutzt werden dürfen.

Die SSL-Konfiguration eines virtuellen Servers könnte also so aussehen, wie unten angegeben. Dieses Beispiel enth"alt jedoch nur die fürr die Nutzung von SSL absolut notwendigen Direktiven.

Die komplexeste Direktive dürfte zweifelsohne `SSLCipherSuite` sein. Sie besteht aus einer Liste verschiedener Verschlüsselungsverfahren und präfixen, die durch Doppelpunkte von einander getrennt werden. Der Client und der Webserver einigen sich bei ihrem Handshake auf eines der erlaubten Verfahren, und nutzen dies anschließend für die Verschlüsselung der Verbindung.

Die Liste kann aus vier verschiedenen Komponenten bestehen:

- Schlüsselaustauschalgorithmen
- Authentifizierungsalgorithmen
- Verschlüsselungsalgorithmen
- MAC-Digest-Algorithmen

Zusätzlich gibt es Aliasnamen, die stellvertretend für eine ganze Gruppe von Algorithmen stehen, wie z. B. ALL, HIGH, MEDIUM, LOW oder SSLv2. Die in der Liste verwendeten Präfixe bedeuten:

- Ohne Präfix : Hinzufügen des Algorithmus/der Gruppe von Algorithmen zur Liste
- +: Der Algorithmus soll an der genannten Stelle in die Liste eingefügt werden.
- -: Der Algorithmus soll aus der Liste entfernt werden, um später an einer anderen Stelle wieder eingefügt werden zu können.
- !: Unwiderruflicher Ausschluss des Algorithmus/der Gruppe von Algorithmen aus der Liste.

Ein Beispiel für die SSLCipherSuite-Direktive ist unten angegeben. Es bedeutet:

- ALL : Alle Algorithmen sind zul"assig
- !ADH : Schließe alle Algorithmen aus, die zur Gruppe der "Anonymen Diffie-Hellman-Algorithmen" gehören.
- Entferne die Gruppe der MEDIUM-Verschlüsselungsalgorithmen an dieser Stelle aus der Liste.
- Füge die Gruppe der SSLv3-Algorithmen an dieser Stelle in die Liste ein.
- Füge die Gruppe der MEDIUM-Algorithmen an der neuen Stelle wieder in die Liste ein.

Die Erstellung eines CipherSuite-Strings sollte grundsätzlich durch einen Security-Admin, und nicht durch den Webadmin selbst erfolgen!

Fertigen Sie jetzt eine Sicherheitskopie ihrer Hauptkonfigurationsdatei an! Erstellen Sie einen Webserver, der die folgenden Webauftrittebereitstellt! Alle V-Hosts haben eigene Error- und Access-Logs (CLF), die im 2 Stunden Rhythmus rotieren. Jeder V-Host stellt eine Seite dar, ohne das diese vom Nutzer angegeben werden muss (z. B. www.selfhtml.de zeigt direkt die Seite index.htm). Wenn ein Nutzer versucht ohne HTTPS-Verbindung auf einen der beiden mit HTTPS gesicherten Hosts zuzugreifen (z. B. http://selfhtml.de:443), soll er auf die Adresse https://selfhtml.de umgeleitet werden.

Listing 5.26: Den Apache mit SSL konfigurieren - Die virtuellen Hosts

```
# Die SSL-Engine einschalten
SSLEngine On

# Das Serverzertifikat + Private Key festlegen
SSLCertificateFile    "E:/ssl/certs/server.crt"
SSLCertificateKeyFile "E:/ssl/keys/server.key"
SLLCipherSuite        ALL:!ADH:-MEDIUM:+SSLv3:+MEDIUM

```

5.3 Tomcat

Apache Tomcat

5.3.1 Der Apache Tomcat - Ein Applikationsserver

Was ist ein Applikationsserver

Ein Anwendungsserver (engl. application server) ist im Allgemeinen ein Server in einem Computernetzwerk, auf dem Anwendungsprogramme ausgeführt werden. Im engeren Sinne bezeichnet der Begriff eine Software, die spezielle Dienst zur Verfügung stellt, wie beispielsweise Transaktionen, Authentifizierung oder den Zugriff auf Verzeichnisdienste und Datenbanken über definierte Schnittstellen. (Quelle: Wikipedia http://de.wikipedia.org/wiki/Anwendungsserver)

Für die meisten Webinhalte, wie z. B. HTML, PHP oder PERL ist ein Webserver absolut ausreichend. Sollen jedoch komplexe Anwendungsprogramme über ein Netzwerk verfügbar gemacht werden ändert sich dies, da hier zusätzliche Technologien wie z. B. Java oder ähnliche zum Einsatz kommen. Diese Fähigkeitslücke der Webserver schließen die Applikationsserver bzw. Anwendungsserver. Einer der Application Server ist der Apache Tomcat, der hier als stellvertretendes Beispiel für diese Gruppe der Server dienen soll.

5.3.2 Architektur des Tomcat

Architektur

Servlets

Servlets sind Java-Klassen, die nicht auf einem Client, sondern Server seitig abgearbeitet werden. Sie nehmen Anfragen von Clients entgegen und bearbeiten diese. Die Antwort eines Servlets kann, so wie dies auch bei PHP der Fall ist, eine dynamisch generierte Seite sein. Eine mit den Java Servlets vergleichbare Technologie stellen die Microsoft Active Server Pages, kurz ASP-Seiten dar.

Context

Ein Context-Element stellt ein Java-Programm dar, das aus einem oder mehreren Servlets besteht. Jeder Context besteht aus einer WAR-Datei bzw. aus einem Verzeichnis, dass alle Servlets in ihrer entpackten Form enthält.

Jedes Context-Element muss sein eigenes Verzeichnis haben! Der Name dieses Verzeichnisses wird als "Context path" bezeichnet.

Benutzen Sie Wikipedia, um herauszufinden, was eine WAR-Datei ist!

Richtet ein Client eine Anfrage an den Tomcat, wird ein Vergleich der URI mit dem Context path aller Context-Elemente durchgeführt. Es wird dann der Context ausgewählt, dessen Context path die passendste Übereinstimmung liefert.

Beispiel

Eine Anwendung wird im Verzeichnis foo abgelegt. Der daraus resultierende Context path lautet /foo. Somit würde der Aufruf für die Anwendung "foo" wie folgt lauten: http://localhost/foo.

Host

Host-Container stellen virtuelle Hosts dar. Für jeden Tomcat-Server muss mindestens ein Host-Container definiert sein, was bedeutet, dass der Tomcat grundsätzlich mit virtual Hosting arbeitet, im Gegensatz zum Apache HTTPD. Grundsätzlich gilt auch für den Tomcat, das für jeden virtuellen Host ein entsprechende DNS-Eintrag existieren muss, damit Clients diesen Host erreichen können.

Engine

Die Engine ist das zentrale Element, welches alle Clientanfragen bearbeitet. Die Kommunikation zwischen den Clients und der Engine wird mittels der Connectoren ermöglicht.

Es darf nur genau ein Engine-Element innerhalb eines Services geben!

Connector

Ein Connector stellt einen Verbindungs-Endpunkt dar, d. h. mittels eines Connectors kann ein Client eine Verbindung zum Tomcat-Server herstellen. Connectoren sind die Element, welche alle verbindungsspezifischen Attribute definieren. Hierzu gehören unter anderem auch das genutzte Protokoll, wie z. B. "HTTP/1.1" oder "AJP/1.3" und der Port (Standard ist 8080 für HTTP/1.1 und 8009 für AJP/1.3)

Server

Das Server-Element repräsentiert den Tomcat-Server selbst. Deshalb darf auch nur genau ein Server-Element, welches als äußerstes Element angegeben werden muss, existieren.

Service

Ein Service-Element stellt die Verbindung aus einem oder mehreren Connectoren und genau einer Engine dar. Es kann mehrere Service-Elemente innerhalb eines Servers geben.

Ein abschließendes Beispiel

Ein Client versucht mittels der folgenden URI eine Verbindung zu einem Server herzustellen:

http://www.google.de/search?hl=de&q=wetter&btnG=Google-Suche&meta=

1. Der Client verbindet sich mit dem Server über das TCP/IP-Protokoll.
2. Die Daten werden vom Server an den Service weitergeleitet und von dort zum spezifischen Connector, hier ist dies der HTTP/1.1-Connector.
3. An dieser Stellt wird erst die URL ausgewertet.
4. Die Engine prüft, ob der Host www.google.de existiert. Ist dem nicht so, wird die Anfrage von der Engine an den Default host delegiert. Anderen Falls wird die Anfrage an www.google.de verwiesen.
5. Im entsprechenden Host wird geprüft, welcher Context für die Abarbeitung der Anfrage zuständig ist.
6. Der Conntext prüft seinerseits, welches Servlet oder welches Skript mit welchen Parametern angesprochen werden soll. Das Skript bzw. das Servlet bearbeitet dies und gibt die Antwort wieder zurück.

5.3.3 Installation

Vorbereiten der Installation

Der Tomcat Applicationserver benötigt als Grundlage für seine Installation ein funktionsfähiges Java Runtime Environment. Diese Software kann unter dem folgenden Link heruntergeladen werden: http://www.java.com/en/download/manual_v6.jsp. Benutzen Sie zur Installation das Windows Offline Paket.

Laden Sie das Java JRE Version 6 Update 33 herunter, und installieren Sie es nach C:\Programme\Java\jre6

Durchführen der Tomcat-Installation

Herunterladen der Software

Nachdem das System durch die Installation von Java vorbereitet wurde, kann die Installation des Tomcat erfolgen.

Laden Sie Tomcat 7 in seiner neuesten Version herunter! (http://tomcat.apache.org/download-70.cgi)

Installieren des Tomcat

Installieren Sie den Tomcat in das Verzeichnis E:\apache\tomcat\x.x.x. x.x.x Stellt die Versionsnummer dar, also z. B. E:\apache\tomcat\7.0.29! Benutzen Sie für die Installation die in der folgenden Abbildung gezeigten Optionen und starten Sie den Tomcat direkt nach der Installation! Das Passwort für den Tomcat Administrator lautet: topsecret

Sobald der Tomcat installiert wurde, zeigt sich ein neues Icon in der Windows Taskleiste.

Links befindet sich der Monitor für den Apache und rechts der Monitor für den Tomcat, welcher gerade aktiv ist. Zum Testen des Tomcat öffnen Sie einen Browser und geben folgende Adresse an:

http://localhost:8080

wenn alles erfolgreich läuft erhalten Sie die Startseite des Apache Tomcat. Sollte dies nicht der Fall sein, so prüfen Sie zunächst:

- Ob Sie die richtige URL angegeben haben,
- ob der Tomcat tatsächlich läuft,
- ob der Apache HTTPD einen virtuellen Host auf Port 8080 laufen hat.

Konfigurieren eines Serviceaccounts

Finden Sie heraus welches Benutzerkonto der Tomcat-Dienst zum starten verwendet!

Ein Grundsätzliches Sicherheitsprinzip ist, dass ein Dienst nur die Berechtigungen bekommt, die er auch benötigt. So soll vermieden werden, dass bei evtl. erfolgreichen Angriffen auf einen Dienst der Angreifer größeren Schaden auf dem Server anrichten kann, wie z. B. die Installation von Schadsoftware. Um einen Serviceaccount für den Tomcat einzurichten, sind die in der folgenden Übung beschriebenen Schritte notwendig.

Erstellen Sie einen sicheren Serviceaccount für den Tomcat, mit den folgenden Angaben.

- Accountname: tomcatsrvs
- Passwort: Tom!CaT0.
- Passworteinstellungen: Kennwort läuft nie ab, Benutzer kann Kennwort nicht ändern
- Gruppenmitgliedschaft: Benutzer
- tomcatsrvs übernimmt den Besitz des Verzeichnisses E:\apache\tomcat\x.x.x.
- tomcatsrvs benötigt lesende Berechtigungen für das Verzeichnis E:\apache\tomcat\x.x.x und alle Unterverzeichnisse.
- Für die Verzeichnisse x.x.x/conf, x.x.x/logs, x.x.x/temp und x.x.x/work benötigt tomcatsrvs das die Berechtigung "Ändern".
- tomcatsrvs muss in die lokale Sicherheitsrichtlinie "Anmelden als Dienst" eingetragen werden.

Starten Sie den Tomcat mit dem neuen Serviceaccount tomcatsrvs!

5.3.4 Konfiguration des Tomcat

Die Verzeichnisstruktur

Das Installationsverzeichnis des Tomcat 7 enthält die folgenden Unterordner:

bin	Ausführbare Dateien des Tomcat
conf	Konfigurationsdateien, inklusive der Hauptkonfigurationsdatei server.xml
lib	Programmbibliotheken, die der Tomcat für seinen Betrieb benötigt
logs	Logdateien
temp	Temporäre Dateien, die bei der Ausführung von Webapplicationen entstehen
webapps	Stanardverzeichnis für Webapplicationen
work	Arbeitsverzeichnis für Webapplicationen

Die Konfigurationsdateien des Servers - server.xml

Was für den Apache HTTPD die Datei httpd.conf ist, ist die Datei server.xml. In dieser Datei erfolgt die Konfiguration der gesamten Architektur des Tomcat (Server, Services, Engine, Hosts, usw.).

Fertigen Sie eine Sicherungskopie der Datei E:\apache\tomcat\x.x.x\conf\server.xml mit dem Namen E:\apache\tomcat\x.x.x\conf\server.bakan! Löschen Sie jetzt den Inhalt der Datei E:\apache\tomcat\x.x.x\conf\server.xml!

Unten abgebildet sehen Sie eine minimale E:\apache\tomcat\x.x.x\conf\server.xml-Datei. Die XML-Tags spiegeln sehr deutlich die Architektur des Tomcat wieder. Wie bereits erwähnt steht an oberster Stelle der `<Server>`-Tag mit den beiden Attributen `port` und `shutdown`.

Benutzen Sie die Onlinedokumentation des Tomcat, um herauszufinden, was die beiden Attribute `port` und `shutdown` des `<Server>`-Tags bedeuten!

Für den Server wird ein Service mit dem Namen "Catalina" erstellt (Catalina ist der Name der Servlet-Engine), innerhalb dessen die restlichen Komponenten konfiguriert werden.

Der `<Connector>`-Tag erstellt in diesem Beispiel einen HTTP/1.1 Connector, welcher auf Port 8080 agiert und einen Connectiontimeout von 20 Sekunden benutzt.

Was bedeuten die Attribute `connectionTimeout` und `redirectPort`?

Auf der gleichen Ebene wie der `<Connector>`-Tag steht der `<Engine>`-Tag. Dieser konfiguriert die Engine, welche für die Verarbeitung der Anfragen zuständig ist. Das Attribut `name`-Attribut gibt an, welche Engine genutzt werden soll. Standardmäßig ist dies immer die "Catalina" Engine. Mittels des Attributs `defaultHost` wird festgelegt, welcher Host der Default host ist, d. h. welcher Host antwortet, falls eine Anfrage keinem expliziten Host zugeordnet werden kann.

In einem Tomcat-Server muss ein Host als "Default host" eingetragen sein, d. h. sein `name`-Attribut muss dem `defaultHost`-Attribut der Engine entsprechen. Für den Hostname des Default host muss kein DNS-Eintrag existieren, da alle Anforderungen, die keinem Host zugewiesen werden können, automatisch an den Default Host umgeleitet werden.

Der letzte, für die Minimalkonfiguration relevante Tag, ist der `<Host>`-Tag. Er definiert einen virtuellen Host.

Finden Sie heraus, was die unten angegebenen Attribute des `<Host>`-Tags bedeuten!

Listing 5.27: Die Konfigurationsdateien des Servers - server.xml

```html
<?xml version='1.0' encoding='utf-8'?>

<Server port="8005" shutdown="SHUTDOWN">
  <Service name="Catalina">

    <Connector port="8080" protocol="HTTP/1.1"
      connectionTimeout="20000"
      redirectPort="8443" />
    <Connector port="8009" protocol="AJP/1.3" redirectPort="8443" />

    <Engine name="Catalina" defaultHost="localhost">
      <Host name="localhost"  appBase="webapps"
        unpackWARs="true" autoDeploy="true">
      </Host>
    </Engine>
  </Service>
</Server>
```

5.3.5 Anwendungen bereitstellen

Automatisches Deployment

Mit Hilfe des automatischen Deployments ist es möglich, eine Webanwendung durch einfaches Kopieren in das Verzeichnis webapps auf dem Tomcat bereitzustellen.

Laden Sie sich die Datei helloApp.war herunter und kopieren Sie sie in das Verzeichnis E:\apache\tomcat\7.0.29\webapps! Beobachten Sie, was passiert! (Hinweis: Der Tomcat muss vorher gestartet werden.)

Der Aufbau einer Webanwendung

Die Verzeichnisstruktur

Die Verzeichnisstruktur einer Java-Webanwendung folgt einer strikten Vorgabe. Als Beispiel für eine solche Anwendung dient hier d Datei helloApp.war. Sie besitzt folgende Verzeichnisstruktur:

In dieser Verzeichnisstruktur sind drei Dateien enthalten:

- META-INF\MANIFEST.MF
- WEB-INF\web.xml
- WEB-INF\classes\HelloServlet.class

Die Datei HelloServlet.class ist das Servlet, dessen Funktionalität genutzt werden soll. Sie stellt in diesem Falle die gesamte Webanwendung dar, da hier nur ein einziges Servlet von Nöten ist.

Der Deployment descriptor

Der `<servlet>`-Tag

In der Datei web.xml ist der "Deployment descriptor" enthalten. Als solches werden die Angaben bezeichnet, die der Tomcat benötigt, um die Anwendung bereitzustellen. Unten ist ein einfaches Beispiel für einen Deployment descriptor angegeben. Besondere Beachtung sollte den Zeilen 9 bis 17 geschenkt werden.

Der Tag `servlet` beschreibt das Servlet an sich näher. Im Falle dieses Beispiels legt er einen Namen für das Servlet fest (Tag `servlet-name`) und gibt an, welche Java-Klasse zu Beginn geladen werden soll (Tag `servlet-class`).

Es ist wichtig, dass für jedes Servlet ein `servlet-name` angegeben wird, da andere Tags, wie z. B. der `servlet-mapping`-Tag mit Hilfe des Namens das Servlet referenzieren, um weitere Eigenschaften festzulegen!

Der Wert des Tags `servlet-class` muss dem Namen einer Java-Klassendatei entsprechen, welche sich im Verzeichnis WEB-INF\classes befindet!

Der `<servlet-mapping>`-Tag

Im `<servlet-mapping>`-Tag wird mit Hilfe des `<url-pattern>`-Tags der Anwendung ein URL-Pattern zugewiesen, durch dass sie gestartet werden kann. Die vollständige URL zum starten der Anwendung lautet:

http://localhost:8080/helloApp/hello

Die Angabe "helloApp" ist der Context path, also der Verzeichnispfad, in dem sich die Anwendung befindet (hier: E:\apache\tomcat\7.0.29\webapps\helloApp. Das "hello" ist das URL-Pattern. Durch es weiß der Tomcat, welche Java-Klasse bzw. welches Servlet gestartet werden muss.

`<display-name>` und `<description>`

Die beiden Tags `display-name` und `description` haben lediglich für die Tomcat Web Application Manager Anwendung eine Bedeutung. Diese Web-Anwendung kann mittels der URL http://localhost:8080/manager/html erreicht werden, und erlaubt einfachsts administrative Tätigkeiten, wie z. B. das Application deployment.

Zusammenfassung

Der Anwendung "helloApp.war" wird der Name "HelloServlet" zugewiesen. Die Klasse, welche die Anwendung startet, heißt HelloServlet und befindet sich in der Datei WEB-INF\classes\HelloServlet.class. Das URL-Pattern zum starten der Anwendung lautet /hello.

Listing 5.28: Der Deployment descriptor

```
1  <?xml version="1.0"?>
2
3  <web-app xmlns="http://java.sun.com/xml/ns/javaee"
4          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5          xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
6                  http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
7          version="2.5">
8
9    <display-name>Hello World Servlet</display-name>
10   <description>
11     This is a small servlet with greetings for everyone in the world.
12   </description>
13   <servlet>
14     <servlet-name>HelloServlet</servlet-name>
15     <servlet-class>HelloServlet</servlet-class>
16   </servlet>
17
18   <servlet-mapping>
19     <servlet-name>HelloServlet</servlet-name>
20     <url-pattern>/hello</url-pattern>
21   </servlet-mapping>
22 </web-app>
23
```

Zentrale Deployment descriptoren

Da nicht jede Webanwendung zwingend eine web.xml-Datei mitbringen muss, bietet der Tomcat die Möglichkeit Deployment descriptoren zentral, in der Datei E:\apache\tomcat\7.0.29\conf\web.xml zur Verfügung zu stellen.

Inhalte die in die Datei E:\apache\tomcat\7.0.29\conf\web.xml eingetragen werden stehen allen Webanwendungen zur Verfügung. Einträge in dieser zentralen Datei können durch Einträge in der web.xml-Datei einer Anwendung überschrieben werden, d. h. die web.xml-Datei einer Anwendung hat Vorrang vor der zentralen web.xml-Datei.

- Fügen Sie in die Datei E:\apache\tomcat\7.0.29\conf\web.xml die unten gezeigten Zeilen 23 bis 31 ein!
- Unter welcher URI ist die "helloApp" jetzt verfügbar?
- Löschen Sie die Datei E:\apache\tomcat\7.0.29\webapps\helloApp\WEB-INF\web.xml
- Unter welcher URI ist die "helloApp" jetzt verfügbar?

Listing 5.29: Zentrale Deployment descriptoren

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!--
  Licensed to the Apache Software Foundation (ASF) under one or more
  contributor license agreements.  See the NOTICE file distributed with
  this work for additional information regarding copyright ownership.
  The ASF licenses this file to You under the Apache License, Version 2.0
  (the "License"); you may not use this file except in compliance with
  the License.  You may obtain a copy of the License at

      http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License.
-->
<web-app xmlns="http://java.sun.com/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
                      http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
  version="3.0">
  <servlet>
    <servlet-name>HelloServlet</servlet-name>
    <servlet-class>HelloServlet</servlet-class>
  </servlet>

  <servlet-mapping>
    <servlet-name>HelloServlet</servlet-name>
    <url-pattern>/app</url-pattern>
  </servlet-mapping>
```

Ein eigener Context

Ein Context-Element stellt ein Java-Programm dar, das aus einem oder mehreren Servlets besteht.

Wird eine Web-Anwendung mittels des automatischen Anwendungs-Deployments bereitgestellt, erstellt der Tomcat auch automatisch den hierfür benötigten Kontext. Falls es notwendig ist, die Parameter eines Kontextes zu beeinflussen, muss jedoch ein eigener Kontext erstellt werden. Es gibt drei Möglichkeiten einen Kontext zu erstellen:

- Durch eine xml-Datei im META-INF-Verzeichnis der Web-Anwendung.
- Mit Hilfe einer xml-Datei im Verzeichnis E:\apache\tomcat\7.0.29\conf\[enginename]\[hostname]\
- Mittels eines `<Context>`-Tags in der server.xml-Datei

Es wird davon abgeraten einen Kontext direkt in der Datei server.xml anzulegen, da eine Änderung eines solchen Kontextes nur mittels eines Neustarts des Tomcats erreicht werden kann.

Im folgenden Beispiel muss für die "HelloApp"-Anwendung ein eigener Kontext erstellt werden, da diese nicht im Verzeichnis E:\apache\tomcat\7.0.29\webapps, sondern im Verzeichnis E:\webapps zur Verfügung gestellt werden soll.

- Legen Sie das Verzeichnis E:\webapps\helloApp an!
- Kopieren Sie die Datei helloApp.war in dieses neue Verzeichnis!
- Legen Sie die Datei E:\apache\tomcat\7.0.29\conf\catalina\localhost\helloApp.xml! Der Inhalt dieser Datei ist unten zu sehen.
- Welche Bedeutung hat das `docBase`-Attribut?
- Unter welcher URI ist die Anwendung "helloApp" erreichbar, wenn sich der Context path am Dateinamen der soeben angelegten Kontext-Datei orientiert?

Listing 5.30: Ein eigener Context

```
<Context docBase="E:\webapps\helloApp\helloApp.war"/>
```

5.3.6 Virtual Hosting

Virtuelle Hosts erstellen

Genau wie der Apache HTTPD kennt auch der Tomcat die Möglichkeit virtuelle Hosts zu erstellen. Auch der Hintergrund, warum virtuelle Hosts genutzt werden bleibt gleich. Es sollen unterschiedliche Web-Anwendungen getrennt auf ein und dem selben Server zur Verfügung gestellt werden. Um einen neuen VHost zu erstellen, muss ein Eintrag in die server.xml-Datei gemacht werden.

- Legen Sie die im folgenden abgebildete Verzeichnisstruktur auf Laufwerk E: an!

- Kopieren Sie die Datei helloApp.war in das Verzeichnis E:\hello.de\helloApp!
- Legen Sie den unten abgebildeten virtuellen Host in der Datei server.xml an! Beachten Sie dabei, das der `<Host>`-Tag zwischen die `<Engine>`-Tags platziert werden muss!
- Was bedeuten die einzelnen Attribute dieses `<Host>`-Tags?
- Sorgen Sie dafür, dass Sie mit Hilfe Ihres Browsers auf den Host www.hello.de zugreifen können
- Starten Sie den Tomcat neu und greifen Sie auf die "helloApp"-Anwendung zu! Unter welcher URI ist diese Verfügbar?

Listing 5.31: Virtuelle Hosts erstellen

```
<Host name="www.hello.de" appBase="E:\hello.de\helloApp"
      unpackWARs="false" autoDeploy="false"
      deployOnStartup="true">
</Host>

```

Übung - Virtuelle Hosts

- Starten Sie Ihren Tomcat-Server, falls dies noch nicht geschehen ist!
- Wo kann ersehen werden, welche Web-Applikationen aktuell durch Ihren Tomcat zur Verfügung gestellt wurden?
- Stellen Sie die Beispielanwendungen (examples) über einen eigenen virtuellen Host www.example.com zur Verfügung. Dieser Host darf WAR-Dateien nicht entpacken, und diese nur während des Startup verteilen. Die appBase ist E:\webapps\examples!
- Stellen Sie die helloApp über einen eigenen virtuellen Host www.hello.de zur Verfügung. Dieser Host darf WAR-Dateien nicht entpacken, und diese nur während des Startup verteilen. Die appBase ist E:\webapps\helloApp!

Übung - Virtuelle Hosts

- Starten Sie Ihren Tomcat-Server, falls dies noch nicht geschehen ist!
- Wo kann ersehen werden, welche Web-Applikationen aktuell durch Ihren Tomcat zur Verfügung gestellt wurden?
- Stellen Sie die Beispielanwendungen (examples) über einen eigenen virtuellen Host www.example.com zur Verfügung. Dieser Host darf WAR-Dateien nicht entpacken, und diese nur während des Startup verteilen. Die appBase ist E:\webapps\examples!

5.3.7 HTTPD to Tomcat interconnect

Apache HTTPD und Tomcat verbinden

Anwendungsserver, wie z. B. der Apache Tomcat, und Webserver haben unterschiedliche Aufgaben. Während Webserver für die Verteilung von statischen Webseiten, und Webseiten deren Inhalt mittels Skriptsprachen, wie z. B. PHP generiert wird zuständig sind, kommen Anwendungsserver immer dann ins Spiel, wenn Anwendungen die mit höheren Programmiersprachen erstellt wurden, über den Browser verteilt werden sollen. Da Webserver meist sehr viel flexibler und performanter bei der Abarbeitung von Anfragen sind, bieten Anwendungsserver oftmals die Möglichkeit, sich mit einem Webserver zu verbinden, und diesen quasi als "Frontserver" zu benutzen. Das bedeutet, dass der Webserver für das Entgegennehmen der Anfragen und das Versenden der Antworten zuständig ist, während der Anwendungsserver nur noch im Hintergrund, als Container für die eigentlichen Anwendungen dient.

Das Apache JServ Protocol - AJP

Um den Apache HTTPD und den Tomcat-Server zu verbinden, wurde eigens das Apache JServ Protocol, kurz AJP geschaffen. Hierbei handelt es sich um ein für diesen Zweck optimiertes Binärprotokoll. Es kann dem Apache HTTPD in Form eines Moduls zur Verfügung gestellt werden.

- Laden Sie die Datei tomcat-connectors-1.2.37-windows-x86_64-httpd-2.4.x.zip herunter!
- Entpacken Sie die Datei mod_jk.so" in das Modulverzeichnis des Apache HTTPD (E:\apache\httpd\2.4.2\modules
- Sorgen Sie dafür, dass das `jk_module` beim starten des Apache HTTPD geladen wird!

Worker-Threads machen die Arbeit

Die eigentliche Arbeit auf der Verbindungsstrecke zwischen dem Apache HTTPD und dem Tomcat wird durch sogenannte "Worker" getätigt. Hierbei handelt es sich um Instanzen des Tomcat, welche auf Servlet-Anfragen des Apache HTTPD reagieren und diese bearbeiten.

Worker werden nicht automatische beim Herstellen der Verbindung durch mod_jk erstellt, sondern müssen vorher konfiguriert werden. Hierzu genügt eine einfache Textdatei, die normalerweise den Namen workers.properties trägt. Der Name ist jedoch frei wählbar. Unten ist eine solche Datei abgebildet.

In Zeile zwei definiert die Anweisung `worker.list` die Namen der gewünschten Worker. In diesem Beispiel wird nur ein Worker, mit dem Namen "workaholic" erstellt. Werden mehrere Worker benötigt, müssen deren Namen durch Kommata getrennt werden, also z. B. `worker.list=workaholic,arbeitstier,esel`.

Mit der Anweisugn `worker.workaholic.type` in Zeile fünf wie die Protokollversion festgelegt, welche der Worker "workaholic" benutzen soll. Es gibt folgende Möglichkeiten:

- ajp12: Das Apache JServ Protokoll in der Version 1.2
- ajp13: Das Apache JServ Protokoll in der Version 1.3
- lb: Ein Load-Balancing Protokoll
- status: Das Status-Protokoll zur Verwaltung von Load-Balancern

In den Zeilen neuen und zehn werden IP-Adresse und Port des Tomcat-Servers angegeben, mit dem der Worker "workaholic" kommunizieren soll. In diesem Fall, befindet sich der Tomcat-Server auf dem gleichen Rechner, wie der HTTPD.

- Erstellen Sie im Verzeichnis E:/apache/httpd/2.4.2/conf die Datei workers.properties, mit dem unten gezeigten Inhalt!
- Laden Sie die Datei greetings.war herunter und stellen Sie sie auf dem Standardhost Ihres Tomcat-Server zur Verfügung!

Damit der Apache HTTPD weiß, welche Worker zur Verfügung stehen, muss die workers.properties-Datei noch eingebunden werden. Dies geschieht mit der Direktive `JkWorkersFile "E:/apache/httpd/2.4.2/conf/workers.properties"`.

Binden Sie Ihre workers.properties-Datei in Ihren HTTPD ein!

Listing 5.32: Worker-Threads machen die Arbeit

```
# Eine Liste der definierten Worker
worker.list=workaholic

# Das von Worker "Workaholic" benutzte Protokoll
worker.workaholic.type=ajp13

# Host und Port, den Workaholic benutzt, um mit einem
# Tomcat zu kommunizieren
worker.workaholic.host=127.0.0.1
worker.workaholic.port=8009
```

Das URI-Mapping

Mit Hilfe der URI http://localhost:8080/greetings kann die neue Web-Anwendung direkt über den Tomcat aufgerufen werden. Damit diese Aufruf auch mittels des HTTPD möglich ist, muss der Kontext `greetings` dem Webserver bekannt gemacht werden. Hierfür wird eine URI-Worker-Map angelegt. Es handelt sich dabei um eine einfache Textdatei, die eine Zuweisung zwischen einem Kontext und einem Worker macht.

Da die URI-Worker-Map für den HTTPD benötigt wird, sollte diese, genauso wie die workers.properties, im conf-Ordner des HTTPD gespeichert werden. Ihr Standardname lautet uriworkermap.properties! Damit der HTTPD diese Datei nutzen kann, muss sie mittels der Direktive `JkMountFile "E:/apache/httpd/2.4.2/conf/uriworkermap.properties` eingebunden werden.

Legen Sie die Datei uriworkermap.properties im Order E:/apache/httpd/2.4.2/conf an!

Listing 5.33: Das URI-Mapping

```
# Der Kontext /greetings wird dem Worker workaholic zugewiesen
/greetings=workaholic

# Alle Dateien unterhalb des Context path /greetings werden ebenfalls
# dem Worker workaholic zugewiesen
/greetings/*=workaholic
```

Zusammenfassung

Unten sind die für den HTTPD notwendigen Konfigurationsschritte noch einmal kurz zusammengefasst. Die beiden Direktiven `JkLogFile` und `JkLogLevel` sind nicht zwingend erforderlich. Sie sorgen jedoch dafür, dass ein LogFile für die AJP-Verbindung angelegt wird, welches bei der Fehlersuche sehr nützlich sein kann.

Listing 5.34: Zusammenfassung

```
CustomLog     "logs/greetings.log" common
DocumentRoot  "htdocs"
JkLogFile     "E:/apache/httpd/2.4.2/logs/mod_jk.log"
JkLogLevel    info
JkMountFile   "E:/apache/httpd/2.4.2/conf/uriworkermap.properties"
JkWorkersFile "E:/apache/httpd/2.4.2/conf/workers.properties"
LogFormat     "%h %l %u %t "%r" %>s %b" common
ServerName    "localhost"

```

5.4 Internet Information Services

5.4.1 Herkunft und Installation der IIS

Herkunft

Die Internet Information Services (kurz IIS) sind eine Entwicklung der Firma Microsoft, für deren Windows Produktpalette. Sie stellen einen Webserver Dienst mit vielen modularen Zusatzfunktionen dar, ähnlich dem marktführenden Konkurrenten Apache.

Die Entwicklung der IIS begann mit Microsoft Windows NT 3.51.

Microsoft Windows NT 3.51	IIS 1.0	Als freies Add-On
Microsoft Windows NT 4.0	IIS 2.0	
Microsoft Windows NT 4.0	IIS 3.0	Service Pack 3
Microsoft Windows NT 4.0	IIS 4.0	Option Pack
Windows 2000	IIS 5.0	
Windows XP	IIS 5.1	Professional + Media Center Edition
Windows Server 2003	IIS 6.0	
Windows Server 2008 / Vista	IIS 7.0	
Windows Server 2008 R2	IIS 7.5	Auch auf Windows 7 verfügbar

Die IIS sind nicht standardmässig installiert. Sie müssen als Zusatzoption nachinstalliert werden!

Nutzen Sie Wikipedia, um herauszufinden, wer die IIS entwickelt hat bzw. wo sie ursprünglich entwickelt wurden!

Die IIS sind in der Lage, nicht nur HTML-Seiten auszuliefern, sondern sie stellen auch noch viele andere Dienste zur Verfügung:

- HTTP
- FTP
- SMTP
- NNTP
- WebDAV
- IIS Admin Service

Bei der Installation der IIS werden nur der HTTP-Dienst und die IIS Admin Services installiert. Alle anderen Dienst müssen nachinstalliert bzw. aktiviert werden.

Alle Dienste, die die IIS zur Verfügung stellen können, sind in sogenannten ISAPI-DLL Modulen verpackt. Diese können nach bedarf hinzugefügt oder entfernt werden.

Installation

Es gibt drei Möglichkeiten die IIS zu installieren:

1. Die Installation über den Server-Konfigurationsassistenten
2. Die Installation mit Hilfe der Systemsteuerung
3. Die unbeaufsichtigte Installation

Microsoft empfiehlt aus Sicherheitsgründen die Installation der IIS nur auf einem NTFS-formatierten Laufwerk vorzunehmen, da dieses Dateisystem mehr Sicherheit bietet, als FAT bzw. FAT32.

Installation über den Server-Konfigurationsassistenten

1. Klicken Sie im Startmenü auf Serververwaltung.

2. Klicken Sie unter Serverfunktionen verwalten auf Funktion hinzufügen oder entfernen.

3. Klicken Sie auf Weiter.

4. Klicken Sie unter Serverfunktion auf Anwendungsserver (IIS, ASP.NET) und dann auf Weiter.

5. Wählen Sie zusätzliche Anwendungsserveroptionen aus, falls Sie diese benötigen und klicken dann auf Weiter.
6. Klicken Sie auf Weiter.
7. Klicken Sie auf Fertigstellen

Die Internet Information Services werden installiert. Nach der Installation kann das IIS Admin Tool im Windows Startmenü aufgerufen werden.

Bei dieser Installationsmethode wird ASP.NET automatisch aktiviert!

Installation mit Hilfe der Systemsteuerung

1. Klicken Sie im Startmenü auf Systemsteuerung.
2. Klicken Sie auf Software.
3. Klicken Sie auf Windows-Komponenten hinzufügen/entfernen.
4. Klicken Sie im Listenfeld Komponenten auf Anwendungsserver.
5. Klicken Sie auf Details.
6. Klicken Sie auf Internetinformationsdienste-Manager.
7. Klicken Sie auf Details, um eine Liste der optionalen IIS-Komponenten anzuzeigen. Wählen Sie die benötigten Komponenten aus, um diese zu installieren.
8. Klicken Sie auf OK, bis Sie zum Assistenten für Windows-Komponenten zurückkehren.
9. Klicken Sie auf Weiter, um den Assistenten für Windows-Komponenten abzuschließen.

Wichtige Verzeichnisse der IIS

Während der Installation der IIS werden die folgenden Verzeichnisse angelegt:

- \InetPub
- %SYSTEMROOT%\Help\IIsHelp
- %SYSTEMROOT%\System32\InetSrv
- %SYSTEMROOT%\System32\InetSrv\MetaBack

Diese Verzeichnisse enthalten Benutzerdaten und können nicht verschoben werden.

Bei der Deinstallation von IIS wird das Verzeichnis IISHelp entfernt. Die Verzeichnisse InetPub und InetSrv bleiben auf dem Computer.

Optionale Komponenten

Die IIS umfassen optionale Komponenten die jederzeit nachinstalliert werden können. Die kann auf unterschiedlichen Wegen geschehen:

- Über die Option Windows Komponenten hinzufügen/entfernen
- Im IIS Admin Tool

Hinzufügen optionaler Komponenten

Benutzen Sie das Microsoft Technet, um herauszufinden, wie optionale Komponenten nachinstalliert bzw. deinstalliert werden können! Der folgende Link führt sie zum Technet: http://technet.microsoft.com/de-de/library/default.aspx

5.4.2 Datensicherung

Sichern und Wiederherstellen der Metabase

Die Metabase enthält eine hierarchische Struktur, die die Konfiguration der IIS speichert. Seit Version 6 Sie besteht aus zwei XML-Dateien: MetaBase.XML und MBSchema.XML. Beide sind im Verzeichnis %SYSTEMROOT%\System32\InetSrv abgelegt. Nur Mitglieder der Gruppe der Administratoren können diese Dateien verändern.

Werden die IIS gestartet, wird der Inhalt der Datei MetaBase.XML in einen Bereich des Arbeitsspeichers kopiert, der als "In-Memory Metabase" bezeichnet wird. Die Metabase besteht somit insgesamt aus den drei Teilen:

- MetaBase.XML
- MBSchema.XML
- In-Memory Metabase

Werden Veränderungen an der Konfiguration der IIS vorgenommen, werden diese zuerst in der In-Memory MetaBase zwischengespeichert und anschließend persistent in die XML-Dateien übertragen. Diese Übertragung findet in bestimmten Zeitintervallen statt.

IIS-Admins sollten in regelmäßigen Zeitabständen eine Sicherung der Metabase durchführen, so dass ein Verlust der aktuellen Konfiguration ausgeschlossen ist. Die erste automatische Sicherung der MetaBase erfolgt direkt im Anschluss an die Installation der IIS. Diese Sicherung ist notwendig, damit der Urzustand der IIS-Installation bei bedarf wiederhergestellt werden kann.

Die Metabase kann mit Hilfe des IIS Admin Tools oder eines Skripts gesichert werden. Beim Sichern werden Kopien der beiden XML-Dateien MetaBase.XML und MBSchema.XML erstellt. Jede Sicherung kann bzw. sollte mit einem Passwort versehen werden.

Die Sicherungskopien der MetaBase werden im Verzeichnis %SYSTEMROOT%\System32\InetSrv\MetaBack gespeichert.

Sichern der Metabase

1. Starten Sie das IIS Admin Tool im Windows Startmenü.
2. Klicken Sie mit der rechten Maustaste auf den lokalen Computer.
3. Wählen Sie im Kontextmenü Alle Aufgaben aus.
4. Klicken Sie auf Konfiguration Sichern Wiederherstellen.

5. Klicken Sie auf Sicherungskopie erstellen.

6. Geben Sie der Sicherungskopie einen Namen und ein Passwort.

7. Klicken Sie auf OK.

Wiederherstellen der Metabase

1. Starten Sie das IIS Admin Tool im Windows Startmenü.
2. Klicken Sie mit der rechten Maustaste auf den lokalen Computer.
3. Wählen Sie im Kontextmenü Alle Aufgaben aus.
4. Klicken Sie auf Konfiguration Sichern/Wiederherstellen.

5. Klicken Sie im Listenfeld Sicherungskopien auf die automatische Sicherungs-kopie, die Sie wiederherstellen möchten.

6. Klicken Sie auf Wiederherstellen.

7. Geben Sie das Passwort ein und klicken Sie auf OK.

8. Klicken Sie auf OK und der Vorgang wird beendet.

9. Klicken Sie auf Schließen

5.4.3 Remoteverwaltung

Einrichten der Remoteverwaltung

Benutzen Sie das Microsoft Technet, um herauszufinden, wie die Remoteverwaltung der IIS per IIS Admin Tool und HTTP konfiguriert werden kann! Der folgende Link führt sie zum Technet: http://technet.microsoft.com/de-de/library/default.aspx

5.4.4 Bereitstellen von Webseiten

Die Standardwebsite

Das Bereitstellen von Websites durch die Internet Information Services geschieht in zwei Schritten:

- Erstellen der Website
- Erstellen einer Konfiguration für die IIS, zum Bereitstellen der Website.

An dieser Stelle wird nur der zweite Aufzählungspunkt behandelt, das Erstellen der Konfiguration für die Bereitstellung der Website durch die IIS.

Nach der Installation stellen die IIS automatisch eine Website dar, die Standardwebsite. Diese besteht aus zwei Dateien, die sich im Verzeichnis InetPub\wwwroot befinden. Sie heißen:

- iisstart.htm
- pagerror.gif

Diese Seite zeigt die Meldung an, das auf diesem Webserver noch keine Standardwebseite eingerichtet wurde, bzw. das diese sich noch in Bearbeitung befindet.

Um eine eigene Standardwebsite für Ihren Webserver zu konfigurieren, unternehmen Sie folgende Schritte:

1. Öffnen Sie die IIS Admin Tools
2. Erweitern Sie die Hierarchie in der linken Fensterhälfte: Internetinformationsdienste → (lokaler Computer) → Websites → Standardwebsite
3. Klicken Sie mit der rechten Maustaste auf Standardwebsite und öffnen Sie die Eigenschaften der Standardwebsite.
4. Wählen Sie die Registerkarte "Dokumente".

5. Setzen Sie den Haken bei "Standardinhaltseite anzeigen", falls dies nicht bereits geschehen ist.
6. Benutzen Sie die Schaltfläche "Entfernen", um alle nicht benötigten Dateinamen aus der Liste zu entfernen.
7. Fügen Sie, mit Hilfe der Schaltfläche "Hinzuf"ugen" den Dateinamen Ihrer Standardwebsite hinzu.
8. Klicken Sie auf "OK".
9. Testen Sie mit dem Browser Ihrer Wahl, ob die Änderung funktioniert.

Erstellen neuer Websites

1. Öffnen Sie die IIS Admin Tools
2. Um Ressourcenkonflikte zu vermeiden, beenden Sie die Standardwebseite
3. Klicken Sie mit der rechten Maustaste auf den Ordner "Websites" um das Kontextmenü zu öffnen, und klicken Sie auf die Eigenschaft "Neu" â†’ "Website...".

4. Klicken Sie auf "Weiter"

5. Geben Sie einen Namen für Ihre Website an. Dieser wird nur innerhalb der Admin Tools angezeigt und klicken Sie auf "Weiter".

6. Geben Sie das Verzeichnis an, in dem Ihre Website erstellt werden soll und klicken Sie auf "Weiter".

7. Legen Sie die Websiteberechtigungen für Ihre Website fest und klicken Sie auf "Weiter".

8. Klicken Sie auf "Fertigstellen".
9. Klicken Sie mit der rechten Maustaste auf den Ordner "Websites" um das Kontextmenü zu öffnen, und klicken Sie auf "Eigenschaftten"

10. Öffnen Sie die Registerkarte "Dokumenten" und entfernen Sie alle Einträge aus der Liste der Standardinhaltsseiten.

11. Fügen Sie die Seite "leistungsnachweis.html" als neue Standardinhaltsseite hinzu.

12. Klicken Sie auf "OK".

Virtuelle Verzeichnisse

Mit Hilfe eines Webservers können nicht nur Websites angezeigt werden, sondern auch Dateien jeglicher Art veröffentlicht werden. Diese müssen sich dazu entweder im Stammverzeichnis der Website, oder in einem virtuellen Verzeichnis befinden.

Virtuelle Verzeichnisse können sein:

- Ein beliebiges Verzeichnis auf dem Webserver, außerhalb des Stammverzeichnisses Ihrer Website.
- Eine Windowsfreigabe auf einem Dateiserver in Ihrem Netzwerk.

Ein virtuelles Verzeichnis ist ein Verzeichnis, das nicht im Basisverzeichnis enthalten ist, das jedoch Clientbrowsern so angezeigt wird, als ob dies der Fall wäre.

Virtuelle Verzeichnisse können auf zwei unterschiedliche Arten angelegt werden:

- Mit Hilfe der IIS Admin Tools
- Mit dem Windows Explorer

Um ein Virtuelles Verzeichnis mit Hilfe der IIS Admin Tools anzulegen...

1. Öffnen Sie die IIS Admin Tools
2. Erweitern Sie die Hierarchie in der linken Fensterhälfte: Internetinformationsdienste → (lokaler Computer) → Websites
3. Klicken Sie mit der rechten Maustaste auf die Website, der Sie ein virtuelles Verzeichnis hinzufügen wollen.
4. Wählen Sie im Kontextmenü den Punkt Neu → Virtuelles Verzeichnis. Der Assistent zum Erstellen von virtuellen Verzeichnissen öffnet sich.

5. Klicken Sie auf "Weiter".
6. Geben Sie im Feld Alias einen kurzen, einprägsamen Namen für Ihr Verzeichnis an.

7. Klicken Sie auf "Weiter".
8. Geben Sie im Feld Pfad das physikalische Verzeichnis an, auf das Ihr virtuelles Verzeichnis verweist.

9. Klicken Sie auf "Weiter".
10. Legen Sie die Berechtigungen fest, die für Ihr Verzeichnis gelten sollen.

11. Klicken Sie auf "Weiter".
12. Klicken Sie auf "Fertigstellen", um den Vorgang zu beenden.

Finden Sie, mit Hilfe des Microsoft Technet, heraus, wie Sie im Windows Explorer Virtuelle Verzeichnisse anlegen können! Der folgende Link führt sie zum Technet: http://technet.microsoft.com/de-de/library/default.aspx

5.4.5 Authentifizierung und Autorisierung

Anonyme Authentifizierung

Die Internet Information Services können so konfiguriert werden, dass jeder Nutzer zuerst einen Nutzernamen und ein Passwort eingeben muss, bevor er zugriff auf eine bestimmte Website erhält. Dies wird als Authentifizierung bezeichnet. Es stehen mehrere Authentifizierungsverfahren zur Verfügung:

- Anonyme Authentifizierung
- Standardauthentifizierung
- Digestauthentifizierung
- Erweiterte Digestauthentifizierung
- Integrierte Windows-Authentifizierung
- UNC-Authentifizierung
- .NET Passport-Authentifizierung
- Authentifizierung von FTP-Sites

Mit Hilfe der anonymen Authentifizierung können Nutzer, ohne die Angabe von Nutzername und Passwort, auf öffentliche Bereiche einer Website zugriff erhalten. Für dieses Verfahren wird auf dem Webserver das Nutzerkonto

"IUSR_Computername" genutzt, das für das Setup zur Gruppe "Gäste" hinzugefügt wird. Mit Ausnahme des Kontos "Gast", haben Mitglieder der Gruppe "Gäste" die gleichen Rechte, wie Die der Gruppe "Benutzer".

Um die Anonyme Authentifizierung zu aktivieren, gehen Sie wie folgt vor:

1. Wählen Sie im IIS-Manager das gewünschte Objekt, aus für das die anonyme Authentifizierung aktiviert werden soll (z. B. ein virtuelles Verzeichnis oder eine Website).

2. Öffnen Sie mit der rechten Maustaste das Kontextmenü des betreffenden Objekts und wählen Sie den Punkt "Eigenschaften".
3. Klicken Sie auf die Registerkarte "Verzeichnissicherheit" bzw. "Dateisicherheit".

4. Klicken Sie unter "Authentifizierung und Zugriffssteuerung" auf die Schaltfläche bearbeiten.
5. Aktivieren Sie das Häkchen für die anonyme Authentifizierung und tragen Sie ein entsprechendes Benutzerkonto ein.

6. Klicken Sie auf "OK".
7. Klicken Sie auf "OK".

Standardauthentifizierung

Die Standardauthentifizierung stelle eine Methode dar, bei der Nutzernamen und Passwörter unverschlüsselt über das Netzwerk übertragen werden. Es ist jedoch möglich, die Standardauthentifizierung in Zusammenhang mit den Verschlüsselungstechniken des Webservers zu nutzen, um so die Sicherheit zu erhöhen.

Wird für eine Website die Standardauthentifizierung aktiviert, so müssen auch entsprechende Windowsbenutzerkonten erstellt und NTFS-Rechte gesetzt werden.

1. Wählen Sie im IIS-Manager das gewünschte Objekt, aus für das die Standardauthentifizierung aktiviert werden soll (z. B. ein virtuelles Verzeichnis oder eine Website).
2. Öffnen Sie mit der rechten Maustaste das Kontextmenü des betreffenden Objekts und wählen Sie den Punkt "Eigenschaften".
3. Klicken Sie auf die Registerkarte "Verzeichnissicherheit" bzw. "Dateisicherheit".
4. Klicken Sie unter "Authentifizierung und Zugriffssteuerung" auf die Schaltfläche bearbeiten.
5. Deaktivieren Sie das Häkchen für die anonyme Authentifizierung
6. Aktivieren Sie das Häkchen für die Standardauthentifizierung.
7. Da diese Methode, wie bereits erwähnt, die Nutzernamen und Passwörter im Klartext verschickt, erscheint an dieser Stelle eine Warnmeldung, die Sie auf diesen Umstand hinweist. Klicken Sie auf JA.

8. Geben Sie im Feld Standarddomäne ein Standarddämonensuffix an. Dieses wird immer dann verwendet, wenn der Nutzer bei der Authentifizierung kein Domänensuffix angibt.

Beispiel 1: Als Standarddomäne wird "it-training.fus" eingetragen. Der Nutzer gibt als Nutzername den Wert "cMueller" ein. Sein Nutzername wird durch die Standarddomäne in "cMueller@it-training.fus" ergänzt.

Beispiel 2: Als Standarddomäne wird "it-training.fus" eingetragen. Der Nutzer gibt als Nutzername den Wert "cMueller@FueUstgSBw.bw" ein. Es erfolgt keine Ergänzung, da der Nutzername bereits ein Domänensuffix enthält.

9. Klicken Sie auf OK.
10. Klicken Sie auf OK.

Welche Aufgabe hat das Textfeld "Bereich" im Dialogfeld "Authentifizierungsmethoden"? Was passiert mit dort eingetragenen Werten?

Außer der Tatsache, das Nutzernamen und Passwörter unverschlüsselt übertragen werden, stellt auch der Tokencache in Problem dar. Die Aufgabe des Tokencaches ist es, Anmeldeinformationen zwischenzuspeichern. D. h. hat sich ein Nutzer einmal am Webserver angemeldet, wird dessen Anmeldeinformation, das sog. Benutzertoken, im Tokencache

zwischengespeichert. Dies sorgt dafür, das ein einmal angemeldeter Nutzer sich nicht bei jeder weiteren Webseite, die eine Authentifizierung fordert, erneut anmelden muss.

Problematisch ist, dass ein Angreifer diesen Tokencache nutzen könnte, um sich unberechtigten Zugriff zum Webserver zu verschaffen. Dies kann jedoch durch die richtige Konfiguration des Tokencaches verhindert, bzw. das Risiko kann verringert, werden.

Die Einstellungen für den Tokencache werden in der Windowsregistrierungsdatei vorgenommen.

1. Öffnen Sie den Windows Registrierungseditor Regedit32.exe
2. Wählen Sie den Registrierungsschlüssel \\HKEY_LOCAL_MACHINE\System\CurrentControlSet\ Services\InetInfo\ Parameters.
3. Klicken Sie auf das Menü "Bearbeiten" und wählen Sie den Menüpunkt "Neu".
4. Wählen Sie den Typ "DWORD-Wert"

Tokencache

5. Geben als Name "UserTokenTTL" ein.
6. Doppelklicken Sie auf "UserTokenTTL" und geben Sie als Wert eine Zahl ein. Diese legt die Sekunden fest, wie lange ein Benutzertoken gültig ist, bzw. wann es aus dem Cache entfernt wird.

Tokencache

7. Schließen Sie den Registrierungseditor.
8. Starten Sie die IIS neu.

Integrierte Windows-Authentifizierung

Im Gegensatz zur Standard-/Basisauthentifizierung stellt die Integrierte Windows-Authentifizierung eine sogenannte "sichere Authentifizierungsmethode" dar. Sicher heißt in diesem Falle, das kein Nutzername und kein Passwort über das Netzwerk übertragen werden, sondern nur ein Hashwert. Vormals wurde diese Art der Authentifizierung auch als NTLM-Authentifizierung bezeichnet.

1. Wählen Sie im IIS-Manager das gewünschte Objekt, aus für das die Integrierte Windows-Authentifizierung aktiviert werden soll (z. B. ein virtuelles Verzeichnis oder eine Website).
2. Öffnen Sie mit der rechten Maustaste das Kontextmenü des betreffenden Objekts und wählen Sie den Punkt "Eigenschaften".
3. Klicken Sie auf die Registerkarte "Verzeichnissicherheit" bzw. "Dateisicherheit".
4. Klicken Sie unter "Authentifizierung und Zugriffssteuerung" auf die Schaltfläche bearbeiten.
5. Deaktivieren Sie das Häkchen für die anonyme Authentifizierung
6. Aktivieren Sie das Häkchen für die integrierte Windows-Authentifizierung.
7. Klicken Sie auf "OK".
8. Klicken Sie auf "OK".

Autorisierung

NTFS-Berechtigungen einsetzen

Welche Empfehlungen gelten beim Sichern von Dateien mit NTFS-Berechtigungen? Ermitteln Sie dies mit Hilfe des Microsoft TechNet!

1. Wählen Sie im IIS-Manager die Website/Datei aus die Sie sichern möchten.
2. Klicken Sie mit der rechten Maustaste und wählen Sie im Kontextmenü den Punkt "Berechtigungen".
3. Vergeben Sie, die entsprechenden Berechtigungen.
4. Klicken Sie auf "OK".

Benutzung von Websiteberechtigungen

Websiteberechtigungen sind nur als Ergänzung zu den NTFS-Berechtigungen zu sehen, nicht aber als alleiniges Mittel, um eine Website zu sichern. Im Gegensatz zu NTFS-Berechtigungen gelten Websiteberechtigungen unabhängig von Nutzerkonten, also immer für alle Nutzer.

rmitteln Sie, mit Hilfe des Microsoft TechNet, welche Regeln gelten, wenn Sie NTFS-Berechtigungen zusammen mit Websiteberechtigungen einsetzen und wie Sie Websiteberechtigungen einrichten!

Ermöglichen von anonymen Zugriffen

Anonymer Zugriff ist die am häufigsten verwendete Zugriffsform. Hiermit kann ein Nutzer ohne weitere Authentifizierung auf eine Website zugreifen. Wie diese Art der Authentifizierung konfiguriert wird, wurde bereits im Vorfeld beschrieben. Genutzt wird sie, in dem der gewünschten Website die NTFS-Berechtigungen:

- Lesen und Ausführen
- Ordnerinhalt auflisten
- Lesen

zugewiesen werden.

6 Coding Guidelines

6.1 HTML

coding is art

Im Gegensatz zu z.B. der Programmiersprache **Python** hat man im Quellcode von HTML fast keinerlei Vorgaben, was Zeilenumbrüche oder Einrückungen angeht. Es ist theoretisch möglich, und auch teilweise praktisch vollzogen, den kompletten Inhalt einer HTML - Datei in eine Zeile zu schreiben. In der Praxis wird dies auf einigen Seiten durchgeführt. Selbst ein Leerzeichen nimmt Speicherplatz ein - wenn auch nur gering. Es gibt jedoch mehrere Gründe, warum ein sauberer Code große Vorteile hat:

- Eine eventuelle Fehlersuche geht später schneller.
- Nachträgliches Ändern und Hinzufügen von Informationen ist problemloser.
- Sie selbst sehen Ihre Fehler schneller.
- Der Hörsaalleiter erkennt Tippfehler besser und kann Ihnen schneller helfen

Einfache Kommentare

Wie jede andere Programmiersprache bietet auch **HTML** die Möglichkeit den Quellcode zu kommentieren. Die eingebundenen Kommentare werden dann vom Browser nicht im gerenderten Frontend angezeigt.

Listing 6.1: Einfache Kommentare

```html
<!-- Dies ist ein Kommentar -->
```

Mehrzeilige Kommentare

HTML unterscheidet nicht zwischen ein- und mehrzeiligen Kommentaren. Ein Kommentar-Absatz wird daher genauso erstellt wie ein einzeiliger Kommentar.

Listing 6.2: Mehrzeilige Kommentare

```html
<!--
        Dies ist ein mehrzeiliger Kommentar

        Zweite Zeile des Kommentars
-->
```

Einrückungen

In HTML sollten Tags, die innerhalb eines anderen Tags liegen, immer um 4 Zeichen, bzw. einen Tab eingerückt werden. Es empfiehlt sich in der Praxis, wenn Sie mit mehreren Personen arbeiten, eine Einrücktiefe von 4 Zeichen. Dies wird immer genau gleich dargestellt. Die Breite eines Tabs kann von System zu System variieren, wodurch Code optisch gesprengt werden könnte.

Einrückungen am Beispiel einer unordered list:

Listing 6.3: Einrückungen am Beispiel einer unordered list:

```html
<ul>
    <li>Listenelement 1</li>
    <li>Listenelement 2</li>
    <li>Listenelement 3</li>
</ul>
```

Einrückungen bei einer ganzen HTML Seite

Innerhalb eines HTML - Dokuments hat es sich als praktisch erwiesen, erst ab den Elementen `<head>` bzw. `<body>` einzurücken. Würde man schon `<html>` und `<body>` selbst einrücken, bekäme der erste Inhalt bereits eine Tiefe von 8 Zeichen.

Listing 6.4: Einrückungen bei einer ganzen HTML Seite

```html
<!doctype html>
<html>

<head>

    <title>Titel der Seite</title>

</head>

<body>

    <div class="div1">
        <h1>Überschrift 1</h1>

        <ul>
            <li>Element 1</li>
            <li>Element 2</li>
        </ul>

    </div> <!-- end of DIV1 -->

</body>

</html>
```

Ein Negativbeispiel

So soll es **nicht** aussehen:

Listing 6.5: Ein Negativbeispiel

```html
<ul><li>erster Aufzählungspunkt</li><li>zweiter
Aufzählungspunkt</li><li>dritter Aufzählungspunkt, der
mehr Platz benötigt</li><li>nach Ende keine
Einrückung mehr</li></ul><div><p>Absatz in p in div</p></div>
```

Selbstschließende Tags

Obwohl es seit HTML5 nicht mehr nötig ist, die `<li>` Elemente einzeln zu schließen, sollte trotzdem, aus Gründen der Einheitlichkeit, jedes `<li>` explizit geschlossen werden. Das gleiche gilt für das Schließen eines `<img>`-Tags.

Unquoted Attributes

In HTML5 haben Sie auch die Freiheit, Klassen ohne Anführungszeichen zu vergeben. Dies funktioniert jedoch nur, wenn Sie lediglich genau eine Klasse benötigen. Sollte es aber vorkommen, dass Sie später noch zusätzliche Klassen an ein Element anhängen wollen, hätten Sie dann den Mehraufwand extra " " setzen zu müssen. Deshalb muss sich ein Entwickler von vornherein angewöhnen, auch Einzelklassen in Anführungszeichen zu schreiben. Zusätzlich wird durch diese Maßnahme die Einheitlichkeit und somit die bessere Lesbarkeit des Quellcodes gewährleistet.

Listing 6.6: Unquoted Attributes

```html
<div class=beispiel></div>

<div class="beispiel"></div>

<div class="beispiel beispiel2"></div>
```

Kein Inhalt ohne Tags

In HTML ist es möglich in den Body der Seite einfach normalen Text zu schreiben, ohne diesen mit Tags zu umschließen. Dieser wird unformatiert auf der Seite ausgegeben.

> Sie sollten aber eine Grundregel beherzigen: Kein Inhalt ohne Tags!

Nutzen Sie immer, auch wenn Sie nur schnell Text ausgeben wollen, das Element `<p>`. Dadurch gewinnen Sie die Möglichkeit den Text später zentral mittels CSS anpassen zu können.

Klassen oder IDs?

> Versuchen Sie grundsätzlich Klassen zu verwenden. Nutzen Sie IDs nur an den Stellen, wo dies unumgänglich ist!

6.2 CSS

Kommentare in CSS

Parallel zu HTML gibt es auch in CSS keine Unterscheidung zwischen einzeiligen und mehrzeiligen Kommentaren.

> Kommentare werden in CSS mit "/*" eingeleitet und mit "*/" wieder beendet

Listing 6.7: Kommentare in CSS

```
h1 {
    color:green;
    /* Kommentar in CSS*/

    /*******************
    Optisch hervorgehobener Kommentarblock
    *******************/
}
```

Einrückungen

- Selektoren stehen immer an Position 0.
- Die Attribute der Selektoren werden um 4 Zeichen eingerückt.
- Die öffnenden Klammern der Selektoren stehen, mit einem Leerzeichen getrennt, hinter dem Selektor.
- Die schließenden Klammern stehen an Position 0.