Webdesign

FüUstgSBw
First Edition
HF Weidinger
L Siegerth

Inhaltsverzeichnis

1 Grundlagen 4

2 HTML 6

2.1 HTM...Was? . . . 7
2.2 Tags . . . 9
2.2.1 Tables . . . 10
2.2.2 Headings . . . 12
2.2.3 Paragraphs . . . 13
2.2.4 Images . . . 14
2.2.5 Links . . . 14
2.2.6 Formulare . . . 16
2.2.7 Kommentare . . . 18
2.2.8 Textformatierung . . . 19
2.2.9 Listen . . . 19
2.3 Das Grundgeruest . . . 20
2.3.1 HEAD und BODY . . . 20
2.3.2 Absaetze . . . 20
2.4 DOCTYPE . . . 20
2.5 Sonderzeichen . . . 20
2.6 Zeilenumbrueche . . . 20
2.7 Blockelemente vs. Inline Elemente . . . 21

3 CSS 23

4 WEBSERVER 24

4.1 HTTP und das Web . . . 24

4.1.1 Webserver und Protokolle 24
4.2 Apache 27
4.2.1 Einfuehrung und Installation 27
4.2.2 Konfiguration 30
4.2.3 Sicherheit 34
4.2.4 Indexes 40
4.2.5 SSI 48
4.2.6 CGI 51
4.2.7 PERL mit mod_perl.so 55
4.2.8 Apache und PHP 58
4.2.9 Die Log-Dateien des Apache 60

1 Grundlagen

Woher kommt das Internet

Die ersten Anfänge - Das ARPANet

Im amerikanischen Verteidigungsministerium wurde schon seit den 1960er Jahren darüber nachgedacht, wie man wichtige Daten, auch im Falle eines atomaren Angriffs, schützen könnte. Die aus diesen Überlegungen resultierende Grundidee war, die Daten redundant, auf mehreren Rechnern gleichzeitig zu halten. Um die Aktualität der Daten gewÃƒÆ'Ã‚Â¤hrleisten zu können, mussten diese in der Lage sein, sich selbständig, auf direktem Wege, abzugleichen. Die Schlussfolgerung aus diesen Anforderungen war, dass man ein Netzwerk benötigte, dass diese Rechner verbinden konnte. Eine weitere wichtige Forderung war, dass die Daten auf mehreren unterschiedlichen Wegen, von einem Rechner zum anderen gelangen konnten, damit auch im Falle eines Ausfalls einzelner Rechner, die Übertragung noch möglich war. Somit musste etwas bisher noch nicht da gewesenes geschaen werden. Das ARPANet stellt den Ursprung des heutigen Internets dar. Es wurde in den 1960er Jahren, während des kalten Kriegs, zwischen den USA und der UdSSR, von den Amerikanern, entwickelt. Seinen Namen verdankt es der ARPA (Advanced Research Projects Agency), einer Gruppe von Wissenschaftlern, die 1962, unter der Leitung des Massachusetts Institute of Technology und des US-Verteidigungsministeriums, für die US-Luftwaffe, dieses Netzwerk entwickelten. Enscheidend für seine Entstehung waren die beiden Forscher Paul Baran und Donald Watts Davies. Sie steuerten wichtige Impulse zur Entwicklung bei.

Wie ging es weiter?

Bald sollte sich herausstellen, dass ein solches Netzwerk nicht nur für militärischer Zwecke interessant war, sondern auch für die zivile Wissenschaft. Wissenschaftler in den USA hatten in den 1970er Jahren die Möglichkeit, Daten mit anderen Instituten, über das ARPANet, auszutauschen. Nicht die Tatsache, Daten redundant auf mehreren Rechnern halten zu können, sondern der Datenaustausch an sicht waren für sie interessant. Zu beginn der 1980er Jahre erkannte man, dass durch die nun deutlich gestiegene Rechnerzahl im ARPANet, und die vermischte zivile und militärische Nutzung, eine Aufteilung des Netzwerks erforderlich wurde. Nur so war es den Militärs möglich Geheimnisse, und eigene Interessen zu wahren. Für den militärischen Datenaustausch wurde das "MilNet" geschaffen, das ARPANet wurde vollständig

der zivilen Nutzung überlassen. Durch diese Abkopplung stand einem weiteren Anwachsen des ARPANets nichts mehr im Wege. Und so bürgerte sich nach und nach der Name "Internet" für dieses Netzwerk ein. Der Name ARPANet verschwand im laufe der 80er Jahre. Eng verknüpft mit dem Siegeszug des Internets, war die Entstehung des Datenaustauschprotokolls, TCP/IP.
Durch diese Abkopplung stand einem weiteren Anwachsen des ARPANets nichts mehr im Wege. Und so bürgerte sich nach und nach der Name "Internet" für dieses Netzwerk ein. Der Name ARPANet verschwand im laufe der 80er Jahre. Eng verknüpft mit dem Siegeszug des Internets, war die Entstehung des Datenaustauschprotokolls, TCP/IP.

Weitere Informationen zur Entstehung des Internet

Informationen zu TCP/IP und Weiteres zur Entstehung des Internets, sind in der Tour 2 des Einstiegskapitels von SELFHTML zusammengefasst.

Erarbeiten Sie sich die in Tour 2 bei SELFHTML aufbereiteten Informationen zum Thema Grundlagen des Internets und betrachten Sie sie als Hintergundwissen bzw. Basis der folgenden Kapitel. [17, SELFHTML - Tour 2]

2 HTML

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 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 HyperText Markup Language

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.

2.1 HTM...Was?

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	

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. 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

```
1  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/st    dtd">
2  <html>
3
4      <head>
5      </head>
6
7      <body>
8      </body>
9
10 </html>
```

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

```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>

<h1>Willkommen auf meiner Webseite</h1>

<h2>Hier habe ich eine Liste fÃ¼r euch erstellt</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>
```

2.2 Tags

TAGS in HTML

Ein Element in HTML wird parallel zu XML meist durch einen öffnenden und schließenden Tag abgegrenzt. In HTML nennen sich Begrenzer tag. öffnender HTML Tag span schließender HTML Tag /span
daraus ergibt sich die folgende Syntax für z.B. eine Überschrift
h1Überschrift/h1

Listing 2.4

```HTML
<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

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

2.2.1 Tables

Tabellen in HTML

Das folgende Beispiel zeigt den minimalen Aufbau einer Tabelle.
Tabellen bestehen aus Reihen und Zeilen (rows tr und columns td).

Listing 2.6

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

Sinnvolle Tabellen

Zelleninhalt	Zelleninhalt	Zelleninhalt
Zelleninhalt	Zelleninhalt	Zelleninhalt
Zelleninhalt	Zelleninhalt	Zelleninhalt

Listing 2.7

```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 Element

- thead und
- tbody

ergänzt.
Diese beiden Elemente dienen dazu innerhalb der Tabelle 2 verschiedene Bereiche zu definieren:

- Den Tabellenkopf mit der Beschreibung und
- die Tabelleninhalte

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

Listing 2.8

```html
<table class="example">
        <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 TabellenEnde 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.

2.2.2 Headings

Headings - Überschriften

Überschriften in HTML dienen der Strukturierung der Inhalte.
Eine Überschrift H1 stellt immer die wichtigste Überschrift auf einer Seite dar.
Eine Suchmaschine wie zum Beispiel Google bewertet die Wichtigkeit des Textes einer Überschrift nach der Klassifizierung mit dem jeweiligen H Tag.

Heading 1
Heading 2
Heading 3

Heading 4
Heading 5
Heading 6

Listing 2.9

```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>
```

2.2.3 Paragraphs

Paragraphen und Zeilenumbrüche

Ein Paragraph in HTML stellt ähnlich wie in HTML einen zusammenhängenden Text dar.
Jeder Paragraph soll eine semantische Trennung darstellen.

Listing 2.10

```HTML
<p>Dies ist ein Absatz</p>
<p>Dies ist ein weiterer Absatz</p>
<p>Dies ist der dritte Absatz</p>
```

Zeilenumbrüche

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 2.11

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

Beispiele für Zeilenumbrüche:

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

Listing 2.12

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

Zusammenfassung

- Ein Paragraph in HTML entspricht dem Tastenkommando "ENTER" in Word.
- Ein Zeilenumbruch in HTML entsprichet dem Tastenkommando "SHIFT-ENTER" in Word.

2.2.4 Images

Bilder in HTML

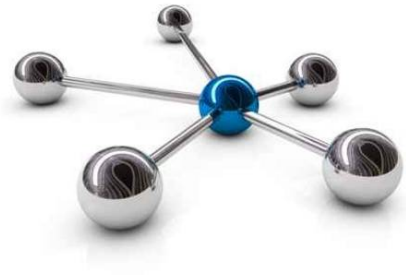
Bilder werden in HTML mittels des IMG tag eingebunden.

Listing 2.13

```HTML
<img src="./img/1342446643_Logo_512.png" alt="HTML5 Logo" />
```

2.2.5 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 er 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 der beiden steht klickbar.

Es kann somit nicht nur Text, sondern auch Bilder auch ganze Container mit einem link versehen werden

Listing 2.14

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

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

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

Links in neuen Fenstern

Mittels der Attribute eines Links eingestellt werden, dass ein Link in einem neuen Fenster geöffnet werden soll. Dies geschieht durch Angabe des Attributes: target 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

Google in neuem Fenster

Listing 2.16

```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

```
<span class="h4">Bilder als Links</span>
<a href="./img/1342449951le-pic.jpg"><img class="small" src="./img/1342449951le-pic.
<div class="gradientbox floatright">Ein Klick auf den Hund Ã¶ffnet das Bild in seinem
<div class="clear"></div>
<span class="h4">Container als Links</span>
<a href=""><div class="blackbox">Ein komplettes DIV als Link</div></a>
<span class="h4">Buttons als Links</span>
<a href=""><button class="button">Button</button></a>
<span class="h4">Text als Links</span>
<a href="">Dieser Text ist ein Link</a>
```

2.2.6 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.

form

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 2.18

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

das input element

das input Element stellt das meistegenutze 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

Durch setzen des Attributs maxlength="4" wird die Eingabe in dieses Feld auf 40 Zeichen begrenzt

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

Mittels der Attribute cols und rows definiert man, wieviele Spalten bzw. Reihen vorgegeben werden sollen. Moderne Browser erlauben zudem, das textarea Element mit der Maus auf die jeweiligen Bedürfnisse des Nutzers temporär zu vergrößern

Listing 2.19

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

2.2.7 Kommentare

Einfache Kommentare

Wie in einer normalen Programmiersprache bietet auch HTML die Möglichkeit den Quellcode zu kommentieren.
Die eingebundenen Kommentare werden dann vom Browser dann nicht im gerenderten Frontend angezeigt

Listing 2.20

```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 2.21

```html
<!--
        Dies ist ein mehrzeiliger Kommentar
        Zweite Zeile des Kommentars
-->
```

2.2.8 Textformatierung

Einfache Formatierungen

This text is bold
This text is italic
`This is computer output`
This is subscript and superscript
~~Dieser Text ist durchgestrichen~~

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

2.2.9 Listen

Geordnete und ungeordnete Listen

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

2.3 Das Grundgeruest

2.3.1 HEAD und BODY

Kopf und Körper - HEAD und BODY

Das Grundgerüst einer HTML-Datei setzt sich aus drei Teilen zusammen:

- der Deklaration des Dokumenttyps,
- dem Kopf der HTML-Datei (Header)
- und dem Körper der HTML-Datei, dem Body.

Jeder dieser drei Teile wird in Form eines Tags angegeben bzw. von einem öffnenden und einem schließenden Tag umgeben.

Listing 2.22

```HTML
<!doctype html>
<html>

        <head>
                <title>Der Titel der HTML - Seite </title>
        </head>

    <body>

        </body>

</html>
```

2.3.2 Absaetze

2.4 DOCTYPE

2.5 Sonderzeichen

2.6 Zeilenumbrueche

Grundsätzlich wird ein Text immer am Ende des Browserfensters bzw. am Ende des umgebenden Blocks (z. B. der Tabellenzelle) automatisch an einem Leerzeichen umgebrochen. Um einen Zeilenumbruch an einer bestimmten Stelle zu erzwingen, verwendet man das Tag BR

Listing 2.23

```HTML
<h1>Ohne UmbrÃ¼che</h1 >
<p>Dieser Text steht in einem ersten Absatz .</p>
<p>Dieser Text in einem weiteren .</p>
<p>In diesem dritten Absatz ist nun noch etwas mehr Text vorhanden .</p>
<h1>Mit UmbrÃ¼mchen </h1 >
<p>Dieser Text steht in <br > einem ersten Absatz .</p>
<p>Dieser Text <br > in einem weiteren .</p>
<p>In diesem dritten <br > Absatz ist nun noch etwas mehr Text
vorhanden .</p>
```

2.7 Blockelemente vs. Inline Elemente

Gruppierung von Elementen

HTML-Elemente können mit Hilfe der beiden Tags und zu Gruppen zusammengefasst werden

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 DIVS

Listing 2.24

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


<p class="orange">Inhalt des orange DIVS</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 orangeSPAN

Listing 2.25

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


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

3 CSS

jeweils viele kleine Nachfrager

Zahl der Anbieter	vollkommener Markt	unvollkommener Markt
viele kleine	vollständige Konkurrenz	monopolistische Konkurrenz
wenige mittlere	homogenes Oligopol	heterogenes Oligopol
ein großer	Monopol	monopolistische Preisdifferenzierung

egalitärer Liberalismus vs. Naturrechtslibertarismus

	Liberalismus	Libertarismus
Privateigentum	pragmatisch (Maximin)	unverletzlich
öffentliche Produktion	pragmatisch (Maximin)	abgelehnt
Steuern	probates politisches Mittel	staatlicher Diebstahl
Privateigentum	pragmatisch (Maximin)	unverletzlich
Umverteilung	Befürwortung (Maximin)	Verteilungsgerechtigkeit irrelevant
Soziale Sicherungssysteme	unterstützt	abgelehnt

4 WEBSERVER

4.1 HTTP und das Web

4.1.1 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?

4.2 Apache

4.2.1 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.

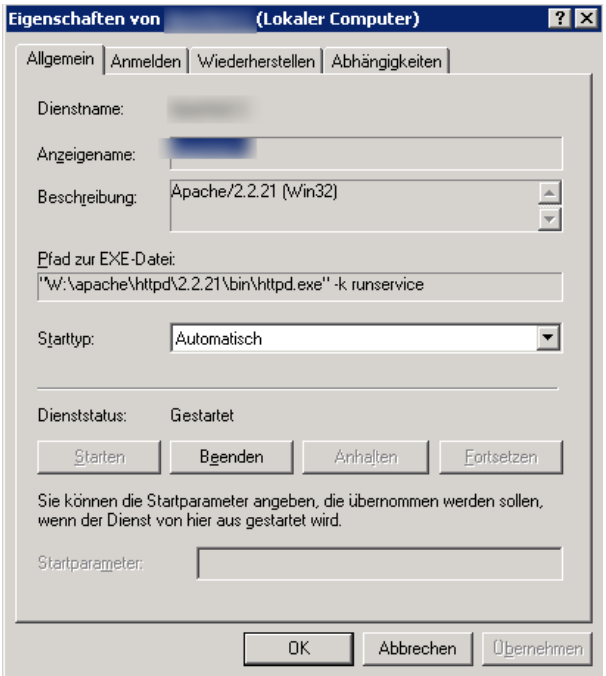
Die Steuerkonsole (Apache Service Monitor)
Mit einem Doppelklick können Sie den Apache Service Monitor öffnen.

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

4.2.2 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
ServerRootE:/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!

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 4.1

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

4.2.3 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 4.2

```
<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 verhät 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 4.3

```
# 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 4.4

```
# Konfiguration 1
  Require all denied
  Require local

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

4.2.4 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 4.5

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

  # FancyIndexing aktivieren
  IndexOptions FancyIndexing

  Options      Indexes
</Directory>

```

Icons hinzufügen

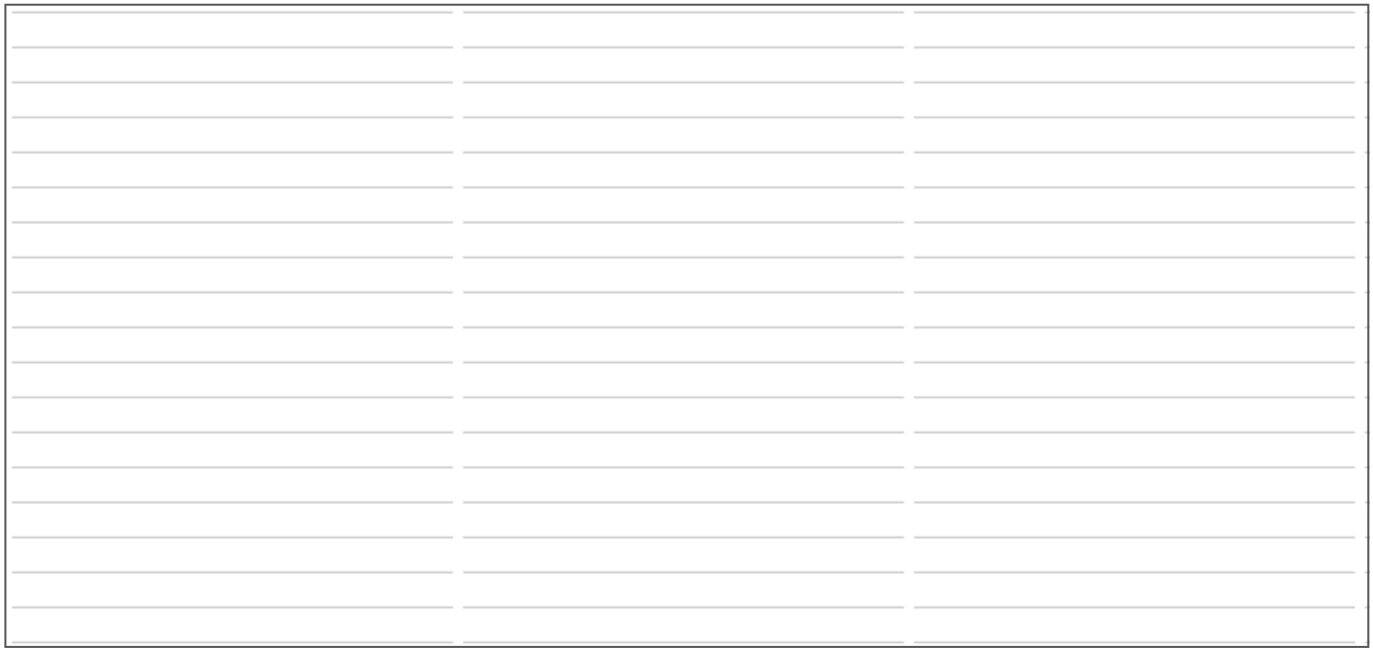
Der Apache HTTP-Server kennt zwei Wege, um Dateitypen mit Icons zu verbinden:

- 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 4.6

```
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 4.7

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

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

```

4.2.5 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 4.8

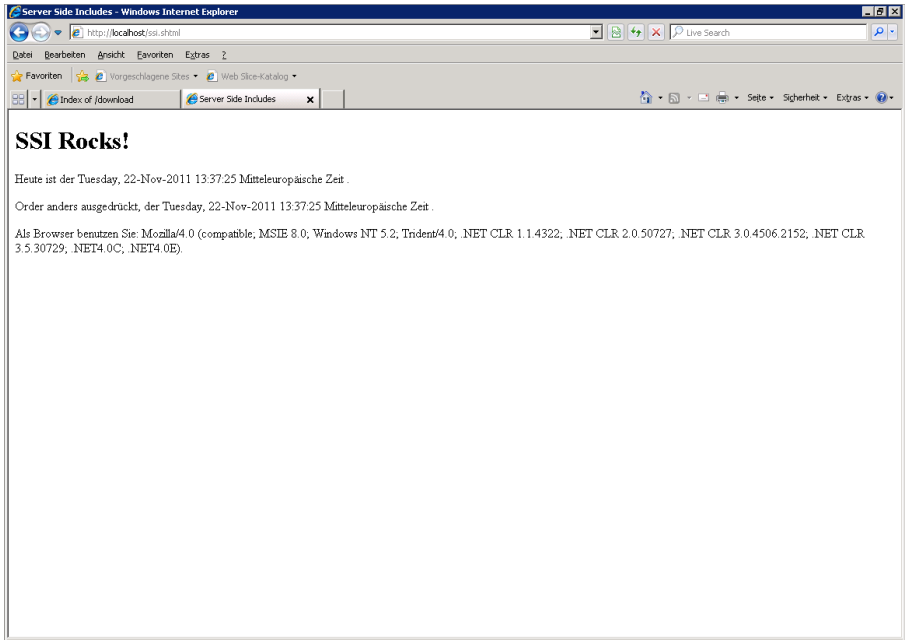
```
LoadModule include_module "modules/mod_include.so"

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 4.9

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 ausgedrÃ¼ckt, der <!--#echo var="DATE_LOCAL" -->.</p>
        <p>Als Browser benutzen Sie: <!--#echo var="HTTP_USER_AGENT" -->.</p>
  </body>
</html>

```

4.2.6 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:perlstrawberry5.14.2.1 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 4.10

```
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 4.11

```
1  Alias "/cgi-bin_2" "E:/apache/httpd/2.4.2/cgi-bin_2"
2
3  <Directory "E:/apache/httpd/2.2.21/cgi-bin_2">
4    #Hinzufuegen eines Actionhandlers fuer CGI
5    AddHandler    cgi-script .cgi .pl
6
7    AllowOverride None
8    Require all granted
9
10   #Die Ausfuehrung von CGI-Skripten zulassen
11   Options       ExecCGI
12 </Directory>
13
```

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 4.12

```
#!"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 4.13

```
#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 gegrÃ¼sst!</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="BegrÃ¼ssen">
        </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?");

```

4.2.7 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 4.14

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

```

4.2.8 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 4.15

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

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!

4.2.9 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.