Webdesign

FüUstgSBw
First Edition
HF Weidinger
L Siegerth

Inhaltsverzeichnis

1 Grundlagen

1.0.0.1 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.

1.0.0.2 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]

1.1 Testseite

1.2 Testseite

2 HTML

2.1 HTM...Was?

2.2 Tags

2.3 Das Grundgeruest

2.3.1 HEAD und BODY

2.3.1.1 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 2.1

```
1.  Das Grundger&uuml;st einer HTML-Datei setzt sich aus drei Teilen zusammen:
2.  <ul>
3.      <li>der <strong>Deklaration</strong> des Dokumenttyps,</li>
4.      <li>dem Kopf der HTML-Datei (Header)</li>
5.      <li>und dem K&ouml;rper der HTML-Datei, dem Body.</li>
6.  </ul>
7.  <strong class="conclusion">Jeder dieser drei Teile wird in Form eines Tags angegeben bzw. von einem
8.  einem schlie&szlig;enden Tag umgeben.</strong>
```

2.3.2 Absaetze

2.4 DOCTYPE

2.5 Sonderzeichen

2.6 Zeilenumbrueche

2.6.2.1

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

3 CSS

3.1 Embedding

4 WEBSERVER

4.1 HTTP und das Web

4.1.1 Webserver und Protokolle

4.1.1.1 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!

4.1.1.2 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"

4.1.1.3 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

4.2.1.1 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!

4.2.1.2 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: DATENPARTITION\apache\httpd\2.4.2

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

4.2.2.1 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!

4.2.2.2 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	

4.2.2.3 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

```
<p>
Wie bereits zu Anfang erw&auml;hnt, ist der Apache HTTP-Server ein modular aufgebautes Produk
</p>
<div class="homework">
        <ul>
            <li>Recherchieren Sie in der Onlinedokumentation (Dynamic Shared Objects), wie das Lade
                <li>Recherchieren Sie in der Dokumentation was die Direktive <span class="apache">D
                <li>Welches Modul muss geladen werden, damit die <span class="apache">DirectoryInde
                <li>Laden Sie das betreffende Modul, tragen Sie diese Direktive in Ihre Datei <span
    </ul>
</div>
<div class="notes 100"></div>
<span class="h3">Bedingtes laden von Modulen</span>
<p>
In manchen Umgebungen kann es notwendig sein, dass ein Apache-Modul nur dann geladen werden darf, w
</p>
<div class="information">
<p>
    Ein Kontainer besteht aus einem &ouml;ffnenden Tag (z. B. <span class="apachecontainer">IfModul
    </p>
</div>
<p>
Im konkreten Falle des <span class="apachecontainer">IfModule</span>-Kontainers hei&szlig;t das, da
</p>
```

CSS

4.2.3 Sicherheit

4.2.3.1 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

```
<span class="apachecontainer">Directory</span>-Kontainer stellen einen zentralen Sicherhei hani
<div class="homework">
    <ul>
                <li>Erl&auml;utern Sie die Bedeutung der Anweisungen im <span class="apachecontaine
        <li>Welches Modul muss geladen sein, damit die Direktive <span class="apache">Require</span
                <li>Laden Sie SELFHTML aus dem Internet herunter, falls dies noch nicht geschehen i
                <li>Erstellen Sie in der Datei <span class="url">httpd.conf</span> einen <span clas
        </ul>
</div>
<div class="notes 70"></div>
```

PHP

4.2.3.2 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 Verzeichniszugriff 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

```
<p>
In machen Situationen kann es notwendig sein, dass die Konfiguration der Zugriffsrechte eines Verze
</p>

<ul>
        <li>Der Serveradministrator gibt der betreffenden Person Zugriff auf die Hauptkonfiguration
        <li>Es werden <span class="url">.htaccess</span>-Dateien genutzt (man beachte den Punkt vor
</ul>
<p>
    Im Allgemeinen d&uuml;rfte die zweite Option, die Nutzung der <span class="url">.htaccess</span
</p>
<div class="information">
Unter MS-Windows ist es nicht m&ouml;glich, dass ein Dateiname mit einem Punkt beginnt. Deshalb mus
</div>
<span class="h3">Benutzung von .htaccess-Dateien</span>
<p>
    <span class="url">.htaccess</span>-Dateien erm&ouml;glichen es, denn Zugriff auf die Verzeichni
</p>
<div class="information">
Die Direktiven einer <span class="url">.htaccess</span>-Datei gelten immer f&uuml;r das Verzeichnis
</div>
<p>
    <span class="url">.htaccess</span>-Dateien nutzen die gleiche Syntax, wie die Hauptkonfiguratio
<div class="information">
    Innerhalb der <span class="url">.htaccess</span>-Datei entf&auml;llt die Angabe des <span class
</div>
<div class="homework">
    Testen Sie, ob die auf dieser Seite gezeigte <span class="url">.htaccess</span>-Datei bereits A
   dert werden, um die <span class="url">.htaccess</span>-Datei wirksam werden zu lassen?
</div>
<div class="notes 70"></div>
<span class="h3">Negative Effekte</span>
<p>
    Bei der Nutzung von <span class="url">.htaccess</span>-Dateien treten im Wesentlichen drei nega
</p>
<ol>
        <li>Sobald mit Hilfe der <span class="apache">AllowOverride</span>-Direktive die Nutzung vo
        <li>Eine <span class="url">.htaccess</span>-Datei wird bei jedem Verzeichniszugriff erneut
        <li>Da die Direktiven aus den <span class="url">.htaccess</span>-Dateien immer kumulativ be
</ol>
<div class="information">
Direktiven aus verschiedenen <span class="url">.htaccess</span>-Dateien werden immer in der Reihenf
</div>
<p>
Ein Beispiel zum vorangegangenen Merksatz:
</p>
<p>
    Es existieren die folgenden Verzeichnisse: <span class="url">htdocs/selfhtml</span> und <span c
</p>
<p>
    Im Verzeichniss <span class="url">htdocs/selfhtml/css</span> ist jedoch in der<span class="url"
```

4.2.4 Indexes

4.2.4.1 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?

4.2.4.2 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 4.4

```
<p>
Das Modul "mod_autoindex.so" bietet viele verschiedene M&ouml;glichkeiten, um einen einfachen
</p>
<div class="homework">
    Aktivieren Sie die Index-Options <span class="apache">FancyIndexing</span>! Was ver&auml;ndert
</div>
<div class="notes height50"></div>
```

CODE

4.2.4.3 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 \beispiel{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 4.5

CODE

```
<p>
Der Apache HTTP-Server kennt zwei Wege, um Dateitypen mit Icons zu verbinden:
</p>
<ul>
        <li>Durch Angabe des Dateityps</li>
        <li>Durch Angabe eines MIME-Types</li>
</ul>
<p>
Hierf&uuml;r existieren drei Direktiven:
</p>
<table style="width:100%">
    <tr><th>Direktive</th><th>Bedeutung</th></tr>
        <tr><td><span class="apache">AddIcon</span></td><td>Verbindet verschiedene Dateitypen mit e
    <tr><td><span class="apache">AddIconByEncoding</span></td><td>Verkn&uuml;pft Dateien mit Hilfe
        <tr><td><span class="apache">AddIconByType</span></td><td>Verh&auml;lt sich genau so, wie <
</table>
<div class="homework">
Was ist ein <strong>MIME</strong>-Type?
</div>
<div class="notes height40"></div>
<p>
Da mit der HTTP-Server MIME-Typen verarbeiten kann, muss das Modul "mod_mime.so" geladen werden. Di
</p>
<div class="homework">
    Nehmen Sie die Direktiven aus \beispiel{TypesConfig} in Ihre Hauptkonfigurationsdatei auf, und
</div>
<div class="notes height70"></div>
```

4.2.4.4 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 4.6

```
<p>
Das das Laden des Icons in der letzten &Uuml;bung fehlschlug lagt daran, dass die ben&ouml;t
<div class="information">
Verzeichnisaliase erm&ouml;glichen es, Ressourcen au&szlig;erhalb des DocumentRoot-Verzeichnisses
</div>
<div class="homework">
Laden Sie das Modul <span class="url">mod_alias.so</span>, und recherchieren Sie, welche Direktive
</div>
<div class="notes height50"></div>
<div class="homework">
Pr&uuml;fen Sie, ob im Verzeichnisindex jetzt ein Icon angezeigt wird! Falls        nicht, kontroll
</div>
<div class="notes height50"></div>
<p>
Um den Zugriff auf die Icon-Ressource zu erm&ouml;glichen, muss dem Nutzer der Zugriff auf das <spa
<div class="homework">
Worin unterscheiden sich die beiden folgenden <span class="apachecontainer">Directory</span>-Kontai
</div>
<div class="notes height50"></div>
<div class="homework">
Formatieren Sie den automatischen DirectoryIndex des Downloadordners nach den folgenden Angaben. Be
    <ul>
                <li>Als Icon f&uuml;r das &uuml;bergeordnete Verzeichnis soll <span class="url">dir
                <li>F&uuml;r <span class="url">*.msi</span>-Dateien soll das Icon <span class="url"
                <li>Das Standardicon f&uuml;r alle unbekannten Dateitypen muss <span class="url">un
                <li>Im Index sollen Verzeichnisse immer an erster Stelle aufgelistet werden, und da
                <li>Die Icons m&uuml;ssen ein Teil der Links sein.</li>
                <li>Dateinamen m&uuml;ssen in ihrer vollen L&auml;nge erhalten bleiben und d&uuml;r
                <li>Die Spalte "Description" soll ausgeblendet werden.</li>
                <li>Die Standardsortierung f&uuml;r den Verzeichnisindex ist eine aufsteigende Reih
    </ul>
</div>
<div class="notes height70"></div>
```

4.2.5 SSI

4.2.6 CGI