



Fakultät für Ingenieurwissenschaften
Studiengang Kommunikationsinformatik

Bachelor Thesis

LaTeX-Vorlage
für eine Bachelorarbeit

Vorname Name
4. November 2010

Prüfer: Prof. Dr.-Ing. Vorname Name (Uni/Hochschule)
Betreuer: Vorname Name (Firma XYZ)

Firma XYZ GmbH
Musterstraße 123
12345 Musterstadt
Deutschland

Uni/Hochschule
Musterstraße 123
12345 Musterstadt
Deutschland

Angaben zum Autor

Name: Vorname Name
Anschrift: Musterstraße 123
12345 Musterstadt

Mat.-Nr.: 1234567

Telefon: 01234/5678
E-Mail: vorname.name@mail.de

Eidesstattliche Erklärung

Hiermit erkläre ich,

Vorname Name
geboren am 01. Januar 1980
in Musterstadt

an Eides statt, dass die vorliegende Bachelor Thesis mit dem Titel

Titel der Arbeit

selbständig verfasst worden ist, dass keine anderen Quellen und Hilfsmittel als die angegebenen benutzt worden sind und dass die Stellen der Arbeit, die anderen Werken – auch elektronischen Medien wie z.B. Internet – dem Wortlaut oder Sinn nach entnommen wurden, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht worden sind. Dies gilt in gleichem Maße für Bilder, Diagramme, Tabellen und sonstige Abbildungen, Programmbeschreibungen oder Quellcodes, etc.

Musterstadt, den 4. November 2010

Danksagung

An dieser Stelle möchte ich mich bei allen bedanken, die mich bei der Realisierung dieser Arbeit unterstützt haben.

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

Ebenfalls möchte ich mich bei meiner Familie und meinen Freunden für die großartige Unterstützung während des gesamten Studiums bedanken.

Inhaltsverzeichnis

Einleitung	iii
1 Hypertext Transfer Protocol (HTTP)	1
1.1 Ablauf einer HTTP-Verbindung	1
1.2 HTTP-Adressierung	2
1.3 Aufbau von HTTP-Nachrichten	3
1.4 HTTP-Methoden	3
1.5 HTTP-Statuscodes	5
Glossar	I
Literaturverzeichnis	III
Abbildungsverzeichnis	V
Tabellenverzeichnis	VII
Listings	IX

Einleitung

Aufbau

Die Bachelor Thesis ist in drei Bereiche gegliedert.

Teil 1 (Kapitel 1, 2, 3):

Dieser Teil befasst sich mit der Einführung und den grundlegenden Informationen zum Thema.

Teil 2 (Kapitel 4, 5, 6):

Im zweiten Teil der Arbeit werden die Probleme dargestellt und mögliche Lösungsansätze diskutiert.

Teil 3 (Kapitel 7):

Der abschließende Teil der Arbeit befasst sich mit der technischen Realisierung der vorgestellten Lösung.

Kapitel 8:

Das letzte Kapitel beinhaltet das Fazit, das die Bearbeitung des gesamten Themas und die dabei aufgetretenen Probleme noch einmal kurz zusammenfasst.

Sonstiges

Eine CD-ROM mit dem aktuellen Quellcode des Projektes sowie einer digitalen Ausführung dieser Arbeit befindet sich im Anhang.

Aufgabenstellung

Kontext

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

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

Ziel

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

Arbeitsablauf

- Punkt 1
- Punkt 2
- Punkt 3

1 Hypertext Transfer Protocol (HTTP)

Das Hypertext Transfer Protokoll (HTTP) ist ein Netzwerkprotokoll zur Übertragung von Daten und ist in der RFC 1945 [1] und 2616 [2] beschrieben. Es wird hauptsächlich zur Übertragung von Webseiten und anderen Webinhalten (Bilder, Texte, Musik, Videos, etc.) eingesetzt. HTTP ist nicht nur auf Hypertext beschränkt, sondern kann durch Erweiterungen der Anfragemethoden, Header-Informationen und Statuscodes zum Austausch von beliebigen Daten verwendet werden.

Im TCP/IP-Referenzmodell ist HTTP der Anwendungsschicht zugeordnet. Die Anwendungsschicht ist die höchste Schicht im TCP/IP-Modell und ist für die Kommunikation zwischen Anwendungen und Diensten der Transportschicht zuständig. Sie ist also nicht mit den Anwendungen selbst gleichzusetzen, sondern verbindet diese lediglich mit dem Netz. Im ISO/OSI Modell entspricht die Anwendungsschicht den Schichten 5-7.

TCP/IP-Schicht		ISO/OSI-Schicht
Anwendung	HTTP	5-7
Transport	TCP	4
Internet	IP (IPv4, IPv6)	3
Netzzugang	Ethernet	1-2

Tabelle 1.1: HTTP im TCP/IP-Protokollstapel

1.1 Ablauf einer HTTP-Verbindung

Die Kommunikation zwischen Client und Webserver erfolgt durch den Austausch von HTTP-Nachrichten. Es gibt zwei unterschiedliche Arten von Nachrichten: die Anfrage (engl. Request) vom Client an den Server und die Antwort (engl. Response) darauf vom

Server an den Client. Zum Austausch der Nachrichten bauen Client und Server eine TCP-Verbindung auf. In der Regel erfolgt dies über Port 80. Da HTTP ein unidirektionales Protokoll ist, stellt ausschließlich der Client Anfragen und der Server antwortet lediglich, stellt aber keine eigene Anfragen.

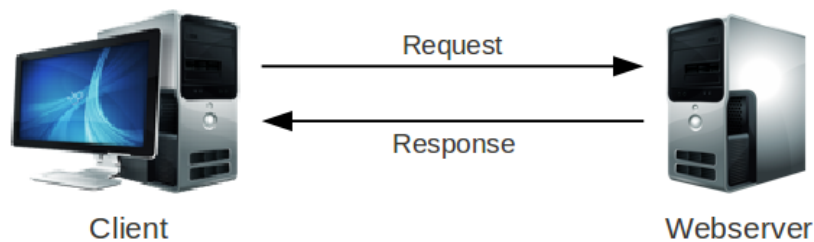


Abbildung 1.1: Client-Webserver Kommunikation

1.2 HTTP-Adressierung

Das Hypertext Transfer Protokoll verwendet zur Adressierung von Ressourcen URLs (Uniform Resource Locator). Eine URL identifiziert und lokalisiert eine Ressource über das verwendete Netzwerkprotokoll und den Ort in einem Computernetzwerk. Die erforderlichen Mindestbestandteile einer URL sind das Protokoll, der Host und der Pfad zu der angeforderten Ressource, wobei die Pfadangabe optional ist.

Protokoll://Host:TCP-Port/Pfad/Datei

`http://www.beispielserver.de:80/images/logo.png`

Als Erstes wird das verwendete Netzwerkprotokoll (z.B. HTTP, HTTPS oder FTP) angegeben. Dann folgt die Host-Komponente in Form einer IPv4 Adresse in dezimaler Schreibweise oder die Angabe eines Domainnamens. Die Angabe des TCP-Ports ist optional. Wird kein Port angegeben, so wird der Standardport des jeweiligen Protokolls (http: 80, https: 443 und bei ftp: 21) verwendet. Der Pfad beschreibt eine bestimmte Ressource auf dem Server. Ist diese Ressource vorhanden, liefert der Webserver diese als Antwort an den Client zurück. Ansonsten wird eine Fehlermeldung an den Client gesendet. Wird kein Pfad angegeben, liefert der Server je nach Einstellung entweder eine vorher festgelegte Seite (z.B. /index.html) oder eine Auflistung der Dateien die sich in dem Verzeichnis befinden zurück.

1.3 Aufbau von HTTP-Nachrichten

Die einzelnen Nachrichten bestehen hauptsächlich aus zwei Teilen, dem Header (Nachrichtenkopf) und dem Body (Nachrichtenkörper). Der Header enthält dabei alle notwendigen Steuerinformationen, während der Body nur die eigentlichen Nutzdaten enthält. Die einzelnen Zeilen des Headers werden als Header-Felder bezeichnet. Ein Header-Feld besteht aus einem Schlüsselwort und den dazugehörigen Informationen.

Eine HTTP-Anfrage besteht mindestens aus einer Methode und einem Hostnamen, gefolgt von Header-Informationen, wie zum Beispiel einer Kennung des verwendeten verwendeten User-Agents (dt. Client-Programm). Wird beispielsweise die URL „http://www.beispielserver.de/images/logo.png“ im Webbrowser aufgerufen, so wird an den Webserver mit dem Hostnamen „www.beispielserver.de“ eine Anfrage für die Ressource „/images/logo.png“ gesendet. Der Client sendet in dem Fall folgende Anfrage an den Server:

```
GET /images/logo.png HTTP/1.1
Host: www.beispielserver.de
```

Listing 1.1: Beispiel für einen HTTP-Request

Die Antwort des Webserver besteht aus einem Statuscode, Header-Informationen des Servers und dem tatsächlichen Inhalt der Nachricht, zum Beispiel dem Dateiinhalt einer angefragten Datei.

```
HTTP/1.0 200 OK
Server: Apache/1.3.29 (Unix) PHP/4.3.4
Content-Length: 7330
Content-Type: image/png
Connection: close

(Dateiinhalt von logo.png)
```

Listing 1.2: Beispiel für eine HTTP-Response

1.4 HTTP-Methoden

Der Client bestimmt durch die Angabe einer Methode, was mit der angefragten Ressource geschehen soll. Die aktuelle HTTP-Spezifikation sieht acht Methoden vor: GET, POST, HEAD, PUT, DELETE, TRACE, OPTIONS und CONNECT.

Die mit Abstand wichtigste Methode ist **GET**. Sie dient zur Anforderung einer bestimmten Ressource unter Angabe eines URI (Uniform Resource Identifier). Durch das

Hinzufügen von Bedingungen im Header-Feld *Conditional* wird aus dem GET ein konditionales GET. Die Anforderung der Daten ist dann an bestimmte Bedingungen geknüpft. Oft verwendete Bedingungen sind zum Beispiel If-Modified-Since, If-Unmodified-Since oder If-Match. Mit Hilfe dieser Bedingungen lässt sich die Netzbelastung deutlich verringern, da nur noch die wirklich benötigten Daten übertragen werden. In der Praxis wird diese Methode zum Beispiel von Proxyserver genutzt um die mehrfache Übertragung von Daten, die sich bereits im Cache befinden zu verhindern. Das gleiche Ziel verfolgt die partielle GET-Methode. Mit Hilfe des Range-Headers fordert der Client einen oder mehrere Bereiche einer Datei an den/die er noch benötigt. Diese Technik erlaubt zum Beispiel die Wiederaufnahme eines unterbrochenen Download ohne bereits heruntergeladene Daten erneut zu übertragen.

Die Methode **POST** dient hauptsächlich dazu um Formulardaten an einen Webserver zu senden. Sie übermittelt auch ein URI, die in diesem Fall lediglich als Referenz dient, welche Routine auf dem Server die Bearbeitung der Daten übernimmt.

Die Methode **HEAD** weist den Server an, den gleichen Nachrichtenkopf wie bei GET zu senden, jedoch ohne die eigentlichen Nutzdaten. Somit lässt sich zum Beispiel schnell die Gültigkeit einer Datei im Cache überprüfen.

Eher seltener werden die folgenden Methoden verwendet:

PUT :

Dient dazu eine Ressource (z.B. eine Datei) auf einen Webserver hochzuladen.
Die URI gibt in diesem Fall das Ziel auf dem Webserver an, unter dem die Datei gespeichert wird.

DELETE :

Dient zum Löschen einer Ressource auf dem Server, die durch die URI referenziert wird.

TRACE :

Dient zum Debuggen von HTTP-Verbindungen.

OPTIONS :

Dient zur Ermittlung von Kommunikationsoptionen durch den HTTP-Client.

CONNECT :

Weist einen Proxy-Server an einen HTTP-Tunnel aufzubauen.

1.5 HTTP-Statuscodes

Jede HTTP-Anfrage wird vom Server mit einem HTTP-Statuscode beantwortet, der Auskunft darüber gibt, ob die Anfrage erfolgreich bearbeitet wurde oder bei der Bearbeitung ein Fehler aufgetreten ist. Ein Statuscode besteht aus einer dreistelligen Zahl. Die erste Ziffer gibt die Status-Klasse an und die beiden weiteren Ziffer spezifizieren den Status genauer.

Statuscode	Bedeutung
1XX	Informativ – Anfrage wurde empfangen, Bearbeitung steht noch aus.
2XX	Erfolg – Anfrage wurde erfolgreich bearbeitet.
3XX	Weiterleitung – Für eine erfolgreiche Bearbeitung der Anfrage sind weitere Schritte des Clients erforderlich.
4XX	Client-Fehler – Die Anfrage enthält eine fehlerhafte Syntax oder kann nicht bearbeitet werden.
5XX	Server-Fehler – Die Anfrage scheint gültig zu sein, jedoch ist die Bearbeitung seitens des Servers fehlgeschlagen.

Tabelle 1.2: HTTP Statuscodes

Zusätzlich zum Statuscode enthält die Antwort des Servers eine Beschreibung des Fehlers im Klartext. So lautet beispielsweise die Antwort des Servers wenn die angeforderte Ressource nicht gefunden wurde: `HTTP/1.1 404 Not Found`

Glossar

CDN

Content Distribution Network, ein Verbund aus mehreren geografisch verteilten Server.

DNS

Domain Name System, ein online verteiltes Datenbanksystem, das IP-Adressen in Domainnamen umsetzt.

HTML

Hypertext Markup Language (dt. Hypertext-Auszeichnungssprache), eine textbasierte Auszeichnungssprache zur Strukturierung von Inhalten wie Texten, Bildern und Hyperlinks in Dokumenten.

ISP

Internet Service Provider, stellt dem Benutzer einen Zugang zum Internet zur Verfügung.

NAT

Network Address Translation, Sammelbegriff für Verfahren um verschiedene Rechnernetze zu verbinden.

Regex, RegExp

Regular expression (dt. regulärer Ausdruck), dient der Beschreibung von Mengen beziehungsweise Untermengen von Zeichenketten mit Hilfe bestimmter syntaktischer Regeln.

URI

Uniform Resource Identifier, dient zur Identifizierung einer abstrakten oder physischen Ressource.

URL

Uniform Resource Locator, identifiziert und lokalisiert eine Ressource in einem Computernetzwerk.

Literaturverzeichnis

- [1] T. Berners-Lee, R. Fielding, H. Frystyk:
RFC 1945 Hypertext Transfer Protocol – HTTP/1.0, Mai 1996
URL: <http://tools.ietf.org/html/rfc1945>
- [2] T. Berners-Lee, R. Fielding, H. Frystyk, J. Gettys, P. Leach, L. Masinter, J. Mogul:
RFC 2616 Hypertext Transfer Protocol – HTTP/1.1, Juni 1999
URL: <http://tools.ietf.org/html/rfc2616>
- [3] The Apache Software Foundation:
Apache HTTP Server Version 2.2 Documentation - FileETag Directive, 2009
URL: <http://httpd.apache.org/docs/2.2/mod/core.html#fileetag>
- [4] Dirk Dithardt: *Squid Administrationshandbuch zum Proxyserver*, 2003
URL: http://www.squid-handbuch.de/hb/node10_mn.html
- [5] Akamai Technologies: *Facts & Figures*, 2007
URL: http://www.akamai.com/html/about/facts_figures.html#4
- [6] J. Dilley, B. Maggs, J. Parikh, H. Prokop, R. Sitaraman, and B. Wehl:
Globally Distributed Content Delivery,
IEEE Internet Computing, September/Oktober 2002,
URL: [http://www.akamai.com/dl/technical_publications/
GloballyDistributedContentDelivery.pdf](http://www.akamai.com/dl/technical_publications/GloballyDistributedContentDelivery.pdf)

Abbildungsverzeichnis

1.1	Client-Webserver Kommunikation	2
-----	--	---

Tabellenverzeichnis

1.1	HTTP im TCP/IP-Protokollstapel	1
1.2	HTTP Statuscodes	5

Listings

1.1	Beispiel für einen HTTP-Request	3
1.2	Beispiel für eine HTTP-Response	3