HOCHSCHULE MITTWEIDA
UNIVERSITY OF APPLIED SCIENCES

Internet (3)
Dienste und deren Anwendungsprotokolle

Prof. Dr.-Ing. habil. Lutz Winkler
Fakultät Elektro- und Informationstechnik
https://www.telecom.hs-mittweida.de
lutz.winkler@hs-mittweida.de

2014-10

Ziel und Inhalt der Vorlesung

HOCHSCHULE MITTWEIDA
UNIVERSITY OF APPLIED SCIENCES

- Ziel: Kennen lernen wesentlicher Internetdienste und deren Anwendungsprotokolle

- Inhalt

Telnet – Telecommunications Network Protocol /RFC 854/

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

- Telnet ermöglicht ein Remote Login.
- Voraussetzung ist:
  - das auf dem entfernten System eine Telnet-Serveranwendung läuft,
  - das auf dem lokalen System ein Telnet-Client existiert (dieser ist oft Teil des Betriebssystems).

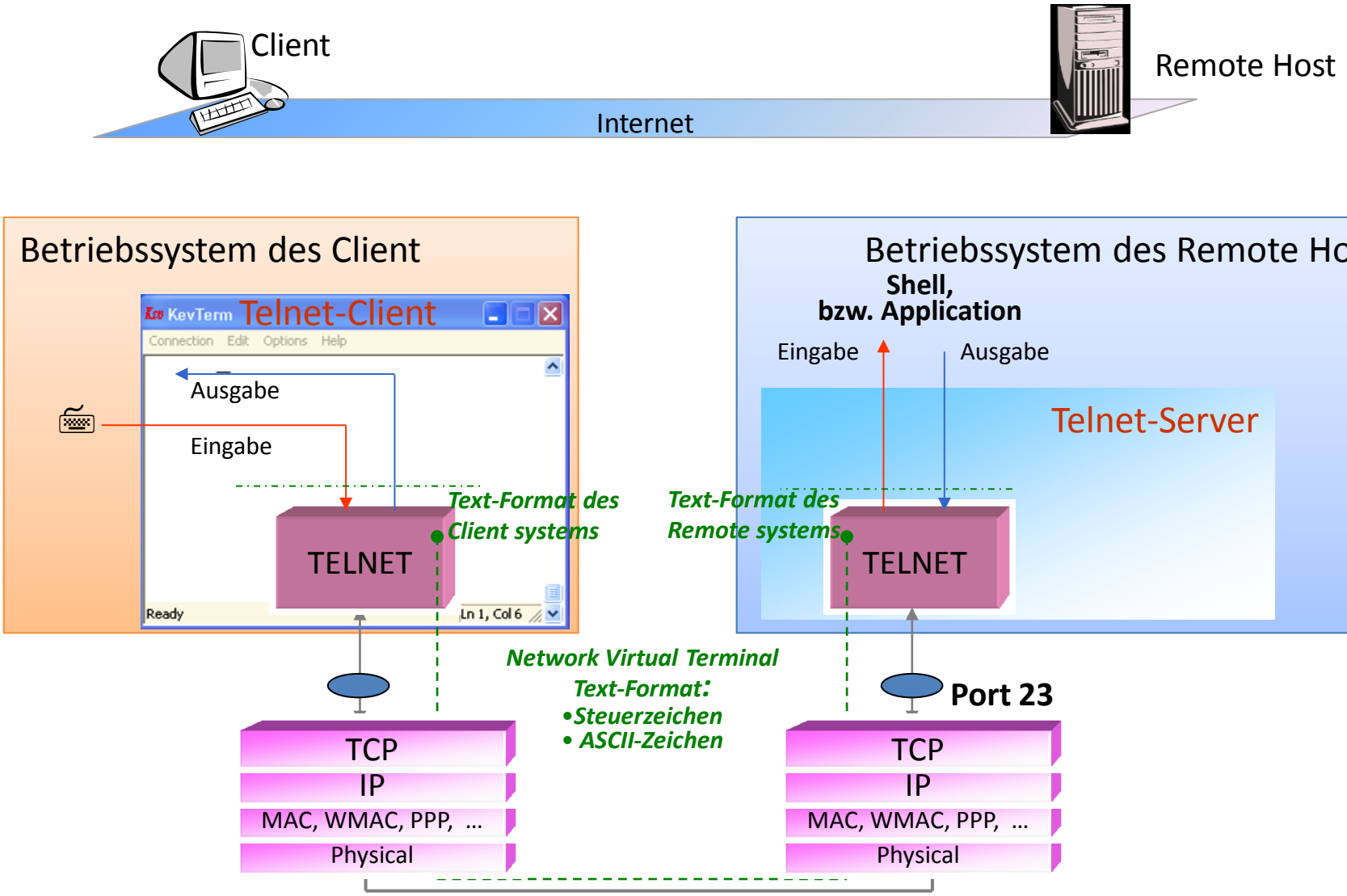
- Telnet-Clients bieten in der Regel mehrere Terminals an, die emuliert werden können:

Bezeichnung	Eigenschaften
IBM3270[1)]	Terminal verhält sich wie ein IBM-Terminal der Serie 3270,
VT52, VT100[2)]	Terminal verhält sich wie die entsprechenden DEC-Terminals,
TTY	Terminal verhält sich wie ein Fernschreiber (teletyper), es wird Text mit CR und LF unterstützt, aber nicht HT, VT, BS, FF usw.
ANSI	Terminal unterstützt den ASCII-Code für alle darstellbaren Zeichen und einen Satz von Format- und Übertragungssteuerzeichen (CR, LF, BEL, BS, HT, VT, FF).

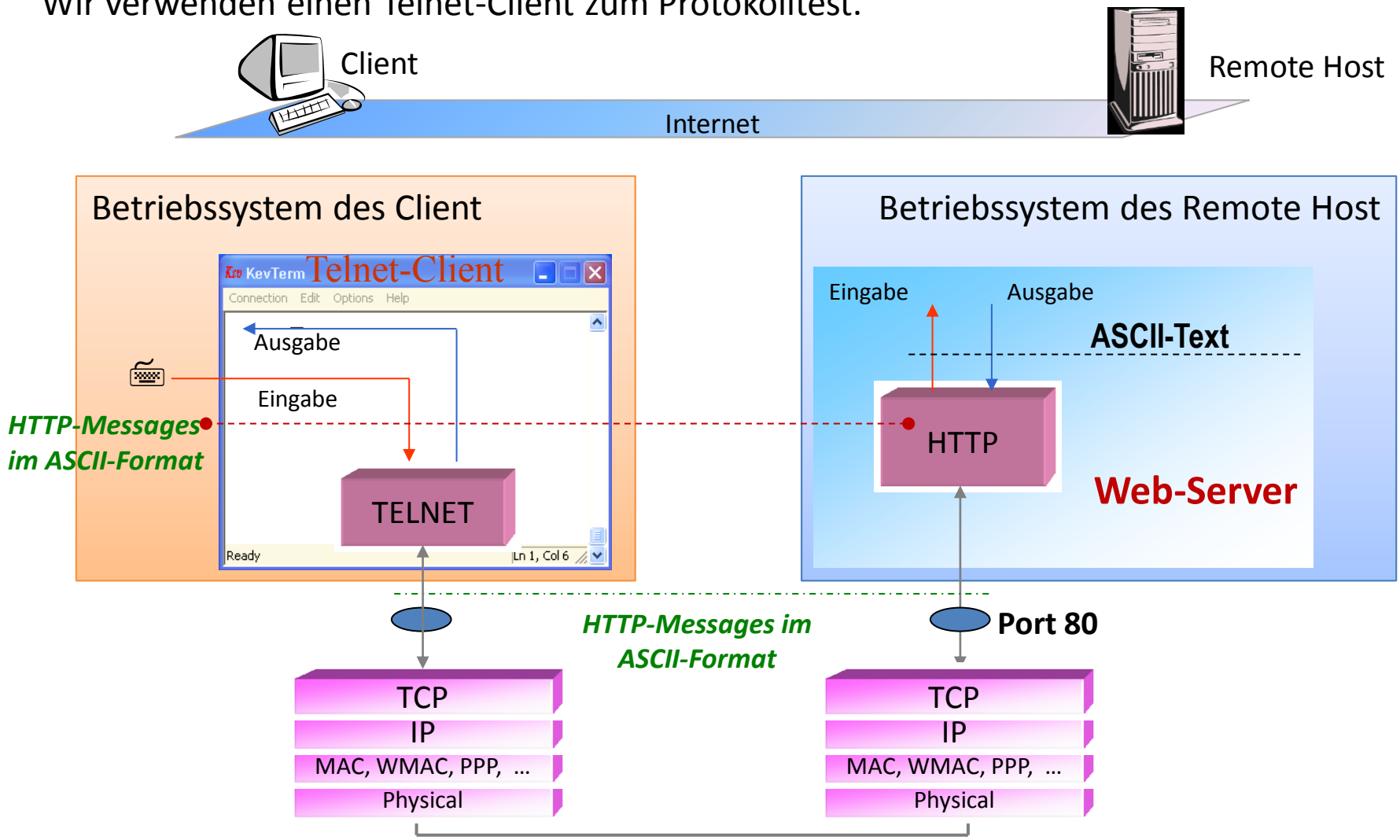
- Das Telnet-Protokoll basiert auf zwei asynchronen, zeichenorientierten Datenströmen, einem vom Client zum Host, der andere vom Host zum Client.

[1)] ab 1970 von **International Business Machines Corp.** eingesetzte Großrechnerfamilie
[2)] textorientierte, asynchron arbeitende Terminals von **Digital Equipment Corporation**

Telnet: Szenario Client-/Serverfunktionen



- Wir verwenden einen Telnet-Client zum Protokolltest.



Telnet: **Telnet-Client unter Windows**

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

- Viele Betriebssystem unterstützen Telnet, indem eine Telnet-Client/-Server-Software bereit gestellt wird (in W7 /W8 muss dieser Dienst extra aktiviert werden).

⌨ Öffnen Sie eine DOS-Shell!

⌨ c:\>**telnet** ↵ //Telnet-Client starten

⌨ Microsoft Telnet>**help**↵ //Hilfe zu Telnet-Client

Befehle können abgekürzt werden. Folgende Befehle werden unterstützt:

c - close	Trennt die aktuelle Verbindung.
d - display	Zeigt Befehlsparameter an.
o - open	Stellt Verbindung her.
q - quit	Beendet Telnet.
set - set	Legt Optionen fest (geben Sie 'set ?' für eine Liste ein).
sen - send	Sendet Zeichenketten an den Server.
st - status	Zeigt Statusinformationen an.
u - unset	Hebt Optionsfestlegungen auf (geben Sie 'unset ?' für eine Liste ein).
?/h - help	Zeigt die Hilfe an.

Starten Sie einen Telnet-Client und nutzen Sie folgende Befehle:

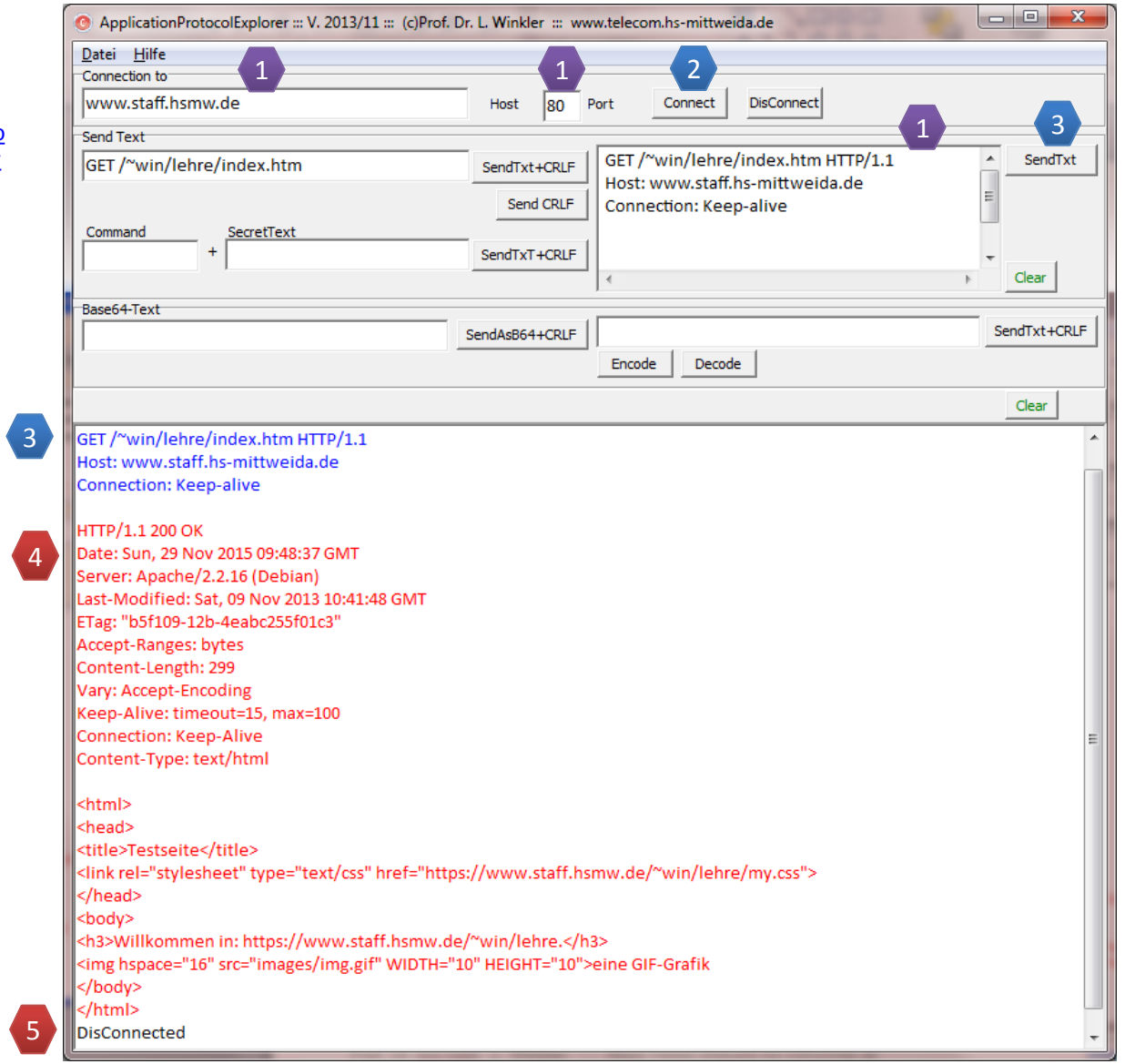
⌨ open www.staff.hsmw.de 80 ↵ //Client soll eine Connection zu Server herstellen

⌨ GET /~win/lehre/index.htm ↵ //An den Server die HTTP-Message "GET" senden

ApplicationProtocolExplorer

Diesen Explorer finden Sie unter https://www.telecom.hs-mittweida.de/fileadmin/verzeichnisfreigaben/telecom/winkler/teachware/MyApplicationProtocolExplorer.exe

- Im Menü Hilfe finden Sie eine Kurzanleitung zum Explorer.
- Im Menü Datei gibt es Befehle mittels derer man Szenarien speichern und laden kann.
- Im Screenshot sieht man folgendes:
  1. Vorbereitende Einträge
  2. Verbindung zum Host www.staff.hsmw.de, Port 80 herstellen
  3. Senden eines GET /~win/… mit der HTTP/1.1-Protokollversion. Der Server soll die Verbindung "wach halten".
  4. Der Server liefert die Ressource dem "UA".
  5. Nach 15 s Inaktivität beendet Server die Verbindung.



Generelle Protokollregeln

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

- Viele Anwendungsprotokolle im Internet sind textbasiert:
  - man kann sie dadurch schnell verstehen,
  - man kann die Texte mittels der Teachware "ApplicationProtocolExplorer" oder eines Telnet-Client zum Server senden.
- Die Protokollnachrichten zum/vom Server unterliegen einer strengen Syntax.
  - Beispiel HTTP:
    - **GET** /~win/lehre/index.htm HTTP/1.1 ist eine gültige Methode aber
    - **get** /~win/lehre/index.htm HTTP/1.1 nicht!
  - Beispiel SMTP:
    - **MAIL FROM**:win@hsmw.de oder **mail from:**win@hsmw.de sind zulässig
    - aber, zwischen ":" und der Mailadresse darf kein Leerzeichen stehen!

- Protokoll-Messages sind überwiegend zeilenorientiert. Eine Zeile wird durch die ASCII-Steuerzeichenfolge CRLF abgeschlossen!

	Dezimal	Hex	String	Tastatur
CR - Carriage Return	#13	0x0d	\r	
LF - Line Feed	#10	0x0a	\n	

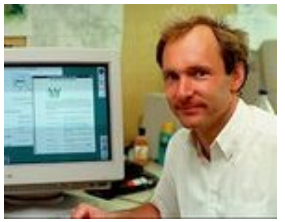
- In den Protokollbeispielen dieses Skriptes sind i.d.R.:
  - Nutzereingaben von Protokollnachrichten in der Textfarbe blau.
  - Protokollnachrichten vom Server in der Textfarbe rot.

WWW - World Wide Web

HOCHSCHULE MITTWEIDA
UNIVERSITY OF APPLIED SCIENCES

- Historie:
  - Entwicklung des WWW begann 1989 am CERN[1)]
  - Weiterentwicklung des Dienstes "Gopher", der verlinkte Textinhalte darstellen konnte.
- Tim Berners-Lee (.uk) und Robert Cailliau (.be) entwickelten:
  - Client/Server-Architektur (WWW-Browser/Server),
  - mit Hypermedia-Inhalten und Links auf weitere Objekte,
  - basierend auf der Textauszeichnungssprache HTML und
  - dem HTTP – Hypertext Transfer Protocol.

Robert Cailliau



Tim Berners-Lee

  - **1990**, erster Prototyp eines WWW-Systems,
  - **1991**, erstes WWW-Basismodell.
  - **1992**, öffentlicher Auftritt im Internet.
  - **1993**, Marc Andressen vom NCSA (National Center für Supercomputing Applications) stellt Mosaic-Browser vor. A. wurde Mitbegründer von Netscape.
  - Ab **1993/1994** stark zunehmende Verwendung.
- Die Anwendungen "World Wide Web" und "E-Mail" machten das Internet zum Massennetz.

[1)] CERN – "Conseil Européenne pour la Recherche Nucléaire" - Europäisches Kernforschungszentrum in Genf

WWW: auch Web|W3|3W|W3

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

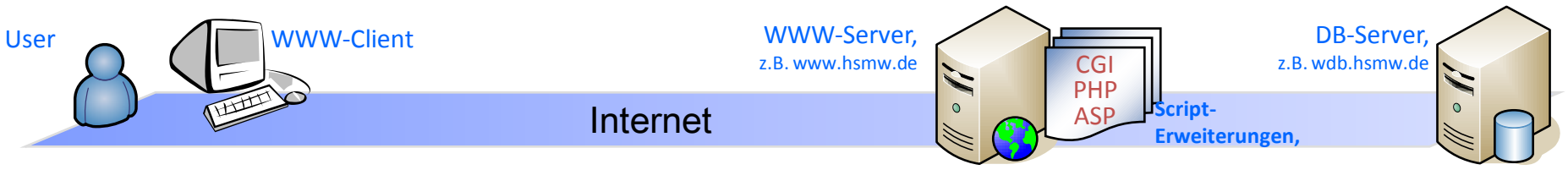
- Dienst WWW ist eine verteilte Anwendung, bestehend aus Client- und Serverteil:
  - **WWW-Clients** (Webbrowser): Internet Explorer, Netscape, Opera, Firefox, …
  - **WWW-Server**: Apache, Internet Information Server, …
- World Wide Web (www) →ist ein Hyper-Media-**Dienst**

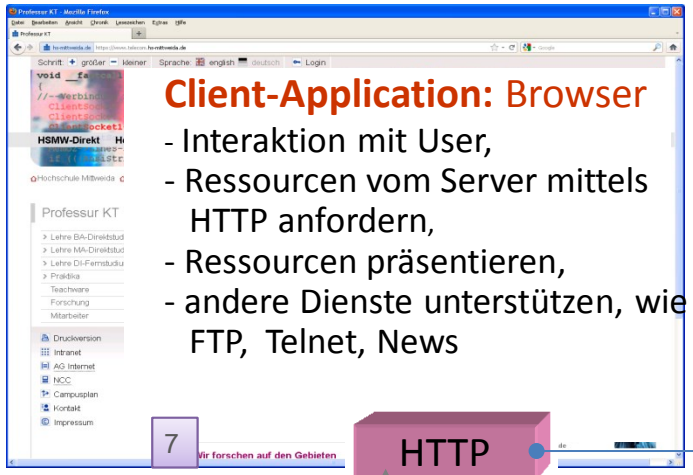
  Hyper-Media = Hypertext + Multimedia

  - **Hypertext**[1)] ist eine *multi-lineare* Organisation von Objekten, deren netzartige Struktur durch logische Verbindungen (Hyperlinks) realisiert wird.
  - **Multimedia:** Nachrichten, bestehend aus mindestens zwei unabhängigen Medien, wovon mindesten eins dynamisch sein muss.
- Das WWW unterstützt den Zugriff und die Präsentation auf:
  - **statische** Ressourcen: werden durch einen Autor endgültig beschrieben und auf Server gespeichert,
  - **dynamische** Ressourcen: werden zur Anforderungszeit teilweise oder vollständig vom Server erstellt,
  - **aktive** Ressourcen: sind dynamische Ressourcen, die zusätzlich beim Client geändert werden können.
- Beachte:
  - Das WWW wird oft synonym für "Das Internet" verwendet, was falsch ist!
  - Das **Internet** ist ein **Verbund von Hosts**.
  - Der **Dienst WWW** ist ein **Verbund von Resources**, bereitgestellt von Servern, auf die man mit einem Webbrowser zugreifen kann.

[1)] nach Wkipedia

WWW: Szenario Client-/Server





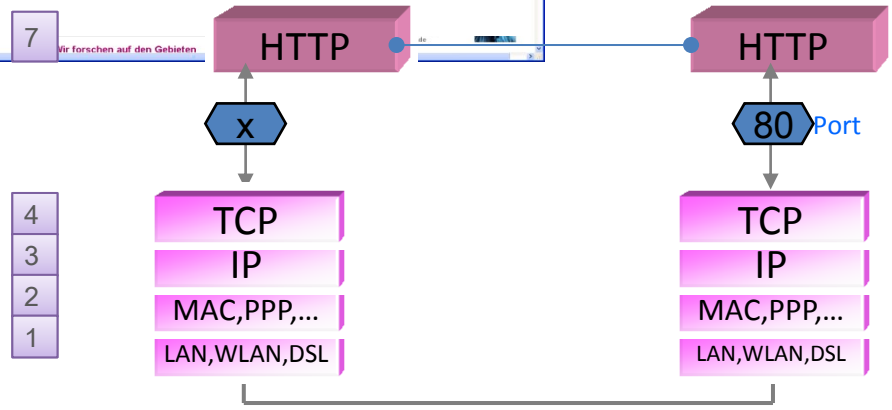
Client-Application: Browser

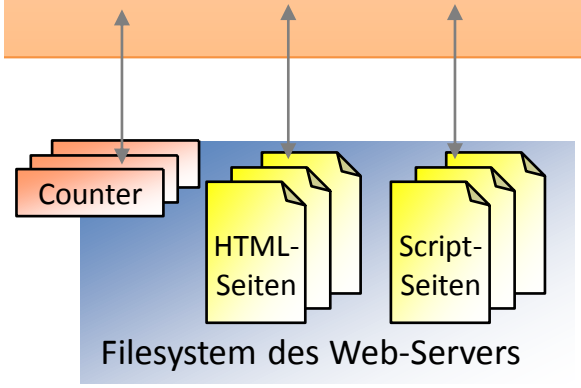
- Interaktion mit User,
- Ressourcen vom Server mittels HTTP anfordern,
- Ressourcen präsentieren,
- andere Dienste unterstützen, wie FTP, Telnet, News

Server-Application: Web-Server

- Anfragen von Browsern per HTTP annehmen,
- Ressourcen per HTTP zu Browsern schicken,
- Fehlermeldung: "404-Datei nicht gefunden",
- Berechtigungen prüfen (Name, PWD)
- Scriptverarbeitung (PHP, CGI, ASP),
  →oft mit Zugriff auf Datentabellen
- Zugriffs-Zähler verwalten,

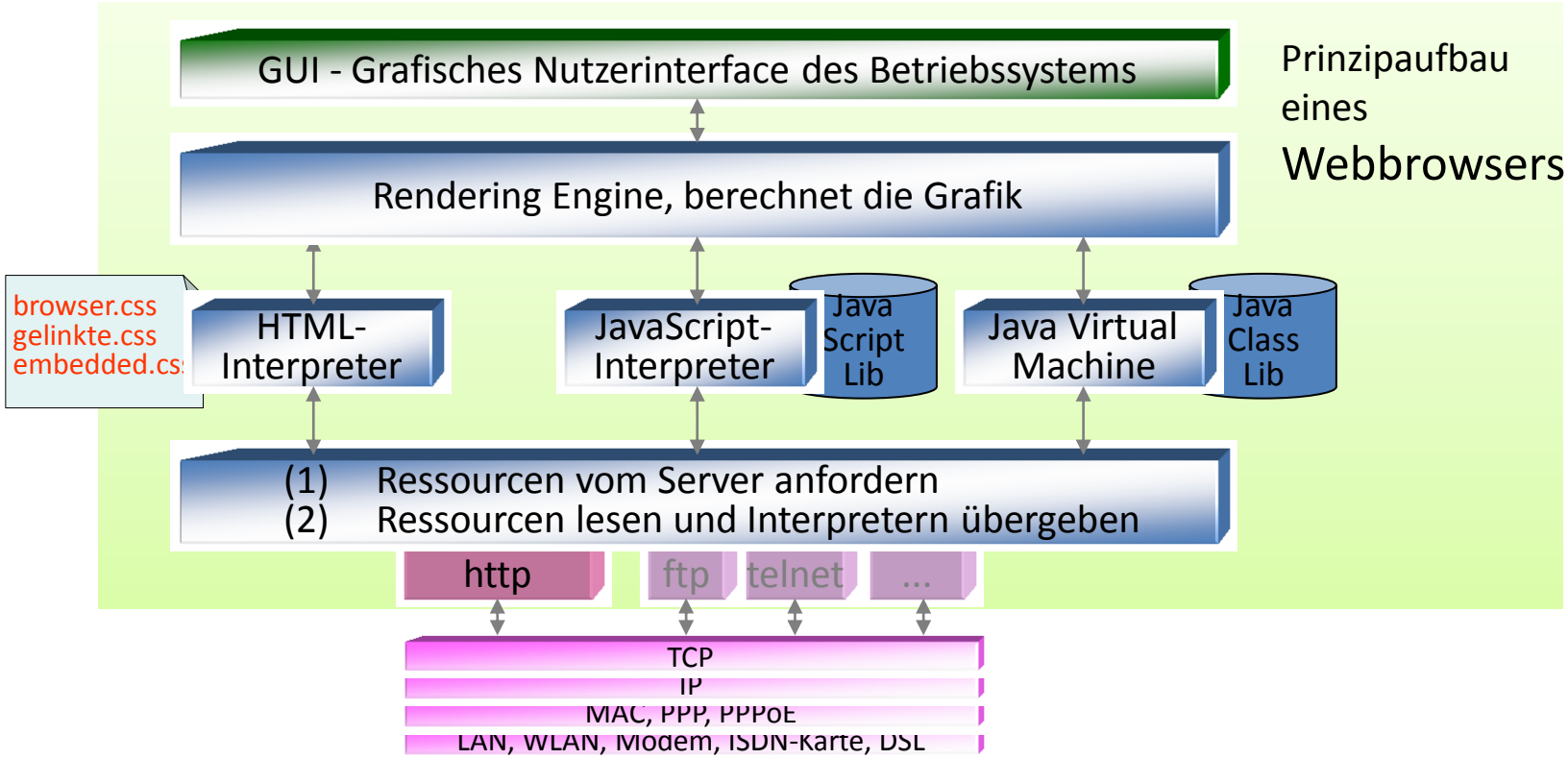
Daten-banken mit Daten-tabellen





WWW: Prinzip eines Webbrowsers

- Webbrowser sind meist Multi-Clients für WWW, FTP, Telnet usw..
- Die WWW-Funktionalität besteht aus der:
  - Darstellung von HTML –Text (basierend auf CSS), Bilder (gif, jpg, png) usw.
  - Ausführung von JavaScript und Java.
- Mittels Plug-In's können WWW-Browser weitere Formate unterstützen (pdf, doc, …).



WWW: Basiskonzepte des Dienstes

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

- **HTML** (Hyper Text Markup Language), ist eine Textauszeichnungssprache.
- **XHTML** (eXtensible HTML), stringentere XML-basierte Auszeichnungssprache
- **CSS** (Cascading Style Sheets):
  - ist eine Sprache zur Darstellungsbeschreibung von HTML-Elementen
  - bestimmt, wie HTML-Elemente dargestellt werden.
- **JavaScript**
  - Programmiersprache, die vom Webbrowser unterstützt wird,
  - erlaubt aktive Dokumente,
  - Überprüfung von Formulareingaben, Senden/Empfangen von Daten (Ajax), …

- **URI** (Uniform Resource Identifier), zur Identifizierung von Ressourcen (HTML-Seiten, Bilder, Audio, Video, Dienste, …)
- **HTTP** (Hyper Text Transfer Protocol):
  - ist das WWW-Anwendungs-Protokoll, genutzt zur Client-Server-Kommunikation.
  - ein Client sendet HTTP-Requests, der Server antwortet mit HTTP-Responses.

- **HTML** ist eine Textauszeichnungssprache, basierend auf
  `elements = <tag attr="..."> inhalt </tag>`
- Eigenschaften der HTML-Elemente werden durch Tags und deren Attribute beschrieben. Die Darstellung basiert auf Cascading Style Sheets (CSS).
- HTML-Standardisierung erfolgt durch das W3C (World Wide Web Consortium).

Versionen		Merkmale
HTML		Darstellung von Text (normal, fett, kursiv), Bilder, Listen
HTML 2.0	1995	+Formulare (Interaktion mit den Nutzern)
HTML 3.2	1997	+Tabellen, Ausrichtungsoptionen von Text, Applets, Frames
HTML 4.01	1999	Bessere Unterstützung von Multi-Media, Script-Sprachen, Stilvorlagen, striktere Trennung von Inhalt (HTML) und Darstellung (CSS)
XHTML 1.0	2000	ist äquivalent zu HTML 4.01, strenger DTD-basiert (strict, transitional, frameset)
XHTML 1.1	2001	Modulorientierte Eigenschaften, transitional und frameset entfallen.
HTML 5	2009	Darstellung von Videos, Audios ohne Plug-In's, Browsergames (basierend auf Canvas und JavaScript) anstelle Flash, …????

Siehe auch deutschsprachige W3C-Site: http://www.w3c.de

WWW-Basiskonzepte: **URI = URL | URN** /RFC 3986, 2005/

- **URI** (Uniform Resource Identifier): ist ein Konzept zur Ressourcen-Identifizierung.
- Ressourcen sind Webseiten, Bilder, Audios, Videos, Dienste, … .
- Ressourcen können identifiziert werden durch:
  - einen **URL** (Uniform Resource Locator),
  - einen **URN** (Uniform Resource Name),
  - oder durch **beide**.

	Beispiele
URLs	ftp://ftp.is.co.za/rfc/rfc1808.txt
	http://tools.ietf.org/html/rfc3986#page-6
	ldaps://ldap.hs-mittweida.de:636/uid=winlehre,dc=hs-mittweida,dc=de
	mailto: winlehre@hs-mittweida.de
	tel: +49-3727-58-1290
	telnet: / / mail.hs-mittweida.de:25
URN | URL	urn:nbn:de:hbz:6-85659545082 http://miami.uni-muenster.de/resolver/urn:nbn:de:hbz:6-85659545082
URN | URL	urn:nbn:de:gbv:7-isbn-90-6984-508-3-8 http://resolver.sub.uni-goettingen.de/purl/?isbn-90-6984-508-3

Hintergrundinfos
http://www.d-nb.de/netzpub/erschl_lza/np_urn.htm
http://www.nbn-resolving.org/
http://www.nbn-resolving.org/resolve_urn.htm
http://www.nbn-resolving.org/resolve_url.htm

nbn - National Bibliography Number

- Ein URI besteht max. aus Angaben: zum Schema, dem **Hier**archieteil, Anfrageteil, Fragmentteil [1].

```
URI         = scheme ":" hier-part [ "?" query ] [ "#" fragment ]

scheme      = ftp | http | mailto | telnet | ldap

hier-part   = "//" authority path-abempty | path-absolute | path-rootless
              | path-empty

authority     = [user[":"password]"@"] host  [":"port]

path          =  path-abempty     ; begins with "/" or is empty
                | path-absolute   ; begins with "/" but not "//"
                | path-rootless   ; begins with a segment
                | path-empty      ; zero characters

path-abempty  = *( "/" segment )
path-absolute = "/" [ segment-nz *( "/" segment ) ]
path-rootless = segment-nz *( "/" segment )
path-empty    = 0<pchar>

segment       = *pchar
segment-nz    = 1*pchar
pchar[2]      = unreserved | pct[3]-encoded | sub-delims | ":" | "@"
```

1) deklariert in ABNF – Augmented Backus Naur Form, RFC 2234
2) printable characters 0x20 bis 0x7E ASCII
3) Percent encoded, z.B. das Leerzeichen %20

- Der Hierarchie-Part wird immer durch "//" eingeleitet und endet z.B.:
  - ohne was `https://www.telecom.hs-mittweida.de`
  - mit einem "/", `https://www.eit.hs-mittweida.de/studium.html`
- Weitere Beispiele:
  - `https://www.eit.hs-mittweida.de/en/studium/direktstudium/elektro-und-informationstechnik-bsc.html#c23449`
  - `https://www.intranet.hs-mittweida.de/nsoft/his/spl/spl.dozent.asp?tabs_selected=Stundenplan&data=sppp&id=490&semester=30`
  - `ftp://win@ftp-www.hs-mittweida.de`
  - `ftp://winlehre:winpwd@ftp-www.hs-mittweida.de`
  - `telnet://141.55.192.84:25`
- Sonderzeichen in einem URI:

```
:  Trennzeichen zwischen Schema und Rest
//Einleitung eines URL
/  Beginn eines Pfades
?  Beginn der Query (Abfrage, Anfrage)
=  Trennzeichen zwischen Variablenname und Variablenwert
&  Trennzeichen zwischen Variablenpaaren.

#  leitet Verweis auf ein Fragment (Marke) ein.
%  Escapezeichen, z.B. Leerzeichen %20
```

- Beachte: URL-Länge < 2000 Zeichen!

- Der **URN**, als **weltweit dauerhafter, einmaliger Name**, ist wie folgt definiert:

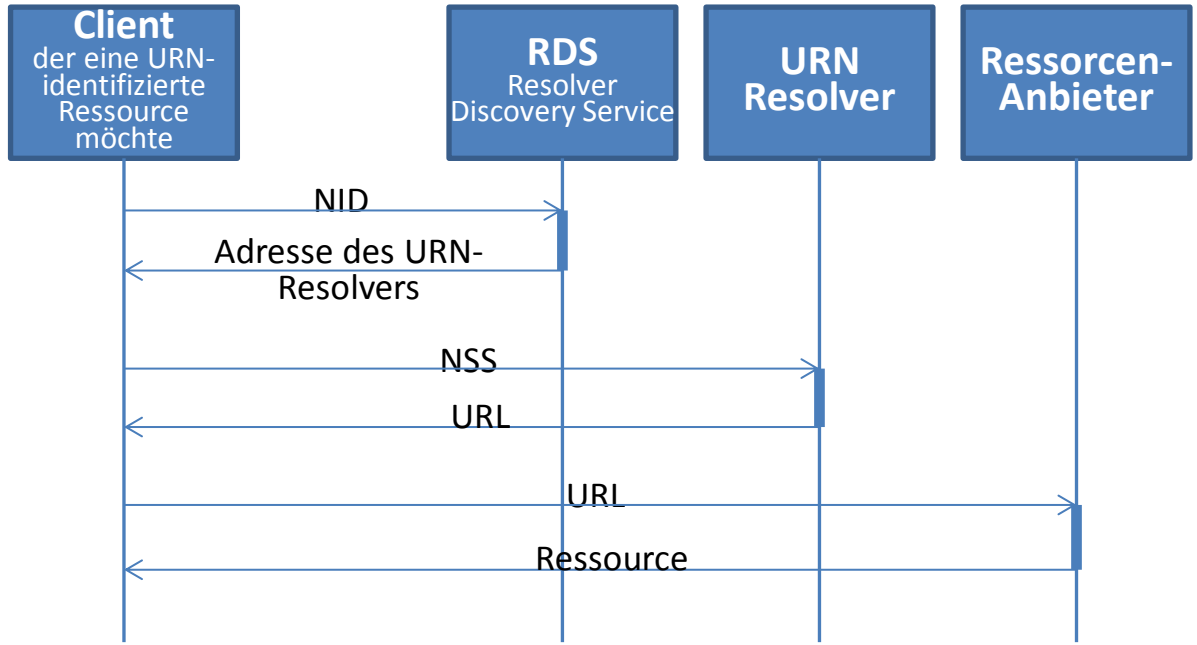
  `<URN> = urn ":" <NID> ":" <NSS>` /RFC 2041/

  - **NID** = Namespace Identifier, sind eindeutige Namensräume. Übersicht: www.iana.org/assignments/urn-namespaces
  - **NSS** = Namespace Specific String, konkreter Name der Ressource, wird vom NID-Besitzer verwaltet.

- Beispiel: **urn:isbn:978-3-936931-51-8**, identifiziert J. Rech: Wireless LAN, Heise, 3. aktualisierte und erweiterte Auflage 2008
- Will man auf eine URN-identifizierte Ressource zugreifen muss der URN auf einen URL aufgelöst werden.

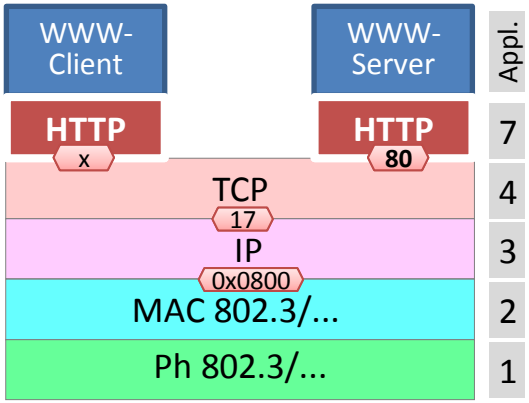
- Die URN-zu-URL-Auflösung erfolgt in drei Schritten:

1. Resolveradresse des NID-Besitzers
2. Auflösung eines URN auf einen (mehrere) URL
3. Mittels URL Zugriff auf Ressource.



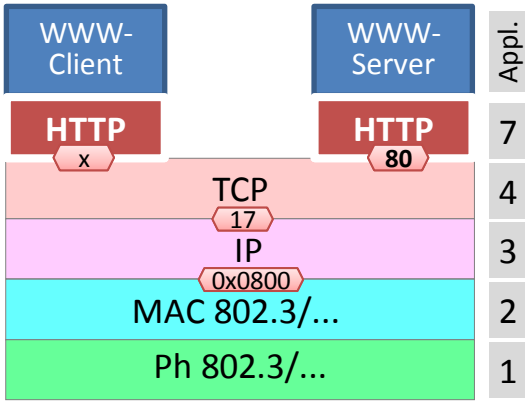
WWW-Basiskonzepte: **HTTP** – Hyper Text Transfer Protocol

- HTTP (Hyper Text Transfer Protocol):
  - Anwendungs-Protokoll, genutzt zur Kommunikation zwischen WWW-Client und WWW-Server.
  - Der Client sendet HTTP-Requests, Server antwortet mit HTTP-Responses.
  - Die Serveranwendung nutzt TCP-Port 80.
  - HTTP ist ein textbasiertes Protokoll
- Man könnte sagen: "WWW-Browser und WWW-Server unterhalten sich HTTP-lerisch"



WWW-Basiskonzepte: **HTTP** – Hyper Text Transfer Protocol

- HTTP (Hyper Text Transfer Protocol):
  - Anwendungs-Protokoll, zur Kommunikation zwischen Client und Server,
  - Client sendet HTTP-Requests, Server antwortet mit -Responses,
  - nutzt TCP/IP und ist ein textbasiertes Protokoll.
  - Man könnte sagen: "WWW-Browser und WWW-Server unterhalten sich HTTP-lerisch"



Version	RFC/Jahr	Merkmale
HTTP 0.9	1991	nur Methode GET, keine Headers
HTTP 1.0	RFC 1945, 1996	– Einführung von General-, Request-, Response- und Entity-Headers, – einfache Authentikation, – neben GET zwei weitere Methoden (POST, HEAD), – Fehlercodes.
HTTP 1.1	RFC 2068, 1999 RFC 2616, 1997	– Entity-Format: MIME-Typen (MIME - Multipurpose Internet Mail Extensions), – unterstützt Webhosting (d.h., mehrere virtuelle Domains unter gleicher IP-Adresse) – Verbindungsmanagement (d.h. Bezug mehrerer Ressourcen über eine Verbindung) – Weitere Methoden wie PUT, DELETE, ... (damit wäre z.B. ein direktes Editieren von Webseiten möglich)
HTTPbis	RFC xxxx, 2012	http://tools.ietf.org/wg/httpbis/

WWW: HTTP-Terminologie /RFC 2616/

Begriff	Erläuterung
Connection	TCP-Transportverbindung zwischen WWW-Client und WWW-Server
Message	Basiseinheit der HTTP-Kommunikation
Request	Message vom Typ "Request" →Anforderung an einen Server.
Response	Message vom Typ "Response" →Reaktion des Servers
Resource	Ein Datenobjekt oder Dienst, referenziert durch einen URI
Entity	Genaue Darstellung eines Datenobjektes oder Antwort von einem Dienst, verwendet in Request und Response: `entity = entity header [entity]`
Client	Anwendungsteil des Nutzers: Verbindung herstellen, Ressourcen anfordern u. darstellen
User Agent	Anwendungsprogramm welches Nutzereingaben ausführt →Der Webbrowser
Server	Anwendungsteil des Dienstbereitstellers: →Verbindungen akzeptieren→geforderte Ressourcen bereitstellen
Origin-Server	Server, auf dem das Original einer Ressource liegt oder erzeugt wird.
Proxy	Ein Zwischensystem (clientseitiges Portal) zwischen Clients und Servers. Alle Anforderungen gehen über den Proxy. Der bedient Requests aus Cache oder "leitet" den Request an den Origin-Server weiter
Gateway	Ein Zwischensystem (serverseitiges Portal), welches Requests bedient, wobei die Ressourcen im Gateway-Cache oder auf (verdeckten) Origin-Servern liegen.
Cache	Programm zum Speichern empfangener Responses, verbunden mit einem Subsystem zum "Speichern", "Löschen" und "Erneutem Anfordern" von Ressourcen.

WWW: HTTP/1.0 -Message /RFC 1945/

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

- HTTP messages consist of requests from client to server and responses from server to client.

```
HTTP-message = Simple-Request | Simple Response    ; HTTP/0.9 messages
             | Full-Request   | Full Response      ; HTTP/1.0 messages
```

- Full-Request and Full-Response use the generic message format of RFC 822 for transferring entities. Both messages may include optional header fields (also known as "headers") and an entity body. The entity body is separated from the headers by a null line (i.e., a line with nothing preceding the CRLF).

```
Full-Request = Request-Line
               *((General-Header|Request-Header|Entity-Header)CRLF)
               CRLF
               [ Entity-Body ]
Full-Response= Status-Line
               *((General-Header|Response-Header|Entity-Header)CRLF)
               CRLF
               [ Entity-Body ]

Simple-Request  = "GET" SP Request-URI

Simple-Response = [ Entity-Body ]
```

WWW: HTTP/1.1 – Message /RFC 2616/

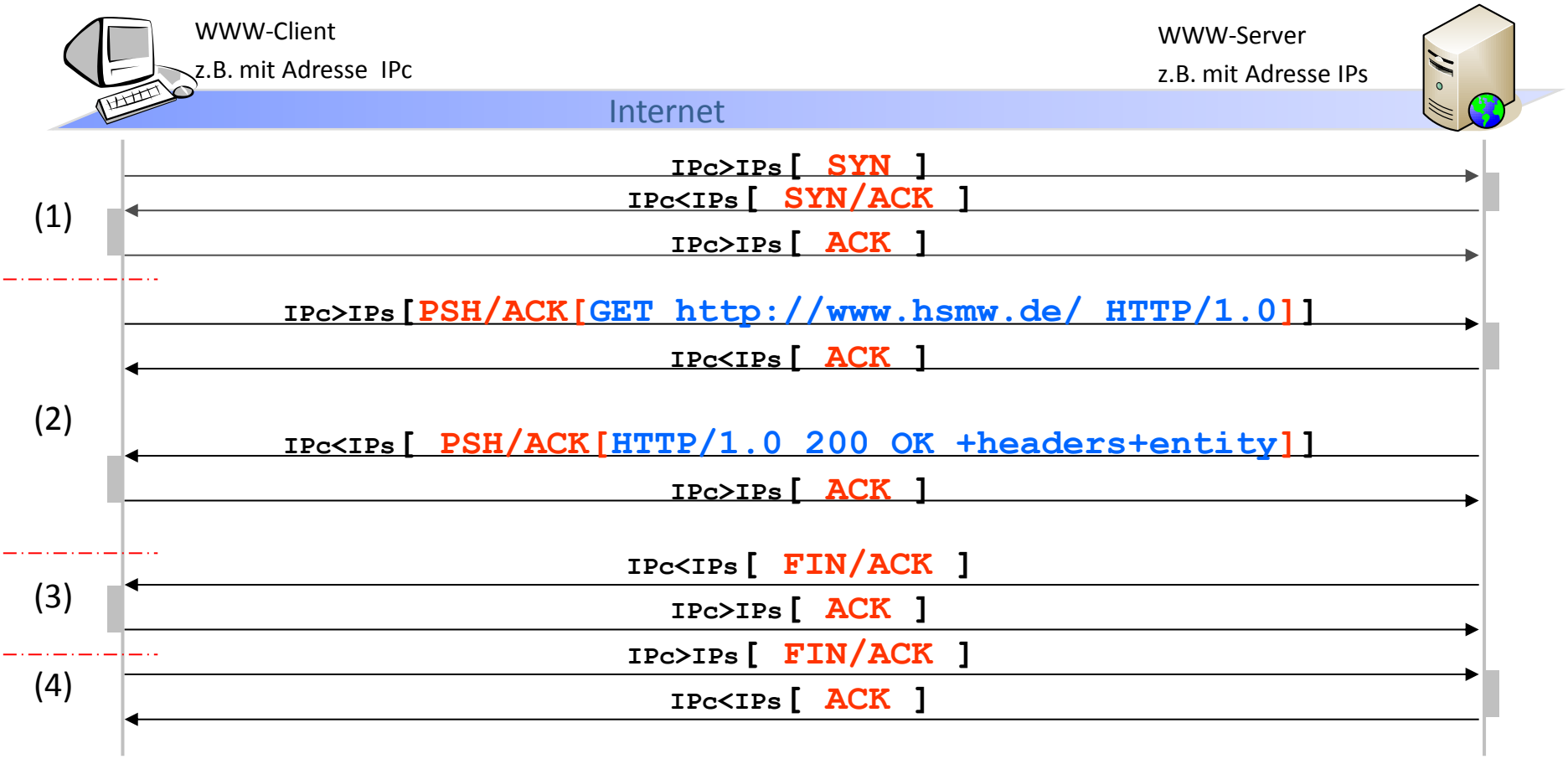
```
HTTP-message      = Request | Response ; HTTP/1.1 messages

generic-message   = start-line *(message-header CRLF) CRLF [ message-body ]
start-line        = Request-Line | Status-Line

Request           = Request-Line
                    *(( general-header | request-header | entity-header ) CRLF)
                    CRLF
                    [ message-body ]

Request-Line      = Method SP Request-URI SP HTTP-Version CRLF
Method            = "OPTIONS" | "GET" | "HEAD" | "POST" | "PUT" | "DELETE"
                    | "TRACE" | "CONNECT"  | extension-method
extension-method  = token

Response          = Status-Line
                    *(( general-header | response-header | entity-header ) CRLF)
                    CRLF
                    [ message-body ]
Status-Line       = HTTP-Version SP Status-Code SP Reason-Phrase CRLF
```

WWW: HTTP/1.1 - Methoden

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

Method	0.9	1.0	1.1	
DELETE	-	-	o	Löschen von Dokumenten auf dem Server.
GET	m	m	m	Anforderung, der im Request-URI angeforderten Ressource. **GET** /~win/lehre/index.htm HTTP/1.1 CRLF Host:www.staff.hsmw.de CRLF CRLF
HEAD	-	m	m	Anforderung des Headers einer Ressorce. Damit kann man die Existenz einer Ressource und die Aktualität einer Ressource überprüfen (Cache). **HEAD** /~win/lehre/index.htm HTTP/1.1 CRLF Host:www.staff.hsmw.de CRLF CRLF
OPTIONS	-	-	o	Kommunikationsoptionen abfragen für eine Ressource oder den WWW-Server **OPTIONS** /~win/lehre/index.htm HTTP/1.1 CRLF Host: www.staff.hsmw.de CRLF CRLF **OPTIONS** * HTTP/1.1 CRLF Host: www.staff.hsmw.de CRLF CRLF
POST	-	o	o	E-Mails oder Daten aus einem Formular werden in der Entity übermittelt. Notwendig sind ein Content-Type und ein Content-Length-Field. **POST** /~win/lehre/post.php HTTP/1.1 CRLF Host: www.staff.hsmw.de CRLF Content-Type: application/x-www-form-urlencoded CRLF Content-Length: 7 CRLF CRLF a=1&b=2
PUT	-	-	o	Wird zur Übertragung einer Ressource vom Client zum Server verwendet. Die Ressource wird im Filesystem des Servers gespeichert. → Man kann also mittels HTTP auch ein Upload realisieren.
TRACE	-	-	o	Der Request wird vom Server zum Client zurück gesendet, inklusive der Wegpunkte. **TRACE** /~win/lehre/index HTTP/1.1 CRLF Host:www.staff.hsmw.de CRLF CRLF

WWW: TCP-Ablauf eines HTTP-Request-Response



(1) TCP-Verbindungsaufbau, eingeleitet durch Client

(2) Clt fordert von www.hsmw.de die ROOT-Ressource, Srv quittiert die Anforderung

(3) Srv sendet die angeforderte Ressource und Clt quittiert

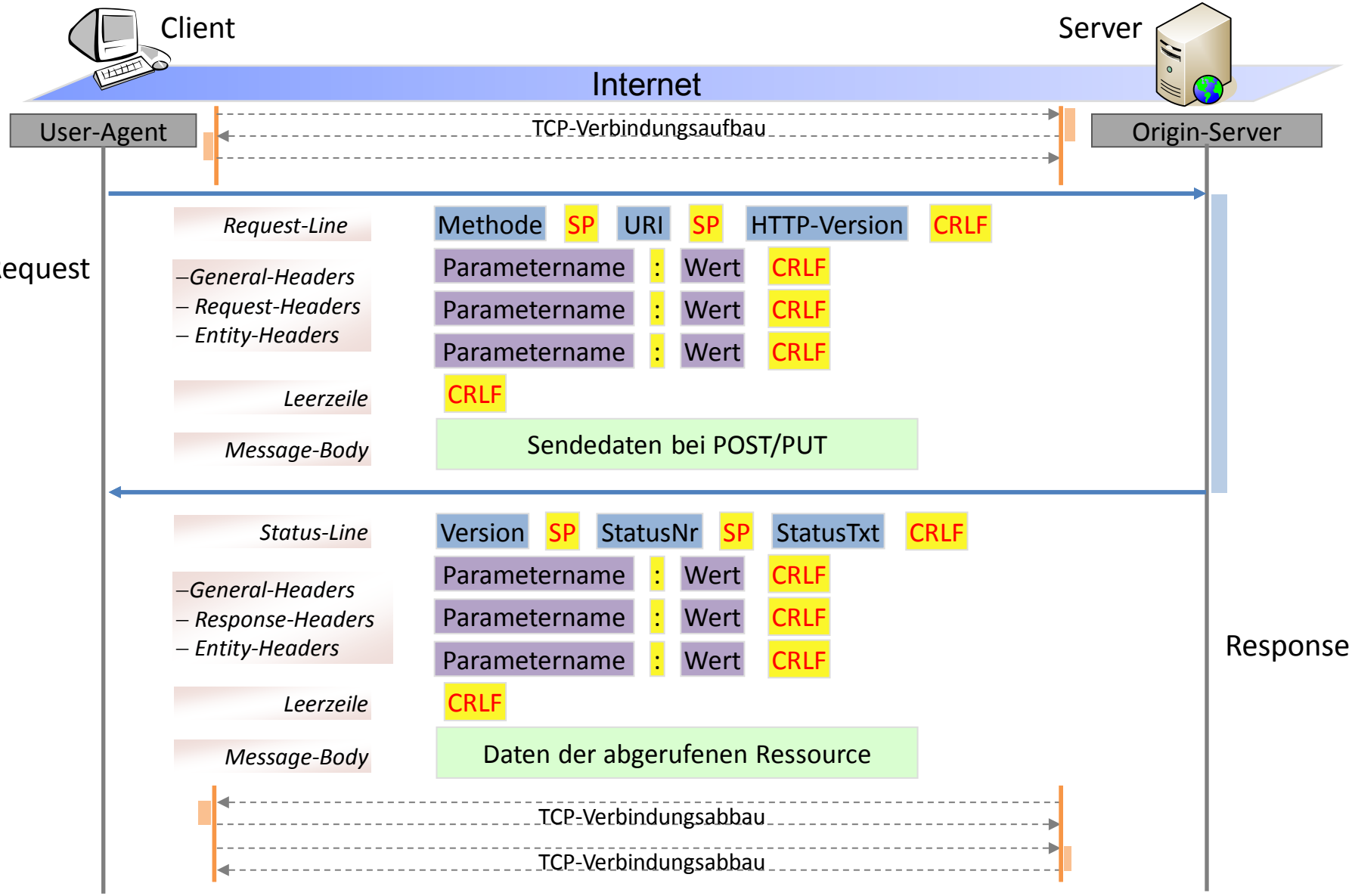
(4) Srv baut Halbverbindung zu Clt ab, Clt baut Halbverbindung zu Srv ab

WWW: HTTP-GET-Requests nach HTTP/0.9, 1.0, 1.1

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

Vers.	Request	Response
0.9	Verbindungsaufbau zu www.staff.hsmw.de:80 **GET /~win/lehre/index.htm** CRLF	<html><head> <title>Testseite</title> <link rel="stylesheet" type="text/css" href="https://www.staff.hsmw.de/~win/lehre/my.css"> </head> <body> Willkommen in: https://www.staff.hsmw.de/~win/lehre. <img hspace="16" src="images/blebul1d.gif" WIDTH="10" HEIGHT="10">eine GIF-Grafik </body></html>
1.0	Verbindungsaufbau zu www.staff.hsmw.de:80 **GET /~win/lehre/index.htm HTTP/1.0** CRLF CRLF	**HTTP/1.1 200 OK** Date: Sun, 16 Oct 2011 08:41:48 GMT Server: Apache/2.2.8 (Ubuntu) mod_ssl/2.2.8 OpenSSL/0.9.8g Last-Modified: Sun, 16 Oct 2011 08:36:14 GMT ETag: "6e4504c7-130-4af6662165b80" Accept-Ranges: bytes Content-Length: 304 Connection: close Content-Type: text/html
1.1	Verbindungsaufbau zu www.staff.hsmw.de:80 **GET /~win/lehre/index.htm HTTP/1.1** CRLF **host:www.staff.hsmw.de** CRLF CRLF	<html><head> <title>Testseite</title> <link rel="stylesheet" type="text/css" href="https://www.staff.hsmw.de/~win/lehre/my.css"> </head> <body> Willkommen in: https://www.staff.hsmw.de/~win/lehre. <img hspace="16" src="images/blebul1d.gif" WIDTH="10" HEIGHT="10">eine GIF-Grafik </body></html>

WWW: HTTP-Request-Response mit Headers



WWW: HTTP/1.1 - General Header Fields

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

- General Header Fields enthalten allgemeine Informationen, wie:
  - Cache-Informationen (Cache-Control)
  - Generierungsdatum eines Requests bzw. Responses (Date)
  - Codierungsart der Message (Transfer-Encoding)
  - Informationen zu Zwischensystemen (Via)

Header field	Requ	Resp	
Cache-Control	o	o	**`Cache-Control:no-cache | max-age=delta-seconds |...| min-fresh=delta-seconds |`** …
Connection	m	m	Verbindung nach einer Request-Response-Folge soll beendet | erhalten werden : **`Connection: close | Keep-Alive`**
Date	o	m	**`Date: Thu, 20 Oct 2011 09:19:15 GMT`**
Trailer	o	o	zusätzliche Header-Felder werden im Anhang (Trailer) gesendet.
Transfer-Encoding	o	o	Übertragungs-Codierung der Message (z.B. Verschlüsselung). Häufigste Anwendung ist aber **`Transfer-Encoding: chunked`** (in Teilen), wird bei dynamisch erzeugten Dokumenten verwendet, da man anfangs nicht weiß, wie lang das Dokument ist. Vor jedem Brocken (chunk) wird die Bytezahl in Hex angegeben. Das Ende wird mit 0x0 angezeigt.
Upgrade	o	o	Client kann damit anzeigen, welche Protokolle er neben HTTP/1.1 noch unterstützt: z.B. **`Upgrade: HTTP/2.0, SHTTP/1.3`**. Der Server kann ein Protokoll-Upgrade innerhalb eines "**`101 Switching Protocols`**"-Response mittels **`Upgrade:HTTP/2.0`** angeben.
Via	-	m	Zwischensysteme (Proxy und Gateway) müssen mittels Via-Header eine Protokoll- und Hostkennung hinterlegen, z.B.: **`Via: 1.1 www.hs-mittweida.de`**

WWW: HTTP/1.1 - Request Header Fields

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

Field	HTTP/1.1	
Accept	o	Liste der darstellbaren MIME-Typen, z.B. **`Accept:audio/basic | Accept:audio/*`** Wunschformate: **`Accept: text/plain; q=0.5, text/html`** q – Quality-Parameter gibt an, wie „gern“ man ein bestimmetes Format haben möchte. Ohne Angabe q=1, kleinster Wert=0;
Accept-Charset	o	**`Accept-Charset: iso-8859-5, unicode-1-1;q=0.8`** Alle möglichen Sets: http://www.iana.org/assignments/character-sets
Accept-Encoding	o	Akzeptierte Kompressionsmethoden des UA, z.B. **`Accept-Encodung: compress, gzip`** **`Accept-Encoding: compress;q=0.5, gzip;q=1.0`**
Accept-Language	o	**`Accept-Language: en;q=0.5, en-US;q=0.3, de`**
Authorization	o	nach Erhalt von **`401 Unauthorized`** kann sich der UA gegenüber Srv authentifizieren. Z.B.: mit der Methode "Basic" SP Base64(user ":" pwd) (RFC 2617), wenn user=alf und pwd=katze sei **`Authorization: Basic YWxmOmthdHpl`**
From	o	E-Mail-Adresse des UA-Nutzers. Z.B.: **`From: lutz.winkler@hsmw.de`**
Host	m	Host, von dem die Ressource gefordert wird. Ermöglicht Webhosting. **`Host: www.telecom.hsmw.de`**
If-Match	o	Bedingter Request: Entity-Tag auf Gleichheit prüfen (Cache-Control)
If-Modified-Since	o	Bedingter Request: z.B.: Sat, 29 Oct 1994 19:43:31 GMT (Cache-Control)
If-None-Match	o	Bedingter Request: wenn sich Entity-Tag nicht geändert hat (Cache-Control)
If-Unmodified-Since	o	Bedingter Request: z.B.: Sat, 29 Oct 1994 19:43:31 GMT (Cache-Control)
Max-Forwards	o	Bedingter Request: für TRACE | OPTIONS zur Zwischensystembeschränkung
Proxy-Authorization	o	Authentikation des UA gegenüber Proxy, "BASIC" SP Base64(user ":" pwd) mit user=alf und pwd=katze wird übergeben BASIC YWxmOmthdHpl
Range	o	Bedingter Request: zur Anforderung einer bestimmten Anzahl von Bytes
Referer	o	Seite, auf der der Link ist: z.B.: http://www.htwm.de/hsm/index.htm Ist leer, wenn man den URI direkt im Browser eingibt!
User-Agent	o	Browsername, z.B.: Mozilla/4.0 ...

WWW: HTTP/1.1 - Response Header Fields

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

Field	HTTP/1.1	
Accept-Ranges		Server unterstützt Sendung von Byte-Bereichen **`Accept-Ranges: bytes`**
Age		Altersangabe einer Ressource in delta-seconds, liegt diese innerhalb einer freshness lifetime, wird z.B. nicht neu gecacht.
E-Tag		Entity-Tag: Hashwert über Entity-Body, unabhängig vom Datum, wird vom Server berechnet. E-Tag kann strong oder weak (schwach) sein. strong= Oktettüberein-stimmung , weak =semantische Gleichheit. **`ETag: "6e4504c7-130-4af6662165b80" //ist strong`** **`ETag: "W/6e4504c7-130-4af6662165b80" //wäre weak`** Client kann bedingten Request mit dem Request-Header-Field If-None-Match mit Vergleichs-Tag absetzen. Bei Über-einstimmung, reagiert Server mit Status-Code 304, andernfalls wird die Ressource übertragen.
Location		Redirektion-URI, z.B. http://www.w3.org/pub/WWW/People.html
Proxy-Authenticate	407	Muss im Response mit **`407 Proxy Authentication Required`** enthalten sein. UA muss sich gegenüber dem Proxy ausweisen, z.B. mit **`Proxy-Authorization: BASIC YWxmOmthdHpl`**
Retry-After	503	Zeitangabe, wo Server wieder dienstbereit, z.B. 120 (in 120 s) oder ab **`Retry-After Fri, 31 Dec 1999 23:59:59 GMT`**
Server		Informationen zur Serversoftware, z.B. **`Server: Apache/2.2.8`**
WWW-Authenticate	401	Angabe der Authentikationsmethode und Bereich (realm) für den Authentication erforderlich ist. Z.B: **`WWW-Authenticate: Basic realm="intranet"`**

WWW: HTTP/1.1 - Entity Header Fields

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

Field	HTTP/1.1	
Allow	405	Anzeige der unterstützten Methoden, z.B. GET, HEAD, POST
Content-Encoding		Angabe der Codierungsart: gzipx-gzip, compress|x-compress,… ,identity (nicht kodiert).
Content-Language		Sprache, die im Body verwendet wird, z.B. de, en
Content-Length		Größe des Entity-Body in Oktetts, z.B. 1345
Content-Location		Mitteilung, Ressource ist auch noch anders erreichbar, z.B. absoluteURI | relativeURI
Content-MD5		Methode, beschrieben in RFC 1864, mit der Integrität des Entity-Body gesichert werden soll
Content-Range		Wenn z.B. von einer Ressource mit 100 Byte ein Bereich gefordert wird, z.B. **`Range: bytes=0-66`**, wird **`Content-Range: bytes 0-66/100`** lauten.
Content-Type		.Angabe des Medientypes einer Ressource, z.B. **`Content-Type: text/html; charset=ISO-8859-1`**
Expires		Angabe der Zeit, an dem eine Ressource verfällt, z.B. **`Expires: Thu, 01 Dec 1994 16:00:00 GMT`** Ist wichtig für Cache-Control.
Last-Modified		Modifikations-Zeit einer Ressource auf dem Origin-Server, z.B. **`Last-Modified: Sun, 16 Oct 2011 08:36:14 GMT`**
Extension-header		Konzept zur Einführung proprietärer Felder, ohne Standardänderung.

WWW: HTTP/1.1 - Status-Codes

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

- 5 Statuscodegruppen:
  - 1xx: Informational - Request received, continuing process
  - 2xx: Success - The action was successfully received, understood and accepted
  - 3xx: Redirection - Further action must be taken in order to complete the request
  - 4xx: Client Error - The request contains bad syntax or cannot be fulfilled
  - 5xx: Server Error - The server failed to fulfill an apparently valid request

Code	Text	
100	**Continue**	Server wartet auf Folgerequest
101	**Switching Protocols**	Mitteilung, Server verwendet zur Antwort Upgrade-Protokoll
200	**OK**	Anforderung (GET, HEAD, POST, TRACE) OK, hier ist Response
201	**Created**	Das Erzeugen einer Ressource war erfolgreich
202	**Accepted**	Das Erzeugen einer Ressource ist möglich, aber noch nicht erfolgt
204	**No Content**	Der Entity-Body der angeforderten Ressource ist leer
205	**Reset Content**	Der Request war erfolgreich (z.B. Übergabe von Formulardaten), der User-Agent kann z.B. Formulardaten zurücksetzen
206	**Partial Content**	Antwort auf bedingtes GET mit Range-Header-Field

WWW: HTTP/1.1 - Status-Codes

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

Code		
300	**Multiple Choices**	Die angeforderte Ressource ist auf mehreren Servern vorhanden, Auswahl des Servers ist möglich
301	**Moved Permanently**	Ressource permanent umgezogen
302	**Found**	Ressource temporär umgezogen, im Location-Field: temporärer URI. Der nächste Request auf Ressource, sollte auf Original-URI erfolgen (←→ 307)
303	**See Other**	The response to the request can be found under a different URI and SHOULD be retrieved using a GET method on that resource
304	**Not Modified**	Antwort auf bedingtes GET
305	**Use Proxy**	Die angeforderte Ressource muss beim Proxy (Location-Field) abgerufen werden.
307	**Temporary Redirect**	Ressource temporär umgezogen, im Location-Field: temporärer URI. Der nächste Request auf Ressource, sollte auf Temporary-URI erfolgen
500	**Internal Server Error**	
501	**Not Implemented**	Methode nicht implementiert
502	**Bad Gateway**	
503	**Service Unavailable**	
504	**Gateway Time-out**	
505	**HTTP Version not supported**	

WWW: HTTP/1.1 - Status-Codes

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

Code		
400	**Bad Request**	

401	**Unauthorized**	Client ist nicht berechtigt auf Ressource zuzugreifen
		`HTTP/1.1 401 Access Denied`
		`Server: Microsoft-IIS/5.0`
		`Date: Fri, 08 Dec 2006 16:33:55 GMT`
		`WWW-Authenticate: Basic realm="141.55.242.66"`

402	**Payment Required**	für künftige Anwendungen
403	**Forbidden**	Ausführung der Methode verweigert
404	**Not Found**	
405	**Method Not Allowed**	Ausführung der Methode auf dieser Ressource nicht möglich
406	**Not Acceptable**	Header-Informationen nicht akzeptabel
407	**Proxy Authentication Required**	künftig
408	**Request Time-out**	Methode konnte in bestimmter Zeitspanne nicht ausgeführt werden
409	**Conflict**	Ressource soll geändert werden. Änderung ist älter (bei PUT).
410	**Gone**	Ressource ist nicht mehr vorhanden.
411	**Length Required**	Längenangabe erforderlich
413	**Request Entity Too Large**	Entity-Body zu groß
414	**Request-URI Too Large**	URI zu lang (>2000 Zeichen)
415	**Unsupported Media Type**	Angeforderter Medientyp nicht unterstützt
416	**Requested range not satisfiable**	Angeforderter Byte-Bereich nicht möglich

WWW-Session: **HTTP/1.0 - GET**

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

c:\>**telnet CRLF**
Microsoft Telnet>**open www.staff.hs-mittweida.de 80 CRLF**
Verbindungsaufbau zu www.staff.hs-mittweida.de 80 ...

(1) TELNET-Session

(2) HTTP-"Session"

```
GET /~win/lehre/index.htm HTTP/1.0 CRLF CRLF                    ;Request-Line
```

```
HTTP/1.1 200 OK                                                 ;Status-Line
Date: Thu, 20 Oct 2011 12:42:35 GMT                             ;Resp.-Header
Server: Apache/2.2.8 (Ubuntu) mod_ssl/2.2.8 OpenSSL/0.9.8g      ;Resp.-Header
Last-Modified: Sun, 16 Oct 2011 08:36:14 GMT                    ;Entity-Header
ETag: "6e4504c7-130-4af6662165b80"                              ;Entity-Header
Accept-Ranges: bytes                                            ;Resp.-Header
Content-Length: 304                                             ;Entity-Header
Content-Type: text/html                                         ;Entity-Header
<html>                                                          ;Entity
<head>
<title>Testseite</title>
<link rel="stylesheet" type="text/css"
href="https://www.staff.hsmw.de/~win/lehre/my.css">
</head>
<body>
<h3>Willkommen in: https://www.staff.hsmw.de/~win/lehre.</h3>
<img hspace="16" src="images/blebul1d.gif" WIDTH="10" HEIGHT="10">eine GIF-Grafik
</body>
</html>
```

Verbindung zu Host verloren.

Welches Ergebnis erhält man bei Verwendung der Protokollvariante 0.9?

WWW-Session: **HTTP/1.1 - GET**

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

(1) TELNET-Session

(2) HTTP-"Session"

```
c:\>telnet CRLF
Microsoft Telnet>open www.staff.hs-mittweida.de 80 CRLF
Verbindungsaufbau zu www.staff.hs-mittweida.de 80 ...
```

```
GET /~win/lehre/varab.php?a=11&b=99 HTTP/1.1 CRLF
Host: www.staff.hs-mittweida.de CRLF
CRLF
```

```
HTTP/1.1 200 OK
Date: Sun, 23 Oct 2011 08:19:43 GMT
Server: Apache/2.2.8 (Ubuntu) mod_ssl/2.2.8 OpenSSL/0.9.8g
X-Powered-By: PHP/5.2.4-2ubuntu5.17
Transfer-Encoding: chunked
Content-Type: text/html

c9                                                  ;1. Teil Länge= 0xc9 = 201
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
       <head>
          <title>Test von POST</title>
          <meta http-equiv="Content-type" content="text/html; charset=utf-8" />
       </head><body>

1d                                                  ;2. Teil Länge= 0x1d = 29

Wert der Variablen a: 11<br/>
18                                                  ;3. Teil Länge= 0x18 = 24
Wert der Variablen b: 99

12
                                                    ;4. Teil Länge= 0x12 =18
</body></html>

0                                                   ;EndeMarker = 0x0
```

WWW-Session: **HTTP/1.1 - GET**

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

(1) TELNET-Session

(2) HTTP-"Session"

```
c:\>telnet CRLF
Microsoft Telnet>open www.staff.hs-mittweida.de 80 CRLF
Verbindungsaufbau zu www.staff.hs-mittweida.de 80 ...
```

```
GET /~win/lehre/index.htm HTTP/1.1 CRLF                    ;Request-Line
Host: www.staff.hs-mittweida.de CRLF                       ;Requ.-Header
CRLF
```

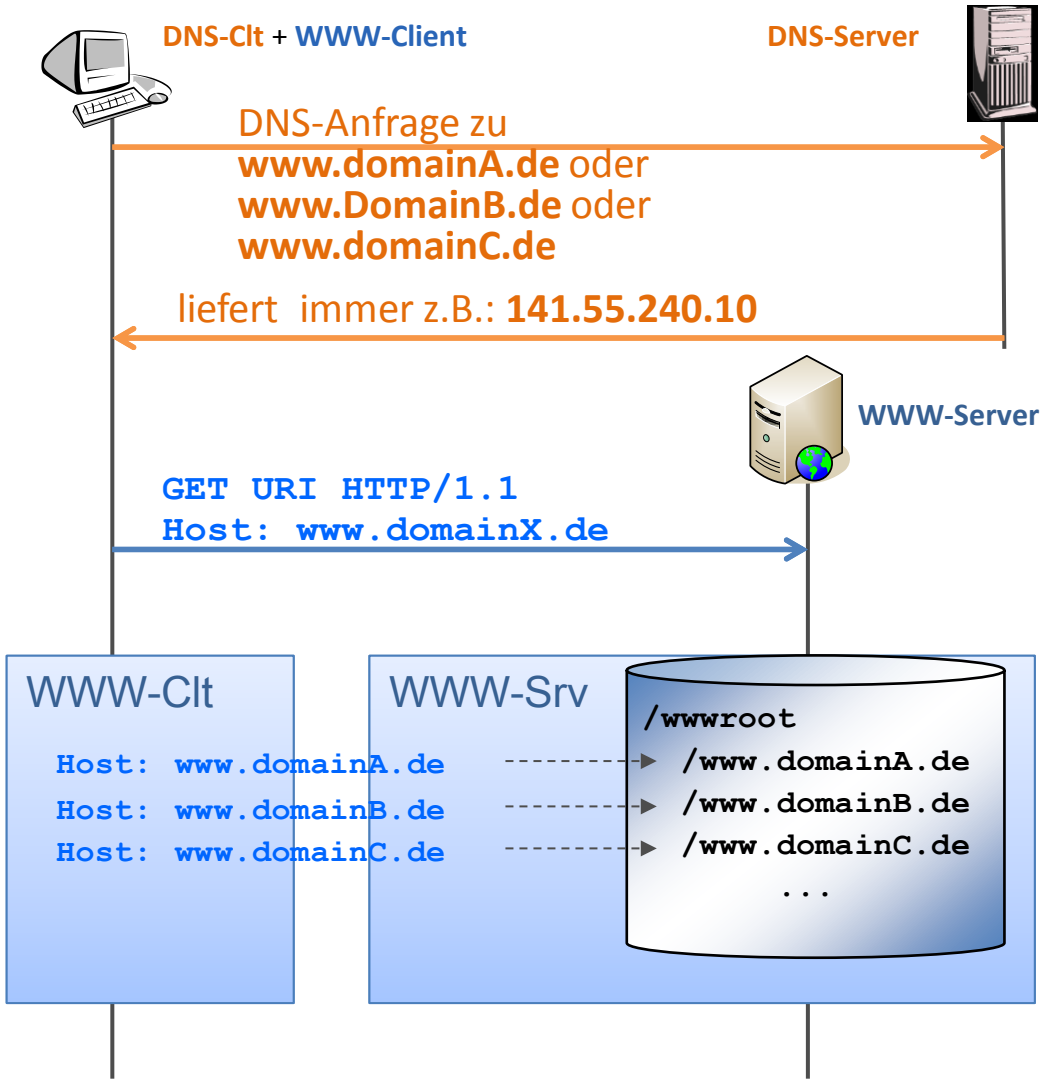
```
HTTP/1.1 200 OK                                            ;Status-Line
Date: Thu, 20 Oct 2011 12:42:35 GMT                        ;Resp.-Header
Server: Apache/2.2.8 (Ubuntu) mod_ssl/2.2.8 OpenSSL/0.9.8g ;Resp.-Header
Last-Modified: Sun, 16 Oct 2011 08:36:14 GMT               ;Entity-Header
ETag: "6e4504c7-130-4af6662165b80"                         ;Entity-Header
Accept-Ranges: bytes                                       ;Resp.-Header
Content-Length: 304                                        ;Entity-Header
Content-Type: text/html                                    ;Entity-Header

<html>                                                     ;Entity
<head>
<title>Testseite</title>
<link rel="stylesheet" type="text/css"
href="https://www.staff.hsmw.de/~win/lehre/my.css">
</head>
<body>
<h3>Willkommen in: https://www.staff.hsmw.de/~win/lehre.</h3>
<img hspace="16" src="images/blebul1d.gif" WIDTH="10" HEIGHT="10">eine GIF-Grafik
</body>
</html>
```

Verbindung zu Host verloren.

WWW: HTTP/1.1 - GET mit `Host: www.DomainX.yy`

- **Webhosting**:
  - mehrere Domains,
  - auf einem Server,
  - alle haben gleiche IP-Adresse.
- Im dargestellten Beispiel sind auf dem Host die Webs von drei virtuellen Domäns untergebracht.
- Der DNS-Client erhält also für alle drei Domains die gleiche IP-Adresse.

- **Wie kann der Web-Client gezielt auf eins dieser drei Webangebote zugreifen???**
  - Mittels HTTP1.1-Request-Header `Host: www.domainB.de` kann man den "virtuellen" Domainnamen angeben.
  - Der Web-Server greift auf das entsprechende Verzeichnis zu.



WWW: HTTP/1.1 - GET mit Connection:Keep-alive

c:\>telnet CRLF
Microsoft Telnet>open www.staff.hs-mittweida.de 80 CRLF
Verbindungsaufbau zu www.staff.hs-mittweida.de 80 ...

(1) TELNET-Session

(2) HTTP-"Session"

```
GET /~win/lehre/index.htm HTTP/1.1 CRLF          ;Request-Line
Host: www.staff.hs-mittweida.de CRLF             ;Requ.-Header
Connection: Keep-Alive     CRLF CRLF             ;Gener.-Header
```

```
HTTP/1.1 200 OK
Date: Fri, 21 Oct 2011 08:30:21 GMT
Server: Apache/2.2.8 (Ubuntu) mod_ssl/2.2.8 OpenSSL/0.9.8g
Last-Modified: Sun, 16 Oct 2011 08:36:14 GMT
ETag: "6e4504c7-130-4af6662165b80"
Accept-Ranges: bytes
Content-Length: 304
Keep-Alive: timeout=15, max=100
Connection: Keep-Alive
Content-Type: text/html

<html>
<head>
<title>Testseite</title>
<link rel="stylesheet" type="text/css"
href="https://www.staff.hsmw.de/~win/lehre/my.css">
</head>
<body>
<h3>Willkommen in: https://www.staff.hsmw.de/~win/lehre.</h3>
<img hspace="16" src="images/blebul1d.gif" WIDTH="10" HEIGHT="10">eine GIF-Grafik
</body>
</html>
```

Verbindung zu Host verloren.

WWW: HTTP/1.1 – GET `Connection:close|Keep-Alive`

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

`Connection: close`

(1) TCP-Verbindungsaufbau, eingeleitet durch Client

(2) Clt fordert von Srv Ressource index.htm, Srv quittiert

(3) Srv sendet Ressource und Clt quittiert

(4) Srv baut sofort Halbverbindung zu Clt ab, Clt baut Halbverbindung zu Srv ab

`Connection: keep-alive`

(1) TCP-Verbindungsaufbau,

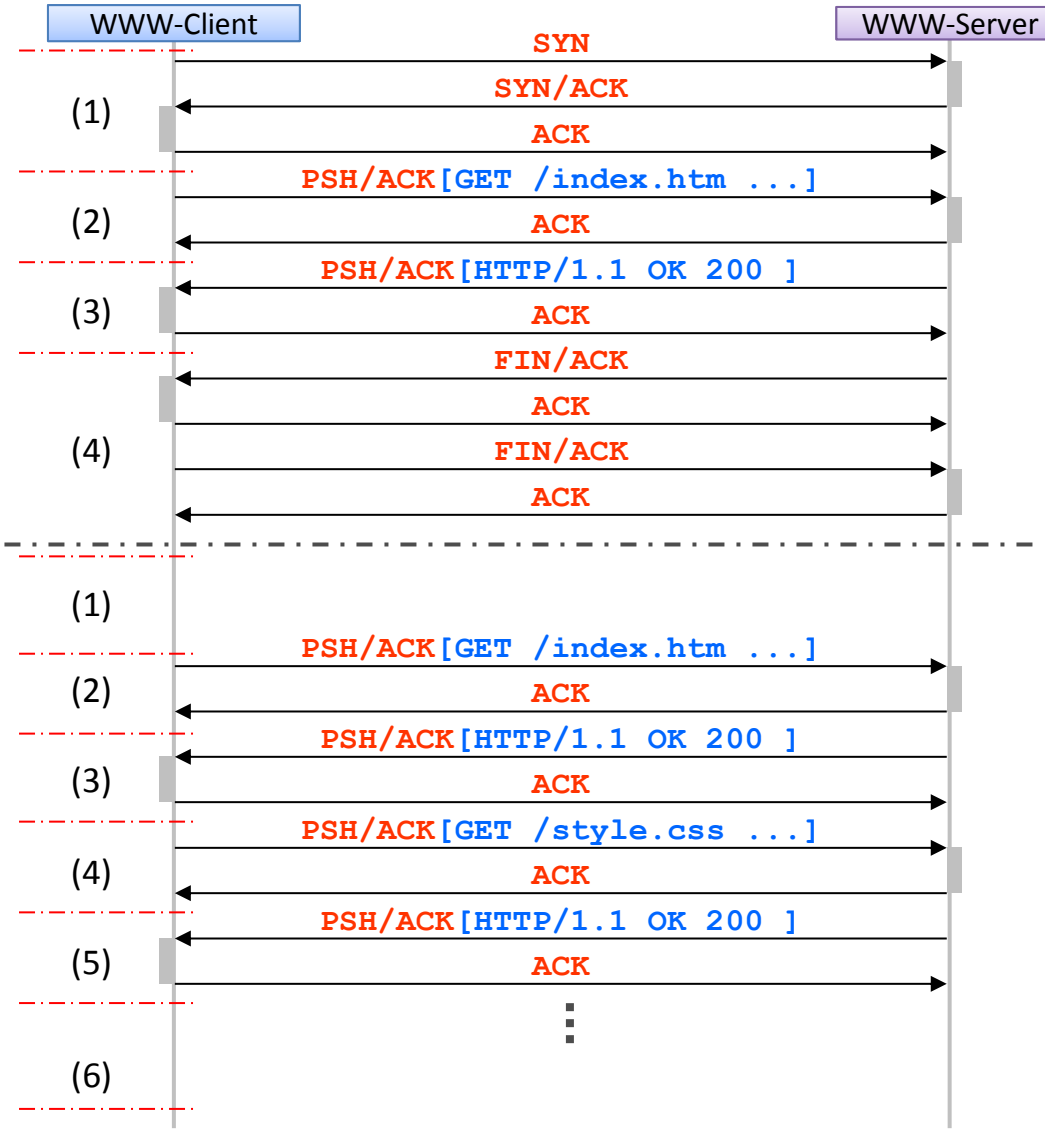
(2) Clt fordert 1. Ressource, Srv quittiert

(3) Srv sendet 1. Ressource, Clt quittiert

(4) Clt fordert 2. Ressource, Srv quittiert

(5) Srv sendet 2. Ressource, Clt quittiert

(6) Srv baut nach Wartezeit Halbverbindung zu Clt ab, Clt baut Halbverbindung zu Srv ab



WWW: HTTP/1.1 - Conditional GET

- Bedingtes GET kann durch Request-Header gesteuert werden, wie z.B.:
  - **If-Match: "5a0d5-12d-457ab1dd"** //Liefere, wenn ETag identisch mit "5a..."
  - **If-None-Match: "5a0d5-12d-457ab1dd"** //Liefere, wenn ETag nicht "5a..." ist
  - **If-Modified-Since: Sun, 16 Oct 2011 08:36:14 GMT**

c:\>**telnet CRLF**
Microsoft Telnet>**open www.staff.hs-mittweida.de 80 CRLF**
Verbindungsaufbau zu www.staff.hs-mittweida.de 80 ...

(1) TELNET-Session

(2) HTTP-"Session"

```
GET /~win/lehre/index.htm HTTP/1.1 CRLF
Host: www.staff.hs-mittweida.de CRLF
If-Modified-Since: Sun, 16 Oct 2011 08:36:14 GMT CRLF
CRLF
```

```
HTTP/1.1 304 Not Modified
Date: Fri, 21 Oct 2011 08:59:31 GMT
Server: Apache/2.2.8 (Ubuntu) mod_ssl/2.2.8 OpenSSL/0.9.8g
ETag: "6e4504c7-130-4af6662165b80"
```

Was passiert, wenn Sie **GET /~win/lehre/index.htm** mit folgende Conditions nutzen?

(1) If-Match: "6e4504c7-130-4af6662165b80"

(2) If-Non-Match: "6e4504c7-130-4af6662165b80"

WWW: HTTP/1.1 - Partial GET

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

- Teilweises GET kann gesteuert werden durch den Request-Header **Range:**
  - **Range: bytes=0-20** //von Byte 0 bis 20
  - **Range: bytes=25-** //ab Byte 25 den Rest
  - **Range: bytes=-25** //die letzten 25 Byte einer Ressource

```
c:\>telnet CRLF
Microsoft Telnet>open www.staff.hs-mittweida.de 80 CRLF
Verbindungsaufbau zu www.staff.hs-mittweida.de 80 ...
```

(1) TELNET-Session

```
GET /~win/lehre/index.htm HTTP/1.1 CRLF
Host: www.staff.hs-mittweida.de CRLF
Range: bytes=0-8 CRLF
CRLF
```

(2) HTTP-"Session"

```
HTTP/1.1 206 Partial Content
Date: Fri, 21 Oct 2011 09:35:22 GMT
Server: Apache/2.2.8 (Ubuntu) mod_ssl/2.2.8 OpenSSL/0.9.8g
Last-Modified: Sun, 16 Oct 2011 08:36:14 GMT
ETag: "6e4504c7-130-4af6662165b80"
Accept-Ranges: bytes
Content-Length: 9
Content-Range: bytes 0-8/304
Content-Type: text/html

<html>                                  ;3c 68 74 6d 6c 3e 0a
<h                                      ;3c 68
```

WWW: HTTP/1.1 - HEAD

c:\>**telnet CRLF**
Microsoft Telnet>**open www.staff.hs-mittweida.de 80 CRLF**
Verbindungsaufbau zu www.staff.hs-mittweida.de 80 ...

(1) TELNET-Session

(2) HTTP-"Session"

```
HEAD /~win/lehre/index.htm HTTP/1.1 CRLF
Host: www.staff.hs-mittweida.de CRLF
CRLF
```

```
HTTP/1.1 200 OK
Date: Fri, 21 Oct 2011 15:10:03 GMT
Server: Apache/2.2.8 (Ubuntu) mod_ssl/2.2.8 OpenSSL/0.9.8g
Last-Modified: Sun, 16 Oct 2011 08:36:14 GMT
ETag: "6e4504c7-130-4af6662165b80"
Accept-Ranges: bytes
Content-Length: 304
Content-Type: text/html
```

WWW: HTTP/1.1 - OPTIONS

```
c:\>telnet CRLF
Microsoft Telnet>open www.staff.hs-mittweida.de 80 CRLF
Verbindungsaufbau zu www.staff.hs-mittweida.de 80 ...
```

(1) TELNET-Session

(2) HTTP-"Session"

```
OPTIONS /~win/lehre/index.htm HTTP/1.1 CRLF
Host: www.staff.hs-mittweida.de CRLF
CRLF
```

```
HTTP/1.1 200 OK
Date: Fri, 21 Oct 2011 15:15:40 GMT
Server: Apache/2.2.8 (Ubuntu) mod_ssl/2.2.8 OpenSSL/0.9.8g
Allow: GET,HEAD,POST,OPTIONS,TRACE
Content-Length: 0
Content-Type: text/html
```

```
OPTIONS /~win/ HTTP/1.1 CRLF
Host: www.staff.hs-mittweida.de CRLF
CRLF
```

```
HTTP/1.1 200 OK
Date: Fri, 21 Oct 2011 15:17:34 GMT
Server: Apache/2.2.8 (Ubuntu) mod_ssl/2.2.8 OpenSSL/0.9.8g
Allow: GET,HEAD,POST,OPTIONS,TRACE
Content-Length: 0
Content-Type: text/html
```

WWW: HTTP/1.1 - TRACE

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

```
c:\>telnet CRLF
Microsoft Telnet>open www.staff.hs-mittweida.de 80 CRLF
Verbindungsaufbau zu www.staff.hs-mittweida.de 80 ...
```

(1) TELNET-Session

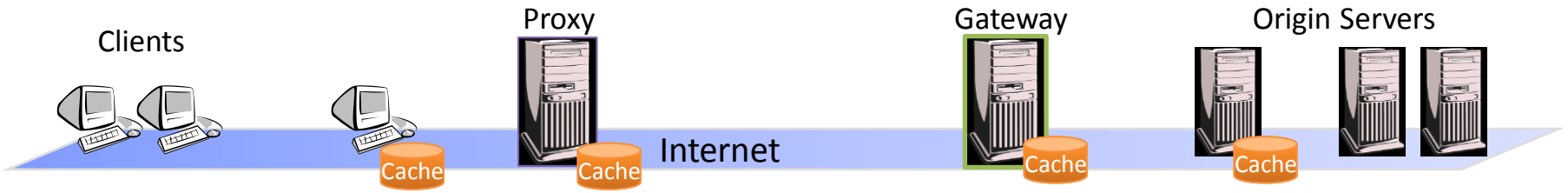
(2) HTTP-"Session"

```
TRACE /~win/lehre/index.htm HTTP/1.1 CRLF
Host: www.staff.hs-mittweida.de CRLF
CRLF
```

```
HTTP/1.1 200 OK
Date: Fri, 21 Oct 2011 15:19:56 GMT
Server: Apache/2.2.8 (Ubuntu) mod_ssl/2.2.8 OpenSSL/0.9.8g
Transfer-Encoding: chunked                    ;Sendung in Teilen
Content-Type: message/http

49                                            ;Länge 0x49 = 73
TRACE /~win/lehre/index.htm HTTP/1.1
Host: www.staff.hs-mittweida.de



0                                             ;0x0 Ende der Sendung in Teilen
```

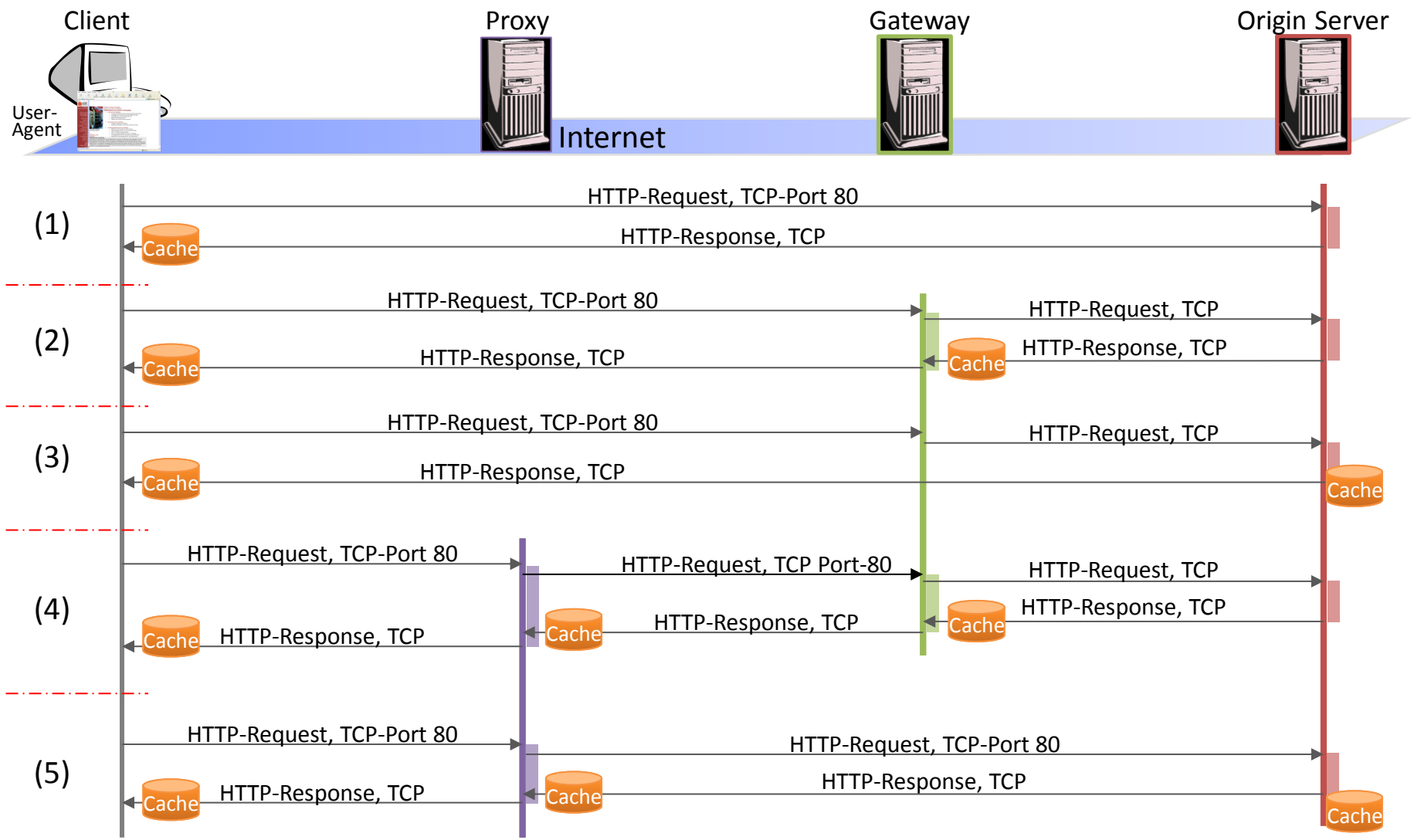
WWW: Cach - Das Problem

- Exponierte Ressourcen werden häufig frequentiert →Mehrfachübertragung →Netzlast →Wartezeiten.



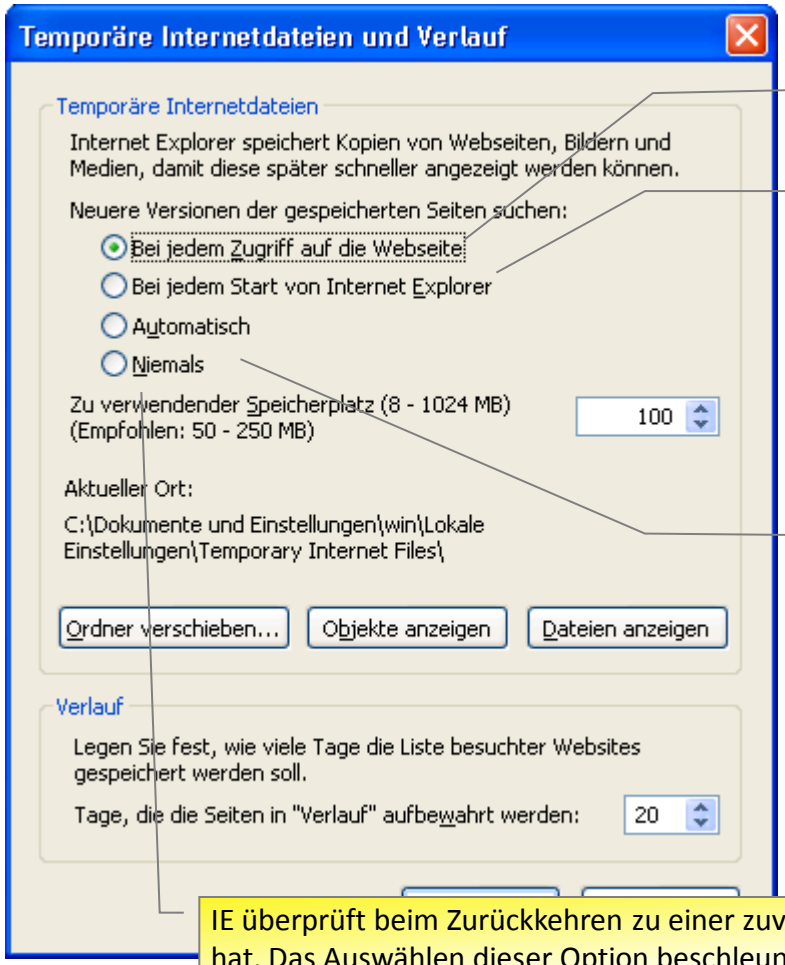
- Speicherung im Cache kann Problem teilweise mindern →Unterscheidung:
  - **Gateway- oder Servercache**: geringere Zugriffszeit, Mehrfachübertragung bleibt.
  - **Proxycache**: geringere Zugriffszeit, Mehrfachübertragung entfällt.
    Aber: Duplikaterzeugung, Aktualitätsproblem.
  - **Clientcache**: geringe Zugriffszeit, keine Mehrfachübertragung, Aktualitätsproblem.
- HTTP-Konzepte für Cache-Steuerung
  - Basis: Ezeugungs- bzw. Aktualisierungsdatum und Entity-Tag einer Ressource
  - Cache-Control-Field im General Header
  - Bedingte Requests durch Request-Header-Field

WWW: Cache - Szenarien



WWW: Cache - Browsereistellungen

- IE8, IE7: Menü Extras →Internetoptionen →Allgemein →Browserverlauf



IE prüft beim Zurückkehren zu einer zuvor angezeigten Seite, ob sich diese geändert hat. Hat sich Seite geändert, zeigt IE die neue Seite an und speichert sie im Ordner Temporäre Internetdateien.

Gibt an, dass IE beim Zurückkehren zu einer zuvor angezeigten Seite nicht überprüft, ob diese sich seit dem letzten Anzeigen geändert hat. IE überprüft nur dann auf neue Inhalte, wenn Sie zu einer Seite zurückkehren, die in einer vorhergehenden Sitzung mit IE oder an einem vorhergehenden Tag angezeigt wurde.
Wenn Sie die aktuellste Version einer Seite anzeigen möchten, auch wenn diese Option ausgewählt ist, klicken Sie "Aktualisieren".

IE überprüft nicht beim Zurückkehren zu einer bereits zuvor Seite, ob diese sich seit dem letzten Anzeigen geändert hat.
IE überprüft nur dann auf neue Inhalte, wenn Sie zu einer Seite zurückkehren, die in einer vorhergehenden Sitzung mit IE oder an einem vorhergehenden Tag angezeigt wurde. Wenn IE mit der Zeit festgestellt, dass die Bilder auf dieser Seite sich selten ändern, wird noch seltener auf neue Bilder überprüft.
Auswahl der aktuellsten Version: "Aktualisieren".

IE überprüft beim Zurückkehren zu einer zuvor angezeigten Seite niemals, ob diese sich seit dem letzten Anzeigen geändert hat. Das Auswählen dieser Option beschleunigt das Wechseln zwischen Seiten, die Sie bereits angezeigt haben.
Die aktuellste Version einer Seite erhält man, auch wenn diese Option ausgewählt ist, mit "Aktualisieren".

WWW: **Cookies** /RFC 6265/

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

- Die **Kommunikation** auf WWW-Anwendungsebene **ist zustandslos**. Es gibt kein Gedächtnis, welcher UA, was und wann von einem Server wollte.
- Dadurch ist das Protokoll einfach und einigermaßen Privatheit gegeben.
- Cookies wurden ursprünglich von Netscape eingeführt, mit dem Ziel:
  - auf dem UA eine Mitteilung zu hinterlassen
  - die zu einem späteren Zeitpunkt zum Server mit gesendet wird, woraus dieser auf vorhergehende Zugriffe schließen kann.
- Cookies:
  - **sind Textinformation von max. 4000 Byte**, vom Server an den UA übermittelt,
  - Mittels Cookie kann man Zustände speichern, wodurch eine zustandsbasierte Session mittels des zustandslosen HTTP-Protokolls möglich wird.
  - **pro Domain** können **50 Cookies** gespeichert werden,
  - Browser senden Cookies beim Aufruf dieser Domain im Header mit,
  - **Ein Browser** soll mindestens **3000 Cookies speichern können**.
- Cookies erhält man vom Server (in einem Response) mittels Header "Set-Cookie" :

```
Set-Cookie: Name=Value; expires=DATE; path=PATH; domain=DOMAIN_NAME; secur
```

- Cookies werden mittels Header "Cookie" an den Server gesendet:

```
Cookie: Name=Value
```

WWW: **Cookies** /RFC 6265/

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

```
Set-Cookie: Name=Value; expires=DATE; path=PATH; domain=DOMAIN_NAME; secure
```

Name	Zeichenkette (außer ; , SP), z.B. MyCookie1
Value	Eine Kundennummer, ein Nutzerkonto, ein Absprungzähler, ein Warenkorb-Id, …
expires	Damit legt der Server die Gültighkeitsdauer fest: Wochentag, DD-MM-YYYY HH:MM:SS GMT. Wenn expires leer ist, gilt er eine Session. Mit dem Schließen des Browsers wird auch der Cookie gelöscht.
path	Cookie wird nur zum Server gesendet beim Zugriff auf Ressourcen, die unter diesem Pfad liegen
domain	Cookie wird nur zum Server gesendet beim Zugriff auf Ressourcen, die unter dieser Domain liegen, d.h. auch die Subdomains. hs-mittweida.de schließt auch staff.hs-mittweida.de ein usw.
secure	Ist "secure" im Set-Cookie vorhanden , wird dieser nur dann übertragen, wenn zum Server eine sichere Verbindung besteht (HTTPS).

Beispiele: Server sendet an UA	**UA sendet an Server**
Set-Cookie: SID=31d4d96e407aad42	**Cookie:** SID=31d4d96e407aad42
Set-Cookie: SID=31d4d96e407aad42; Path=/; Secure **Set-Cookie:** lang=en-US; Path=/; Domain=example.com	**Cookie:** SID=31d4d96e407aad42; lang=en-US
Set-Cookie: MyCookie1=expires+10s++path+win+lehre ++domain+hsmw+de; expires=Mon, 24-Oct-2011 11:05:23 GMT; path=/~win/lehre/; domain=hsmw.de **Set-Cookie:** MyCookie2=expires+session	**Cookie:** MyCookie1=expires+10s++path+win+lehre++domain+hsmw+de; MyCookie2=expires+session

WWW: Cookies - Praxis

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

- In **http://www.staff.hs-mittweida.de/~win/lehre/cookies.php** befindet sich folgender PHP- und HTML-Text

```
<?php //--cookies setzen
setcookie("MyCookie1",'Wert_abcd',time()+10,"/~win/lehre/","hs-mittweida.de",0);
setcookie("MyCookie2",'Wert_xyz',0,"","",0); ?>
<html><head><title>cookies</title></head><body>
<?php //--cookiename und -werte ausgeben, stehen im array _COOKIE
echo 'Folgende Cookies wurden vom UA an den Server gesendet: <br/>';
foreach($_COOKIE as $key => $value) {
   echo $key . ': ' . $value . '<br/>'; }?>
</body></html>
```

(1) Löschen Sie im UA alle Cookies:
   - IE7/8: Extras >Internetoptionen >Allgemein >Löschen >Löschen
   - MF3: Extras >Einstellungen > Datenschutz > einzelne Cookies

(2) Rufen Sie **http://www.staff.hs-mittweida.de/~win/lehre/cookies.php** auf:

```
Folgende Cookies wurden vom UA an den Server gesendet:
```

(3) Rufen Sie nochmal **http://www.staff.hs-mittweida.de/~win/lehre/cookies.php** auf

```
Folgende Cookies wurden vom UA an den Server gesendet:
MyCookie1: Wert_abcd
MyCookie2: Wert_xyz
```

(4) Kontrollieren Sie anschließend die Einträge:
   - IE7: Extras > Allgemein > Browserverlauf > Einstellungen > Daten anzeigen
   - MF3: Extras > Internetoptionen>Einstellungen > Datenschutz > einzelne Cookies

WWW: Cookies – Set-Cookie, Cookie

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

ApplicationProtocolExplorer: **www.staff.hs-mittweida.de, Port 80**

```
GET /~win/lehre/cookies.php HTTP/1.0 CRLF CRLF
```

```
HTTP/1.1 200 OK
Date: Tue, 25 Oct 2011 09:04:18 GMT
Server: Apache/2.2.8 (Ubuntu) mod_ssl/2.2.8 OpenSSL/0.9.8g
Set-Cookie: MyCookie1=Wert_abcd; expires=Tue, 25-Oct-2011 09:05:18 GMT; path=/~win/lehre/;
    domain=hsmw.de
Set-Cookie: MyCookie2=Wert_xyz
Connection: close
Content-Type: text/html

<html><head><title>cookies</title></head><body>
Folgende Cookies wurden vom UA an den Server gesendet: <br/>
</body></html>
```

```
GET /~win/lehre/cookies.php HTTP/1.0 CRLF
Cookie: MyCookie2=Wert_xyz; MyCookie1=Wert_abcd CRLF CRLF
```

```
HTTP/1.1 200 OK
Date: Tue, 25 Oct 2011 09:04:18 GMT
Set-Cookie: MyCookie1=Wert_abcd; expires=Tue, 25-Oct-2011 09:06:03 GMT; path=/~win/lehre/; domain=hsmw.de
Set-Cookie: MyCookie2=Wert_xyz
Connection: close
Content-Type: text/html

<html>
<head><title>cookies</title></head>
<body>
Folgende Cookies wurden vom UA an den Server gesendet: <br/>MyCookie2: Wert_xyz<br/>
MyCookie1: Wert_abcd<br/></body></html>
```

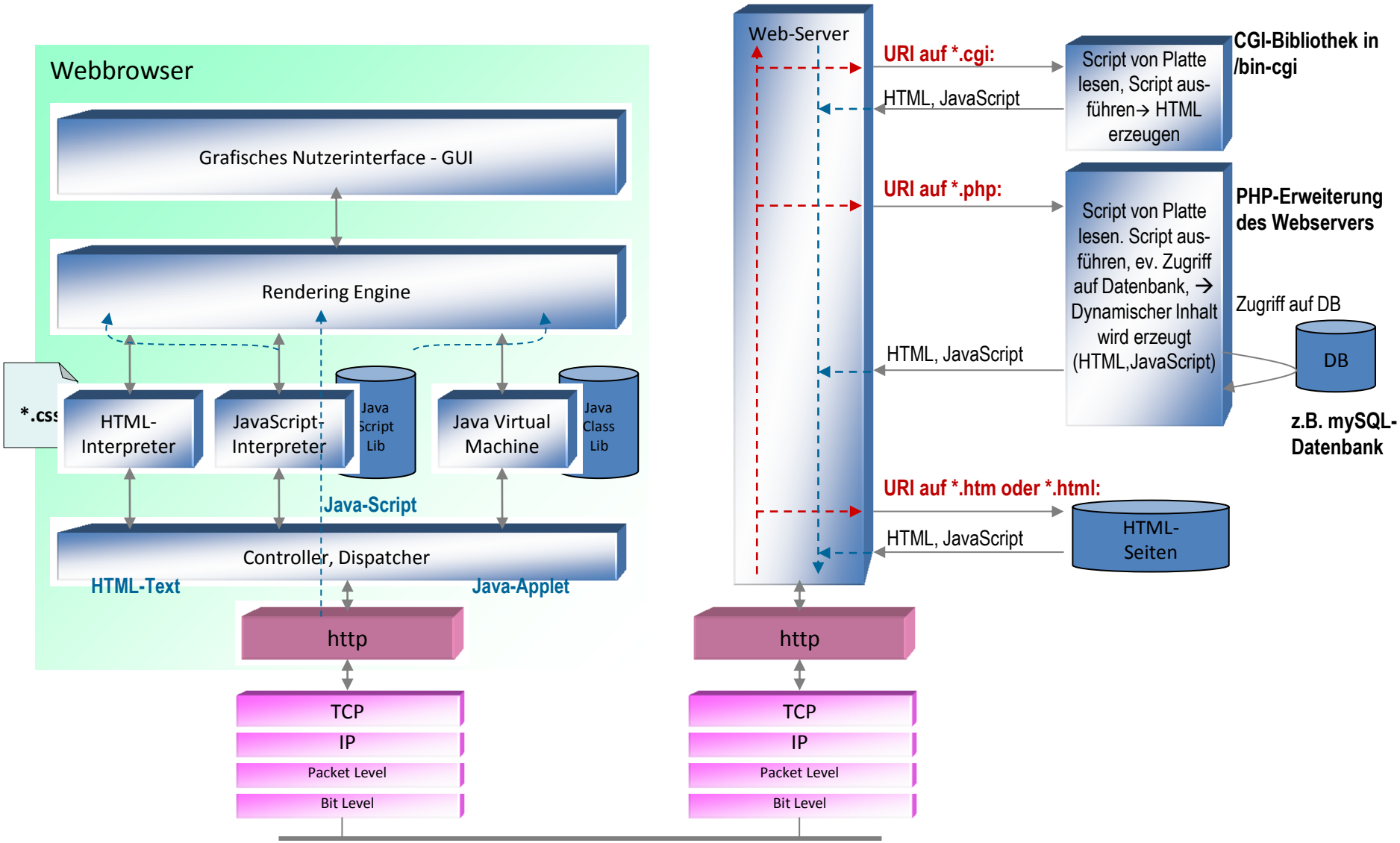
WWW: Dynamische Seiten - Was ist das?

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

- Statische Seiten:
  - in HTML-codierte Seiten, die von einem Server abrufbar sind.
  - Zugriff erfolgt lesend, Interaktionen nicht möglich.
- Dynamische Seiten:
  - Gesamtheit oder Teile einer HTML-Seite werden beim Aufruf generiert und z.B. mit Informationen aus Datenbanken versehen.
  - Interaktionen durch Formulare auf der Clientseite möglich. Formularinhalte werden mittels GET oder POST zum Server übermittelt.

- Die Realisierung dynamischer Seiten kann erfolgen:
  - Clientseitig, mit eingeschränkter Funktionalität:
    - Java-Applet, Ausführung in einem Bereich der Webseite durch die "Java Virtual Machine" als Browsererweiterung.
    - Java-Script, ausgeführt durch den "Java-Script-Interpreter", als Browser-Plug-In.
  - Serverseitig, mit mächtiger Funktionalität:
    - CGI (Common Gateway Interface)-Scripte,
    - Aktive Serverseiten (ASP – Active Server Pages, PHP – Hypertext Preprocessor),
    - Servlets (serverseitige Java-Erweiterung).

WWW: Dynamische Seiten – Wie geht das?



WWW: Dynamische Seiten – PHP-Beispiel

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

- Sehr aktuelle und mächtige Scriptsprache. Viele gute Webangebote:
  - http://www.php-homepage.de/
  - http://www.php3.com
  - http://www.selfphp4.de/index.php4

Beispiel: Formular mit Auswertung

(1) Mounten Sie Ihren Webspace \\samba.hs-mittweida.de
Ist ihr Host außerhalb der Domäne hs-mittweida.de, muss zuerst eine VPN-Verbindung hergestellt werden.

(2) Legen Sie die Seite formular.php an

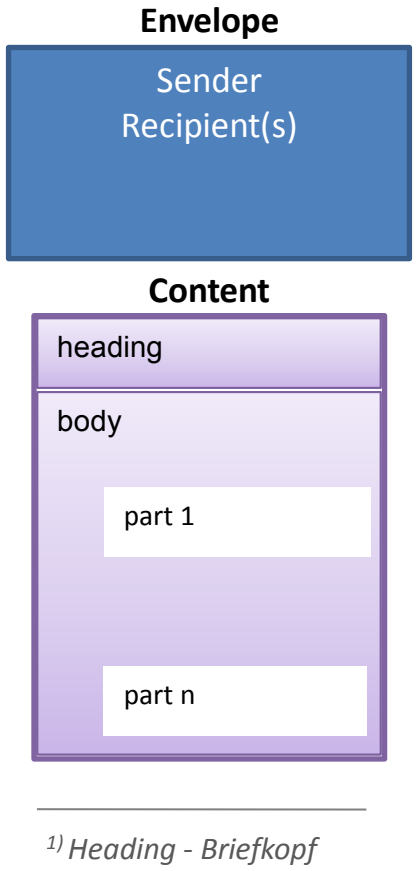
(3) Schreiben Sie den nebenstehenden Quelltext

(4) Wie die Seite aussehen sollte, sehen Sie unter:
www.staff.hs-mittweida.de/~win/lehre/formular.php

```
<html>
 <head>
   <title>Formulare und PHP</title>
 </head>
 <body>
  <form method="GET" action="<?PHP echo $PHP_SELF ?>" >
  <input type="text" name="T1" size="20"> Name
  <input type="text" name="T2" size="20"> Vorname<br/>
  <input type="radio" value="11" name="R1" checked> 20-40
  <input type="radio" value="12" name="R1">41-60
  <input type="radio" value="13" name="R1"> älter als 61<br/>
  <input type="submit" value="Abschicken" name="B1">
  <input type="reset" value="Zurücksetzen" name="B2">
  </form>
<?PHP      echo "Name: ",$T1;
           echo "<br>Vorname: ",$T2;
           switch ($R1) {
                    case 11: echo "<br>Alter: 20-40"; break;
                    case 12: echo "<br>Alter: 41-60"; break;
                    case 13: echo "<br>Alter: älter als 60";
break;
           } ?>
 </body></html>
```

E-Mail: **Electronic-Mail - Steckbrief**

- 1982 wurde der " STANDARD FOR THE FORMAT OF ARPA INTERNET TEXT MESSAGES" im **RFC 822** veröffentlicht.
- Darin wurde festgelegt, dass eine Mail aus dem Umschlag und dem Inhalt besteht, gleich einem klassischem Brief.
- Auch wurde hier die Struktur einer E-Mail-Adresse, user@host.network festgelegt



[1] *Heading - Briefkopf*

- Ebenfalls 1982 wurde das SMTP - "Simple Mail Transfer Protocol" zum **Senden** von Mails zu einem Postfach durch **RFC 821** standardisiert.
- 1984 folgten mit **RFC 918** das POP – "Post Office Protocol" Festlegungen zum **Abholen** von Mails aus einem Postfach.

- Nachfolgend werden zuerst das Senden und Empfangen von Mails betrachtet und dann erst die Struktur vorgestellt.

E-Mail: Electronic-Mail - Steckbrief

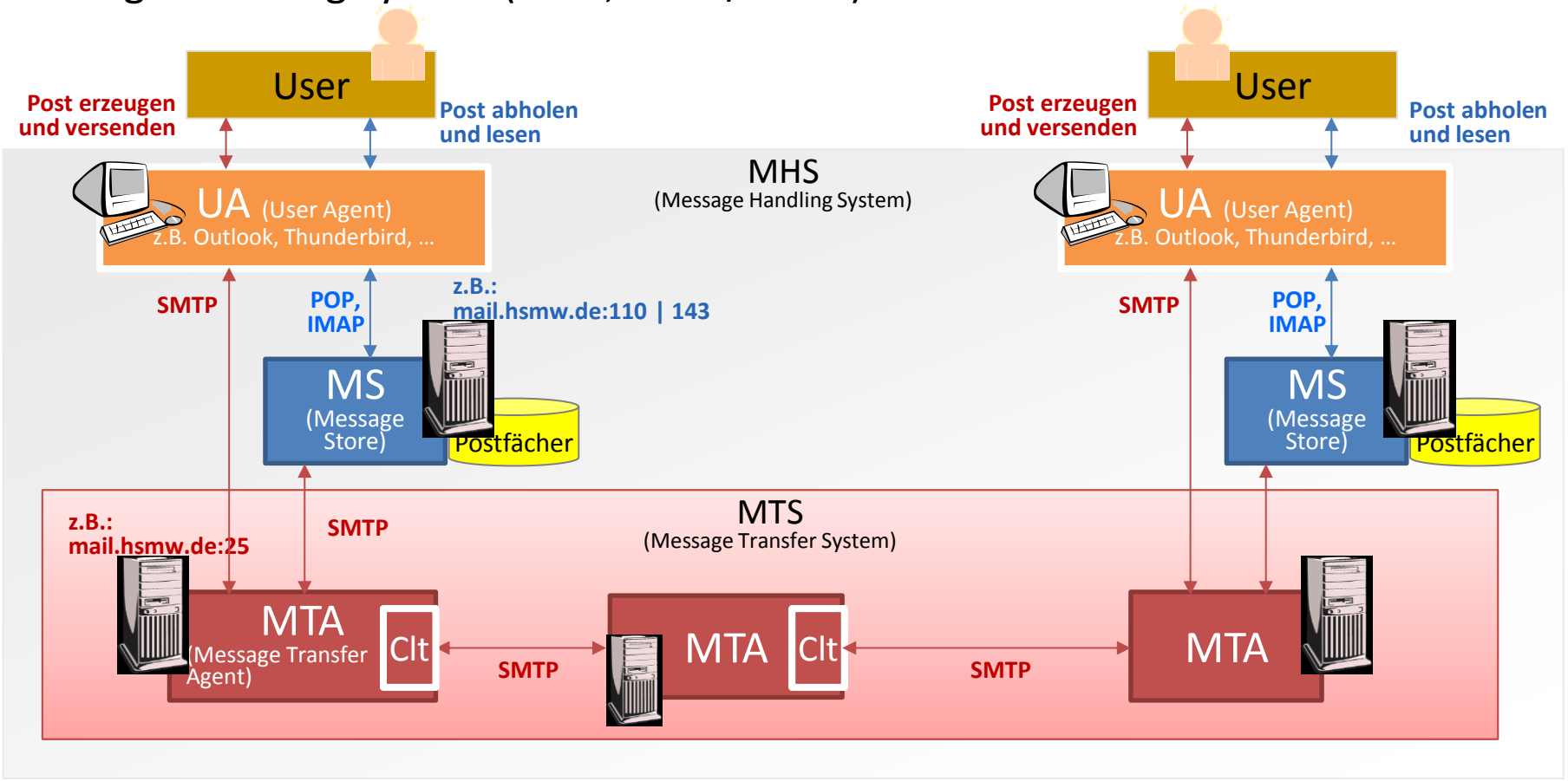
HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

- **Mail-Sendeprotokolle**:
  - **Simple Mail Transfer Protocol (SMTP)** RFC 5321(2008)
    - RFC 5321 beschreibt, wie der Umschlag (envelope) und der Inhalt (content) an einen Mail-Server übergeben werden.
    - Zwei Protokollvarianten sind nutzbar: SMTP und ESMTP (extended SMTP).
- **Mail-Abholprotokolle**
  - **Post Office Protocol – Version 3 (POP3)**, RFC 1939 (1996)
    - Ist ein einfaches Protokoll zum Abholen und Löschen von Mails
  - **Internet Message Access Protocol (IMAP),** RFC 2060(1996)
    - Ist ein leistungsfähiges Protokoll zum Abholen vom Server und Verwalten von Mail-Boxen und Mails auf dem Server
- **Postfachadressierung**:

```
<Nutzeridentifikation>@<E-Mail-Server>
z.B.:   Max-Mustermann@muster.mus
        Max.Mustermann@muster.mus
```

E-Mail: **MHS - Functional model**

- E-Mail: Nachbildung der Briefpost mit elektronischen Mitteln.
- Impulse für Standardisierung und Anwendungsinstanzen war der OSI-Dienst Message Handling System (MHS, F.400/X.400).



E-Mail: (E)SMTP-Kommando-Units

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

(E)SMTP-Phase	(E)SMTP-Kdo's	ESMTP	Bedeutung
(1) Anmeldung des UA beim Server	**HELO** hostname CRLF		Client meldet sich beim Server an für SMTP
	EHLO hostname CRLF	*	Client meldet sich beim Server an für ESMTP
	AUTH LOGIN CRLF base64_encoded(UserName) CRLF base64_encoded(PassWord) CRLF	*	Authentifizierte Anmeldung Übergabe Nutzername und Passwort Base_64-codiert
(2) Umschläge (envelop) adressieren	**MAIL FROM**:sender-address CRLF		Übergabe der Absenderadresse für den Umschlag (envelope)
	RCPT TO:recipient-address CRLF		Übergabe der Empfängeradresse(n) (recipient(s)) für den Umschlag (envelope)
	VRFY address CRLF		Verifiziere, ob an Adresse ausgeliefert werden kann (verify)
(3) Content-Übergabe	**DATA** CRLF		Leitet die Übergabe der Daten (Header und Content ein)
	. CRLF		Abschluss der Datenübergabe: Punkt CRLF
	HELP leer | Kdo CRLF	*	Hilfeinfos vom Server, leer: alle Kdos, Kdo: Hilfe zu Kdo
	RSET CRLF		Abbruch der momentanen Transaktion, Reset der Verbindung
	NOOP CRLF		Erwarte OK, zur Prüfung, ob Server noch da ist
(4) Abmeldung	**QUIT** CRLF		Sessionende, Abmelden beim Server

- ESMTP →Enhanced SMTP→ auch bezeichnet als ASMTP Authenticated SMTP
- Die Kommandos können auch klein geschrieben werden.

```
CR, Carriage Return:      #13, 0x0d, \r
LF, Line Feed;            #10, 0x0a, \n
```

E-Mail: **SMTP - Reply-Units**

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

211	System status, or system help reply
214	Help message [Information on how to use the receiver or the meaning of a particular non-standard command; this reply is useful only to the human user]
220	<domain> Service ready
221	<domain> Service closing transmission channel
250	Requested mail action okay, completed
251	User not local; will forward to <forward-path>
354	Start mail input; end with <CRLF>.<CRLF>
421	<domain> Service not available, closing transmission channel [This may be a reply to any command if the service knows it must shut down]
450	Requested mail action not taken: mailbox unavailable [E.g., mailbox busy]
451	Requested action aborted: local error in processing
452	Requested action not taken: insufficient system storage
500	Syntax error, command unrecognized [This may include errors such as command line too long]
501	Syntax error in parameters or arguments
502	Command not implemented
503	Bad sequence of commands
504	Command parameter not implemented
550	Requested action not taken: mailbox unavailable [E.g., mailbox not found, no access]
551	User not local; please try <forward-path>
552	Requested mail action aborted: exceeded storage allocation
553	Requested action not taken: mailbox name not allowed [E.g., mailbox syntax incorrect]
554	Transaction failed

E-Mail: **Clt-Srv-Phasen bezogen auf die Message structure**

(1) Clt stellt eine TCP-Verbindung (Port 25) zu Mail-Srv her.

(2) Clt meldet sich mittels SMTP –Units beim Srv an, eventuell mit Passwort.

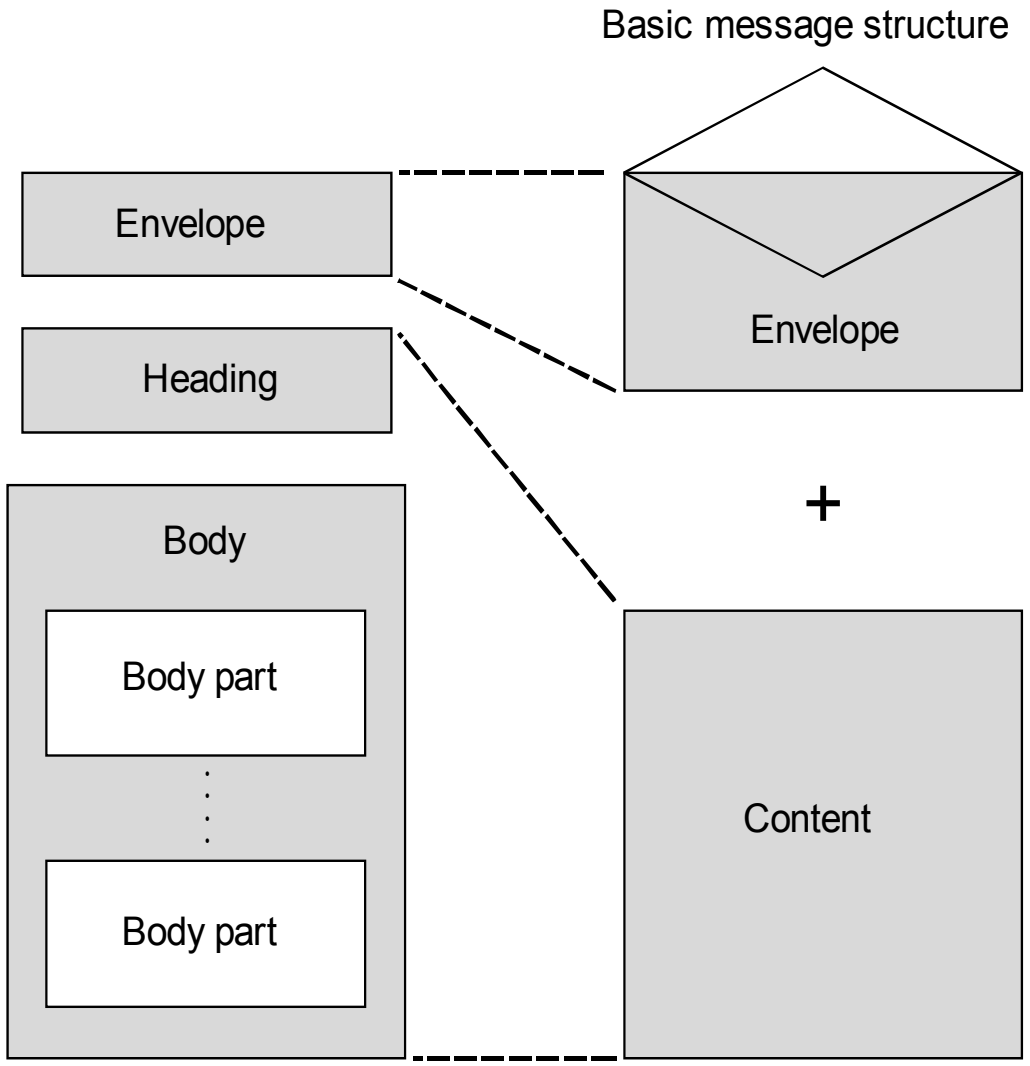
(3) Clt teilt mittels SMTP-Units dem Srv die Mail-Adresse des Absenders und der Empfänger mit.
→Umschläge für Mail

(4) Clt übergibt mittels SMTP-DATA-Unit dem Srv
-die Heading-Informationen und
-den Inhalt.

Einfacher ASCII-Inhalt wird durch RFC 5322 geregelt

Komplexe Inhalte sind nach RFC 2045 bis 2049 zu gestalten

(5) Clt meldet sich mittels SMTP–Units beim Srv ab, TCP-Verbindung wird beendet.



T0102850-93/d09

E-Mail: **SMTP-Session per Telnet**

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

(1) TELNET-Session

```
c:\>telnet CRLF
Microsoft Telnet>open mail.hsmw.de 25 CRLF
220 mail.htwm.de ESMTP Exim 4.24 Fri, 10 Oct 2008 12:45:30 +0200
```

(2) SMTP-Session

```
HELO localhostCRLF                                  //Anmeldung am Server für smtp
250 mail.htwm.de Hello range-146-2.vpn.htwm.de [141.55.146.2]

MAIL FROM:win@htwm.deCRLF                           //Absenderadresse für envelope
250 ok

RCPT TO:lutz.winkler@hs-mittweida.deCRLF            //Empfängeradresse für envelope
250 Accepted

rcpt to:lutz.winkler@web.deCRLF                     //Empfängeradresse für envelope
250 Accepted

DATACRLF                                            //Einleitung der Contentübergabe
354 Enter message, ending with "." on a line by itself

Subject: Mail mit unformatiertem TextCRLF           //Betreff-Header
Cc: lutz.winkler@web.deCRLF                         //CarbonCopy-Header

Lieber Empfänger, dies ist eine Mail mit unformatiertem Text.CRLF //Body
Der AbsenderCRLF
.CRLF                                                //Ende des Content Body

250 OK id=1KoFqr-0003OC-7z

quitCRLF                                            //Abmelden beim Server
221 mail.htwm.de closing connection
```

E-Mail: **POP – Post Office Protocol** RFC 1939

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

- POP3 dient zum Abholen von Mails aus einer Message-Box.
  - POP-Kommandos sind case-insensitive, nach einem SP können Argumente folgen (max. 40 Zeichen lang). Jede Kommandozeile wird durch CRLF abgeschlossen
  - POP-Antworten bestehen aus einem Statusindikator, einem Keyword, gefolgt von zusätzlichen Informationen (max. 512 Zeichen lang), abgeschlossen durch CRLF.
    - +OK text (z.B. +OK POP3 server ready)
    - -ERR text (z.B. -ERR no such message)

POP3-Commands	Bedeutung
APOP username digest	Verschlüsseltes Login. Server die APOP unterstützen, liefern im Begrüßungstext ein <Timestamp >. AusTimestamp SPuserpwd wird ein 128-Bit-Hashwert erzeugt (MD5), der als 32-stellige Hexzahl (digest) übermittelt wird.
DELE	Markiert eine Message zum Löschen. Löschung erfolgt am Sitzungsende mit QUIT
LAST none	Gibt höchste bisher bearbeitete Messagenummer zurück
LIST (none|messagenumber)	leer alle Messages und deren Größe, messagenumber: Größe der Message
NOOP none	Dient der Session-Erhaltung. Server beendet nach längerer Inaktivität die Session.
PASS userpwd	Nutzerpasswort
RSET	Mit Reset werden alle DELETE-Markierungen zurückgesetzt
RETR messagenumber	Abrufen (retrieve) einer Mail (Head und Body)
STAT none	Status liefert die Gesamtzahl der Nachrichten und Größe der Mailbox
TOP messagenumber SP lines	Top liefert Head einer Message und Anzahl von Body-Lines
UIDL messagenumber	Unique (einzigartig) ID Listing
USER usrname	Nutzername
QUIT none	Abmelden beim Server, DELETE-markierte Messages werden gelöscht.

E-Mail: **POP–Session (1)** per ApplProtocolExplorer

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

```
Connected
+OK Dovecot ready.

user winlehre
+OK

pass ****************
+OK Logged in.

stat
+OK 5 7924

list
+OK 5 messages:
1 846
2 1359
3 862
4 2907
5 1950
.
```

```
top 1 1
+OK
Return-path: <win@hs-mittweida.de>
Envelope-to: winlehre@hs-mittweida.de
Delivery-date: Fri, 04 Nov 2011 08:45:36 +0100
Received: from range-146-1.vpn.htwm.de ([141.55.146.1]
      helo=PC)
      by mail.hs-mittweida.de with esmtpsa (TLS-
      1.0:RSA_ARCFOUR_MD5:16)
      (Exim 4.69)
      (envelope-from <win@hs-mittweida.de>)
      id 1RMET1-0003cZ-UL
      for winlehre@hs-mittweida.de; Fri, 04 Nov 2011
      08:45:36 +0100
From: Lutz Winkler <win@hs-mittweida.de>
To: <winlehre@hs-mittweida.de>
Subject: Mail mit reinem Text
Date: Fri, 4 Nov 2011 08:45:30 +0100
Message-ID: <000001cc9ac5$bb638c80$322aa580$@de>
MIME-Version: 1.0
Content-Type: text/plain;
      charset="us-ascii"
Content-Transfer-Encoding: 7bit
X-Mailer: Microsoft Office Outlook 12.0
Content-Language: de

Dies ist eine Mail mit reinem Text.
.
```

E-Mail: **POP–Session (2)** per ApplProtocolExplorer

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

retr 2
+OK 1359 octets
Return-path: <win@hs-mittweida.de>
Envelope-to: winlehre@hs-mittweida.de
Delivery-date: Fri, 04 Nov 2011 08:47:19 +0100
Received: from range-146-1.vpn.htwm.de ([141.55.146.1]
helo=PC)
by mail.hs-mittweida.de with esmtpsa (TLS-1.0:RSA_ARCFOUR_MD5:16)
(Exim 4.69)
(envelope-from <win@hs-mittweida.de>)
id 1RMEUh-0003n4-9A
for winlehre@hs-mittweida.de; Fri, 04 Nov 2011 08:47:19 +0100
From: Lutz Winkler <win@hs-mittweida.de>
To: <winlehre@hs-mittweida.de>
Subject: Mail mit reinem Text und einem GIF
Date: Fri, 4 Nov 2011 08:47:15 +0100
Message-ID: <000301cc9ac5$f8f2fa90$ead8efb0$@de>
MIME-Version: 1.0
Content-Type: multipart/mixed;
boundary="----=_NextPart_000_0004_01CC9ACE.5AB76290"
X-Mailer: Microsoft Office Outlook 12.0
Content-Language: de

This is a multi-part message in MIME format.

------=_NextPart_000_0004_01CC9ACE.5AB76290
Content-Type: text/plain;
charset="us-ascii"
Content-Transfer-Encoding: 7bit

Dies ist eine Mail mit reinem Text und einem Gif "5x5.gif".
Ulf

------=_NextPart_000_0004_01CC9ACE.5AB76290
Content-Type: image/gif; name="5x5.gif"
Content-Transfer-Encoding: base64
Content-Disposition: attachment;
filename="5x5.gif"

R0lGODlhBQAFAIAAAKVfpAAAACH5BAAAAAAALAAAAAAF
AAUAAAIEhI+pWAA7

------=_NextPart_000_0004_01CC9ACE.5AB76290—
.

quit
+OK Logging out.

DisConnected

- **IMAP** verwendet **TCP, Port 143**

- IMAP ist mächtiger als POP:
  - Messages können auf dem Server untersucht und analysiert werden,
  - Während bei POP die gesamte Box zum Client übermittelt wird, kann man bei IMAP:
    - Messages von einem bestimmten Absender,
    - Messages bis zu einer bestimmten Größe,
    - Messages eines bestimmten Zeitraumes abholen.
  - Des weiteren kann man Mail-Boxen verwalten (anlegen, löschen, umbenennen, auswählen), Mails in Mail-Boxen ablegen, Mails verschieben, Mails suchen usw.
- IMAP unterstützt Verschlüsselungsverfahren. Diese werden in der Authentifizierungsphase ausgehandelt.
- Jedes Client-Kdo. beginnt mit einem Identifier (Tag genannt), z.B. 1|01|a1. Der Server verwendet in der Antwort dieses Tag. Dadurch können vom Client mehrere Befehle an den Server gesendet werden.
- Mehrzeilige Antworten werden durch das Jokerzeichen "*" eingeleitet.

E-Mail: **IMAP-Kommando-PDU's** (Auswahl)

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

PDU arguments	Bedeutung
login user-name SP usr-pwd	Authentikation
list "" "*"	Auflistung aller Mailordner
select mailbox-name	Mailbox-Auswahl (Standard inbox). Antwort: Status der Box
fetch message-set SP message-data-item-names	Anforderung von bestimmten Informationen einer|mehrerer Message(s) Bsp.: fetch 2:5 rfc822.header, liefert für Messages 2-5 die Header Bsp.: fetch 110 rfc822.text, liefert für Message 110 den Body.Text
noop none	Dient der Session-Erhaltung.
status mailbox-name SP status-data-item-names	Abfrage des Status einer Mailbox Bsp.: 01 status inbox (messages recent)
create mailbox-name	Erzeugen einer neuen Message-Box
delete mailbox-name	Löschen einer Message-Box
rename old-mailbox-name SP new-mailbox-name	Umbenennen einer Message-Box
expunge none	Löschen von Messages mit gesetztem Delete-Flag
store message-set SP message-data-item-name SP value-for-message-data-item	Änderungen (store) von Flags einer|mehrerer Messages
close none	Schließen der aktuellen Mailbox
logout none	Abmelden am Server

- Server-Status-Responses sind: OK *response*, BAD *response*, NO *response*, BYE *response*
- Fast alle Kdo.-PDU's haben eine entsprechende Response-PDU zur Folge.

E-Mail: **IMAP–Session (1)** per ApplProtocolExplorer

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

```
Connected
* OK Dovecot ready.

1 login  ******************
1 OK Logged in.

2 status inbox (messages recent)\r\n
* STATUS "inbox" (MESSAGES 5 RECENT 5)
      //5 Nachrichten insgesamt, 5 sind neu seit letzter Session
2 OK Status completed.

3 select inbox
* FLAGS (\Answered \Flagged \Deleted \Seen \Draft)
* OK [PERMANENTFLAGS (\Answered \Flagged \Deleted \Seen
      \Draft \*)] Flags permitted.
      //Anzeige der nutzbaren Flags in diesem Ordner
* 5 EXISTS       //Anzahl der Mails
* 5 RECENT      //Anzahl neuer Mails seit letzter Session
* OK [UNSEEN 1] First unseen.
* OK [UIDVALIDITY 1320231203] UIDs valid
* OK [UIDNEXT 62] Predicted next UID
3 OK [READ-WRITE] Select completed.

4 fetch 1 rfc822.header\r\n
* 1 FETCH (RFC822.HEADER {802}
Return-path: <win@hs-mittweida.de>
Envelope-to: winlehre@hs-mittweida.de
Delivery-date: Fri, 04 Nov 2011 09:23:24 +0100
Received: from range-146-1.vpn.htwm.de ([141.55.146.1] helo=PC)
      by mail.hs-mittweida.de with esmtpsa (TLS-
      1.0:RSA_ARCFOUR_MD5:16)
      (Exim 4.69)
      (envelope-from <win@hs-mittweida.de>)
      id 1RMF3c-0008Bt-M0
      for winlehre@hs-mittweida.de; Fri, 04 Nov 2011 09:23:24
      +0100

From: Lutz Winkler <win@hs-mittweida.de>
To: <winlehre@hs-mittweida.de>
Subject: Mail mit reinem Text
Date: Fri, 4 Nov 2011 09:23:21 +0100
Message-ID: <003401cc9acb$03a1b670$0ae52350$@de>
MIME-Version: 1.0
Content-Type: text/plain;
      charset="us-ascii"
Content-Transfer-Encoding: 7bit
X-Mailer: Microsoft Office Outlook 12.0
Thread-Index: AcyaywNOyjFt9ecUR4W5QQNPUy+EsA==
Content-Language: de

)
4 OK Fetch completed.
```

//Durch den Autor hinzugefügte Kommentare

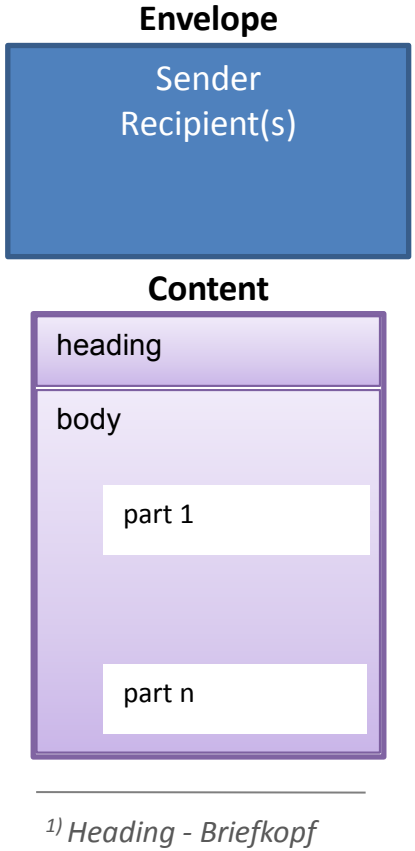
E-Mail: **IMAP–Session (2)** per ApplProtocolExplorer

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

```
5 fetch 2 rfc822.text\r\n
* 2 FETCH (RFC822.TEXT {537}
This is a multi-part message in MIME format.

------=_NextPart_000_0038_01CC9AD3.9D107460
        Content-Type: text/plain;
        charset="us-ascii"
        Content-Transfer-Encoding: 7bit

Dies ist eine Mail mit reinem Text und einem GIF "5x5.gif".
Ulf

------=_NextPart_000_0038_01CC9AD3.9D107460
        Content-Type: image/gif;  name="5x5.gif"
        Content-Transfer-Encoding: base64
        Content-Disposition: attachment; filename="5x5.gif"

R0lGODlhBQAFAIAAAKVfpAAAACH5BAAAAAAALAAAAAAFAAUAAAI
        EhI+pWAA7

------=_NextPart_000_0038_01CC9AD3.9D107460--

)
5 OK Fetch completed.


6 fetch 3 rfc822\r\n
* 3 FETCH (FLAGS (\Seen \Recent) RFC822 {891}
Return-path: <win@hs-mittweida.de>
Envelope-to: winlehre@hs-mittweida.de
Delivery-date: Fri, 04 Nov 2011 09:27:53 +0100
Received: from range-146-1.vpn.htwm.de ([141.55.146.1] helo=PC)
        by mail.hs-mittweida.de with esmtpsa (TLS-
        1.0:RSA_ARCFOUR_MD5:16)
        (Exim 4.69)
        (envelope-from <win@hs-mittweida.de>)
        id 1RMF7x-00009f-Dt
        for winlehre@hs-mittweida.de; Fri, 04 Nov 2011 09:27:53
        +0100
From: Lutz Winkler <win@hs-mittweida.de>
To: <winlehre@hs-mittweida.de>
Subject: Mail mit Sonderzeichen
Date: Fri, 4 Nov 2011 09:27:50 +0100
Message-ID: <004201cc9acb$a3d1b550$eb751ff0$@de>
        MIME-Version: 1.0
        Content-Type: text/plain;
        charset="iso-8859-1"
        Content-Transfer-Encoding: quoted-printable
X-Mailer: Microsoft Office Outlook 12.0
Thread-Index: Acyay6OFrRAzTxMLRVWsvn4I0ecaAA==
Content-Language: de

Umlaute: =C4, =D6, =DC, =E4, =F6, =FC
Sz: =DF
Istgleich: =3D
=DClf

)
6 OK Fetch completed.

7 logout
* BYE Logging out
7 OK Logout completed.

DisConnected
```

E-Mail: **Electronic-Mail - Struktur**

- **Struktur** - Eine Mail besteht aus:
  - dem Umschlag (envelope),
  - dem Inhalt (content).
- **Content**-Beschreibungsstandard ist RFC 5322 2008 "Internet Message Format".
  - Dieser legt den Aufbau (heading, body) einer Mail fest und wie eine einfache ASCII-codierte Mail aussieht.
- Wenn Nicht-ASCII-Inhalte versendet werden sollen (Bilder, Videos, ISO-8859-Texte, *.exe, ...) müssen diese Inhalte in Pseudo-ASCII-Texte kodiert und beschrieben werden.
- Dafür nutzt man MIME – "Multipurpose Internet Mail Extensions ", RFC 2045 bis RFC 2049 1996.
- Anhand von zwei Beispiel-Mails soll die Struktur des Contents sichtbar gemacht werden. Anschließend werden die Headers etwas näher betrachtet und danach MIME.

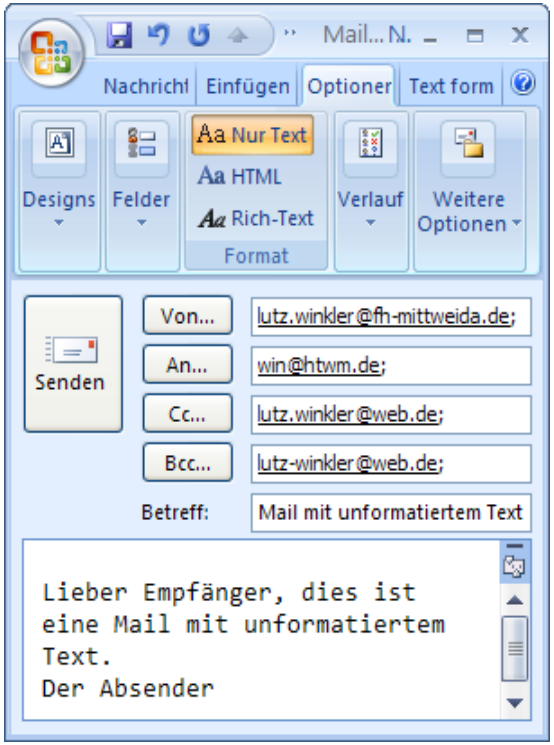


[1] *Heading - Briefkopf*

E-Mail: **Beispiel-Mail 1 - Absenderseite**

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

Diese Mail wurde versendet



So sehen **Briefkopf (Heading) und Inhalt (Body) der gesendeten Mail** aus

```
From:           <lutz.winkler@fh-mittweida.de>\r\n
Sender:         "Prof. L. Winkler" <win@htwm.de>\r\n
To:             <win@htwm.de>\r\n
Cc:             <lutz.winkler@web.de>\r\n
Subject:        Mail mit unformatiertem Text\r\n
Date:           Thu,9 Oct 20 08 11:50:53 +0200\r\n
Message-ID:     <000e01c929f4$851e8a20$8f5b9e60$@winkler@fh-
                 mittweida.de>\r\n
\r\n
MIME-Version:   1.0\r\n
Content-Type:   text/plain;\r\n charset ="iso-8859-1"\r\n
Content-Transfer-Encoding: quoted-printable\r\n
X-Mailer:       Microsoft Office Outlook 12.0\r\n
Thread-Index:   Ackp9IS890MWK30ASPy5MR36tUe1hQ==\r\n
Content-Language: de\r\n
\r\n
Lieber Empf=E4nger, dies ist eine Mail mit unformatiertem Text.\r\n
Der Absender\r\n
```

Ersteller und Sender der Mail

Empfänger der Mail

Betreff und Identifikation d. Mail

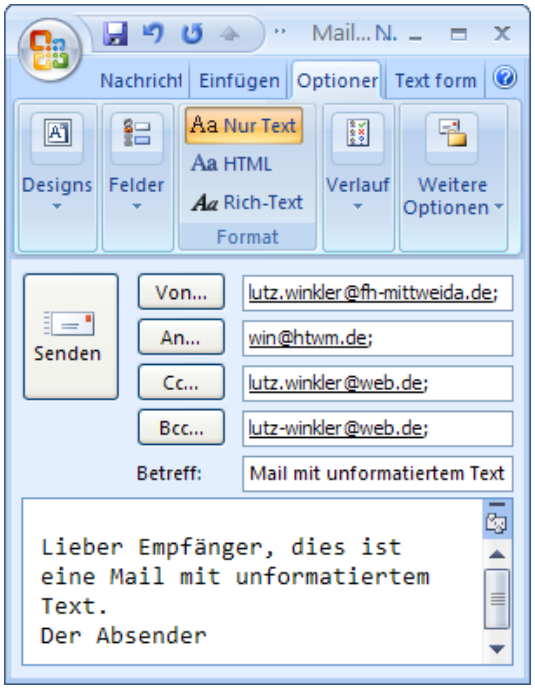
MIME-Headers

E-Mail: Beispiel-Mail 1 - Empfängerseite

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

Diese Mail wurde versendet

So sehen **Briefkopf (Heading) und Inhalt (Body) beim Empfänger** aus



RFC 5322 Headers

```
Return-path:    <win@noauth.htwm.de>\r\n //Absenderadresse "mail from:"
Envelope-to:    win@htwm.de\r\n
Delivery-date:  Thu, 09 Oct 2008 11:50:51 +0200\r\n
Received:       from range-146-6.vpn.htwm.de //Absenderhost
                     ([141.55.146.6]:1197 helo=winhome)
                by mail.htwm.de with esmtp (Exim 4.24)
                id 1KnsAU-0007wW-VN; Thu, 09 Oct 2008 11:50:51 +0200
From:                <lutz.winkler@fh-mittweida.de>\r\n
Sender:              "Prof. L. Winkler" <win@htwm.de>\r\n
To:                        <win@htwm.de>\r\n
Cc:                        <lutz.winkler@web.de>\r\n
Subject:             Mail mit unformatiertem Text\r\n
Date:                Thu,9 Oct 2008 11:50:53+0200\r\n
Message-ID:          <000e01c929f4$851e8a20$8f5b9e60$@winkler@fh-
                     mittweida.de>\r\n
\r\n
MIME-Version: 1.0\r\n
Content-Type: text/plain;\r\n charset ="iso-8859-1"\r\n
Content-Transfer-Encoding: quoted-printable\r\n
X-Mailer:            Microsoft Office Outlook 12.0\r\n
Thread-Index: Ackp9IS890MWK30ASPy5MR36tUe1hQ==\r\n
Content-Language: de\r\n
\r\n
Lieber Empf=E4nger, dies ist eine Mail mit unformatiertem Text.\r\n
Der Absender\r\n
```

Transportsystem-Headers

Ersteller und Sender der Mail

Empfänger der Mail

Betreff und Identifikation d. Mail

MIME-Headers

RFC 5322 Body

E-Mail: **Beispiel-Mail 2 - Absenderseite**

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

Diese Mail wurde versendet

So sehen **Briefkopf (Heading) und Inhalt (Body) beim Absender** aus

RFC 5322 Headers

```
From: <lutz.winkler@fh-mittweida.de>\r\n
Sender: "Prof. L. Winkler"<win@htwm.de>\r\n
To: <win@htwm.de>\r\n
Cc: <lutz.winkler@web.de>\r\n
Subject: Mail mit Text und Bild\r\n
Date: Thu,9Oct 2008 14:00:09+0200\r\n
Message-ID:<002e01c92a06$943dfab0$bcb9f010$@winkler@fh-mittweida.de>\r\n
MIME-Version : 1.0\r\n
Content-Type: multipart/mixed;\r\n.
boundary ="----=_NextPart_000_002F_01C92A17.57C6CAB0"\r\n
X-Mailer: Microsoft Office Outlook 12.0\r\n
Thread-Index: AckqBo+VK9q9IOHfSRmOzWadETU wxA==\r\n
Content-Language: de\r\n
Dies ist eine mehrteilige Nachricht im MIME-Format\r\n
\r\n
------=_NextPart_000_002F_01C92A17.57C6CAB0\r\n
Content-Type: text/plain;\r\n. charset="iso-8859-1"\r\n
Content-Transfer-Encoding: quoted-printable\r\n
\r\n
Lieber Empf=E4nger,dies ist eine Mail mit unformatiertem Text
Und einem\r\n GIF-Bild.\r\n Der Absender\r\n
\r\n
\r\n
------=_NextPart_000 _002F_01C92A17.57C6CAB0\r\n
Content -Type: image/gif ;\r\n Name="5x5.gif"\r\n
Content-Transfer-Encoding: base64\r\n
Content-Disposition: attachment;\r\n filename="5x5.gif"\r\n
\r\n
R0lGODlhBQAFAIAAAP8AAAAACH5BAAAAAAALAAAAAAFAAUAAAIEhI+pWAA7\r\n
\r\n
------=_NextPart_000_002F_01C92A 17.57C6CAB0--\r\n
```

Ersteller und Sender der Mail

Empfänger der Mail

Betreff und Identifikation

MIME-Headers

RFC 5322 Multipart Body

E-Mail: **Headers nach** RFC5322

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

Field types	Fields, Notation in ABNF	Bedeutung
Origination Date Field	"Date:" date-time CRLF	Absender-Zeitpunkt mit Datum und Zeit
Originator Fields	"From:" mailbox-list CRLF	Autor der Mail
	"Sender:" mailbox CRLF	Sender der Mail
	"Reply-To:" address-list CRLF	Adresse (n), an die eine Antwort gesendet werden soll
Destination Address Fields	"To:" address-list CRLF	Adresse des primären Empfängers
	"Cc:" address-list CRLF	Adressen, die einen "Durchschlag" bekommen
	"Bcc:" [address-list/CFWS] CRLF	Adressen, die Durchschlag bekommen mit unterschiedlicher Sichtbarkeit CFWS - comments and folding whitespaces
Identification Fields	"Message-Id:" msg-id CRLF	einmaliger maschinenlesbarer Identifier
	"In-Reply-To:" 1*msg-id CRLF	Enthält den Message-Id der Mail, auf die sich die Antwort bezieht.
	"References:" 1*msg-id CRLF	enthält die Id-History aller Antworten, bei Reply auf Reply usw.
Informational Fields	"Subject:" unstructured CRLF	Betreff der Message . Bei Replay kann "Re:" davor stehen
	"Comments:" unstructured CRLF	Enthält Kommentare zum Body
	"Keywords:" phrase *("," phrase) CRLF	Kommagetrennte Schlagworte, den Body betreffend
Resent Fields	"Resent-From:" mailbox-list CRLF	Wird eine Mail nochmals gesendet, werden die ursprünglichen Header beibehalten. Die ReSent-Header werden informativ ergänzend hinzugefügt.
	"Resent-Sender:" mailbox CRLF	
	"Resent-To:" adress-list CRLF	
	"Resent-Cc:" adress-list CRLF	
	"Resent-Bcc:" adress-list CRLF	
	"Resent-Message-Id:" msg-id CRLF	
Trace Fields	"Return-Path:" path CRLF	Enthält die Adresse, die im SMTP-Cmd "mail from:" angegeben wurde
	"Received:" *received-token ";"date-time CRLF	Wird durch jeden Message-Transfer-Agent eingefügt: Absenderhost-Name und -IP-Adresse, Mail-Servername, Datum-Zeit-Stempel

E-Mail: **Trace-Headers** einer Mail von web.de zu hsmw.de

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

	Field type
Return-path: <prof.win@web.de>	5322-Trace
Envelope-to: win@hs-mittweida.de	SMTP RCPT TO:
Delivery-date: Thu, 03 Nov 2011 10:18:55 +0100	SMTP
Received: from [172.16.192.83] (helo=mailto1.hs-mittweida.de) by mail.hs-mittweida.de with esmtp (Exim 4.69) (envelope-from <prof.win@web.de>) id 1RLtRn-0006jR-20 for win@hs-mittweida.de; Thu, 03 Nov 2011 10:18:55 +0100	5322-Trace
Received: from fmmailgate05.web.de ([217.72.192.243]) by mailto1.hs-mittweida.de with esmtp (Exim 4.69) (envelope-from <prof.win@web.de>) id 1RLtRn-0000ZU-20 for win@hs-mittweida.de; Thu, 03 Nov 2011 10:18:55 +0100	5322-Trace
Received: from web.de by fmmailgate05.web.de (Postfix) with ESMTP id 99DCB672B361 for <win@hs-mittweida.de>; Thu, 3 Nov 2011 09:42:24 +0100 (CET)	5322-Trace
Received: from [77.64.211.250] by mwmweb088 with HTTP; Thu Nov 03 09:42:24 CET 2011 Date: Thu, 3 Nov 2011 09:42:24 +0100 (CET)	5322-Trace
From: "Lutz Winkler" <prof.win@web.de>	5322-Originator
To: win@hs-mittweida.de	5322-Destination Address
Message-ID: <491934726.1441.1320309744613.JavaMail.fmail@mwmweb088>	5322-Identification
Subject: Test	5322-Informational
MIME-Version: 1.0 Content-Type: text/html; charset="UTF-8" Content-Transfer-Encoding: 7bit	MIME-Headers

E-Mail: **Internet Message Format**

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

- Wie bereits erwähnt, wird in RFC 5322 das **generelle Internet Message Format** festgelegt.
- Der RFC 5322 legt die Syntax fest, wie Mails zwischen UserAgent und dem Mail-Transportsystem ausgetauscht werden:
  - Kap. 2 Lexikalische Analyse: Allgemeines, Header, Body
  - Kap. 3 Syntax: Allgemeines, Lexikalische Tokens, Gesamt-Message-Syntax, Header-Felder

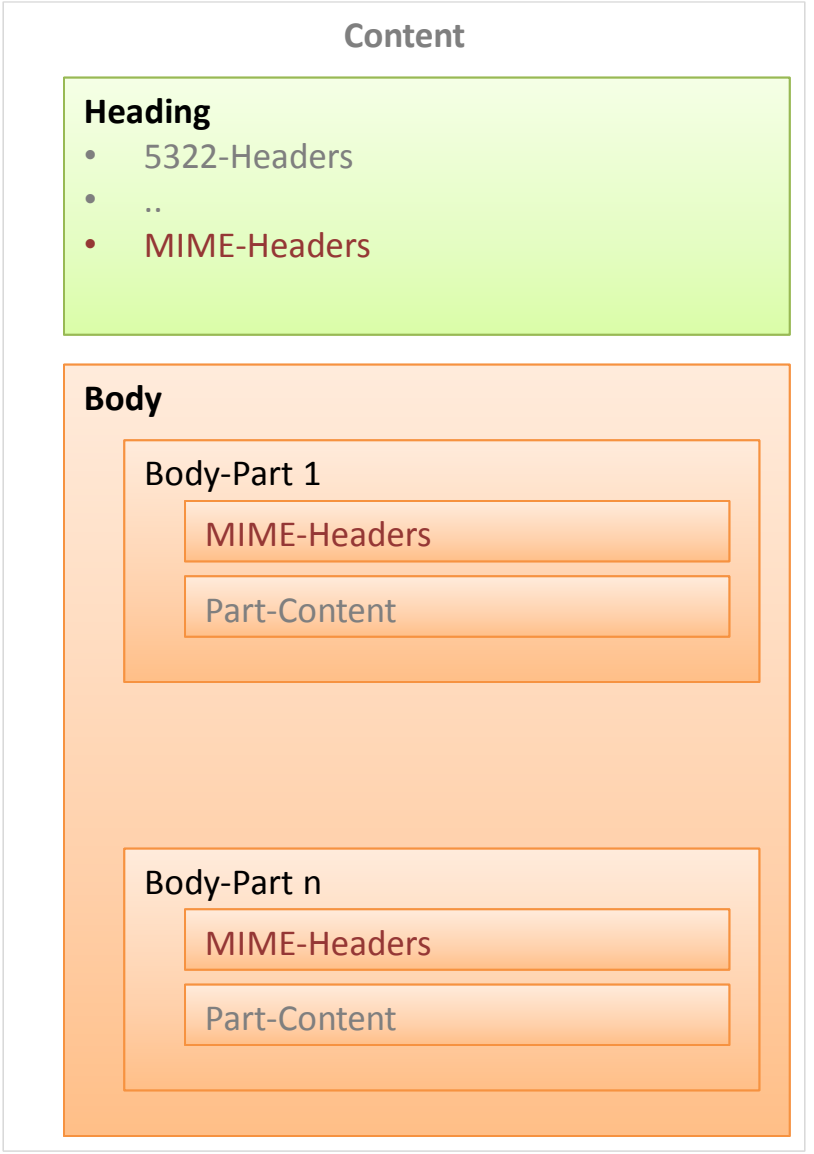
- **Allgemeine Festlegungen**
  - Der Message-Body darf nur aus Zeilen darstellbarer ACSII-Zeichen bestehen.
  - CR und LF treten immer zusammen als Zeilenbegrenzer auf.
  - Eine Zeile darf nicht mehr als 998 Zeichen haben ohne CRLF.
  - Empfohlen werden Zeilen mit max. 78 Zeichen ohne CRLF

- Will man Bilder, EXE's oder andere Dateiformate per E-Mail übertragen:
  - muss man diese in einen Pseudo-ASCII-Text wandeln →**Transfercodierung**
  - dem Empfänger mitteilen, welche Transfercodierung angewendet wurde
  - dem Empfänger mitteilen, wie die ursprüngliche Codierung war
  - dem Empfänger mitteilen, wie der Dateiname und die Dateiendung lauteten.
  - **Dafür nutzt man MIME (Multi-Purpose Internet Mail Extension).**

E-Mail **MIME–Headers** RFC 2045

- MIME-Headers einer Multipart- Message
  - können Teil des 5322-Headers sein (generelle Angaben),
  - und beschreiben in einer mehrteiligen Message die Parts näher.

- Generelle MIME-Headers sind z.B.:
  - **MIME-Version**: derzeit 1.0
  - **Content-Type**: z.B. multipart/parallel
- Partbezogene MIME-Headers sind
  - **Content -Type**: image/gif ;Name="5x5.gif"
  - **Content-Transfer-Encoding**: base64
  - **Content-Disposition**: attachment;filename="5x5.gif"
  - **Content-Description**: lesbare ASCII-Kurzbeschreibung des Inhaltes



E-Mail: **MIME-Headers nach** RFC 2045 u.a.

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

Header	
MIME-Version	bisher gibt es nur die Version 1.0, `MIME-Version:1.0`
Content-Type (siehe extra Folie)	MIME text, image, audio, video, application Defaultwert text/plain; charset=us-ascii
Content-Transfer-Encoding (siehe extra Folie)	Übertragungs-Codierungsverfahren: 7Bit | 8Bit | quoted printable | base64
Content-ID	MIME-ID der Mail, weltweit eindeutig
Content-Description	Optionale Inhalts-Beschreibung, z.B. `Content-Description: Ein Bild des Nikolaus`

Content-Language RFC 3282	`Content-type: application/dictionary`
	`Content-Language: en, de (This is a dictionary)`
	An official European Commission document (in a few of its official languages):
	`Content-type: multipart/alternative`
	`Content-Language: da, de, el, en, fr, it`

Basierend auf CCITT-Empfehlung X.400, ISO 10021	
X-Sender	(user defined field)
X-Mailer	(user defined field), z.B. `X-Mailer: Microsoft Office Outlook 12.0`
X-Gateway	(user defined field)
X-Priority	(user defined field)

E-Mail: **Einige MIME – Content-Types** RFC 2046

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

Typ	Subtypes	
text	plain	Klartext, ohne Formatangaben wie z.B. bei rtf oder html
	rtf	Rich-Text-Format, mit Angaben : Schriftart, -größe, -stil, -Farbe
	html	HTML-formatierter Text
	xml	XML-Text
	css	CSS-Text
image	jpeg	Foto
	gif,	Grafik
	tiff	Grafik
	g3fax	Fax
	cgm	
audio	basic	Sprache, Musik
	32kadpcm	
video	mpeg	
	quicktime	

Typ	Subtypes	
multipart	mixed	Mehrteilige Nachricht, jede mit eigenem Content.Type und Content Transfer Encoding
	alternative	Gleiche Nachricht, verschieden codiert
	parallel	Nachrichtenteile müssen gleichzeitig dargestellt werden (Synchr. Sprache, Bild)
	digest	Jeder Teil ist vollständige 5322-Message
message	rfc822	Vollständige Message
	external-body	Message auf FTP-Server
application	octet-stream	Angabe, wenn Dateiformat nicht bekannt ist oder für ausführbare Dateien wie EXE
	zip	ZIP-Archiv-Datei
	msword	
	postscript	

Gesamtübersicht:
http://www.iana.org/assignments/media-types/index.html

E-Mail: **MIME – Content-Transfer-Encoding** RFC 2045

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

Content-Transfer-Encoding:	
7bit Data	Erlaubt sind alle Codes von 1..127. Verboten ist NUL (0). CR (13) und LF (10) sind zusammen als Zeilenendekennung zulässig.
8bit Data	Erlaubt sind alle Codes von 1..255 . Es gelten die gleichen Einschränkungen, wie bei 7bit Data.
Binary Data	Erlaubt sind alle Codes von 0..255.
SMTP kann aber nur 7bit-ASCII-Daten transportieren, mit einer max. Zeilenlänge von 1000 Zeichen (inkl. CRLF). Damit man beliebige Daten mittels SMTP transportieren kann, müssen diese in Quasi-ASCII-Texte kodiert werden. Dafür geeignet sind quoted printable und base64.	
quoted printable	Diese Kodierung wurde eingeführt, um Texte zu kodieren, die hauptsächlich aus ASCII-Zeichen bestehen. Z.B. ISO-8859-codierte Texte. ASCII-Zeichen der Bereiche 33..60, 62-126 werden so belassen. Besondere Zeichen (HT, =) und darstellbare Zeichen >127 werden umschrieben durch: "="HH, wobei HH dem Hexwert des Zeichens in der Codetabelle entspricht. →Details, siehe extra Folie.
base 64	Diese Kodierung ist universell einsetzbar. Die zu sendenden Daten werden in 6-Bit-Schnipsel zerlegt, die dann anhand einer Codetabelle in einen Pseudo-ASCII-Text umgesetzt werden. Die so codierten Daten sind etwa 33% größer als das Original →Details, siehe extra Folie.

E-Mail: **Quoted-Printable** Content-Transfer-Encoding RFC 2045

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

- Dieses Format wurde für Inhalte eingeführt, die hauptsächlich aus druckbaren ASCII-Zeichen bestehen. Es gelten folgende **Codierungsregeln**:

(1) (**8bit-Werte**) Ein Oktett, außer CR oder LF als Teil eines CRLF, werden codiert durch "=" gefolgt von zwei Hexziffern, die den Dezimalwert des Oktetts repräsentieren. Hexziffern sind "123456789ABCDE". Kleinschreibung ist nicht zulässig.
→**Das Gleichheitszeichen (61) wird codiert in "=3D"**.

(2) (**Literale**) Oktetts mit den Dezimalwerten 33 .. 60 und 62 .. 126 können durch US-ASCII-Zeichen dargestellt werden.

(3) (**White Space**) Oktetts mit den Dezimalwerten 9 (TAB) und 32 (SP), repräsentieren sich selbst. Aber nicht, wenn sie am Zeilenende stehen. In diesem Fall werden sie als "=09" bzw. "=20" codiert. Manche MTAs löschen White Spaces am Zeilenende oder Leerzeilen.

(4) (**Line Breaks**) In einem Text wird der Zeilenumbruch CRLF durch die Zeichen selber repräsentiert. In Nicht-Textdaten werden CR durch "=0D" und LF durch "=0A" dargestellt.

(5) (**Soft Line Breaks**) Eine quoted-printable-codierte Zeile darf nicht länger als 76 Zeichen sein. Encoder fügen deshalb ein Soft Line Break in Form eines "=" ein. Der Decoder entfernt diesen Umbruch wieder. Dieser Soft-Line-Break wird durch Encoder typisch nach 70 … 76 Zeichen eingefügt.

E-Mail: **Quoted-Printable** Beispiele

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

- Einen QT-Encoder/Decoder für Text findet man unter:
  https://www.telecom.hs-mittweida.de/lehre-di-fernstudium/teachware.html#c26525

Beispiel	1 2 3 4 5 6 7 8 12345678901234567890123456789012345678901234567890123456789012345678901234567890
1	*Hätten Hüte keine Krempen, wären Hüte keine Hüte sondern nur öde Kappen.*
	H=E4tten H=FCte keine Krempen, w=E4ren H=FCte keine H=FCte sondern nur= =F6de Kappen.
2	*Weiteres Beispiel:* *Grüne Hüte tragen schöne Försterinnen und Förster beim späten Jäten junger Bäume.*
	Weiteres Beispiel:=20 Gr=FCne H=FCte tragen sch=F6ne F=F6rsterinnen und F=F6rster beim sp=E4= ten J=E4ten junger B=E4ume. Das ASCII-Zeichen "=3D" wird als "=3D3D" codiert.
3	*Das ASCII-Zeichen = ist das Erweiterungszeichen und muss deshalb auch besonders codiert werden. Es wird als =3D codiert.*
	Das ASCII-Zeichen =3D ist das Erweiterungszeichen und muss deshalb auch besonders codiert werden. Es wird als =3D3D codiert.

E-Mail: **Base64** Content-Transfer-Encoding RFC 2045

- Die **base64**-Kodierung beruht darauf, dass beliebige Dateiinhalte in einen Pseudo-ASCII-Text gewandelt werden, der dann problemlos übertragen werden kann.
- Kern bildet eine Kodetabelle, bestehend aus 64 ASCII-Zeichen.
- Kodierung:
  - (1) Die zu kodierenden Bytes werden als Bitstrom aufgefasst.
  - (2) Dieser Bitstrom wird in 6-Bit-Schnipsel geteilt und mittels der Tabelle in Pseudotext kodiert.
  - (3) Damit jeder Bitstrom durch 6 teilbar wird, muss man diesen u.U. mit 2 oder 4 Binärnullen "auffüllen".
  - (4) Dieses Padding wird durch das "="-Zeichen angezeigt.
    - zwei Nullen aufgefüllt: =
    - vier Nullen aufgefüllt: ==
  - Nach spätestens 76 Base-64-Zeichen wird bei E-Mail-Anwendung ein CRLF eingefügt.

Wert		Kode	Wert		Code	Wert		Code	Wert		Code
00 0000	0	A	01 0001	17	R	10 0010	34	i	11 0011	51	z
00 0001	1	B	01 0010	18	S	10 0011	35	j	11 0100	52	0
00 0010	2	C	01 0011	19	T	10 0100	36	k	11 0101	53	1
00 0011	3	D	01 0100	20	U	10 0101	37	l	11 0110	54	2
00 0100	4	E	01 0101	21	V	10 0110	38	m	11 0111	55	3
00 0101	5	F	01 0110	22	W	10 0111	39	n	11 1000	56	4
00 0110	6	G	01 0111	23	X	10 1000	40	o	11 1001	57	5
00 0111	7	H	01 1000	24	Y	10 1001	41	p	11 1010	58	6
00 1000	8	I	01 1001	25	Z	10 1010	42	q	11 1011	59	7
00 1001	9	J	01 1010	26	a	10 1011	43	r	11 1100	60	8
00 1010	10	K	01 1011	27	b	10 1100	44	s	11 1101	61	9
00 1011	11	L	01 1100	28	c	10 1101	45	t	11 1110	62	+
00 1100	12	M	01 1101	29	d	10 1110	46	u	11 1111	63	/
00 1101	13	N	01 1110	30	e	10 1111	47	v			
00 1110	14	O	01 1111	31	f	11 0000	48	w			
00 1111	15	P	10 0000	32	g	11 0001	49	x			
01 0000	16	Q	10 0001	33	h	11 0010	50	y			

E-Mail: **Base64** Kodierungsbeispiele

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

- Kodieren Sie den UNI-Code-Text "IT/ET" Base64!

Text	I		T		/		E		T		
Hexcode	49		54		2F		45		54		
Binärcode	01001001		01010100		00101111		01000101		01010100	00	
6-Bit-Schnipsel	010010	010101	010000	101111	010001	010101	010000				
Dezimalwert	18	21	16	47	17	21	16				
Base64-Kode	S	V	Q	v	R	V	Q=				

- Kodieren Sie die folgende Bytes Base64: 3E F9 84 51 06 BA 2C!

Hexcode	3E		F9		84		51		06		BA		2C		
Binärcode															
6-Bit-Schnipsel															
Dezimalwert															
Base64-Kode															

E-Mail: **Base64** Dekodierungsbeispiele

- Dekodieren Sie Base64 in UNI-Code-Text!

Text								
Hexcode								
Binärcode								
6-Bit-Schnipsel								
Wert								
Base64-Kode	Q	m	F	z	Z	T	Y	0

- Dekodieren Sie Base64 in Bytes!

Hexcode							
Binärcode							
6-Bit-Schnipsel							
Wert							
Base64-Kode	8	O	H	C	s	w	==

E-Mail →MIME: Codierungsbeispiele

HOCHSCHULE MITTWEIDA UNIVERSITY OF APPLIED SCIENCES

- Text, dessen base64-Codierung wie folgt lautet: **`RVQwMXdLMQ==`**

- Geben Sie den base64-Code folgender Binärdatei an:

```
34   01   ff   56   03   02   20   67   67   fe   fc
```

Literatur

- Stevens: TCP/IP Illustrated, Volume 1 The Protocols
  Addison Wesley 1994, ISBN 0-201-63346-9
- Halsall: Data Communications, Computer Networks and Open Systems
  Addison Wesley 1996, ISBN 0-201-42293-X
- Stein: Taschenbuch Rechnernetze und Internet, Fachbuchverlag Leipzig, ISBN 3-446-21542-5
- Henning: Taschenbuch Multimedia, Fachbuchverlag Leipzig, ISBN 3-446-21751-7
- Badach, Hoffmann: Technik der IP-Netze, ISBN 3-446-21501-8
- RFC‘s als *.txt → http://www.ietf.org (internet engineering task force)
- RFC‘s als *htm → http://tools.ietf.org/html/
- ITU-T: Message handling system and service overview, Recommendation F.400/X.400
- Links:
  - http://packetlife.net u.a. sehr viele WireShark-Captures für sehr viele Protokolle