Verteilte Systeme 1

Technologien des World Wide Web

christian.zirpins@hs-karlsruhe.de

HTTP, die Sprache des Web

Hochschule Karlsruhe
Technik und Wirtschaft
UNIVERSITY OF APPLIED SCIENCES

Ein Wort vom W3C…

Nach dieser Vorlesung können Sie…

- …*erklären*, wie **Web Server und Web Clients** interagieren (über **TCP/IP** und **HTTP**).
- …**HTTP-Messages** *schreiben*, die **Web Ressourcen** von Web Servern anfragen und die Antworten *verstehen*.
- …die unterschiedlichen Komponenten von **URLs** und deren Bedeutung *erklären*.

World Wide Web vs. Internet

Das Web: eine kurze Geschichte

World Wide Web (WWW): ein globales System verbundener *Hypertext Dokumente*, die über das *Internet* verfügbar sind.

- **1960er**: Das US Department of Defense realisiert den Vorgänger des Internets (ARPANET).
  - Erste Dienste: Electronic Mail, File Transfer
- **Späte 1980er**: Das Internet öffnet sich kommerziellen Interessen.
- **1989**: Tim Berners-Lee (CERN) entwickelt das WWW.
- **1994**: Netscape veröffentlicht ersten Web Browser.
- **1995**: Microsoft veröffentlicht Internet Explorer v1.
- **1998**: Google wird gegründet.
- **2002**: Mozilla veröffentlicht Firefox v1.6.

Schlüsselaspekte des Internet

Internet: zusammenhängende Computer Netzwerke (Subnetze), die die Erde umspannen; sie kommunizieren über einen gemeinsamen Standard (TCP/IP).

- Subnetze funktionieren **autonom**!
- **Keine zentralisierte** (globale) Kontrollinstanz.
- Geräte betreten und verlassen das Netz **dynamisch.**
- Geräte interagieren über **vereinbarte offene Standards**; jeder kann ein neues Gerät entwickeln.
- **Einfach** zu verwenden; Server/Client Software ist weit verbreitet.

Zwei wichtige Organisationen

- Internet Engineering Taskforce (IETF)

The mission of the IETF is to make the Internet work better by producing high quality, relevant technical documents that influence the way people design, use, and manage the Internet.

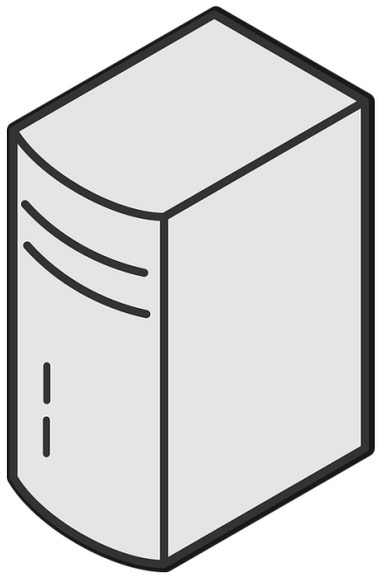
Request for Comments (RFC)

- World Wide Web Consortium (W3C)

The W3C mission is to lead the World Wide Web to its full potential by developing protocols and guidelines that ensure the long-term growth of the Web.

HTTP-Nachrichten

Web Server und Clients



(1) HTTP Request

(2) HTTP Response

Anzeigen

Ausführen

Music Player

Acrobat Reader

- Server warten auf Daten-Anforderungen (Requests)
- Antworten tausenden Clients gleichzeitig
- Stellen **Web Ressourcen** bereit

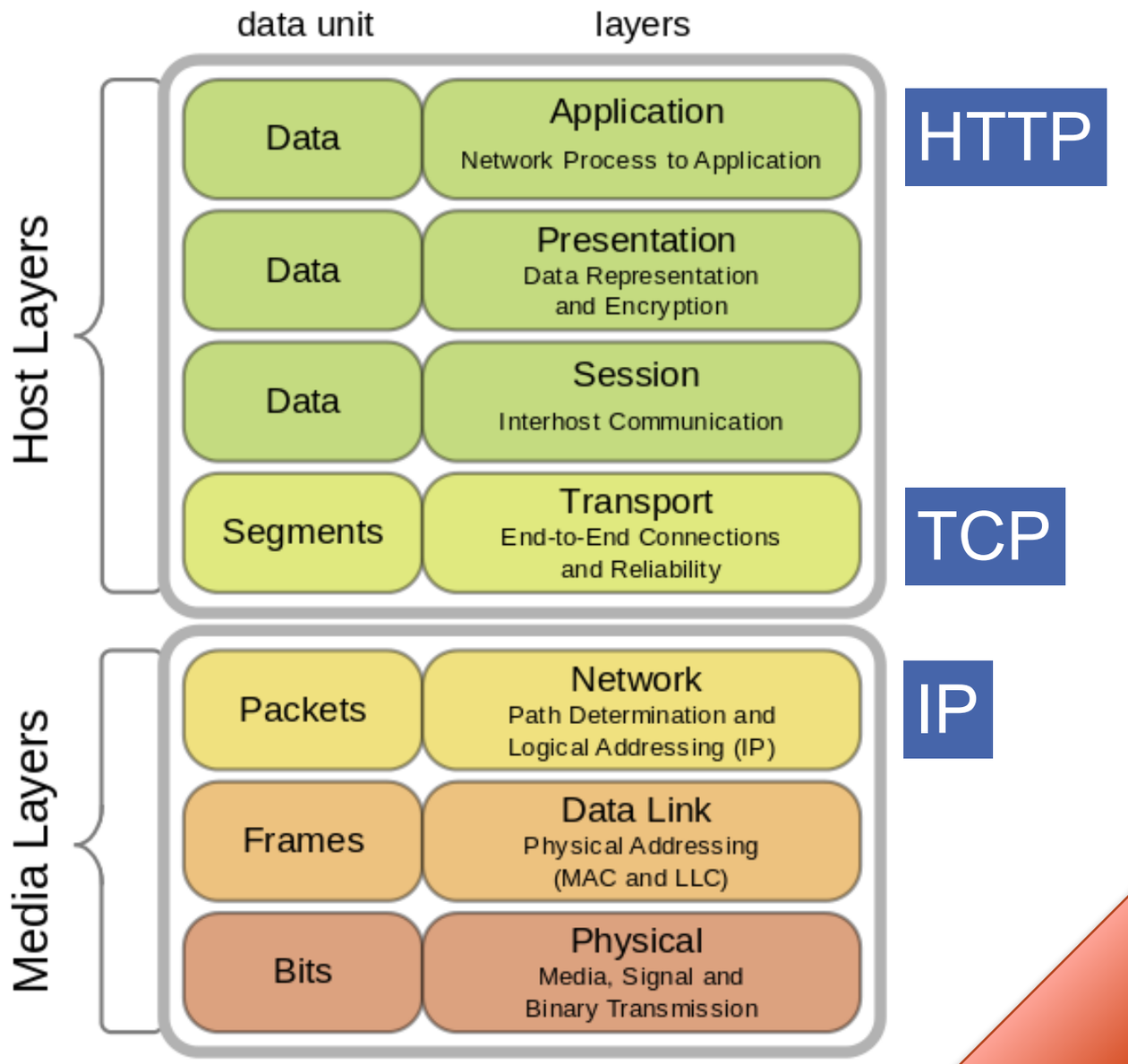
- Clients sind meistens **Web Browser**
- Telnet

Web Ressource: jede Art von Inhalt mit einer Identität, wie statische Dateien (z.B. Text, Bilder, Video), Software Programme, Webcam etc.

Netzwerkkommunikation

- Konzeptionelles Modell **Open Systems Interconnection (OSI).**
- Netzwerk Protokolle werden verschiedenen “Schichten” zugeordnet.
- Es gibt viele Netzwerkprotokolle (SSH, IMAP, SMTP, FTP usw.), drei davon sind interessant für uns:

IP: Internet Protocol
TCP: Transmission Control Protocol
HTTP: Hypertext Transfer Protocol



HTTP nutzt **verläßliche** Datenübertragungsprotokolle

Lehrbuch Modell: Kommunikationsnetze

Demo: www.hs-karlsruhe.de

Moderne Web Seiten beinhalten viele Ressourcen.

Eine Kaskade von HTTP-Transaktionen ist nötig.

Meist findet man verschiedene MIME-Typen.

Firebug* & Co: essentielle Tools für die Web Entwicklung.

*Seit *Firefox Quantum* sind die Firebug Funktionen Teil der "*Web-Entwickler Werkzeuge*". Danke Firebug für 12 Jahre Pionierarbeit!

Laden Sie die HsKA Seite und öffnen Sie Firebug/Entwicklertools. HTTP Requests/Responses und Cookies werden sichtbar (u.a.)

HTTP-Request Message

Nur Text: zeilenorientierte Zeichenfolgen

```
GET /home.html HTTP/1.1
Host: www.hs-karlsruhe.de
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X
10.12; rv:47.0) Gecko/20100101 Firefox/47.0
Accept: text/html,application/xhtml+xml,
application/xml;q=0.9,*/*;q=0.8
Accept-Language: de,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Cookie: _pk_id.1.ad89=0a5649e382c8b5e7…
Connection: keep-alive
Cache-Control: max-age=0
```

Frage: was ist das größte Problem des Plain Text Formats im Vergleich zu einem Binärformat?

https://www.w3.org/People/Bos/DesignGuide/binary-or-text.html

HTTP-Response Message

```
HTTP/1.1 200 OK
```

Startzeile

```
Date: Thu, 22 Sep 2016 14:52:37 GMT
Server: Apache/2.4.7 (Ubuntu)
X-Powered-By: PHP/5.5.9-1ubuntu4.19
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Type: text/html; charset=utf-8
Keep-Alive: timeout=5, max=500
Connection: Keep-Alive
Transfer-Encoding: chunked
```

Headerfelder

`name:wert`

```
.....
.....
```

Body (optional)

HTTP-Header im Einzelnen

Headerfelder zur Beschreibung von Inhalten

Content-Type	Der MIME-Typ der angeforderten Datei / des gesendeten Body
Content-Length	Länge des Body in Bytes
Content-Language	Die Sprache, in der die Datei vorliegt
Content-Encoding	Codierung des Inhalts
Content-Location	Alternativer Name/Speicherplatz für das angeforderte Element
Content-Range	Welchen Bereich des Gesamtbodys der gesendete Inhalt abdeckt
Content-MD5	Die Base64-codierte MD5-Checksumme des Body
Last-Modified	Zeitpunkt der letzten Änderung an der Datei
Expires	Ab wann die Datei als veraltet angesehen werden kann
Allow	Erlaubte Aktionen für eine bestimmte Ressource

Quelle: Wikipedia

Wichtig: der Message Body enthält nur rohe Daten, zur Interpretation werden Header-Informationen benötigt.

Content-Type

- Alle HTTP-Datenobjekte sind mit **MIME-Typen** gekennzeichnet.
  - **Multipurpose Internet Mail Extensions** (historische Gründe)
  - Moderne Bezeichnung: **(Internet) Media Type**
- Der MIME-Typ bestimmt die Reaktion des Clients auf die empfangenen Daten.
- Muster: **`[Primärerer Typ]/[Subtyp]`**
  - e.g. **`text/plain`**, **`text/html`**, **`image/jpeg`**, **`video/quicktime`**, **`application/vnd.ms-powerpoint`**

Content-Types gibt es viele

Die populärsten Typen
text/html
image/jpg
text/xml
application/rss+xml
text/plain
application/xml
text/calendar
application/pdf
application/atom+xml
unknown/unknown

Die unpopulärsten Typen
application/pgp-keys
application/x-httpd-php4
chemical/x-pdb
model/mesh
application/x-perl
audio/x_mpegurl
application/bib
application/postscript
application/x-msdos-program

Content-Length

- Gibt die **Größe** des Message Body an.
- Hierdurch kann ein vorzeitiger **Nachrichtenabbruch** erkannt werden (z.B. durch einen Server Crash oder fehlerhaften Proxy).
- Essentiell für **persistente Verbindungen**, um zu erkennen, wann eine HTTP Nachricht endet und die nächste Nachricht beginnt.

Persistente Verbindungen verwenden dieselbe TCP Verbindung für mehrere HTTP Request/Response Messages.

Content-Encoding

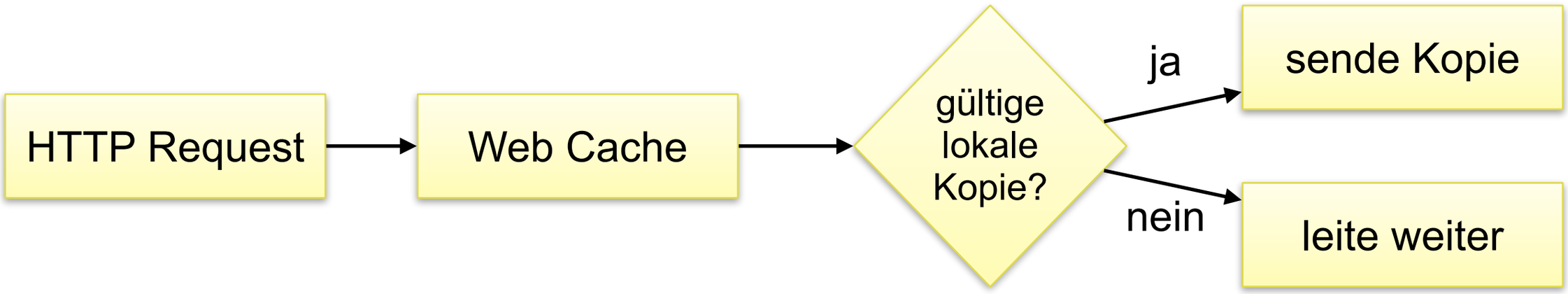
- Häufig entweder *gzip*, *compress* (Unix-Komprimierung), *deflate* (zlib Kompression) oder *identity* (keine Kodierung).
- Server wählen **Kodierungen**, die Clients verstehen können.
  - Clients senden eine Liste akzeptabler Kodierungen im **`Accept-Encoding`** Request-Header.
- **Komprimierung** spart Netzwerkbandbreite (weniger Bits zu übertragen), aber erhöht Verarbeitungskosten zum dekomprimieren.

Content-MD5

- HTTP-Nachrichten werden über TCP/IP gesendet (garantiert zuverlässiger Transport).
- **Aber**: Das Internet ist groß; *viele Server interagieren* um eine Nachricht mit unterschiedlichen Implementierungen von etablierten Protokollen zu transportieren (die *fehlerhaft* sein können).
- **Plausibiltätsprüfung**: Absender erzeugt eine **Prüfsumme** (MD5) des Inhalts um unbeabsichtigte Änderungen zu erkennen.

Expires

- **Web-Caches** halten Kopien von *beliebten* Dokumenten..



- **Vorteile**:
  A. …reduziert *redundante Datenübertragung*
  B. …reduziert *Engpässe im Netzwerk*
  C. …geringere *Nachfrage auf den Ursprungsservern*
  D. …reduziert *Entfernungsverzögerung ("Latenz")*

- **`Expires`** zeigt an, wann eine Ressource nicht mehr gültig ist und vom Ursprungsserver abgerufen werden muss.

Hinweis: Web-Browser speichern auch zwischen, wenn möglich.

Expires und Cache-Management

- Inhalt auf dem Ursprungsserver kann sich ändern.
- Caches müssen sicherstellen, dass ihre Kopien **konsistent** mit dem Ursprungsserver sind.
- Caches können ihre Kopien jederzeit mit einer beliebigen Frequenz erneuern (**aber**: dies wäre nicht effizient).
- **`Expires`** im HTTP-Antwort-Header zeigt ein **Ablaufdatum des Dokuments** in **absoluten** Zahlen — legt fest, wie lange der Inhalt frisch ist; Cache Erneuerung zu diesem Zeitpunkt.
- Alternative zu **`Expires`** ist **`Cache-Control`**: zeigt ein Ablaufdatum des Dokuments in **relativen** Zahlen (Anzahl der Sekunden seit gesendet wurde).

Last-Modified

- Enthält das Datum, zu dem das Dokument zuletzt **geändert** wurde (in HTTP-Antwort).
- Keine Aussage über die Menge der Änderungen im Dokument.
- Oft verwendet in Kombination mit **`If-Modified-Since`** für Cache Erneuerung — Ursprungsserver gibt das Dokument nur zurück, wenn es seit dem angegebenen Datum geändert wurde. (sonst **`304 Not Modified`**).
- **`Last-Modified`** Daten sind **nicht immer zuverlässig** (z.B. durch den Ursprungsservern manipuliert, um hohe Raten an Cache Erneuerungen zu gewährleisten).

Zur Erinnerung: HTTP-Response Message

```
HTTP/1.1 200 OK
```

Startzeile

```
Date: Thu, 22 Sep 2016 14:52:37 GMT
Server: Apache/2.4.7 (Ubuntu)
X-Powered-By: PHP/5.5.9-1ubuntu4.19
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Type: text/html; charset=utf-8
Keep-Alive: timeout=5, max=500
Connection: Keep-Alive
Transfer-Encoding: chunked
```

Headerfelder

`name:wert`

```
.....
.....
```

Body (optional)

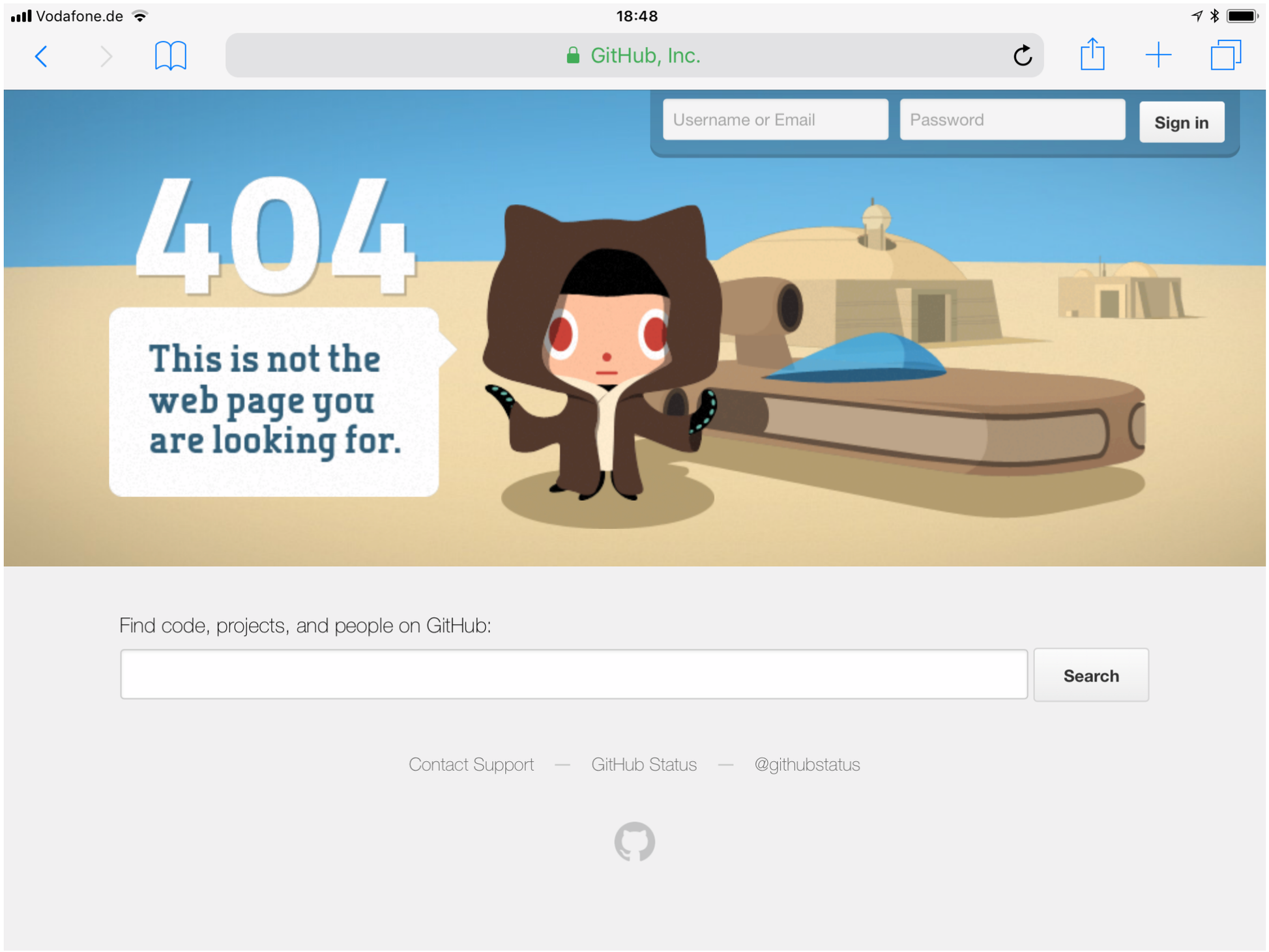
Gängige Status Codes

1xx	Informationen
2xx	Erfolgreiche Operation (200 OK)
3xx	Umleitung
4xx	Client Fehler (404 Not Found)
5xx	Server Fehler

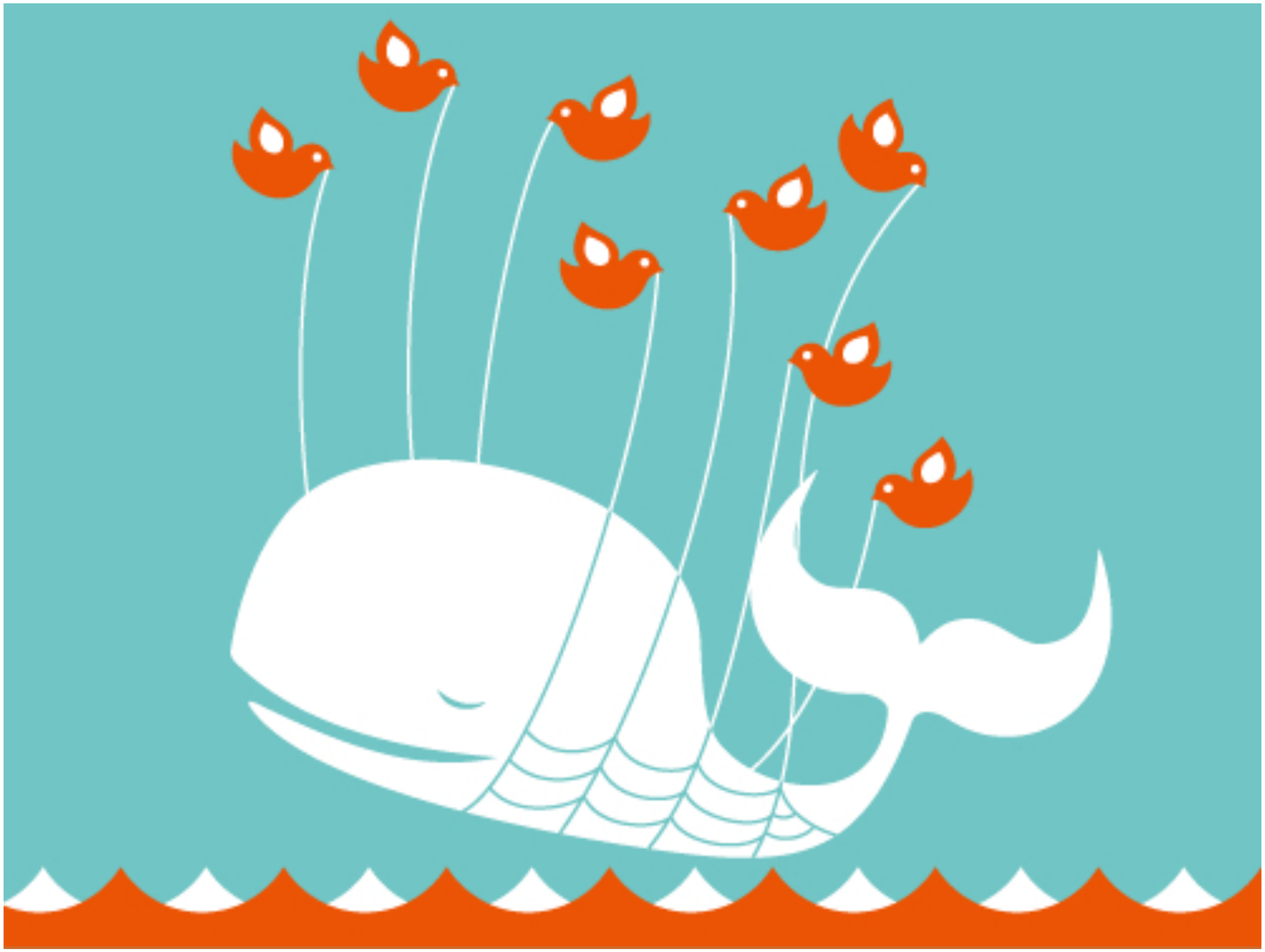
In der Praxis werden nur wenige Codes pro Kategorie unterstützt

Click für detaillierten Überblick

https://github.com/therealcoockiemonster



„Lifting a Dreamer“



HTTP-Methoden

Gängige HTTP Methoden

GET	Hole ein Dokument vom Web Server
HEAD	Hole den Header eines Dokumentes vom Web Server
POST	Sende Daten vom Client zum Server zur Verarbeitung
PUT	Speichere den Request Body auf dem Server
TRACE	Verfolge die Nachricht zum Server (z.B. durch Proxy)
OPTIONS	Hole unterstützte Methoden vom Server
DELETE	Entferne ein Dokument vom Server

Frage: In welchen Situationen könnte HEAD nützlich sein?

Demo: Telnet

Telnet öffnet TCP-Verbindung zum Web Server; Zeichen werden direkt in den Port eingegeben. Der Server behandelt Telnet als Web Client, die Response wird auf dem Bildschirm angezeigt.

Öffnen Sie ein Terminal und starten Sie telnet.

```
$ telnet www.hs-karlsruhe.de 80
```

Spielen Sie einige GET/HEAD Beispiele durch.
PUT/POST/DELETE Requests sind etwas komplizierter.

Z.B.

```
GET / HTTP/1.1
Host: www.hs-karlsruhe.de
```

Das ist eine minimale HTTP Request Nachricht!

Mehrere HTTP-Requests können nötig sein, um auf der endgültigen (gewollten) Seite zu landen.

TCP & HTTP: Anzeige einer einfachen HTML Ressource im Browser

Hochschule Karlsruhe
Technik und Wirtschaft
UNIVERSITY OF APPLIED SCIENCES

Domain Name **Port**

`http://www.microsoft.com:80/index.html`

Pfad

1. Browser extrahiert Domain Name aus URL
2. Browser wandelt Domain Name in IP Adresse
3. Browser extrahiert Port Nummer (default: 80)
4. Browser baut TCP Verbindung mit Web Server auf
5. Browser sendet HTTP Request
6. Web Server sendet HTTP Response
7. Verbindung ist geschlossen, Browser zeigt das **Dokument**

HTTP ist *zustandslos*. Jede/r Request/Response ist isoliert.

TCP & HTTP: Anzeige einer einfachen HTML Ressource im Browser

TCP liefert…

- *fehlerfreie* Datenübertragung
- *geordnete* Lieferung
- einen nicht segmentierten *Datenstrom*

`http://www.microsoft.com:80/index.html`

`http://65.55.57.27:80/index.html`

Domain Name

Port

Pfad

IP Adresse

32-Bit IP Adresse: vier 8-Bit Werte (0-255)

Uniform Resource Locators (URLs)

Uniform Resource Locators (URLs)

Frage: Welche der folgenden URLs sind gültig?

A. mailto:christian.zirpins@hs-karlsruhe.de

B. ftp://anonymous:mypass@ftp.halifax.rwth-aachen.de/gnu/

C. http://www.bing.com/?scope=images&nr=1#top

D. https://duckduckgo.com/html?q=karlsruhe

E. http://myshop.nl/comp;typ=c/apple;class=a;date=today/index.html;fr=delft

F. http://правительство.рф

Uniform Resource Locators (URLs)

Frage: Welche der folgenden URLs sind gültig?

A. mailto:christian.zirpins@hs-karlsruhe.de

B. ftp://anonymous:mypass@ftp.halifax.rwth-aachen.de/gnu/

C. http://www.bing.com/?scope=images&nr=1#top

D. https://duckduckgo.com/html?q=karlsruhe

E. http://myshop.nl/comp;typ=c/apple;class=a;date=today/index.html;fr=delft

F. http://правительство.рф

Alle sind syntaktisch gültig!

URL Syntax

- **Uniform Resource Locators (URLs)** bieten einen standardisierten Weg, um *auf jede Ressource im Internet zu verweisen*.
- Nicht auf das HTTP-Scheme beschränkt, die Syntax variiert für verschiedene Schemes.
- Generelles Format (von den meisten Schemes eingehalten):

<scheme>://<user>:<password>@<host>:<port>/<path>;<params>?<query>#<frag>

Bestimmt das Protokoll zur Verbindung mit dem Server.

URL Syntax

- **Uniform Resource Locators (URLs)** bieten einen standardisierten Weg, um *auf jede Ressource im Internet zu verweisen*.
- Nicht auf das HTTP-Scheme beschränkt, die Syntax variiert für verschiedene Schemes.
- Generelles Format (von den meisten Schemes eingehalten):

<scheme>://<user>:<password>@<host>:<port>/<path>;<params>?<query>#<frag>

Benutzername/Passwort (ggf. nötig um eine Ressource zuzugreifen)

URL Syntax

- **Uniform Resource Locators (URLs)** bieten einen standardisierten Weg, um *auf jede Ressource im Internet zu verweisen*.
- Nicht auf das HTTP-Scheme beschränkt, die Syntax variiert für verschiedene Schemes.
- Generelles Format (von den meisten Schemes eingehalten):

<scheme>://<user>:<password>@<host>:<port>/<path>;<params>?<query>#<frag>

Domain Name (Host Name) oder IP Adresse des Servers.

URL Syntax

- **Uniform Resource Locators (URLs)** bieten einen standardisierten Weg, um *auf jede Ressource im Internet zu verweisen*.
- Nicht auf das HTTP-Scheme beschränkt, die Syntax variiert für verschiedene Schemes.
- Generelles Format (von den meisten Schemes eingehalten):

<scheme>://<user>:<password>@<host>:<port>/<path>;<params>?<query>#<frag>

Der Port an dem der Server Requests für die Ressource erwartet.

URL Syntax

- **Uniform Resource Locators (URLs)** bieten einen standardisierten Weg, um *auf jede Ressource im Internet zu verweisen*.
- Nicht auf das HTTP-Scheme beschränkt, die Syntax variiert für verschiedene Schemes.
- Generelles Format (von den meisten Schemes eingehalten):

<scheme>://<user>:<password>@<host>:<port>/<path>;<params>?<query>#<frag>

Der lokale Pfad (Name) der Ressource auf dem Server.

URL Syntax

- **Uniform Resource Locators (URLs)** bieten einen standardisierten Weg, um *auf jede Ressource im Internet zu verweisen*.
- Nicht auf das HTTP-Scheme beschränkt, die Syntax variiert für verschiedene Schemes.
- Generelles Format (von den meisten Schemes eingehalten):

<scheme>://<user>:<password>@<host>:<port>/<path>;<params>?<query>#<frag>

Zusätzliche Eingabeparameter, die von einer Anwendung ggf. benötigt werden, um auf eine Ressource des Servers korrekt zuzugreifen. Können pro Pfadsegment gesetzt werden.

URL Syntax

- **Uniform Resource Locators (URLs)** bieten einen standardisierten Weg, um *auf jede Ressource im Internet zu verweisen*.
- Nicht auf das HTTP-Scheme beschränkt, die Syntax variiert für verschiedene Schemes.
- Generelles Format (von den meisten Schemes eingehalten):

<scheme>://<user>:<password>@<host>:<port>/<path>;<params>?<query>#<frag>

Parameter für eine Gateway Ressource, vor allem Anwendungen (identifiziert über den Pfad) wie z.B. eine Suchmaschine.

URL Syntax

- **Uniform Resource Locators (URLs)** bieten einen standardisierten Weg, um *auf jede Ressource im Internet zu verweisen*.
- Nicht auf das HTTP-Scheme beschränkt, die Syntax variiert für verschiedene Schemes.
- Generelles Format (von den meisten Schemes eingehalten):

<scheme>://<user>:<password>@<host>:<port>/<path>;<params>?<query>#<frag>

Der Name eines Teils der Ressource. Wird nur vom Client verwendet - das Fragment wird nicht zum Server übermittelt.

URL Syntax: Query

<scheme>://<user>:<password>@<host>:<port>/<path>;<params>?<query>#<frag>

https://duckduckgo.com/html?q=karlsruhe

- Query-Teil wird an die Anwendung auf dem Server übergeben.
- Nützlich um interaktive Anwendungen zu ermöglichen.
- Gängige Konvention: **`name1=wert1&name2=wert2&`**...

Schemes: mehr als nur HTTP

`http://<host>:<port>/<path>?<query>#<frag>`

`https` ist analog zu `http` mit `https` als Scheme
Name (Ende-zu-Ende Verschlüsselung von HTTP Verbindungen)

`mailto:<valid-email-address>`

`ftp://<user>:<password>@<host>:<port>/<path>;<params>`

`file://<host>/<path>`, z.B.
`file:///Users/admin/tmp.html`

Relative und absolute URLs

Basis URL

`http://www.acme.com/~peter/new/index.html` absolut

```
<h1>My Hobbies</h1>
<ol>
 <li><a href="rides/biking.html">Bike</a></li>
 <li><a href="../ski.html">Ski</a></li>
</ol>
```

relativ

nicht trivial: URLs können komplex sein

```
http://www.acme.com/~peter/new/rides/biking.html
http://www.acme.com/~peter/ski.html
```

Frage: Was ist der wichtigste Vorteil von relativen URLs?

Zusammenfassung

HTTP stellt sicher, dass Inhalte …

A. …**korrekt identifiziert** werden können.

B. …**richtig entpackt** werden können.

C. …**aktuell** sind.

D. …in **passendem Format** vorliegen.

E. …**vollständig** und **unverfälscht** ankommen.

Wir haben folgendes behandelt

- World Wide Web vs. Internet
- Einführung in HTTP-Nachrichten
- HTTP-Inhalte im einzelnen (Header, Status, Operationen…)
- Uniform Resource Locators

Literatur (heutige VL)

- Empfehlung: *HTTP: The Definitive Guide* (O'Reilly, 2002), Kapitel 1, 2, 3 und 12

- Die nächste Vorlesung setzt HTML Grundkenntnisse voraus!
  - **Bitte lesen Sie *vorab*:**
    Learning Web App Development, Kapitel 2