HTTP: the language of Web communication

Claudia Hauff
TI1506: Web and Database Technology
ti1506-ewi@tudelft.nl

TU Delft

Course overview [Web]

1. **http: the language of Web communication**
2. HTML & Web app design
3. JavaScript: interactions in the browser
4. node.js: JavaScript on the server
5. CSS: Lets make the app pretty
6. Ajax: asynchronous JavaScript
7. Personalization: Cookies & sessions
8. Securing your application

Source: https://vimeo.com/110256895

Source: https://vimeo.com/110256895

At the end of this lecture, you should be able to …

- **Describe** how Web servers and clients interact with each other (via TCP/IP and HTTP)
- **Write** HTTP messages that request Web resources from Web servers and understand the responses
- **Describe** the different components of URLs and their purpose
- **Understand** and **employ** basic HTTP authentication
- **Explain** the difference between HTTP and HTTPS

World Wide Web

vs.

Internet

The Web: a brief history

World Wide Web: a global system of interconnected hypertext documents available via the Internet

(envisioned already in 1945)

Image sources: Wikipedia & Techspot

The Web: a brief history

World Wide Web: a global system of interconnected hypertext documents available via the Internet

(envisioned already in 1945)

- **1960s**: Precursor to the Internet (ARPANET) devised by the US department of Defense

Image sources: Wikipedia & Techspot

The Web: a brief history

World Wide Web: a global system of interconnected hypertext documents available via the Internet

(envisioned already in 1945)

- **1960s**: Precursor to the Internet (ARPANET) devised by the US department of Defense
- Initial services: electronic mail, file transfer

Image sources: Wikipedia & Techspot

The Web: a brief history

World Wide Web: a global system of interconnected hypertext documents available via the Internet

(envisioned already in 1945)

- **1960s**: Precursor to the Internet (ARPANET) devised by the US department of Defense
  - Initial services: electronic mail, file transfer
- Late **1980s**: Internet opened to commercial interests

Image sources: Wikipedia & Techspot

The Web: a brief history

World Wide Web: a global system of interconnected hypertext documents available via the Internet

(envisioned already in 1945)

- **1960s**: Precursor to the Internet (ARPANET) devised by the US department of Defense
  - Initial services: electronic mail, file transfer
- Late **1980s**: Internet opened to commercial interests
- **1989**: WWW created by Tim Berners-Lee (CERN)

Image sources: Wikipedia & Techspot

#copyright

Your continued donations keep Wikipedia running!

Lynx (web browser)

From Wikipedia, the free encyclopedia

Jump to: navigation, search

CAPTION: Lynx

Wikipedia Main Page displayed in Lynx
Wikipedia Main Page displayed in Lynx

Maintainer:	Thomas Dickey
Stable release:	2.8.5 (February 4, 2004) [[+/-]]
Preview release:	2.8.6 (?) [[+/-]]
OS:	Cross-platform
Use:	web browser
License:	GPL
Website:	lynx.isc.org

Lynx is a text-only Web browser and Internet Gopher client for use on cursor-addressable, character cell terminals.

Browsing in Lynx consists of highlighting the chosen link using cursor keys, or having all links on a page numbered and entering the chosen link's number. Current versions support SSL and many HTML features. Tables are linearized (scrunched together one cell after another without tabular structure), while frames are identified by name and can be explored as if they were separate pages.

Lynx is a product of the Distributed Computing Group within Academic Computing Services of the University of Kansas, and was initially developed in 1992 by a team of students at the university (Lou Montulli, Michael Grobe and Charles Rezac) as a hypertext browser used solely to distribute campus information as part of a Campus-Wide Information Server. In 1993 Montulli added an Internet interface and released a new version (2.0) of the browser [1] [2] [3].

The Web: a brief history

World Wide Web: a global system of interconnected hypertext documents available via the Internet

(envisioned already in 1945)

- **1960s**: Precursor to the Internet (ARPANET) devised by the US department of Defense
  - Initial services: electronic mail, file transfer
- Late **1980s**: Internet opened to commercial interests
- **1989**: WWW created by Tim Berners-Lee (CERN)

Image sources: Wikipedia & Techspot

The Web: a brief history

World Wide Web: a global system of interconnected hypertext documents available via the Internet

(envisioned already in 1945)

- **1960s**: Precursor to the Internet (ARPANET) devised by the US department of Defense
  - Initial services: electronic mail, file transfer
- Late **1980s**: Internet opened to commercial interests
- **1989**: WWW created by Tim Berners-Lee (CERN)
- **1994**: Netscape released its first Web browser

Image sources: Wikipedia & Techspot

Mosaic Netscape version 0.9 beta

This is *BETA* software subject to the license agreement set forth in the README file.
Please read and agree to all terms before using this software.

Report any problems to win_cbug@mcom.com.

The Web: a brief history

World Wide Web: a global system of interconnected hypertext documents available via the Internet

(envisioned already in 1945)

- **1960s**: Precursor to the Internet (ARPANET) devised by the US department of Defense
  - Initial services: electronic mail, file transfer
- Late **1980s**: Internet opened to commercial interests
- **1989**: WWW created by Tim Berners-Lee (CERN)
- **1994**: Netscape released its first Web browser

Image sources: Wikipedia & Techspot

The Web: a brief history

World Wide Web: a global system of interconnected hypertext documents available via the Internet

(envisioned already in 1945)

- **1960s**: Precursor to the Internet (ARPANET) devised by the US department of Defense
  - Initial services: electronic mail, file transfer
- Late **1980s**: Internet opened to commercial interests
- **1989**: WWW created by Tim Berners-Lee (CERN)
- **1994**: Netscape released its first Web browser
- **1995**: Microsoft released Internet Explorer v1

Image sources: Wikipedia & Techspot

The Web: a brief history

World Wide Web: a global system of interconnected hypertext documents available via the Internet

(envisioned already in 1945)

- **1960s**: Precursor to the Internet (ARPANET) devised by the US department of Defense
  - Initial services: electronic mail, file transfer
- Late **1980s**: Internet opened to commercial interests
- **1989**: WWW created by Tim Berners-Lee (CERN)
- **1994**: Netscape released its first Web browser
- **1995**: Microsoft released Internet Explorer v1
- **1998**: Google was founded

Image sources: Wikipedia & Techspot

The Web: a brief history

World Wide Web: a global system of interconnected hypertext documents available via the Internet

(envisioned already in 1945)

- **1960s**: Precursor to the Internet (ARPANET) devised by the US department of Defense
  - Initial services: electronic mail, file transfer
- Late **1980s**: Internet opened to commercial interests
- **1989**: WWW created by Tim Berners-Lee (CERN)
- **1994**: Netscape released its first Web browser
- **1995**: Microsoft released Internet Explorer v1
- **1998**: Google was founded
- **2002**: Mozilla released Firefox v1

Image sources: Wikipedia & Techspot

Key aspects of the Internet

Key aspects of the Internet

Internet: interconnected computer networks (sub-networks) that span the globe; communicating through a common standard (TCP/IP)

Key aspects of the Internet

Internet: interconnected computer networks (sub-networks) that span the globe; communicating through a common standard (TCP/IP)

- Sub-networks function **autonomously**

Key aspects of the Internet

Internet: interconnected computer networks (sub-networks) that span the globe; communicating through a common standard (TCP/IP)

- Sub-networks function **autonomously**
- **No centralised** (global) control instance

Key aspects of the Internet

Internet: interconnected computer networks (sub-networks) that span the globe; communicating through a common standard (TCP/IP)

- Sub-networks function **autonomously**
- **No centralised** (global) control instance
- Devices **dynamically** join and leave the network

Key aspects of the Internet

Internet: interconnected computer networks (sub-networks) that span the globe; communicating through a common standard (TCP/IP)

- Sub-networks function **autonomously**
- **No centralised** (global) control instance
- Devices **dynamically** join and leave the network
- Devices interact through **agreed-upon open standards**; anyone can create a new device

Key aspects of the Internet

Internet: interconnected computer networks (sub-networks) that span the globe; communicating through a common standard (TCP/IP)

- Sub-networks function **autonomously**
- **No centralised** (global) control instance
- Devices **dynamically** join and leave the network
- Devices interact through **agreed-upon open standards**; anyone can create a new device
- **Easy** to use: server/client software is widely available

Two important organisations

- **Internet Engineering Task Force (IETF)**

- **World Wide Web Consortium (W3C)**

Two important organisations

- **Internet Engineering Task Force (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. "

- **World Wide Web Consortium (W3C)**

Two important organisations

- **Internet Engineering Task Force (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. "

- **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."

Two important organisations

- **Internet Engineering Task Force (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."

An introduction to HTTP messages

HTTP 1.1

RFC 2068

1997

HTTP/2

RFC 7540

2015

HTTP 1.1

RFC 2068

1997

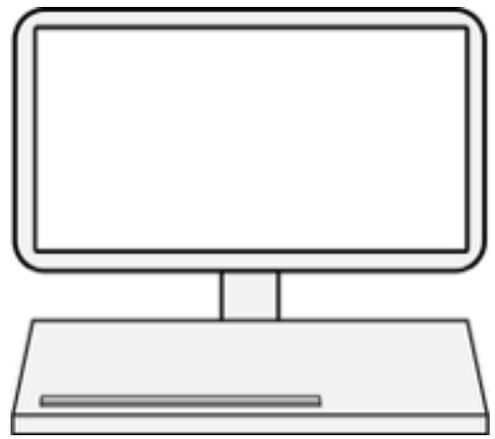
HTTP/2

RFC 7540

2015

Web servers and clients

- Servers wait for data requests
- Answer thousands of clients simultaneously
- Host **web resources**

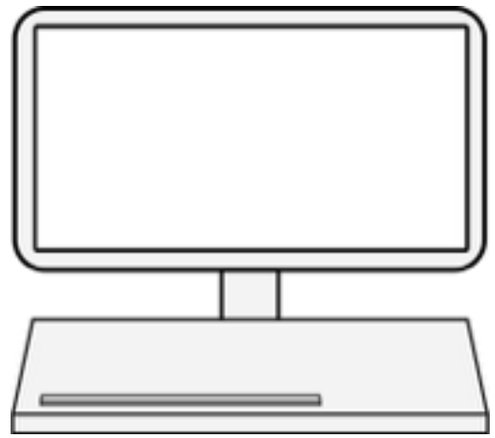


- Clients are most often Web browsers
- Telnet

Image source: openclipart.org

Web servers and clients

- Servers wait for data requests
- Answer thousands of clients simultaneously
- Host **web resources**



- Clients are most often Web browsers
- Telnet

Web resource: any kind of content with an identity, including static files (e.g. text, images, video), software programs, Web cam gateway, etc.

Image source: openclipart.org

Web servers and clients

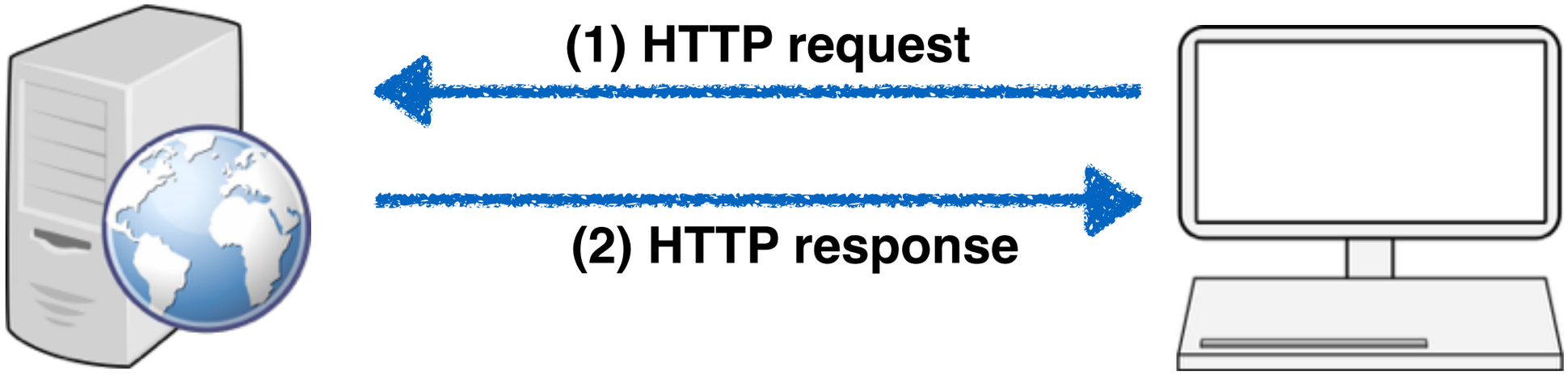
- Servers wait for data requests
- Answer thousands of clients simultaneously
- Host **web resources**

- Clients are most often Web browsers
- Telnet

Web resource: any kind of content with an identity, including static files (e.g. text, images, video), software programs, Web cam gateway, etc.

Image source: openclipart.org

Web servers and clients



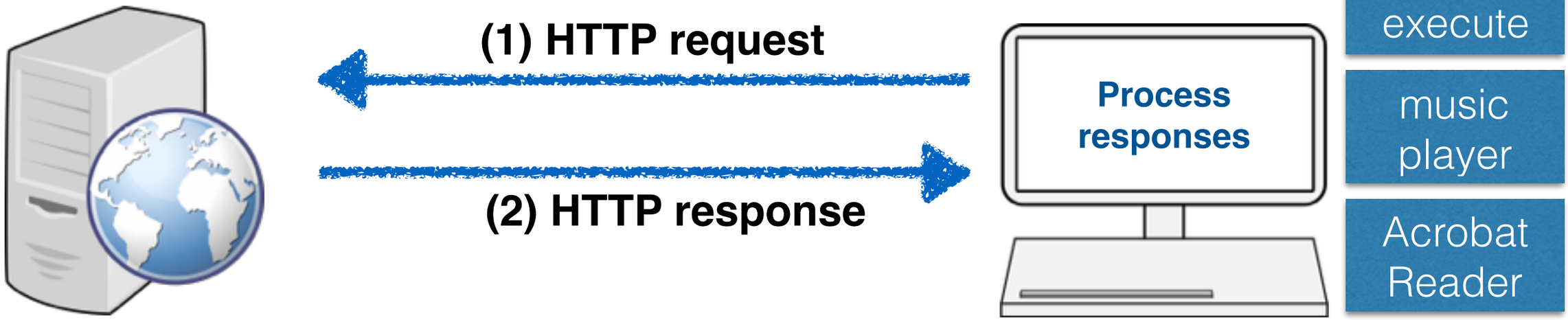
- Servers wait for data requests
- Answer thousands of clients simultaneously
- Host **web resources**

- Clients are most often Web browsers
- Telnet

Web resource: any kind of content with an identity, including static files (e.g. text, images, video), software programs, Web cam gateway, etc.

Image source: openclipart.org

Web servers and clients



- Servers wait for data requests
- Answer thousands of clients simultaneously
- Host **web resources**

- Clients are most often Web browsers
- Telnet

Web resource: any kind of content with an identity, including static files (e.g. text, images, video), software programs, Web cam gateway, etc.

Image source: openclipart.org

Network communication

- Conceptual model Open Systems Interconnection (OSI)
- Network protocols are matched to different “layers”
- Many network protocols exist, three are of interest to us:

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

others include: SSH, IMAP, STMP, FTP ….

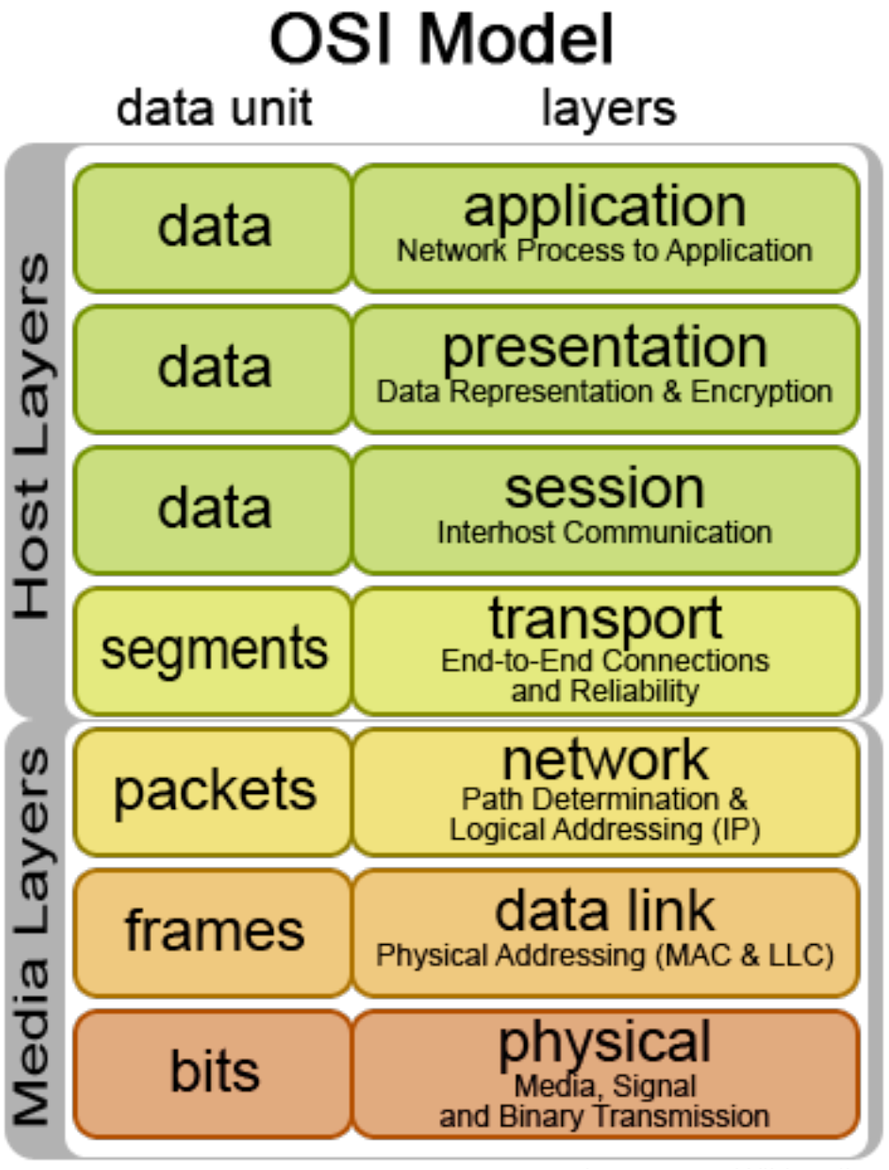


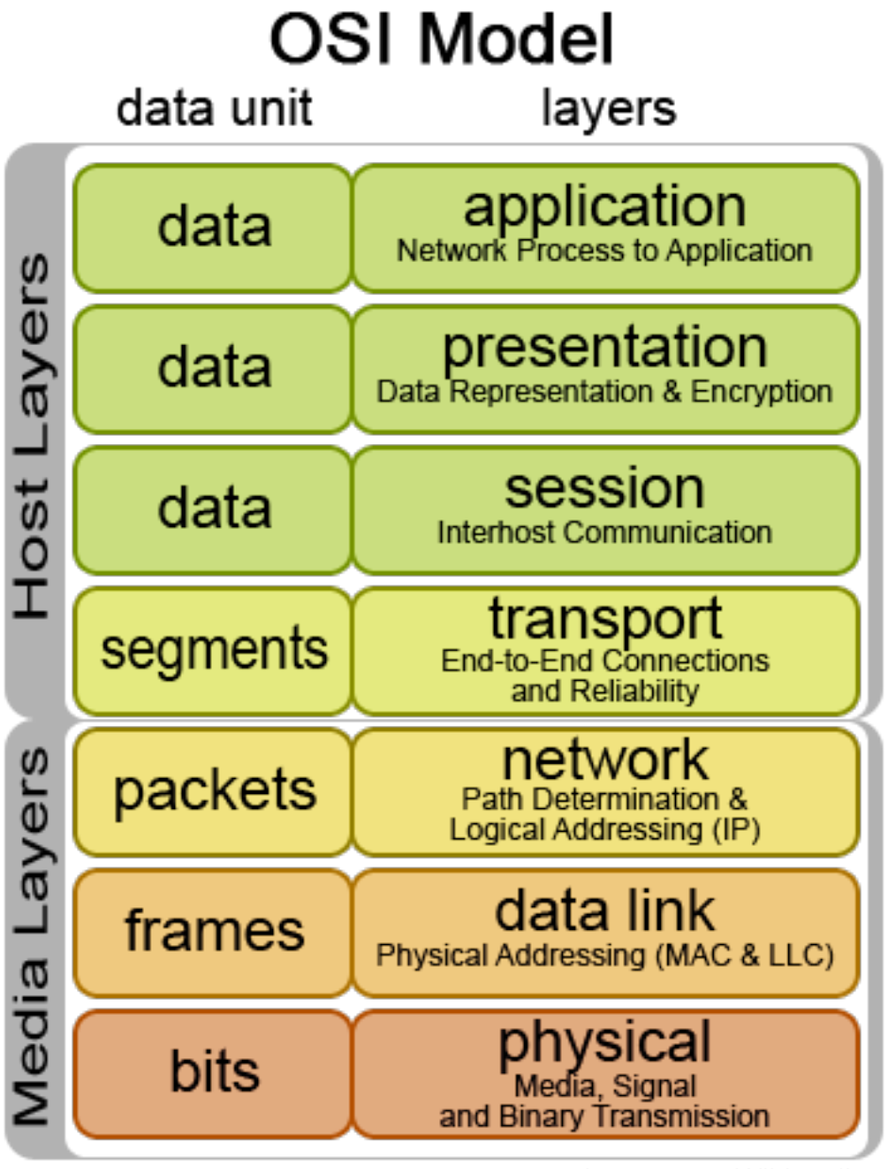
Image src: Wikipedia

Network communication

- Conceptual model Open Systems Interconnection (OSI)
- Network protocols are matched to different “layers”
- Many network protocols exist, three are of interest to us:

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

others include: SSH, IMAP, STMP, FTP ….



HTTP

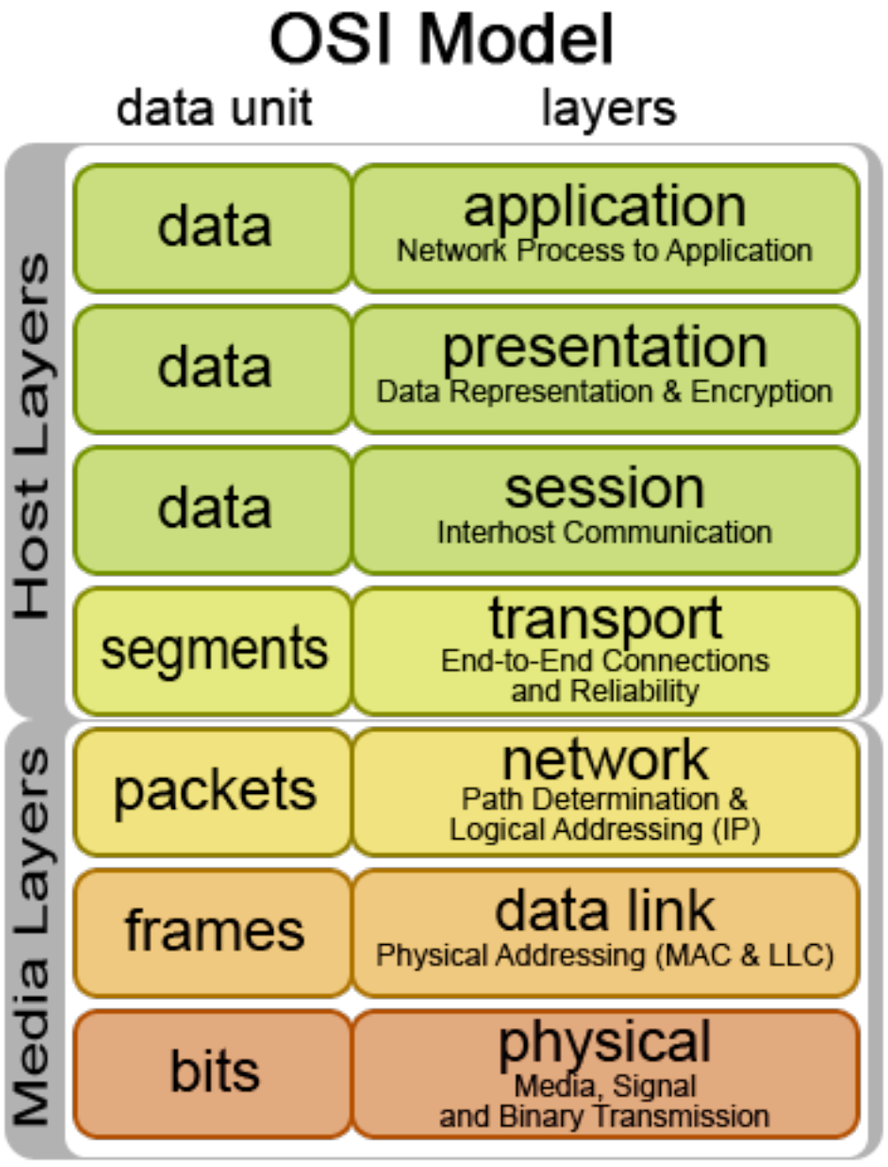
Image src: Wikipedia

Network communication

- Conceptual model Open Systems Interconnection (OSI)
- Network protocols are matched to different “layers”
- Many network protocols exist, three are of interest to us:

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

others include: SSH, IMAP, STMP, FTP ….



HTTP

TCP

Image src: Wikipedia

Network communication

- Conceptual model Open Systems Interconnection (OSI)
- Network protocols are matched to different “layers”
- Many network protocols exist, three are of interest to us:

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

others include: SSH, IMAP, STMP, FTP ….

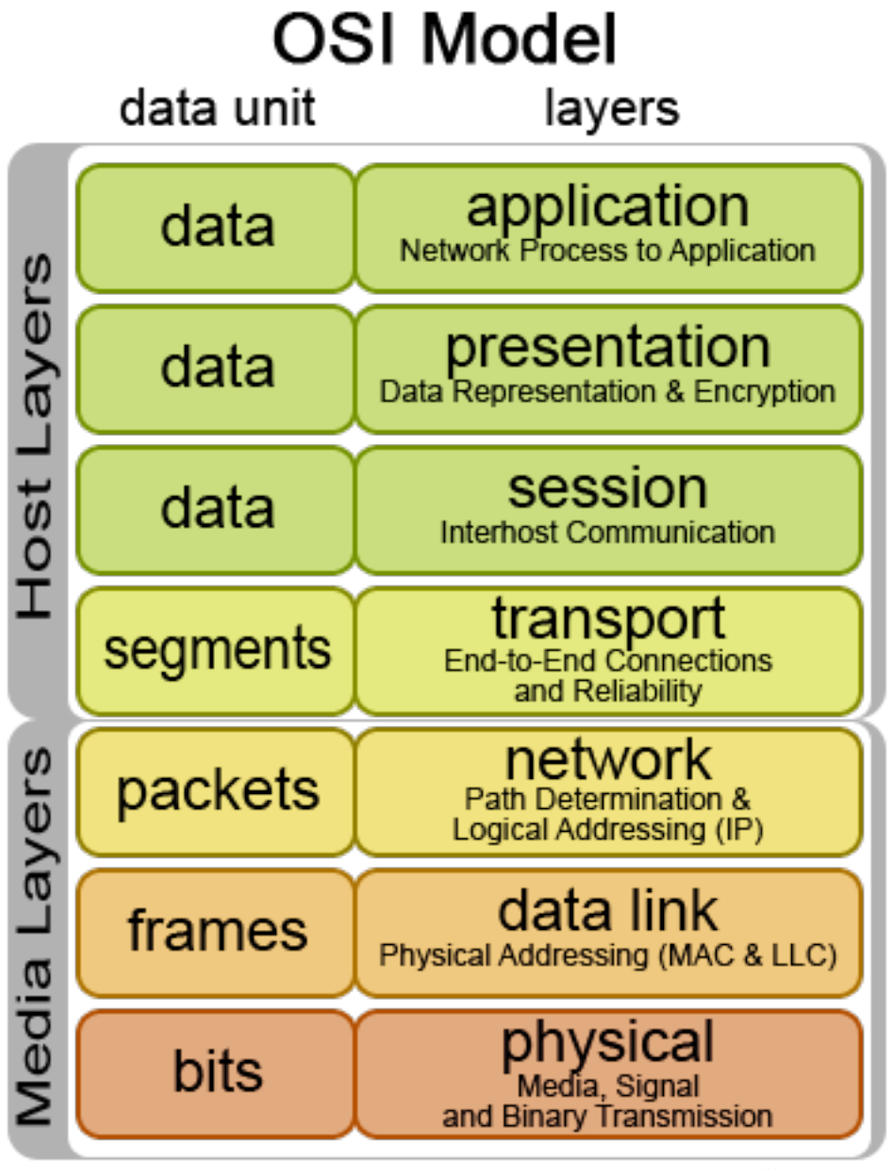


Image src: Wikipedia

Network communication

- Conceptual model Open Systems Interconnection (OSI)
- Network protocols are matched to different “layers”
- Many network protocols exist, three are of interest to us:

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

others include: SSH, IMAP, STMP, FTP ….

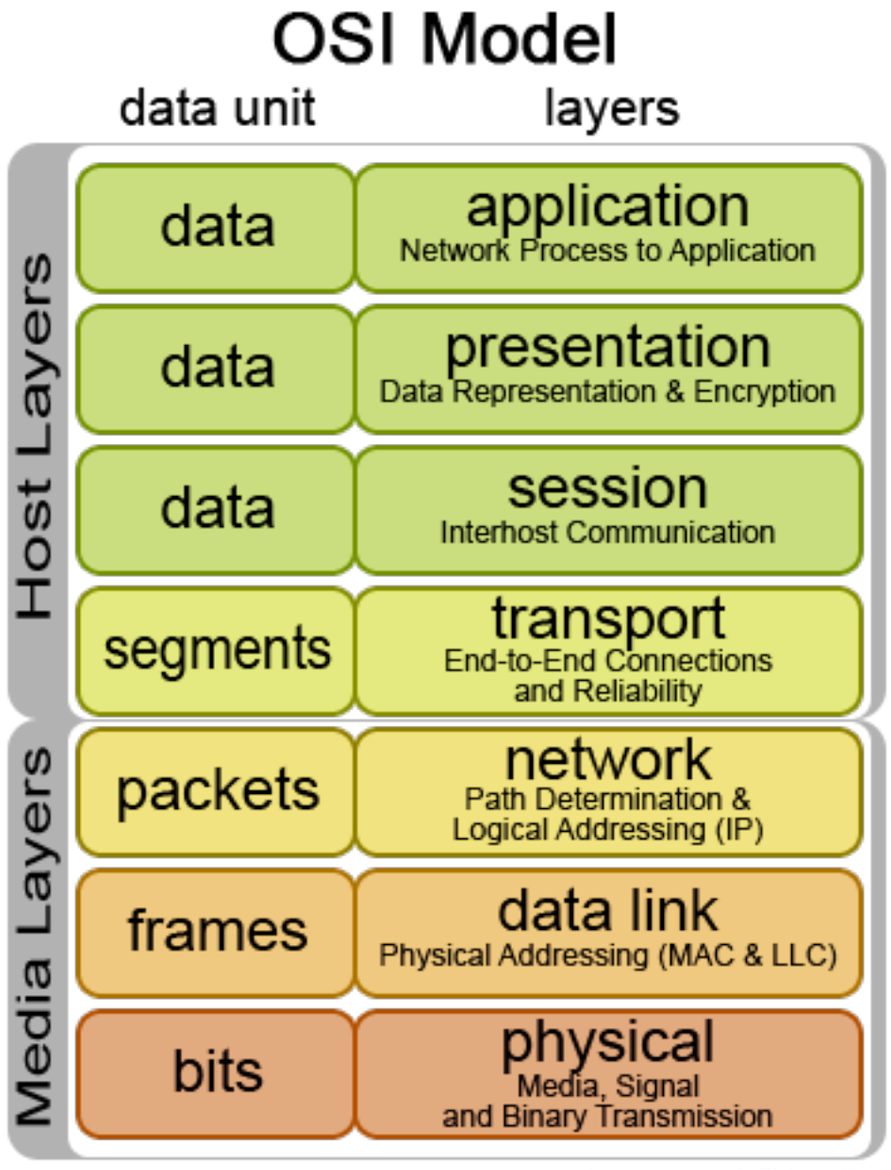


Image src: Wikipedia

HTTP uses **reliable** data-transmission protocols.

Network communication

textbook model; IN2400 Computer Networks

- Conceptual model Open Systems Interconnection (OSI)
- Network protocols are matched to different “layers”
- Many network protocols exist, three are of interest to us:

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

others include: SSH, IMAP, STMP, FTP ….

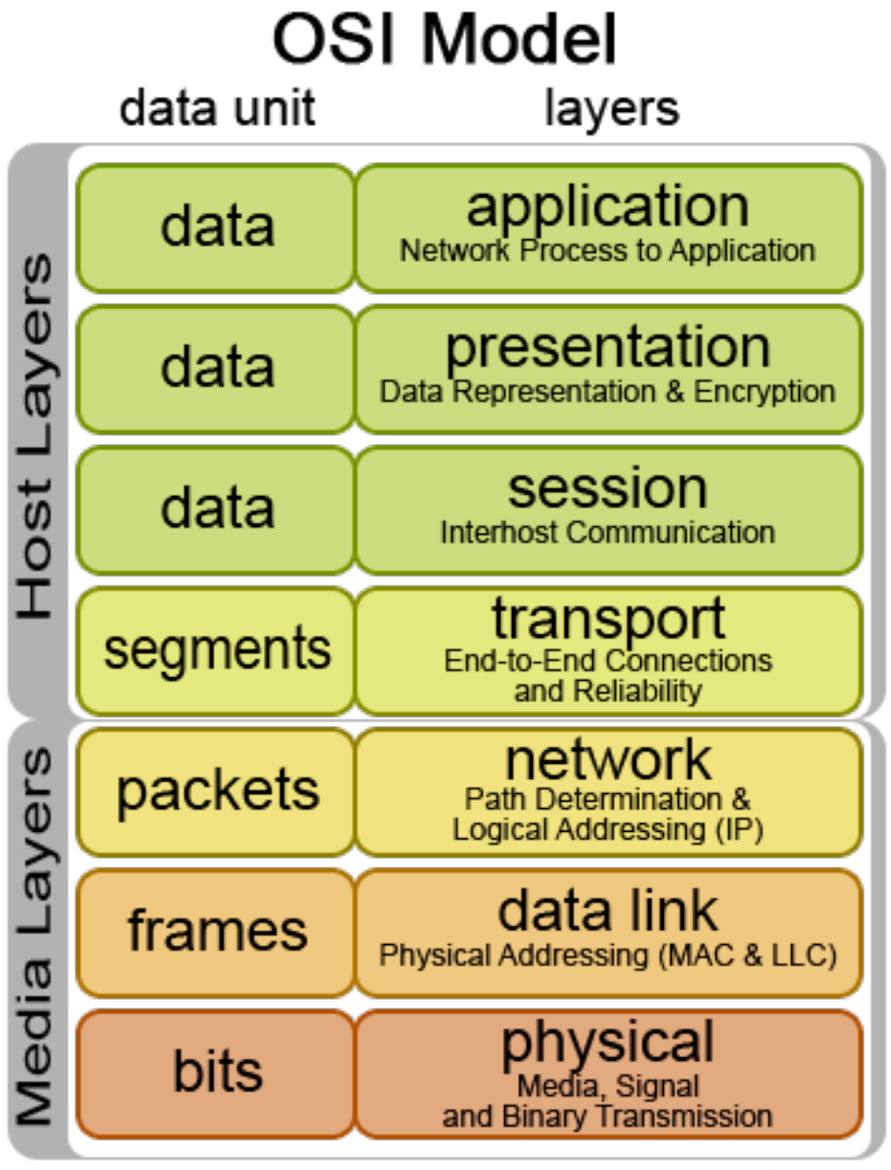
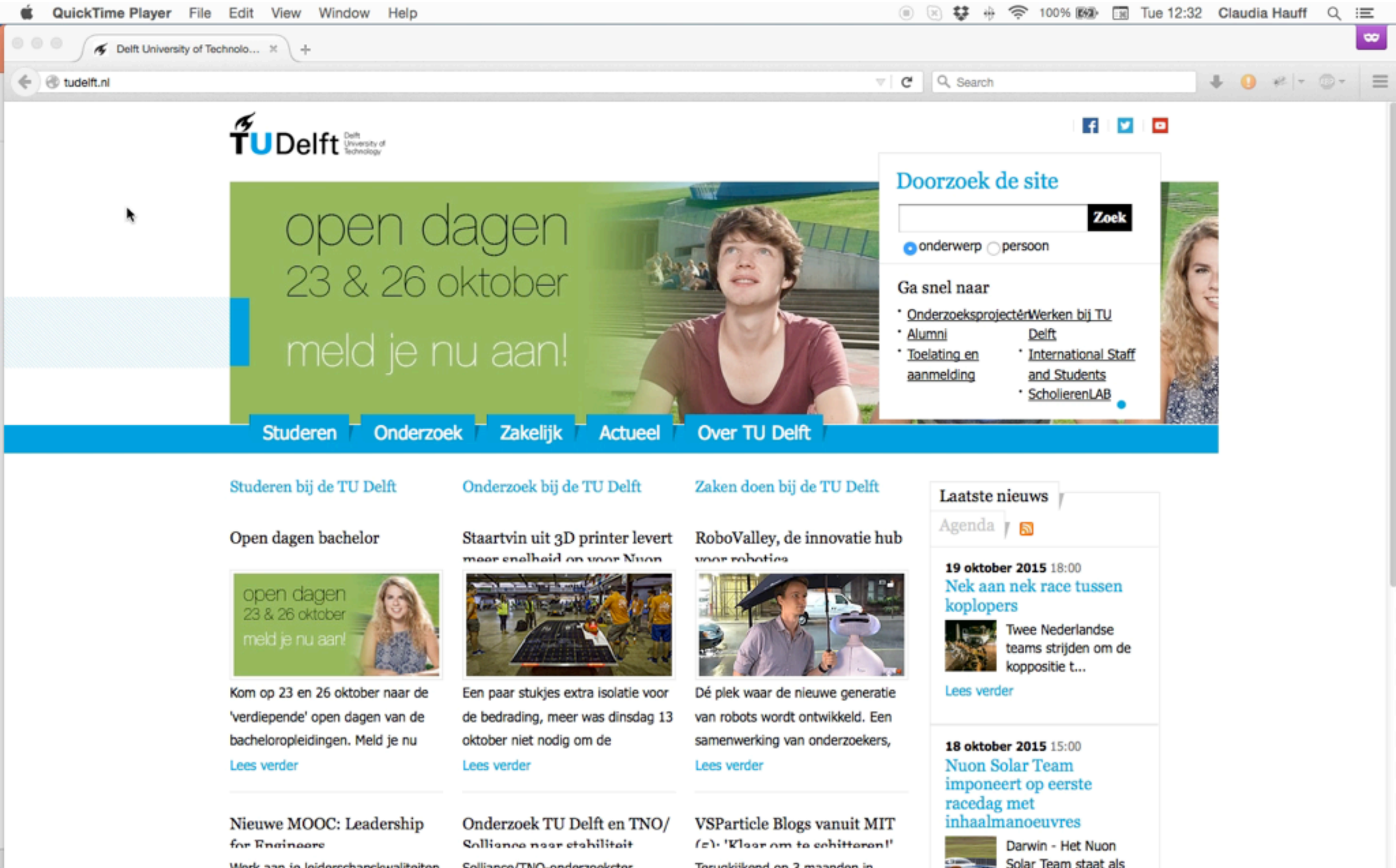


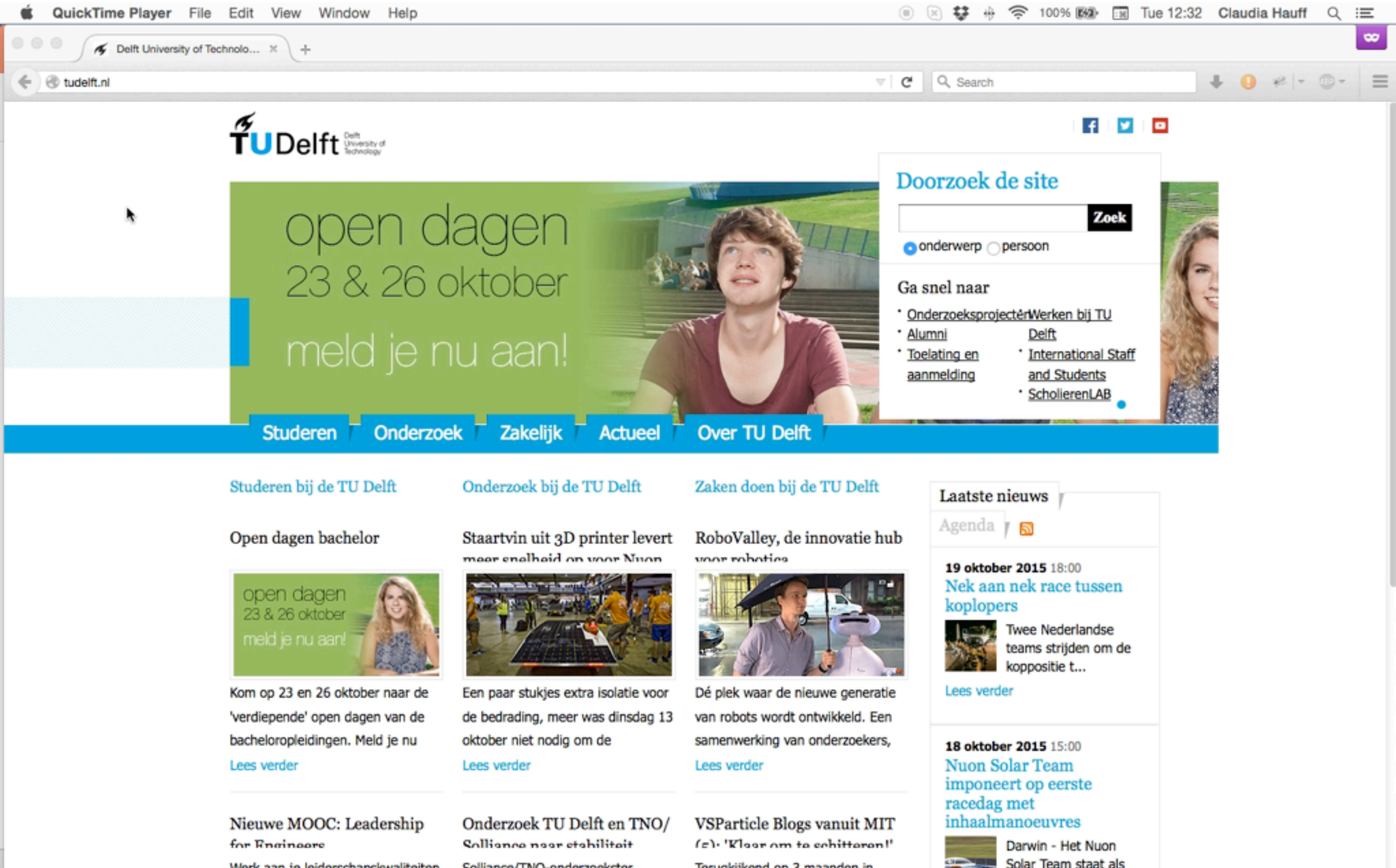
Image src: Wikipedia

HTTP uses **reliable** data-transmission protocols.

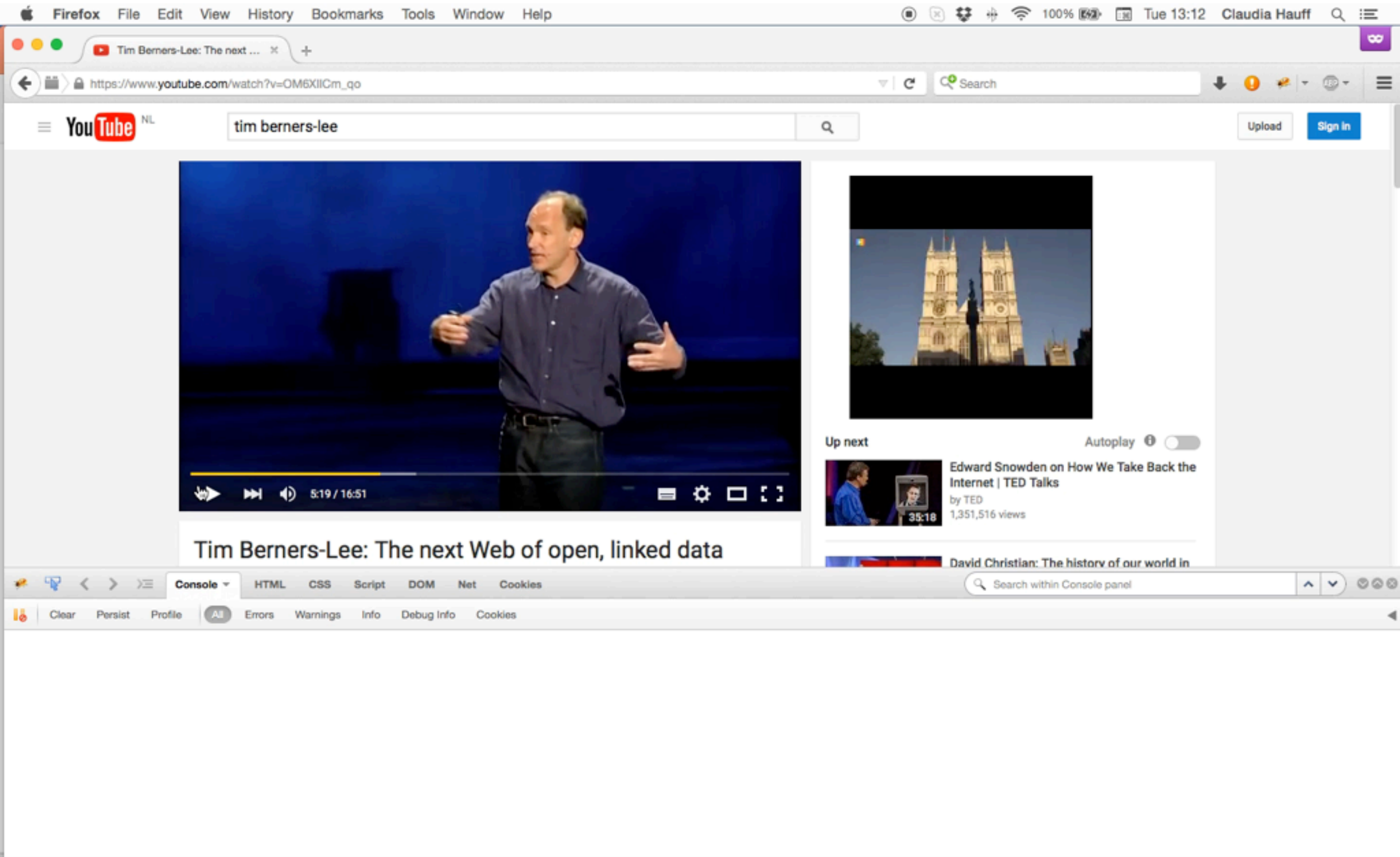
Demo: a look at tudelft.nl



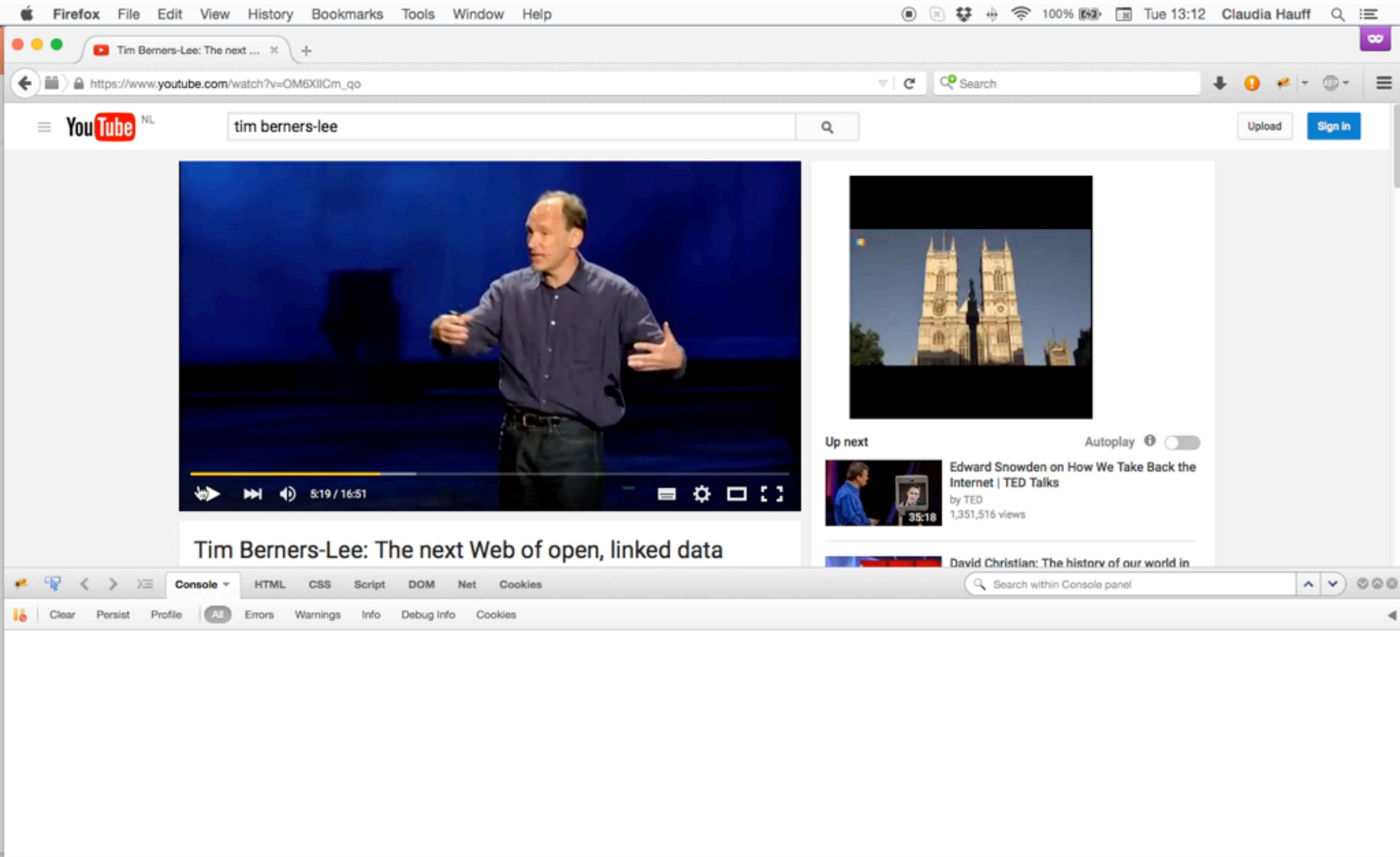
Demo: a look at tudelft.nl



Demo: watching a video



Demo: watching a video



Demos: main points

Modern web sites consist of many resources.

A cascade of HTTP transactions is required.

Different MIME types are typically found.

Firebug: one (popular) tool of many existing tools for Web development.

HTTP request message

```
GET / HTTP/1.1
Host: www.tudelft.nl
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X
10.9; rv:31.0) Gecko/20100101 Firefox/31.0
Accept: text/html,application/xhtml+xml,application/
xml;q=0.9,*/*;q=0.8
Accept-Language: en-gb,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Cookie:
__utma=1.209235779361l1.16111.19805.2;utmcmd=(none);
```

HTTP request message

```
GET / HTTP/1.1
Host: www.tudelft.nl
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X
10.9; rv:31.0) Gecko/20100101 Firefox/31.0
Accept: text/html,application/xhtml+xml,application/
xml;q=0.9,*/*;q=0.8
Accept-Language: en-gb,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Cookie:
__utma=1.209235779361l1.16111.19805.2;utmcmd=(none);
```

HTTP request message

plain text, line-oriented character sequences

```
GET / HTTP/1.1
Host: www.tudelft.nl
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X
10.9; rv:31.0) Gecko/20100101 Firefox/31.0
Accept: text/html,application/xhtml+xml,application/
xml;q=0.9,*/*;q=0.8
Accept-Language: en-gb,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Cookie:
__utma=1.209235779361l1.16111.19805.2;utmcmd=(none);
```

HTTP response message

```
HTTP/1.1 200 OK
Date: Fri, 01 Aug 2014 13:35:55 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 5994
Connection: keep-alive
Set-Cookie: fe_typo_user=d5e20a55a4a92e0;
path=/; domain=tudelft.nl
[....]
Server: TU Delft Web Server

....
....
```

HTTP response message

```
HTTP/1.1 200 OK
Date: Fri, 01 Aug 2014 13:35:55 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 5994
Connection: keep-alive
Set-Cookie: fe_typo_user=d5e20a55a4a92e0;
path=/; domain=tudelft.nl
[....]
Server: TU Delft Web Server

....
....
```

HTTP response message

`HTTP/1.1 200 OK`	**start line**

`Date: Fri, 01 Aug 2014 13:35:55 GMT`
`Content-Type: text/html; charset=utf-8`
`Content-Length: 5994`
`Connection: keep-alive`
`Set-Cookie: fe_typo_user=d5e20a55a4a92e0; path=/; domain=tudelft.nl`
`[....]`
`Server: TU Delft Web Server`
	header fields
	`name:value`

`....` `....`	**body (optional)**

HTTP headers dissected

Primary entity header fields

Content-Type	Entity type
Content-Length	Length/size of the message
Content-Language	Language of the entity sent (e.g. English)
Content-Encoding	Data transformations applied to the entity
Content-Location	Alternative location of the entity
Content-Range	For partial entities, range defines the pieces sent
Content-MD5	Checksum of the content
Last-Modified	Date on which this entity was created/modified
Expires	Date at which the entity will become stale
Allow	Lists the legal request methods for the entity

Primary entity header fields

Content-Type	Entity type
Content-Length	Length/size of the message
Content-Language	Language of the entity sent (e.g. English)
Content-Encoding	Data transformations applied to the entity
Content-Location	Alternative location of the entity
Content-Range	For partial entities, range defines the pieces sent
Content-MD5	Checksum of the content
Last-Modified	Date on which this entity was created/modified
Expires	Date at which the entity will become stale
Allow	Lists the legal request methods for the entity

Important: Entity bodies only contain **raw** data, **header** information required to **interpret** the data.

Content-Type

- **MIME** types are attached to all HTTP object data

  Multipurpose Internet **Mail** Extensions

  historic reasons

- MIME type determines clients reaction to the received data

- Pattern: `[primary object type]/[subtype]`

  e.g. `text/plain`, `text/html`, `image/jpeg`, `video/quicktime`, `application/vnd.ms-powerpoint`

Content types are diverse

Most popular
text/html
image/jpg
text/xml
application/rss+xml
text/plain
application/xml
text/calendar
application/pdf
application/atom+xml
unknown/unknown

list based on a sample of CommonCrawl 2014

Content types are diverse

Most popular
text/html
image/jpg
text/xml
application/rss+xml
text/plain
application/xml
text/calendar
application/pdf
application/atom+xml
unknown/unknown

Least popular
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

list based on a sample of CommonCrawl 2014

Content-Length

- Indicates the **size** of the entity body in the message

- Necessary to detect premature message truncation (e.g. due to a server crash, faulty proxy)

- Essential for **persistent connections** to discover where one HTTP message ends and the next one begins

Content-Length

- Indicates the **size** of the entity body in the message

- Necessary to detect premature message truncation (e.g. due to a server crash, faulty proxy)

- Essential for **persistent connections** to discover where one HTTP message ends and the next one begins

Persistent connections reuse the same TCP connection for multiple HTTP request/response messages.

Content-Encoding

Content-Encoding

- Commonly either gzip, compress (Unix compression), deflate (zlib compression) or identity (no encoding)

Content-Encoding

- Commonly either gzip, compress (Unix compression), deflate (zlib compression) or identity (no encoding)

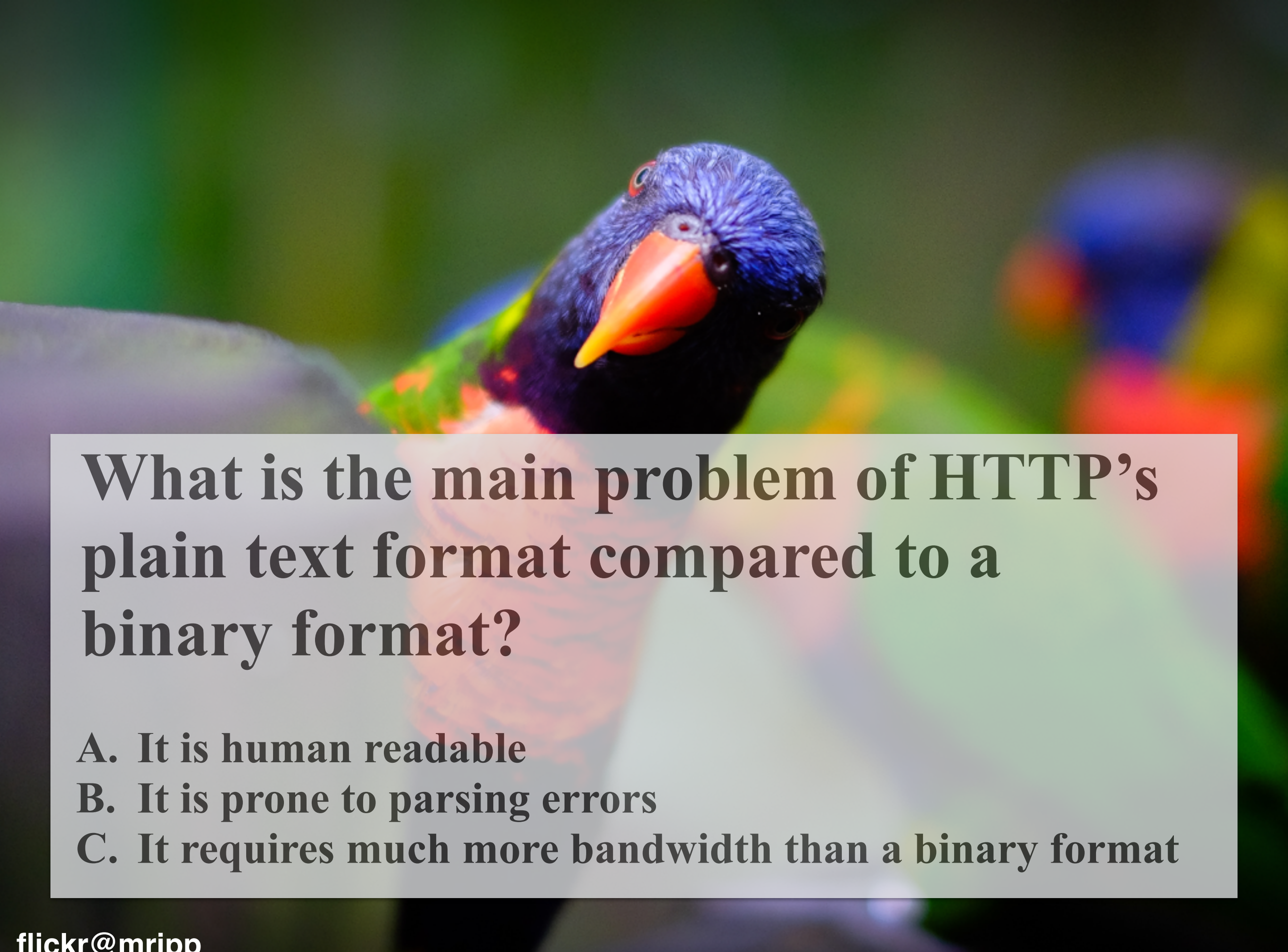
- Servers aim to use **encodings** that clients understand
- Clients send a list of acceptable encodings in the **`Accept-Encoding`** request header

```
Accept-Encoding: gzip, deflate
```

What are the main advantage and disadvantage of using compression?

A. Advantage: lower processing costs,
   Disadvantage: decreased security
B. Advantage: less bandwidth required,
   Disadvantage: higher processing costs
C. Advantage: increased security,
   Disadvantage: higher throughput
D. Advantage: increased security,
   Disadvantage: increased processing costs

flickr@132033298@N04



What is the main problem of HTTP's plain text format compared to a binary format?

A. It is human readable
B. It is prone to parsing errors
C. It requires much more bandwidth than a binary format

flickr@mripp

What is commonly compressed?

A. HTTP header only
B. HTTP body only
C. HTTP header and body

flickr@philippzieger

Content-Encoding

- Commonly either gzip, compress (Unix compression), deflate (zlib compression) or identity (no encoding)

- Servers aim to use **encodings** that clients understand
  - Clients send a list of acceptable encodings in the **`Accept-Encoding`** request header

```
Accept-Encoding: gzip, deflate
```

- **Compression** saves network bandwidth (fewer bits to transmit) but increases processing costs to decompress

Content-MD5

- HTTP messages are sent via TCP/IP (ensuring reliable transport)

- **But**: the Internet is huge, many servers interact to transport a message with different implementations of established protocols (which may be **buggy**)

- **Sanity check**: Sender generates a **MD5 checksum** of the content (hashed into a 128 bit value) to detect unintended modifications of the content

Content-MD5

- HTTP messages are sent via TCP/IP (ensuring reliable transport)

- **But**: the Internet is huge, many servers interact to transport a message with different implementations of established protocols (which may be **buggy**)

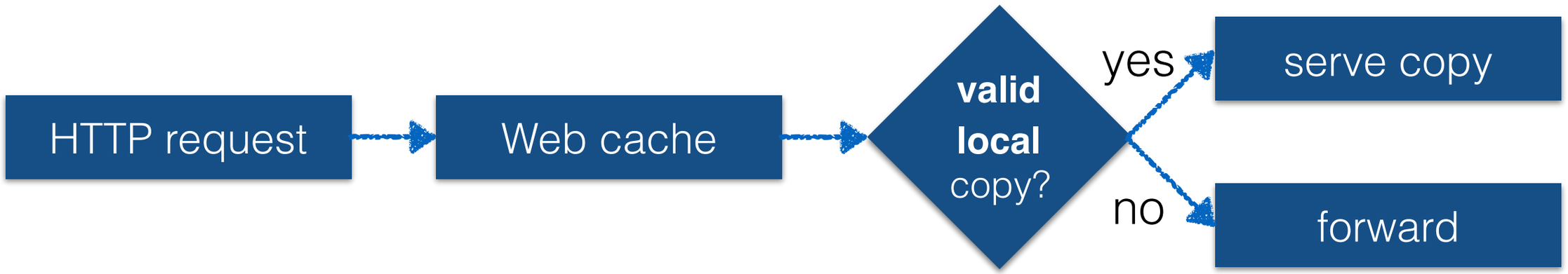
- **Sanity check**: Sender generates a **MD5 checksum** of the content (hashed into a 128 bit value) to detect unintended modifications of the content

MD5 is cryptographically broken

Expires

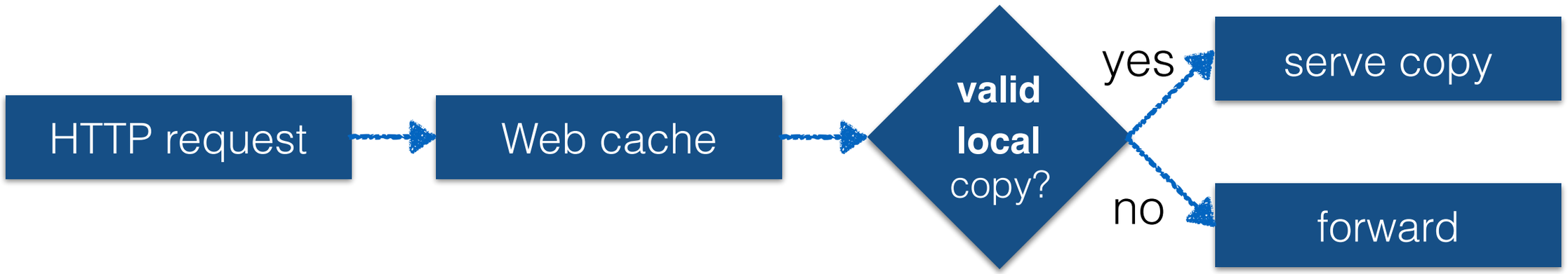
Expires

- **Web caches** keep copies of *popular* documents



Expires

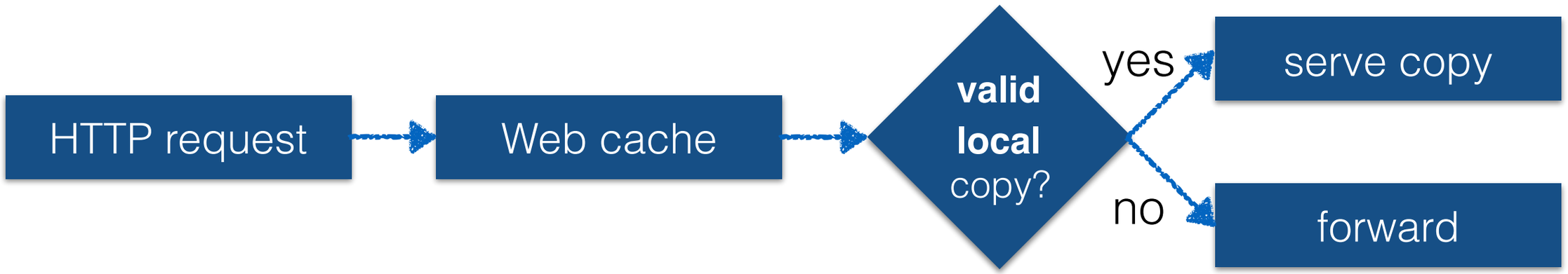
- **Web caches** keep copies of *popular* documents



- Advantages of Web caches
  A. **Reduction** of redundant data transfer
  B. **Reduction** of network bottlenecks
  C. **Reduction** demand on origin servers
  D. **Reduced** distance delay

Expires

- **Web caches** keep copies of *popular* documents

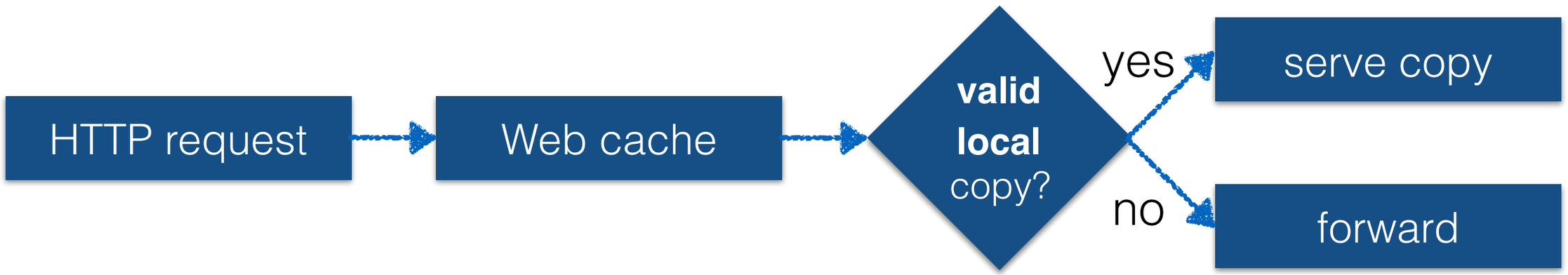


- Advantages of Web caches
  A. **Reduction** of redundant data transfer
  B. **Reduction** of network bottlenecks
  C. **Reduction** demand on origin servers
  D. **Reduced** distance delay
- `Expires` indicates when fetched resource is no longer valid and needs to be retrieved from the origin server

Note: Web browsers also cache when possible.

Expires

- **Web caches** keep copies of *popular* documents



- Advantages of Web caches
  A. **Reduction** of redundant data transfer
  B. **Reduction** of network bottlenecks
  C. **Reduction** demand on origin servers
  D. **Reduced** distance delay
- `Expires` indicates when fetched resource is no longer valid and needs to be retrieved from the origin server

Expires & Cache-Control

Expires & Cache-Control

- Content on the origin server can change
- Caches need to ensure that their copies are **in sync** with the origin server

Expires & Cache-Control

- Content on the origin server can change
- Caches need to ensure that their copies are **in sync** with the origin server

- Caches can revalidate their copies at any time (**in**efficient)

Expires & Cache-Control

- Content on the origin server can change
- Caches need to ensure that their copies are **in sync** with the origin server

- Caches can revalidate their copies at any time (**in**efficient)

- `Expires` in HTTP response header indicates a **document's expiration date** in **absolute** terms — date determines when the cache revalidates
- `Cache-Control`: indicates a document's expiration date in **relative** terms (number of seconds since being sent)

Last-Modified

Last-Modified

- Contains the date when the document was last **altered** (in HTTP response)
- No indication about the amount of changes in the document

Last-Modified

- Contains the date when the document was last **altered** (in HTTP response)
- No indication about the amount of changes in the document

- Often used in combination with `If-Modified-Since` for cache revalidation requests — origin server only returns the document if it changed since the given date

Last-Modified

- Contains the date when the document was last **altered** (in HTTP response)
- No indication about the amount of changes in the document

- Often used in combination with `If-Modified-Since` for cache revalidation requests — origin server only returns the document if it changed since the given date

- Last-Modified dates are **not reliable**

Remember: HTTP response message

```
HTTP/1.1 200 OK
Date: Fri, 01 Aug 2014 13:35:55 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 5994
Connection: keep-alive
Set-Cookie: fe_typo_user=d5e20a55a4a92e0;
path=/; domain=tudelft.nl
....
Server: TU Delft Web Server

....
```

start line

header fields

`name:value`

body (optional)

Common status codes

1xx	Informational
2xx	Success (200 OK)
3xx	Redirected
4xx	Client error (404 Not Found)
5xx	Server error

In practice only a few codes per category are supported

A more detailed overview.

HTTP methods

```
GET / HTTP/1.1
Host: www.tudelft.nl
User-Agent: Mozilla/5.0 (Macinto
10.9; rv:31.0) Gecko/20100101 Fi
Accept: text/html,application/xh
xml;q=0.9,*/*;q=0.8
Accept-Language: en-gb,en;q=0.5
Accept-Encoding: gzip, deflate
```

HTTP methods

```
GET / HTTP/1.1
Host: www.tudelft.nl
User-Agent: Mozilla/5.0 (Macinto
10.9; rv:31.0) Gecko/20100101 Fi
Accept: text/html,application/xh
xml;q=0.9,*/*;q=0.8
Accept-Language: en-gb,en;q=0.5
Accept-Encoding: gzip, deflate
```

Common HTTP methods

GET	Get a document from the Web server.
HEAD	Get the header of a document from the Web server.
POST	Send data from the client to the server for processing.
PUT	Save the body of the request on the server.
TRACE	Trace the message through proxy servers to the server.
OPTIONS	Determine what methods can operate on a server.
DELETE	Remove a document from a Web server.

Common HTTP methods

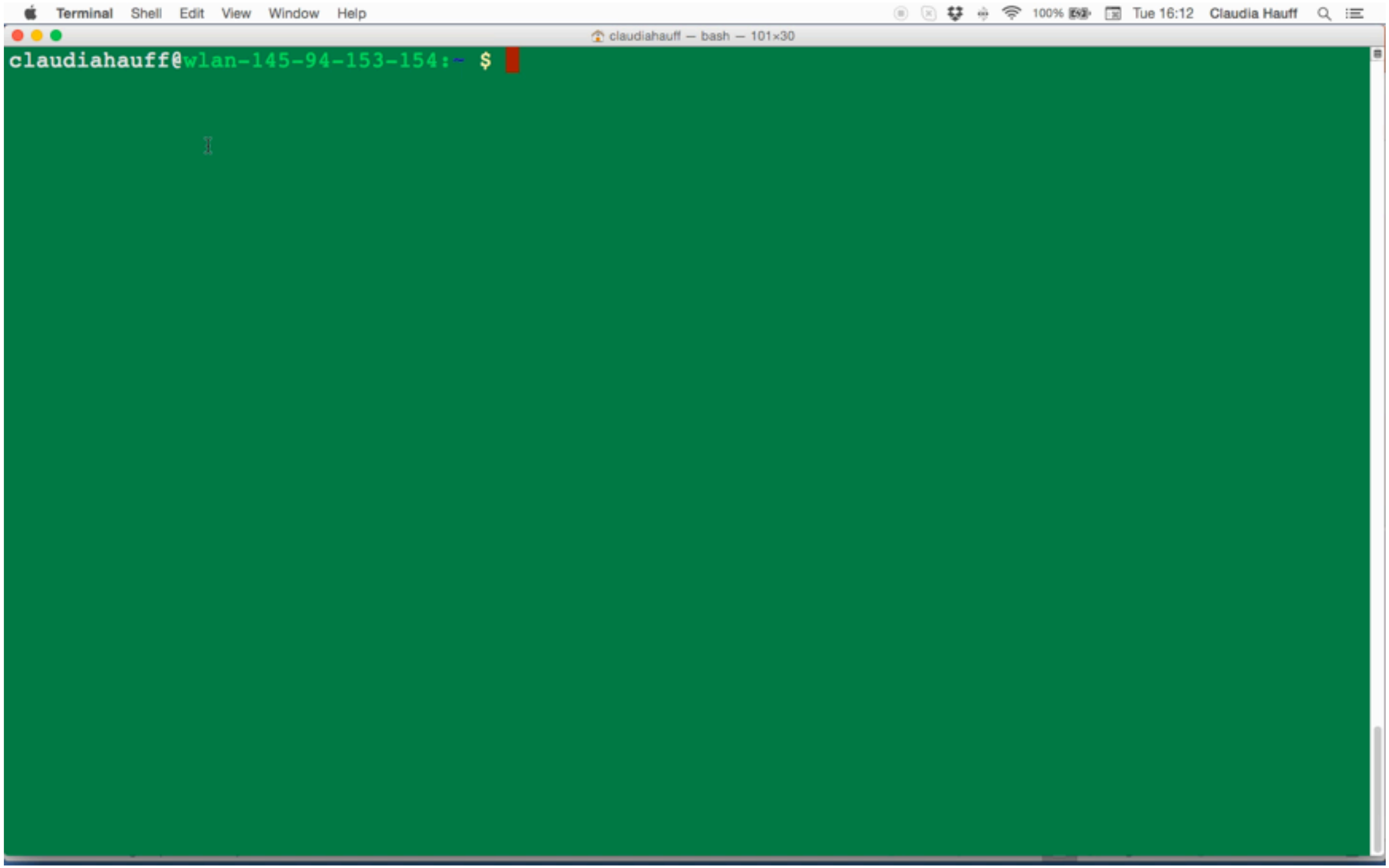
GET	Get a document from the Web server.
HEAD	Get the header of a document from the Web server.
POST	Send data from the client to the server for processing.
PUT	Save the body of the request on the server.
TRACE	Trace the message through proxy servers to the server.
OPTIONS	Determine what methods can operate on a server.
DELETE	Remove a document from a Web server.

Servers may implement more or fewer methods than shown.

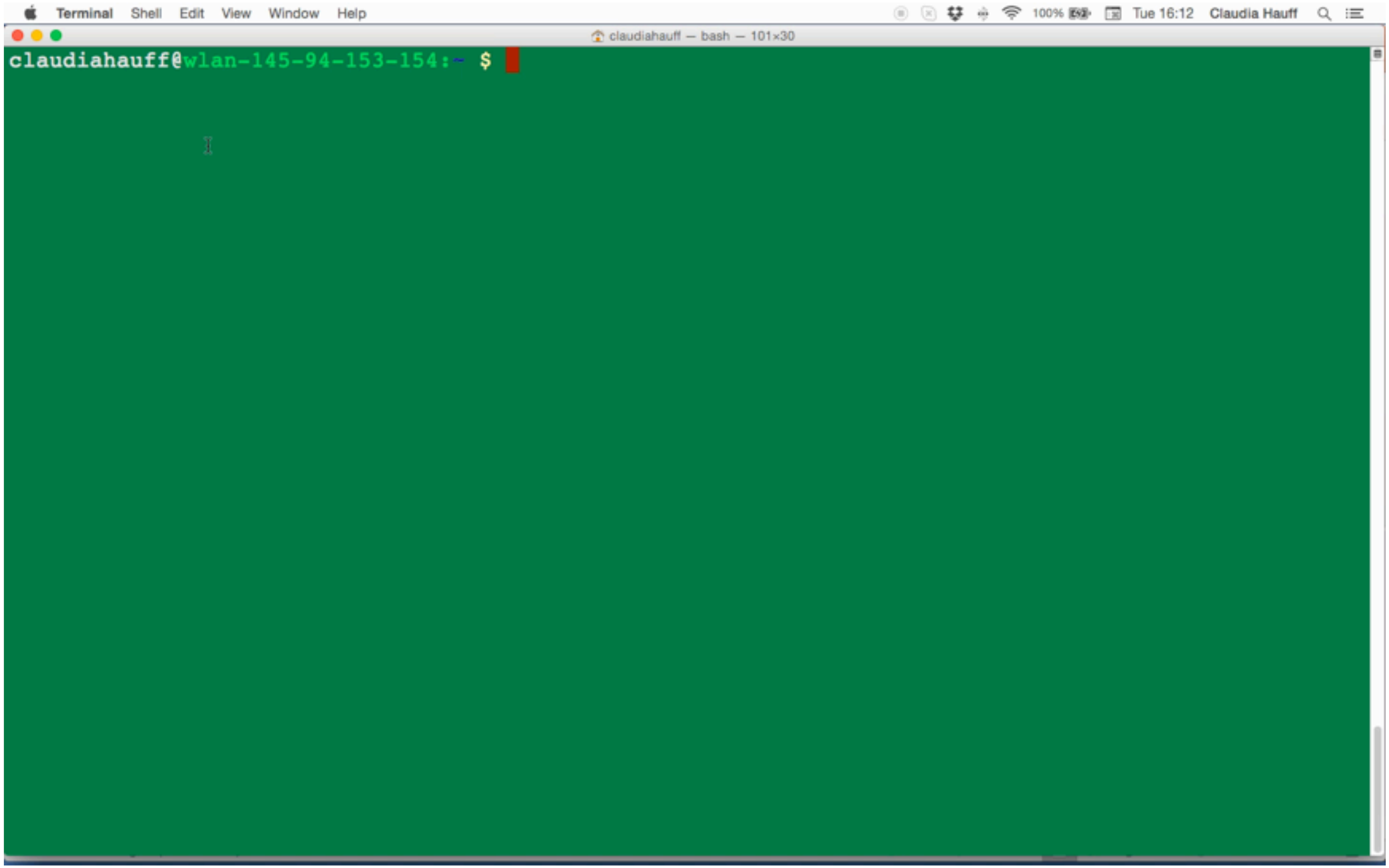
Demo: telnet

Telnet opens a **TCP connection** to a Web server; chars are typed directly into the port. The server treats telnet as Web client, the returned data is displayed onscreen.

Demo: telnet


Terminal Shell Edit View Window Help
100% Tue 16:12 Claudia Hauff
claudiahauff — bash — 101×30
claudiahauff@wlan-145-94-153-154:~ $

Demo: telnet


Terminal Shell Edit View Window Help
100% Tue 16:12 Claudia Hauff
claudiahauff — bash — 101×30
claudiahauff@wlan-145-94-153-154:~ $

Demo: telnet

Telnet opens a **TCP connection** to a Web server; chars are typed directly into the port. The server treats telnet as Web client, the returned data is displayed onscreen.

A number of HTTP requests may be required to end up at the final (wanted) page.

Often Web servers treat Web-browser requests differently from machine-generated requests.

From domain to IP address

```
claudiahauff@Claudias-MacBook-Air:~ $ telnet microsoft.com 80
Trying 134.170.185.46...
Connected to microsoft.com.
Escape character is '^]'.
```

From domain to IP address

```
claudiahauff@Claudias-MacBook-Air:~ $ telnet microsoft.com 80
Trying 134.170.185.46...
Connected to microsoft.com.
Escape character is '^]'.
```

domain

From domain to IP address

```
claudiahauff@Claudias-MacBook-Air:~ $ telnet microsoft.com 80
Trying 134.170.185.46...
Connected to microsoft.com.
Escape character is '^]'.
```

IP address (v4): 32 bit

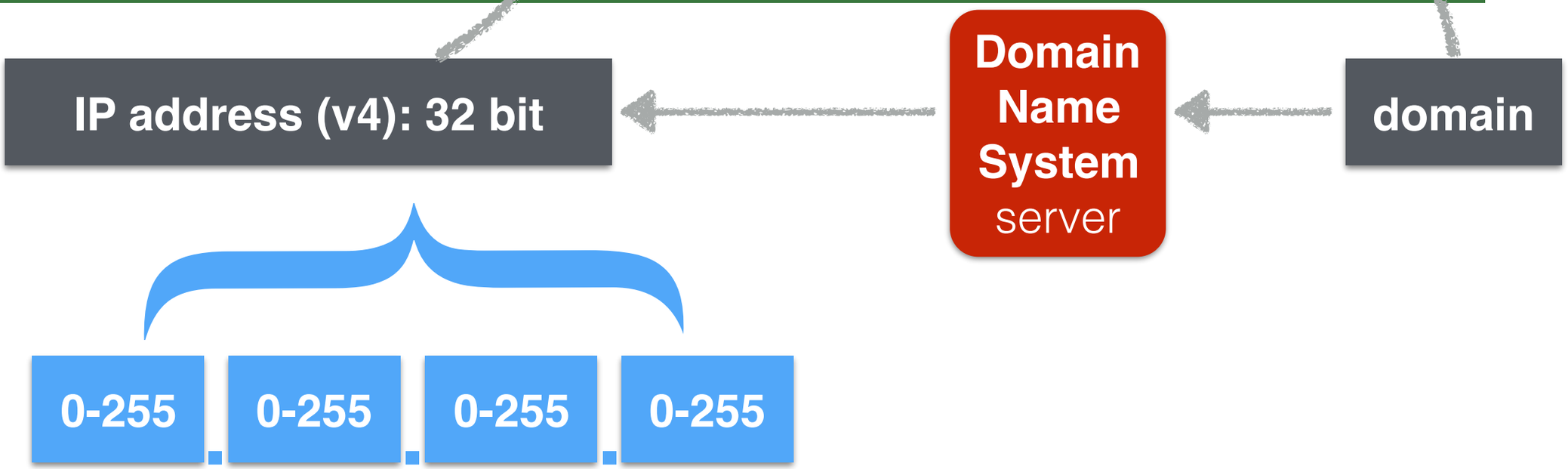
domain

From domain to IP address

```
claudiahauff@Claudias-MacBook-Air:~ $ telnet microsoft.com 80
Trying 134.170.185.46...
Connected to microsoft.com.
Escape character is '^]'.
```

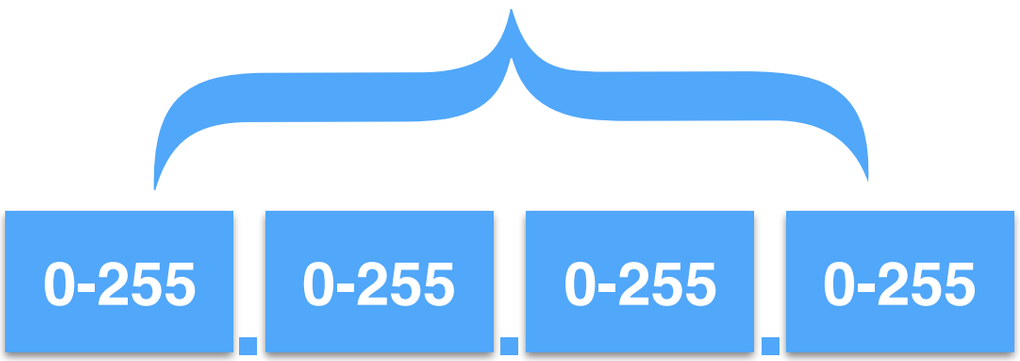
From domain to IP address

```
claudiahauff@Claudias-MacBook-Air:~ $ telnet microsoft.com 80
Trying 134.170.185.46...
Connected to microsoft.com.
Escape character is '^]'.
```



From domain to IP address

```
claudiahauff@Claudias-MacBook-Air:~ $ telnet microsoft.com 80
Trying 134.170.185.46...
Connected to microsoft.com.
Escape character is '^]'.
```



4x8 bits; **4,294,967,296 addresses**

From domain to IP address

```
claudiahauff@Claudias-MacBook-Air:~ $ telnet microsoft.com 80
Trying 134.170.185.46...
Connected to microsoft.com.
Escape character is '^]'.
```

IP address (v4): 32 bit

Domain Name System server

domain

0-255 . **0-255** . **0-255** . **0-255**

4x8 bits; **4,294,967,296 addresses**

IP address (v6): 128 bit

From domain to IP address

```
claudiahauff@Claudias-MacBook-Air:~ $ telnet microsoft.com 80
Trying 134.170.185.46...
Connected to microsoft.com.
Escape character is '^]'.
```

IP address (v4): 32 bit ← **Domain Name System** server ← **domain**

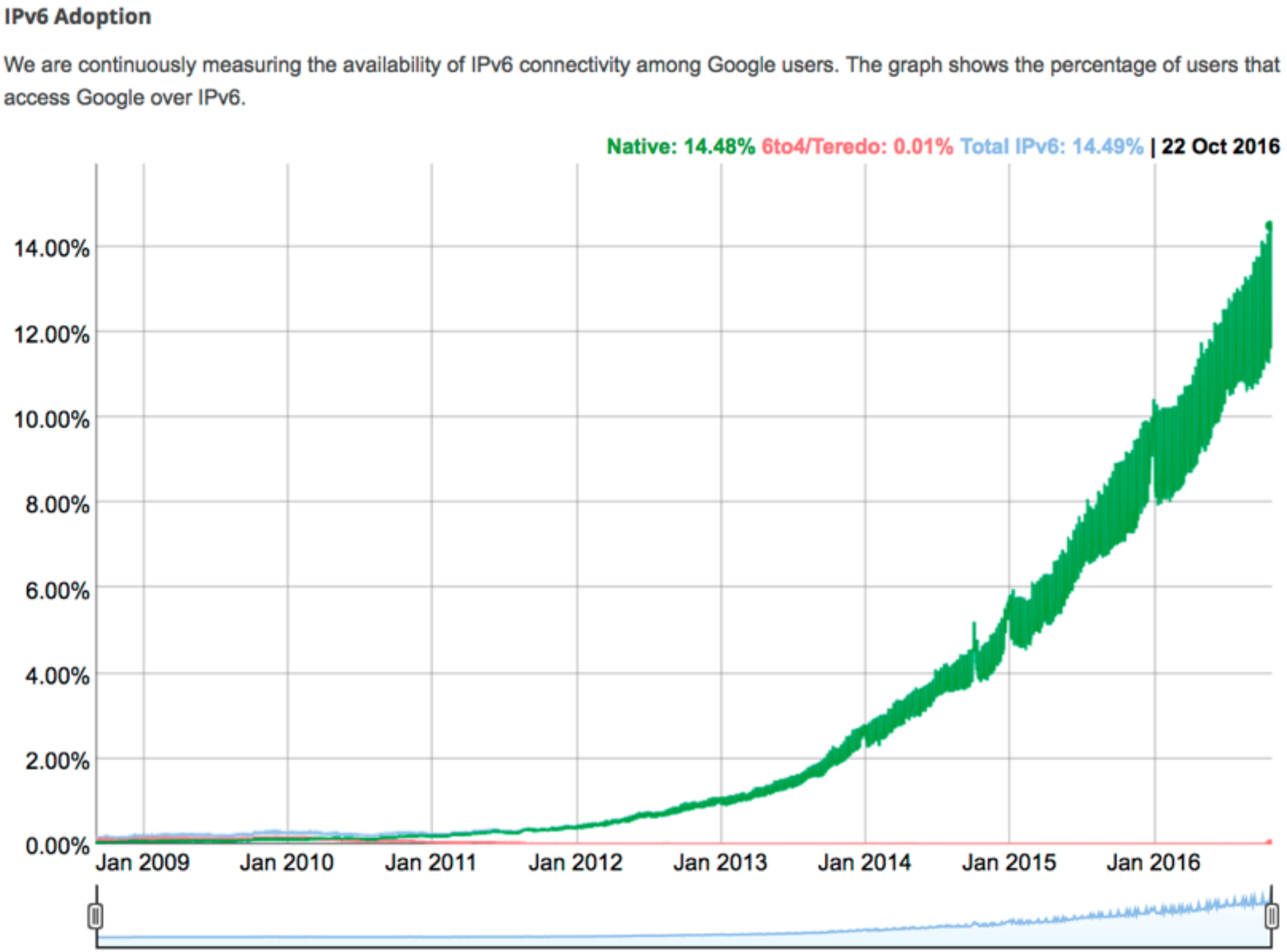
0-255 . **0-255** . **0-255** . **0-255**

4x8 bits; **4,294,967,296 addresses**

IP address (v6): 128 bit

2A41 : **0000** : **3341** : **3AAA** : **0000** : **FFFF** : **86AA** : **B92E**

From domain to IP address



Source: http://www.google.com/intl/en/ipv6/statistics.html

In what circumstance might `HEAD` be useful?

A. To determine when a resource has been last modified
B. To determine the number of images included in an HTML page
C. To determine the download speed of the resource
D. To decide how much of the resource to download

```
HTTP/1.1 200 OK
Date: Fri, 01 Aug 2014 13:35:55 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 5994
Connection: keep-alive
Set-Cookie: fe_typo_user=d5e20a55a4a92e0;
path=/; domain=tudelft.nl
[....]
Server: TU Delft Web Server
```

flickr@ana_cotta

In what order do the following operations occur when you enter http://www.microsoft.com in the browser's address bar?

A. A → D → B → C
B. D → C → A → B
C. A → D → C → B
D. D → A → C → B

A Convert domain to IP address	**C** Send HTTP request
B Receive HTTP response	**D** Establish a TCP connection

flickr@gruesome

How many HTTP requests does a browser make to render this (simplified) version of the Google homepage?

A. 1
B. 3
C. 5
D. 7

Google Search I'm Feeling Lucky

flickr@akahodag

Uniform Resource Locators (URLs)

How many of the following URLs are valid?

A `mailto:c.hauff@tudelft.nl`

B `ftp://anonymous:mypass@ftp.csx.cam.ac.uk/gnu;date=today` (ignore the line break)

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

D `https://duckduckgo.com/html?q=delft`

E `http://myshop.nl/comp;typ=c/apple;class=a;date=today/index.html;fr=delft` (ignore the line break)

F `http://правительство.рф`

URL syntax

- Uniform resource locators offer a standardised way to point to any resource on the Internet

- Not restricted to the `http` scheme, syntax slightly varies from scheme to scheme

- General format (adhered to by most schemes):

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

URL syntax

- Uniform resource locators offer a standardised way to point to any resource on the Internet

- Not restricted to the `http` scheme, syntax slightly varies from scheme to scheme

- General format (adhered to by most schemes):

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

determines the protocol to use when connecting to the server.

URL syntax

- Uniform resource locators offer a standardised way to point to any resource on the Internet

- Not restricted to the `http` scheme, syntax slightly varies from scheme to scheme

- General format (adhered to by most schemes):

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

the username/password (may be necessary to access a resource)

URL syntax

- Uniform resource locators offer a standardised way to point to any resource on the Internet

- Not restricted to the `http` scheme, syntax slightly varies from scheme to scheme

- General format (adhered to by most schemes):

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

domain name (host name) or numeric IP address of the server

URL syntax

- Uniform resource locators offer a standardised way to point to any resource on the Internet

- Not restricted to the `http` scheme, syntax slightly varies from scheme to scheme

- General format (adhered to by most schemes):

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

the port on which the server is expecting requests for the resource

URL syntax

- Uniform resource locators offer a standardised way to point to any resource on the Internet

- Not restricted to the `http` scheme, syntax slightly varies from scheme to scheme

- General format (adhered to by most schemes):

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

the local path to the resource

URL syntax

- Uniform resource locators offer a standardised way to point to any resource on the Internet
- Not restricted to the `http` scheme, syntax slightly varies from scheme to scheme
- General format (adhered to by most schemes):

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

Additional input parameters applications may require to access a resource on the server correctly. Can be set per path segment.

URL syntax

- Uniform resource locators offer a standardised way to point to any resource on the Internet

- Not restricted to the `http` scheme, syntax slightly varies from scheme to scheme

- General format (adhered to by most schemes):

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

Parameters passed to gateway resources, i.e. applications [identified by the path] such as search engines.

URL syntax

- Uniform resource locators offer a standardised way to point to any resource on the Internet

- Not restricted to the `http` scheme, syntax slightly varies from scheme to scheme

- General format (adhered to by most schemes):

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

The name of a piece of a resource. Only used by the client - the fragment is not transmitted to the server.

URL syntax: query
(most important to us)

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

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

- Query component is passed to the application accessed at the Web server (“gateway resource”)
- Necessary to enable interactive applications (like our ToDo Web application!)
- Common convention: **`name1=value1&name2=value2&…`**

Schemes: more than just http

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

Schemes: more than just http

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

`https` is analogous to **`http`** with **`https`** as scheme name (end-to-end encryption of HTTP connections)

Schemes: more than just http

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

`https` is analogous to **`http`** with **`https`** as scheme name (end-to-end encryption of HTTP connections)

`mailto:<valid-email-address>`

Schemes: more than just http

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

`https` is analogous to **`http`** with **`https`** as scheme name (end-to-end encryption of HTTP connections)

`mailto:<valid-email-address>`

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

Schemes: more than just http

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

`https` is analogous to **`http`** with **`https`** as scheme name (end-to-end encryption of HTTP connections)

`mailto:<valid-email-address>`

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

`file://<host>/<path>`, e.g.

`file:///Users/my_home_dir/tmp.html`

Relative vs. absolute URLs

absolute

```
http://www.st.ewi.tudelft.nl/~hauff/new/index.html
```

relative

```
<h1>Visualizations</h1>
<ol>
   <li><a href="vis/trecvis.html">TREC</a></li>
   <li><a href=" ../airsvis.html">AIRS</a></li>
</ol>
```

Relative vs. absolute URLs

base url

absolute

`http://www.st.ewi.tudelft.nl/~hauff/new/index.html`

relative

```
<h1>Visualizations</h1>
<ol>
   <li><a href="vis/trecvis.html">TREC</a></li>
   <li><a href=" ../airsvis.html">AIRS</a></li>
</ol>
```

Relative vs. absolute URLs

base url

absolute

`http://www.st.ewi.tudelft.nl/~hauff/new/index.html`

relative

```
<h1>Visualizations</h1>
<ol>
   <li><a href="vis/trecvis.html">TREC</a></li>
   <li><a href=" ../airsvis.html">AIRS</a></li>
</ol>
```

converted via **base url** to

`http://www.st.ewi.tudelft.nl/~hauff/new/vis/trecvis.html`
`http://www.st.ewi.tudelft.nl/~hauff/airsvis.html`

Relative vs. absolute URLs

base url

absolute

`http://www.st.ewi.tudelft.nl/~hauff/new/`index.html

relative

```
<h1>Visualizations</h1>
<ol>
   <li><a href="vis/trecvis.html">TREC</a></li>
   <li><a href=" ../airsvis.html">AIRS</a></li>
</ol>
```

Not trivial: URLs can be complex, RFC 3986 governs conversion rules.

http://www.st.ewi.tudelft.nl/~hauff/new/vis/trecvis.html

http://www.st.ewi.tudelft.nl/~hauff/airsvis.html

URL design restrictions

URL design restrictions

- **Initial design goals**: *portable* across protocols and human *readable* (no invisible/non-printing chars.)

URL design restrictions

- **Initial design goals**: *portable* across protocols and human *readable* (no invisible/non-printing chars.)
- URLs initially restricted to a very small “safe” alphabet: ASCII (American Standard Code for Information Interchange)

> latin alphabet, `0123456789 - _ .~`
> and additional reserved chars like `! ( ) @ &`

URL design restrictions

- **Initial design goals**: *portable* across protocols and human *readable* (no invisible/non-printing chars.)
- URLs initially restricted to a very small "safe" alphabet: ASCII

Heavily biased in favor of English speakers

latin alphabet, `0123456789 - _ .~`
and additional reserved chars like `! ( ) @ &`

URL design restrictions

- **Initial design goals**: *portable* across protocols and human *readable* (no invisible/non-printing chars.)
- URLs initially restricted to a very small "safe" alphabet: ASCII **Heavily biased in favor of English speakers**

  latin alphabet, `0123456789 - _ .~`
  and additional reserved chars like `! ( ) @ &`

- **Added later**: character encoding, e.g. whitespace as `%20`

URL design restrictions

- **Initial design goals**: *portable* across protocols and human *readable* (no invisible/non-printing chars.)
- URLs initially restricted to a very small “safe” alphabet: ASCII **Heavily biased in favor of English speakers**

  latin alphabet, `0123456789 - _ .~`
  and additional reserved chars like `! ( ) @ &`

- **Added later**: character encoding, e.g. whitespace as `%20`

`http://правительство.рф`

Punycode (RFC3492)

"Punycode is a simple and efficient transfer encoding syntax designed for use with Internationalized Domain Names in Applications. It **uniquely** and **reversibly** transforms a Unicode string into an ASCII string."

Punycode (RFC3492)

"Punycode is a simple and efficient transfer encoding syntax designed for use with Internationalized Domain Names in Applications. It **uniquely** and **reversibly** transforms a Unicode string into an ASCII string."

```
http://правительство.рф
      => http//:xn--80aealotwbjpid2k.xn--p1ai
```

Punycode (RFC3492)

"Punycode is a simple and efficient transfer encoding syntax designed for use with Internationalized Domain Names in Applications. It **uniquely** and **reversibly** transforms a Unicode string into an ASCII string."

```
http://правительство.рф
      => http//:xn--80aealotwbjpid2k.xn--p1ai
http://nl.wikipedia.org/wiki/Itali%C3%AB
      => http://nl.wikipedia.org/wiki/Itali%C3%AB
```

Punycode (RFC3492)

"Punycode is a simple and efficient transfer encoding syntax designed for use with Internationalized Domain Names in Applications. It **uniquely** and **reversibly** transforms a Unicode string into an ASCII string."

```
http://правительство.рф
      => http//:xn--80aealotwbjpid2k.xn--p1ai
http://nl.wikipedia.org/wiki/Itali%C3%AB
      => http://nl.wikipedia.org/wiki/Itali%C3%AB
```

A potential security issue in *mixed scripts*: http://paypal.com

Punycode (RFC3492)

"Punycode is a simple and efficient transfer encoding syntax designed for use with Internationalized Domain Names in Applications. It **uniquely** and **reversibly** transforms a Unicode string into an ASCII string."

```
http://правительство.рф
      => http//:xn--80aealotwbjpid2k.xn--p1ai
http://nl.wikipedia.org/wiki/Itali%C3%AB
      => http://nl.wikipedia.org/wiki/Itali%C3%AB
```

A potential security issue in *mixed scripts*: http://paypal.com

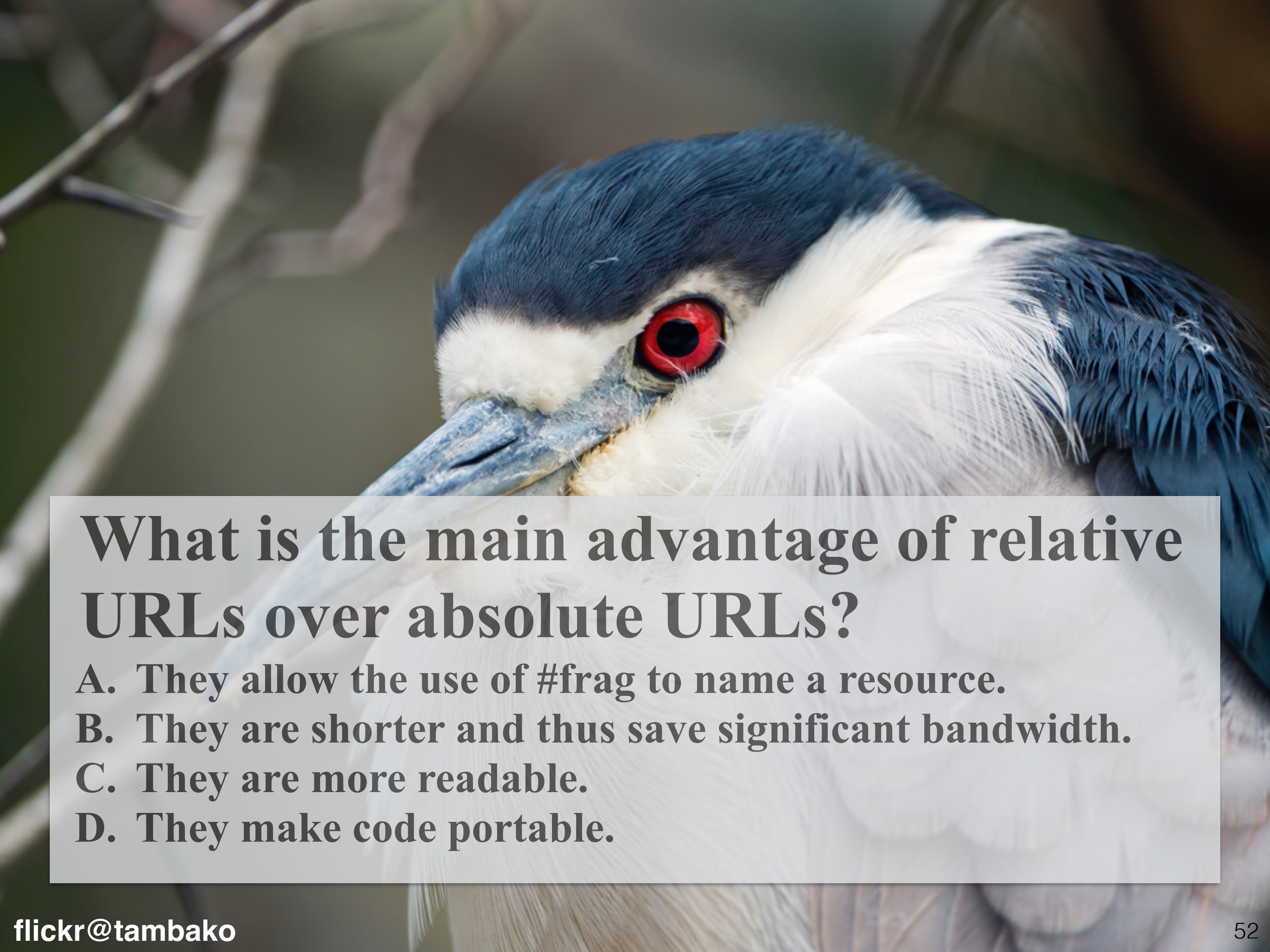
Punycode (RFC3492)

"Punycode is a simple and efficient transfer encoding syntax designed for use with Internationalized Domain Names in Applications. It **uniquely** and **reversibly** transforms a Unicode string into an ASCII string."

```
http://правительство.рф
      => http//:xn--80aealotwbjpid2k.xn--p1ai
http://nl.wikipedia.org/wiki/Itali%C3%AB
      => http://nl.wikipedia.org/wiki/Itali%C3%AB
```

A potential security issue in *mixed scripts*: http://paypal.com

IDNA demo: http://demo.icu-project.org/icu-bin/idnbrowser



What is the main advantage of relative URLs over absolute URLs?

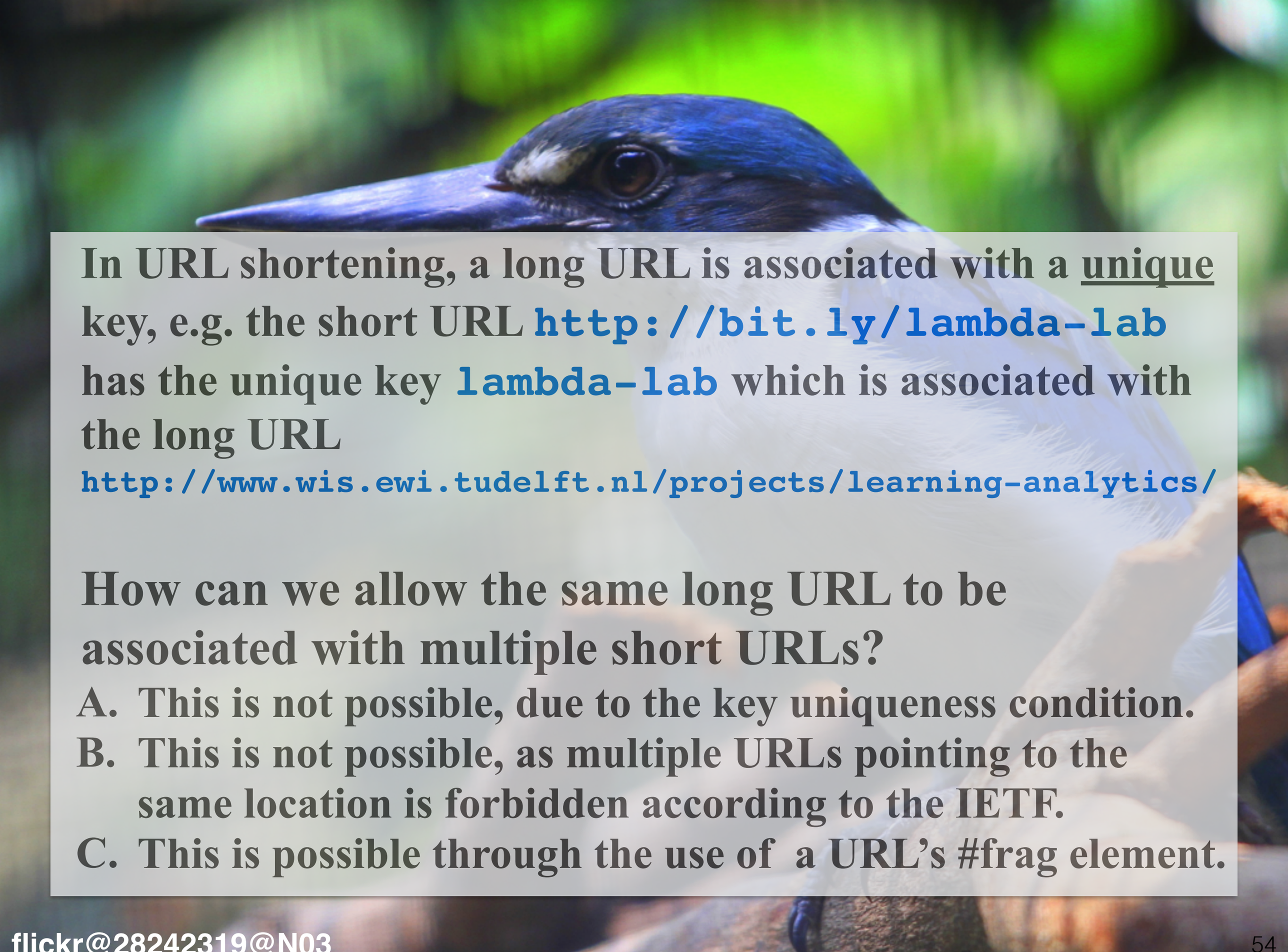
A. They allow the use of #frag to name a resource.
B. They are shorter and thus save significant bandwidth.
C. They are more readable.
D. They make code portable.

flickr@tambako

What is the main weakness of URLs as they are in use today?

A. URLs are unnecessarily long.
B. We are running out of URL space for ASCII-based URLs.
C. URLs point to a location instead of a Web resource.
D. URLs point to a Web resource instead of a location.

flickr@28242319@N03



In URL shortening, a long URL is associated with a unique key, e.g. the short URL `http://bit.ly/lambda-lab` has the unique key `lambda-lab` which is associated with the long URL

`http://www.wis.ewi.tudelft.nl/projects/learning-analytics/`

How can we allow the same long URL to be associated with multiple short URLs?

A. This is not possible, due to the key uniqueness condition.
B. This is not possible, as multiple URLs pointing to the same location is forbidden according to the IETF.
C. This is possible through the use of a URL's #frag element.

flickr@28242319@N03

You should now know…

- How Web servers and clients interact with each other
- How to write HTTP requests
- How to interpret HTTP responses
- How to interpret and create valid URLs

Recommended reading:
HTTP: The Definite Guide (O'REILLY, 2002)
Chapters 1, 2, 3 & 12

(the lecture material is largely based on them)

- Find a **lab partner** as soon as possible! Register via **Blackboard**! Know your team's **cluster**! If you are a TI student adhere to the **BDF/ACE** division!

- Work through **Chapter 2** (intro to HTML) of the Web development book **before** this Friday's lecture!

- Start working on this week's **quiz** (by yourself) & lab **assignment** (as a team)!

flickr@courtneyrian