Web Sites
and
Web Apps

An introduction to the world of
Web-site and Web-App
development

AARHUS
UNIVERSITY
SCHOOL OF ENGINEERING

Agenda

- The Basic Web architecture
  - The Origins of the WWW
  - HTML
  - URL
  - HTTP
  - The Web Server
  - The Browser
- Web enhancements
  - Dynamic html generation
  - CSS
  - Java applets
  - Java script
  - Ajax
- Web Dev Tools

The Origins of the WWW

- WWW was invented by Tim Berners-Lee (a physicist) at CERN in 1989-1992
- Main purpose:

Hypertext across the Internet

(replacing FTP)

- Five constituents:

HTML	Mark-up language for hypertext
URL	Notation for locating files on serves
HTTP/HTTPS	High-level protocol for file transfers
Web server	Sends a file as a http response when requested
Browser	Receives HTML documents and render them as visible pages

The Origin of HTML

- HTML is an acronym for Hyper Text Mark-up Language
- HTML 1.0 was a simplification of SGML (Standard Generalized Markup Language) with the addition of the Link element

Year	Version
1991	*HTML Tags*, an informal CERN document was first mentioned in public.
1992	HTML 1.0. First informal draft of the HTML standard. *Tim Berners-Lee proposal.*
1995	HTML 2.0 was published as IETF RFC 1866
1996	The HTML standard is now developed by W3C
1997	HTML 3.2 was published as a W3C Recommendation. *The Browser War ends.*
1997/98	HTML 4.0. Style sheets are introduced - CSS.
2000-02	XHTML 1.0 published as a W3C Recommendation. An XML version of HTML 4.01.
2008	HTML5 was published as a Working Draft by the W3C.
2014	HTML5 was published as a W3C Recommendation.
2016	HTML5.1 was published as a W3C Recommendation.
2017	HTML5.2 was published as a W3C Recommendation.

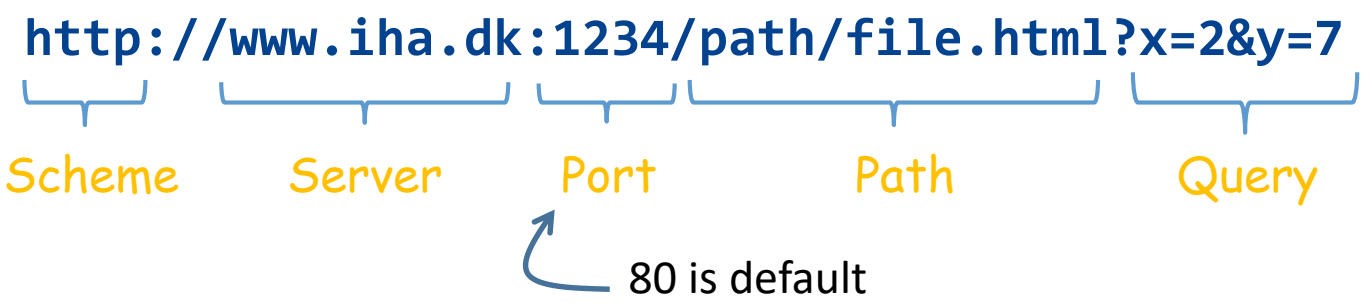
HTML

- HTML describes the *logical structure* of a document
- HTML uses tags *<tag>* to structure the text

```
<html>
  <head>
    <title>I4GUI</title>
  </head>
  <body>
    <h1>GUI programmering</h1>
    <h2>I4GUI</h2>
    <p>Til Web-applikationer anvendes <b>HTML</b>,
       CSS og Javascript.</p>
  </body>
</html>
```

Uniform Resource Locator - URL

- A Web resource is located by a URL:



- A relative URL:

```
path/file.htm
```

- Fragment identifier

URI

- URLs are a subset of the more general concept of Uniform Resource Identifiers (URIs)
- The general URI syntax is very flexible:

  scheme:scheme-specific-part

- Many different schemes are defined besides http:

  ftp, file, mailto, imap, https, dict, geo …

- The official register of schemes is maintained by the Internet Assigned Numbers Authority (IANA):

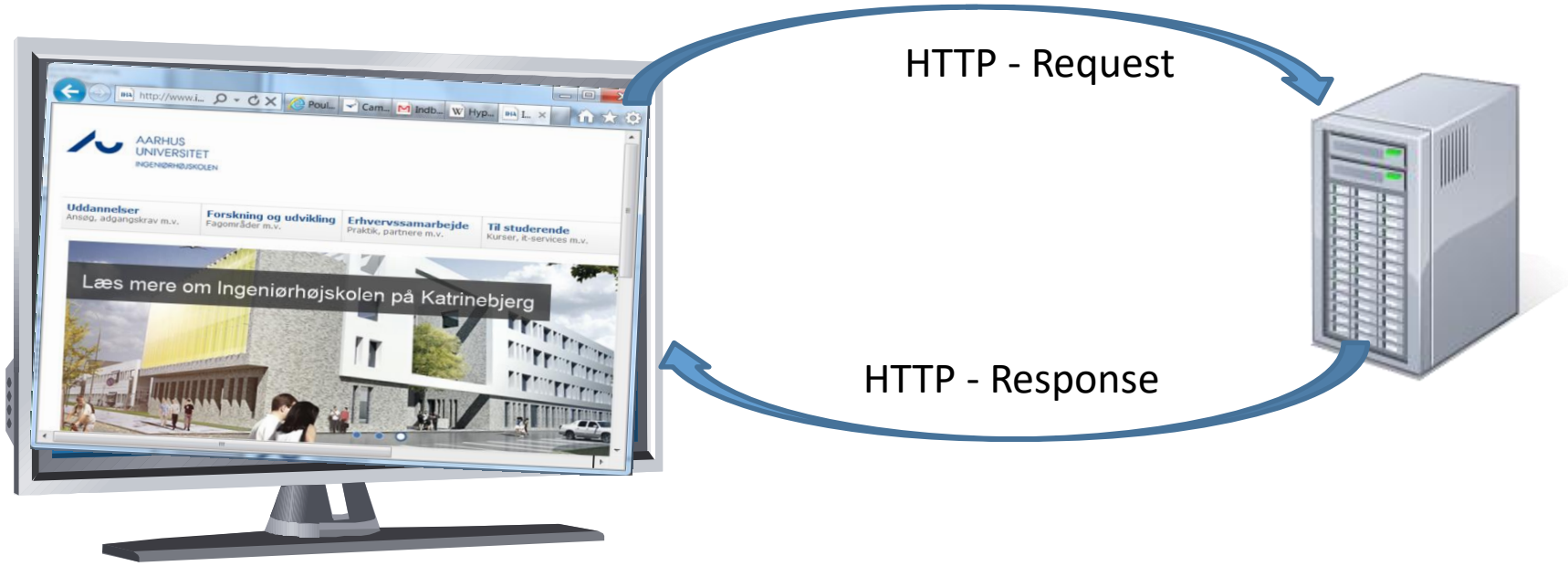
  http://www.iana.org/assignments/uri-schemes.html

URL Rules

- All URLs follow certain rules:
  - '/' implies a hierarchical structure
  - '?' separates the queryable resource from the query string
  - '#' separates a fragment identifier from the URI
  - Special symbols are escaped with the notation '%NN' (NN is the characters hexadecimal code), e.g. %20 is ' ' (*space*)

HTTP HyperText Transfer Protocol

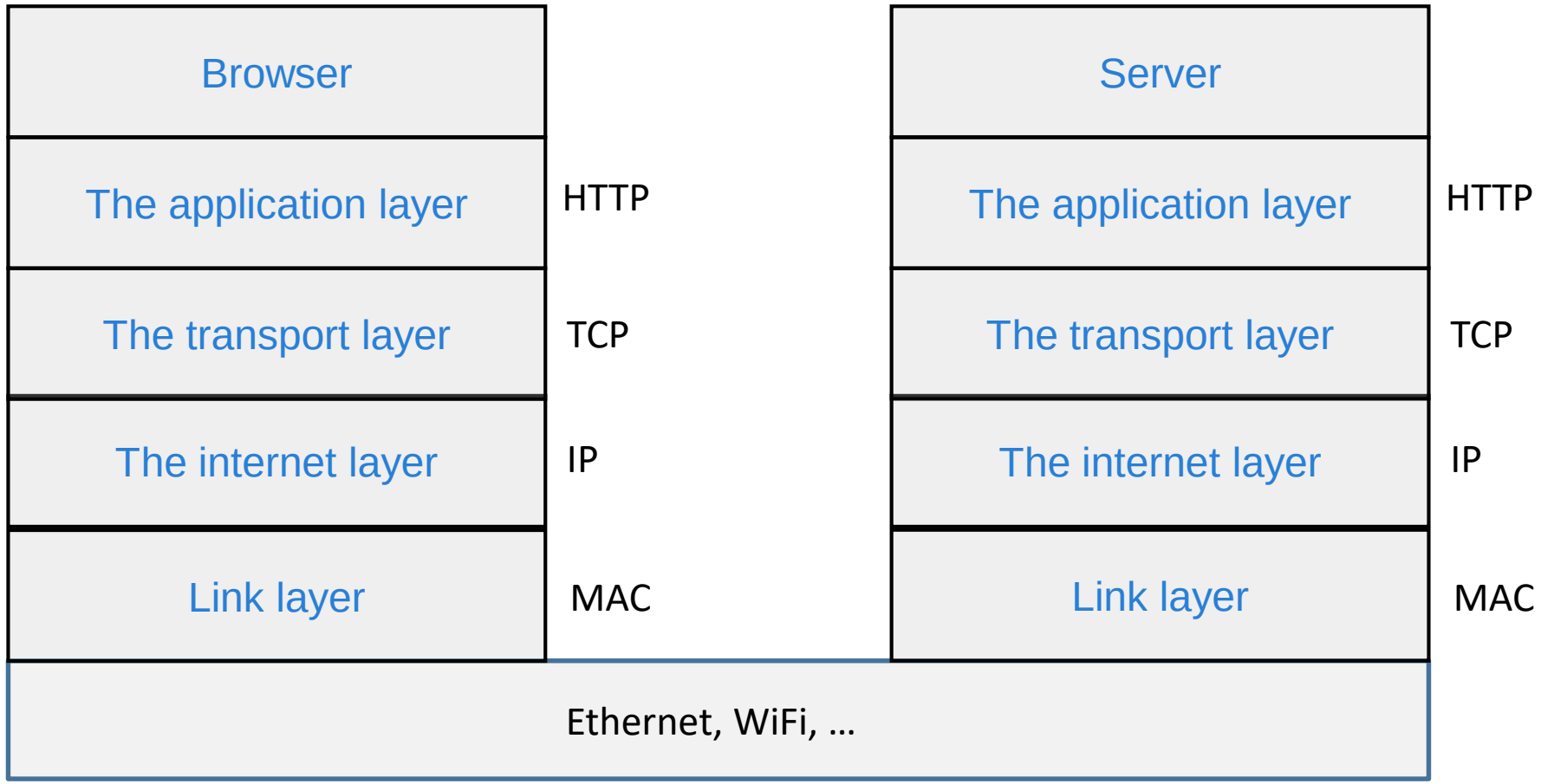
- Client-Server model following a **Request-Response** pattern
- Previus version v1.1 (RFC 2068) from 1997
  - Updated in 1999 (RFC 2616)
- Current version is HTTP/2 (RFC 7540) from May 2015



HTTP/2

- The HTTP/2 specification was published as RFC 7540 in May 2015
  - The specification was developed by the Hypertext Transfer Protocol Bis (httpbis) working group of the IETF
- **Goals for HTTP 2.0 include:**
  - **Asynchronous connection multiplexing**
    - One of the bottlenecks of HTTP v1.1 implementations is that HTTP relies on multiple connections for concurrency
  - **Header compression**
    - Reduces overhead
  - **Server push technologies**
    - This allows the server to supply data it knows a web browser will need
- Result:
  - **Page load speedup ranging from 11.81% to 47.7%**
- Is backwards compatibility with the semantics of HTTP 1.1
  - The element that is modified is how the data is framed and transported between the client and the server

Network Layers



HTTP Verbs

- The client submits an HTTP **request** message to the server
- The server returns a **response** message to the client
- The response contains completion status information about the request and may also contain requested content (e.g. a html document) in its message body

- HTTP is a simple protocol with only few Request methods (verbs):
  - GET *fetch* an existing ressource
  - POST *create* a new resource
  - PUT *update* an existing resource
  - DELETE *delete* an existing resource
  - *And a few others*

HTTP Status Codes

- The client initiate a requests to the server with URLs and verbs
- In return, the server responds with status codes and message payloads
- Status Codes (extract):
  - 1xx: Informational Messages
  - 2xx: Successful
    - **200 OK**
  - 3xx: Redirection
    - 301 Moved Permanently
  - 4xx: Client Error
    - 400 Bad Request
    - 401 Unauthorized
    - 403 Forbidden
    - 404 Not found
  - 5xx: Server Error
    - 503 Service Unavailable

Web Client

- Connected to the Internet when needed
- Usually a web browser
  (*such as Chrome, Edge or Safari)*
- Uses HTTP (Hypertext Transfer Protocol)
- Requests web pages, files or data from server
- Receives web pages and files from server
- Renders the received html on the screen

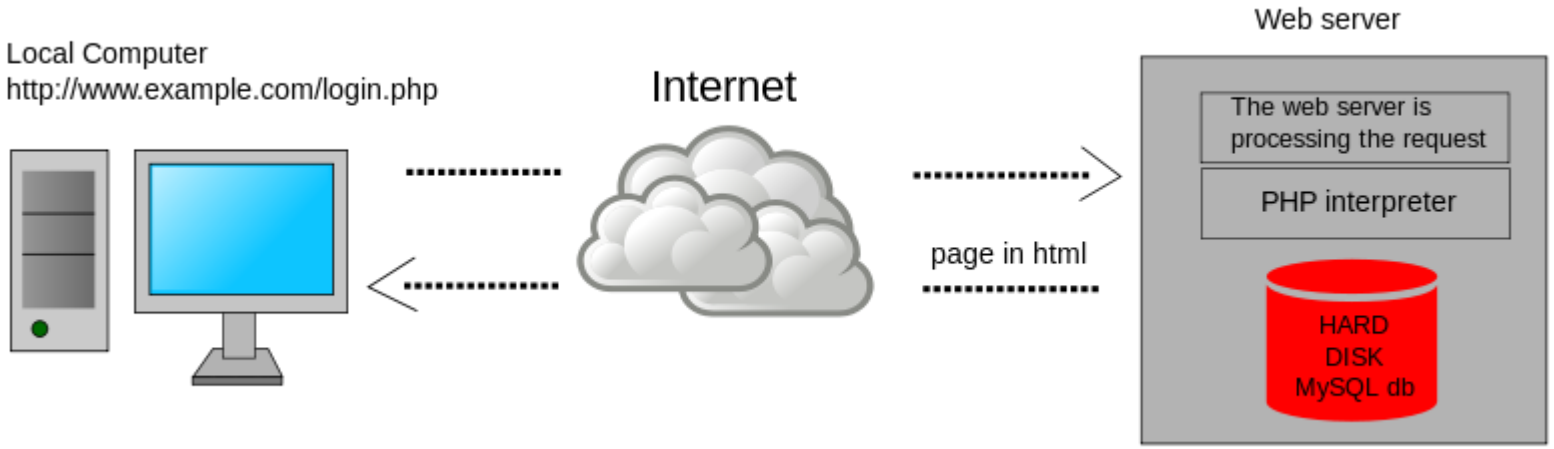
Web Server

- Continually connected to the Internet
- Runs web server software
  (such as Apache, Internet Information Server or Node.js)
- Uses HTTP (Hypertext Transfer Protocol)
- Receives request for a web page
- Responds to request and transmits status code, web page, and associated files or data

WEB ENHANCEMENTS

Dynamic web pages

- **Dynamic web pages** are web sites that are generated at the time of access by a user or change as a result of interaction with the user
- A program running on the web server (server-side scripting) is used to change the web content on the web pages sent back to the client
- Typical server-side languages are PHP, ASP, JSP, Perl, Ruby, C#, Java, and Javascript



Dynamic Html Generation

- Dynamic web pages usually consist of a static part (HTML) and a dynamic part, which is code that generates HTML
- The code that generates the HTML can do this based on variables in a template, or on code
- The text to be generated can come from a database, thereby making it possible to dramatically reduce the number of pages in a site

Consider the example of a real estate agent with 500 houses for sale.
In a static web site, the agent would have to create 500 web pages in order to make the information available.
In a dynamic website, the agent could potentially connect a single dynamic web page to a database table of 500 records.

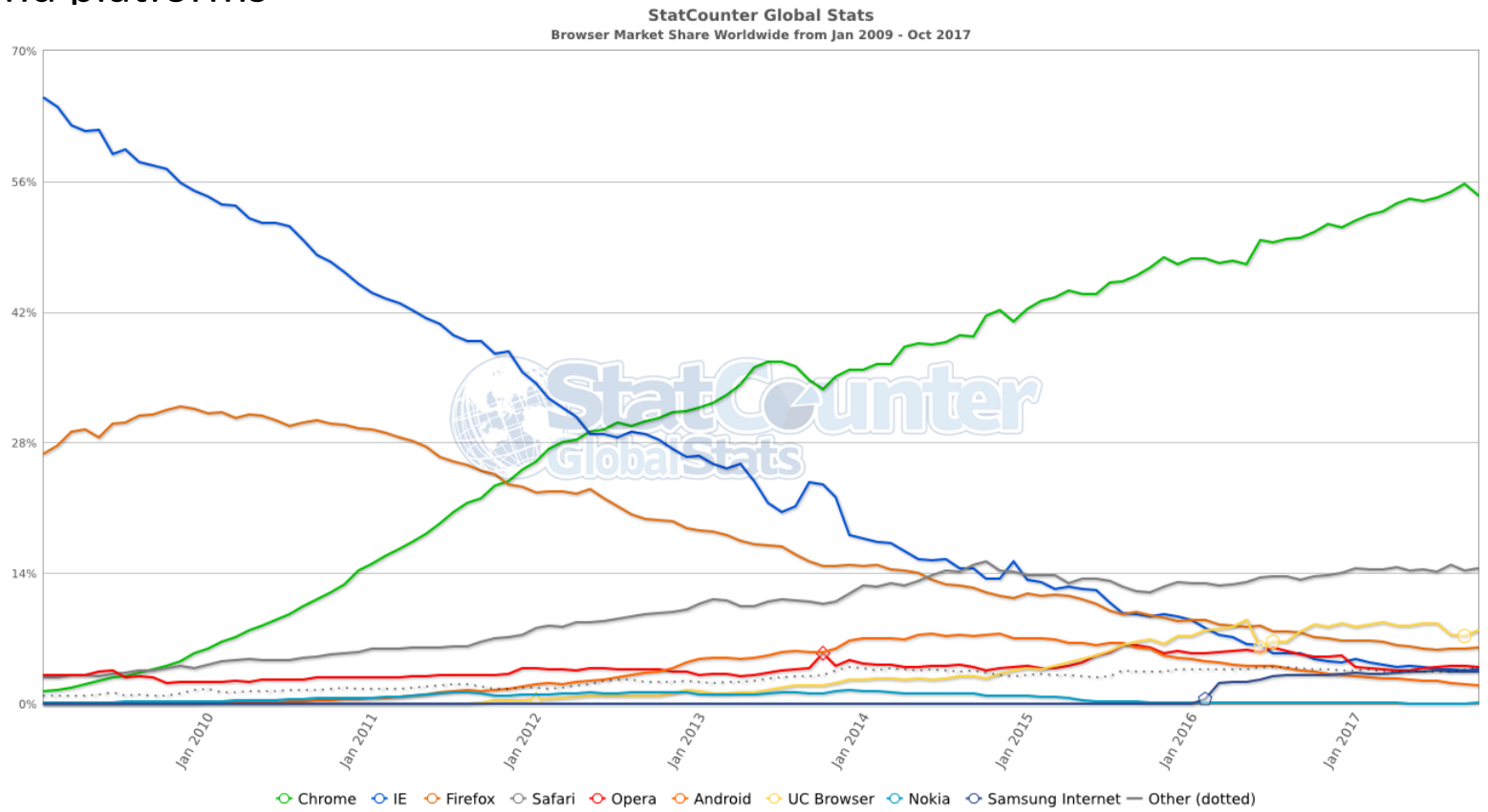
Client Side Scripting

- Client-side scripting is changing interface behaviors within a specific web page in response to mouse or keyboard actions, or at specified timing events
- In this case, the dynamic behavior occurs within the presentation
- The Client-side content is generated on the user's local computer system
- The client-side scripting languages is **JavaScript**
  - But there are outdated alternatives like Flash and Silverlight

WEB DEV TOOLS

Browsers

- Web Applications are typical developed to target all/most browsers and on different platforms
- So you have to test your pages / apps in different browsers and on different platforms
- So install all the common browsers on your development machine, and then use services on the Internet to visualize your pages on the remaining browsers and platforms



Editor or IDE?

- On Windows
  - MS Visual Studio
  - Visual Studio Code
  - Or just a simple editor like Notepad++
- All Platforms (Windows, Mac, Linux)
  - Visual Studio Code
  - JetBrains WebStorm
  - Brackets
  - Sublime Text
  - Atom
  - …

Validation

- Some IDEs have integrated validation of HTML and CSS, but if your tools does not include this service you can find it on the Internet
- HTML validation:
  - http://validator.w3.org/
  - http://html5.validator.nu
  - http://lint.brihten.com/html
- CSS validation:
  - http://jigsaw.w3.org/css-validator/
- Javascrit validation:
  - http://www.jslint.com/

Debugging

- Most browsers have some debugging aid build in (press F12)
  - Google Chrome is probably the best https://developers.google.com/chrome-developer-tools/ http://www.dotsauce.com/chrome-developer-tools/
- But sometimes it is useful to use different browsers for debugging
- Or use Fiddler or Charles Proxy:
  https://www.telerik.com/fiddler
  http://www.charlesproxy.com/

Testing Environment

- For static web pages you only need browsers
- If you use a framework like ASP.Net, PHP etc. you need a local webserver for test and debugging

- If you use Visual Studio it will install IIS express locally

- If you plan to deploy to a Apache server you can install XAMPP locally (http://apachefriends.org)

References and Links

- Wikipedia
- Web Development and Design Foundations with HTML5
  http://webdevfoundations.net/