



JACOBS
UNIVERSITY

Web Service Architectures; HTML, XML

Ramakrishnan & Gehrke, Chapter 7

www.w3schools.com

www.webdesign.com

...



Really everybody can design an own website

Overview

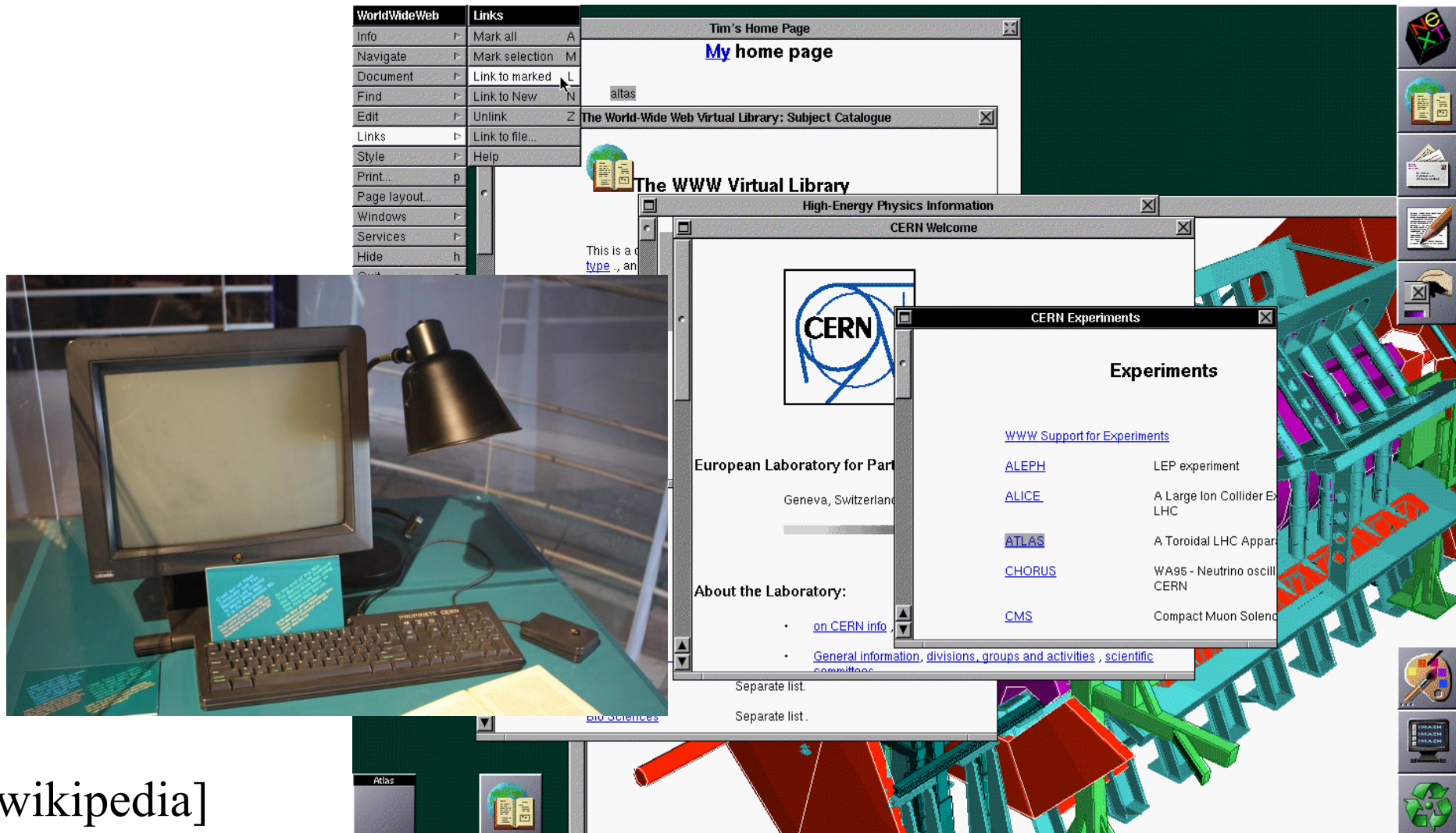
- Internet / Web Concepts
- Three-tier architectures
- Presentation layer
- Middle tier

History: The Internet and the Web

- 13th century
Incas use Quipu
- 1945 idea of linking together microfiche published by Vannevar Bush
- 1960s Internet as (D)ARPA project:
fault-tolerant, heterogeneous WAN (cold war!)
term "Hypertext" coined by Ted Nelson at ACM 20th National Conference
- 1976 Queen Elizabeth sends her first email. She's the first state leader to do so.
- 1980 Berners-Lee at CERN writes notebook program to link arbitrary nodes
- 1989 Berners-Lee makes a proposal on information management at CERN
- 1990 Berners-Lee's boss approves purchase of a NeXT cube
Berners-Lee begins hypertext GUI browser+editor and dubs it "WorldWideWeb"
First web server developed



WWW: The Beginnings



[wikipedia]

History: The Internet and the Web

- 13th century Incas use Quipu
- 1945 idea of linking together microfiche published by Vannevar Bush
- 1960s Internet as (D)ARPA project:
fault-tolerant, heterogeneous WAN (cold war!)
term "Hypertext" coined by Ted Nelson
at ACM 20th National Conference
- 1976 Queen Elizabeth sends her first email. She's the first
- 1980 Berners-Lee at CERN writes notebook program to link arbitrary nodes
- 1989 Berners-Lee makes a proposal on information management at CERN
- 1990 Berners-Lee's boss approves purchase of a NeXT cube
Berners-Lee begins hypertext GUI browser+editor and dubs it "WorldWideWeb"
First web server developed
- 1991 May 17 – general release of WWW on central CERN machines
- 1992 more browsers: Viola & Erwise released
- 1994 > 200 web servers by start of year
Mosaic: easy to install, great support, first inline images ("much sexier")
Andreessen & colleagues leave NCSA to form "Mosaic Comm. Corp"; later "Netscape"



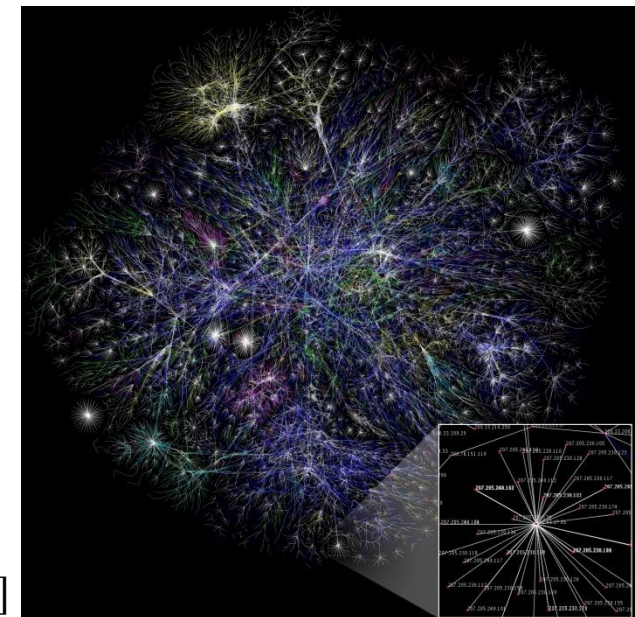
Internet & WWW

- Internet originally **4 basic services**, based on TCP & IP:
 - telnet, ftp, mail, news
 - Later many more: IRC, SSL, NTP, ...
- Each computer has worldwide unique id
 - **IP address**: n.n.n.n (32 bit IPv4, 128 bit IPv6)
 - **Domain name**: subdomain.host.top-level-domain
 - **DNS** to resolve
- **World-Wide Web** just another Internet service
 - HTTP: Hypertext Transfer Protocol
 - HTML: Hypertext Markup Language
 - **URIs** (Uniform Resource Identifiers)

telnet, ftp, ..., http
(application layer)

TCP
(transport layer)

IP
(network layer)



[wikipedia]

Uniform Resource Identifiers

- Uniform **naming schema** to identify resources on the Internet
 - resource can be anything: index.html, mysong.mp3, picture.jpg
 - Syntax: **scheme** ":" [**authority**] [**path**] ["?" **query**]
 - Ex: http://www.cs.wisc.edu/index.html, mailto:webmaster@bookstore.com, telnet:127.0.0.1
- Structure of an http URI: `http://www.cs.wisc.edu/~dbbook/index.html`
 - **Naming scheme** (http)
 - Name of **host computer** + optionally **port#** (//www.cs.wisc.edu:80) – 80 is default
 - Name of **resource** (~dbbook/index.html)
- **URL** = Uniform Resource **Locator** (subset of URIs; old term)
 - Identification via network "location"

Hypertext Transfer Protocol

- What is a **communication protocol**?
 - Set of rules that defines the structure of messages & communication process
 - Examples: TCP, IP, **HTTP**
- What happens if you click on www.cs.wisc.edu/~dbbook/index.html?
 - Client **connects** to server, **transmits** HTTP request to server
 - Server **generates** response, **transmits** to client
 - Both **disconnect**
- HTTP **header** describes content/action (text = ISO-8859-1), **content** for data
 - RFC 2616

HTTP Sample Request/Response

- Client sends:

```
GET ~dbbook/index.html HTTP/1.1
User-agent: Mozilla/4.0
Accept: text/*, image/gif, image/jpeg
```

- Server responds:

```
HTTP/1.1 200 OK
Date: Mon, 04 Mar 2002 12:00:00 GMT
Server: Apache/1.3.0 (Linux)
Last-Modified: Mon, 01 Mar 2002 09:23:24 GMT
Content-Length: 1024
Content-Type: text/html
```

```
<html> <head></head>
<body>
<h1>Burns and Nobble Internet Bookstore</h1>
Our inventory:
<h3>Science</h3>
<b>The Character of Physical Law</b>
...
</body></html>
```

Try this:
\$ telnet google.com 80
GET / HTTP/1.1
<3x newline>

HTTP Request Structure

■ Request line

GET ~/index.html HTTP/1.1

- Http **method** field (GET and POST, more later)
- local **resource** field
- HTTP **version** field

■ Type of client

User-agent: Mozilla/4.0

■ What types of files (MIME types) the client will accept

Accept: text/*, image/gif, image/jpeg

- **MIME** = Multipurpose Internet Mail (!) Extensions = file type naming system
- MIME types other than text/*, image/jpeg, image/gif, image/png need **browser plug-in** or **helper application**

HTTP Response Structure

■ Status line

HTTP/1.1 200 OK

- HTTP version: HTTP/1.1
- Status code
- Server message, textual

- *200 OK: Request succeeded*
- *400 Bad Request: Request could not be fulfilled by the server*
- *404 Not Found: Requested object does not exist on the server*
- *505 HTTP Version not supported*

■ Date when the object was created

Last-Modified: Mon, 01 Mar 2002 09:23:24 GMT

■ Number of bytes being sent

Content-Length: 1024

■ What type is the object being sent

Content-Type: text/html

■ ...plus potentially many more items, such as server type, server time, etc.

■ The payload!

<html>...</html>

HTTP Doesn't Remember!

- HTTP **stateless** on the granularity of requests
 - No “sessions”
 - Every message completely self-contained
 - No previous interaction “remembered” by protocol
- Implication for applications:
Any **state information** (shopping carts, user login information, ...) need to be **encoded** in **every** HTTP request *and* response!
- Popular **methods** on how to maintain state:
 - Cookies
 - Dynamically generate unique URLs
 - Hidden form fields

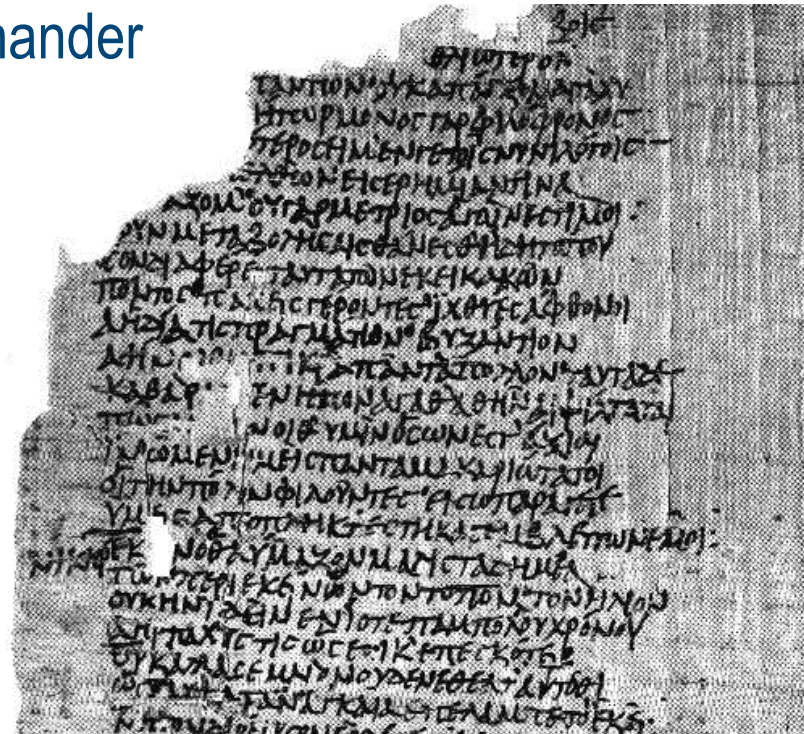
Conventions

- **index.html** (Windows: **index.htm**), .php, ...
 - If local path ends with directory, this file is assumed
 - *Ex: `http://www.myserver.foo/Downloads`*
 - If not found: **directory listing** is displayed
 - *Put dummy index.html if you don't want this, or disable default in server*
- Local path **~name/path**
 - leads to `~name/public_html/path` where *name* is local user name

Intermezzo: Documents

■ *Samia* ('The Woman from Samos') by Menander

- no space between words,
no punctuation,
no speaker's indication
- Paragraphus, ¶: A critical sign used to mark the beginning of a paragraph or section [Parkes 1992]



■ Later: Document Management Systems (DMS)

- store all enterprise documents (contracts!)
- scans (images → display) + "fulltext" (maybe via OCR → searchable)
- Ex: `Select C.pageno, C.image from Contract C where C.text like '%Adams%'`
- Problem: DMS doesn't know position/context/meaning of my search string in text body

SGML and HTML

- Task: within document, **isolate contents / structure / layout**
- **SGML** = Standard Generalized Markup Language
 - Idea: **make document structure explicit** by adding **mark(up)s** ("tags")
 - Cf. Search engines: *hit in <h1>...</h1> weighted higher than in the middle of a <p>...</p> section*
 - **Document definition** lists allowed tags → typed documents
 - Problem: complexity → not widely used
 - Focuses on contents & structure, no layout considerations
 - NB: ODA (Office Document Architecture) grasps contents+structure+layout orthogonally
- **HTML** = Hypertext Markup Language
 - SGML-based
 - Idea: format document according to **logical structure**, browser will make "something useful" out of it (h1, h2, h3, p, li, ...)
 - Practice: people (mis)use tags to enforce *layout* (b, i, ...), tweak code

"optimised for
MS IE 6.0
and 1024x768"

HTML Primer

- HTML is a data exchange format

- Unformatted ASCII

- *Proper indentation increases readability*

- Text interspersed with **tags**, some with **attributes**;
usually start and end tag:

```
<h1 align="center">headline</h1>
```

- Opening tags: “<” element name “>”

- Closing tags: “</” element name “>”

- Tags can be **nested**:

```
<h1><em>my</em> text</h1>
```

- Many editors automatically generate HTML directly from your document

- But you **need to know HTML** too, want to generate it later on!
 - And tool's code sometimes has bad quality, cf. Microsoft Word “Save as html”

HTML Primer (contd.)

```
<a name="top">
<title>My first HTML document</title>

<h1>An important heading</h1>
<h2>A slightly less important heading</h2>
<p>This is the <em>first</em> paragraph.</p>


My link list:
<ul>
  <li>This is a link to <a href="http://www.w3.org/">W3C</a>
  <li>This a link to <a href="peter.html">Peter's page</a>
  <li>Go to <a href="#top">top</a>
  <li><a href="/"></a>
</ul>
```

HTML Primer (contd.)

■ Text structuring

- Title (for browser title bar)
- Headlines
- Paragraphs, text emphasis

■ Links

- External
- Relative
- Internal

■ Images

■ Text structuring (contd.)

- Lists

```
<a name="top">
```

```
<title>My first HTML document</title>
```

```
<h1>An important heading</h1>
```

```
<h2>A slightly less important heading</h2>
```

```
<p>This is the <em>first</em> paragraph.</p>
```

```

```

My link list:

```
<ul>
```

```
<li>This is a link to <a href="http://www.w3.org/">W3C</a>
```

```
<li>This a link to <a href="peter.html">Peter's page</a>
```

```
<li>Go to <a href="#top">top</a>
```

```
<li><a href="/"></a>
```

```
</ul>
```


HTML Primer (contd.)

■ Text structuring (contd.)

- tables
- row
- column heading
- regular column

Year	Sales
2000	\$18M
2001	\$25M
2002	\$36M

```

<table>
  <tr>
    <th>Year</th>
    <th>Sales</th>
  </tr>
  <tr>
    <td>2000</td>
    <td>$18M</td>
  </tr>
  <tr>
    <td>2001</td>
    <td>$25M</td>
  </tr>
  <tr>
    <td>2002</td>
    <td>$36M</td>
  </tr>
</table>

```

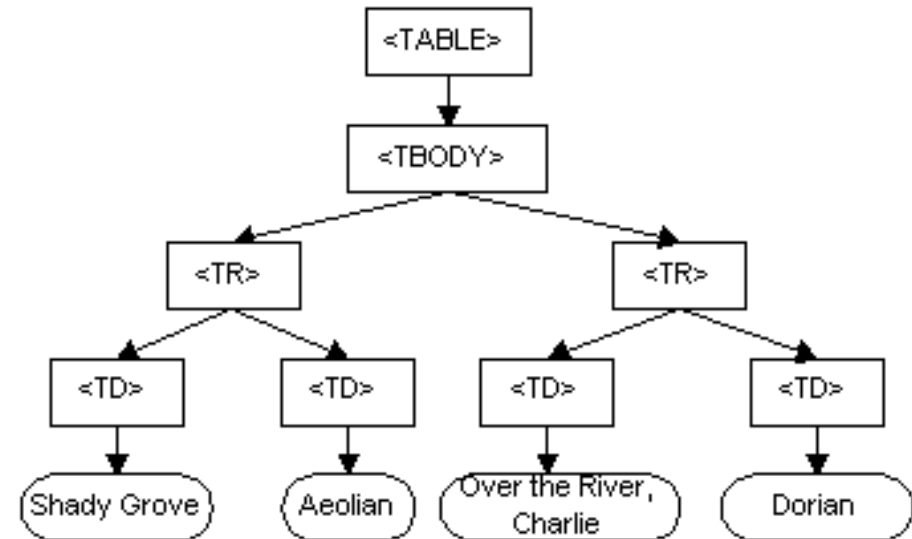
HTML Forms

- Common way to communicate data from client to server
- General format of a form:
 - ```
<form action="page.jsp" method="GET" name="loginForm">
 <input type=... value=... name=...>
</form>
```
- Components of an HTML form tag:
  - action: URI that handles the content
  - method: HTTP GET or POST
  - name: Name of the form; can be used in client-side scripts to refer to the form

# HTML and DOM

```
<TABLE>
 <TBODY>
 <TR>
 <TD>Shady Grove</TD>
 <TD>Aeolian</TD>
 </TR>
 <TR>
 <TD>Over the River, Charlie</TD>
 <TD>Dorian</TD>
 </TR>
 </TBODY>
</TABLE>
```

Exercise:  
draw DOM tree  
for some HTML snippet



# Document Object Model

- HTML document actually describes a **tree structure**
  - ...that becomes manifest as "real" tree only within browser
- So far: how can I describe such a tree for input into rendering engine?
- **Dynamic HTML**: *manipulate* tree representation while being displayed
- **Document Object Model (DOM)** =  
platform and language neutral interface that allows programs and scripts to dynamically access and update content & structure of HTML documents
  - Intro: <http://www.w3schools.com/html/dom/default.asp>
  - Definition: <http://www.w3.org/TR/DOM-Level-2-HTML>

# CSS: Cascading Style Sheets

- Idea: Separate **display style** from **structure & contents**
  - W3C recommendation = standard
- File reference to CSS, placed in HTML `<head>` section
  - `<link rel="style sheet" type="text/css" href="books.css">`
- Media specific style sheets
  - `<link rel="stylesheet" type="text/css" media="screen" href="website.css">`  
`<link rel="stylesheet" type="text/css" media="print, embossed" href="print.css">`  
`<link rel="stylesheet" type="text/css" media="aural" href="speaker.css">`



# CSS Syntax

## ■ CSS syntax (simplified)

- `css-file ::= css-def*`
- `css-def ::= selector "{" ( prop ":" val )* "}"`
- `selector ::= tag`  
| `[ tag ] "." class`  
| `[ tag ] ":" pseudo`
- `elem ::= STRING`
- `class ::= STRING`
- `pseudo ::= "link" | "visited" | ...`
- `prop ::= <predefined prop names>`
- `val ::= STRING`  
| `NUMBER [ "px" | "cm" | ... ]`

```
body { font-family:Arial,sans-serif; }
a:link { color:red }
.special { color:green; font-size:large; }
```

## ■ Effect on HTML page display:

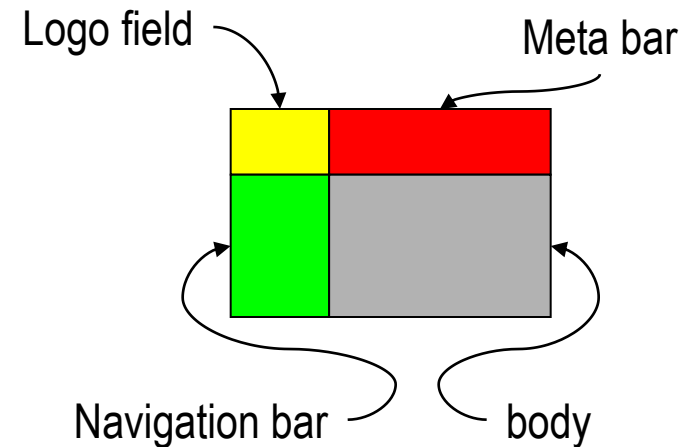
- same effect as:  
`<h1 style="font-family:Arial,sans-serif">`  
but applies to all `<h1>`
- Style used in a tag:  
`<a href="...">` is red  
(overriding a default & a definition in CSS)
- Style can be used with any tag:  
`<p class="special">`

# Web Design

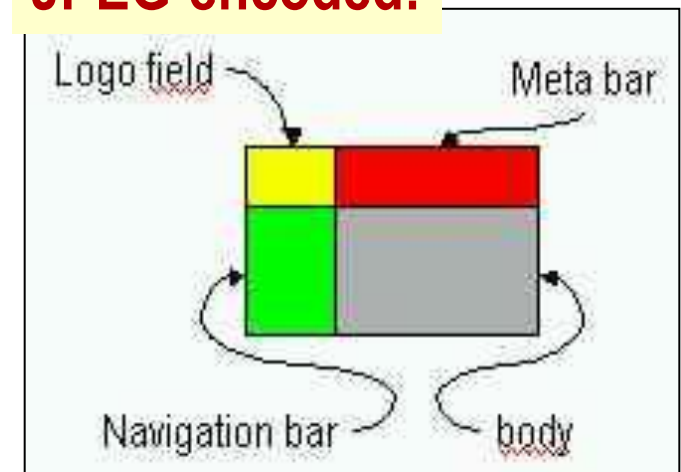
- **Corporate Design (CD)**
  - set of rules defining (visual) appearance of all company material
- **Goal of CD**
  - **recognition** of company across all media
  - transport & amplify **message**
- **preferably cooperate with a professional graphics designer!**
- **rules of thumb:**
  - **few concepts**, clearly identified
  - always have in mind **target group** (B2B vs B2C; food vs entertainment; ...)

# Web Design: Key Design Elements

- Title & key phrase & logo
  - Logo: preferably no shades, simple symbol
- Overall look & feel
  - Describe targeted CD in one sentence
- Colors: primary / secondary / background
  - Define as RGB values, PANTONE, RAL, ...; HTML!
  - Image formats: JPEG, GIF, PNG
- Fonts & typesetting
  - **serif** or **sans-serif**; max 2!
- Window subdivision
  - Scalable with window size!



## JPEG-encoded:



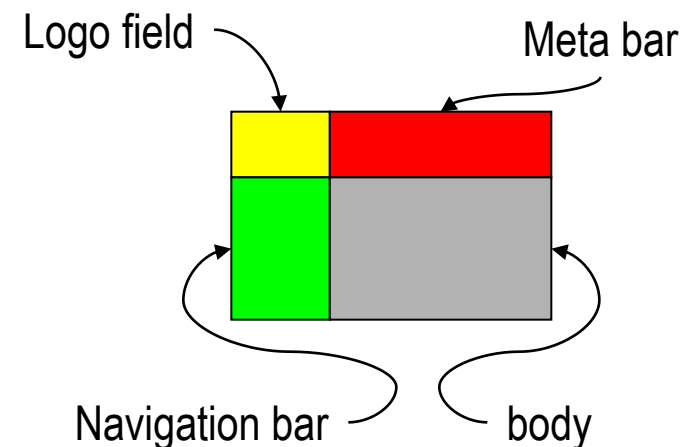
# Web Design: Common Pages

## ■ Navigation bar:

- News
- About
- The service offered
  - *Products*
  - *Solutions*
  - *Services*
- Links to related information sources

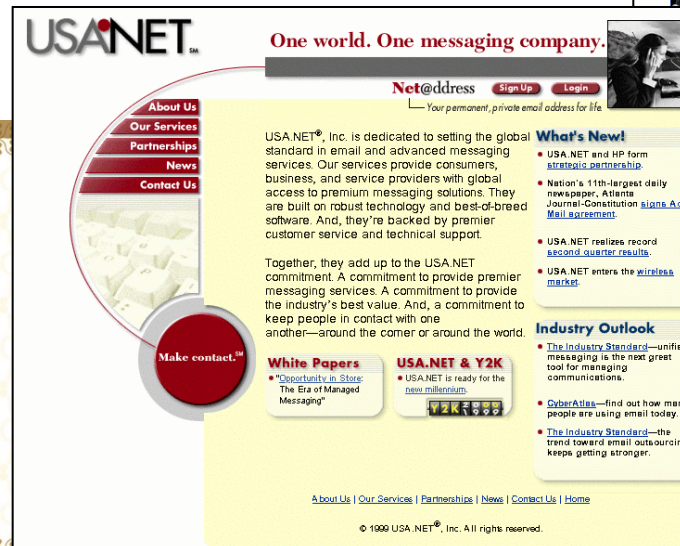
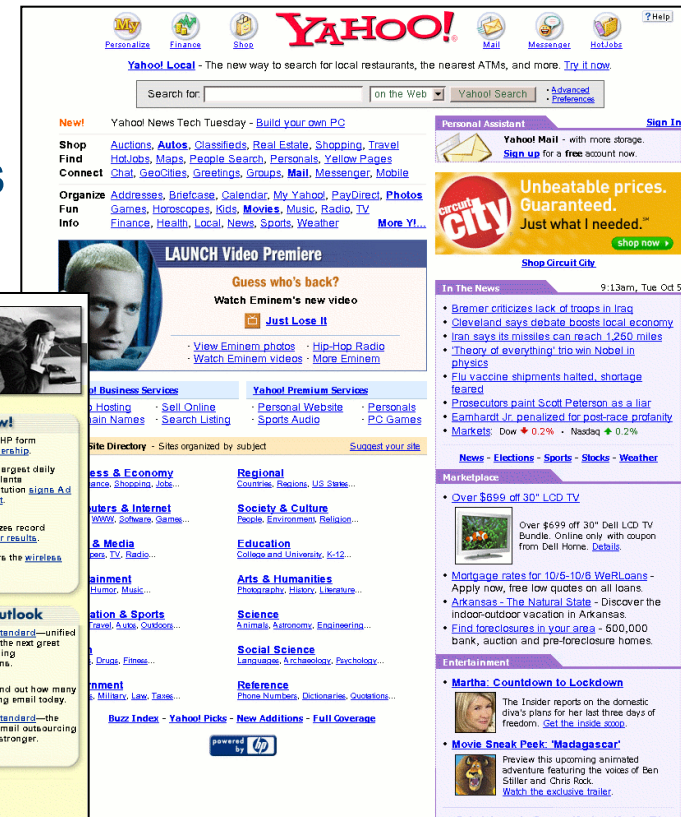
## ■ Meta bar:

- Search
- Sitemap (for larger sites)
- Contact / webmaster
- Imprint



# Web Design: Home Page Variants

- „front door“ home page approach
  - Have nice & appealing impression first, information area later
- „information rich“ home page approach
  - Give information to client with minimal mouse clicks
- Mixed approaches



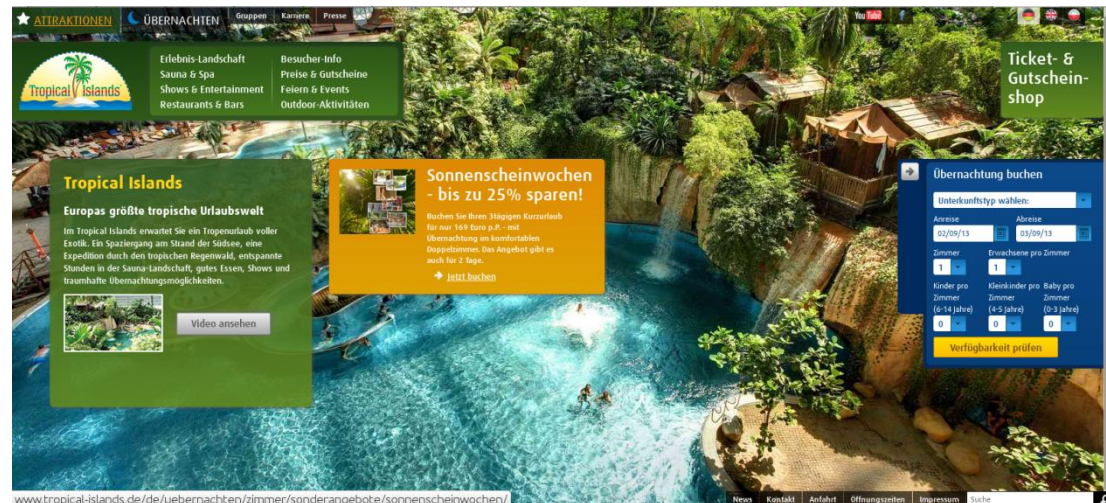
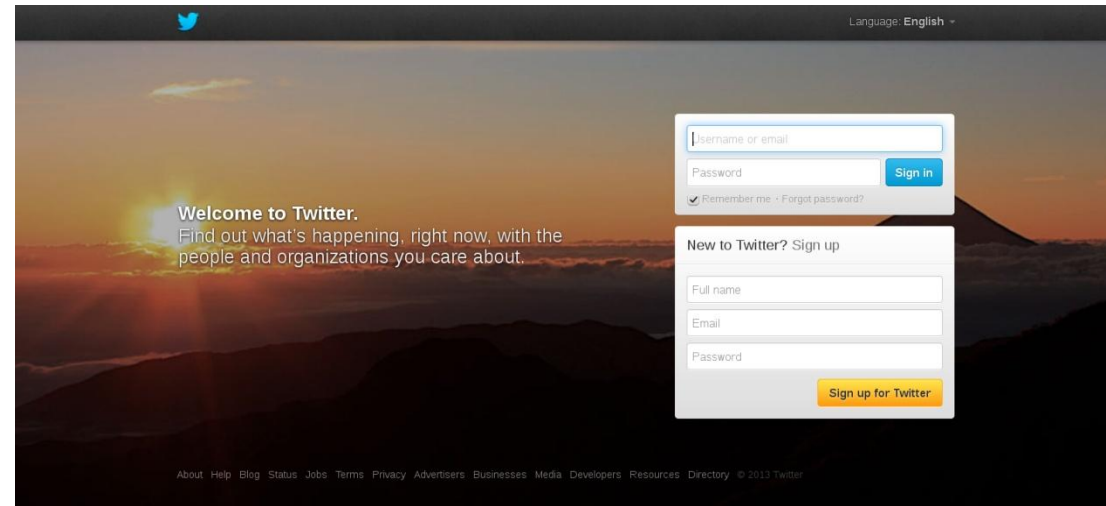
www 2001 india  
Enter

20th International World Wide Web Conference  
28th March - 1st April 2001  
HICC, Hyderabad, India



# Trendy Looks

- Responsive!
  - HDTV, table, smarthphone
- 3D effect with background
- Preferred action prominent
- Ancillary navigation at rim
  - black band at top
  - Fly-in
- ...find your own champion!



# Web Design: Good Style

- Browser independent – test it!
  - HTML checkers
  - at least Firefox & Microsoft Internet Explorer
- Suitable for handicapped clients?
- Use CSS to separate layout from contents & structure
- Use tools, such as jQuery <http://jquery.com/>  
and Twitter Bootstrap <http://getbootstrap.com/>
- ...see homework and [www.webdesign.org](http://www.webdesign.org) for more links

# Summary: WWW and HTML

- WWW: another **Internet service**, aimed at easily traversing interconnected documents
- **Protocol**: HTTP, data exchange **format**: HTML
  - captures document structure according to fixed schema
- Browser = program that
  - gets page address; fetches HTML (+ likely additional files); renders page for display
- CSS for tailoring layout
- Dynamic HTML for changing page while displayed