An Introduction to XML and Web Technologies

HTML and Web Pages

Anders Møller & Michael I. Schwartzbach

Objectives

- The history of HTML
- URLs and related schemes
- Survivor's guides to HTML and CSS
- Limitations of HTML
- Unicode
- The World Wide Web Consortium (W3C)

Hypertext

- Collections of document connected by *hyperlinks*
- Paul Otlet, philosophical treatise (1934)
- Vannevar Bush, hypothetical Memex system (1945)
- Ted Nelson introduced *hypertext* (1968)
- *Hypermedia* generalizes hypertext beyond text

Markup Languages

- Notation for adding formal structure to text
- Charles Goldfarb, the INLINE system (1970)
- Standard Generalized Markup Language, SGML (1986)
- *DTD, element, attribute, tag, entity*:

```
<!DOCTYPE greeting [
  <!ELEMENT greeting (#PCDATA)>
  <!ATTLIST greeting style (big|small) "small">
  <!ENTITY hi "Hello">
]>
<greeting style="big"> &hi; world! </greeting>
```

The Origins of the WWW

- WWW was invented by Tim Berners-Lee at CERN (1989)
- Hypertext across the Internet (replacing FTP)
- Three constituents: HTML + URL + HTTP

- HTML is an SGML language for *hypertext*
- URL is an notation for *locating files* on serves
- HTTP is a *high-level protocol* for file transfers

The Design of HTML

- Simple, purist design principles
- HTML describes the *logical structure* of a document
- Browsers are free to *interpret tags* differently
- HTML is a *lightweight* file format
- Size of file containing just "`Hello world!`":

Postscript	11,274 bytes
PDF	4,915 bytes
MS Word	19,456 bytes
HTML	28 bytes

The History of HTML

- 1992: **HTML 1.0**, Tim Berners-Lee original proposal
- 1993: HTML+, some physical layout
- 1994: HTML 2.0, standard with best features
- 1995: Non-standard Netscape features
- 1996: Competing Netscape and Explorer features
- 1996: **HTML 3.2**, the Browser Wars end
- 1997: HTML 4.0, stylesheets are introduced
- 1999: **HTML 4.01**, we have a winner!
- 2000: **XHTML 1.0**, an XML version of HTML 4.01
- 2001: XHTML 1.1, modularization
- 2002: XHTML 2.0, simplified and generalized

Uniform Resource Locator

- A Web resource is located by a URL

`http://www.w3.org/TR/html4/`

scheme *server* *path*

- Relative URL
  `sgml/dtd.html`

- Fragment identifier
  `http://www.w3.org/TR/HTML4/#minitoc`

URIs, URNs, and IRIs

- Uniform Resource Identifier (URI)
  scheme:scheme-specific-part
  Conventions about use of `/`, `#`, and `?`
- Uniform Resource Name (URN)
  `urn:isbn:0-471-94128-X`
- International Resource Identifier (IRI)
  `http://www.blåbærgrød.dk/blåbærgrød.html`
  `http://www.xn--blbrgrd-fxak7p.dk/bl%E5b%E6rgr%F8d.html`

Survivor's Guide to HTML

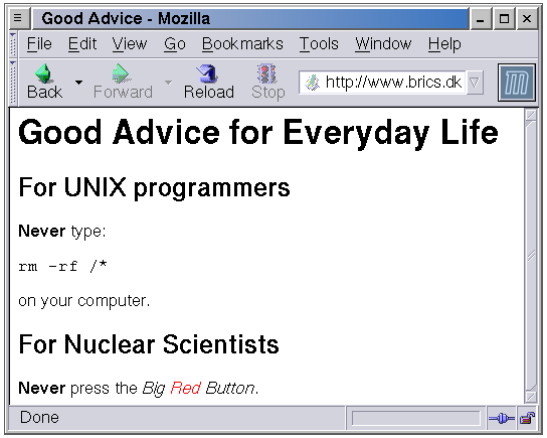
- Overall structure of an HTML document

```
<html>
  <head>
    <title>The Title of the Document</title>
  </head>
  <body bgcolor="white">
  ...
  </body>
</html>
```


Simple Formatting (1/2)

```
<html>
  <head>
    <title>Good Advice</title>
  </head>
  <body>
    <h1>Good Advice for Everyday Life</h1>
    <h2>For UNIX programmers</h2>
    <b>Never</b> type:
    <p><tt>rm -rf /*</tt><p>
    on your computer.
    <h2>For Nuclear Scientists</h2>
    <b>Never</b> press the
    <i>Big <font color="red">Red</font> Button</i>.
  </body>
</html>
```


Simple Formatting (2/2)



More Formatting

```
<html>
  <head>
    <title>Things To Do</title>
  </head>
  <body>
    <ol>
      <li>Feed the cat.
      <li>Try out the shell command:
      <pre>foreach x ( `ls` )
  cat $x | tr "aeiouy" "x" > $x
end</pre>
      <li>Buy ticket for Timbuktu.
    </ol>
  </body>
</html>
```

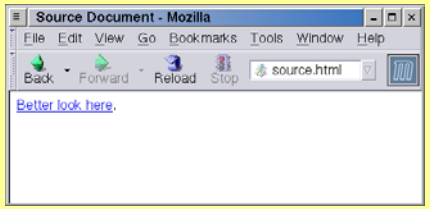
1. Feed the cat.
2. Try out the shell command:

```
foreach x ( `ls` )
  cat $x | tr "aeiouy" "x" > $x
end
```

3. Buy ticket for Timbuktu.

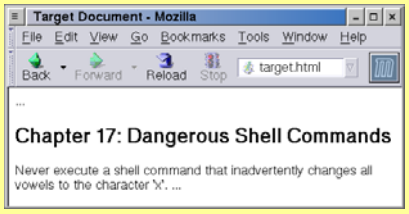
Hyperlinks: Source Document

```
<html>
  <head>
    <title>Source Document</title>
  </head>
  <body>
    <a href="target.html#danger">Better look here</a>.
  </body>
</html>
```



Hyperlinks: Target Document

```
<html>
  <head>
    <title>Target Document</title>
  </head>
  <body>
    ...
    <a name="danger"></a>
    <h2>Chapter 17: Dangerous Shell Commands</h2>
    Never execute a shell command that inadvertently changes
    all vowels to the character 'x'.
  </body>
</html>
```



Tables

```
<table border="1">
  <tr>
    <td>PostScript</td>
    <td align="right">11,274 bytes</td>
  </tr>
  <tr>
    <td>PDF</td>
    <td align="right">4,915 bytes</td>
  </tr>
  <tr>
    <td>MS Word</td>
    <td align="right">19,456 bytes</td>
  </tr>
  <tr>
    <td>HTML</td>
    <td align="right">28 bytes</td>
  </tr>
</table>
```

PostScript	11,274 bytes
PDF	4,915 bytes
MS Word	19,456 bytes
HTML	28 bytes

Tables for Alignment

```
<table width="100%">
  <tr>
    <td align="left">
      <a href="index.html"><img src="home.gif" border="0"></a>
      <a href="info.html"><img src="info.gif" border="0"></a>
    </td>
    <td align="right">
      <a href="links.html"><img src="left.gif" border="0"></a>
      <a href="survivor.html"><img src="right.gif" border="0"></a>
    </td>
  </tr>
</table>
<h1>Using Tables</h1>
```

Fill-Out Forms

Collects named values from the client:

```
<form method="get" action="http://www.google.com/search">
  <input type="text" name="q">
  <input type="submit" name="btnG" value="Google Search">
</form>
```

Google Search

GUI Elements

```
<input name="foo" type="text" size="20">
<hr>
<input name="bar" type="radio" value="s">Small
<input name="bar" type="radio" value="m">Medium
<input name="bar" type="radio" value="l">Large
<hr>
<input name="baz" type="checkbox" value="c">Cheese
<input name="baz" type="checkbox" value="p">Pepperoni
<input name="baz" type="checkbox" value="a">Anchovies
<hr>
<select name="bar">
  <option value="s">Small
  <option value="m">Medium
  <option value="l">Large
</select>
<hr>
<select name="baz" multiple>
  <option value="c">Cheese
  <option value="p">Pepperoni
  <option value="a">Anchovies
</select>
<hr>
<textarea name="foo" rows="5" cols="20">
Write something here...
</textarea>
<hr>
<input name="foo" type="password" value="tomato">
<hr>
<input name="foo" type="file">
<hr>
<input name="foo" type="hidden" value="you can't see this">
<hr>
<input name="qux" type="image" src="Denmark.gif">
<hr>
<input type="submit" value="Submit this form">
<hr>
<input type="reset" value="Reset this form"
```

Logical Versus Physical

Phone Numbers

- John Doe, ***(202) 555-1414***
- Jane Dow, ***(202) 555-9132***
- Jack Doe, ***(212) 555-1742***

Logical structure

- the page starts with a header
- the entries are written in a list
- numbers are emphasized

Physical layout

- headers are centered, huge, and grey
- lists have square bullets
- emphasis is rendered in bold-style italics

Survivor's Guide to CSS

- Cascading Stylesheets separate structure from layout
- The essential concepts are *selectors* and *properties*
- Properties may have different *values*:

color	red, yellow, rgb(212,120,20)
font-style	normal, italics, oblique
font-size	12pt, larger, 150%, 1.5em
text-align	left, right, center, justify
line-height	normal, 1.2em, 120%
display	block, inline, list-item, none

Structure of a Stylesheet

- A selector is a *list of tag names*
- For each selector, some properties are assigned values:

```
b {color: red; font-size: 12pt}
i {color: green}
```

- Longer selectors give *context sensitivity*:

```
table b {color: red; font-size: 12pt}
form b {color: yellow; font-size: 12pt}
i {color: green}
```

- The most *specific* selector is chosen to apply

Specificity in Action

```
<head>
  <style type="text/css">
    b {color: red;}
    b b {color: blue;}
    b.foo {color: green;}
    b b.foo {color: yellow;}
    b.bar {color: maroon;}
  </style>
  <title>CSS Test</title>
</head>
```

```
<body>
  <b class=foo>Hey!</b>
  <b>Wow!
     <b>Amazing!</b>
     <b class=foo>Impressive!</b>
     <b class=bar>kOOl!</b>
     <i>Fantastic!</i>
  </b>
</body>
```

Hey! Wow! Amazing! Impressive! KOOl! *Fantastic!*

Applying a Stylesheet

```
h1 { color: #888; font: 50px/50px "Impact"; text-align: center; }
ul { list-style-type: square; }
em { font-style: italic; font-weight: bold; }
```

```
<html>
  <head>
    <title>Phone Numbers</title>
    <link href="style.css" rel="stylesheet" type="text/css">
  </head>
  <body>
    <h1>Phone Numbers</h1>
    <ul>
      <li>John Doe, <em>(202) 555-1414</em>
      <li>Jane Dow, <em>(202) 555-9132</em>
      <li>Jack Doe, <em>(212) 555-1742</em>
    </ul>
  </body>
</html>
```

HTML Validity

- HTML has a formal syntax specification
- 800 lines of DTD notation
- A *validator* gives syntax errors for invalid documents
- Most HTML documents on the Web are *invalid*:

www.microsoft.com	123 errors
www.cnn.com	58 errors
www.ibm.com	30 errors
www.google.com	27 errors
www.sun.com	19 errors

- Valid documents may contain this logo: 

Validation Errors

```
Line 3, column 7: document type does not allow element "BODY" here.
    <body>
          ^
Line 4, column 13: document type does not allow element "B" here; assuming missing "CAPTION" start-tag
    <table><b>123</i></table>
               ^
Line 4, column 20: end tag for element
    <table><b>123</i></table>
                      ^
Line 4, column 28: end tag for "B" omit
    <table><b>123</i></table>
                            ^
Line 4, column 11: start tag was here.
    <table><b>123</i></table>
          ^
Line 4, column 28: end tag for "CAPTION" omitted, but its declaration does not permit this.
    <table><b>123</i></table>
                            ^
Line 4, column 11: start tag was here.
    <table><b>123</i></table>
          ^
Line 4, column 28: end tag for "TABLE" which is not finished.
    <table><b>123</i></table>
                            ^
Line 6, column 6: end tag for "HTML" which is not finished.
    </html>
```

```
<html>
  <body>
    <table><b>123</i></table>
  </body>
</html>
```


Reasons for Invalidity

- Ignorance of the HTML standard
- Lack of testing
  - ”This page is optimized for the XYZ browser”
  - ”This page is best viewed in 1024x768”
- Automatic tools generate invalid HTML output
- Forgiving browsers try to interpret invalid input

```
<h2>Lousy HTML</h1>
<li><a>This is not very</b> good.
<li><i>In fact, it is quite bad</em>
</ul>
But the browser does <a naem="goof">something.
```

Lousy HTML

- This is not very good.
- *In fact, it is quite bad*

Problems with Invalidity

- There are several different browsers
- Each browsers has many different implementations
- Each implementation must *interpret* invalid HTML
- There are many arbitrary *choices* to make

- The HTML standard has been *undermined*
- HTML renders differently for most clients

A Standard for Invalid HTML

- The HTML Tidy tool tries to save the situation
- Invalid HTML is transformed to (almost) valid HTML
- Still many arbitrary choices, but now we agree

```
<h2>Lousy HTML</h1>
<li><a>This is not very</b> good.
<li><i>In fact, it is quite bad</em>
</ul>
But the browser does <a naem="goof">something.
```

```
<html>
<head>
<title></title>
</head>
<body>
<h2>Lousy HTML</h2>
<ul class="noindent">
<li><a>This is not very good.</a></li>
<li><i>In fact, it is quite bad</i></li>
</ul>
But the browser does <a naem="goof">something.</a>
</body>
</html>
```


HTML for Recipes

```
<h1>Rhubarb Cobbler</h1>
<h2>Wed, 4 Jun 95</h2>
This recipe is suggested by Jane Dow.
Rhubarb Cobbler made with bananas as the main sweetener.
It was delicious.

  <table>
  <tr><td> 2 1/2 cups <td> diced rhubarb
  <tr><td> 2 tablespoons <td> sugar
  <tr><td> 2 <td> fairly ripe bananas
  <tr><td> 1/4 teaspoon <td> cinnamon
  <tr><td> dash of <td> nutmeg
  </table>

<i>Combine all and use as cobbler, pie, or crisp.</i>
<p>
This recipe has 170 calories, 28% from fat,
58% from carbohydrates, and 14% from protein.
<p>
Related recipes: <a href="#GardenQuiche">Garden Quiche</a>
is also yummy.
```

Rhubarb Cobbler

Wed, 4 Jun 95

This recipe is suggested by Jane Dow. Rhubarb Cobbler made with bananas as the main sweetener. It was delicious.

2 1/2 cups	diced rhubarb
2 tablespoons	sugar
2	fairly ripe bananas
1/4 teaspoon	cinnamon
dash of	nutmeg

Combine all and use as cobbler, pie, or crisp.

This recipe has 170 calories, 28% from fat, 58% from carbohydrates, and 14% from protein.

Related recipes: Garden Quiche is also yummy.

Limitations of HTML

- HTML is designed for hypertext, not for recipes
- Structure and presentation is intertwined
- HTML validation is less than recipe validation
- HTML standards have been undermined

- We need a special *Recipe Markup Language*!

Bytes vs. Characters

- HTML files are represented as text files
- A text file is logically a sequence of **characters** but physically a sequence of **bytes**
- Several mappings exist:
  - ASCII
  - EBCDIC
  - **Unicode**
- Unicode aims to cover all characters in all past or present written languages

Unicode Characters

- A **character** is a symbol that appears in a text
  - letters of the alphabet
  - pictograms (like ©)
  - accents
- Unicode characters are abstract entities:
  - `LATIN CAPITAL LETTER A`
  - `LATIN CAPITAL LETTER A WITH RING ABOVE`
  - `HIRAGANA LETTER SA`
  - `RUNIC LETTER THURISAZ THURS THORN`

Unicode Glyphs

- A **glyph** is a graphical presentation
- A typical example is: Å
- This may represent several characters:
  - `LATIN CAPITAL LETTER A WITH RING ABOVE`
  - `ANGSTROM SIGN`
- Or even a sequence of characters:
  - `LATIN CAPITAL LETTER A`
    `COMBINING RING ABOVE`
- Some characters even result in several glyphs

Unicode Code Points

- A **code point** is a unique number assigned to every Unicode character
- Code points are between 0 and 1,114,112
- Only around 100,000 are used today
- The character `HIRAGANA LETTER SA` is assigned the code point 12,373
- Code point 0 through 127 coincide with ASCII
- Some code point are never assigned

Unicode Character Encoding

- A **character encoding** interprets a sequence of bytes as a sequence of code points
- The bytes are first parsed into **code units**
- Code units have a fixed length
- One or more code units may be required to denote a code point
- Examples are UTF-8, UTF-16, UTF-32

UTF-8

- A code unit is a single byte
- A code point is from 1 to 4 code units
- Code units between 0 and 127 directly represent the corresponding code points
- `110XXXXX` indicates that 2 code units are used
- `1110XXXX` indicates that 3 code units are used
- `11110XXX` indicates that 4 code units are used
- The remaining code units looks like `10XXXXXX`

UTF-8 Example

- `11100011 10000001 10010101`
- **`1110`**`0011` **`10`**`000001` **`10`**`010101`
- `11000001010101`
- `12,373`
- `HIRAGANA LETTER SA`

UTF-16

- A code unit consists of 2 bytes
- Code points below 65,536 are in a single code unit
- Higher code points are represented as:
  - `110110XXXXXXXXXX 110111XXXXXXXXXX`

  (after subtracting 65,536)
- This makes sense because Unicode assign no code points between the numbers:
  `1101100000000000 (55,296)`

  and

  `1101111111111111 (57,343)`

Byte Order

- When reading several bytes at once, we must consider the **byte order** of the architecture
- UTF-16 starts any text with the special code point:
  `1111111011111111 (65,279)`

  called **zero-width non-breaking space**
- The dual code point
  `1111111111111110 (65,534)`

  is never assigned
- UTF-16LE and UTF-16BE may avoid this

UTF-16 Example

- 11111110 11111111 00110000 01010101
- **11111110 11111111** 00110000 01010101
- 00110000 01010101
- 12,373
- HIRAGANA LETTER SA

Unicode in Java

- Java represents characters as UTF-16 code units – not as UTF-16 code points!
- A pragmatic choice to use only 16 bits
- The `length` function on strings may be wrong (see `String.codePointCount`)
- Some strings may represent illegal data

ISO-8859-1

- Another popular character encoding
- Only 256 code points
- Single byte code units
- Coincides with ASCII on code points 0-127
- Cannot represent general Unicode

- In all, there are hundreds of different encodings...

Character Encodings in HTML

- The document may declare its own encoding:

```
<meta http-equiv="Content-Type"
      content="text/html; charset=ISO-8859-1">
```

- This works if the encoding coincides with ASCII

- Unicode characters may be represented as:
  `&#12373;`

World Wide Web Consortium (W3C)

- Develops HTML, CSS, and most Web technology
- Founded in 1994
- Has 380 companies and organizations as members
- Is directed by Tim Berners-Lee
- Located at MIT (US), Inria (France), Keiko (Japan)

W3C Players

- Members ($50,000 per year)
- Team
- Advisory board
- Technical Architecture Group
- Working Groups

W3C Documents

- Working Drafts
- Candidate Recommendations
- Proposed Recommendations
- Recommendations

- Working Group Notes
- Member Submissions
- Staff Comments
- Team Submissions

W3C Principles

- Consensus among members
- Limited intellectual property rights
- Free Web access to technical reports (unlike ISO)

Summary

- History and structure of HTML, CSS, URL/URI/URN/IRI, and Unicode
- Survivor's guides to these technologies
- Limitations of HTML for general data (the recipe collection example)

Essential Online Resources

- http://www.w3.org/TR/html4/
- http://www.w3.org/Addressing/
- http://www.w3.org/Style/CSS/
- http://validator.w3.org/
- http://www.w3.org/