

HTTP Part 2

HTML on the Client

Most often the HTTP response to a client request is Hypertext Markup Language (HTML)

The HTML includes URL links that, when clicked, lead to further HTTP transactions.

The HTML is designed to be portable – it should be possible for many different types of client to display it.

The browser (Safari, Internet Explorer, Mozilla, ...) displays the HTML as best it can – HTML will look slightly different on each platform. This flexibility is what makes HTML portable.

HyperText Markup Language – HTML

Portable Content

Represent textual and image content.

Defines the *structure* of the document, not its exact *appearance*. Better than plain ASCII, not as rich in appearance as Postscript or PDF.

Encode text and image "page" content in a way that can be displayed on a wide variety of platforms

Page is made of text content, some layout information, images, and URL links

Different Platforms

Different OSes, different window/screen sizes, different installed fonts that render at different sizes.

Users can have particular preferences on what font to use.

References

HTML Primer

<http://archive.ncsa.uiuc.edu/General/Internet/WWW/HTMLPrimer.html>

<http://www.htmlprimer.com/> a site I blundered across

W3C

World Wide Web Consortium [W3C] – the non-profit which tries to maintain a portable definition of HTML and other Internet related standards (XML)

<http://www.w3.org/MarkUp/>

Setting up a page in leland space

<http://www.stanford.edu/leland/howto.shtml>

View Source

To learn how a page is written, use the "View Source" option in your browser to look at the HTML source code. This is a quick way to learn.

Basic Document Structure

An HTML file is just a text file. The HTTP server sends it over to the client for display in the client browser.

Structure vs. Appearance

HTML defines the basic structure of a document, but the exact appearance is determined by the browser dynamically.

Text and tags.

Tags enclosed in <>'s. Some tags occur by themselves e.g.
, others are paired to enclose some text <h1>...</h1>. Upper and lower case tags act the same.

The HTML itself is plain text — no bold, no fonts, no rulers, etc. Just a sequence of characters. You can edit the HTML text in a word-processor or text editor. Just be sure to save it as "plain text" and not some other format.

Whitespace

“Whitespace” (spaces, tabs, end-of-lines, ...) are largely ignored.

In particular, since \r \n all count equally as whitespace, the EOLN convention used will not change the way an HTML document renders.

What Defines Appearance?

Appearance in the browser is determined by..

What fonts and sizes the user has chosen in their browser preferences.

The current width of their browser window.

Pour

The HTML “pours into” this structure set by the user.

Try changing the width of your window — the content defined by the HTML should rearrange to present itself in the window.

Properly written HTML will look reasonable on many different platforms – different browsers, and even on things like Palm Pilots (essentially just a very narrow window). See <http://cslibrary.stanford.edu/104/> for an example of a page with many elements that “pours” into the browser page – try varying the window width and font sizing to see how it adjusts.

HTML Pet Peeve

Smart HTML authors understand that they control the structure, but the browser, its fonts, its width, etc. control the exact appearance. HTML authors who do not “get” this basic feature, are doomed to (a) be frustrated, (b) spend 10x too long tweaking their documents, and (c) their documents will only look right on the platform/browser/fonts combination the author happened to use or test on.

Basic Tags

Preamble line

```
<!doctype html public "-//w3c//dtd html 4.0 transitional//en">
```

Identifies the content to follow as HTML. As a practical matter, can be omitted.

```
<html>...</html>
```

Encloses the entire HTML source.

```
<head>...</head>
```

Encloses the "header" section of an HTML document. The header section most often just contains the document title, but it can contain other meta information about the document.

Such meta information may be better developed in the future.

<meta> tags – used inside the head section to specify information about the document (these are from the page <http://cslibrary.stanford.edu/104/>)

```
<meta name="description" content="Stanford CS Education Library: a fun 3 minute video
  that explains the basics features of pointers.">
```

```
<meta name="keywords" content="pointer, pointee, reference, pointer fun, basic pointer,
  pointer introduction, video, animation, animated">
```

```
<title>...</title>
```

The title phrase for this page. Polite to always have one. Shows up as the title of the browser window, although that's a little to subtle to be useful. Also used by search engines, bookmarks, etc. as a one phrase name for this page. I frequently have use the same phrase in the <title> and a top of page <h1>.

```
<body>...</body>
```

The body is basically all of the document but the header.

Add a bgcolor=white attribute to set the page background color.

```
<h1>Large Header</h1>
```

```
<h2>Smaller Header</h2>
```

```
<p>
```

Marks the beginning of a block of text which should stick together as single paragraphs. In the browser, each paragraph will be set off from its surroundings by some vertical whitespace.

All of the text after the <p> until another paragraph or header marker will be gathered up into a single paragraph. Blank lines etc. do not terminate the paragraph. The </p> is not required, but it may be someday.

```
<ul>...</ul>
```

Unordered list. Each item (below) shows up with a bullet. There's an ordered list

```
<ol>..</ol>
```

variant where the items are numbered.

```
<li>
```

Each item in the list begins with an . The items themselves behave pretty much like little paragraphs. There's no tag, the beginning of one list item marks the end of the previous.

Miscellaneous Adornment Tags

```
<br>
```

Force a line break (aka a “return”) in the appearance of the text. Does not introduce extra vertical whitespace...Use <p> if you want a line break and vertical whitespace.

```
<hr>
```

Horizontal Rule. Draws a horizontal line. Good cheap way to provide visual separation of logically separate elements.

Physical Styles

Indicate character appearance. These have won out over logical styles mostly

```
<b>...</b>
```

Bold.

```
<i>...</i>
```

Italic.

```
<blink>...</blink>
```

Regarded as quite-vulgar. On problem is: how do you print it on paper? I heard that Marc Andreesson admitted to being drunk when he added this feature to Mosaic. Most browsers will not render blink tags any more.

Logical Styles

Define the logical role of the text. Also causes the text to have a distinguishing appearance in the browser. The exact appearance is up to the browser.

`...`

Emphasis. Makes the text stand out in some way. (Often italic)

`...`

Strong emphasis. Make the text stand out more so. (Often bold)

`<cite>...</cite>`

Encloses the name of something, book, movie, etc. Someday, this information may be used to build more intelligent cross-references of the web.

`<address>...</address>`

A postal or email address or reference. Typically shows up on its own line.

Special Characters

Note that each of the special sequences ends with a semicolon (;)

`<` – "<"

`>` – ">"

`&` – "&"

` ` – a space

Table

To get multiple things to align vertically or horizontally, put them in a table.

To align top edges of things, put them all in the same row

To align the left edges of things, put them all in the same column

`<table> ... </table>` around the whole thing

`<tr>...</tr>` – for each row

`<td>...</td>` – for each element within a row

Use `valign=top` to force the contents of a `<td>` to its top

`<th>...</th>` -- the "header" version of a `<td>`, gives it a bold appearance suitable for a heading

Basic HTML Example

```
<html>
```

```
<head>
```

```
<title> HTML Basic </title>
```

```
</head>
```

```
<body bgcolor=white>
```

```
<!--
```

```
    This is what a comment looks like
```

```
-->
```

```
<h1> HTML Basic </h1>
```

```
<h2> Slightly Smaller Header </h2>
```

`<p>` Here's some body text. HTML runs all this text together, basically ignoring spaces, tabs, and blank lines until it gets to a tag which

tells it to
be something other
than a

body text paragraph.

`<p>` Another body text paragraph. Each of these is typically separated from whatever precedes it by a little vertical whitespace.

```
<h3>&lt;br&gt;</h3>
```

`<p>`Use the `
` tag to force a linebreak. Try not to use this -- it tends to produce less portable results since it stops accounting for the window width in use.

```
<h3>&lt;hr&gt;</h3>
```

Use `<hr>` to introduce a horizontal line to divide sections.

```
<h2>Lists!</h2>
```

```
<p>
And now some exciting foods...
<ul>
<li>Bannana -- a yellow fruit</li>
<li>Apple -- a red fruit
<li>Potato -- a starchy tuber
</ul>
```

```
<h2>Physical Character Styles</h2>
<ul>
<li><b>Bold</b>
<li><i>Italic</i>
<li><tt>Fixed Width</tt>
</ul>
```

```
<h2>Logical Character Styles</h2>
<ul>
<li><cite>Cite</cite>
<li><address>Italic</address>
</ul>
```

```
<h2>Tables</h2>
```

```
<table border=1>
<tr>
  <th>Fruit</th>
```

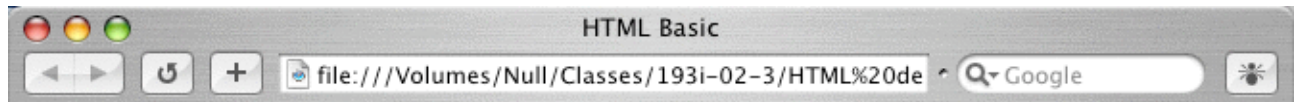
```
<th>Feature</th>
</tr>

<tr>
  <td>Bannana</td><td>Squishy yellowness</td>
</tr>

<tr>
  <td>Apricot</td><td>Dried out orangey-ness</td>
</tr>

<tr>
  <td>Mystery Fruit with a long name</td>
  <td>Mysteriousness -- note how this
  left aligns with the earlier features. Tables are the portable way
  to get things to line up.</td>
</tr>
</table>

</body>
</html>
```



HTML Basic

Slightly Smaller Header

Here's some body text. HTML runs all this text together, basically ignoring spaces, tabs, and blank lines until it gets to a tag which tells it to be something other than a body text paragraph.

Another body text paragraph. Each of these is typically separated from whatever precedes it by a little vertical whitespace.

`
`

Use the `
`

tag to force a linebreak. Try not to use this -- it tends to produce less portable results since it stops accounting for the window width in use.

`<hr>`

Use `<hr>`

To introduce a horizontal line to divide sections.

Lists!

And now some exciting foods...

- Bannana -- a yellow fruit
- Apple -- a red fruit
- Potato -- a starchy tuber

Physical Character Styles

- **Bold**
- *Italic*
- Fixed Width

Logical Character Styles

- *Cite*
- *Italic*

Tables

| Fruit | Feature |
|--------------------------------|--|
| Bannana | Squishy yellowness |
| Apricot | Dried out orangey-ness |
| Mystery Fruit with a long name | Mysteriousness -- note how this left aligns with the earlier features. Tables are the portable way to get things to line up. |

URLs

URL

Uniform Resource Locator – specify the location where a document may be retrieved. There's research into URN Uniform Resource Name which identifies a document reliably no matter where it is stored. URNs are not currently really used, but it's a clear future direction.

For example: `http://www.stanford.edu/class/cs193i/`

Scheme or protocol – `http`

Host – `www.stanford.edu`

Can include a port number other than 80 like this – `www.foo.com:8080`

“Path” – `/class/cs193i/`

Later, we'll divide the path in to sub-parts

Anchor Tag `<a>`

`underlined anchor text`

There can be other `foo="yadda yadda"` bindings in the `<a>` tag – don't assume the `href` is the only one. Also, the quotes may be omitted if the url contains only ordinary characters

The anchor tag can make any element in the document (words in paragraphs, header, lists, pictures ...) a “hyperlink” to another document. In the browser, the link will show up with a distinctive appearance – typically blue underlining.

E.G.

`CS193i`

Shows up as an underlined CS193i which links to the class page when clicked.

Blue Underline

It's nice to not override default appearance for links, since even the most naive user has learned to associate the blue underline with the "link" idea. If pages use different appearances for linking, then everything begins to look potentially clickable which is a step backwards in usability for the web.

Usability Rule: clickable things should look clickable

Style

Don't put too much text in the HREF— it's confusing to look at once the line breaks. Just anchor around the key phrases.

Do name it what it is: check out the document archive, or surf over to our horrific Summer Vacation Photos.

Some regard it as poor style to use phrases like click here. Some use phrasing like "see our large complicated and difficult to explain with just a few words site here." I prefer to anchor the description itself if at all possible.

URL Characters

% Encoding

Whitespace and most other non `[a-zA-Z0-9]` characters are encoded in the URL as a `'%` followed by the character's ASCII code in hex

e.g. a space character within URL is encoded as `"%20"` since hex 20 = 32 decimal which is the ASCII code for a space.

Characters that should be % encoded if they are part of the text content of the URL

whitespace

< > " – used by HTML to delimit the URL. Always need to encode these.

% = ? & / # – have special meaning within URLs in common cases, and so should be encoded

; : \$, + @ – as above, but more rare

Some characters, such as ~, technically should be encoded (%7e), but web software has generally learned to process the ~ directly, and so the encoding is not required as a practical matter. This has the unfortunate effect of making the URL syntax definition a little imprecise at a practical level – must the ~ be encoded or not?

W3C addressing

<http://www.w3.org/Addressing/>

Good URLs don't change

Design your URL organization so that links will work for years...

<http://www.w3.org/Provider/Style/URI.html>

Page relative URL lookup

Suppose you are within the CS193i page who's full URL is...

<http://www.stanford.edu/class/cs193i/index.html>

The URL of an enclosing page is sometimes known as the "base" URL.

<http://www.stanford.edu/class/cs193i/> is the base URL.

URLs within the page are evaluated relative to its base URL. That means that if a URL is incomplete, the system will assume any missing components are the same as the base URL.

e.g. a url anchor in the index.html file reads: `Help Me!`

This means that the URL for the file is <http://www.stanford.edu/class/cs193i/help.html>

Effectively, this just means that URLs within a page are assumed to point to things which are "near" the referring page. Gives you the latitude to use short, relative URLs within a page.

Relative Example

Suppose we are writing HTML which is at <http://www.stanford.edu/class/cs193i/index.html>

The URL of the document where the `` is embedded is the "base" URL.

Relative

`Binky!`

The short, relative URL omits the protocol, host, and front part of the path of the base URL.

The browser expands the relative URL the full URL using the base URL...

<http://www.stanford.edu/class/cs193i/binky.html>

foo.html

So if you have a document called foo.html in the same directory as your home page, you can refer to it just as "foo.html" in your home page and count on relative lookup to find it.

If you want to link to foo.com, you can't just say "foo.com" since it looks like a relative URL to a file called "foo.com" – must use the full <http://foo.com/> form.

Extra For Experts

Relative addressing allows you to develop a group of interlinked documents in one place using relative references to refer to each other. Later you can move all the documents

somewhere else (another directory, another web server), and it all continues to work even though the document's full URLs are all different.

You can use "../foo.html" to go up a directory – this is a "fringe" use of HTML that few authors use (or understand). Unix users will understand this.

Root Relative

A relative url beginning with a '/' uses the protocol and host, but no part of the original path. So it essentially goes up to the document root of the same server.

/images/jeffsad.jpeg

e.g. refers to an images directory in the document root of the same server as the base URL.

The full URL would be http://www.stanford.edu/images/jeffsad.jpeg

Image Tag

Each image is a separate HTTP request – HTML page with 10 images = 11 requests total. One for the first page + 1 for each image.

HTTP 1.1 allows these multiple requests to use a single TCP connection, instead of making a new one for each request – we'll look at that later.

Include with=xxx height=xxx bindings in the tag, so the browser can lay out the page correctly before the image loads. Can also be used to scale the image on the browser side to fit that size.

Image Files

Image file formats – compression

JPEG

public standard – Joint Photographic Expert Group.

File Extension usually ".jpg" or ".jpeg"

Best for photos

24 bit color depth

Used by digital cameras, scanning,

Adjustable lossy compression – 10x or 20x compression. Does a very good job. At 10x, you can't even tell.

JPEG2000 revision is moderately better. Adoption has been slow, in part due to patent concerns. Hopefully, there will be a patent-free path for JPEG2000. (recall "submarine patents").

GIF

".gif"

Best for simple images (the older standard). Unisys charged for the use of the GIF compression scheme in source code, so there was a move afoot to ditch it. LZW compression scheme patent by Unisys expired June 2003, so it's free now. ☺

Transparency

Interlaced

Animated (ugh!)

PNG

Portable Network Graphics

An up-and-coming public standard replacement for GIF

Somewhat better features, and avoids the patent problems of GIF

SVG

Scalable vector graphics

Like PDF, but more standard and lightweight

NOT a bitmap – a series of drawing commands. 100x more compact than a pixel image + can support animation and other transforms driven by javascript.

Uses XML

Much smaller and better looking to distribute, say, a banner advertisement animation, as an SVG (drawing commands) than an animated gif (pixels)

Usually put image file in the same directory as the referring HTML and then use relative URLs.

Image Style

Images make a page fun and give it intuitive visual appeal.

However, slow load time is the #1 complaint of web users, and images load 100 times more slowly than text. Therefore, use text for main content, and sprinkle in a few images for fun decoration (that can fill-in after the main content has displayed).

e.g. yahoo.com, slashdot.org, nytimes.com

Amateurish sites tend to have a lot of images, since making images is fun (more fun than creating actual text content anyway).

Name Tags

`` marks a particular location in a document.

A URL can refer to that place in the document as `` or with a relative reference within the same document just ``.

The `<a>` tag may contain both an `href=` and a `name=`