CSC 346 - Cloud Computing

03 - Networking, HTTP & HTML

HTTP

Getting What We Ask For

HTTP: How Browsers & Servers Communicate

HTTP 1.1 - http://www.w3.org/Protocols/

- TCP Connection, usually over port 80 or 443
- Text Based Instructions
- Simple Verbs
  - GET, POST, PUT, DELETE, HEAD, CONNECT, OPTIONS, TRACE
- Optional Headers

HTTP

Basic GET Example

- HOST header is required for HTTP/1.1
- Two CRLF to indicate the request has finished
  - CRLF = \r\n Although most Web Servers will accept \n

```
GET / HTTP/1.1
Host: www.example.com
```

"Although the line terminator for the start-line and header fields is the sequence CRLF, a recipient MAY recognize a single LF as a line terminator and ignore any preceding CR."

http://tools.ietf.org/html/rfc7230#section-3.5

- Verbs and HTTP versions **are** Case Sensitive

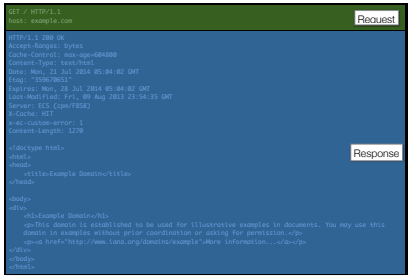
```
get / HTTP/1.1
host: example.com

HTTP/1.1 501 Not Implemented
```

```
get / http/1.1
host: example.com

HTTP/1.0 505 HTTP Version Not Supported
```

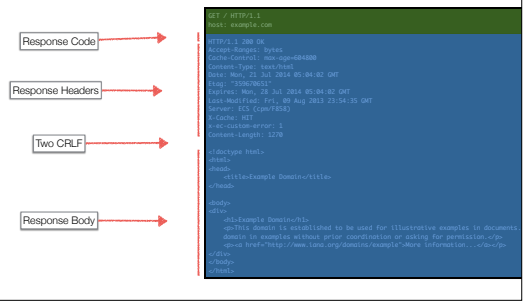
- Headers **are not** Case Sensitive

```
GET / HTTP/1.1
hoSt: exAMPle.cOm

HTTP/1.1 200 OK
```

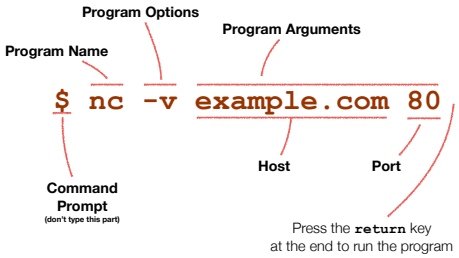
Basic HTTP Example



Basic HTTP Example

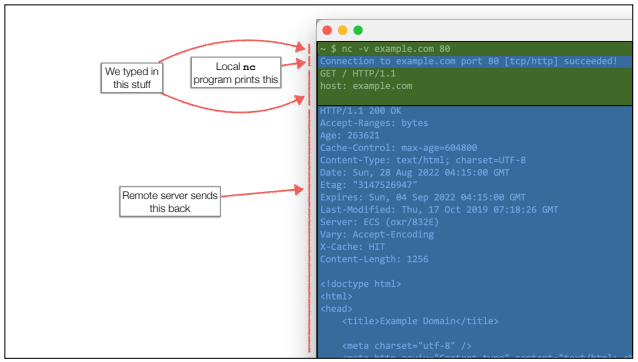


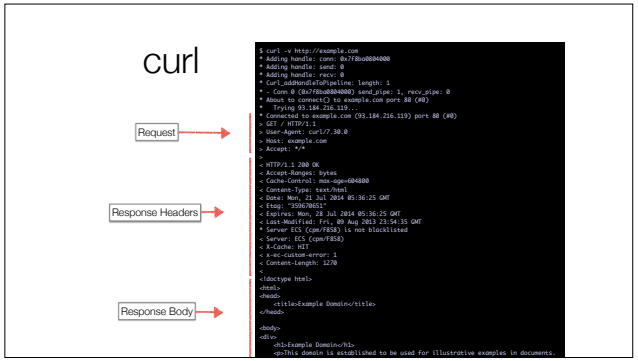
Command Line Basics



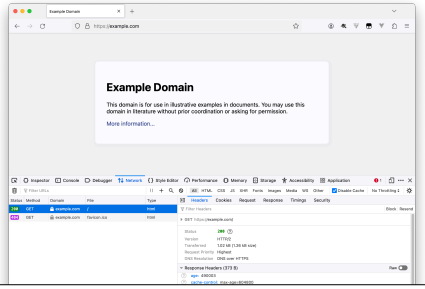
HTTP With NetCat - nc

- We used to do this with **`telnet`** but most environments no longer have this available by default
- Use nc (netcat) now instead
  - Opens a raw TCP socket connection to the target
- Key parts: **`host`** and **`port`**





Examine Requests in Firefox



Response Codes

1XX	Informational
2XX	Successful
200	OK
3XX	Redirection
301	Moved
4XX	Client Error
404	Not Found
5XX	Server Error
500	Internal Server Error

http://tools.ietf.org/html/rfc7231#page-4

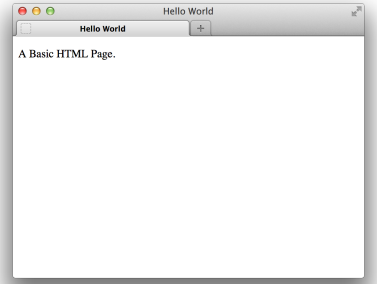
HTTP/2.0

- New binary method of allowing multiple requests through a single TCP socket
- More of a change to how the protocol is implemented on the wire than in the concepts of how the protocol works
- Advanced topic, if you're interested in more details:
  - http://http2-explained.readthedocs.org/en/latest/src/http2protocol.html
- Otherwise, just know its a thing

Some HTML

/some.html

```
<!doctype html>
<html>
<head>
  <title>Hello World</title>
</head>

<body>
  <p>A Basic HTML Page.</p>
</body>

</html>
```



HTML Defines Content and Structure

- Content consists of Text, Images, Links, Media Assets, etc
- Structure defines the basic formatting and semantic meaning of elements
  - i.e. `<title>Hello World</title>` defines the title of the page
  - Programs can analyze the structure of a document to derive meaning
    - `h1`, `h2`, `h3` tags could be used to generate a document outline
    - Headers in a table (`<th>`) could be used by screen readers to describe data to a visually impaired individual
  - We can use the document structure to define display styles

Structure of an Element

```
<title>Hello World</title>
```

- The entire line is referred to as ***the*** `title` ***element***
- The ***name*** of this element is “title”
- `<title>` is an ***opening tag***
- `</title>` is a ***closing tag***
- `Hello World` is the ***content*** of this element

Not All Elements Need a Closing Tag

```
<body>
  <p>
   Paragraph elements can have closing tags
  </p>
  <p>or not
  <ul>
    <li>List Item elements
    <li>may also omit closing tags
  </ul>
</body>
```

http://www.w3.org/TR/html5/syntax.html#optional-tags

Not All Elements Have Content

- `<br>` the Break tag acts as a newline character for HTML
- `<hr>` the Horizontal Rule tag draws a line across a page
- `<img src="foo.gif">` the Image Tag tells the browser to go load an image in this location
- These elements are called ***void elements*** and *must not have* closing tags

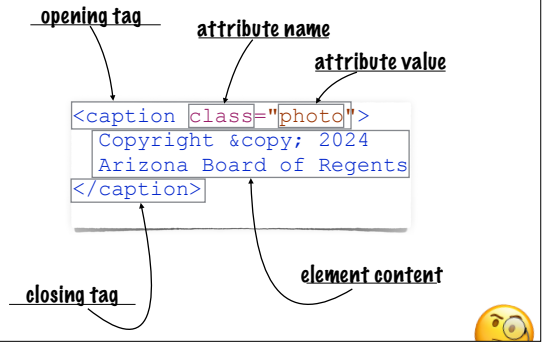
http://www.w3.org/TR/html5/syntax.html#void-elements

Attributes

```
<img src="foo.gif" class=thumbnail>
```

- Attributes for an element are defined in the element's ***opening tag***
- Attributes always have an ***attribute name***
- Attributes may optionally have a ***value***
- Attribute values may be surrounded with either single quotes, double quotes, or nothing, depending on the content of the value

http://www.w3.org/TR/html5/syntax.html#attributes-0



<!doctype …>

- The `<!doctype …>` preamble is *not* an HTML element.
- `<!doctype …>` tells the rendering engine what type of markup to expect
- HTML4.1 Transitional
  - `<!doctype html public "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">`
- HTML5
  - `<!doctype html>`

DOCTYPEs are required for legacy reasons. When omitted, browsers tend to use a different rendering mode that is incompatible with some specifications. Including the DOCTYPE in a document ensures that the browser makes a best-effort attempt at following the relevant specifications.

http://www.w3.org/TR/html5/syntax.html#the-doctype

<html>

- The `<html>` element is the root element of our element tree
- The HTML Element can only be preceded by whitespace characters and comments
- The HTML Element can only have two children: one `<head>` element and one `<body>` element
- From the HTML specification:
  - *An html element's start tag can be omitted if the first thing inside the html element is not a comment.*
  - *An html element's end tag can be omitted if the html element is not immediately followed by a comment.*

<head>

- The <head> element represents a collection of metadata for the Document.
- A <title> tag is the only required child element

```
<head>
  <meta charset="utf-8">
  <base href="http://www.example.com/">
  <title>A New Hope</title>
  <link rel="stylesheet" href="default.css">
  <script src="example.js"></script>
</head>
```

<body>

- The <body> element represents the content of the Document.
- Basically this holds everything you see.

```
<body>
  <img src="http://goo.gl/5LEYnp" alt="regrets">
  <form action="search.php" method="post">
    <input type="text" name="search">
    <input type="submit" value="Find Droids">
  </form>
</body>
```

Links

```
Come visit the <a href="http://www.arizona.edu">University of Arizona</a> campus.
```

- <a> Anchor tag
- Used to define a link to another document, or location in the same document.

Links

```
<a href="http://www.arizona.edu">University of Arizona</a>
```

- `href` attribute defines what to link do.
  - This is the *Hyper* in HyperText
- Must contain a valid URL
  - Universal Resource Locator

URL

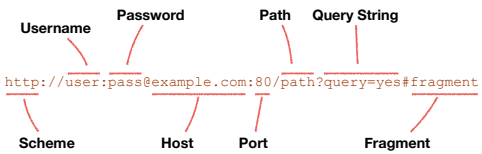
- A basic absolute URL

```
http://www.arizona.edu
```

- A basic relative URL

```
../images/image.png
```

URL



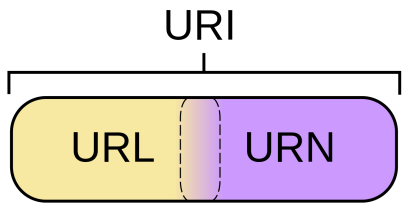
URL

- Most of these parts are null most of the time
- The following are all valid URLs

```
https://example.com
/path/to/something.html
mailto:fischerm@email.arizona.edu
foo
//ajax.googleapis.com/libs/jquery.min.js
../somepage.php?key=123
anotherpage.html#figure1
#droids
```

URI, URL, URN

- URI - Universal Resource Identifier
- URL - Universal Resource Locator
- URN - Universal Resource Name
- These are NOT interchangeable. Each has a different meaning, although there can be significant overlap
- We're almost always going to use URLs unless otherwise explicitly mentioned



http://en.wikipedia.org/wiki/File:URI_Euler_Diagram_no_lone_URIs.svg

URI

```
The generic URI syntax consists of a hierarchical sequence of
components referred to as the scheme, authority, path, query, and
fragment.

   URI         = scheme ":" hier-part [ "?" query ] [ "#" fragment ]

   hier-part   = "//" authority path-abempty
               / path-absolute
               / path-rootless
               / path-empty
```

http://tools.ietf.org/html/rfc3305

http://tools.ietf.org/html/std66

URL Schemes

`http://user:pass@example.c`

Scheme

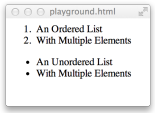
- The Scheme tells the client how to access the resource.
- `file:///` loads the file directly from the local filesystem
- `http://` initiates an HTTP connection over TPC/IP
- `https://` establishes a secure connection over SSL, then communicates via HTTP
- `email:` hands off control to an email client
- `tel:` hands off control to a phone client
- `myapp:` Mobile platforms let you register a URL Scheme for your app

More Elements

Ordered and Unordered Lists

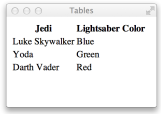
- `<ol>` Ordered List
- `<ul>` Unordered List
- `<li>` List Element - Used for both types of lists
- Closing Tag for `<li>` may be omitted

```
<ol>
  <li>An Ordered List</li>
  <li>With Multiple Elements</li>
</ol>

<ul>
  <li>An Unordered List
  <li>With Multiple Elements
</ul>
```



Tables

- `<table>` begins a table
- `<th>` table header
- `<tr>` table row
- `<td>` table data




```
<table>
  <tr>
    <th>Jedi</th>
    <th>Lightsaber Color</th>
  </tr>
  <tr>
    <td>Luke Skywalker</td>
    <td>Blue</td>
  </tr>
  <tr>
    <td>Yoda</td>
    <td>Green</td>
  </tr>
  <tr>
    <td>Darth Vader</td>
    <td>Red</td>
  </tr>
</table>
```

Headings

- `<h1>` 1st level heading - Biggest
- `<h6>` 6th level heading - Smallest
- `<h1> <h2> <h3>`
  `<h4> <h5> <h6>`

Images

```
<figure>
  <img src="img/droid.jpg" alt="A Droid">
  <figcaption>
    https://www.flickr.com/photos/dunechaser/6987810377
  </figcaption>
</figure>
```

- Something other than text!
- The img tag is a void element, so it has no closing tag
- By default images are displayed at their native pixel size

Images

- Images can be resized with CSS, or with `width` and `height` attributes.
- Resized images are not resampled. The full image is sent to the browser no matter what size the image is ultimately displayed at.
- Assigning just `width` or `height` will scale the image and preserve the aspect ratio. (`width:height`)

Images

- The `alt` attribute should always be present, and should describe the image as best you can.
- Accessibility should be thought about from the very start of an HTML project, and not at the very end.
- If an image provides no useful information (a spacer image, or background gradient) an empty `alt` attribute should be used: `alt=""`

Images

- Three widely supported Image formats
  - GIF - Graphics Interchange Format
  - JPEG - Joint Photographic Experts Group
  - PNG - Portable Network Graphics
- HTML Specification does not mandate support for any particular format

GIF

- 256 distinct colors. Each GIF can have its own color pallet.
- One color can be designated as transparent.
- Can contain multiple frames for animation.
- Lossless compression, but limited format.

JPEG

- Millions of colors
- Lossy compression
  - Higher quality, less compression, larger file size
  - Smaller file size, higher compression, less quality
- Designed to be good at compressing photographs.
- No transparency

PNG

- Lossless compression
- No animation
- Several bit depth variants
  - PNG-8: 256 colors
  - PNG-24: 16 Million colors (3 8-bit channels)
  - PNG-32: 16 Million colors + 8-bit transparency
    - Allows for smooth anti-aliased transparency

WebP

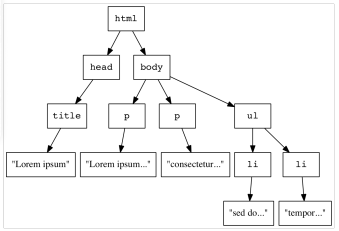
- Lossless *or* lossy compression
- Animation
- Wide variety of bit-depths
- Supports Transparency (alpha channel)
- Good support for recent browsers (2020 on)

Images

	GIF	JPEG	PNG
Photograph		✔	
Animated	✔		
Icon or Drawing	✔		✔
Transparency	✔		✔

DOM Tree

```
<!doctype html>
<head>
  <title>Lorem Ipsum</title>
</head>

<body>
  <p>
   Lorem ipsum dolor sit amet
  </p>
  <p>consectetur adipisicing elit
  <ul>
    <li>sed do eiusmod tempor incididunt
    <li>tempor incididunt
  </ul>
</body>
</html>
```



Misc Details

- HTML Tags and attribute names are **not case sensitive**
- Comments: `<!-- ••• -->`
  - Cannot nest comments. No inline comments
- Whitespace is mostly ignored. Multiple whitespace characters are condensed to a single space when rendered
- Text nodes and attribute values must be a tab, newline, form-feed, carriage-return or unicode characters ≥ than U+0020 (space)

next up: Networking Sockets