ITMD-361
WEEK 2
JANUARY 17, 2018

TONIGHT'S AGENDA

- **Discuss the Software and Development Tools you use**
- **My first website**
- **The browser as a platform**
- **HTTP Requests**
- **Introduction to the technologies**
- **Intro to HTML**

ITMD-361

SOFTWARE AND DEVELOPMENT TOOLS

COMMON SOFTWARE

Text Editor

- Notepad++ (win)
- Textpad (win)
- TextWrangler (mac)
- Sublime Text 3 (multi)
- Brackets (multi)
- Atom (mac)
- Many more…
- http://en.wikipedia.org/wiki/Comparison_of_text_editors

SFTP

- WinSCP (win)
- Filezilla (multi)
- Cyberduck (multi)
- Transmit (mac)
- Browser Extensions

SSH

- Terminal (multi)
- PuTTY (win)

BROWSER AND DEVELOPMENT TOOLS

For class, please use a modern developer friendly browser to develop and test your web pages. These browsers have built-in tools for development and extensions available to add additional features.

- **Mozilla Firefox (https://www.mozilla.org/)**
  - Developer Edition (Formerly Aurora Channel)
- **Google Chrome (https://www.google.com/chrome)**
- **Apple Safari (Mac only)**
- **Microsoft Edge (Microsoft only)**

BROWSER AND DEVELOPMENT TOOLS

Mozilla Firefox

- **Can run stable release and developer version together**
- **Native Dev tools – inspector, debugger, and more**
  - https://developer.mozilla.org/docs/Tools
- **Firebug extension – inspector, debugger, and more**
- **Web developer toolbar extension**

Google Chrome

- **Can run stable release and canary version together**
- **Native Dev tools – inspector, debugger, and more**
  - https://developer.chrome.com/devtools
- **Free Code School course to learn Chrome Dev Tools**
  - http://discover-devtools.codeschool.com/
- **Firebug lite extension – if you really like firebug**

BROWSER AND DEVELOPMENT TOOLS

Apple Safari

- **Native Dev tools – inspector, debugger, and more**
  - https://developer.apple.com/library/safari/documentation/AppleApplications/Conceptual/Safari_Developer_Guide/Introduction/Introduction.html
- **Must enable developer menu in preferences -> advanced**
- **Tools for inspecting pages on iPhone and iPad connected to the Mac via USB.**

Microsoft Edge

- **Native Dev tools (F12) – inspector, debugger, and more**
  - https://developer.microsoft.com/en-us/microsoft-edge/platform/documentation/f12-devtools-guide/
- **Recent version is much better than older ones**
- **Allows you to render page as it would look in older versions of IE with limits (no conditional comments, ie10+)**

My First Website

Setting up Github

ITMD-361

INTRODUCTION TO THE TECHNOLOGIES

INTRODUCTION TO THE TECHNOLOGIES

Meet the “Client Side Technologies”

- **HTML: Structure**
- **CSS: Presentation**
- **JavaScript: Behavior**

What about “Server Side” and “Backend” code?

- **Can be almost any language including but not limited to: PHP, Ruby, Python, Java, C#, Visual Basic, C, C++, Perl, JavaScript, and more…**
- **http://en.wikipedia.org/wiki/Server-side_scripting**

HTML

- **HTML – HyperText Markup Language**
- **HTML was invented in the early 90s - created by Tim Berners-Lee**
- **HTML is the CONTENT layer of your webpage / website. HTML provides the STRUCTURE and SEMANTIC meaning of your content**
- **Your HTML is how Google, Google spiders, and other search bots, see your page, so make it good.**
- **We are on HTML version 5: specifications**
  - Published as a W3C recommendation in October 2014

HTML

Major Version History

- **HTML Tags – Informal CERN document with 18 tags, October 1991**
- **HTML 1.1 – First draft with a version number, November 1992**
- **HTML 2.0 – Published as IETF RFC 1866, November 1995**
- **HTML 3.2 – Published as W3C recommendation, January 1997**
- **HTML 4.0 – Published as W3C recommendation, December 1997 and reissued in April 1998 with minor edits**
- **HTML 4.1 – Published as W3C recommendation, December 1999**
    - XHTML 1.0 – Published as W3C recommendation, January 2000 and reviesed in April 2002
    - XHTML 1.1 – Published as W3C recommendation, May 2001 and based on XHTML 1.0 Strict
    - XHTML 2.0 – Was a working draft but abandoned in 2009 in favor of HTML5
- **HTML 5 – Published as W3C recommendation, October 28, 2014**

CSS

- **CSS – Cascading Style Sheets**
- **CSS is the PRESENTATION layer of your website / web page**
  - DECORATING your html tags
- **CSS allows you to make style rules to govern position, colors, fonts, images, and other visual formatting to your HTML tags**
- **Currently using CSS 2.1 and CSS 3**

CSS

Version History

- **CSS 1 – Not the standard: competed with others: 1996**
- **CSS 2 – Published as a recommendation May 1998**
- **CSS 2.1 – Published as a recommendation June 2011**
    - Single large specification
- **CSS 3 – Current standard**
    - Full specification is split up into modules
    - Started being worked on when CSS 2 was published
    - Earliest CSS 3 drafts were June 1999
    - Four modules published as official recommendations
        - Media Queries, Selectors Level 3, Namespaces, Color
- **http://www.w3.org/Style/CSS/specs**

JAVASCRIPT

1. JavaScript – ECMA compliant Object Oriented language
2. JavaScript or JS is the BEHAVIOR layer of your web page or website.
3. JavaScript is a programming language.
4. It is capable of the following
   - Moving things around the screen
   - Manipulating HTML tags and CSS rules
   - Implementing cool browser APIs like location and local storage.
   - And much more...

JAVASCRIPT

- Originally developed by Brendan Eich at Netscape
- Developed with code name Mocha, then officially called LiveScript in beta Netscape Navigator 2 in Sept 1995.
- Renamed JavaScript before final release of Netscape Navigator 2 (B3). Marketing ploy by Netscape to cash in on the popularity of the new language Java which was also included in Netscape 2.
- November 1996 – Netscape submitted JavaScript to Ecma International to be considered a standard.
- June 1997 – Ecma International published the ECMA-262 Specification in version 1.
- Current version of the ECMAScript standard is 7 and was published in June 2016
- Current version of JavaScript is 1.8.5 and was published July 2010.

SERVERS

- **The server is just a powerful computer with a lot of processors and a lot of RAM**
- **The server listens for HTTP requests and serves responses to clients when requested**
- **To set up a server, we must have 3 things**
  - Static IP Address / constant internet connection
  - Domain Name mapped to the static IP address
  - Apache, IIS, or other web server software running and configured with the IP address and Domain name information

SERVER SIDE/BACKEND CODE

- Backend technology allows you to use various programming languages on the server. Server side code is processed by code execution processes on the server, and may integrate with a database and the file system of the server.
- Server side code can react to HTTP GET requests, HTTP POST information as a result of a client filling out a form, or any of the other HTTP methods
- Server side code reacts to method requests, runs some code and talks to databases, and returns HTML, CSS, and JavaScript
- We do not discuss these technologies in detail in this class. Future classes will discuss these concepts.

THE BROWSER AS A PLATFORM

THE BROWSER IS THE PLATFORM

What is a browser?

THE BROWSER IS THE PLATFORM

Browsers are compiled programs written in one or more programming languages to a specific operating system, API, and computer architecture.

The browser is responsible for the following:

- **Works with the computer's networking card to package up, send, and receive HTTP and other commands.**
- **Reading, parsing, and rendering HTML, CSS, & Javascript**
- **Providing the user with an interface to use the HTTP protocol and domain name services (DNS)**

THE BROWSER IS THE PLATFORM

The browser implements the following protocols and technologies:

- **HTTP: HyperText Transfer Protocol**
- **HTTPS: Secure HyperText Transfer Protocol**
- **FTP: File Transfer Protocol**
- **DNS: Domain Name Service - Translates URLs addresses to IPs**
  - URL: (Uniform Resource Locator)
  - URI: (Uniform Resource Identifier)
- **The HTML (HyperText Markup Language) specification**
  - http://www.w3.org/TR/html5/
- **The JavaScript (ECMA-262 Script) specification**
  - http://www.ecma-international.org/publications/standards/Ecma-262.htm
- **The CSS (Cascading Style Sheet) specification**
  - http://www.w3.org/Style/CSS/specs.en.html

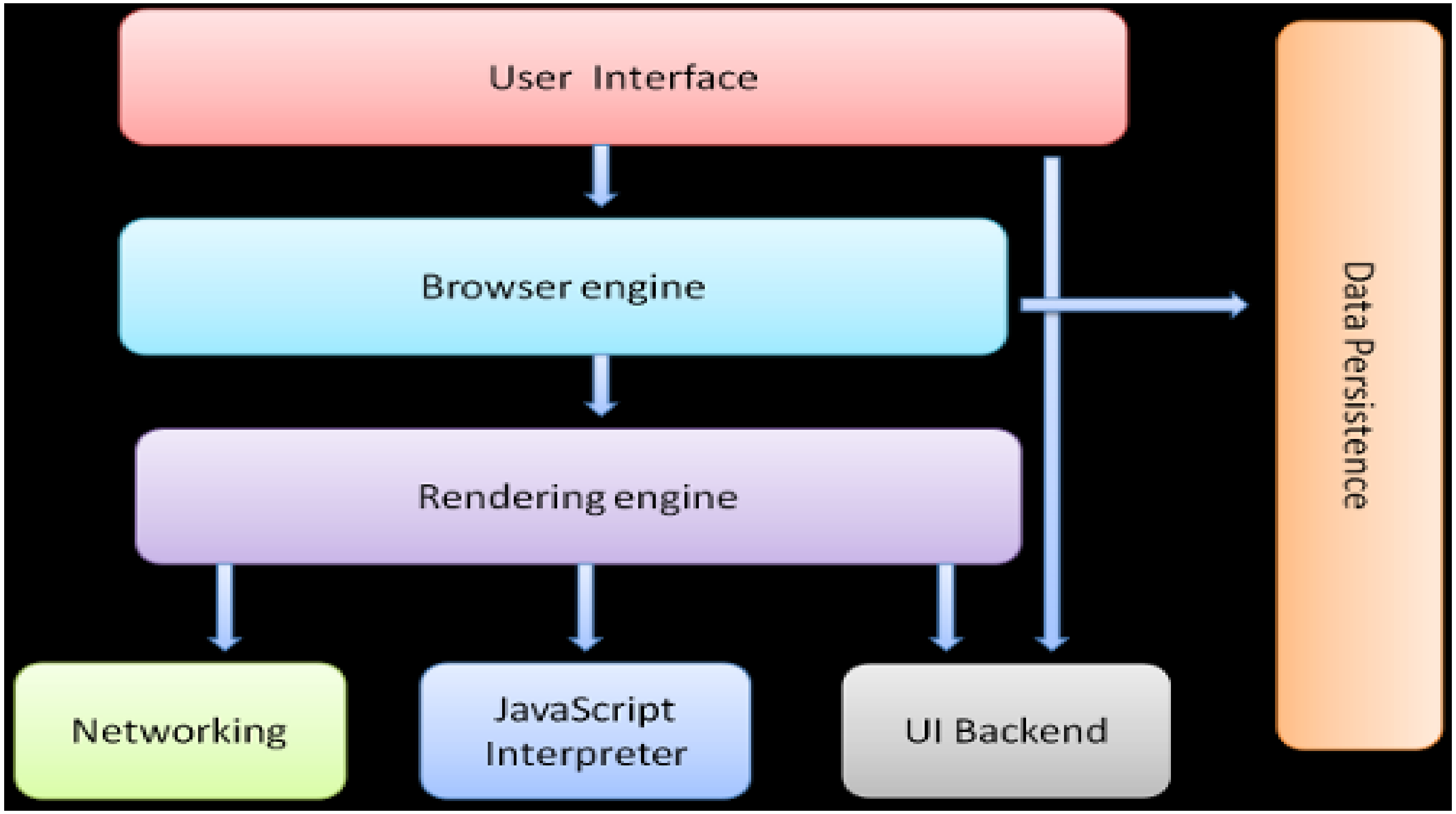
THE BROWSER COMPONENTS

In most cases, a modern browser is designed as a combination of multiple components.

- **The User Interface**
- **The Browser Engine**
- **The Rendering Engine**
- **Networking**
- **UI Backend**
- **JavaScript Interpreter**
- **Data Persistence**

http://www.html5rocks.com/en/tutorials/internals/howbrowserswork/

THE BROWSER COMPONENTS



http://www.html5rocks.com/en/tutorials/internals/howbrowserswork/

THE BROWSER COMPONENTS

- **The User Interface**
  - Address bar, back/forward buttons and all other parts of the browser display except the main content area.
- **The Browser Engine**
  - Controls the interaction of the UI and rendering engine
- **The Rendering (Layout) Engine**
  - Responsible for parsing and displaying the content
- **Networking**
  - Used for networking calls like HTTP. Uses a platform independent interface with a platform specific implementation underneath.

THE BROWSER COMPONENTS

- **UI Backend**
  - Used for drawing basic widgets like combo boxes and windows. This backend exposes a generic interface that is not platform specific. Underneath it uses operating system user interface methods.
- **JavaScript Interpreter**
  - Parses and Executes JavaScript
- **Data Storage**
  - A persistence layer. The browser needs to save various things to the system hard disk, some examples are cookies, cache items, bookmarks. HTML5 spec defines a 'Indexed Database' which is a lightweight database in the browser and 'Web Storage' which is a simple key value store.
  - http://www.html5rocks.com/en/features/storage

RENDERING/LAYOUT ENGINE

The rendering (layout) engine parses the marked up content (HTML) and formatting information (CSS) and displays it on the screen.

Each browser has a different rendering engine

- **Firefox – Gecko**
- **Safari – Webkit (forked from KHTML)**
- **Chrome – Webkit (Version < 27), Blink (Version > 28)**
- **Edge – EdgeHTML (forked from old Trident)**
- **Opera – Presto (Version < 14), Blink (Version > 15)**
- **Samsung - Blink**
- **Amazon Silk – Blink**

A description of browser parsing and rendering of content is available at this link.

- **http://www.html5rocks.com/en/tutorials/internals/howbrowserswork/**

JAVASCRIPT ENGINE

- **JavaScript engine parses, interprets, and executes JavaScript (or ECMAScript).**
- **"The Browser Wars 2.0"**
  - 1997 Netscape = 79%
  - 2002 IE = 96%
  - 2017 Chrome = 43.19
- **New generation engines implement just-in-time compilation (JIT)**
  - Program code is stored in memory
  - Code can be compiled to native machine code just before it is executed
  - Provides high-speed execution of interpreted or byte code

JAVASCRIPT ENGINE

JavaScript Engines implementing JIT variations

- **Firefox – SpiderMonkey → IonMonkey and JägerMonkey compilers**
- **Chrome, node.js – V8**
- **Safari – SquirrelFish also known as Nitro**
- **Opera – Carakan (version 10.5 – 14), V8 (version 15+)**
- **Edge – Chakra**
- **http://en.wikipedia.org/wiki/List_of_ECMAScript_engines**

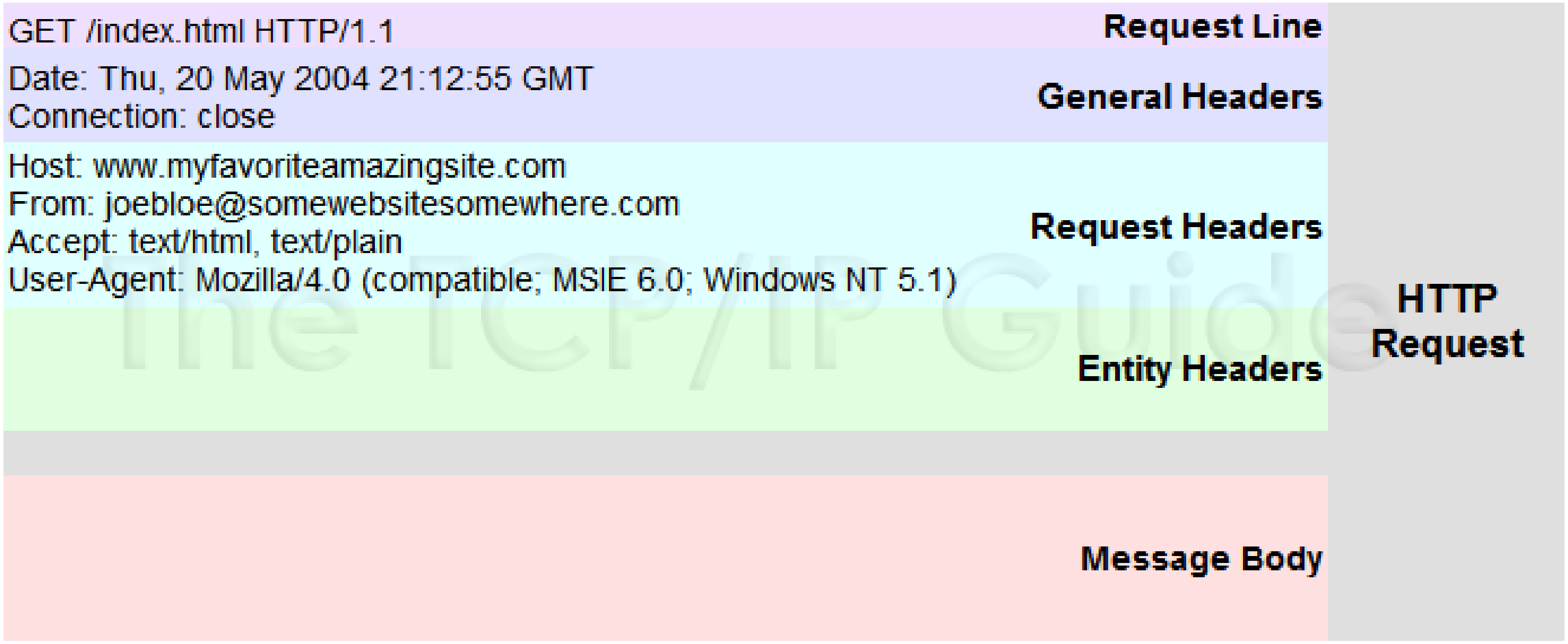
IMTD-361

HTTP Requests

HTTP Request

- Client Parses the URI
  - protocol://server/request
- Client sends request to Server
  - Usually HTTP protocol
  - [METH] [REQUEST-URI] HTTP/[VER]
  - [fieldname1]: [field-value]
  - ...
  - [request body, if any (used for POST and PUT)]
- Example - GET / HTTP/1.1

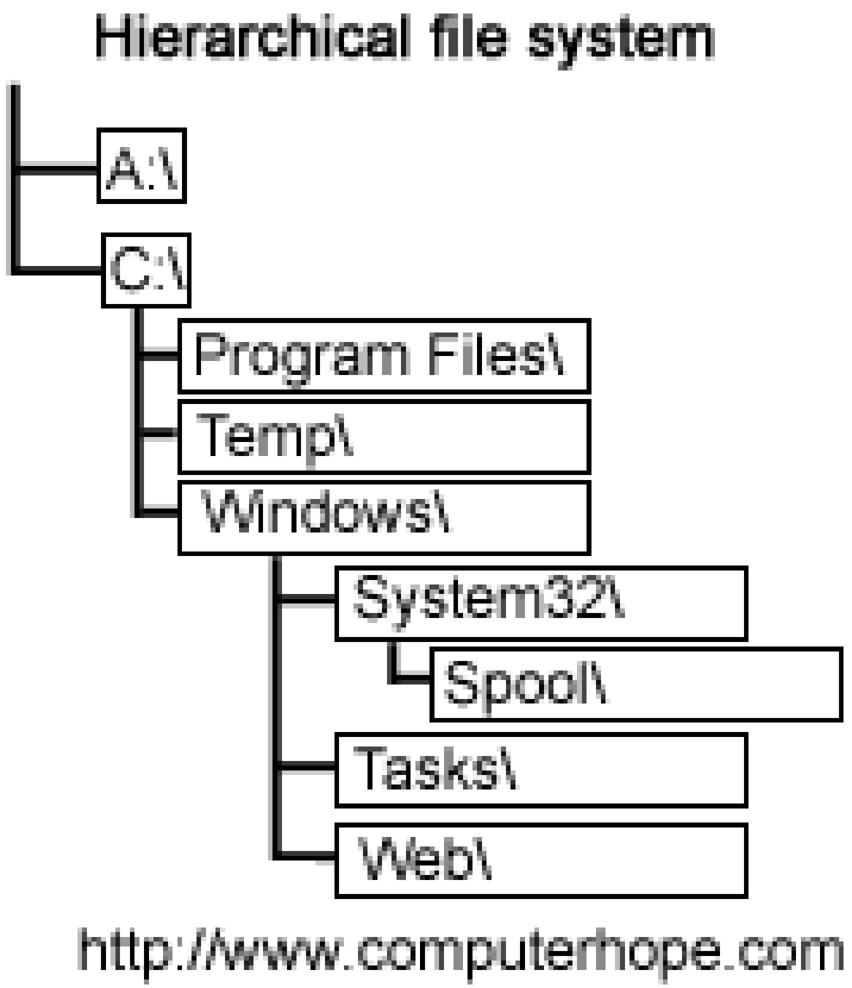
HTTP Request



```
GET /index.html HTTP/1.1
Date: Thu, 20 May 2004 21:12:55 GMT
Connection: close
Host: www.myfavoriteamazingsite.com
From: joebloe@somewebsitesomewhere.com
Accept: text/html, text/plain
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
```

The TCP/IP Guide

http://www.tcpipguide.com/free/t_HTTPRequestMessageFormat.htm


Hierarchical file system
A:\
C:\
Program Files\
Temp\
Windows\
System32\
Spool\
Tasks\
Web\
http://www.computerhope.com

HTTP Request

- HTTP Methods
  - GET, POST, PUT, DELETE, HEAD, TRACE, CONNECT
  - First 4 are the common ones. Mostly GET.
  - http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html
- GET
  - Most common, Basically get me this document
  - Any variable or form data is sent as part of the URL
  - http://www.domain.com/?q=232&name=joe
  - Data q=232 and name=joe is available to target page

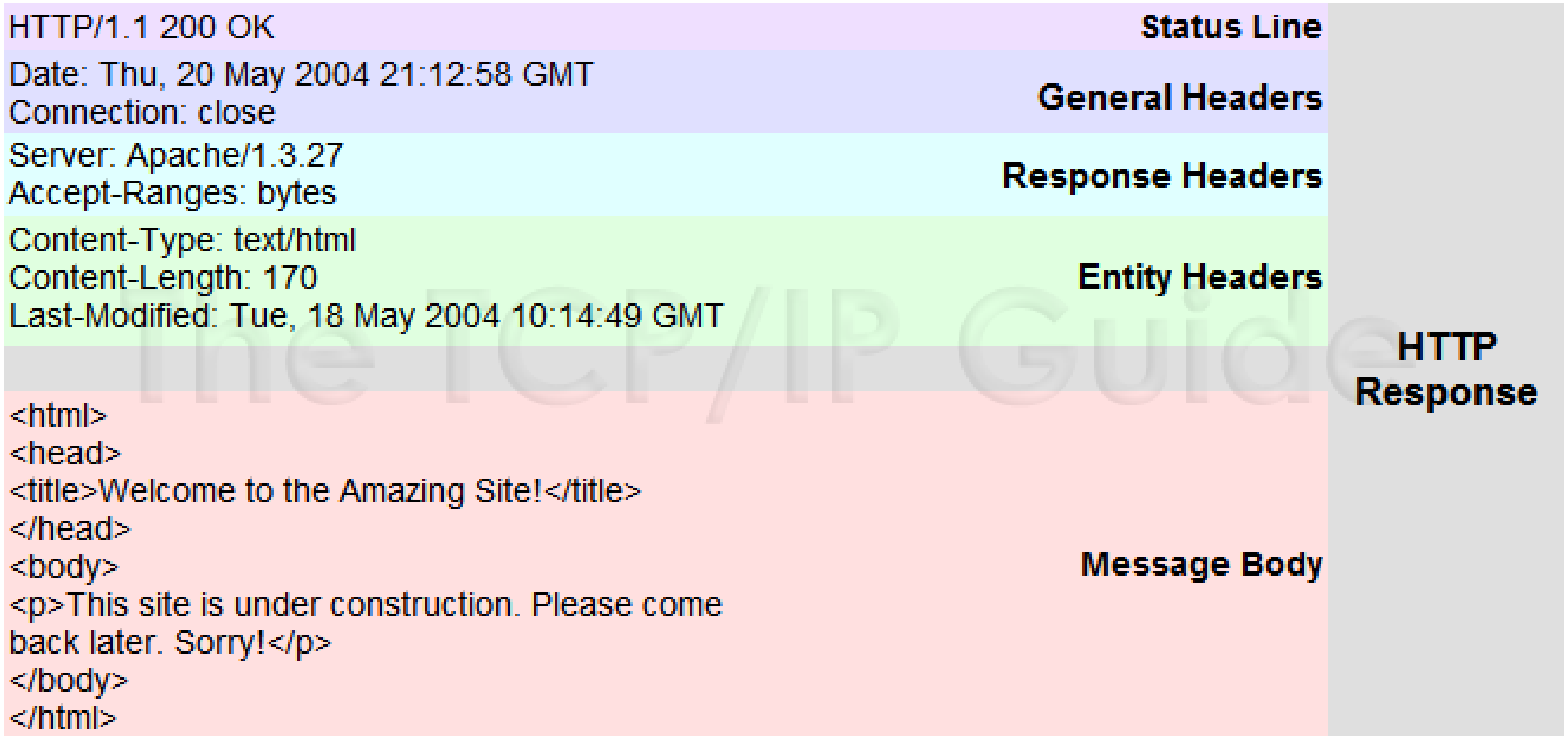
HTTP Request

- POST
  - Second most common method
  - Used often to send form data
  - Any variable or form data is sent in the request body and not appended to the URL
- PUT & DELETE
  - Used mostly with web programming frameworks
  - Used in Ruby on Rails
- HEAD: Returns only the Response headers

HTTP Response

- Server sends response to client
  - Usually HTTP Protocol
  - HTTP/[ver] code text
  - [fieldname1]: [field-value]
  - ...
  - [response body]
- First line is status of request
- Then multiple header fields can follow
- Lastly the response body follows

HTTP Response



```
HTTP/1.1 200 OK
Date: Thu, 20 May 2004 21:12:58 GMT
Connection: close
Server: Apache/1.3.27
Accept-Ranges: bytes
Content-Type: text/html
Content-Length: 170
Last-Modified: Tue, 18 May 2004 10:14:49 GMT

<html>
<head>
<title>Welcome to the Amazing Site!</title>
</head>
<body>
<p>This site is under construction. Please come
back later. Sorry!</p>
</body>
</html>
```

http://www.tcpipguide.com/free/t_HTTPResponseMessageFormat.htm

HTTP Response

- Status Codes
  - 3 digit numbers grouped into 5 groups by first digit
- 1xx – Informational
  - No 1xx status codes are defined, expermental
- 2xx – Successful
  - 200 OK – Server did request and all is well
  - Rest of the 200's are mostly used for scripting, not commonly seen

- Status Codes continued
- 3xx – Redirection
  - 301 Moved permanently
    - The resource is somewhere else and links and references should be updated
  - 302 Moved temporarily
    - Means same as 301 but links and references should not be updated since it may move again in the future
  - 304 Not modified
    - Returned if the if-modified-since header used
    - Basically means cached version should be displayed

HTTP Response

- Status Codes continued
- 4xx Client error
  - 400 Bad request – Incorrect request syntax
  - 401 Unauthorized
    - Client not allowed access to resource
    - May change if client retries with authorization header
  - 403 Forbidden
    - Client not allowed access to resource
    - Authorization header will not help
  - 404 Not found – Dead link

HTTP Response

- Status Codes continued
- 5xx Server error
  - 500 Internal server error
    - Something went wrong inside the server
  - 501 Not implements
    - The request is not supported by the server
  - 503 Service unavailable
    - Usually happens when a server is overloaded
- http://www.w3.org/Protocols/rfc2616/rfc2616-sec6.html

HTTP Response

- Response headers can include
  - Location
  - Server
  - Content-length
  - Content-type
  - Content-encoding
  - Expires
  - Last-modified
  - And others

HTTP 1.1

- HTTP/1.0
  - http://www.w3.org/Protocols/HTTP/1.0/spec.html
- HTTP/1.1 - 1999
  - http://www.w3.org/Protocols/rfc2616/rfc2616.html
- 1.0 vs 1.1
  - 1.0 only had GET, POST, HEAD Methods
  - 1.1 requires host header
  - 1.1 adds some cacheing and persistence and more
  - http://www2.research.att.com/~bala/papers/h0vh1.html

HTTP 2.0

- HTTP/2.0 is the next planned version
- Based on SPDY
  - http://en.wikipedia.org/wiki/SPDY
  - Effort by Google to speed up http protocol with things like compressing and multiplexing
  - Supported in some modern browsers now
- Still in development

ITMD-361

INTRO TO HTML

INTRO TO HTML

- HTML is a Markup Language, it is not a programming language
- HTML is PARSED
- HTML needs valid structure to be parsed accurately
- if you lose the formatting in Microsoft word doc it does not know how to display the page. If your markup is invalid the browser does not know how to display the page.
- HTML is used to structure content on a webpage and provide SEMANTIC meaning of your content

INTRO TO HTML

- **Files should end with .html extension, .htm is a common alternative but I suggest .html**
- **Do not use spaces in a file name or any special chars**
  - Common to use _ or – instead of space
- **Limit filename to letters, numbers, underscores, hyphens and periods.**
- **Filenames may be case-sensitive on some OS's**
- **Shorter filenames are better**
- **All lower case letters with hyphens separating words is a common convention**
- **The browser will ignore extra whitespace, line breaks, unrecognized markup and comments**

INTRO TO HTML

HTML is comprised of text and tags. Tags follow a specific pattern and must be formatted correctly.

HTML vs XHTML

- **Previous versions were HTML 4.01 and XHTML 1**
- **Now we will be using HTML5 influenced by some XHTML syntax**
    - All tags should be lowercase
    - All tags should be opened and closed properly
    - All attributes in tags should be properly quoted
    - All tags should be properly nested

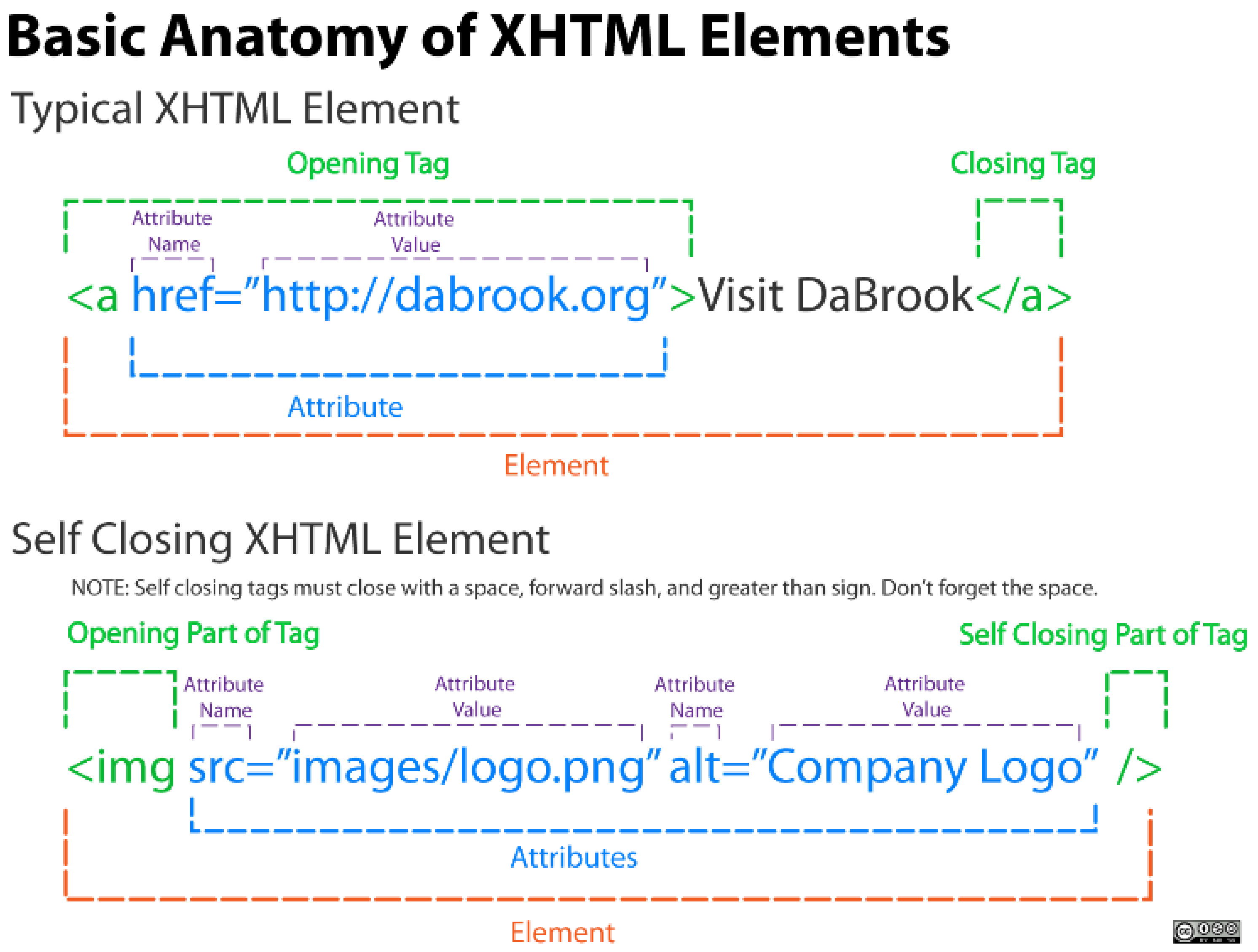
ANATOMY OF AN HTML TAG

<a href="http://www.google.com" title="A Link">Google</a>

- The < is the OPENING angle bracket. The angle bracket tells the browser that markup is starting
- “a” is the ELEMENT. All HTML tags are comprised of elements and attributes.
- href="" is an ATTRIBUTE. HTML tags can have one or more attributes applied to them. Attributes follow name/value pair convention. href is the NAME, http://www.google.com is the VALUE
- “Google” is the TEXT NODE VALUE of the html tag, this is the text a user will see
- </a> is the CLOSING TAG. It repeats the element with a forward slash before it

Do not use the typographic curly quotes “ ”, use straight " " quotes instead. Curly quotes will break your markup. Biggest culprit is not using a plain text editor or coping and pasting carelessly.

ANATOMY OF AN HTML TAG


Basic Anatomy of XHTML Elements
Typical XHTML Element
Opening Tag
Closing Tag
Attribute Name
Attribute Value
<a href="http://dabrook.org">Visit DaBrook</a>
Attribute
Element
Self Closing XHTML Element
NOTE: Self closing tags must close with a space, forward slash, and greater than sign. Don't forget the space.
Opening Part of Tag
Self Closing Part of Tag
Attribute Name
Attribute Value
Attribute Name
Attribute Value
<img src="images/logo.png" alt="Company Logo" />
Attributes
Element

BASIC HTML PAGE SKELETON STRUCTURE

There are 4 main components to a proper HTML5 skeleton structure.

- A doctype declaration – to let the browser know the type of document
- A html root element
- A head section with information describing the page
- A body section for the part of the web page the user will see. This contains all the content.

All pages should have this basic structure. All assignments should follow this structure.

HTML5 SKELETON PAGE STRUCTURE

```
<!DOCTYPE html>
<html>

  <head>
    <meta charset="utf-8">
    <title>Title Here</title>
  </head>

  <body>
  Page content here
  </body>

</html>
```

BASIC TAG OVERVIEW

The following tags are the most basic HTML tags and you should memorize them and their major attributes.

- <h1>Heading</h1> thru <h6>Heading</h6>
- <img src="imagename.jpg” alt="" >
- <a href="page.html">contact us</a>
- <p>Here is a paragraph</p>
- <div>Here is a division or container</div>
- This is <span>a span</span> in a sentence.
- <strong>Strong Importance</strong>
- <em>Emphasized text</em>

BASIC TAG OVERVIEW

BASIC HTML5 SKELETON FILE

Always start here: Link