# HTML Cont'd

CS 190 Version 3.0, Spring 2013

22 January 2013

# Lecture Plan

- Recap from last lab:
  - Unix directory navigation, file permission
  - Set proper permissions for your web pages
  - Webpage: “Hello world!”
- Quick overview of HTTP protocol and HTML
- The **HyperText Markup Language** (HTML)
  - **Markup**: Formatting pages
  - **HyperText:** Links between pages
  - **Language**: has (simple) syntax, semantics
- First assignment (individual)

# Key Objectives for Today

- Get more comfortable with MathCS environment
  - Review Unix filesystem, permissions
  - Working from home (resources)

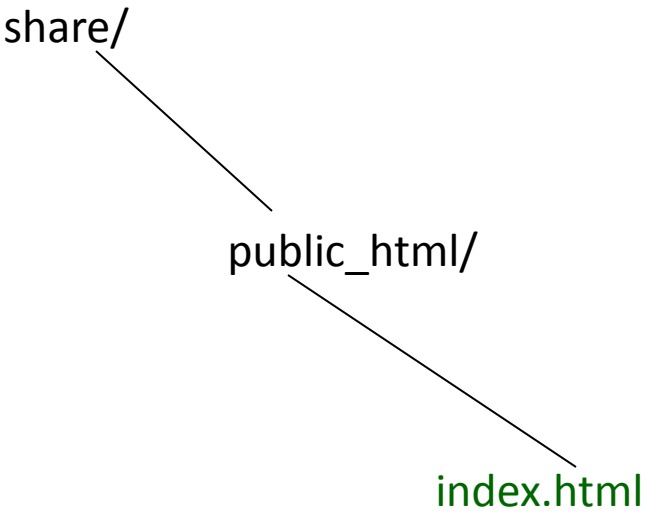
- **Basic HTML:**
  - **Understand basic HTML syntax**
  - **Learn how to make your second Web page ("Hello world with links" and other objects)**

# Recap of Unix OS

- Unix operating system
  - Command line interface (shell)
    - Use Unix "terminal" program to access the shell
    - ➢More powerful shell: "**bash**"
  - ➢Try it now!
    - Open "terminal" from "Applications" menu
    - type "bash"
      - Supports *command history* (allows to modify previous commands instead of re-typing, and other nice features)

# Your web directory

Your home directory should contain a **share** directory with **public_html** subdirectory.
Your **public_html** directory should contain an “index.html” file.



# Accessing your web page

- http://dutch.mathcs.emory.edu/

- Your site:
  http://www.mathcs.emory.edu/~your_login/
  - http://www.mathcs.emory.edu/~cs190000/

➢ **Cannot log in to it!**

- But, **can** control the **files** visible on the web:
  **/home/your_login/cs190/web/**

# Sample HTTP exchange

- **Retrieve file:** http://www.mathcs.emory.edu/~cs190000/
- Request:
  - GET
- Web server translates to:
  - /home/cs190000/share/public_html/index.html
    - Translates URL path to *local* path on server
    - Index.html, index.php, index.htm are *default* files
    - If not present, will return directory listing
- Returns:
  - Header/Status: 200 (OK) or 404 (not found) , etc… and other info
  - Content-Type:
  - Content itself

# Example

HTTP/1.1 200 OK
Date: Fri, 31 Dec 1999 23:59:59 GMT
Content-Type: text/html
Content-Length: 1354

<html>
<body>
Hello World!
</body>
</html>

# Elements of HTML document

1. <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
2. <HTML>
3. <HEAD>
4. <TITLE>The title of the documents</TITLE>
5. <META NAME="description" CONTENT="This is a document">
6. <LINK REV="made" HREF="mailto:galactus@htmlhelp.com">
7. </HEAD>
8. <BODY>
9. ... document body
10. </BODY>
11. </HTML>

# HTML: Hello World Revisited

```
<html>
<head>
   <title> insert_title_here
   </title>
</head>
<body>
        <h1>Hello World!</h1>
        Welcome to CS190!
</body>
</html>
```

# Exercise: Make your homepage

- Change to directory **share/public_html** in your home directory
- Open or create file **index.html** using **gedit**
  - **gedit index.html &**
  - Insert content into index.html:

```
<html>
<h1>Hello World!</h1>
</html>
```

  - Save file
- Run **firefox** in **background** (firefox **&** )
- Open URL http://www.mathcs.emory.edu/~**usern**/

# URLs

- http://www.mathcs.emory.edu/index.html
  - PROTOCOL://SERVER.DOMAIN/PATH/FILE

- http:// This is the protocol, which is almost always HTTP, the Hyper Text Transport Protocol. You'll also occasionally see FTP, the File Transfer Protocol; HTTPS, the secure version of HTTP; and others.

# Relative URLs

- If a file is in the same directory as you, just give its name. If we want to put a link *in* the file buffy.html linking it to willow.html, we can say

```
<a href="willow.html">her best friend Willow</a>

<a href="../season3.html">the Host returned in season 3</a>
```

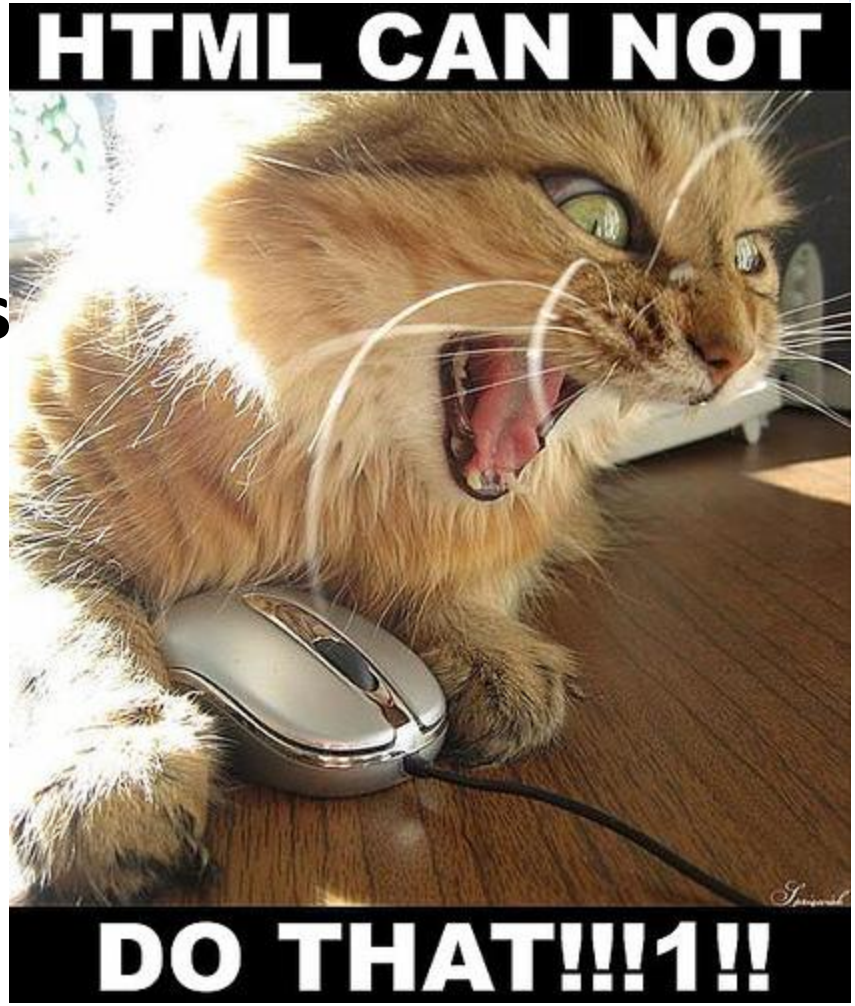
# Assignment 1 (a freebie ☺ )

- "Hello world" **in Video**:

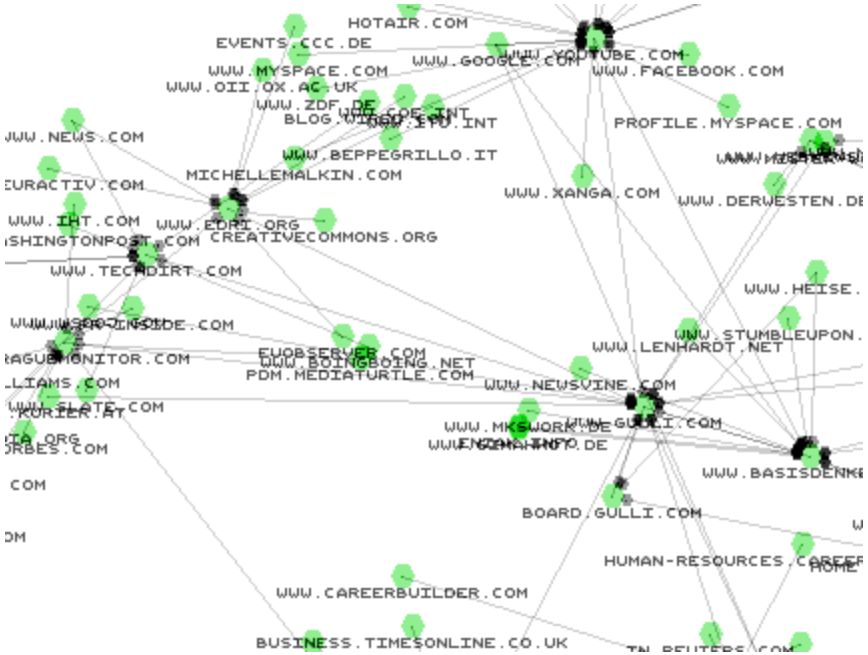
- Deliverables
  - Create a folder (**assignment1**)
  - Create file **index.html** (the main page for your hw)
  - The page should have:
    - Your name, e-mail, nicely formatted
    - **Embed** a video object (e.g., YouTube) video: http://www.youtube.com/watch?v=1IQFjTnDozo
    - Have **links** to 3 other "video hello" pages from other students in the class

# HTML as Programming Language

- Modifying layout
- Creating links
- **Composing pages: frames**



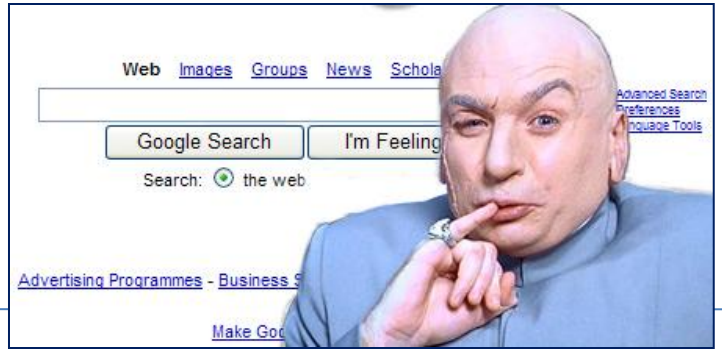
# Web: Content and Links (Revisited)

# HTML Frames


- Frames: some say they are…

http://apptools.com/rants/framesevil.php



# Frames (cont'd)

- The FRAMESET tag replaces the BODY tag.
- COLS attribute means that these frames will be laid out in **columns**.
  - The value of COLS, in this case, is the width of the columns in pixels or % of window. An asterisk means "the rest."
- ROWS attribute means to frames to be laid out in **rows**
  - The value defines the number and size of rows in a frameset (pixels or %)
- One FRAME tag for each column. The SRC attribute can be any URL.
- The NAME attribute allows us to access the frame through the object hierarchy.

# Frames Example: Image Gallery

```
<html>
   <frameset cols="25%,75%">
     <frame src= “menu.html" >
     <frame src= “content.html" >
   </frameset>
</html>
```

Example:

http://www.mathcs.emory.edu/~cs190000/jan22/

# Requirements for framestest

- Source files for frame content
  - menu.html
  - content.html

# Digression: Color and Graphics

- Standard colors:
  - black #000000
  - white #FFFFFF
  - **red #FF0000**
  - **green #00FF00**
  - **blue #0000FF**
- Our retinas happen to have rod-shaped cells that are sensitive to all light, and cone-shaped cells that come in three kinds: red-sensitive, green-sensitive, and blue-sensitive.
- Therefore, there are three (additive) primary colors: **Red**, **Green** and **Blue** or **RGB**.
- All visible colors are seen by exciting these three types of cells.

# Hexadecimal

- People use decimal (base 10)
- Computers use binary (base 2)
- Programmers often use hexadecimal (base 16) for "convenience"
- Each hexadecimal digit represents exactly four binary digits
  - Decimal 0 1 … 9 10 11 12 13 14 15 16 … 31 32 33
  - Hexadecimal 0 1 … 9 A B C D E F 10 …1F 20 21

# Forms

- Your (probably) first **GUI** (graphical user interface)
- The main tag is the **FORM** tag.

```
<form name="my_data">
    ... inputs go here ...
</form>
```

# HTML Forms: Enter The Programming

- <form>
  … input elements
  </form>

- <form>
  Search term:
  <input type="text" name="as_q" />
  <br>
  Language:
  <input type="text" name="lr" />
  </form>

# Form **Action**

- <form name="input"
  action="http://www.google.com/search?”
  method="get">

  …

  <input type="submit" value="Submit" />

  </form>
- **Field values are passed along with the request**
- Example:

  http://www.mathcs.emory.edu/~cs190000/jan22/search.html

# Additional resources

- HTML Markup Guide
  http://www.w3.org/MarkUp/Guide/

- To edit an HTML file (more easily):
  - Open file using **seamonkey**

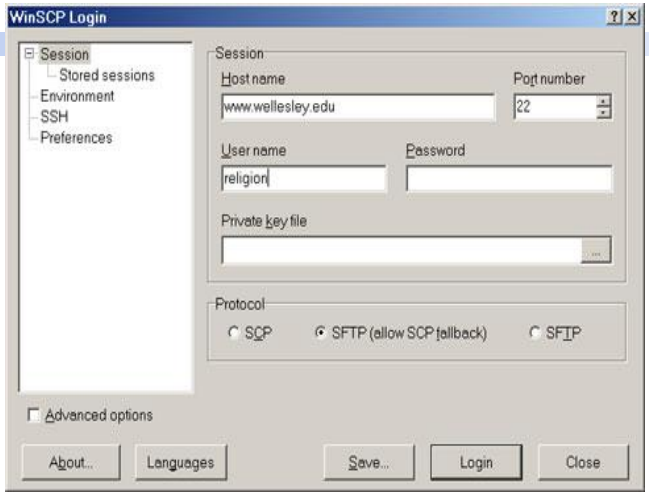
    Type “seamonkey & “ at terminal prompt
    Then select “Open file” from **File** menu.
    After page is opened, select “Edit page” from **File** menu
  - Note that by default page is opened in graphical editor mode; select “<HTML> Source” tab on bottom to edit the actual HTML source, as you will need to do for most projects.

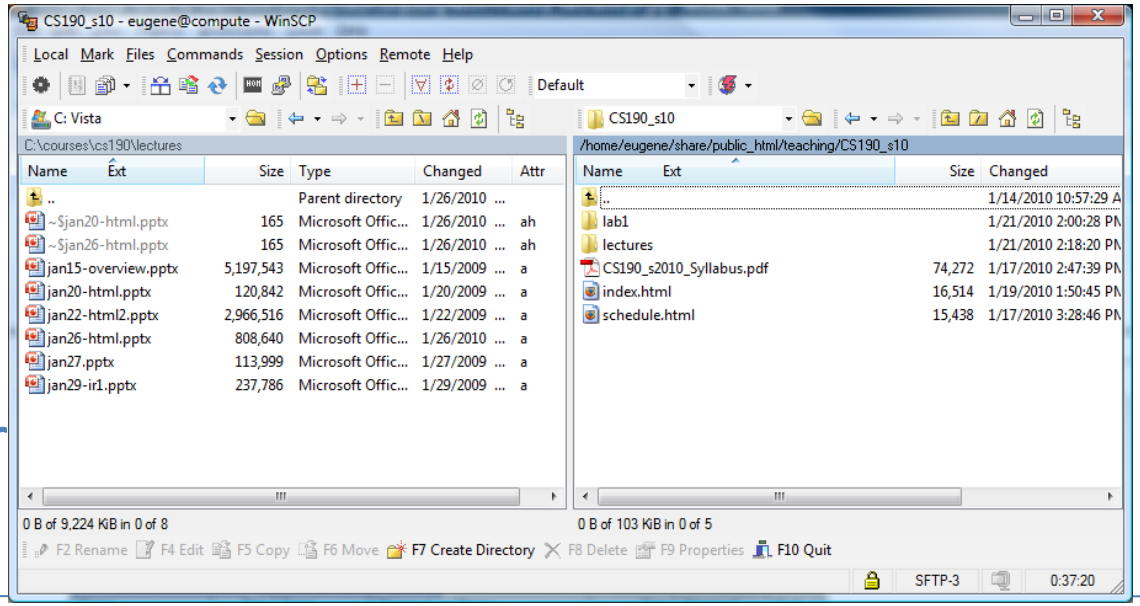
# Working from Home: Option 1

- Can login to any of the lab machines from home
- Can either have **shell** interface (commandline only)
- Or full graphical interface (more setup, but easier to work):
  http://www.mathcs.emory.edu/~cheung/Courses/RemoteAccess/index.html

# Option 2: Local Edit + Transfer Files

- My personal preference
- Tools needed:
  - **Download** file using FTP client (step 1)
    - WinSCP
  - Editor (step 2,3)
    - SeaMonkey,
    - Notepad
      any other editor
  - **Upload** edited file to your web director (step 4)



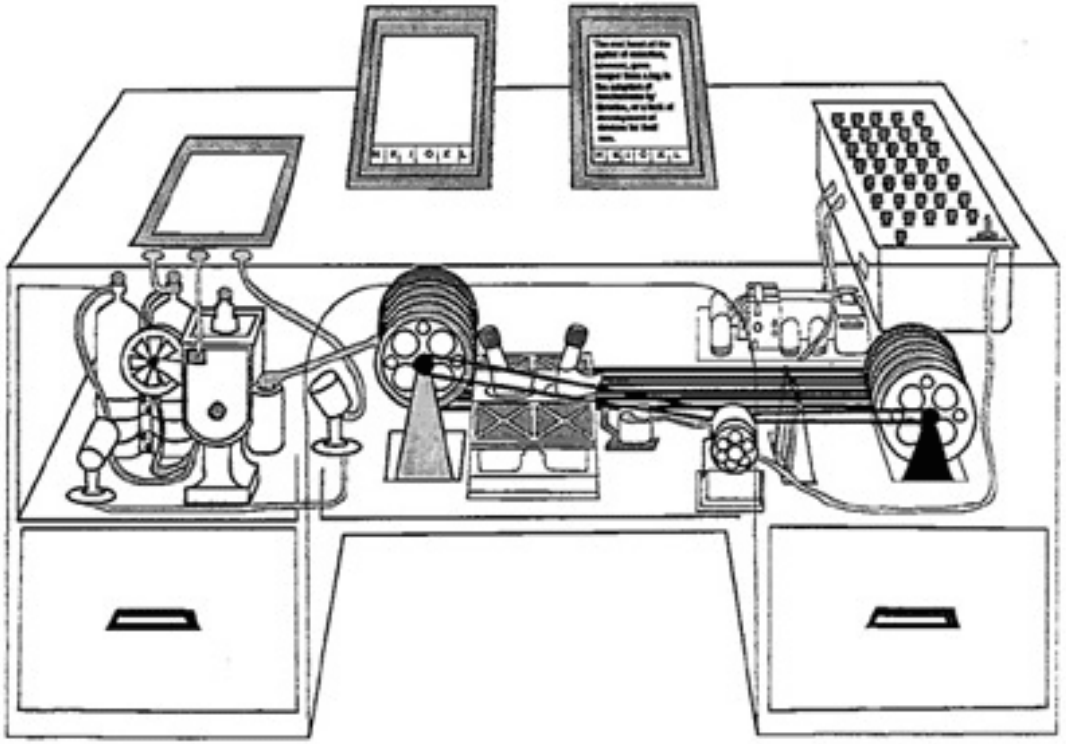



# HTTP + HTML = WEB

# The Vision: Memex

– **Vanevar Bush**: "As we may think":

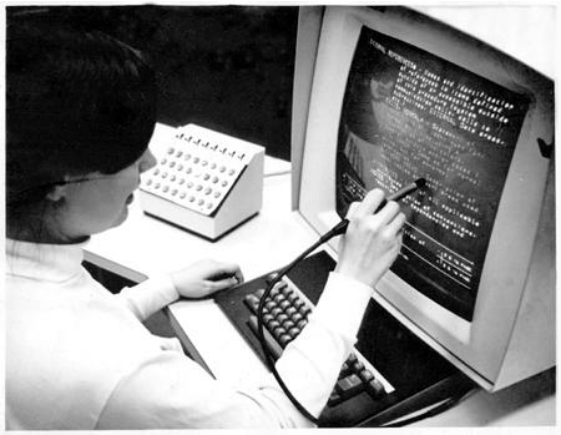
"**Memex**", 1945 → proto-hypertext system /desktop search



# The First Implementation: Hypertext

- Ted Nelson: **Hypertext**

  A text on a computer that will lead the user to other, related information on demand.



Precursor of The Web

# HTML: In the beginning…

- In 1989, Tim Berners-Lee created a proposal for a hypertext document system to be used within the CERN community. Although based in Switzerland, CERN members were scattered throughout the globe.
- Berners-Lee developed and defined the HTML language, which was created and defined using **SGML**, during the development cycle for the first Web browser/editor from October to December 1990.

# 1989: World Wide Web

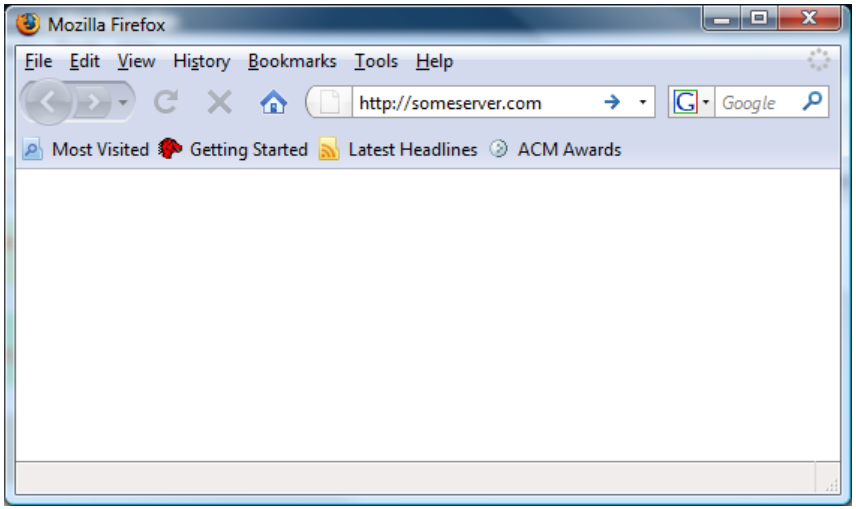
- Tim Berners-Lee connects 2 computers together using the

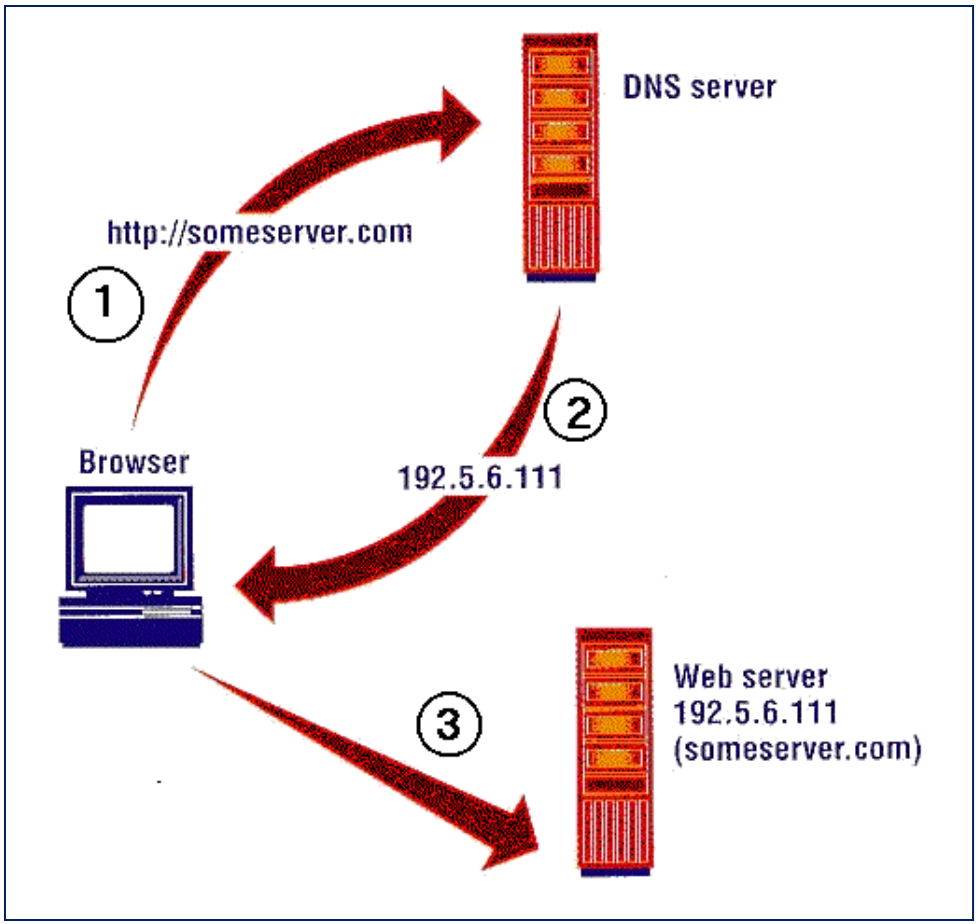
- HTTP protocol

- First **Web server** is born.

# Web **Server**

# Looking up a Web Page



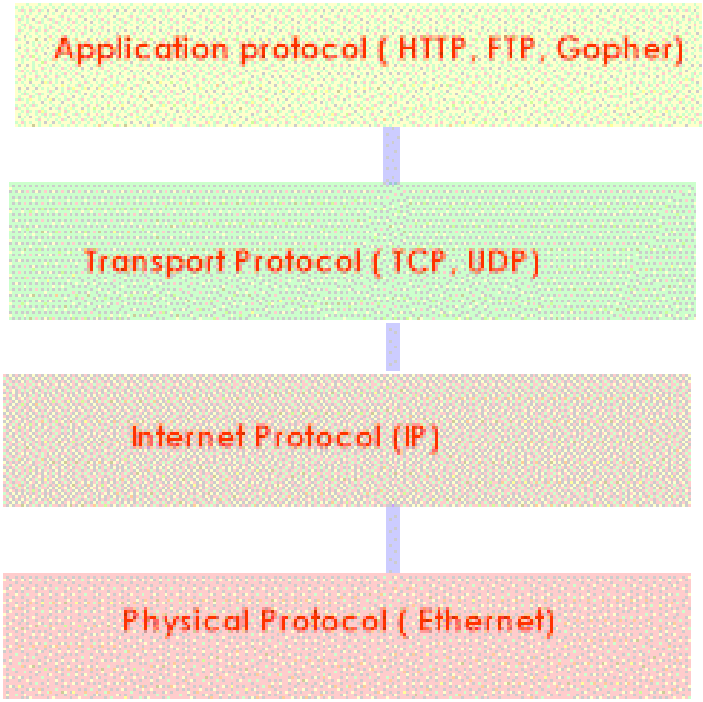
# Infrastructure for HTTP: The Internet

- "In the Beginning, ARPA created the ARPANET. And the ARPANET was without form and void.
- And darkness was upon the deep.
- And the spirit of ARPA moved upon the face of the network and ARPA said, 'Let there be a protocol,' and there was a protocol. And ARPA saw that it was good.
- And ARPA said, 'Let there be more protocols,' and it was so. And ARPA saw that it was good.
- And ARPA said, 'Let there be more networks,' and it was so."

-- *Danny Cohen* , Internet co-inventor

# TCIP/IP Protocol

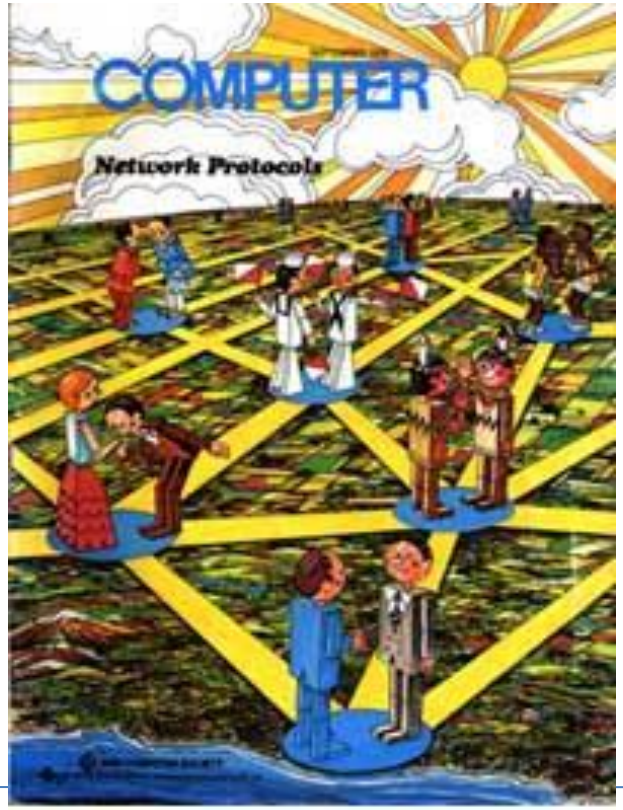
- Robust protocol that connects computers over the internet
- All Web traffic goes over TCIP/
- **IP: Unique Address of your computer**

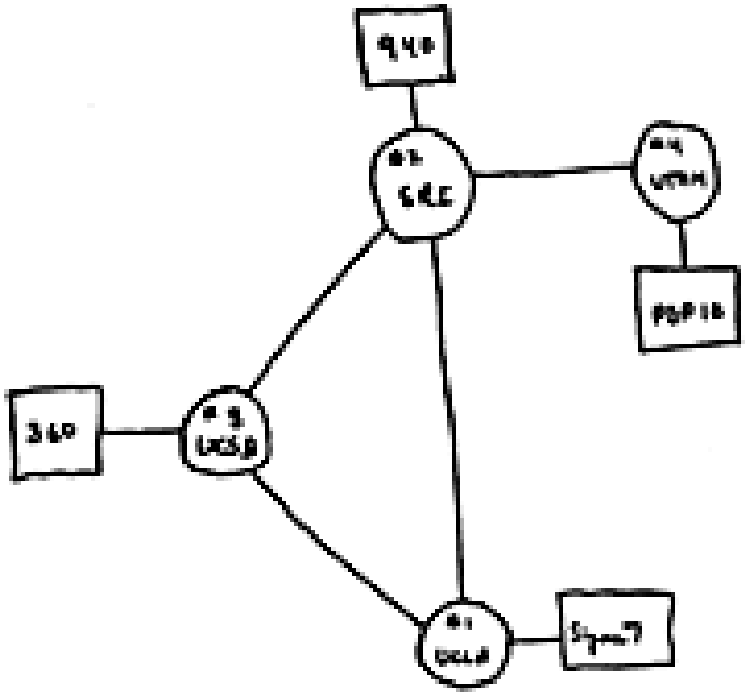




TCP/IP Layering

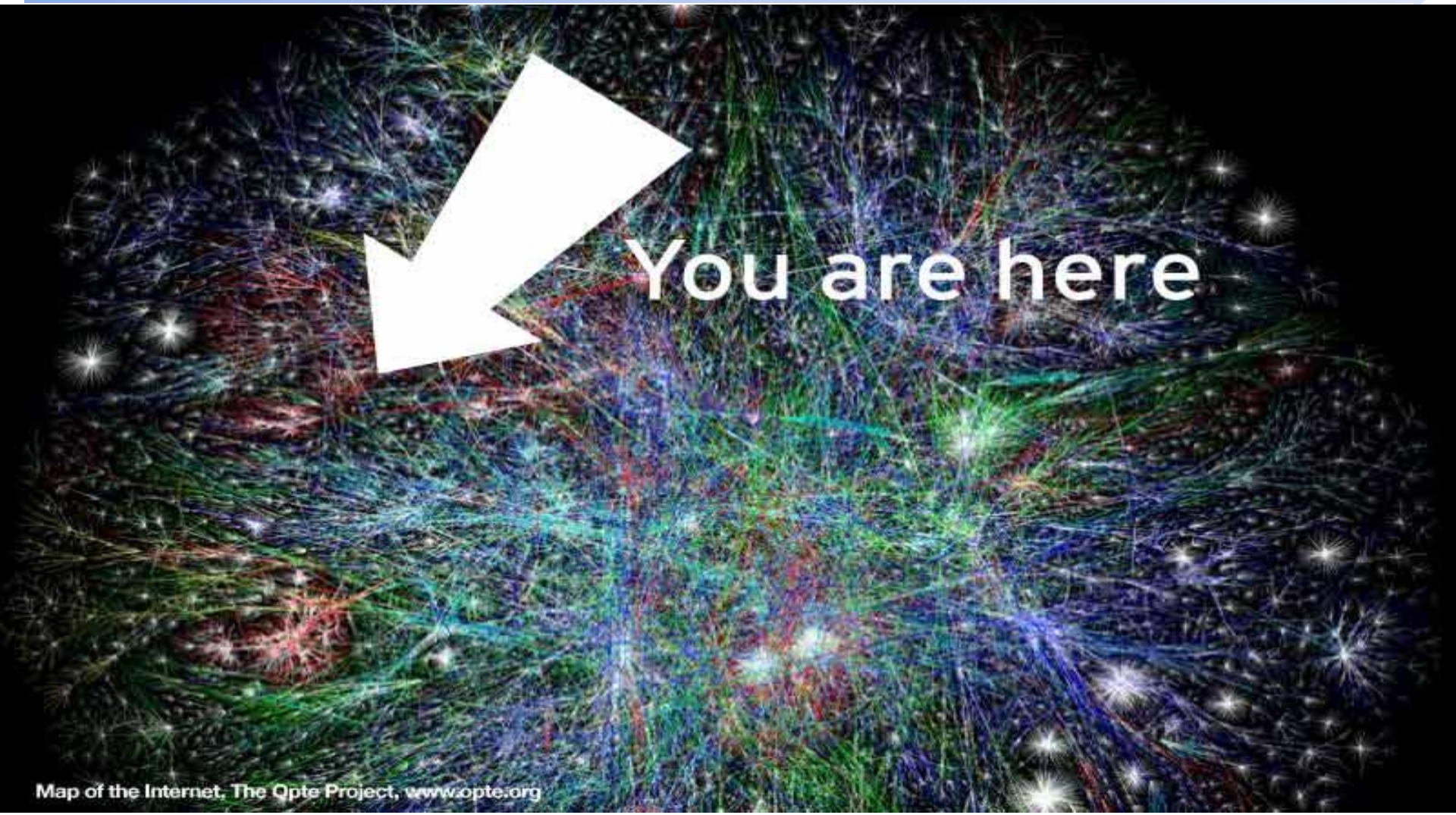
# Internet (cont'd)

- 1960s: First 4 nodes on Internet
- 1970:: thousands of nodes



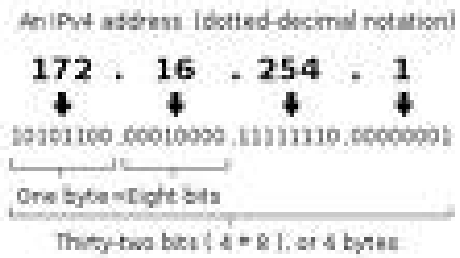



# Internet (Now)



# IP Address

- Uniquely identifies your computer




- Can be mapped (roughly) to geographical location

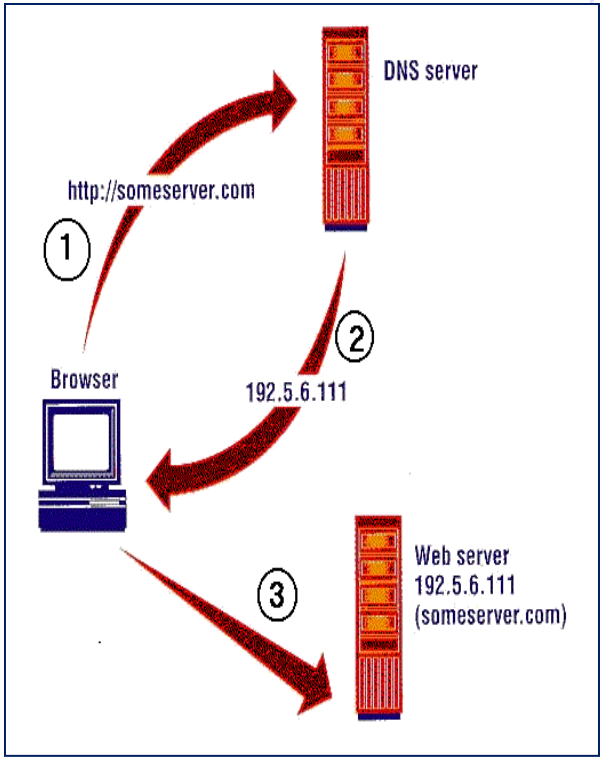
http://www.geobytes.com/IpLocator.htm

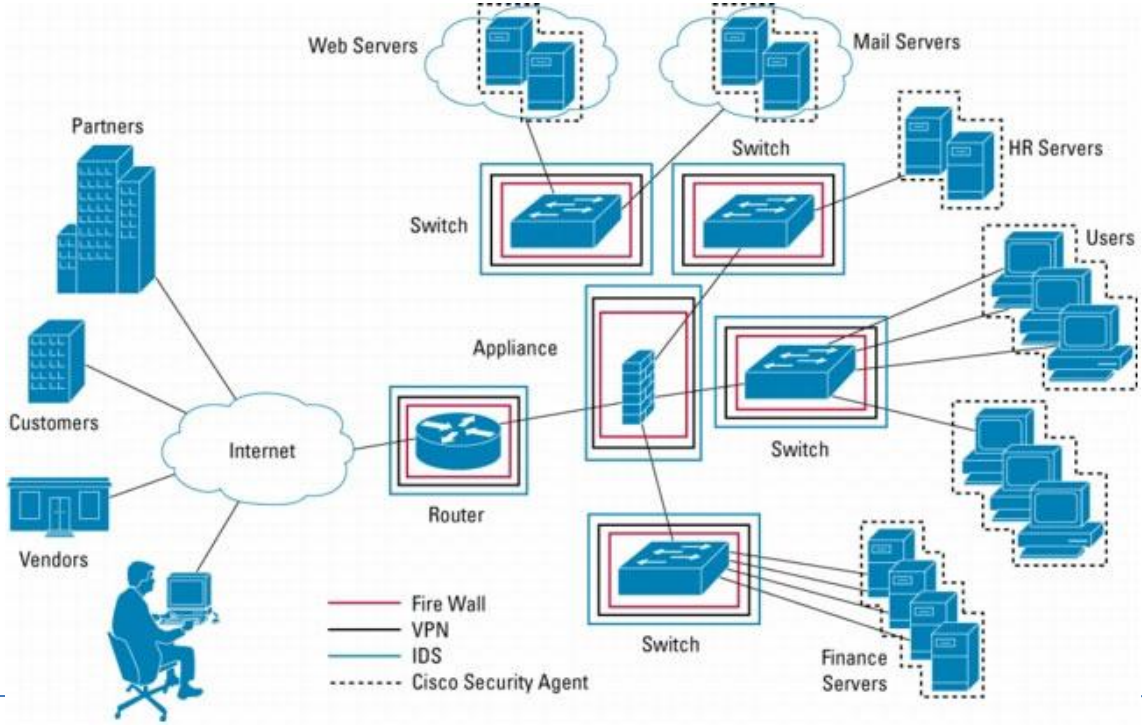
- Associates with contact person

http://whois.arin.net/rest/net/NET-170-140-0-0-1/pft

# Internet "Routing"

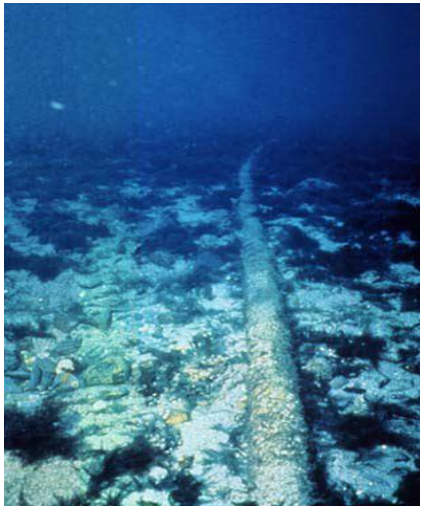
- How to find your way on the internet? That is, how is the *connection* to the webserver found?
- Answer: Internet "Routers" and "Switches"

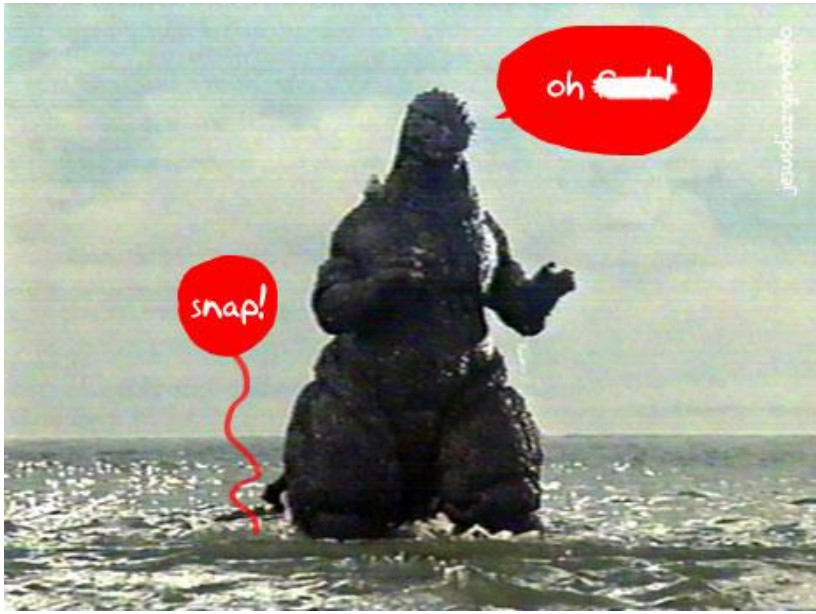


# TCIP/IP (continued)

- TCP/IP is *robust* to failure (things go wrong on the internet/physical layer)!






- Protocol *detects* errors so application can try to find a new *route* to the webpage and back

# Internet Routing (2)

- Each router tries to "direct traffic" based on **IP** addresses in the packets, in the most efficient (but not shortest) path.
  - Interesting and important research and engineering challenges to this day! More in advanced CS classes…

- Examples:

  /usr/sbin/traceroute cs190.mathcs.emory.edu (2 hops)

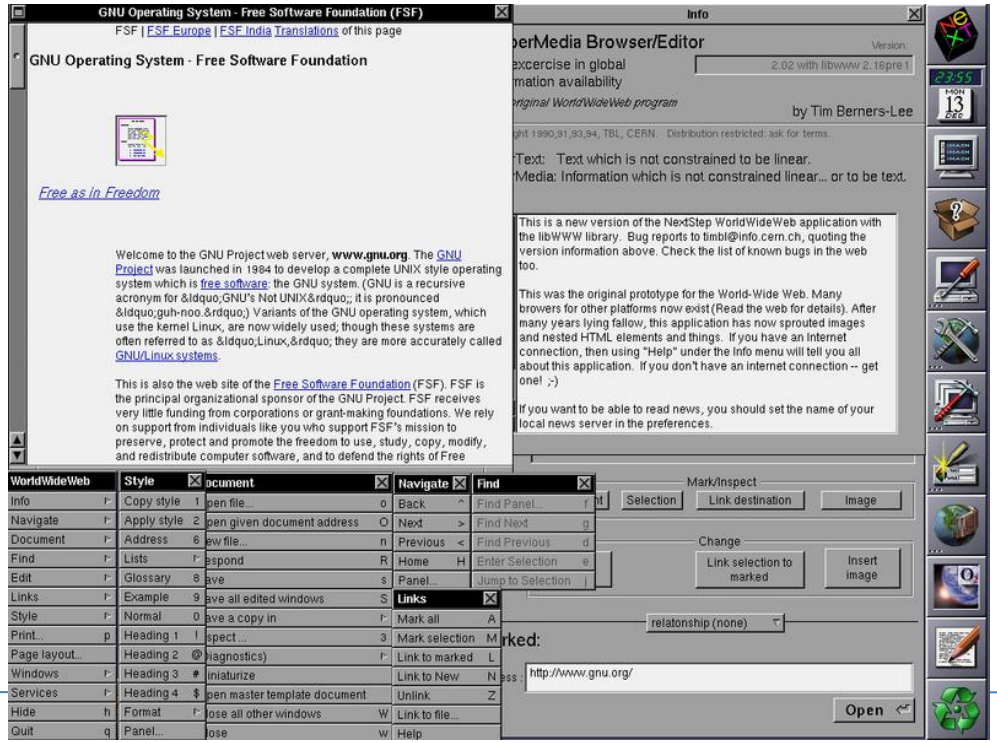
  /usr/sbin/traceroute www.google.com (12 hops)

  /usr/sbin/traceroute www.emory.edu (? hops)

# First Web Browser

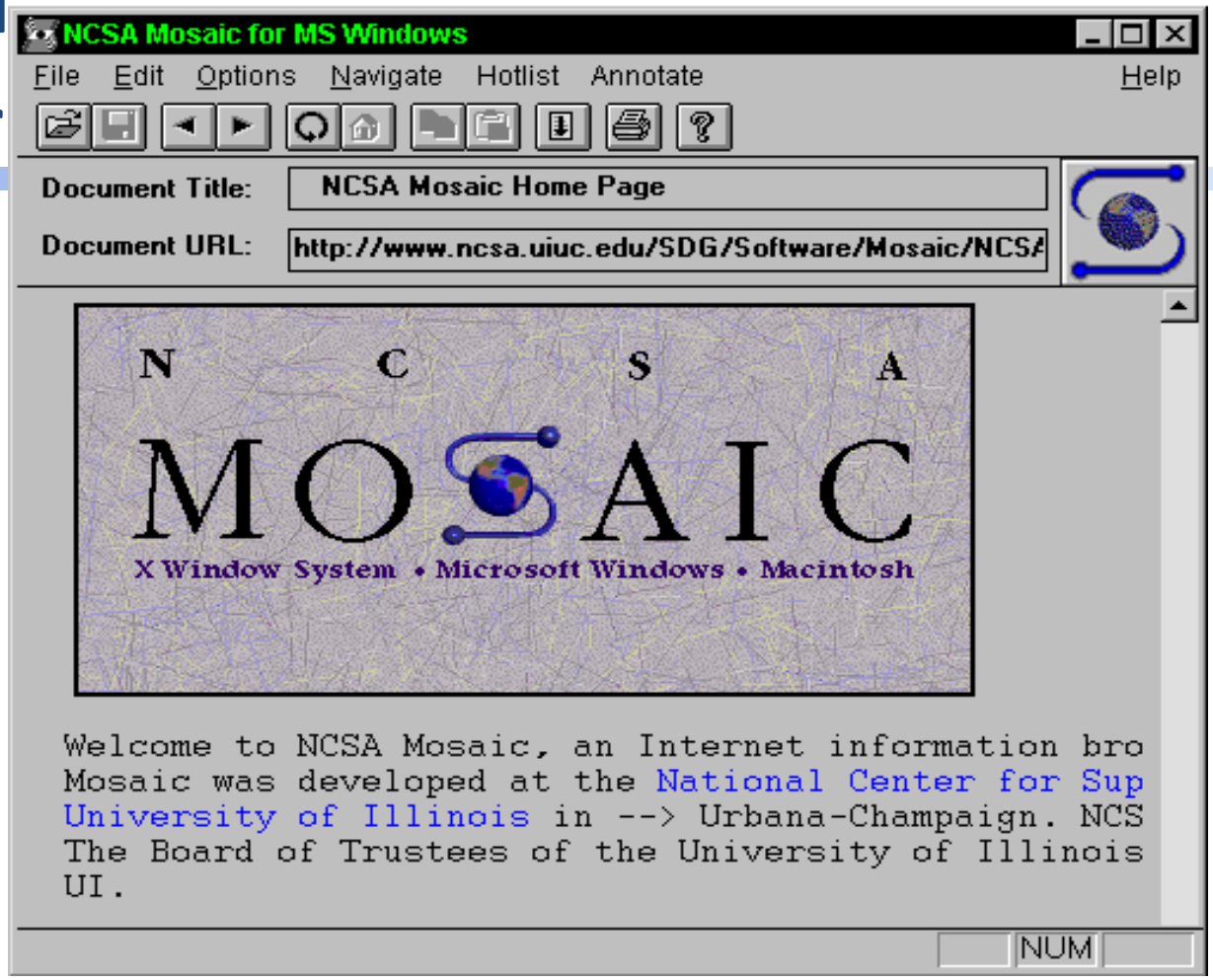
- Berners-Lee developed and defined the HTML language, which was created and defined using **SGML**, during the development cycle for the first Web browser/editor from October to December 1990.

- First *PARSER* for HTML, the WorldWideWeb app:



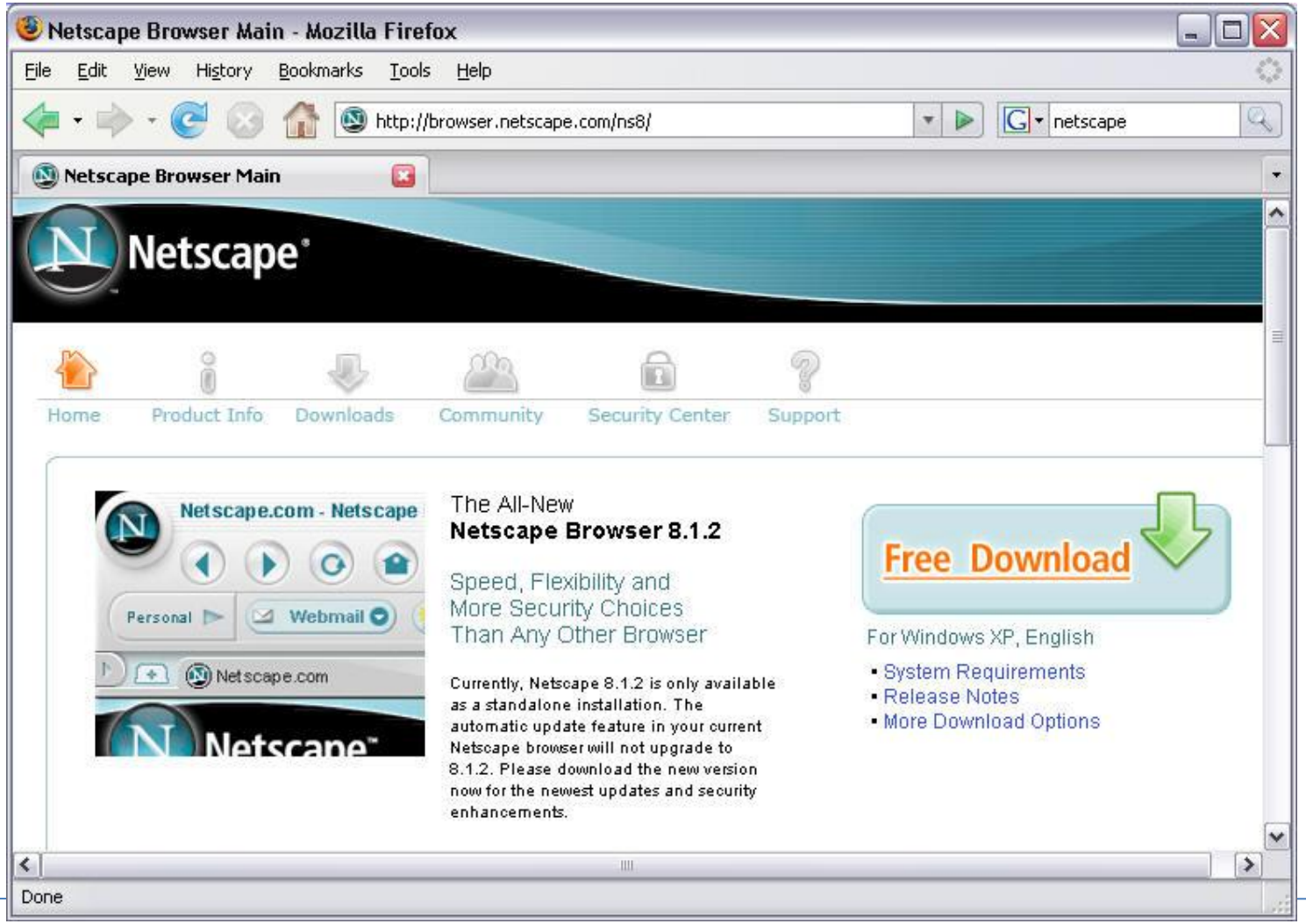
# First Rea
# Browser
# Mosaic



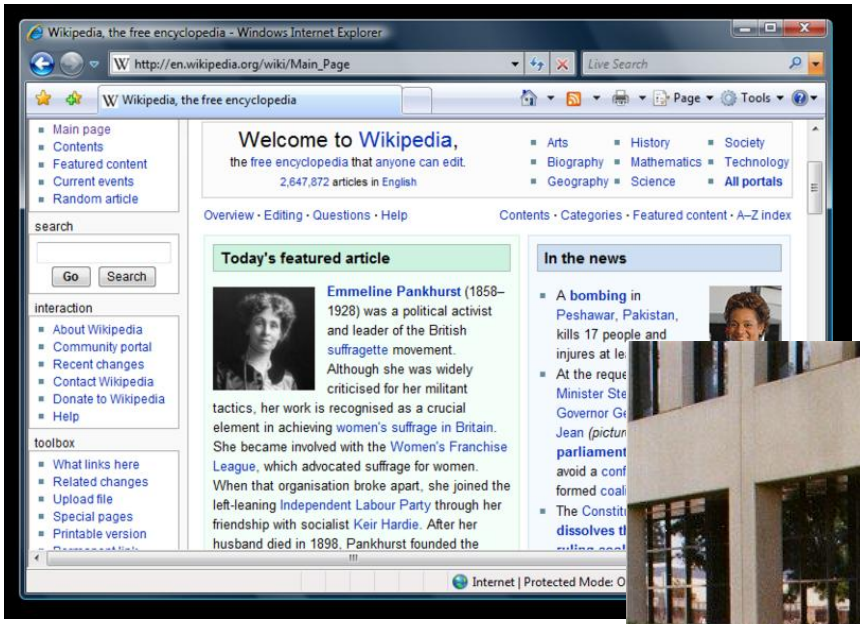



- Developed by Mark Andresen (NCSA), funded by Al Gore's High Performance Computing and Communication Act, 1991

# 1994-1997: Netscape



# 1995-present: MSFT Explorer




# 2003-present: Netscape begat Firefox, and others followed: Opera, Chrome, Safari,…