

1.1 A Brief Intro to the Internet

- *Origins*
 - ARPAnet - late 1960s and early 1970s
 - Network reliability
 - For ARPA-funded research organizations
 - BITnet, CSnet - late 1970s & early 1980s
 - email and file transfer for other institutions
 - NSFnet - 1986
 - Originally for non-DOD funded places
 - Initially connected five supercomputer centers
 - By 1990, it had replaced ARPAnet for non-military uses
 - Soon became the network for all (by 1990)
 - NSFnet eventually became known as the Internet
- *What the Internet is:*
 - A world-wide network of computer networks
 - At the lowest level, since 1982, all connections use TCP/IP
 - TCP/IP hides the differences among devices connected to the Internet

1.1 A Brief Intro to the Internet (continued)

- *Internet Protocol (IP) Addresses*
 - Every node has a unique numeric address
 - Form: 32-bit binary number
 - New standard, IPv6, has 128 bits (1998)
 - Organizations are assigned groups of IPs for their computers
- *Domain names*
 - Form: host-name.domain-names
 - First domain is the smallest; last is the largest
 - Last domain specifies the type of organization
 - *Fully qualified domain name* - the host name and all of the domain names
 - DNS servers - convert fully qualified domain names to IPs
- *Problem:* By the mid-1980s, several different protocols had been invented and were being used on the Internet, all with different user interfaces (Telnet, FTP, Usenet, mailto)

1.2 The World-Wide Web

- A possible solution to the proliferation of different protocols being used on the Internet
- *Origins*
 - Tim Berners-Lee at CERN proposed the Web in 1989
 - Purpose: to allow scientists to have access to many databases of scientific work through their own computers
 - Document form: hypertext
 - Pages? Documents? Resources?
 - We'll call them documents
 - Hypermedia – more than just text – images, sound, etc.
- *Web or Internet?*
 - The Web uses one of the protocols, `http`, that runs on the Internet--there are several others (`telnet`, `mailto`, etc.)

1.3 Web Browsers

- Mosaic - NCSA (Univ. of Illinois), in early 1993
 - First to use a GUI, led to explosion of Web use
 - Initially for X-Windows, under UNIX, but was ported to other platforms by late 1993
- Browsers are clients - always initiate, servers react (although sometimes servers require responses)
- Most requests are for existing documents, using HyperText Transfer Protocol (HTTP)
 - But some requests are for program execution, with the output being returned as a document

1.4 Web Servers

- Provide responses to browser requests, either existing documents or dynamically built documents
- Browser-server connection is now maintained through more than one request-response cycle

1.5 URLs

- General form:

scheme:object-address

- The scheme is often a communications protocol, such as `telnet` or `ftp`
- For the `http` protocol, the object-address is: fully qualified domain name/doc path
- For the `file` protocol, only the doc path is needed
- Host name may include a port number, as in `zeppo:80` (80 is the default, so this is silly)
- URLs cannot include spaces or any of a collection of other special characters (semicolons, colons, ...)
- The doc path may be abbreviated as a *partial path*
 - The rest is furnished by the server configuration
- If the doc path ends with a slash, it means it is a directory

1.6 Multipurpose Internet Mail Extensions (MIME)

- Originally developed for email
- Used to specify to the browser the form of a file returned by the server (attached by the server to the beginning of the document)
- Type specifications
 - Form:
type/subtype
 - Examples: `text/plain`, `text/html`, `image/gif`, `image/jpeg`
- Server gets type from the requested file name's suffix (`.html` implies `text/html`)
- Browser gets the type explicitly from the server
- *Experimental types*
 - Subtype begins with `x-`
e.g., `video/x-msvideo`
 - Experimental types require the server to send a helper application or plug-in so the browser can deal with the file

1.7 The HyperText Transfer Protocol

- The protocol used by ALL Web communications

- *Request Phase*

- Form:

HTTP method domain part of URL HTTP ver.
Header fields
blank line
Message body

- An example of the first line of a request:

```
GET /cs.uccp.edu/degrees.html HTTP/1.1
```

- *Most commonly used methods:*

GET - Fetch a document

POST - Execute the document, using the data in
body

HEAD - Fetch just the header of the document

PUT - Store a new document on the server

DELETE - Remove a document from the server

1.7 The HyperText Transfer Protocol (continued)

- Four categories of header fields:

1. General

2. Request

3. Response

4. Entity

- Common request fields:

```
Accept: text/plain
```

```
Accept: text/*
```

```
If-Modified_since: date
```

- Common response fields:

```
Content-length: 488
```

```
Content-type: text/html
```

1.7 The HyperText Transfer Protocol (continued)

- Response Phase

- Form:

Status line
Response header fields
blank line
Response body

- Status line format:

HTTP version status code explanation

- Example: HTTP/1.1 200 OK

(Current version is 1.1)

- Status code is a three-digit number; first digit specifies the general status

1 => Informational
2 => Success
3 => Redirection
4 => Client error
5 => Server error

- The header field, `Content-type`, is required

1.7 The HyperText Transfer Protocol (continued)

- An example of a complete response header:

```
HTTP/1.1 200 OK
Date: Mon, 27 Jun 2002 17:22:47 GMT
Server: Apache/1.3.22 (Unix) (Red-Hat/Linux)
Last-modified: Wed, 26 Jun 2002 18:12:29 GMT
Etag: "841fb-4b-3d1a0179"
Accept-ranges: bytes
Content-length: 75
Connection: close
Content-type: text/html
```

1.8 The Web Programmer's Toolbox

- *HTML*

- To describe the general form and layout of documents
- An HTML document is a mix of content and controls
 - Controls are tags and their attributes
 - Tags often delimit content and specify something about how the content should be arranged in the document
 - Attributes provide additional information about the content of a tag
- *Tools for creating HTML documents*
 - HTML editors - make document creation easier
 - Shortcuts to typing tag names, spell-checker,
 - WYSIWYG HTML editors
 - Need not know HTML to create HTML documents

1.8 The Web Programmer's Toolbox (continued)

- *Plug ins*

- Integrated into tools like word processors, effectively converting them to WYSIWYG HTML editors

- *Filters*

- Convert documents in other formats to HTML

- *Advantages of both filters and plug-ins:*

- Existing documents produced with other tools can be converted to HTML documents
- Use a tool you already know to produce HTML

- *Disadvantages of both filters and plug-ins:*

- HTML output of both is not perfect - must be fine tuned
- HTML may be non-standard
- You have two versions of the document, which are difficult to synchronize

1.8 The Web Programmer's Toolbox (continued)

- *XML*

- A meta-markup language
- Used to create a new markup language for a particular purpose or area
- Because the tags are designed for a specific area, they can be meaningful
- No presentation details
- A simple and universal way of representing data of any textual kind

- *JavaScript*

- A client-side HTML-embedded scripting language
- Only related to Java through syntax
- Dynamically typed and not object-oriented
- Provides a way to access elements of HTML documents and dynamically change them

1.8 The Web Programmer's Toolbox (continued)

- *Java*

- General purpose object-oriented programming language
- Based on C++, but simpler and safer
- Our focus is on applets and servlets

- *Perl*

- Provides server-side computation for HTML documents, through CGI
- Perl is good for CGI programming because:
 - Direct access to operating systems functions
 - Powerful character string pattern-matching operations
 - Access to database systems
- Perl is highly platform independent, and has been ported to all common platforms
- Perl is not just for CGI

1.8 The Web Programmer's Toolbox (continued)

- *PHP*

- A server-side scripting language**
- An alternative to CGI**
- Similar to JavaScript**
- Great for form processing and database access through the Web**