TA: Xifan Zheng
Email: zhengxifan0403@gmail.com

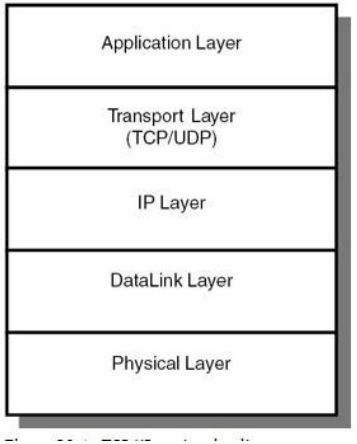
Welcome to CPSC 441!

Today's Tutorial

- HTTP protocol review
- HTTP request/response specification
- Conditional Get
- Redirection

What is HTTP?

- HTTP stands for **Hypertext Transfer Protocol**.
  - Used to deliver virtually all files and other data (collectively called **resources**) on the World Wide Web
  - Usually, HTTP takes place through TCP/IP sockets.



- A browser is an *HTTP client*
  - It sends requests to an *HTTP server* (Web server)
  - The standard/default port for HTTP servers to listen on is 80

- A **resource** is some chunk of data that is referred to by a URL
  - The most common kind of resource is a file
  - A resource may also be a dynamically-generated content, e.g., query result, CGI scrip output, etc.
  - As a practical matter, almost all HTTP resources are currently either files or server-side script output.

Structure of HTTP Transactions

- HTTP uses the client-server model:
  - An *HTTP client* opens a connection and sends a *request message* to an *HTTP server*;
  - The server then returns a *response message*, usually containing the resource that was requested.
  - After delivering the response, the server closes the connection (or not).
- Format of the HTTP request and response messages:
  - Almost the same, human readable (English-oriented)
  - An initial line specifying the method,
  - zero or more header lines,
  - a blank line (i.e. a CRLF by itself), and
  - an optional message body (e.g. a file, or query data, or query output).

```
<initial line, different for request vs. response>
Header1: value1
Header2: value2
Header3: value3
<optional message body, like file or query data; may be many lines, may be binary>
```

Initial Request Line

- The initial line is different for the request than for the response.
- A request line has three parts, separated by spaces:
  - a ***method*** name,
  - the **local path** of the requested resource, (host name will be specified in hearder line)
  - and the **version** of HTTP being used.

- A typical request line is:

```
GET  /path/to/file/index.html  HTTP/1.1
```

- **GET** is the most common HTTP method; it says "give me this resource".
- Other methods include **POST** and **HEAD**, etc.
- Method names are always uppercase.
- The path is the part of the URL after the host name, also called the *request URI* (a URI is like a URL, but more general).
- The HTTP version always takes the form "**HTTP/x.x**", uppercase.

Initial Response Line

- `HTTP/1.0 200 OK`
- **Status line:**
  - The HTTP version,
  - A *response status code* that gives the result of the request,
  - An English *reason phrase* describing the status code.
- Response categories:
  - **1xx** an informational message only
  - **2xx** success of some kind
  - **3xx** redirects the client to another URL
  - **4xx** an error on the client's part
  - **5xx** an error on the server's part

- The most common status codes are:
  - **200 OK** The request succeeded, and the resulting resource is returned in the message body.
  - **404 Not Found**
  - **301 Moved Permanently**
  - **302 Moved Temporarily**
  - **303 See Other** *(HTTP 1.1 only)*The resource has moved to another URL
  - Check RFC 2616 for the complete list

Header Lines

- Header lines provide information about the request, response, or the object sent.
- One line per header, of the form "**Header-Name: value**", ending with CRLF.
- The header name is not case-sensitive (the value may be).
- Header lines beginning with space or tab are actually part of the previous header line, folded into multiple lines. E.g.,

```
Header1: some-long-value-1a, some-long-value-1b
HEADER1: some-long-value-1a,
         some-long-value-1b
```

Header Lines (cont'd)

- HTTP 1.1 defines 46 headers, and one (**Host:**) is required in requests.

- The **User-Agent:** header identifies the program that's making the request, in the form "**Program-name/x.xx**", where **x.xx** is the (mostly) alphanumeric version of the program.
  - For example, Netscape 3.0 sends the header

  "User-agent: Mozilla/3.0Gold".

- Response headers from the server:
  - The **Server:** header is analogous to the **User-Agent:** header: it identifies the server software
  - The **Last-Modified:** header gives the modification date of the resource that's being returned. It's used in caching and other bandwidth-saving activities. Use Greenwich Mean Time, in the format Last-Modified: Fri, 31 Dec 1999 23:59:59 GMT

The Message Body

- After headers, there may be a body of data
- In a response this may be:
  - the requested resource
  - or perhaps explanatory text if there's an error.
- In a request this may be:
  - the user-entered data
  - or uploaded files
- If an HTTP message includes a body, there are usually header lines in the message that describe the body.
  - The **Content-Type:** header gives the MIME-type of the data e.g., **text/html** or **image/gif**.
  - The **Content-Length:** header gives the number of bytes in the body.

Sample HTTP Exchange

HTTP Request

GET /path/file.html HTTP/1.1
Host: www.host1.com:80
User-Agent: HTTPTool/1.0
[blank line here]

HTTP Response

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

<html>
<body>
<h1>Happy New Millennium!</h1>
(more file contents) . . . </body>
</html>

The HEAD Method

- A HEAD request is just like a GET request, except:
  - It asks the server to return the response headers only, not the actual resource. (i.e. no message body)
  - This is used to check characteristics of a resource without actually downloading it
  - HEAD is used when you don't actually need a file's contents.

- The response to a HEAD request must *never* contain a message body, just the status line and headers.

The POST Method

- A POST request is used to send data to the server

- A POST request is different from a GET request in the following ways:
  - There's a block of data sent with the request, in the message body.
  - There are usually extra headers to describe this message body, e.g., **Content-Type:** and **Content-Length:**.
  - The *request URI* is not a resource to retrieve; it's usually a program to handle the data you're sending.
  - The HTTP response is normally program output, not a static file.

- The most common use of POST, is to submit HTML form data to CGI scripts. In this case:
  - The **Content-Type:** header is usually **application/x-www-form-urlencoded**,
  - The **Content-Length:** header gives the length of the HTML form data.

The POST Method

- Here's a typical form submission, using POST:
- You can use a POST request to send whatever data you want, not just form submissions. Just make sure the sender and the receiving program agree on the format.
- The GET method can also be used to submit forms. The form data is URL-encoded and appended to the request URI.

```
POST /login.jsp HTTP/1.1
Host: www.mysite.com
User-Agent: Mozilla/4.0
Content-Length: 27
Content-Type: application/x-www-form-urlencoded

userid=joe&password=guessme
```

Caching

- To avoid sending resources that don't need to be sent, thus saving bandwidth/reduce response time

- Proxy or web browser check if the required content is already available in the cache.
  - A copy of the previous content is saved in the cache
  - Upon a new request, first the cache is searched
  - If found in cache, return the content from cache
  - If not in cache, send request to the server

- But what if the content is out of date?
  - We need to check if the content is modified since last access

The Date: Header

- We need timestamp responses for caching.
- Servers must timestamp every response with a **Date:** header containing the current time e.g.,

  Date: Fri, 31 Dec 1999 23:59:59 GMT

- All responses except those with 100-level status (but including error responses) must include the **Date:** header.
- All time values in HTTP use Greenwich Mean Time.

Conditional Get

- Allow Cache server to **verify** that its objects are up to date or not
- Should include **If-Modified-Since:** This header is used with the **GET** method to check if a content is modified since the last access
  - If the requested resource has been modified since the given date, ignore the header and return the resource.
  - Otherwise, return a "**304 Not Modified**" response, including the **Date:** header and no message body, e.g.,

HTTP/1.1 304 Not Modified
Date: Fri, 31 Dec 1999 23:59:59 GMT
[blank line here]

Conditional Get Example

First time:

Request

GET /sample.html HTTP/1.1
Host: example.com

Response

HTTP/1.1 200 OK
Date: Tue, 27 Dec 2005 11:25:19 GMT
Server: Apache/1.3.33 (Unix) PHP/4.3.10
Last-Modified: Wed, 01 Sep 2004 13:24:52 GMT
(data data data data….)

Next time:

Conditional Get Request

GET /sample.html HTTP/1.1
Host: example.com
If-Modified-Since: Tue, 27 Dec 2005 11:25:19 GMT

Response

HTTP/1.1 304 Not Modified
Date: Wed, 28 Dec 2005 05:25:19 GMT
Server: Apache/1.3.33 (Unix) PHP/4.3.10

(empty entity body)

Redirection Example

Request 1

GET /~carey/index.html HTTP/1.1
Host: www.cpsc.ucalgary.ca
Connection: keep-alive
User-Agent: Mozilla/5.0 […]
Accept: text/html,application/ […]
Accept-Encoding: gzip,deflate,sdch
[…]
\r\n

Request 2

GET /~carey/index.html HTTP/1.1
Host: pages.cpsc.ucalgary.ca
Connection: keep-alive
User-Agent: Mozilla/5.0 […]
Accept: text/html,application/ […]
Accept-Encoding: gzip,deflate,sdch
[…]
\r\n

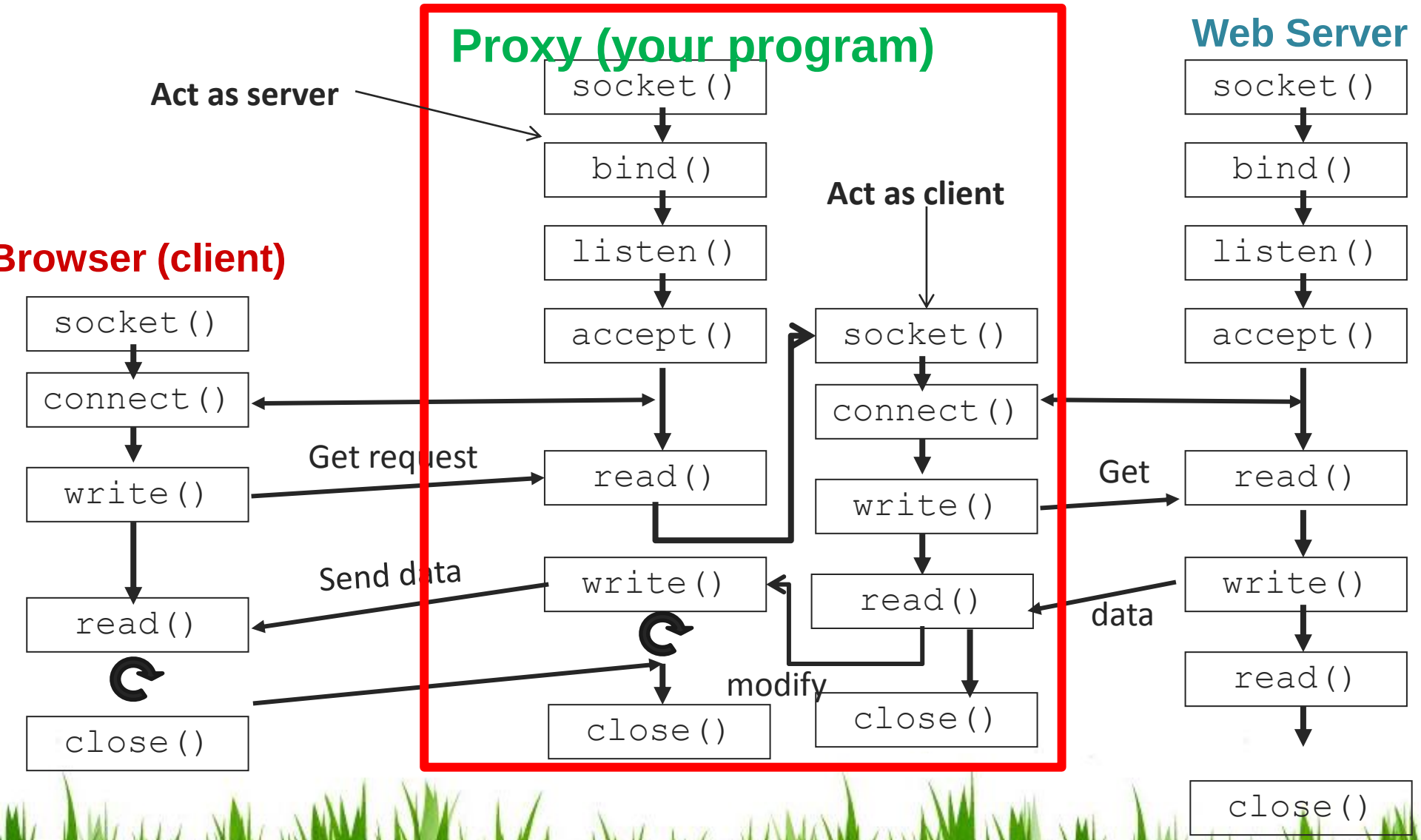
Response 1

HTTP/1.1 302 Found
Date: Sat, 21 Jan 2012 01:10:43 GMT
Server: Apache/2.2.4 (Unix) mod_ssl/2.2.4 OpenSSL/0.9.7a
PHP/5.2.9 mod_jk/1.2.25
Location: http://pages.cpsc.ucalgary.ca/~carey/index.html
\r\n

Response 2

HTTP/1.1 200 OK
Date: Sat, 21 Jan 2012 01:11:49 GMT
Server: Apache/2.2.4 (Unix) […]
Last-Modified: Mon, 16 Jan 2012 05:40:45 GMT
Content-Length: 3157
Keep-Alive: timeout=5
Connection: Keep-Alive
Content-Type: text/html
\r\n
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
[…]
</html>
\r\n

Hint for Assignment1



Thanks for attending!