95-702  Distributed Systems

HTTP
&
Server-Side Programming
The genesis of the WWW

• Sir Tim Berners Lee
  – 2016 Turing Award winner
• Defined 3 standards:
  1. Hypertext Transfer Protocol (HTTP)
  2. Uniform Resource Identifiers (URI)
     • UR Locators (URL) – e.g. http://cmu.edu/heinz
       – scheme://domain:port/path?query_string#fragment_id
     • UR Names (URN) e.g. urn:isbn:0132143011
       – urn:namespace_identifier:namespace_specific_string
  3. Hypertext Markup Language (HTML)
• Then implemented to these standards:
  – A server program
  – A browser program
Who maintains the standards?

The Internet Engineering Task Force (IETF) of the Internet Society, in cooperation with the World Wide Web Consortium (W3C)

w3.org

ietf.org

95-702 Distributed Systems 3
If you have not used HTML before, you should learn the basics.

W3Schools is a good HTML tutorial resource
  – Do the HTML Tutorial from start through the "HTML Attributes" page (6 pages)
    • http://www.w3schools.com/html
  – Do all the "HTML Forms" pages (4 pages)
    • http://www.w3schools.com/html/html_forms.asp

Lynda is another good resource:
  – http://www.cmu.edu/lynda/
Cascading Style Sheets

• Cascading Style Sheets (CSS) provide web-page styling
  – e.g. layout, fonts, colors

• **We do not require any CSS in this class.**

• But you can add some if you like to make your small apps look better.

• W3Schools CSS Tutorial:
  – [http://www.w3schools.com/css](http://www.w3schools.com/css)
JSP = Java Server Pages

• Basic HTML pages are static, and sometimes you want to include dynamic content
  – E.g. include values from a database query in a table.
• As you saw in the lab HelloWorld servlet example, you can write HTML directly in a servlet.
  – e.g. out.println("<html>"); out.println("<head>");
  – Getting the HTML structure right can be troublesome.
• What would be nice, would be to have the HTML structure (as a template), but be able to replace pieces programatically.
• JSP is a templating language
  – Allows you to write HTML templates and fill in values using Java code
• Predecessor to newer templating engines
  – E.g. erb (Ruby) and ejs (JavaScript)
JSP

• JSP is HTML, with Java code embedded in it.
• E.g. put 20 horizontal rule lines:
  
  \[
  \text{(<hr> is the HTML for a single horizontal rule line)}
  \]
  
  \[
  - \text{<% for (int x = 0; x< 20; x ++ ) { %> <hr> <% } %>}
  \]

• JSP is used often to insert values received from the Controller or Model into the View
  
  \[
  - \text{<%= .. %> is shorthand for "print"}
  \]
  
  \[
  - \text{E.g.}
  \]
  
  \[
  - \text{ <img src=<%= request.getAttribute("pictureURL")%> >}
  \]

• Good MVC dictates not to put logic, except for plugging in information, in the View
HTTP Interaction Pattern

• Uses Request / Response interaction pattern
  – Client
    • Establishes a connection with the server
    • Sends a request message and waits
  – Server
    • Processes the request as it is received
    • Sends a response message over the same client connection

• Coupled in time:
  – Both client and server must be available at time of the interaction

• Coupled in space:
  – Both client and server must know the address of each other, and communicate directly point-to-point

• Can also be used for Request / Acknowledge
HTTP Request / Response

Source: Head First Servlets and JSP by Basham, Sierra, & Bates
HTTP Request

<table>
<thead>
<tr>
<th>General Format</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;method&gt;</code> <code>&lt;resource identifier&gt;</code> <code>&lt;HTTP Version&gt;</code> <code>&lt;crlf&gt;</code>[&lt;Header&gt;: &lt;value&gt;] <code>&lt;crlf&gt;</code> ...[&lt;Header&gt;: &lt;value&gt;] <code>&lt;crlf&gt;</code> a blank line[entity body]</td>
<td>GET /course/95-702/ HTTP/1.1 Host: <a href="http://www.andrew.cmu.edu">www.andrew.cmu.edu</a> User-Agent: Joe typing Accept: text/html This line intentionally left blank</td>
</tr>
</tbody>
</table>

- **Method**
  - GET, PUT, DELETE, HEAD, POST, etc.
- **Resource identifier** specifies the name of the target resource;
  - i.e. it's the URL stripped of the protocol and the server domain name.
  - When using the GET method,
    - this field will also contain a series of name=value pairs separated by ‘&’.
  - When using a POST method,
    - the entity body contains these pairs.

- **HTTP version** identifies the protocol used by the client.
## HTTP Response

<table>
<thead>
<tr>
<th>General Format</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;HTTP Version&gt;</code> <code>&lt;Status&gt;</code> <code>&lt;crlf&gt;</code></td>
<td>HTTP/1.1 200 OK</td>
</tr>
<tr>
<td><code>[&lt;Header&gt;: &lt;value&gt;]</code> <code>&lt;crlf&gt;</code></td>
<td>Date: Mon, 13 Jan 2014 15:43:08 GMT</td>
</tr>
<tr>
<td><code>...</code></td>
<td>Server: Apache/1.3.39 (Unix) mod_throttle/3.1.2 ...</td>
</tr>
<tr>
<td><code>[&lt;Header&gt;: &lt;value&gt;]</code> <code>&lt;crlf&gt;</code></td>
<td>Set-Cookie: webstats-cmu=cmu128.2.87.50.8400; ...</td>
</tr>
<tr>
<td>a blank line</td>
<td>Last-Modified: Sun, 12 Jan 2014 21:46:30 GMT</td>
</tr>
<tr>
<td>[response body]</td>
<td>Accept-Ranges: bytes</td>
</tr>
<tr>
<td></td>
<td>Content-Length: 9014</td>
</tr>
<tr>
<td></td>
<td>Content-Type: text/html</td>
</tr>
<tr>
<td></td>
<td>This line intentionally left blank</td>
</tr>
<tr>
<td></td>
<td><code>&lt;HTML&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;HEAD&gt;</code></td>
</tr>
<tr>
<td></td>
<td><code>&lt;META http-equiv=&quot;Content-Type&quot; content=&quot;text/html; charset=UTF-8&quot;&gt;</code></td>
</tr>
<tr>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>

- **HTTP version** identifies the protocol used by the client.
- **Status** indicates the result of the request
Some Common HTTP Status Codes

• 200 OK
• 301 Moved Permanently
• 400 Bad Request
• 401 Unauthorized
• 404 Not found
• 500 Internal Server Error
  – You get this when your server program throws an uncaught exception.
  – You are likely to see this frequently this semester!

• For more information, see the standard:
  – http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html
Show example

• Show in Chrome
  – Intro:
    • https://developers.google.com/web/tools/chrome-devtools
  – Show Network tab
  – Inspect a particular request
    • Request and Response Headers
    • Response data
HTTP Methods

• Most frequently used:
  – GET – retrieve information
  – HEAD – query if there is information to retrieve
  – POST – add or modify a resource's information
    • if the resource identity is not known
  – PUT – add or replace a resource's information
    • if the resource identity is known (add) or can be predicted (replace)
  – DELETE – delete information

• When choosing which HTTP methods to implement, consider:
  – Safety
  – Idempotence
HTTP Methods

• You should study and know:
  – What *safe* means
  – What *idempotent* means
    • Why idempotent is useful (hint: it has to do with failure)
  – The common HTTP methods (previous slide)
    • What they are used for
    • Whether they defined as being safe
    • Whether they defined as being idempotent

• Good resource to study:
Idempotent Methods

• Interesting test question:
  – Should HTTP DELETE be safe?
  – Should HTTP DELETE be idempotent?
  – How would you handle receiving an HTTP DELETE referring to a resource that has already been deleted?
<table>
<thead>
<tr>
<th>Method</th>
<th>Purpose?</th>
<th>Safe?</th>
<th>Idempotent?</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>Retrieve a resource</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>PUT</td>
<td>Insert or replace a resource</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>DELETE</td>
<td>Remove a resource</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>HEAD</td>
<td>Get header information only of a resource</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>POST</td>
<td>Add or modify a resource</td>
<td>N</td>
<td>N</td>
</tr>
</tbody>
</table>
HTTP *standard*

- HTTP is a *standard*.
  - How it is actually implemented depends on the developer.
  - A developer could choose to use HTTP DELETE to add records to a database.
    - Good question to ponder: What benefit is there to following standards?

- Misuse is common

- E.g. Flickr uses GET to do everything, including removing a photo from your favorites list
HTTP/2

• A new version of HTTP is being adopted.
• Read this good HTTP/2 FAQ
  – https://http2.github.io/faq/#general-questions
• Know for the exam, at a high level, what are the key differences from HTTP/1.x?
  – (There is a bullet list in the FAQ.)
Java Servlets

- What does a server do when it receives an HTTP request?
- Java Enterprise Edition (JEE) is the platform (i.e. set of classes) for building distributed systems in Java.
- HttpServlet (commonly called a "Servlet") is the class designed to process HTTP requests.

- Great Reference:
  - Head First Servlets & JSP
    by Bryan Basham; Kathy Sierra; Bert Bates
  - Ch 1 – 3
In `init()` you might:
- Initialize a database
- Open file(s)
- Register with other objects
- Connect (sockets) to other services

Source: Head First Servlets and JSP by Basham, Sierra, & Bates
You must overwrite one of:
- doGet
- doPost
- doPut
- doDelete
- doHead

Source: Head First Servlets and JSP by Basham, Sierra, & Bates
In `destroy()` you might:

• Close a database or file
• Clean up relationships to other objects
• Close (sockets)

Source: Head First Servlets and JSP by Basham, Sierra, & Bates
One Servlet Object, Multiple Threads

You implement, e.g., `doGet(HttpServletRequest request, HttpServletResponse response)`

Source: Head First Servlets and JSP by Basham, Sierra, & Bates
Model – View - Controller

MVC is discussed in the Lab 2 Video 3