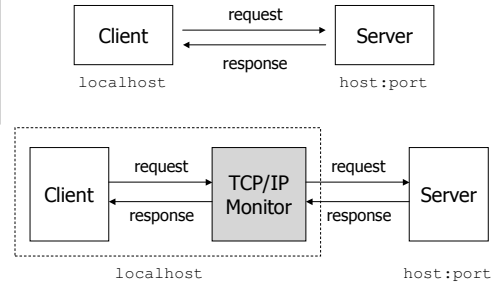


## CS320 Web and Internet Programming Handling HTTP Requests

Chengyu Sun  
California State University, Los Angeles

## TCP/IP Monitor in Eclipse ...



## ... TCP/IP Monitor in Eclipse

- ◆ Window → Preferences →  
Run/Debug → TCP/IP Monitor
- ◆ Example: monitor the access of  
<http://sun.calstatela.edu/~cysun/index.html>
  - Local Monitoring Port?? Host?? Port??
  - Browser URL??

## HTTP Request Example

*http://cs.calstatela.edu:4040/foo*

### GET /foo HTTP/1.1

Host: cs.calstatela.edu:4040  
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.0; en-US; rv:1.7.3) ...  
Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,...  
Accept-Language: en-us,en;q=0.5  
Accept-Encoding: gzip,deflate  
Accept-Charset: ISO-8859-1,utf-8;q=0.7,\*;q=0.7  
Keep-Alive: 300  
Connection: keep-alive  
Cookie: nxt/gateway.dll/uid=4B4CF072; SITESERVER=ID=f1675...

## HTTP Request

- ◆ Request line
  - Method
  - Request URI
  - Protocol
- ◆ Header
- ◆ [Message body]

## Request Methods

- ◆ Actions to be performed regarding the resource identified by the *Request URI*
- ◆ Browser
  - GET
  - POST
- ◆ Editor
  - PUT
  - DELETE
- ◆ Diagnosis
  - HEAD
  - OPTIONS
  - TRACE

## HttpServlet Methods

	→	service()
◆ GET	→	◆ doGet()
◆ POST	→	◆ doPost()
◆ PUT	→	◆ doPut()
◆ DELETE	→	◆ doDelete()
◆ HEAD	→	◆ doHead()
◆ OPTIONS	→	◆ doOptions()
◆ TRACE	→	◆ doTrace()

## Override HttpServlet Methods

- ◆ Default implementation returns an HTTP Bad Request error
- ◆ `doGet()`, `doPost()`
  - Simply call one method from the other

## Request Header Fields

◆ User-Agent	◆ Host
◆ Accept	◆ Connection/Keep-alive
◆ Accept-Charset	◆ Content-Length
◆ Accept-Encoding	◆ Cookie
◆ Accept-Language	◆ Referer

RFC2616, Section 14  
<http://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html>

## HttpServletRequest

- ◆ `getXxx()` methods
- ◆ <http://download.oracle.com/javaee/6/api/javax/servlet/http/HttpServletRequest.html>

## Example: RequestInfo

- ◆ Display some request information
  - Method, request URI, request URL ...
- ◆ Display the IP address of the client
- ◆ Determine if gzip compression supported

## Request Parameters

- ◆ Query string
  - `?param1=value1&param2=value2&...`
- ◆ Form data

## Example: Add

- ◆ Get two integers from request parameter and display the sum

## Parameter Methods

- ◆ Inherited from *ServletRequest*
  - String `getParameter( String p )`
  - Enumeration `getParameterNames()`
  - String[] `getParameterValues( String p )`
  - Map `getParameterMap()`

## HTML Forms

- ◆ `<form>`
  - action – *URL* of the server-side program that process the submitted form data
  - method – GET or POST
  - enctype
    - ◆ multipart/form-data
      - for file uploading
      - require POST

## Understand URL

- ◆ Absolute URL
- ◆ Relative URL
  - Relative to server root
  - Relative to application root
  - Relative to current URL

## `<input>`

- ◆ name
- ◆ value
- ◆ type
  - Text field ??
  - Password field ??
  - Submit button ??
  - Reset button ??
  - Check box ??
  - Radio button ??
- ◆ more types
  - File upload
  - Hidden field

## `<textarea>`

- ◆ name
- ◆ rows
- ◆ cols

## <select>

- ◆ name
- ◆ size
- ◆ multiple
- ◆ <option value="something">some text

## Example: Guest Book

My Guest Book		
John says:	Hello!	<a href="#">Edit</a>   <a href="#">Delete</a>
Jane says:	Your website looks nice.	<a href="#">Edit</a>   <a href="#">Delete</a>
Joe says:	Nice to meet you. I'm from China.	<a href="#">Edit</a>   <a href="#">Delete</a>

[Add Comment](#)

Add Comment	Edit Entry
Your name: <input type="text"/>	Your name: <input type="text" value="Jane"/>
<input type="text"/>	Your website looks nice.
<input type="button" value="Add"/>	<input type="button" value="Save"/>

## Guest Book Implementation



- ◆ Data
  - GuestBookEntry
  - List<GuestBookEntry>
- ◆ Display
  - Create some test data
- ◆ Update (i.e. Add and Edit)
  - Display form using doGet ()
  - Process form using doPost ()
  - response.sendRedirect ()

## Filter HTML Special Characters

- ◆ <: &lt;
- ◆ >: &gt;
- ◆ &: &amp;
- ◆ ": &quot;
- ◆ ': &apos;

## HTTP Request as User Input

- ◆ HTTP Request
  - Request line and header fields
  - Query string
  - Form data
- ◆ Important classes
  - HttpServletRequest and its superclass ServletRequest
- ◆ It's still Java!