

Web Application Development with Servlets

Jeff Offutt

<http://www.cs.gmu.edu/~offutt/>

SWE 432

**Design and Implementation of
Software for the Web**

Web Applications

review

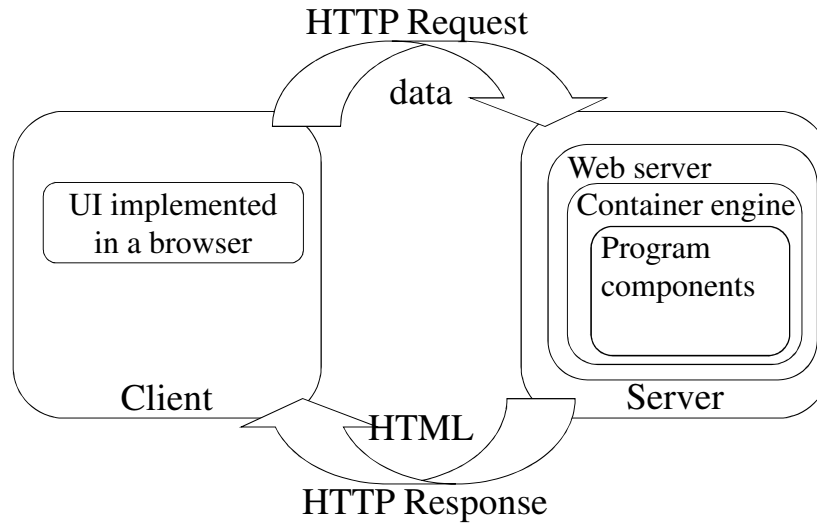
- A web application uses enabling technologies to
 1. make web site contents dynamic
 2. allow users of the system to implement business logic on the server
- Web applications allow users to affect state on the server

A web application is a program deployed on the web

An enabling technology is a mechanism that makes web pages interactive and responsive to user inputs

Server Side Processing

review



10 December 2011

© Ofutt, 2011

3

What We Will Study

review

We will learn about these concepts in the context of several technologies

- PHP ✓
- Ajax ✓
- Servlets Now ...
- JSPs

10 December 2011

© Ofutt, 2011

4

What are Servlets?

- Servlets are small Java classes that
 - Process an HTTP request
 - Return an HTTP response
- Servlet container or engine
 - Connects to network
 - Catches requests
 - Produces responses
 - Creates object instances of servlet classes
 - Hands requests to the appropriate object
- Programmers use a servlet API to write servlet classes

10 December 2011

© Oflutt, 2011

5

Servlets vs. Java Applications

- Servlets do not have a main()
 - The main() is in the server
 - Entry point to servlet is via call to a method (doGet() or doPost())
- Servlet interaction with end user is indirect via request / response object APIs
 - Actual HTTP request / response processing is handled by the server
- Servlet output is usually HTML

10 December 2011

© Oflutt, 2011

6

Servlet Container (or Engine)

- Servlet container is a plug-in for handling Java servlets
- A servlet container has five jobs:
 1. Creates servlet instance
 2. Calls `init()`
 3. Calls `service()` whenever a request is made
 1. `service()` calls a method written by a programmer to handle the request
 2. `doGet()` to handle GET requests, `doPost()` to handle POST requests
 3. More on this later ...
 4. Calls `destroy()` before killing servlet
 5. Destroys instance
- Really a mini operating system

10 December 2011

© Oflutt, 2011

7

Servlet Container (2)

When a request comes to a servlet, the servlet container does one of two things:

1. If there is an active object for the servlet, the container creates a Java thread to handle the request
2. If there is no active object for the servlet, the container instantiates a new object of that class, and the object handles the request

10 December 2011

© Oflutt, 2011

8

Servlet Container (3)

A servlet instance runs until the container decides to destroy it:

- When it gets destroyed is not specified by the servlet rules
- Most servlet containers destroy the object N minutes after the last request
- N is usually 15 or 30, and can be set by the system administrator
- Container can also be configured to never destroy a servlet object

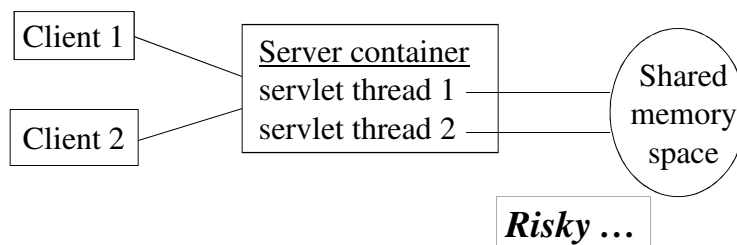
10 December 2011

© Oflutt, 2011

9

Servlet Container (4)

- What if the same servlet gets multiple requests ?
- More than one execution thread may be running at the same time, using the same memory



10 December 2011

© Oflutt, 2011

10

Servlet Container (5)

- By default, there is only one instance of a servlet class per servlet definition in the servlet container
- *Distributable* : If the application is *distributable*, there is one instance of a servlet class per virtual machine
 - Sometimes each VM is on a different computer in the cluster
 - Sometimes multiple VMs are on one computer

10 December 2011

© Oflutt, 2011

11

Allowing Concurrency (SingleThreadModel)

- Container may send multiple service requests to a single instance, using Java threads
 - Threads are Java's concurrency mechanism
- Thus, your service methods (doGet(), doPost(), etc.) should be thread-safe
 - Loosely speaking, "*thread-safe*" means that if two or more requests are operating at the same time, they will not interfere with each other
- If the service methods are not thread-safe, use the SingleThreadModel

10 December 2011

© Oflutt, 2011

12

SingleThreadModel (2)

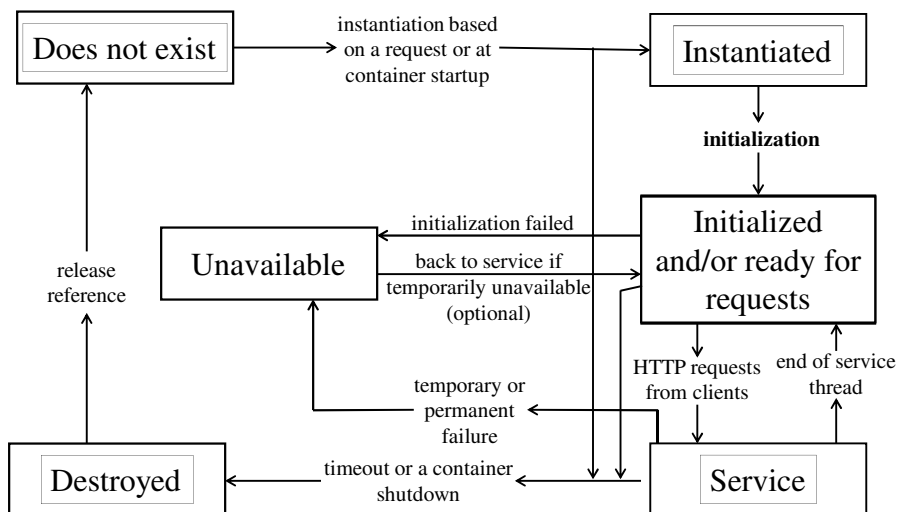
- The SingleThreadModel ensures that only one thread may execute the service() method at a time
- Containers may implement this in two ways:
 1. *Instance Pooling* : A “pool” of several servlet instances are available that do not share memory
 2. *Request Serialization* : Only one request is handled at a time
 3. *Combination* : A pool is used, and if there more requests than servlets in the pool, they are serialized
- This is resource intensive and can be slow
- Better to synchronize only the statements that might interfere with each other

10 December 2011

© Oflutt, 2011

13

Servlet Object Thread Lifecycle UML State Diagram



10 December 2011

© Oflutt, 2011

14

Common Servlet Containers

- Tomcat (open source, most common, installed on hermes)
- Oracle's WebLogic
- IBM's WebSphere (uses Tomcat)
- Adobe's JRun
- JBoss (open source)
- Wikipedia has a longer list:
http://en.wikipedia.org/wiki/List_of_Servlet_containers

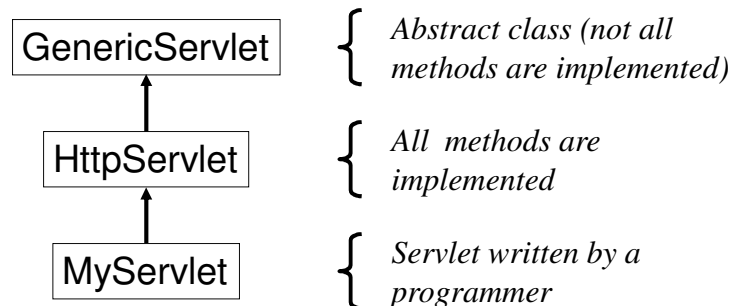
10 December 2011

© Offutt, 2011

15

Servlet API

- javax.servlet – primarily containers
- javax.servlet.http – methods to service requests



10 December 2011

© Offutt, 2011

16

Generic Servlet & HTTP Servlet

Servlets can have five methods:

1. `init()` – called when servlet starts
2. `service()` – called to process requests
3. `destroy()` – called before servlet process ends
4. `getServletConfig()` – servlet can access information about servlet container
5. `getServletInfo()` – servlet container can access info about servlet

10 December 2011

© Oflutt, 2011

17

Generic Servlet & HTTP Servlet (2)

- These methods are defined by the library classes **GenericServlet** and **HttpServlet**
- We write servlets by extending (inheriting from) them
- **GenericServlet** does not implement `service()` (it is abstract)
- **HttpServlet** extends **GenericServlet** with:
service (HttpServletRequest req, HttpServletResponse res)
throws ServletException, IOException

10 December 2011

© Oflutt, 2011

18

1. init ()

- Read configuration data
- Read initialization parameters (javax.servlet.ServletConfig)
- Initialize services:
 - Database driver
 - Connection pool
 - Logging service
- Seldom used in simple applications

2. service ()

- The entry point for the servlet – this is the method that is called from the servlet container
- Called after the initialization (init ())
- Primary purpose is to decide what type of request is coming in and then call the appropriate method
 - doGet ()
 - doPost ()

Types of HTTP Requests

- GET
- POST
- HEAD
- OPTIONS
- DELETE
- PUT
- TRACE

```
doGet ()
doPost ()
doHead ()
doOptions ()
doDelete ()
doPut ()
doTrace()
```

} same signatures
as `service()`

Types of HTTP Requests (2)

- **HttpServlet** implements these methods as “stubs” that print error messages

```
doGet () ...
```

```
{
    print ("Error HTTP 405, doGet() not implemented");
}
```

- Programmers implement services by overriding these methods
 - usually `doGet()` and `doPost()`

3) destroy ()

- Called by container before the servlet instance is killed
- The threads from the service() method are given time to terminate before destroy() is called
- Can be used to clean up the state of the servlet :
 - Un-registering a database driver
 - Closing a connection pool
 - Informing another application the servlet is stopping
 - Saving state from the servlet

4) getServletConfig ()

- Returns a ServletConfig object, which stores information about the servlet's configuration
- The ServletConfig object was passed into init() by the servlet container

5) `getServletInfo ()`

- Returns a String object that stores information about the servlet :
 - Author
 - Creation date
 - Description
 - Usage
 - ...
- This string should be formatted for human readability

10 December 2011

© Ofutt, 2011

25

Simple Servlet Example

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
public class Hello extends HttpServlet
{
    public void doGet (HttpServletRequest req,
                      HttpServletResponse res)
                      throws ServletException, IOException
    {
        res.setContentType ("text/html; charset='UTF-8'");
        PrintWriter out = res.getWriter ();
        out.println ("");
    }
}
```

10 December 2011

© Ofutt, 2011

26

Simple Servlet (2)

```
out.println("<HEAD>");
out.println("<TITLE>Servlet example</TITLE>");
out.println("</HEAD>");
out.println("<BODY>");
out.println("<P>My first servlet.");
out.println("</BODY>");
out.println("</HTML>");
out.close ();
} // end doGet()
} // end Hello
```

<http://apps-swe432.vse.gmu.edu:8080/swe432/servlet/offutt.Hello>

10 December 2011

© Offutt, 2011

27

Servlet Parameters — requests

Parameters are conveniently stored in objects

- String req.getParameter (String KEY)
 - Returns value of field with the name = KEY
 - Names are defined in HTML, and values supplied by the users
- String[] req.getParameterValues (String KEY)
 - Returns all values of KEY
 - For example checkboxes
- Enumeration req.getParameterNames ()
 - Returns an Enumeration object with a list of all parameter names
- String req.getQueryString ()
 - Returns the entire query string

10 December 2011

© Offutt, 2011

28

Servlet Parameters–Transmission

- Parameter data is the Web analog of arguments in a method call :
 - `System.out.println (“aString”);`
 - `http://www.example.com/servlet/PrintThis?arg=aString`
- Query string syntax and semantics
 - Multiple parameters are separated by ‘&’
`http://www.example.com/servlet/PrintThis?color=red&arg=aString`
 - Order of parameters does not matter
`http://www.example.com/servlet/PrintThis?arg=aString&color=red`
 - All parameter values are strings
`http://www.example.com/servlet/PrintThis?arg=&age=39`

Empty string

10 December 2011

© Oflutt, 2011

29

Servlet Parameters–Creation

- HTML forms generate query strings when submitted
- Parameter names are specified as the value of the *name* attributes in the form controls
 - `<input type=“text” name=“username” size=“35” />`
- Parameter values depend on control type
 - `<input type=“checkbox” name=“daysFree” value=“Mon” />Mon`

Value sent to the server :
daysFree=Mon

10 December 2011

© Oflutt, 2011

30

Servlet Parameters—Creation

Controls	Value
input/text input/password textarea	Text that the user has entered into the control field when the form is submitted
input/checkbox input/radio input/submit input/image button/submit	String assigned to the value attribute in the HTML tag The control must be selected or clicked for the parameter to be sent
input/hidden	String assigned to the value attribute Not rendered on the client
select	String assigned to the value attribute of the selected options, or content of any selected option for which the value is not defined

10 December 2011

© Ofutt, 2011

31

Servlet Parameters—Creation

Enter your name: `<input type="text" name="username" size="35" />`

`<p>`

Check all the days that you are free:

`<label>`

`<input type="checkbox" name="daysFree" value="Mon" />Mon`

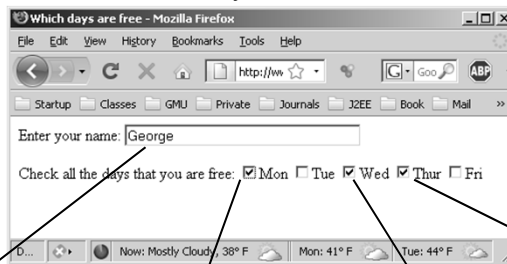
`<input type="checkbox" name="daysFree" value="Tue" />Tue`

`<input type="checkbox" name="daysFree" value="Wed" />Wed`

`<input type="checkbox" name="daysFree" value="Thur" />Thur`

`<input type="checkbox" name="daysFree" value="Fri" />Fri`

`</label>`



`username=George&daysFree=Mon&daysFree=Wed&daysFree=Thur`

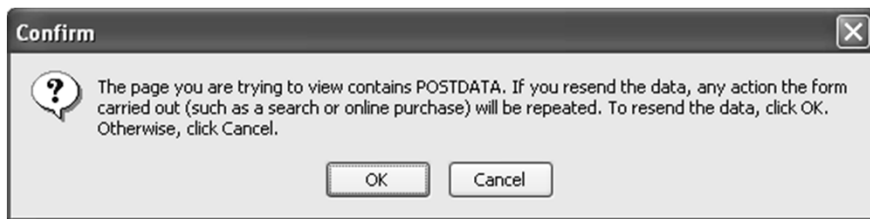
10 December 2011

© Ofutt, 2011

32

Servlet Parameters–Re-Transmission

- Most browsers give a warning before submitting POST data for the second time
 - Avoid duplicate submissions and updates
 - Avoid duplicate purchases
- Users should be very careful before overriding this hesitation
- However ... how many users understand this message?



10 December 2011

© Ofutt, 2011

33

Servlet Output – responses

Standard output is sent directly back to the client browser

- `res.setContentType` (String type)
 - “text/html; charset='UTF-8'” is an HTML page with robust encoding
- `PrintWriter res.getWriter()`
 - Use `print()` and `println()` to write HTML to browser

10 December 2011

© Ofutt, 2011

34

Servlet Performance

- Some servlets will run a lot
- Servlets run as *lightweight threads*, so are fast
- The network speeds usually dominate, but two things can add speed :
 - avoid concatenation (“+”)
 - `out.flush()` – Sends current output to user’s screen while servlet continues processing

10 December 2011

© Oflutt, 2011

35

GET and POST Requests

- An HTTP GET request is generated when the URL is entered directly
 - `doGet ()` is called from `service()`
- An HTML form can generate either a GET or a POST request
 - “... Method=POST” or “... Method=GET”
- GET requests put form data on the URL as parameters
 - `http://www ... /RunForm?NAME=Jeff&TITLE=prof`
- The length of GET parameters is limited by some browsers (usually 1024 bytes)
- POST requests put form data in body of request
- Post requests can be arbitrarily long

10 December 2011

© Oflutt, 2011

36

GET and POST Requests (2)

- Book says :
 - Use GET to retrieve data
 - Use POST to change state on server (update file or DB)
 - Use POST when there are a lot of data items
- This is a little ambiguous and incomplete ...
- Prof's suggestion :
 - Use POST when sending data to server
 - Use GET when no data is sent
- GET is also useful when the entire request needs to be bookmarked
 - Google maps

10 December 2011

© Ofutt, 2011

37

GET and POST Requests (3)

- If a servlet is primarily based on processing data and it POST, good engineering says to implement a simple doGet() method as a filler:

```
...
<BODY>
  <CENTER>A Title ...</CENTER>
  <HR>

  <P>
  You should run this from
  <A Href="http://... .html"> http://... .html</A>
</BODY>
```

10 December 2011

© Ofutt, 2011

38

Sending Mail Messages from Servlets

Common to gather data from form and send through email

- Import mail utilities:
 - `import sun.net.smtp.SmtpClient;`
- Setup mail header :
 - `send = new SmtpClient ("gmu.edu");`
 - `send.from ("offutt@gmu.edu");`
 - `send.to ("offutt@gmu.edu");`
- Send message :
 - `out = send.startMessage ();`
 - `out.println ("... message header and body ...");`
 - `out.flush ();`
 - `out.close ();`
 - `out.closeServer ();`

10 December 2011

© Offutt, 2011

39

Sending Mail Messages (2)

- This is the simplest mechanism for sending email, but is not very powerful
 - Plus, my compiler keeps complaining that it might go away soon
 - Okay ... javac has been complaining about this for since 2003 ...
- JavaMail is a collection of abstract library classes for handling mail with a number of different protocols

10 December 2011

© Offutt, 2011

40

Redirecting to Another URL from Servlets

Servlets usually generate an HTML file as a response, but sometimes you may want to send the client to a different servlet

- `res.sendRedirect ("http://apps-swe432.vse.gmu.edu:8080/ ...");`
- Do not need to set content type (`setContentType()`)
- The client will be “sent” to the specified URL
 - Server tells the client to generate another request to the new URL
 - Browser then repeats request to the new URL
 - Invisible to users ... but both requests logged in history

This is a new *control* mechanism, similar to a method call

Writing to Files from Servlets

Common job is to save data into a file

- File must be in a publicly writeable directory :
 - `/data/tomcat/swe432/WEB-INF/data/`
 - This directory is available to all of us on hermes
- Open a file, write to it, and close it:
 - `FileWriter outfile = new FileWriter`
`("/data/tomcat/swe432/WEB-INF/data/info-file.txt");`
 - `outfile.write (... the data to save ...);`
 - `outfile.close ();`
- Open a file in append mode:
 - `FileWriter outfile = new FileWriter`
`("/data/tomcat/swe432/WEB-INF/data/info-file.txt", true);`
- Remember Unix / Windows path differences !!
 - “info-file” does **NOT** equal “INFO-FILE” !!!
- Remember that we all share the same directory ... include your user name as part of the file name !!!!

Deployment Testing

- Development and deployment computers often differ
- Web apps must be tested on final deployment platform
 - Must test just as real users use it
- Issues to check for:
 - Different platforms (DOS / Unix / Linux / Mac ...)
 - File names and path names (local/nonlocal, DOS/Unix)
 - Upper case dependencies
 - Incomplete deployment
 - Compiler and runtime system version
 - Permissions (data and DB)

10 December 2011

© Oflutt, 2011

43

Servlet Summary

- Servlets are very powerful programming tools for developing robust, large, and reliable web applications
- The container engine and the servlet library insulate programmers from a lot of detailed communication issues
- The separation of the client's UI from the back-end software is very powerful, but makes debugging hard
- Lots of development tools and advanced development frameworks for servlets

10 December 2011

© Oflutt, 2011

44

Examples

<http://www.cs.gmu.edu/~offutt/classes/432/examples/servlets/>

1. Hello : Prints lots of hellos
2. Name : Accepts and prints a name from a form
3. FormHandler : Processes data from any form
4. TwoButtons : Demonstrates handling two buttons
5. ChoosAbs : Processes form data and sends through email
6. LoanCalclater : Compute time to pay off a loan
7. JOConvert : Convert values
8. JOConvert2 : Better value conversion
9. check24Online : The 24 game
10. FileLoad : Uploading files from client
11. StudInfoSys432 : Processes data from students
12. ShowRequestHeaders : Simple demonstration
13. SessionLifeCycle : Demonstrates session data
14. AttributeServlet : Prints values in session data
15. Catalog : Simple shopping cart