# Servlets Intro

Dr. Michael Whitney

** Slides influenced by both textbooks

# Servlets

Small Java classes that

- Process an HTTP request
- Return an HTTP response

Servlet container or engine

- Part of the web server
- Provides runtime environment
- Creates object instances of servlet classes
- Hands requests to appropriate object

# Servlets vs Java Applications

Servlets do not have a main()

- The main() is in the server (for example, Tomcat)
- 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

# Servlet Container

A servlet container has five jobs:

- Creates servlet instance
- Calls init()
- Calls service() whenever a request is made
  - service() calls a method written by a programmer to handle the request
  - doGet() to handle GET requests, doPost() to handle POST requests
- Calls destroy() before killing servlet
- Destroys instance

# Servlet Container

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

# Servlet Container

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

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

# Common Containers

Tomcat

Oracle’s Glassfish

Eclipse's Jetty

JBoss (open source)

http://en.wikipedia.org/wiki/List_of_Servlet_containers

# Simple Example: Plain text

```
import java.io.*;                // input and output streams
import javax.servlet.*;          // primary containers
import javax.servlet.http.*;  // methods to service requests
public class HelloWorld extends HttpServlet {
        public void doGet(HttpServletRequest request,
                        HttpServletResponse response)
                        throws ServletException, IOException {
                        PrintWriter out = response.getWriter();
                        out.println("Hello World");
} }
```

# HTML Generating Servlet

1. Tell the browser that you're sending it HTML:

   response.setContentType("text/html");

2. Modify the println statements to build a legal Web page:

   Print statements should output HTML tags.

3. Check output HTML with a formal syntax validator:

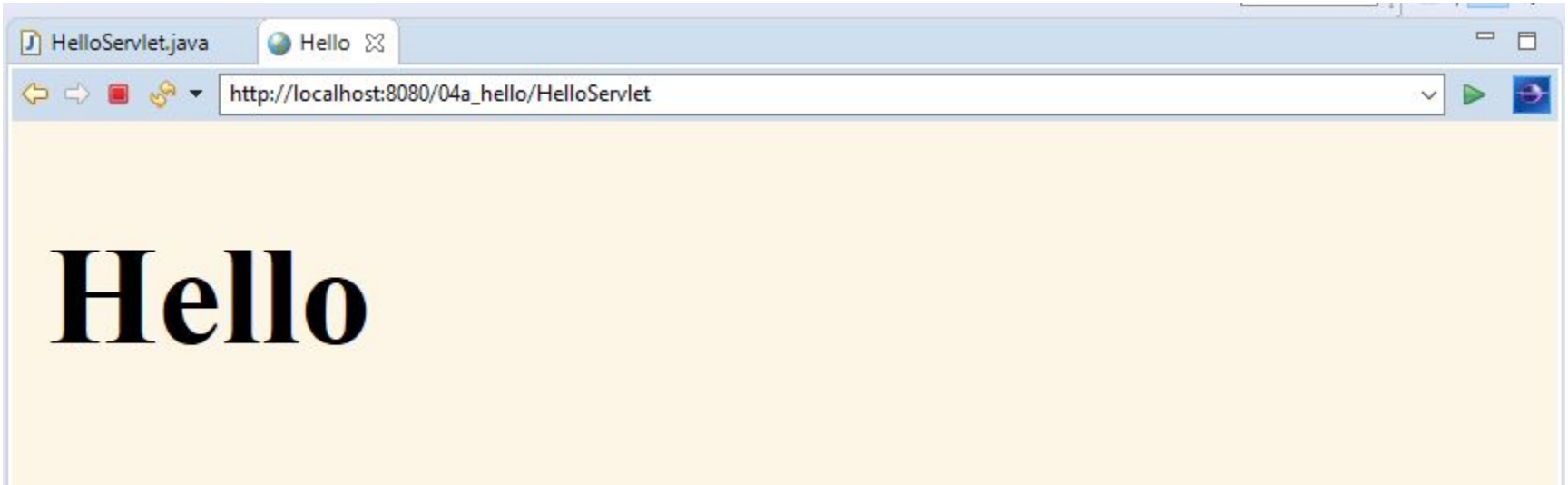
   http://validator.w3.org/

   http://www.htmlhelp.com/tools/validator/

# Simple Example: HTML

```
public class HelloServlet extends HttpServlet {
    public void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.println("<!DOCTYPE html>\n<HTML>\n" +
            "<HEAD><TITLE>Hello </TITLE></HEAD>\n"+
            "<BODY BGCOLOR=\"#FDF5E6\">\n" +
            "<H1>Hello</H1>\n" +
            "</BODY></HTML>");
} }
```

# Let's Code


HelloServlet.java
Hello
http://localhost:8080/04a_hello/HelloServlet
Hello