

Chapter 34 Servlets

1. Common gateway interface is a protocol for server-side programs to generate dynamic Web content.
2. Both GET and POST methods send requests to the Web server. The POST method always triggers the corresponding CGI program to be executed. The GET method may not cause the CGI program to be executed, if the previous same request is cached in the Web browser. Web browsers often cache Web pages so the same request can be quickly responded without contacting the Web server. The browser checks the request sent through the GET method as a URL query string. If the results for the exact same URL are cached on disk, then the previous Web pages for the URL might be displayed. To ensure a new Web page always to be displayed, use the POST method. For example, a POST method should be used if the request will actually update the database. If your request is not time-sensitive, such as finding an address of a student in the database, you should use the GET method to speed up the performance.
3. You can submit a GET request directly from a URL, but a POST request must always be submitted directly from an HTML form.
4. You should use + for space, do not use quotes.

<http://liang:8080/findscore?name=P+Yates>

5. The differences between CGI and servlets are: 1. servlets are Java programs. You can use a vast collection of Java classes to develop servlets; 2. servlets are more efficient than CGI.
6. You can display an HTML file (e.g. c:\test.html) by typing the complete file name in the Address field of Internet Explorer? Can you run a servlet by simply typing the servlet class file name?
7. You should set JDK home directory to JAVA_HOME.
8. You start Tomcat using the **startup** command in the bin directory of Tomcat and stop Tomcat using the **shutdown** command.
9. To check whether Tomcat is running, type <http://localhost:8080> from a Web browser.
- 10.

set classpath=%classpath%;c:\jakarta-tomcat-4.1.27\common\lib\servlet.jar

11. When you run Tomcat, it uses port 8080. If it is already in use, the startup fails.
12. This init, service, and destroy methods are known as life-cycle methods and are called in the following sequence.
 1. The init method is called when the servlet is first created, and is not called again as long as the servlet is not destroyed. This resembles the applet's init method, which is invoked when the applet is created, and is not invoked again as long as applet is not destroyed.
 2. The service method is invoked each time the server receives a request for the servlet. The server spawns a new thread and invokes service.
 3. The destroy method is invoked once all threads within the servlet's service method have exited or after a timeout period has passed. This method releases resources for the servlet.

13. When the servlet was first invoked, it displays

Constructor called
init called
doGet called

when the servlet was invoked for the second time, it displays

doGet called

when Tomcat is shut down

destroy called

14. If you change the content type to "html/plain" in Example 27.1, the raw HTML file would be displayed on the browser.
15. Since out is an instance of PrintWriter, all the print methods in PrintWriter do not throw IOException.
16. It returns null if the parameter does not exist.
17. Text field: e.g., Last Name

Radio button: e.g., Male Female
- Combo box: e.g., Major
<option>Computer Science <option>Mathematics
<option>English <option>Chinese </select>

List: e.g., Minor `<select name="minor" size="2" multiple>`
`<option>Computer Science <option>Mathematics`
`<option>English <option>Chinese </select></p>`

Check box: e.g., Hobby: `<input type="checkbox"`
`name="tennis"> Tennis <input type="checkbox" name="golf">`
`Golf <input type="checkbox" name="pingPong" checked> Ping`
`Pong</p>`

Text area: e.g., `<textarea name="remarks" rows="3"`
`cols="56"></textarea></p>`

18. Use `getParameter(attributeName)` method of a request (`HttpServletRequest` instance) to obtain a value for a parameter and use `getParameterValues(attributeName)` to obtain values in an array of strings.
19. The driver `.jar` file (e.g., `Oracle classes12.jar`) should be placed into `c:\jakarta-tomcat-4.1.27\common\lib`.
20. A *session* can be defined as a series of related interactions between a single client and the Web server over a period of time. To track data among requests in a session is known as *session tracking*. Three techniques for session tracking are: *using hidden values*, *using cookies*, and *using the session tracking tools from servlet API*.
21. To create a cookie, use `new Cookie(String name, String value)`. To send a cookie to the client, use `response.addCookie(cookie)`. To get cookies, use `getCookies()` from `HttpServletRequest`. Use `getName()` to return the name of the cookie. Use `getValue()` to return the value of the cookie. Use the `setValue(String value)` method to set a new value in the cookie. Use `setMaxAge(int)` to set an expiration time.
22. To get a session, use

```
HttpSession session = request.getSession(true);
```

To set an attribute, use

```
session.setAttribute(String name, Object value)
```

To get an attribute, use

```
session.getAttribute(String name)
```
23. To send images to the browser, get output stream by using `response.getOutputStream`.
24. You may embed a get request for displaying the image in the HTML content. When the HTML content is displayed, a separate get request for retrieving the image is then sent

to the server. So the text and image are obtained through two separate get requests.