JAVA A PROFUNDIDAD

El objeto request

Cuando el navegador realiza una solicitud(request) *solicitando* una página web, envía mucha información al servidor web, la cual no puede ser leída directamente porque esta información viaja como una parte de la cabecera (header) de la petición HTTP.

En las clases en video hemos tenido oportunidad de hablar sobre el protocolo http y algunos de los mensajes que se incluyen en la cabecera. En este documento vamos a ser un poco más exhaustivos.

Cabeceras HTTP

A continuación, se muestra un listado con información importante que se incluye en la cabecera de la petición http que proviene del navegador web y que podríamos llegar a utilizarla en la programación web.

Accept Este encabezado especifica los tipos MIME que el navegador u otro cliente puede manejar. Los valores image/png o image/jpeg son las dos posibilidades más comunes.
Accept-Charset Este encabezado especifica el conjunto de caracteres que el navegador puede usar para desplegar la información. Por ejemplo, ISO-8859-1.
Accept-Encoding Esta cabecera especifica el tipo de encodings que el navegador sabe cómo manejar. Los valores **gzip** o **compress** son los dos valores comúnmente posibles.
Accept-Language Esta cabecera especifica los idiomas preferidos del cliente en caso de que el servlet pueda producir resultados en más de un idioma. Por ejemplo, en, en-us, ru, es, etc.
Authorization Los clientes usan este encabezado para identificarse cuando acceden a páginas web protegidas con contraseña.
Connection Este encabezado indica si el cliente puede manejar conexiones HTTP persistentes. Las conexiones persistentes permiten al cliente u otro navegador, recibir múltiples archivos con una sola petición. Un valor **Keep-Alive** significa que las conexiones persistentes pueden ser usadas.

JAVA A PROFUNDIDAD

Content-Length Esta cabecera es aplicable sólo a las peticiones POST y lo que hace es proporcionar el tamaño de los datos enviados por POST en bytes.
Cookie Esta cabecera devuelve las cookies a los servidores que las enviaron previamente al navegador.
Host Este encabezado especifica el *host* y el *puerto* como se indica en la URL original.
If-Modified-Since Esta cabecera indica que el cliente quiere la página solo si esta ha sido cambiada después de la fecha especificada. El servidor envía un código 304 el cuál significa **No modificado** si no hay ningún cambio disponible.
If-Unmodified-Since Esta cabecera es exactamente lo contrario de **If-Modified-Since**; osea, especifica que la operación solo puede ser realizada solo si el documento es más viejo que la fecha especificada.
Referrer Este encabezado indica la URL de la referida página web. Por ejemplo, si se encuentra en la página web 1 y hace clic en un enlace a la página web 2, la URL de la página web 1 se incluye en el encabezado **Referrer** cuando el navegador solicita la página web 2.
User-Agent Este encabezado identifica el navegador o browser o cualquier otro cliente que haga la petición. Esta información puede ser usada para devolver diferente contenido a diferentes tipos de navegadores, por decir algún ejemplo.

Métodos para leer encabezados HTTP

A continuación, listo los métodos con los cuales podemos leer cabeceras HTTP en nuestros **Servlets.** Estos métodos están disponibles en el objeto HttpServletRequest.

JAVA A PROFUNDIDAD

Cookie [] getCookies()
Devuelve un arreglo que contiene todos los objetos Cookie que el cliente envía con la petición.

Enumeration getAttributeNames()
Este método devuelve un objeto **Enumeration** que contiene los nombres de los atributos disponibles en esta petición.

Enumeration getHeaderNames()
Este método devuelve un objeto **Enumeration** de todos los nombres de cabeceras que esta petición contiene.

Enumeration getParameterNames()
Devuelve un objeto **Enumeration** con objetos String que contienen los nombres de los parámetros contenidos en esta petición.

HttpSession getSession()
Devuelve la sesión actual asociada con esta petición, o bien, si la petición no tiene ninguna sesión, entonces crea una.

HttpSession getSession(boolean create)
Devuelve la sesión http actual que está asociada con la petición, o bien, si no hay ninguna sesión actual y el valor del parámetro créate es true, entonces crea una.

Locale getLocale()
Devuelve la configuración regional preferida en la que el cliente aceptará el contenido, dependiendo también del encabezado **Accept-Languague**.

Object getAttribute(String name)
Devuelve el valor del atributo referido en el name. Dicho atributo lo devuelve como un objeto, o null si no existe ningún atributo con el nombre dado. Este método toma mucho sentido en los JSP.

ServletInputStream getInputStream()
Recibe el cuerpo de la petición como dato binario usando un ServletInputStream.

JAVA A PROFUNDIDAD

String getAuthType() Devuelve el nombre del esquema de autentificación usado para proteger el servlet, por ejemplo, “BASIC” o “SSL”, o null si el JSP no está protegido.
String getCharacterEncoding() Devuelve el nombre del encoding usado en el cuerpo de la petición.
String getContentType() Devuelve el tipo MIME del cuerpo de esta petición, o null si el tipo es desconocido.
String getContextPath() Devuelve la porción de la petición URI que indica el contexto de la petición.
String getHeader(String name) Devuelve el valor de la cabecera especificada en esta petición.
String getMethod() Devuelve el nombre del método HTTP con que se realizó la petición, por ejemplo, GET, POST, PUT, DELETE, etc.
String getParameter(String name) Devuelve el valor de un parámetro contenido en la petición como un String, o null si el parámetro no existe.
String getPathInfo() Devuelve cualquier información de ruta extra que esté asociada con la URL que el cliente envía cuando realiza esta petición.
String getProtocol() Devuelve el nombre y versión del protocolo de esta petición.
String getRemoteAddr() Devuelve la dirección IP del cliente que envía la petición.

JAVA A PROFUNDIDAD

String getRemoteHost() Devuelve el nombre cualificado completo del cliente que envía esta petición.
String getRemoteUser() Devuelve el login del usuario que está haciendo esta petición, si el usuario ha sido autentificado, o devuelve null si el usuario no ha sido autentificado.
String getRequestURI() Devuelve la parte de la URL de esta solicitud desde el nombre del protocolo hasta la cadena de consulta en la primera línea de la solicitud HTTP.
String getRequestedSessionId() Devuelve el ID de sesión especificado por el cliente.
String getServletPath() Devuelve la parte de la URL de esta solicitud que llama al JSP.
String[] getParameterValues(String name) Devuelve un arreglo de String que contienen todos los valores de los parámetros dados en la petición, o null si la petición no contiene parámetros.
Boolean isSecure() Devuelve un Boolean indicando si esta petición fue hecha usando un canal seguro como HTTPS.
Int getContentLength() Devuelve la longitud, en bytes, del cuerpo de la petición y que está disponible por el input stream, o bien, devuelve -1 si la longitud no es conocida.
Int getIntHeader(String name) Devuelve el valor de el header de la petición especificada por name como un int.
int getServerPort() Devuelve el número de puerto sobre el cuál se realizó esta petición.

JAVA A PROFUNDIDAD

Ejemplo de solicitud de cabecera HTTP

El siguiente ejemplo utiliza el método getHeaderNames() de HttpServletRequest para leer la información de la cabecera HTTP. Este método devuelve un objeto Enumeration que contiene la información de la cabecera asociada con la petición HTTP actual.

Una vez que tenemos un Enumeration, podemos iterar fácilmente a través del objeto Enumeration de una manera convencional usando hasMoreElements() para determinar cuando parar y también el método nextElement para obtener cada nombre de parámetro.

```
1.  import java.io.*;
2.  import javax.servlet.*;
3.  import javax.servlet.http.*;
4.  import java.util.*;
5.
6.  // Heredamos de HttpServlet
7.  public class DisplayHeader extends HttpServlet {
8.
9.     // Manejo de las peticiones que llegan por el método GET
10.    public void doGet(HttpServletRequest request, HttpServletResponse response)
11.       throws ServletException, IOException {
12.
13.       // Establecemos el tipo de respuesta que daremos al cliente.
14.       response.setContentType("text/html");
15.
16.       PrintWriter out = response.getWriter();
17.       String titulo = "Ejemplo de petición HTTP Header";
18.       String docType =
19.          "<!DOCTYPE">\n";
20.
21.       //En este caso, el HTML de salida que estamos construyendo, es una tabla.
22.       out.println(docType +
23.          "<html>\n" +
24.          "<head><title>" + titulo + "</title></head>\n"+
25.          "<body bgcolor = \"#f0f0f0\">\n" +
26.          "<h1 align = \"center\">" + titulo + "</h1>\n" +
27.          "<table width = \"100%\" border = \"1\" align = \"center\">\n" +
28.          "<tr bgcolor = \"#949494\">\n" +
29.          "<th>Header Name</th><th>Header Value(s)</th>\n"+
30.          "</tr>\n"
31.       );
32.
33.       Enumeration headerNames = request.getHeaderNames();
34.
35.       while(headerNames.hasMoreElements()) {
36.          String paramName = (String)headerNames.nextElement();
37.          out.print("<tr><td>" + paramName + "</td>\n");
38.          String paramValue = request.getHeader(paramName);
```

```
39.
40.          out.println("<td> " + paramValue + "</td></tr>\n");
41.       }
42.       out.println("</table>\n</body></html>");
43.    }
44.
45.    // Método para manejar las peticiones por POST, en este caso, todas las
46.    // mandamos al método doGet.
47.    public void doPost(HttpServletRequest request, HttpServletResponse response)
48.       throws ServletException, IOException {
49.
50.       doGet(request, response);
51.    }
52. }
```

Ahora, llamando a este Servlet obtendremos una salida similar a esta.