HttpServletRequest

Accessing parameters of a request

An HTTP request can have a number of parameters. GET requests are encoded into the request URL. POST requests the HTTP stream carries these parameters.

The servlet API decodes these parameters, and exposes them through four methods:

- ***getParameterNames()*** *: returns all the parameter names available on the request as an Enumeration.
- ***getParameter(String)*** *: takes a string, and returns the parameter associated with the string name, if it exists.
- ***getParameteValues(String)*** *: takes a string, and returns a set of values associated with the string name in the form of an array of String.
- ***getParameterMap()*** : returns a Map whose keys are the names of the parameters, and the values of the associated parameters.

Note the parameter values are always of type String, because it is the type used in the HTTP header. It is therefore up to the application to convert these string into the appropriate types.

Accessing the elements in the HTTP header

There are several methods used to access the elements in the HTTP header.

- ***getHeaderNames():*** returns the names of the header parameters declared in this request as an Enumeration.
- ***getHeader(String), getIntHeader(String), and getDateHeader(String) :*** return the value of the parameter of the header whose name is given, in the form of a String, an int or a Date respectively.
- ***getHeaders()*** : returns all the names of the elements of the HTTP header available in this request, as an Enumeration.
- ***getMethod()*** *: returns the name of the HTTP method used for this request (GET, POST, etc ...).
- ***getCharacterEncoding()*** : returns the encoding used for this request. By default the encoding used by HTTP is ISO-8859-1, but other encodings, including UTF-8 can also be used. Note that there is a setCharacterEncoding() method that allows you to force encoding. It must be called before any access to the contents of the HTTP request.

- ***getLocale() and getLocales() :*** returns the contents of the Accept-Language header element, which indicates the default locale supported by the browser. It is from this parameter that an application can decide to send pages in French or in English, for example.
- ***getContentType() and getContentLength() :**** allow access to the MIME type of the content of the request, and the number of bytes that compose it.
- ***GetInputStream() :*** returns a ServletInputStream bit stream on the request directly, which can be processed without going through all of these methods.

Accessing the elements in a URL

The elements of the resource access URL that we are processing are exposed through three variables: contextPath, servletPath, and pathInfo. These three variables are accessible via their gettersstandards.

In addition, two variables are available: requestURI and requestURL.

- ***getRequestURI()*** : URI of the request, which is what is between the {protocol name, server name, port} group and the parameters of the request.
- ***getRequestURL()*** : returns the complete URL of the request. Note that the return type is StringBuffer, which allows you to modify this URL easily and efficiently.
- ***getContextPath()*** *: returns the context portion of the request URI. It corresponds to the path attribute of the Context element of the context.xml file of this web application, in the case of Tomcat. If the path value is /, then the contextPath is empty.
- ***getServletPath()*** : returns the path under which this resource is located, under the path of the web application. This value corresponds to the servlet-mapping element of the web.xml file of this web application. If servlet-mapping is /, then servletPath is empty.
- ***getPathInfo()*** : return what is left to return. In the case of a servlet, pathInfo is empty. In the case of a static resource (an image, or an HTML page), pathInfo is the name of this resource.

Each of these three variables, if not empty, always starts with the character /.

Note that a JSP page is a servlet, that its servletPath is named after this page and that its pathInfo is empty.

Finally, let's note that we always have:

requestURI = contextPath + servletPath + pathInfo

Accessing client settings

Three methods allow us to access the parameters of the client making the request:

- ***getRemoteHost()*** : provides us with the full name of the client. It is the name of the last proxy used if this client makes its request through a proxy.
- ***getRemoteAddr()*** : same method as the previous one, except that it returns the IP address of the client.
- ***getRemotePort()*** *: same method as the previous one, we return the client port.

Accessing security information (for knowledge)

All servlet servers expose a standard authentication mechanism, which is used to set the identity of a user. Security configuration associates users with roles, and it is these roles that the server gives rights to and imposes restrictions on. Two methods are exposed for accessing role information that an authenticated user has:

- ***getUserPrincipal()***: returns a main object that encapsulates the name of the authenticated user.
- ***isUserInRole(String)*** : returns “true” if the authenticated user that originated this request is declared in the role passed as a parameter.

Accessing a session or Cookie

A method allows access to the HTTP session in which this request is placed.

- ***getSession()*** *: returns an object of type HttpSession, detailed in the rest of this chapter. Note that calling this method creates a session if one does not already exist, which is usually undesirable. The same method exists in a second version, which takes a Boolean parameter. If this boolean is false, then no session is created. In this case the method returns null.
- ***getCookies()*** : returns an array of cookies on the client machine related to the domain name
- ***getServletContext()*** : returns an object of type ServletContext, which models the web application.
- ***getServletConfig()*** : returns an object of type ServletConfig, which encapsulates the initialization parameters of the servlet. This object exposes two methods: getInitParameterNames() and getInitParameter (String), which make it possible to

access the names of the initialization parameters on the one hand, and the associated values on the other hand.