Generating the Server Response

When a Web server responds to a client request, the typical response consists of a status line, some response headers, a blank line, and the document. A typical response looks like this:

```
HTTP/1.1 200 OK
Content-Type: text/html
Header2
: ...
...
HeaderN
: ...
(Blank Line)
<!DOCTYPE ...>
<HTML>
<HEAD>...</HEAD>
<BODY>
...
</BODY></HTML>
```

The status line contains information about HTTP protocol, status code and a short message. The only mandatory header is **`Content-Type`**, which specifies the MIME type of the document that follows, while other headers are optional.

Returning status codes

One of the main tasks of servlet is to set status code in response. Default value of status code is 200. Arbitrary status code can be set by using a method **`setStatus`** of **`HttpServletResponse`**, which can be done before sending any content to client because there is no buffering in HTTP response. **`setStatus`** uses an integer as a parameter, which is typically a constant defined in **`HttpServletResponse`**. List of status codes can be viewed in API documentation (see links). The most important examples are:

```
SC_ACCEPTED
Status code (202) indicating that a request was accepted for processing, but was
not completed.

SC_FOUND
Status code (302) indicating that the resource reside temporarily under a
different URI.

SC_NOT_FOUND
Status code (404) indicating that the requested resource is not available.
```

Two general cases that have specific methods for setting status code are:

- **`public void sendRedirect(String url)`**. This method sets 302 status code and redirect response to a URL indicated in a header **`Location`**.
- **`public void sendError(int code, String message)`**. This method sets 404 status code with a message that can be included in an HTML document.

HTTP 1.1 status codes

Complete specification of HTTP 1.1, and status codes as well, can be found in RFC 2616 document (see links). **Five general categories of status codes** are:

- **100–199**. Codes in the 100s are informational, indicating that the client should respond with some other action.
- **200–299**. Values in the 200s signify that the request was successful.
- **300–399**. Values in the 300s are used for files that have moved and usually include a **`Location`** header indicating the new address.
- **400–499**. Values in the 400s indicate an error by the client.
- **500–599**. Codes in the 500s signify an error by the server.

Redirecting to another Web page (or Web site) from servlets can be done by using a method **`sendRedirect`** from **`HttpServletResponse`**.

Benefits of using redirection from servlets (Web applications) are:

- **Computing address for redirecting a client** - especially if there is a need to do some data processing and redirect based on that processing.
- **Tracking user behavior** - useful for tracking how user leave Web site and go to other Internet addresses.
- **Performing side effects** - setting additional options by using cookies.

Setting HTTP response headers

Setting the HTTP response headers often goes together with setting the status codes in the status line. For example, status codes (300 through 307) have an accompanying **`Location`** header, and a 401 (**Unauthorized**) code always includes an accompanying **`WWW-Authenticate`** header. Other use of HTTP response headers relate to specifying cookies, supplying page modification date (for client-side caching), or instructing browser to reload the page in a specific interval.

The most used methods available in **`HttpServletResponse`** for setting HTTP headers are:

- **`setHeader(String headerName, String headerValue)`** - sets the response header with the specified name to the given value
- **`setContentType(String mimeType)`** - sets the **`Content-Type`** header, which is used for informing the client about the content to be sent (HTML, images, Excel worksheets, etc.).
- **`addCookie(Cookie c)`** - inserts a cookie into the **`Set-Cookie`** header. It is normal to have multiple Set-Cookie lines in HTTP response.

Understanding of HTTP response headers can increase effectiveness of servlet based Web applications. However, it is better to explicitly check the HTTP version with the method **`getRequestProtocol`** before using HTTP-1.1-specific headers.

Common HTTP response headers

The most used response headers are:

- **`Allow`** - specifies the request methods (GET, POST, etc.) that the server supports.
- **`Cache-Control`** - tells the browser or other client the circumstances in which the response document can safely be cached.
- **`Content-Disposition`** - lets you request that the browser ask the user to save the response to disk in a file of the given name:
  `Content-Disposition: attachment; filename=some-file-name`.
- **`Content-Encoding`** - indicates the way in which the page was encoded during transmission.
- **`Content-Language`** - signifies the language in which the document is written.
- **`Content-Type`** - gives the MIME (Multipurpose Internet Mail Extension) type of the response document. Setting this header is so common that there is a special method **`setContentType`** in **`HttpServletResponse`**.
- **`Expires`** - specifies the time at which the content should be considered out-of-date and thus no longer be cached.
- **`Last-Modified`** - indicates when the document was last changed.
- **`Location`** - notifies the browser of the document address for all responses that have a status code in the 300s.
- **`Set-Cookie`** - specifies a cookie associated with the page by using **`addCookie`** method of **`HttpServletResponse`**.

Creating multi-media content from servlets

Servlets should perform the following two main steps to build multi-media content:

1. Inform the browser of the content type to be sent. To do this, servlets set the **`Content-Type`** response header by using the **`setContentType`** method of **`HttpServletResponse`**.
2. Send the output in the appropriate (announced) format. This format varies among document types, which is in most cases in binary format. In these cases, servlets get the raw output stream by using the **`getOutputStream`** method, rather than getting a **`PrintWriter`** by using **`getWriter`**, which is used for sending HTML or text formatted output.

Common MIME types

MIME types are used for specifying the type of the content sent as response from servlet. This is important because servlets are commonly used to connect other types of application to Web, which may require different type of content, such as PDF documents, various types of images, MS Word or Excel documents, multimedia files such as MPEG, etc.

The most common MIME types are:

Type	**Meaning**
application/msword	Microsoft Word document
application/pdf	Acrobat (.pdf) file
application/postscript	PostScript file
application/vnd.ms-excel	Excel spreadsheet
application/x-java-archive	JAR file
application/zip	Zip archive
image/gif	GIF image
text/html	HTML document
text/plain	Plain text
text/xml	XML
video/mpeg	MPEG video clip

New types are regularly registered, so it is advice to check the list of available MIME types before using them in Web applications.

Literature and Links

[1] **The Apache Tomcat**. http://tomcat.apache.org/

[2] Marty Hall and Larry Brown. *Core Servlets and JavaServer Pages, Free Online Version of Second Edition*. http://pdf.coreservlets.com/. Chapter 6: Generating the Server Response: HTTP Status Codes.

[3] Marty Hall and Larry Brown. *Core Servlets and JavaServer Pages, Free Online Version of Second Edition*. http://pdf.coreservlets.com/. Chapter 7: Generating the Server Response: HTTP Response Headers.

[4] https://tomcat.apache.org/tomcat-9.0-doc/servletapi/overview-summary.html

[5] https://tomcat.apache.org/tomcat-9.0-doc/servletapi/index.html.

[6] **Interface HttpServletRequest**.
https://docs.oracle.com/javaee/6/api/javax/servlet/http/HttpServletRequest.html.

[7] **Interface HttpServletResponse**.

https://docs.oracle.com/javaee/6/api/javax/servlet/http/HttpServletResponse.html.

[8] **RFC 2616. Hypertext Transfer Protocol -- HTTP/1.1**. https://www.ietf.org/rfc/rfc2616.txt.