JSP - SERVER RESPONSE

When a Web server responds to a HTTP request to the browser, the response typically 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 consists of the HTTP version *HTTP/1.1intheexample*, a status code *200intheexample*, and a very short message corresponding to the status code *OKintheexample*.

Following is a summary of the most useful HTTP 1.1 response headers which go back to the browser from web server side and you would use them very frequently in web programming:

Header	Description
Allow	This header specifies the request methods *GET, POST, etc.* that the server supports.
Cache-Control	This header specifies the circumstances in which the response document can safely be cached. It can have values **public, private** or **no-cache** etc. Public means document is cacheable, Private means document is for a single user and can only be stored in private *nonshared* caches and no-cache means document should never be cached.
Connection	This header instructs the browser whether to use persistent in HTTP connections or not. A value of **close** instructs the browser not to use persistent HTTP connections and **keep-alive** means using persistent connections.
Content-Disposition	This header lets you request that the browser ask the user to save the response to disk in a file of the given name.
Content-Encoding	This header specifies the way in which the page was encoded during transmission.
Content-Language	This header signifies the language in which the document is written. For example en, en-us, ru, etc.
Content-Length	This header indicates the number of bytes in the response. This information is needed only if the browser is using a persistent *keep − alive* HTTP connection.
Content-Type	This header gives the MIME *MultipurposeInternetMailExtension* type of the response document.
Expires	This header specifies the time at which the content should be considered out-of-date and thus no longer be cached.

Last-Modified	This header indicates when the document was last changed. The client can then cache the document and supply a date by an **If-Modified-Since** request header in later requests.
Location	This header should be included with all responses that have a status code in the 300s. This notifies the browser of the document address. The browser automatically reconnects to this location and retrieves the new document.
Refresh	This header specifies how soon the browser should ask for an updated page. You can specify time in number of seconds after which a page would be refreshed.
Retry-After	This header can be used in conjunction with a 503 *ServiceUnavailable* response to tell the client how soon it can repeat its request.
Set-Cookie	This header specifies a cookie associated with the page.

The HttpServletResponse Object:

The response object is an instance of a javax.servlet.http.HttpServletResponse object. Just as the server creates the request object, it also creates an object to represent the response to the client.

The response object also defines the interfaces that deal with creating new HTTP headers. Through this object the JSP programmer can add new cookies or date stamps, HTTP status codes etc.

There are following methods which can be used to set HTTP response header in your servlet program. These method are available with *HttpServletResponse* object which represents server response.

S.N.	Method & Description
1	**String encodeRedirectURL*Stringurl*** Encodes the specified URL for use in the sendRedirect method or, if encoding is not needed, returns the URL unchanged.
2	**String encodeURL*Stringurl*** Encodes the specified URL by including the session ID in it, or, if encoding is not needed, returns the URL unchanged.
3	**boolean containsHeader*Stringname*** Returns a boolean indicating whether the named response header has already been set.
4	**boolean isCommitted** Returns a boolean indicating if the response has been committed.
5	**void addCookie*Cookiecookie*** Adds the specified cookie to the response.
6	

void addDateHeader*Stringname, longdate*

Adds a response header with the given name and date-value.

7

void addHeader*Stringname, Stringvalue*

Adds a response header with the given name and value.

8

void addIntHeader*Stringname, intvalue*

Adds a response header with the given name and integer value.

9

void flushBuffer

Forces any content in the buffer to be written to the client.

10

void reset

Clears any data that exists in the buffer as well as the status code and headers.

11

void resetBuffer

Clears the content of the underlying buffer in the response without clearing headers or status code.

12

void sendError*intsc*

Sends an error response to the client using the specified status code and clearing the buffer.

13

void sendError*intsc, Stringmsg*

Sends an error response to the client using the specified status.

14

void sendRedirect*Stringlocation*

Sends a temporary redirect response to the client using the specified redirect location URL.

15

void setBufferSize*intsize*

Sets the preferred buffer size for the body of the response.

16

void setCharacterEncoding*Stringcharset*

Sets the character encoding *MIMEcharset* of the response being sent to the client, for example, to UTF-8.

17

void setContentLength*intlen*

Sets the length of the content body in the response In HTTP servlets, this method sets the HTTP Content-Length header.

18

void setContentType*Stringtype*

Sets the content type of the response being sent to the client, if the response has not been committed yet.

19

void setDateHeader*Stringname, longdate*

Sets a response header with the given name and date-value.

20

void setHeader*Stringname, Stringvalue*

Sets a response header with the given name and value.

21

void setIntHeader*Stringname, intvalue*

Sets a response header with the given name and integer value.

22

void setLocale*Localeloc*

Sets the locale of the response, if the response has not been committed yet.

23

void setStatus*intsc*

Sets the status code for this response.

HTTP Header Response Example:

Following example would use **setIntHeader** method to set **Refresh** header to simulate a digital clock:

```
<%@ page import="java.io.*,java.util.*" %>
<html>
<head>
<title>Auto Refresh Header Example</title>
</head>
<body>
<center>
<h2>Auto Refresh Header Example</h2>
<%
   // Set refresh, autoload time as 5 seconds
   response.setIntHeader("Refresh", 5);
   // Get current time
   Calendar calendar = new GregorianCalendar();
   String am_pm;
   int hour = calendar.get(Calendar.HOUR);
   int minute = calendar.get(Calendar.MINUTE);
   int second = calendar.get(Calendar.SECOND);
   if(calendar.get(Calendar.AM_PM) == 0)
      am_pm = "AM";
   else
```

```
      am_pm = "PM";
   String CT = hour+":"+ minute +":"+ second +" "+ am_pm;
   out.println("Current Time is: " + CT + "\n");
%>
</center>
</body>
</html>
```

Now put the above code in main.jsp and try to access it. This would display current system time after every 5 seconds as follows. Just run the JSP and wait to see the result:

```
                    Auto Refresh Header Example

Current Time is: 9:44:50 PM
```

To become more comfortable with other methods you can try few more above listed methods in the same fashion