Experiment -3

Problem 1: Setting Integer Header with `setIntHeader.`

To set an integer value for an HTTP response header using setIntHeader.

1. Create an HTTP Servlet.
2. Within the doGet or doPost method: a. Obtain the HttpServletResponse object. b. Use setIntHeader to set an integer value for a specific header (e.g., "Custom-Integer-Header"). c. Send the response.

Definition: The `setIntHeader()` method is used to set a response header with a specific name and an integer value. If the header already exists, the new value overwrites the previous one.

Code:

```
package Header;

import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;

/**
 * Servlet implementation class IntHeaderServlet
 */
@WebServlet("/IntHeader")
public class IntHeader extends HttpServlet {
	private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public IntHeader() {
        super();
        // TODO Auto-generated constructor stub
    }

	/**
	 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse
response)
	 */
```

```
        protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
                // TODO Auto-generated method stub
                //response.getWriter().append("Served at:
").append(request.getContextPath());
                // STEP 1: set integer header
        response.setIntHeader("Custom-Integer-Header", 101);

        // STEP 2: send response
        response.setContentType("text/html");
        PrintWriter pw = response.getWriter();
        pw.println("<h2>Integer Header Set</h2>");
          }

}
```

Output:

Problem 2: Setting String Header with `setHeader.`

To set a string value for an HTTP response header using setHeader.

1. Create an HTTP Servlet.
2. Within the doGet or doPost method: a. Obtain the HttpServletResponse object. b. Use setHeader to set a string value for a specific header (e.g., "Custom-String-Header"). c. Send the response.

Definition: The `setHeader()` method is the standard way to set a response header with a string value. Like its integer counterpart, it overwrites any existing value for that header name.

Code:

```
package Header;

import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
```

Experiment -3

```
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;

/**
 * Servlet implementation class StringHeaderServlet
 */
@WebServlet("/StringHeader")
public class StringHeader extends HttpServlet {
	private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public StringHeader() {
        super();
        // TODO Auto-generated constructor stub
    }

	/**
	 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse
response)
	 */
	protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
		// TODO Auto-generated method stub
		//response.getWriter().append("Served at: ").append(request.getContextPath());
		  // set string header
        response.setHeader("Custom-String-Header", "Hello-Eclipse");

        response.setContentType("text/html");
        PrintWriter pw = response.getWriter();
        pw.println("<h2>String Header Set</h2>");
	}


}
```

Output:

Experiment -3

Problem 3: Testing Content-Type Header.

To set the Content-Type header using both setIntHeader and setHeader methods and observe the differences.

1. Create an HTTP Servlet.
2. Within the doGet or doPost method: a. Obtain the HttpServletResponse object. b. Use setIntHeader to attempt setting the Content-Type header with an integer value. c. Use setHeader to set the Content-Type header with a string value like "text/plain" or "application/json". d. Compare the results in the response headers using browser developer tools or an HTTP client (like cURL or Postman).

Definition: This experiment observes how `setIntHeader` and `setHeader` interact with standard headers like `Content-Type`. Note that `Content-Type` expects a MIME type (String).

Code:

```
package Header;

import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;

/**
 * Servlet implementation class ContentTypeTest
 */
@WebServlet("/ContentTypeTest")
public class ContentTypeTest extends HttpServlet {
	private static final long serialVersionUID = 1L;
```

Experiment -3

```
        /**
         * @see HttpServlet#HttpServlet()
         */
        public ContentTypeTest() {
            super();
            // TODO Auto-generated constructor stub
        }

            /**
             * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse
response)
             */
            protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
                    // TODO Auto-generated method stub
                    //response.getWriter().append("Served at:
").append(request.getContextPath());
                     // Wrong way
            response.setIntHeader("Content-Type", 123);

            // Correct way
            response.setHeader("Content-Type", "text/plain");

            PrintWriter pw = response.getWriter();
            pw.println("Content-Type Header Testing");
            }

    }
```

Output:

Experiment -3

Problem 4: Implement Error Handling and Exception Scenarios.

Handle potential exceptions or errors while using setIntHeader and setHeader.

1. Create an HTTP Servlet.
2. Introduce scenarios such as passing invalid input or incorrect header names/values to setIntHeader and setHeader.
3. Implement try-catch blocks to handle potential exceptions that may arise due to incorrect usage.
4. Log or display appropriate error messages if exceptions occur.

Definition: While header methods rarely throw checked exceptions, they can throw `IllegalStateException` if called after the response has been committed (e.g., after the buffer is flushed).

Code:

```
package Header;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;

/**
 * Servlet implementation class HeaderError
 */
@WebServlet("/HeaderError")
public class HeaderError extends HttpServlet {
	private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public HeaderError() {
        super();
        // TODO Auto-generated constructor stub
    }

	/**
	 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse
response)
	 */
```

Experiment -3

```
        protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
                // TODO Auto-generated method stub
                //response.getWriter().append("Served at: ").append(request.getContextPath());
                 PrintWriter pw = response.getWriter();

            try {
                response.setHeader(null, "test");
            } catch (Exception e) {
                pw.println("Error: " + e.getMessage());
            }

            try {
                response.setIntHeader("Content-Type", -10);
            } catch (Exception e) {
                pw.println("<br>Error: " + e.getMessage());
            }
        }

}
```

Output:

Problem 5: Impact on Response Size and Performance

Measure and compare the impact on response size and performance when setting headers using setIntHeader and setHeader.

1. Create an HTTP Servlet that generates a sizable response (JSP page, HTML content).
2. Measure the response size and performance (time taken to generate and send the response) with headers set using setIntHeader.
3. Repeat the measurement with headers set using setHeader.

Experiment -3

4. Analyze the differences in response size and performance between the two approaches.

Definition: This test measures the time (in nanoseconds) taken to apply headers and the final content size.

Code:

```
package Header;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.io.PrintWriter;

/**
 * Servlet implementation class Performance
 */
@WebServlet("/Performance")
public class Performance extends HttpServlet {
	private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public Performance() {
        super();
        // TODO Auto-generated constructor stub
    }

	/**
	 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse
response)
	 */
	protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
		response.setContentType("text/html");
		PrintWriter pw = response.getWriter();

		long start = System.currentTimeMillis();

        response.setHeader("Test-Header", "Performance");
        for (int i = 1; i <= 10000000; i++) {
		for(int j = 1; i <= 10000000; i++);
```

```
            pw.print("");
        }

        long end = System.currentTimeMillis();
        pw.println("<br><h2>Time Taken: " + (end - start) + " ms</h2>");
            }

}
```

Output:

Time Taken: 8 ms