Servlet

* Servlet Collaboration
  + We have two ways to collaborate with our servlet:
    1. RequestDispatcher
       - It is for receiving the request of the object that defines it and the send that request to a resource such as Html, Servlet or JSP file.
       - In RequestDispatcher we have use two method to define its interface.
         * forward() method – from the term forward, it is a method that forward the request of a servlet to a resource on the side of the server.
         * include() method – include the entire resource such as Html, servlet or JSP file in response.
    2. sendRedirect() method
       - This method is use to locate another resource and implement there our response.
* Servlet Lifecycle
  + Instantiation => constructor call

- creates a service for the client request.

* + Initialization => init() method

- this method is invoked only once and the servlet will get to be initialized.

* + Request handling => service() method

- service method is called for every service request that is received. It performs the needed logical processes to be able to allow a request to be sent and be given a response.

* + Destruction=> destroy() method

- destroy method is called before a servlet is removed.

* Servlet – Error Handling
  + Use to able to control the error that will be appearing in the Web client.
  + We can use the syntax <error-page/> elements.
  + Different <error-page/> element.
    1. <error-code>
       - It is optional and used as a valid error code.
    2. <exception-type>
       - It is optional and use just like java exception type as an exception in HTTP.
    3. <location>
       - Use to redirect our page to a resource whenever our page responds to an error. The destination or the location may be a servlet or JSP page.
  + Different type of error a Servlet we need to throw:
    1. ServletException
    2. IOException
    3. RuntimeException
    4. Other exception that needs servletException to wrap it up.
* <https://www.javamex.com/tutorials/servlets/http_status_code.shtml>
* DDL and DML sql statements:
  + Data definition language—used to manipulate the structure of the database.
  + Data manipulation language—used to manipulate the data in the database.
* HttpServlet

- handles and generates HTTP responses that are in web containers. There are many “application server” where servlet can be hosted, some examples of these are Apache Tomcat, Oracle GlassFish etc. Use the service() method to call on HTTP request methods like doGet(), doPost(), doPut() etc. Each of the doXXX() methods have two parameters, the HttpServletRequest and HttpServletResponse. The HttpServletRequest is the request sent by the client to the server, on the other hand HttpServletResponse is the reponse from the server sent back to the client.

PHP

* HEADER
  + Raw HTTP header will be send.
  + Two types of header calls:
    - We have the first type that have the “HTTP/” as the beginning statement, we use this type of calls often when we want to know the status code error that will be sent.

Ex. header(“HTTP/2.0 404 Forbidden”)

* + - The second type is the “Location:” returns the status code 302 which means redirect.

Ex. header(“Location:http://www.courseweb.com”);

* + Replace
    - It is optional, it ask whether he should replace the header if the previous header is the same as the present header or just add the header.
  + http\_response\_code
    - if the header is empty it will force the response to be in a specified value.
* Superglobal
  + A “*built-in variable*”.
  + Types of superglobal variables:
    - $GLOBALS
      * It references to the global scope helping us to use variable that is outside the function.
        + Ex.

$exp=”my name”

Function exmpl(){

echo $GLOBALS[“exp”];

}

* + - $\_SERVER
      * It provides information about the paths, header and script locations.
      * Elements you may find in $\_SERVER:
        + PHP\_SELF

Element containing the PHP.

* + - * + GATEWAY\_INTERFACE

Gets the Common Gateway Interface(CGI) specification revision to the server.

* + - * + SERVER\_ADDR

The server IP address where the script is running.

* + - * + SERVER\_NAME

The server host name IP address where the script is running.

* + - * + SERVER\_SOFTWARE

Answer the request, and gives back the version of the Webserver software and also its name.

* + - * + SERVER\_PROTOCOL

Gives information about the connection use.

* + - * + REQUEST\_METHOD

Ask the request method being used such as GET, HEAD POST or PUT.

* + - * + REQUEST\_TIME

Time when the request is started.

* + - * + QUERY\_STRING

Retrieved the information about the query string if we use the query string to access a page.

* + - * + HTTP\_ACCEPT

Information about the accept header.

* + - * + HTTP\_ACCEPT\_CHARSET

Information about Accept\_Charset header.

* + - * + HTTP\_HOST

Information about the Host header.

* + - * + HTTP\_REFERER

Gives information about the URL of the page.

* + - * + HTTPS

Ask if the secure HTTP queried the given script.

* + - * + REMOTE\_ADDR

Gives the IP address information of the user that is currently visiting the page.

* + - * + REMOTE\_PORT

Gives the port information of the user that communicates to the server to access the page.

* + - * + SCRIPT\_FILENAME

Gives information about the location of the script that is currently running.

* + - * + SERVER\_ADMIN

Gives information about the admin’s given value.

* + - * + SERVER\_PORT

Gives information about the port of the server.

* + - * + SERVER\_SIGNATURE

Gives back the Information about the version of the server and name of the virtual host.

* + - * + PATH\_TRANSLATED

Gives the information about the location of the current script running.

* + - * + SCRIPT\_NAME

Gives the location of the current script location.

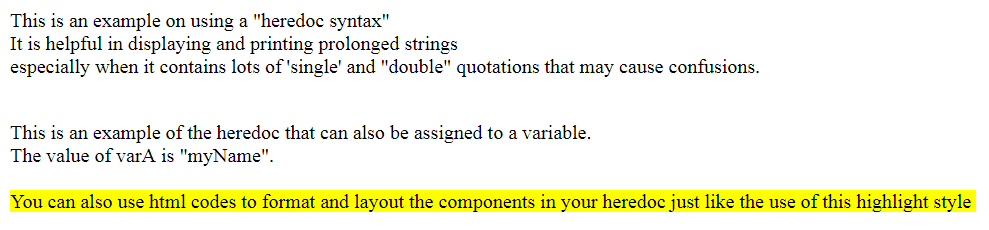
* + - * + SCRIPT\_URI

The current page URI.

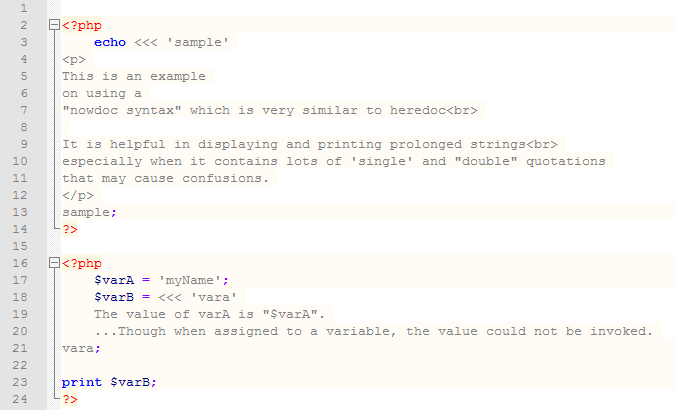
* + - $\_GET
      * Able to collect the data that is being sent in the URL.
    - $\_POST
      * Used to collect the data of HTML form when we submit and also use for variable to be pass widely.
    - $\_FILES
      * For uploading an associative array of item via POST method.
    - $\_COOKIE
      * An array of variable that the script might be needed and passed in the script through cookies.
    - $\_SESSION
      * Contain the functionalities of all the session.
    - $\_REQUEST
      * For collecting information of the inputted data after submitting it in HTML form.
    - $\_ENV
      * It is another array of variable that will be used by the script throughout environment method.
* Heredoc
* Allows us to print a lengthy string of text or assign it to a variable. It helps delimit large amounts of strings. It uses the syntax <<< followed by any identifier which processes all texts inside it until it is closed with the same identifier used.



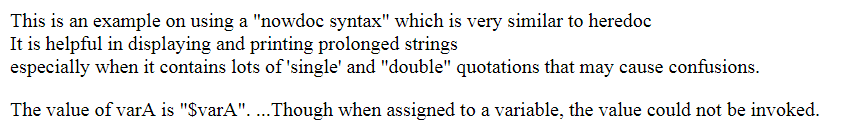
This is the output of the following code above:



* Nowdoc
* Functions the same as the heredoc which uses the same syntax <<< except that its identifier is enclosed in a single quotation.



The output of the code above is the following:



NOTE:

* Array and object are the same it depends on how you implement it.

Ex. Object use the syntax $object[‘value’] and Array use the Syntax $array[10].

* @ symbol is used to avoid any warning.
* URL rewriting is a good way for administrator to prevent data from hacking.

References: