

# Introduction to PHP

## Session 1



# Objectives

- ◆ *Explain the history of PHP*
- ◆ *Identify the need for PHP*
- ◆ *Explain PHP tools and setup*
- ◆ *Explain a simple PHP script*
- ◆ *Explain User Input/Output (I/O)*
- ◆ *Explain the use of PHP to generate HTTP headers*
- ◆ *Describe passing of variables using Uniform Resource Locator (URL)*

## ◆ PHP

- ◆ Stands for Hypertext Preprocessor
  - ◆ Is an open source scripting language
  - ◆ Is used for developing dynamic Web pages
  - ◆ Is embedded in the HyperText Markup Language (HTML)
- 
- ◆ PHP scripts are executed on the Web server

## ◆ 1994

- ◆ PHP was created by Rasmus Lerdorf
- ◆ Later, incorporated with Form Interpreter (FI) to create PHP/FI
- ◆ PHP/FI enables:
  - ◆ Communication with database
  - ◆ Development of dynamic Web application

## ◆ 1997

- ◆ PHP/FI 2.0 version released
- ◆ Lack of features led to the development of PHP 3.0
- ◆ PHP 3.0 provided support for:
  - ◆ Object oriented syntax
  - ◆ Different databases
  - ◆ Protocols
  - ◆ Application Programming Interfaces (APIs)

## ◆ 2000

- ◆ PHP 4.0 version released
- ◆ Features supported in PHP 4.0 are as follows:
  - ◆ Multiple Web servers
  - ◆ HTTP session
  - ◆ Output buffering
  - ◆ Security for user inputs

## ◆ 2004

- ◆ PHP 5.0 version released

## ◆ 2011

- ◆ Current version PHP 5.3.6 released

- ◆ Is a server-side scripting language
- ◆ Advantages are as follows:
  - ◆ Easy to learn, use, and implement
  - ◆ Freely available
  - ◆ Customizable
  - ◆ Executed on any Web server on any platform

- ◆ Uses of PHP are as follows:
  - ◆ **Application Control** – is used to control access logging for HTTP servers
  - ◆ **Database Access** – is used to read and write to any database using Structured Query Language (SQL) or Open Database Connectivity (ODBC)
  - ◆ **File Access** – is used for file and directory maintenance, generate files in Portable Document File (PDF) and HTML formats, and to process eXtensible Markup Language (XML) data

- ❖ **Graphics** – is used to create graphics, charts, and generate images in GIF and PNG formats
- ❖ **Server-Side Scripting** – is used to implement server-side scripting using PHP parser, Web server and Web browser
- ❖ **Command Line Scripting** – is used to execute scripts on Unix/Linux platforms
- ❖ **Desktop Applications** – is used to create GUI-based desktop applications



- ◆ Are text editors
- ◆ Are used for developing and designing Web pages
- ◆ Are implemented after installation of PHP

- ◆ Are programming text editors that enable fast development of Web sites

Table lists tools used for developing dynamic Web sites

Tool	Description
PHPDebugger DBG	Enables step by step execution and debugging of a PHP script without changing the PHP code
ionCube Standalone PHP Encoder	Protects the PHP code and ensures security and runtime performance
Codelock	Enables you to encrypt both PHP and HTML code and protect Web pages
PHing	Is a PHP build system, which enables you to design your Web application in a structured manner
NuSphere PHPEd	Is an Integrated Development Environment (IDE) for PHP and a complete platform for developing PHP based Web applications. It enables you to create, debug, profile, deploy, and integrate PHP code

Tool	Description
xored:WebStudio	Is an IDE for PHP. Built on Eclipse platform, this tool comprises a set of Eclipse editing, debugging, and deployment tools
PHPmole	Is a combination of Dreamweaver and Microsoft Visual Studio and runs on a GNOME platform to work with PHP
Simplewire PHP SMS Software Development Kit (SDK)	Enables to embed messaging services into an application, which can be sent to mobile devices
Quanta Plus Web Development Environment	Is a Web development environment to edit XML, HTML, PHP, and other text based Web documents
K PHP Develop	Is an integrated Web development tool with different modules
gedit	Is a GNOME based text editor for writing PHP scripts

- ◆ The installation of PHP requires:
  - ◆ A Web server
  - ◆ A database
- ◆ Web servers supported are as follows:
  - ◆ Internet Information Services (IIS)
  - ◆ Apache
  - ◆ Zeus
- ◆ Databases supported are as follows:
  - ◆ DB2
  - ◆ MSSQL
  - ◆ MySQL
  - ◆ Oracle
  - ◆ PostgreSQL

- ◆ To install PHP, perform the following steps:

## Step 1

Download the `php-5.3.6.tar.gz` file from <http://www.php.net/downloads.php>

## Step 2

Right-click `php-5.3.6.tar.gz` files and select `Extract Here`. The contents are extracted to the `php-5.3.6` folder

## Step 3

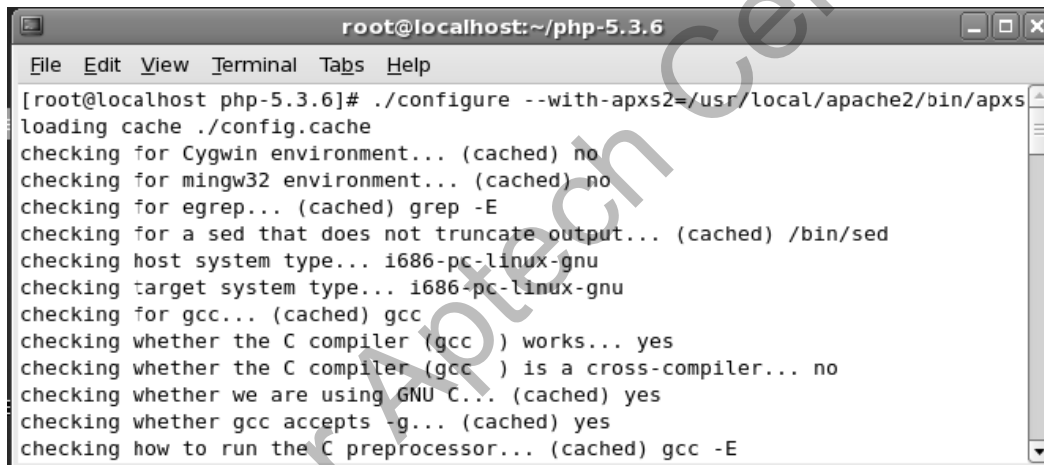
Right-click `php-5.3.6` folder and select `Open In Terminal`

## Step 4

To configure the source code of PHP, enter the following at the command prompt:

```
./configure --with-apxs2=/usr/local/apache2/bin/apxs
```

Displays the following output:



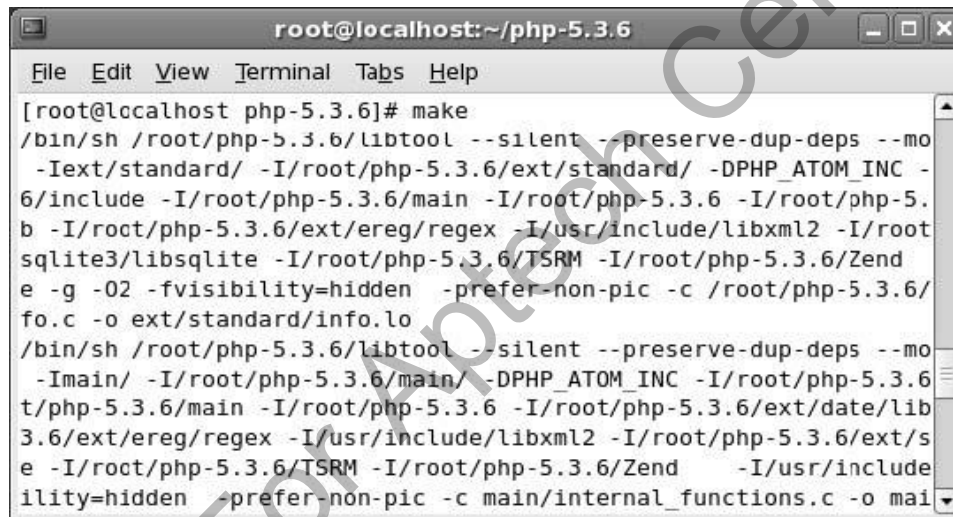
```
root@localhost:~/php-5.3.6
File Edit View Terminal Tabs Help
[root@localhost php-5.3.6]# ./configure --with-apxs2=/usr/local/apache2/bin/apxs
loading cache ./config.cache
checking for Cygwin environment... (cached) no
checking for mingw32 environment... (cached) no
checking for egrep... (cached) grep -E
checking for a sed that does not truncate output... (cached) /bin/sed
checking host system type... i686-pc-linux-gnu
checking target system type... i686-pc-linux-gnu
checking for gcc... (cached) gcc
checking whether the C compiler (gcc) works... yes
checking whether the C compiler (gcc) is a cross-compiler... no
checking whether we are using GNU C... (cached) yes
checking whether gcc accepts -g... (cached) yes
checking how to run the C preprocessor... (cached) gcc -E
```

## Step 5

To build the compiled files, enter the following at the command prompt:

```
make
```

Displays the following output:



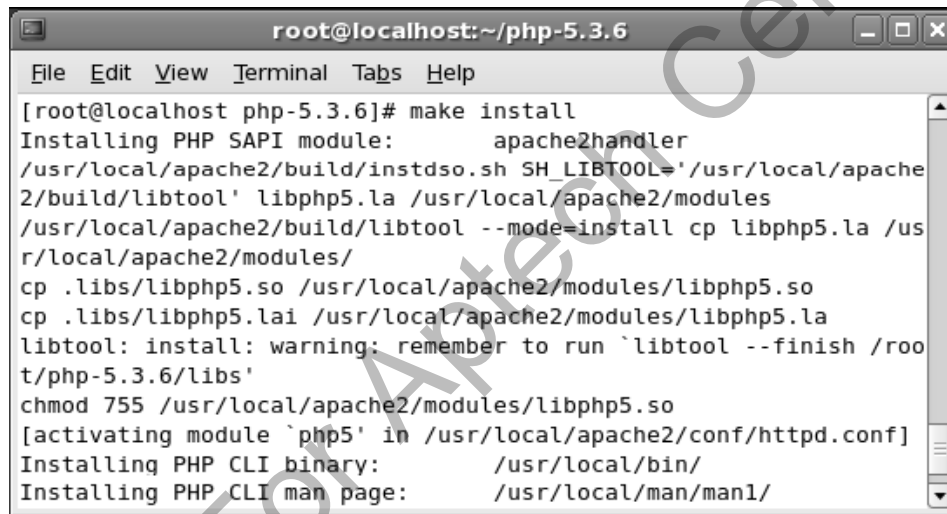
```
root@localhost:~/php-5.3.6
File Edit View Terminal Tabs Help
[root@localhost php-5.3.6]# make
/bin/sh /root/php-5.3.6/libtool --silent --preserve-dup-deps --mo
-Iext/standard/ -I/root/php-5.3.6/ext/standard/ -DPHP_ATOM_INC -
6/include -I/root/php-5.3.6/main -I/root/php-5.3.6 -I/root/php-5.
b -I/root/php-5.3.6/ext/ereg/regex -I/usr/include/libxml2 -I/root
sqlite3/libsqlite -I/root/php-5.3.6/TSRM -I/root/php-5.3.6/Zend
e -g -O2 -fvisibility=hidden -prefer-non-pic -c /root/php-5.3.6/
fo.c -o ext/standard/info.lo
/bin/sh /root/php-5.3.6/libtool --silent --preserve-dup-deps --mo
-Imain/ -I/root/php-5.3.6/main/ -DPHP_ATOM_INC -I/root/php-5.3.6
t/php-5.3.6/main -I/root/php-5.3.6 -I/root/php-5.3.6/ext/date/lib
3.6/ext/ereg/regex -I/usr/include/libxml2 -I/root/php-5.3.6/ext/s
e -I/root/php-5.3.6/TSRM -I/root/php-5.3.6/Zend -I/usr/include
ility=hidden -prefer-non-pic -c main/internal_functions.c -o mai
```

## Step 6

To install PHP, enter the following at the command prompt:

```
make install
```

Displays the following output:



```
root@localhost:~/php-5.3.6
File Edit View Terminal Tabs Help
[root@localhost php-5.3.6]# make install
Installing PHP SAPI module:      apache2handler
/usr/local/apache2/build/instdso.sh SH_LIBTOOL='/usr/local/apache
2/build/libtool' libphp5.la /usr/local/apache2/modules
/usr/local/apache2/build/libtool --mode=install cp libphp5.la /us
r/local/apache2/modules/
cp .libs/libphp5.so /usr/local/apache2/modules/libphp5.so
cp .libs/libphp5.lai /usr/local/apache2/modules/libphp5.la
libtool: install: warning: remember to run `libtool --finish /roo
t/php-5.3.6/libs'
chmod 755 /usr/local/apache2/modules/libphp5.so
[activating module `php5' in /usr/local/apache2/conf/httpd.conf]
Installing PHP CLI binary:      /usr/local/bin/
Installing PHP CLI man page:    /usr/local/man/man1/
```



# Writing a Simple PHP Script

- ◆ Rules followed while creating PHP script are as follows:
  - ◆ Embed PHP scripts in the `BODY` tag of an HTML file
  - ◆ Start and end every block of PHP code with `<?php` and `?>` tags
  - ◆ End a PHP statement with a semicolon, `;`
  - ◆ Save all PHP files with a `.php` extension

## Snippet

```
<html>
<body>
<title>PHP Syntax Example</title>
<?php
echo "Hello World";
?>
</body>
</html>
```

This snippet is saved in a file with a `.php` extension.

The `echo` command displays "Hello World" in the browser when executed.

# Comments in a PHP Script

- ◆ Comments are:
  - ◆ Not displayed in the output
  - ◆ Used to assist a programmer to interpret the meaning of a code
- ◆ Comments supported in PHP are:
  - ◆ Single-line
  - ◆ Multi-line
- ◆ Demonstrating the use of comments in a PHP script

## Snippet

```
<?php
// This is a single-line comment
/* and this is a
multi-line
comment */
?>
```

## ◆ Displaying current date using PHP script

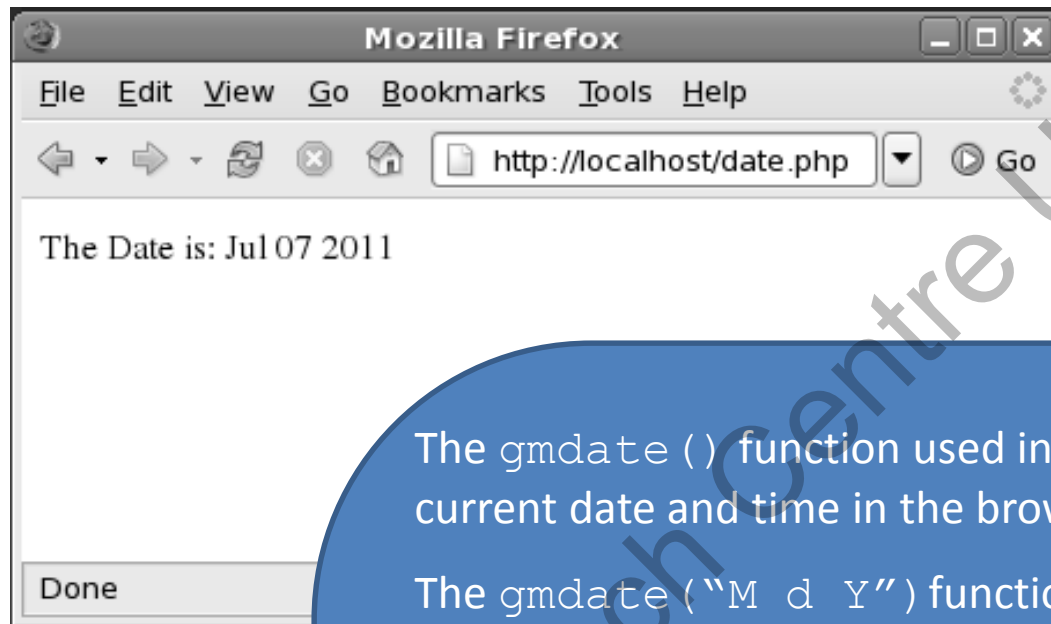
- ◆ Open the `gedit` text editor
- ◆ Enter the following code snippet:

### Snippet

```
<HTML>
<BODY>
The Date is:
<?php echo gmdate("M d Y");
?>
</BODY>
</HTML>
```

- ◆ Save the file as `date.php` in the `/usr/local/apache2/htdocs` directory
- ◆ Open the Mozilla Firefox Web browser and enter `http://localhost/date.php` in the Address bar

Displays the following output:



The `gmdate()` function used in the code snippet displays the current date and time in the browser.

The `gmdate("M d Y")` function takes three parameters to display the current date:

M – displays only three letters of the month

d – displays the current date

Y – displays four digits of the current year

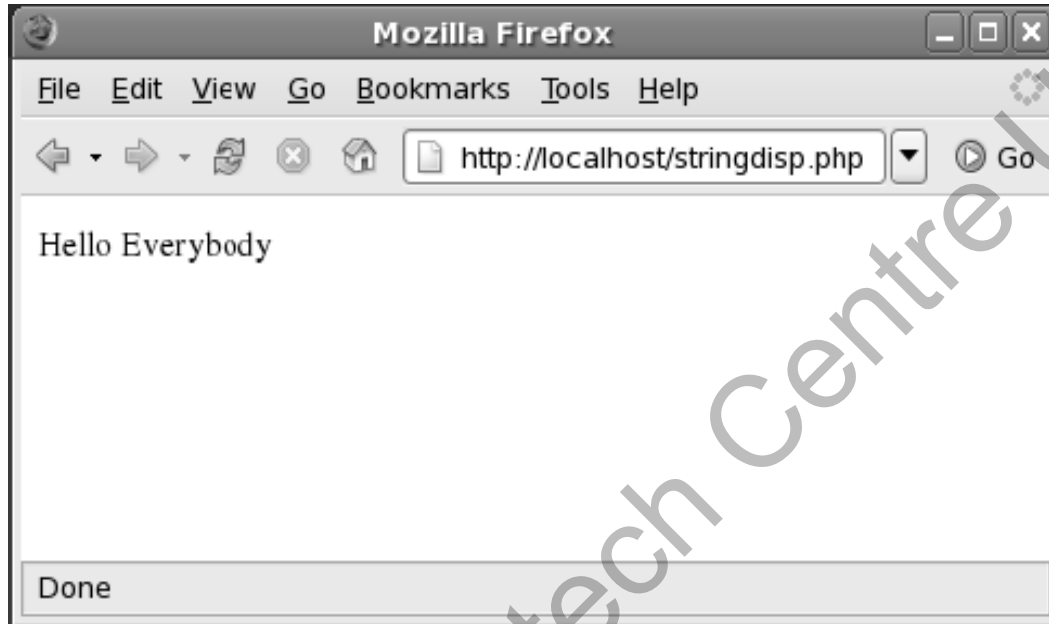
- ◆ Displaying a simple text in the browser using the PHP script
  - ◆ Open the `gedit` text editor
  - ◆ Enter the following code snippet:

## Snippet

```
<HTML>
<BODY>
<?php echo "Hello Everybody";
?>
</BODY>
</HTML>
```

- ◆ Save the file as `stringdisp.php` in the `/usr/local/apache2/htdocs` directory
- ◆ Open the Mozilla Firefox Web browser and enter `http://localhost/stringdisp.php` in the Address bar

Displays the following output:



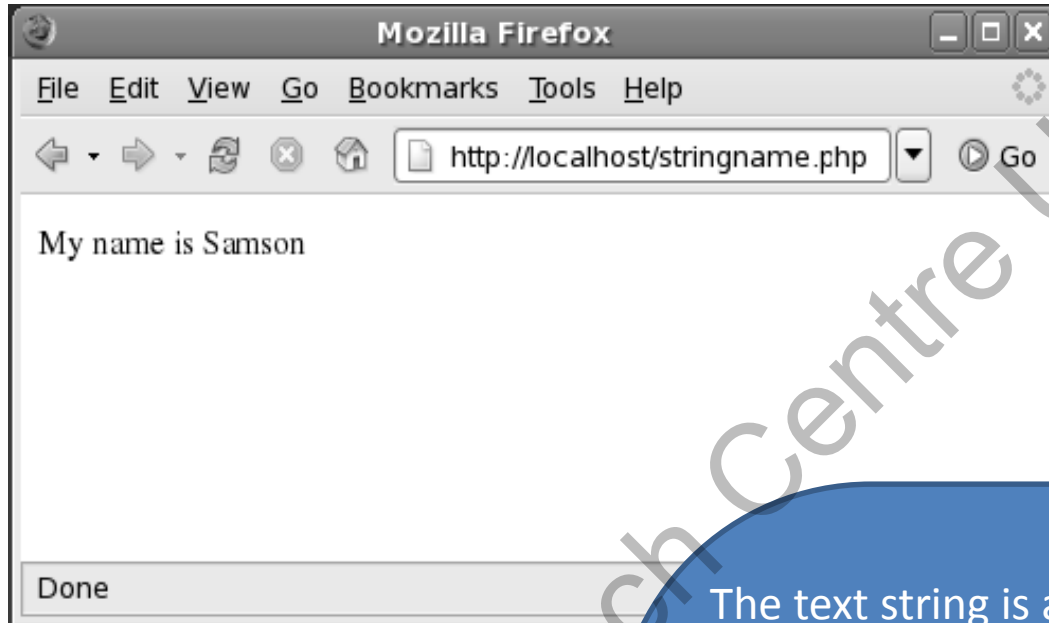
- ◆ Displaying a text in the browser using a variable
  - ◆ Open the `gedit` text editor
  - ◆ Enter the following code snippet:

## Snippet

```
<HTML>
<BODY>
<?php
$str = "My name is Samson";
echo $str;
?>
</BODY>
</HTML>
```

- ◆ Save the file as `stringname.php` in the `/usr/local/apache2/htdocs` directory
- ◆ Open the Mozilla Firefox Web browser and enter `http://localhost/stringname.php` in the Address bar

Displays the following output:



The text string is assigned to a variable named `str`.

The second instruction substitutes the value or content of the variable `str` to the `echo` command.

The `echo` command displays the contents of the `str` variable in the output.



- ◆ It is a network transmission protocol
- ◆ It transfers hypertext files
- ◆ It provides instructions for communication between the client and the server
- ◆ It runs on the Transmission Control Protocol/Internet Protocol (TCP/IP) suite

- ◆ HTTP Header
  - ◆ Is an Internet protocol
  - ◆ Contains instructions to transfer information between a Web client and a Web server
- ◆ The instruction are of two types:
  - ◆ Request - sent by the client to the server
  - ◆ Response - from the server to a client request
- ◆ Format for request or response contains following components:
  - ◆ A request or a response line
  - ◆ HTTP header lines
  - ◆ A blank line
  - ◆ A message body, which is optional

- ◆ Format of an HTTP message is as follows:

## Syntax

```
<initial line, different for request vs. response>  
Header1: value1  
Header2: value2  
Header3: value3  
Blank line  
<optional message body, like file contents or query data>
```

# Initial Request or Response Line

- ◆ Request line contains the following information separated by spaces:
  - ◆ An HTTP method name
  - ◆ Uniform Resource Identifier (URI)
  - ◆ The HTTP version being used

## Snippet

```
GET /sample.html HTTP/1.1
```

- ◆ Response line consists of the following three components separated by spaces:
  - ◆ The HTTP version
  - ◆ A response code indicating the result of the request
  - ◆ An English phrase describing the response code

## Snippet

```
HTTP/1.0 500 Internal Server Error
```

- ◆ Provide information about the request or response or the data sent in the message body

## Syntax

```
Header-Name: value
```

- ◆ Categorized as follows:
  - ◆ General:
    - ◆ Control the processing of a message
    - ◆ Provide extra information to the receiver
  - ◆ Entity:
    - ◆ Provides information about the entity
  - ◆ Request or response:
    - ◆ Provides details about the client's request
    - ◆ Contains response header attached with the response sent by the server

- ◆ Displaying HTTP header lines

## Snippet

```
GET /sample.html HTTP/1.1
User-agent: Mozilla/4.0
Last-Modified: Mon, 11 Apr 2011 23:07:07 GMT
Accept-Language: en
[ blank line above ]
```

Where,

- ◆ **GET** - specifies requested file name and the version of HTTP used
- ◆ **User-agent** - specifies the name of the browser and the version
- ◆ **Last-Modified** - specifies the date and time when the resource was last modified
- ◆ **Accept-Language** - specifies the language preference as English

- ◆ Is the third and an optional component of an HTTP header
- ◆ Appears after the header lines
- ◆ Messages can be of two types:
  - ◆ **Request Message** - contains user data and uploaded files
  - ◆ **Response Message** - contains requested resource

## ◆ header ( ) function:

- ◆ Used to generate the HTTP headers
- ◆ Sends the HTTP commands to the server through HTTP protocols
- ◆ Displays a blank line showing that the header information is complete after the execution of the header ( ) function

### Syntax

```
void header( string string [,bool replace [,int http_response_code]] )
```

## Where,

- ◆ **string** – specifies the header string to be sent
- ◆ **replace** – is an optional parameter and indicates whether should be replaced or not
- ◆ **http\_response\_code** - is an optional parameter and forces the HTTP response code to the specified value



## ◆ Displaying an authentication header

### Snippet

```
<?php  
header('WWW-Authenticate: Negotiate');  
?>
```

Authentication helps to identify if a client is allowed to access to a resource.

Authentication is a means of negotiating access to a secure resource.

## ◆ Authentication schemes are as follows:

### ◆ Http Basic Authentication

- Sends an encoded string
- Contains a user name and password

### ◆ HTTP Digest Authentication

- Is a challenge-response scheme
- Server sends a data string to the client as a challenge
- Client responds with a user name and password

### ◆ NTLM

- Is a challenge-response scheme
- Uses Windows credentials to transform the challenge data
- Requires multiple exchanges between the client and server

### ◆ Negotiate

- Selects between Kerberos and NTLM depending on their availability

- ◆ The `replace` option specifies to replace the previous header or add a second header to the document
- ◆ If `false`, then new header will be added to the document

## Syntax

```
void header('string string', boolean replace)
```

Where,

- ◆ **string** - defines the authentication parameters
- ◆ **replace** - substitutes the existing header or adds new headers to the document. The default value is set to true, so all similar headers are replaced

- ◆ Displaying addition of multiple headers to the document

## Snippet

```
<?php
header('WWW-Authenticate: Negotiate');
header('WWW-Authenticate: NTLM', false);
?>
```

WWW-Authenticate - specifies the authentication string.

NTLM - specifies a challenge-response authentication mechanism.

false - defines the parameter of the replace option.

- ◆ Displays the response of the Web server for a request
- ◆ The request can include the status or the location of the client

### Syntax

```
void header( string string, boolean replace, integer http_response_code )
```

Where,

- ◆ **string** - defines the authentication parameters
- ◆ **replace** - indicates whether previous defined headers need to be replaced or not
- ◆ **http\_response\_code** - forces the HTTP response code to the specified value

- ◆ Displaying a PHP script to redirect the user from one Web page or URL to another Web site

### Snippet

```
header("Location: http://google.com");
```

Location is a type of HTTP header redirecting the browser to the specified URL.

- ◆ Displaying a PHP script with an HTTP response code

### Snippet

```
header("Location: http://google.com", true, 303);
```

- ◆ Location - is an HTTP header that redirects the browser to the specified URL
- ◆ true - defines the parameter of the replace option
- ◆ 303 - is a redirection response code

- ◆ PHP is an open source scripting language embedded within HTML codes and used for developing dynamic Web pages
- ◆ PHP is used for executing scripts from the command line and for developing client-side GUI applications that are platform independent
- ◆ PHP is used for generating files in PDF and HTML formats
- ◆ PHP can generate e-mails by retrieving data from documents and sending it through any standard mail protocol



- ◆ PHP is used to render graphical images, such as GIF and PNG images
- ◆ PHP consists of tools for developing and designing Web pages. These tools are the program text editors. The popular text editor used for writing PHP scripts on Linux platform is gedit
- ◆ A PHP script starts with `<?php` tag and ends with the `?>` tag. These scripts are embedded in the HTML tags
- ◆ A HTTP message or protocol is divided into three parts, the request or response line, the HTTP header, and the body of the protocol