Installing and Configuring PHP 7

Session 2

Objectives

- *Explain the pre-requisites for installing PHP 7.*
- *Describe the steps to configure PHP 7.*
- *Identify the steps to install PHP 7.*
- *Describe the process to create simple PHP scripts.*
- *Explain how to use HTTP headers in PHP.*

Pre-requisites for Installing PHP 7 1-2

- Installation of PHP requires:

A Web server

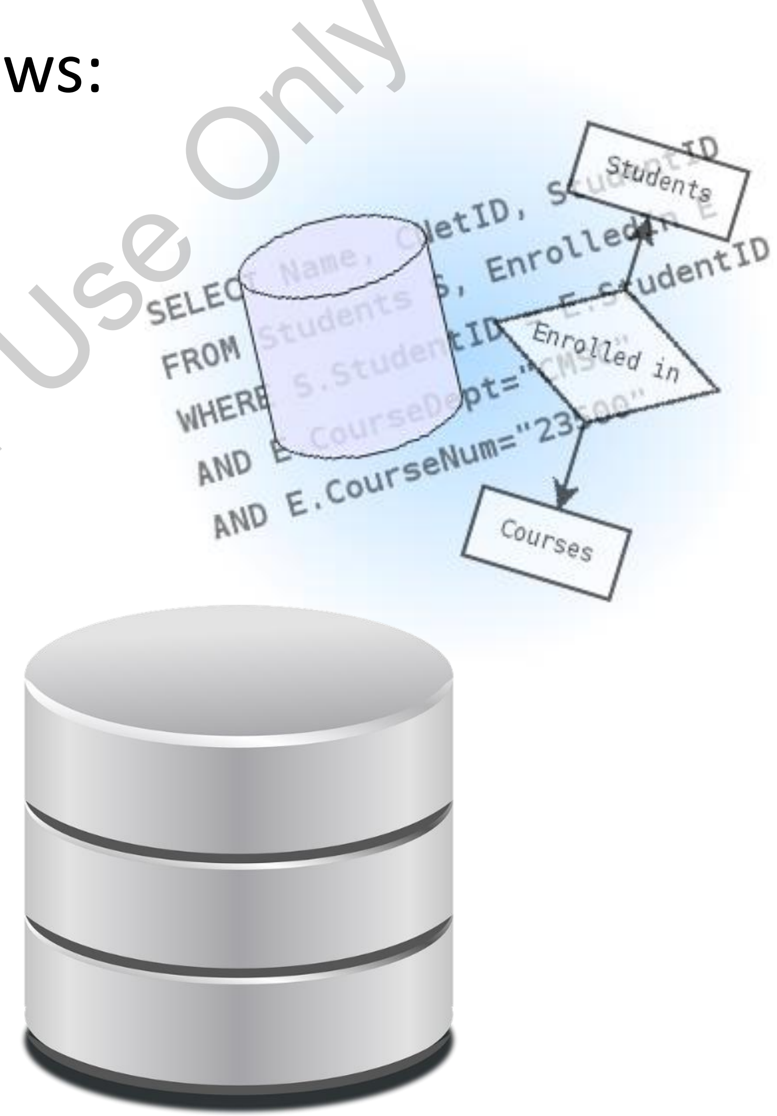
A database

- Web servers supported are as follows:
    - Internet Information Services (IIS)
    - Apache
    - Nginx

Pre-requisites for Installing PHP 7 2-2

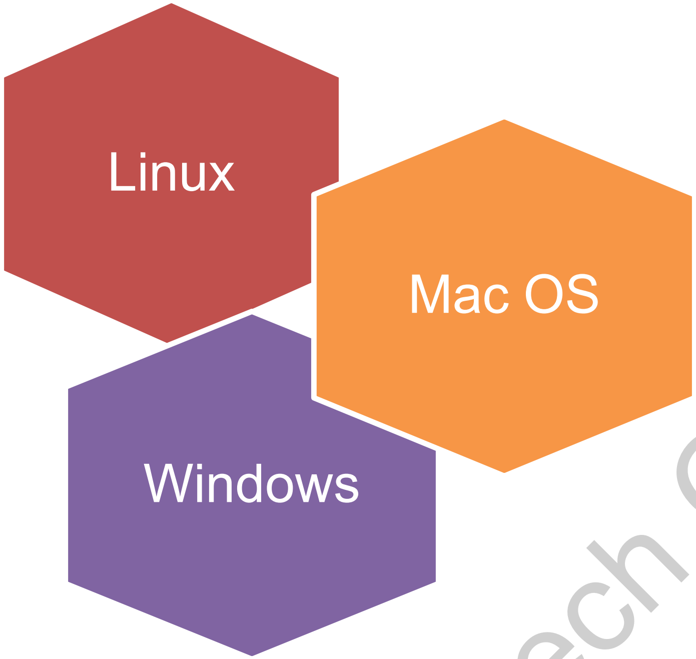
- Databases supported are as follows:

- DB2
- MS SQL
- MySQL
- Oracle
- PostgreSQL



Installing the Packages

- PHP can be installed on:



- Download the relevant package from http://php.net/downloads.php Website.

Installing the Packages 2-2

- To extract PHP packages on Linux, perform these steps:
  - Enter the following command at the command prompt:

```
# tar xjvf php-7.0.4.tar.gz
# tar -xf php-7.0.4.tar
# cd php-7.0.4/
```

  - To install the development package, enter the following command at the command prompt:

```
# dnf install aspell-devel bzip2-devel freetype-devel gmp-
devel libXpm-devel libcurl-devel libjpeg-turbo-devel
libmcrypt-devel libpng-devel libxml2-devel libxslt-devel
mariadb-devel recode-devel uw-imap-devel gcc openssl-devel –y
```

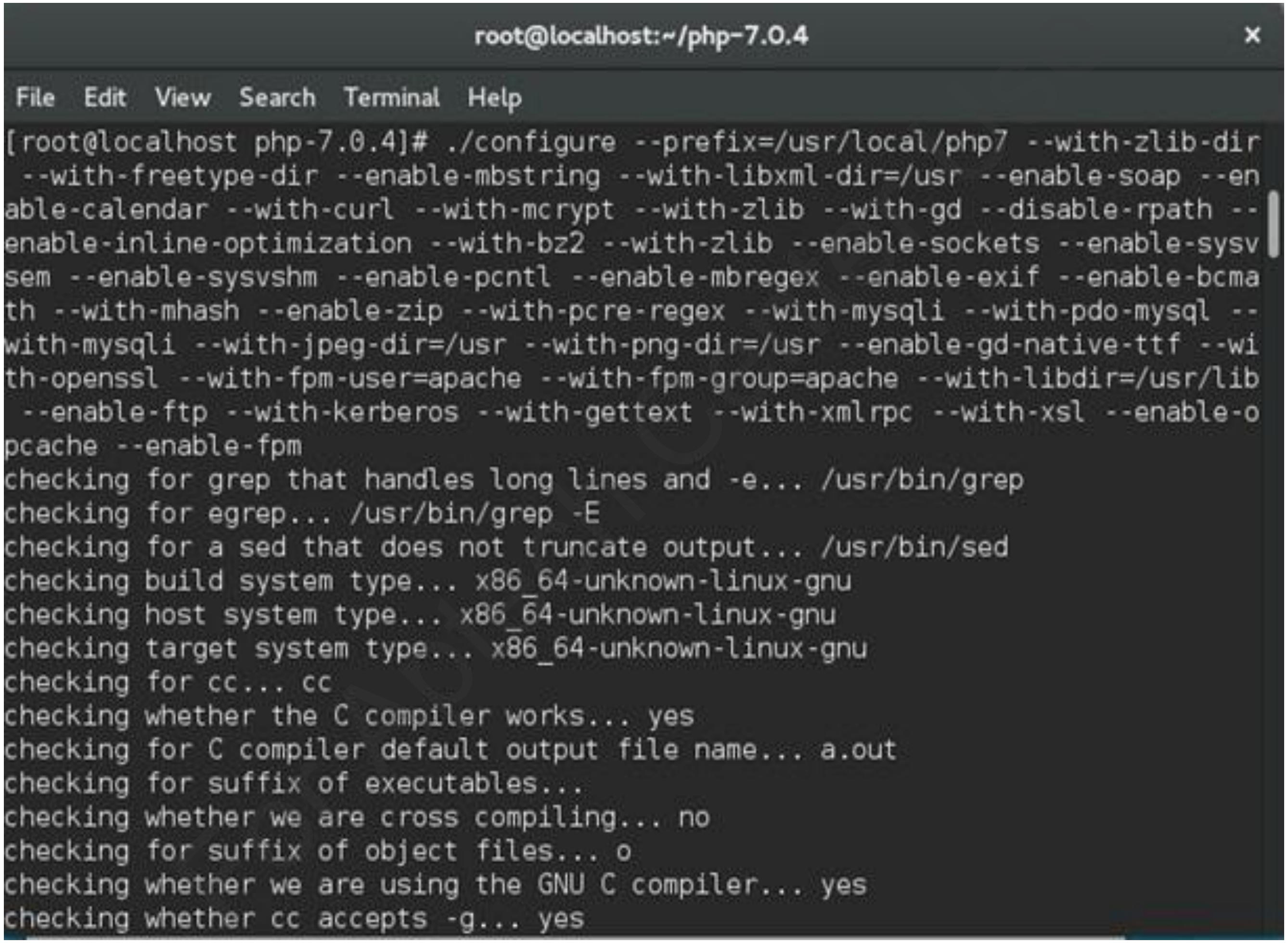
Configuring PHP 7.0.4 1-2

- To configure PHP 7.0.4, perform the following steps:
  - Enter the following command at the command prompt:

```
#cd php-7.0.4
# ./configure --prefix=/usr/local/php7 --with-zlib-
dir --with-freetype-dir --enable-mbstring --with-
libxml-dir=/usr --enable-soap --enable-calendar --
with-curl --with-mcrypt --with-zlib --with-gd --
disable-rpath --enable-inline-optimization --with-bz2
--with-zlib --enable-sockets --enable-sysvsem --
enable-sysvshm --enable-pcntl --enable-mbregex --
enable-exif --enable-bcmath --with-mhash --enable-zip
--with-pcre-regex --with-mysqli --with-pdo-mysql --
with-mysqli --with-jpeg-dir=/usr --with-png-dir=/usr
--enable-gd-native-ttf --with-openssl --with-fpm-
user=apache --with-fpm-group=apache --with-
libdir=/usr/lib --enable-ftp --with-kerberos --with-
gettext --with-xmlrpc --with-xsl --enable-opcache --
enable-fpm
```

Configuring PHP 7.0.4 2-2

Displays the following output:


root@localhost:~/php-7.0.4
×
File Edit View Search Terminal Help

```
[root@localhost php-7.0.4]# ./configure --prefix=/usr/local/php7 --with-zlib-dir
 --with-freetype-dir --enable-mbstring --with-libxml-dir=/usr --enable-soap --en
able-calendar --with-curl --with-mcrypt --with-zlib --with-gd --disable-rpath --
enable-inline-optimization --with-bz2 --with-zlib --enable-sockets --enable-sysv
sem --enable-sysvshm --enable-pcntl --enable-mbregex --enable-exif --enable-bcma
th --with-mhash --enable-zip --with-pcre-regex --with-mysqli --with-pdo-mysql --
with-mysqli --with-jpeg-dir=/usr --with-png-dir=/usr --enable-gd-native-ttf --wi
th-openssl --with-fpm-user=apache --with-fpm-group=apache --with-libdir=/usr/lib
 --enable-ftp --with-kerberos --with-gettext --with-xmlrpc --with-xsl --enable-o
pcache --enable-fpm
checking for grep that handles long lines and -e... /usr/bin/grep
checking for egrep... /usr/bin/grep -E
checking for a sed that does not truncate output... /usr/bin/sed
checking build system type... x86_64-unknown-linux-gnu
checking host system type... x86_64-unknown-linux-gnu
checking target system type... x86_64-unknown-linux-gnu
checking for cc... cc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether cc accepts -g... yes
```

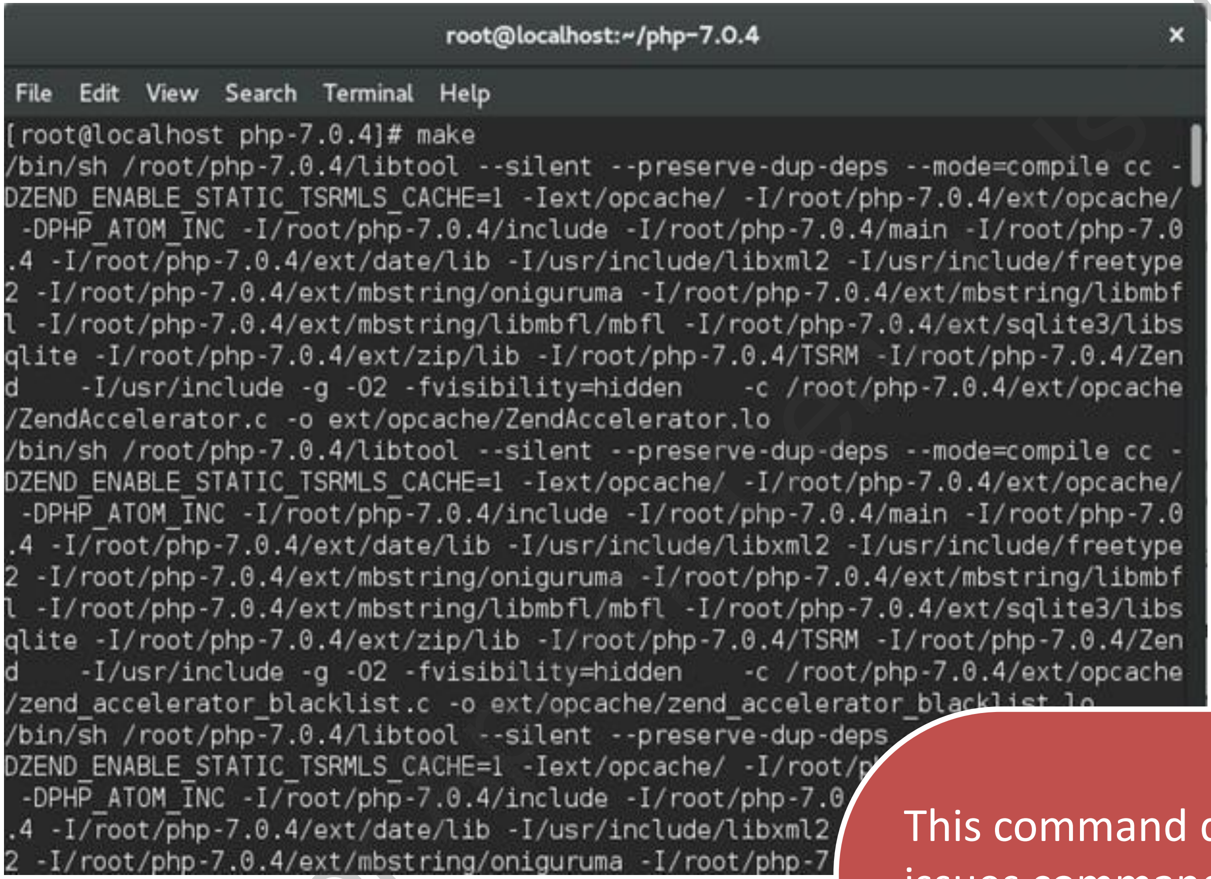
Installing PHP 7.0.4 1-3

Step 1

To extract the packages, install the pre-requisites, and configure the PHP files, enter the following command at the command prompt:

```
#make
```

Displays the following output:


root@localhost:~/php-7.0.4
File Edit View Search Terminal Help

```
[root@localhost php-7.0.4]# make
/bin/sh /root/php-7.0.4/libtool --silent --preserve-dup-deps --mode=compile cc -
DZEND_ENABLE_STATIC_TSRMLS_CACHE=1 -Iext/opcache/ -I/root/php-7.0.4/ext/opcache/
 -DPHP_ATOM_INC -I/root/php-7.0.4/include -I/root/php-7.0.4/main -I/root/php-7.0
.4 -I/root/php-7.0.4/ext/date/lib -I/usr/include/libxml2 -I/usr/include/freetype
2 -I/root/php-7.0.4/ext/mbstring/oniguruma -I/root/php-7.0.4/ext/mbstring/libmbf
l -I/root/php-7.0.4/ext/mbstring/libmbfl/mbfl -I/root/php-7.0.4/ext/sqlite3/libs
qlite -I/root/php-7.0.4/ext/zip/lib -I/root/php-7.0.4/TSRM -I/root/php-7.0.4/Zen
d    -I/usr/include -g -O2 -fvisibility=hidden    -c /root/php-7.0.4/ext/opcache
/ZendAccelerator.c -o ext/opcache/ZendAccelerator.lo
/bin/sh /root/php-7.0.4/libtool --silent --preserve-dup-deps --mode=compile cc -
DZEND_ENABLE_STATIC_TSRMLS_CACHE=1 -Iext/opcache/ -I/root/php-7.0.4/ext/opcache/
 -DPHP_ATOM_INC -I/root/php-7.0.4/include -I/root/php-7.0.4/main -I/root/php-7.0
.4 -I/root/php-7.0.4/ext/date/lib -I/usr/include/libxml2 -I/usr/include/freetype
2 -I/root/php-7.0.4/ext/mbstring/oniguruma -I/root/php-7.0.4/ext/mbstring/libmbf
l -I/root/php-7.0.4/ext/mbstring/libmbfl/mbfl -I/root/php-7.0.4/ext/sqlite3/libs
qlite -I/root/php-7.0.4/ext/zip/lib -I/root/php-7.0.4/TSRM -I/root/php-7.0.4/Zen
d    -I/usr/include -g -O2 -fvisibility=hidden    -c /root/php-7.0.4/ext/opcache
/zend_accelerator_blacklist.c -o ext/opcache/zend_accelerator_blacklist.lo
/bin/sh /root/php-7.0.4/libtool --silent --preserve-dup-deps
DZEND_ENABLE_STATIC_TSRMLS_CACHE=1 -Iext/opcache/ -I/root/p
 -DPHP_ATOM_INC -I/root/php-7.0.4/include -I/root/php-7.0
.4 -I/root/php-7.0.4/ext/date/lib -I/usr/include/libxml2
2 -I/root/php-7.0.4/ext/mbstring/oniguruma -I/root/php-7
```

This command determines large files and issues commands to recompile those large files.

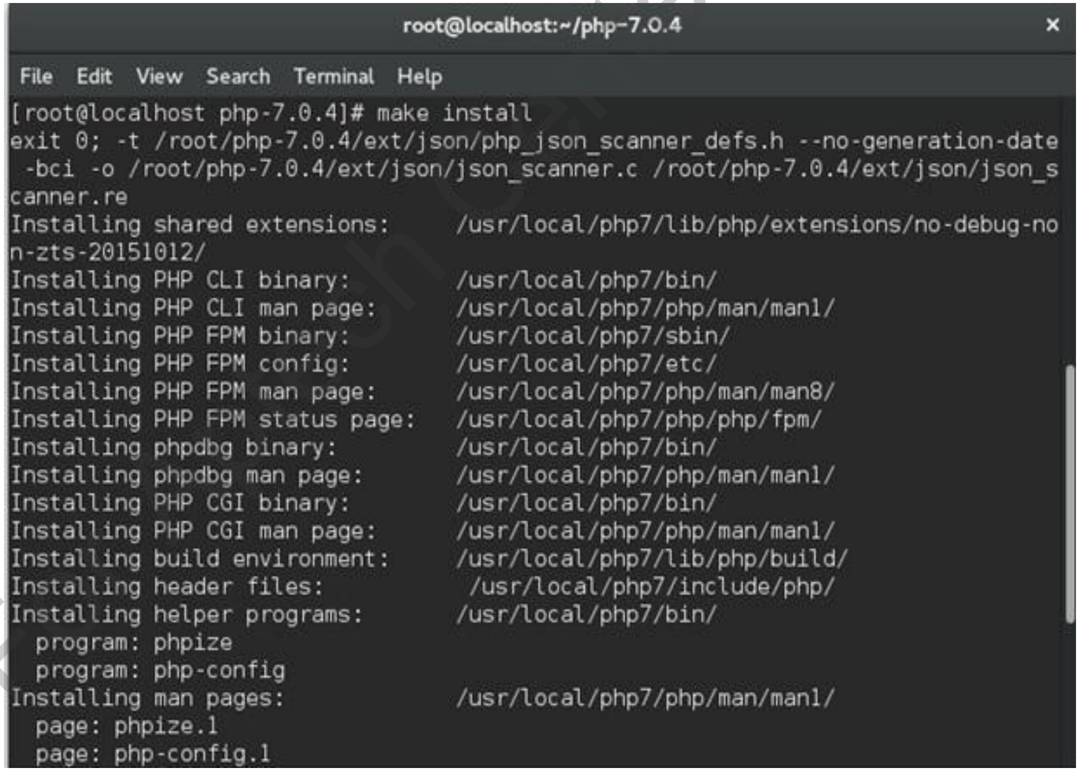
Installing PHP 7.0.4 3-3

Step 2

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

```
# make install
```

Displays the following output:


root@localhost:~/php-7.0.4
×
File Edit View Search Terminal Help

```
[root@localhost php-7.0.4]# make install
exit 0; -t /root/php-7.0.4/ext/json/php_json_scanner_defs.h --no-generation-date
 -bci -o /root/php-7.0.4/ext/json/json_scanner.c /root/php-7.0.4/ext/json/json_s
canner.re
Installing shared extensions:     /usr/local/php7/lib/php/extensions/no-debug-no
n-zts-20151012/
Installing PHP CLI binary:        /usr/local/php7/bin/
Installing PHP CLI man page:      /usr/local/php7/php/man/man1/
Installing PHP FPM binary:        /usr/local/php7/sbin/
Installing PHP FPM config:        /usr/local/php7/etc/
Installing PHP FPM man page:      /usr/local/php7/php/man/man8/
Installing PHP FPM status page:   /usr/local/php7/php/php/fpm/
Installing phpdbg binary:         /usr/local/php7/bin/
Installing phpdbg man page:       /usr/local/php7/php/man/man1/
Installing PHP CGI binary:        /usr/local/php7/bin/
Installing PHP CGI man page:      /usr/local/php7/php/man/man1/
Installing build environment:     /usr/local/php7/lib/php/build/
Installing header files:           /usr/local/php7/include/php/
Installing helper programs:       /usr/local/php7/bin/
  program: phpize
  program: php-config
Installing man pages:             /usr/local/php7/php/man/man1/
  page: phpize.1
  page: php-config.1
```

Setting up Apache to Use PHP

Step 1

Open the `httpd.conf` file.

Step 2

Add the following directives in the file:

```
AddHandler application/x-httpd-php .php
LoadModule php7_module C:\php7.dll
AddType application/x-httpd-php .php
PHPIniDir C:\php
```

Step 3

If required, change the path of the PHP installation folder.

Step 4

Save and restart the Apache Web server.

Writing a Simple PHP Script

- Rules followed while writing 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 1-8

- 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 */
?>
```

Comments in a PHP Script 2-8

- Displaying current date using PHP script
  - Open the `gedit` text editor
  - Enter the following code snippet:

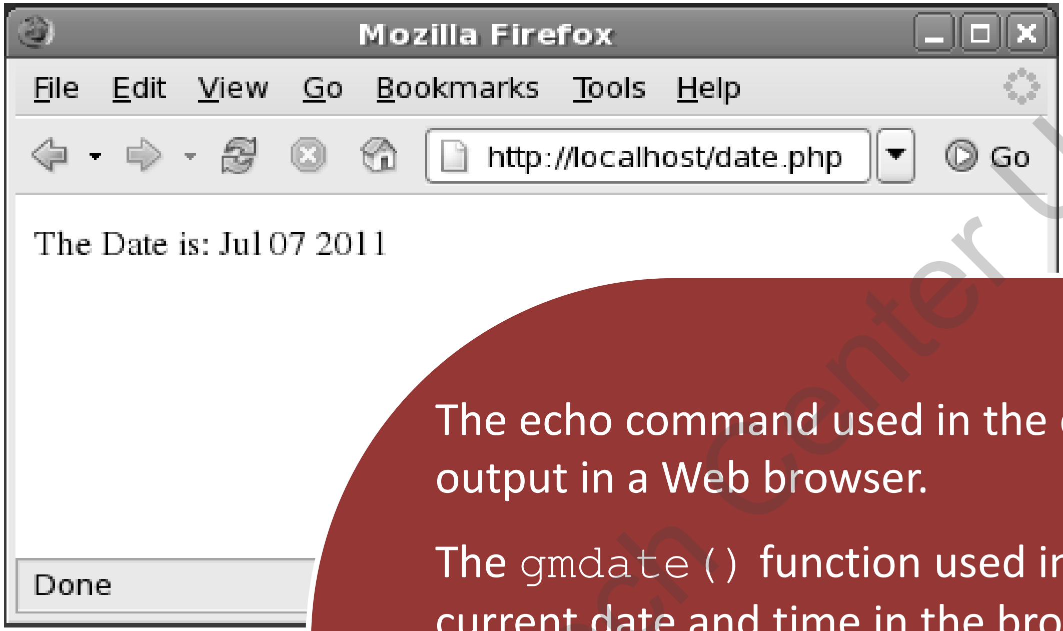
Snippet

```
<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 Mozilla Firefox Web browser and enter `http://localhost/date.php` in the Address bar

Comments in a PHP Script 3-8

Displays the following output:


Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost/date.php
Go
Done

The Date is: Jul 07 2011

The echo command used in the code is used to display the output in a Web browser.

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

Comments in a PHP Script 4-8

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

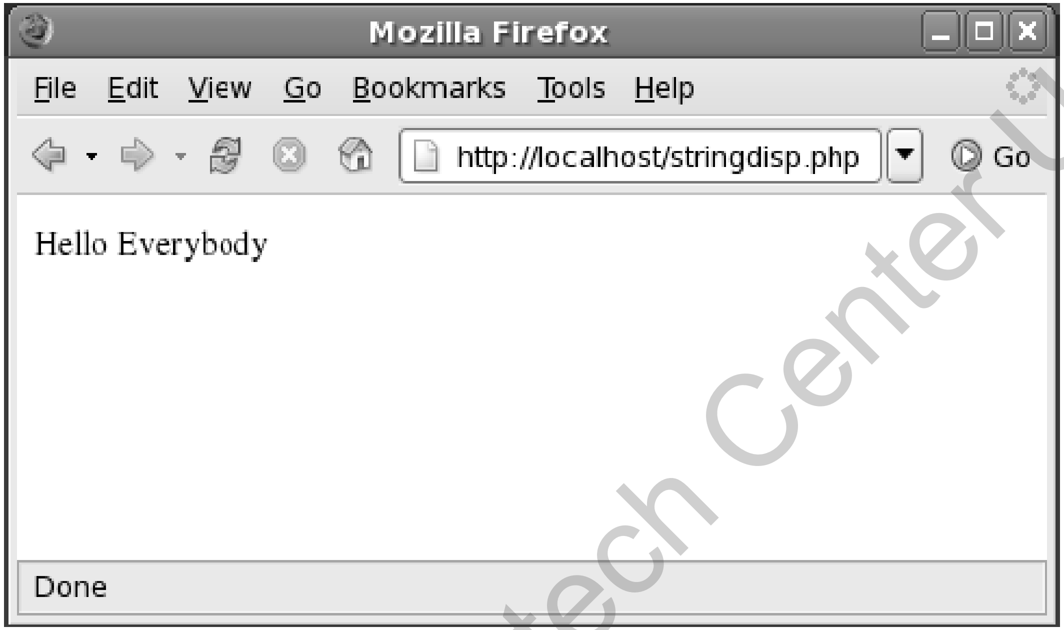
Snippet

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

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

Comments in a PHP Script

Displays the following output:


Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost/stringdisp.php
Go
Done

```
Hello Everybody
```

Comments in a PHP Script 6-8

- Rules followed while using a variable in a PHP script are as follows:
  - Variables:
    - Must start with a dollar sign '$`
    - Can contain strings, numbers, and arrays
  - Variables names:
    - Must start with a letter or an underscore '_'
    - Can only contain alpha-numeric characters and underscores without spaces

Comments in a PHP Script

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

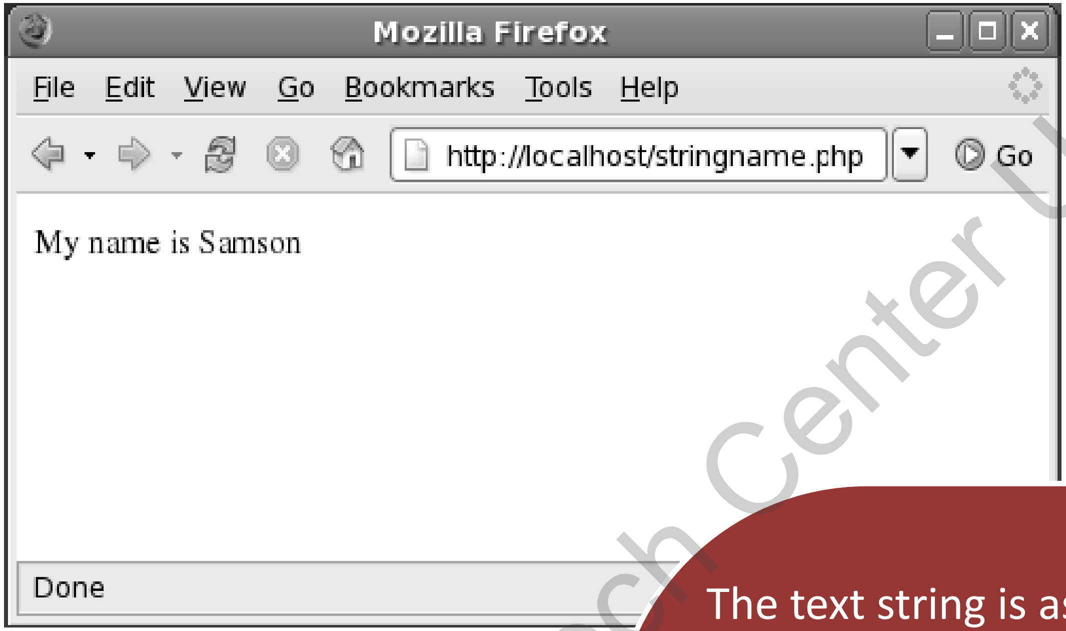
Snippet

```
<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

Comments in a PHP Script

Displays the following output:


Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost/stringname.php
Go
Done

My name is Samson

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.

Using HTTP Headers in PHP 1-3

- `header()` function:
  - Used to generate HTTP headers
  - Sends 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 [,bool replace [,int http_response_code]] )
```

where,

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

Using HTTP Headers in PHP 2-3

- 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.

Using HTTP Headers in PHP 3-3

- 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

header() Function with replace Option 1-2

- 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', 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 that all similar headers are replaced

header() Function with replace Option 2-2

- 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.

header() Function with http_response_code Option 1-4

- 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 , 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

header() Function with http_response_code Option 2-4

- An HTTP response codes consists of three digits that determine the status of a response.
- The status codes are classified as follows:
  - 1xx codes are informational codes
  - 2xx are success codes
  - 3xx are redirection codes
  - 4xx are client error codes
  - 5xx are server error codes

header() Function with http_response_code Option 3-4

- 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.

header() Function with http_response_code Option 4-4

- 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

Summary

- A Web server and a database is required before installing PHP 7.
- Any older version of PHP must be uninstalled before installing PHP 7.
- A PHP file includes simple text, HTML tags, and PHP script.
- PHP supports both single-line and multiple line comments.

Summary

- PHP automatically assigns the correct data type for a variable depending upon the value assigned to the variable.
- 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.