PHP Handout #3

PHP Date() Function

The PHP date() function is used to format a time and/or date.

Syntax

date(*format*,*timestamp*)

format -Required. Specifies the format of the timestamp
timestamp -Optional. Specifies a timestamp. Default is the current date and time

Format the date

The required *format* parameter in the date() function specifies how to format the date/time.
Here are some characters that can be used:

- d - Represents the day of the month (01 to 31)
- m - Represents a month (01 to 12)
- Y - Represents a year (in four digits)

Lets try this code:

```
<?php
echo date("Y/m/d") . "<br />";
echo date("Y.m.d") . "<br />";
echo date("Y-m-d")
?>
```

Adding a time stamp

The optional *timestamp* parameter in the date() function specifies a timestamp. If you do not specify a timestamp, the current date and time will be used.

Syntax for mktime()

mktime(hour,minute,second,month,day,year,is_dst)

Lets try this code that adds one day into the future:

```
<?php
$tomorrow = mktime(0,0,0,date("m"),date("d")+1,date("Y"));
echo "Tomorrow is ".date("Y/m/d", $tomorrow);
?>
```

PHP File Handling

Opening a file

The **fopen()** function is used to open files in PHP.
The first parameter of this function contains the name of the file to be opened and the second parameter specifies in which mode the file should be opened:

```
<html>
<body>

<?php
$file=fopen("welcome.txt","r");
?>

</body>
</html>
```

The file may be opened in any of the following modes:

Modes	Description
r	Read only. Starts at the beginning of the file
r+	Read/Write. Starts at the beginning of the file
w	Write only. Opens and clears the contents of file; or creates a new file if it doesn't exist
w+	Read/Write. Opens and clears the contents of file; or creates a new file if it doesn't exist
a	Append. Opens and writes to the end of the file or creates a new file if it doesn't exist
a+	Read/Append. Preserves file content by writing to the end of the file
x	Write only. Creates a new file. Returns FALSE and an error if file already exists
x+	Read/Write. Creates a new file. Returns FALSE and an error if file already exists

The following example generates a message if the fopen() function is unable to open the specified file:

```
<html>
<body>

<?php
$file=fopen("welcome.txt","r") or exit("Unable to open file!");
?>

</body>
</html>
```

Closing a File

The fclose() function is used to close an open file:

```
<?php
$file = fopen("test.txt","r");

//some code to be executed

fclose($file);
?>
```

Check End-of-file

The feof() function checks if the "end-of-file" (EOF) has been reached.

```
if (feof($file)) echo "End of file";
```

Reading a file line by line

The fgets() function is used to read a single line from a file.

```
<?php
$file = fopen("welcome.txt", "r") or exit("Unable to open file!");
//Output a line of the file until the end is reached
while(!feof($file))
  {
  echo fgets($file). "<br />";
  }
fclose($file);
?>
```

Reading a File Character by Character

The fgetc() function is used to read a single character from a file.

```
<?php
$file=fopen("welcome.txt","r") or exit("Unable to open file!");
while (!feof($file))
  {
  echo fgetc($file);
  }
fclose($file);
?>
```

PHP Cookies

A cookie is often used to identify a user. A cookie is a small file that the server embeds on the user's computer. Each time the same computer requests a page with a browser, it will send the cookie too. With PHP, you can both create and retrieve cookie values.

Basic Syntax

setcookie(name, value, expire, path, domain);

How to create Cookie?

The setcookie() function is used to set a cookie.

Note: *The setcookie() function must appear BEFORE the <html> tag.*

Example 1:

```
<?php
setcookie("user", "Alex Porter", time()+3600);
?>
```

You can also set the expiration time of the cookie in another way. It may be easier than using seconds.

Example 2:

```
<?php
$expire=time()+60*60*24*30;
setcookie("user", "Alex Porter", $expire);
?>
```

*In the example above the expiration time is set to a month (*60 sec * 60 min * 24 hours * 30 days*).

How to retrieve a Cookie value?

He PHP $_COOKIE variable is used to retrieve a cookie value.

```
<?php
// Print a cookie
echo $_COOKIE["user"];

// A way to view all cookies
print_r($_COOKIE);
?>
```

In the following example we use the isset() function to find out if a cookie has been set:

```
<html>
<body>

<?php
if (isset($_COOKIE["user"]))
  echo "Welcome " . $_COOKIE["user"] . "!<br />";
else
  echo "Welcome guest!<br />";
?>

</body>
</html>
```

Deleting a Cookie

When deleting a cookie you should assure that the expiration date is in the past.

```
<?php
// set the expiration date to one hour ago
setcookie("user", "", time()-3600);
?>
```

**If your application deals with browsers that do not support cookies, you will have to use other methods to pass information from one page to another in your application. One method is to pass the data through forms*

PHP Sessions

A PHP session variable is used to store information about, or change settings for a user session. Session variables hold information about one single user, and are available to all pages in one application.

PHP Session variables

A PHP session allows you to store user information on the server for later use (i.e. username, shopping items, etc). However, session information is temporary and will be deleted after the user has left the website. If you need a permanent storage you may want to store the data in a database.

Starting a PHP Session

```
<?php session_start(); ?>

<html>
<body>

</body>
</html>
```

Note: *The session_start() function must appear BEFORE the <html> tag:*

Storing and Retrieving a Session Variable

The correct way to store and retrieve session variables is to use the PHP $_SESSION variable:

Example:

```
<?php
session_start();
// store session data
$_SESSION['views']=1;
?>

<html>
<body>

<?php
//retrieve session data
echo "Pageviews=". $_SESSION['views'];
?>

</body>
</html>
```

In the example below, we create a simple page-views counter. The isset() function checks if the "views" variable has already been set. If "views" has been set, we can increment our counter. If "views" doesn't exist, we create a "views" variable, and set it to 1:

```
<?php
session_start();

if(isset($_SESSION['views']))
$_SESSION['views']=$_SESSION['views']+1;
else
$_SESSION['views']=1;
echo "Views=". $_SESSION['views'];
?>
```

Destroying a Session

If you wish to delete some session data, you can use the unset() or the session_destroy() function.

The unset() function is used to free the specified session variable:

```
<?php
unset($_SESSION['views']);
?>
```

You can also completely destroy the session by calling the session_destroy() function:

```
<?php
session_destroy();
?>
```