Learn with Rabbil

PHP Date

- d - Represents the day of the month
- m - Represents a month
- Y - Represents a year
- l (lowercase 'L') - the day of the week

index.php

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

PHP Time

- H - 24-hour format of an hour (00 to 23)
- h - 12-hour format of an hour with leading zeros (01 to 12)
- i - Minutes with leading zeros (00 to 59)
- s - Seconds with leading zeros (00 to 59)
- a - Lowercase Ante meridiem and Post meridiem (am or pm)

index.php

```php
<?php
echo  date("h:i:sa")
?>

```

Time Zone

- Server is in another country or set up for a different timezone
- Need the time to be correct according to a specific location

index.php

```php
<?php
date_default_timezone_set('Asia/Dhaka');
echo  date("h:i:sa")
?>

```

Date Functions

- date_add()
- Adds days, months, years, hours, minutes, and seconds to a date

index.php

```php
<?php
$date=date_create("2013-03-15");
date_add($date,date_interval_create_from_date_string("40 years"));
echo date_format($date,"Y-m-d");
?>
```

Date Functions

- date_sub()
- Sub days, months, years, hours, minutes, and seconds to a date

index.php

```
2   $date=date_create("2013-03-15");
3   date_sub($date,date_interval_create_from_date_string("40 days"));
4   echo date_format($date,"Y-m-d");
```

Date Functions

- date_diff()
- Returns the difference between two DateTime objects.
- R- for negative positive sign
- a- for absolute value

index.php

```
2   $date1=date_create("2013-03-15");
3   $date2=date_create("2013-12-12");
4   $diff=date_diff($date1,$date2);
5   echo $diff->format("%R%a days");
```

Date Functions

- date_format()
- Returns a date formatted according to the specified format

index.php

```
2   $date=date_create("2013-03-15");
3   echo date_format($date,"Y/m/d H:i:s");
```

Date Functions

- date_sunrise()
- date_sunset()
- Returns the sunrise time for a specified day and location.
- date_sunrise(timestamp, format, latitude, longitude, zenith, gmtoffset)

index.php

```
2   echo(date_sunrise(time(),SUNFUNCS_RET_STRING,38.4,-9,90,1));
3   echo(date_sunset(time(),SUNFUNCS_RET_STRING,38.4,-9,90,1));
```

Learn with Rabbil

Date Functions

- date_timestamp_get()
- Return the Unix timestamp for today's date and time:

index.php

```
2   $date=date_create();
3   echo date_timestamp_get($date);
```

Learn with Rabbil

Exception Handling in PHP

- Exception handling is a powerful mechanism of PHP, which is used to handle runtime errors
- So that the normal flow of the application can be maintained.

try -

The try block contains the code that may have an exception or where an exception can arise.

catch -

The catch block contains the code that executes when a specified exception is thrown.

throw -

It is a keyword used to throw an exception. It also helps to list all the exceptions that a function throws but does not handle itself.

finally -

he finally block contains a code, which is used for clean-up activity in PHP. Basically, it executes the essential code of the program.

Exception Handling Example

index.php

```php
function checkNumber($num) {
    if($num>=1) {
      throw new Exception("Value must be less than 1");
    }
    return true;
}

try {
    checkNumber(5);
    echo 'If you see this text, the passed value is less than 1';
}

catch (Exception $e) {
    echo 'Exception Message: ' .$e->getMessage();
}
finally {
    echo '</br> It is finally block, which always executes.';
}
```

Learn with Rabbil

Exception Handling Example

index.php

```php
function divide($dividend, $divisor) {
  if($divisor == 0) {
    throw new Exception("Division by zero");
  }
  return $dividend / $divisor;
}

try {
  echo divide(5, 0);
} finally {
  echo "Process complete.";
}
```

Learn with Rabbil

PHP Sessions

- A session is a way to store information (data) on the server that can be used across multiple pages of a website or web application.
- In PHP, a session is started with the session_start() function. This function must be called at the beginning of each script that uses session data.
- Session data is stored in a special global variable called $_SESSION. This variable is an associative array that can be used to store and retrieve data throughout the session.
- Session data is stored on the server and is associated with a unique session ID. The session ID is typically stored in a cookie on the client-side, so that subsequent requests to the server can be associated with the correct session data.
- Session data is automatically destroyed when the session ends, either by the user closing their browser or by the session timeout period expiring. However, you can also manually destroy a session with the session_destroy() function.

Learn with Rabbil

PHP Sessions Example

index.php

```php
<?php
// Start the session
session_start();

// Store data in the session
$_SESSION['username'] = 'JohnDoe';
$_SESSION['loggedIn'] = true;

// Retrieve data from the session
$username = $_SESSION['username'];
$loggedIn = $_SESSION['loggedIn'];

// Output the retrieved data
echo "Username: " . $username . "<br>";
echo "Logged In: " . ($loggedIn ? 'Yes' : 'No');
?>
```

Learn with Rabbil

PHP file handling methods

fopen()

This function is used to open a file in PHP. It takes two parameters - the file name and the mode in which the file should be opened (e.g. read-only, write-only, read/write, etc.).

fclose()

This function is used to close an open file handle in PHP. It takes a single parameter - the file handle to be closed.

fread()

This function is used to read data from an open file handle. It takes two parameters - the file handle and the number of bytes to read.

fwrite()

This function is used to write data to an open file handle. It takes two parameters - the file handle and the data to be written

Learn with Rabbil

PHP file handling methods

fgets()

This function is used to read a line of text from an open file handle. It takes a single parameter - the file handle

file_get_contents()

This function is used to read the contents of a file into a string variable. It takes a single parameter - the name of the file to be read.

file_put_contents()

This function is used to write a string of data to a file. It takes two parameters - the name of the file to be written to and the data to be written.

feof()

This function is used to check if the end of a file has been reached. It takes a single parameter - the file handle.

fgets()

This function is used to check if the end of a file has been reached. It takes a single parameter - the file handle.

Learn with Rabbil

PHP file handling example

Open a file and read the file contents using fopen(), fread(), fclose()

index.php

```php
<?php
$myfile = fopen("myfile.txt", "r");
$contents = fread($myfile, filesize("myfile.txt"));
echo $contents;
fclose($myfile);
?>
```

Learn with Rabbil

PHP file handling example

Open a file and write the file contents using fopen(), fwrite(), fclose()

index.php

```php
$myfile = fopen("myfile.txt", "w");
$data = "Hello this is test data";
fwrite($myfile, $data);
fclose($myfile);
```

Learn with Rabbil

PHP file handling example

Open a file to read a single line of text from the file using fgets()

index.php

```php
$myfile = fopen("myfile.txt", "r");
$content=fgets($myfile);
echo $content;
fclose($myfile);
```

Learn with Rabbil

PHP file handling example

Read the contents of a file into a string variable using file_get_contents()

index.php

```php
<?php
$content=file_get_contents("myfile.txt");
echo $content;
?>

```

Learn with Rabbil

PHP file handling example

use file_put_contents() to write data to a file in PHP, that will overwrite the file if it already exists.

index.php

```php
// Write some data to a file
$data = "Hello, world!";
file_put_contents("myfile.txt", $data);
```

FILE_APPEND flag to tell file_put_contents() to append the string "More data\n" to the end of the file instead of overwriting its contents.

index.php

```php
// Write some data to a file
$data = "Hello, world!";
file_put_contents("myfile.txt", $data, FILE_APPEND);
```

PHP Cookies methods

Cookies in PHP are used to store data on the client-side (browser) and then retrieve the data when the user returns to the website. Here are the main methods used for working with cookies in PHP: