8. PHP

1) Why PHP & what is this?
2) Syntax
3) echo and print statement
4) include and require
5) Comment
6) Datatype & Variable
7) Scope
8) Operator
9) Control Structures (Loop, Condition)
10) Function
11) PHP Superglobal
12) JSON
13) File read write
14) Built-in Function & Method
15) OOP (Object-Orient Programming)
16) cURL
17) MySQL
18) Namespace
19) Composer

1) Why PHP?

Php ဆိုတာကတော့ server side script language တခုဖြစ်ပါတယ်။ scripting ဆိုတာကွန်ပျူတာတစ်လုံးရဲ့လုပ်ဆောင်ရမယ့်ညွှန်ကြားချက်အစုတစ်ခုပါ။ဒါကို server side ဘက်မှာရေးသားခြင်းဖြစ်ပါတယ်။

Phpကိုသုံးရတဲ့အားသာချက်ဖြချက်ရှိပါတယ်။

1. လေ့လာရလွယ်ကူစေပါတယ်။
2. Open Source ဖြစ်တယ်။
3. Strong Community support ဖြစ်တယ်။
4. Processing မြန်ဆန်ပြီးတော့ Secureဖြစ်ပါတယ်။
5. Database နဲ့ easy to connect လုပ်နိုင်ပါတယ်။
   MySQL နဲ့တွဲသုံးတာများပါတယ်။
6. Support ပေးတဲ့ platformတွေများစွာရှိပါတယ်။
   (Window,MacOS,Linux..etc.)

Php ကို server,browser တို့မလိုပဲ Command line ကနေလည်း run လို့ရနိုင်ပါတယ်။

Php ရဲ့ Case sensitive နဲ့ insensitive တွေကတော့

Case sensitive

Variables,constants,array keys,class properties,class constants

Case insensitive

Functions,class constructors,class methods,keywords,constructs (if, else, null, foreach, echo etc.)

2) Syntax

```
<?php
// code ;
?>
```

Phpရဲ့file extension ကို.php ဖိုင်နဲ့ saveရပါတယ်။Php ကိုနှစ်မျိုးခွဲပြီး ရေးသားနိုင်ပါတယ်။php only နဲ့ရေးလို့ရသလို html နဲ့လည်းတွဲရေးနိုင်ပါသေးတယ်။

※ အရေးကြီးဆုံးအချက်ကတော့ code တကြောင်းပြီးရင်Semicolon (;)နဲ့ အဆုံးသတ်ပေးရပါတယ်။

Php only

Example: Index.php

```
<?php
        echo “Hey,I am Here”;
?>
```

Php ကိုစတင်ရေးသားမယ်ဆို <? နဲ့စရပြီး ?> နဲ့အဆုံးသတ်ရပါတယ်။ နောက်တခုကတော့ file ထဲမှာ php code တွေပဲရှိတဲ့အခါ အဆုံးသတ် ?> မရေးလည်းရပါတယ်။

With Html

Example: Index.php

```
<html>
<body>
<?php
        echo "Php code with html ";
?>
</body>
</html>
```

Php ကို html နဲ့တွဲရေးတဲ့အခါ မှာဆိုရင်တော့ ?> ကိုမရေးလို့မရပါဘူး။

■ **Phpinfo()**

ကိုယ့်စက်ထဲမှာထည့်ထားတဲ့ Php ရဲ့ information တွေကိုတစုတစည်းတည်း ပြချင်တဲ့အခါ phpinfo() ကို သုံးပြီးထုတ်ကြည့်လို့ရပါတယ်။

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

PHP Version 8.0.11

System	Darwin Deathnote-Device-Macbook-pro.local 20.3.0 Darwin Kernel Version 20.3.0: Thu Jan 21 00:07:06 PST 2021; root:xnu-7195.81.3~1/RELEASE_X86_64 x86_64
Build Date	Sep 23 2021 08:57:54
Build System	Darwin osx-compilation-1010-vm-macmini-3 14.5.0 Darwin Kernel Version 14.5.0: Wed Jul 29 02:26:53 PDT 2015; root:xnu-2782.40.9~1/RELEASE_X86_64 x86_64
Configure Command	'./configure' '--prefix=/Applications/XAMPP/xamppfiles' '--with-apxs2=/Applications/XAMPP/xamppfiles/bin/apxs' '--with-config-file-path=/Applications/XAMPP/xamppfiles/etc' '--with-mysql=mysqlnd' '--enable-inline-optimization' '--disable-debug' '--enable-bcmath' '--enable-calendar' '--enable-ctype' '--enable-ftp' '--enable-gd-native-ttf' '--enable-magic-quotes' '--enable-shmop' '--disable-sigchild' '--enable-sysvsem' '--enable-sysvshm' '--enable-wddx' '--with-gdbm=/Applications/XAMPP/xamppfiles' '--with-jpeg-dir=/Applications/XAMPP/xamppfiles' '--with-png-dir=/Applications/XAMPP/xamppfiles' '--with-freetype-dir=/Applications/XAMPP/xamppfiles' '--with-zlib=yes' '--with-zlib-dir=/Applications/XAMPP/xamppfiles' '--with-openssl=/Applications/XAMPP/xamppfiles' '--with-xsl=/Applications/XAMPP/xamppfiles' '--with-ldap=/Applications/XAMPP/xamppfiles' '--with-gd' '--with-imap=/bitnami/xamppunixinstaller80stack-osx-x64/src/imap-2007e' '--with-imap-ssl' '--with-gettext=/Applications/XAMPP/xamppfiles' '--with-mssql=shared,/Applications/XAMPP/xamppfiles' '--with-pdo-dblib=shared,/Applications/XAMPP/xamppfiles' '--with-sybase-ct=/Applications/XAMPP/xamppfiles' '--with-mysql-sock=/Applications/XAMPP/xamppfiles/var/mysql/mysql.sock' '--with-mcrypt=/Applications/XAMPP/xamppfiles' '--with-mhash=/Applications/XAMPP/xamppfiles' '--enable-sockets' '--enable-mbstring=all' '--with-curl=/Applications/XAMPP/xamppfiles' '--enable-mbregex' '--enable-zend-multibyte' '--enable-exif' '--with-bz2=/Applications/XAMPP/xamppfiles' '--with-sqlite=shared,/Applications/XAMPP/xamppfiles' '--with-sqlite3=/Applications/XAMPP/xamppfiles' '--with-libxml-dir=/Applications/XAMPP/xamppfiles' '--enable-soap' '--with-xmlrpc' '--enable-pcntl' '--with-mysqli=mysqlnd' '--with-pgsql=shared,/Applications/XAMPP/xamppfiles/' '--with-iconv=/usr' '--with-pdo-mysql=mysqlnd' '--with-pdo-pgsql=/Applications/XAMPP/xamppfiles/postgresql' '--with-pdo_sqlite=/Applications/XAMPP/xamppfiles' '--with-icu-dir=/Applications/XAMPP/xamppfiles' '--enable-fileinfo' '--enable-phar' '--enable-zip' '--enable-mbstring' '--disable-huge-code-pages' 'ac_cv_decimal_fp_supported=no' '--with-libzip' '--with-pear' '--enable-gd' '--with-jpeg' '--with-libwebp' '--with-freetype' '--with-zip' 'PKG_CONFIG_PATH=/Applications/XAMPP/xamppfiles/lib/pkgconfig' 'CFLAGS=-std=gnu99 -Qunused-arguments -UROCKSDB_SUPPORT_THREAD_LOCAL -U__MACH__ -I/Applications/XAMPP/xamppfiles/include/c-client -Qunused-arguments -I/Applications/XAMPP/xamppfiles/include/libpng -I/Applications/XAMPP/xamppfiles/include/freetype2 -O3 -L/Applications/XAMPP/xamppfiles/lib -I/Applications/XAMPP/xamppfiles/include -I/Applications/XAMPP/xamppfiles/include/ncurses -arch x86_64' 'LIBXML_CFLAGS=-I/usr/include/libxml2' 'LIBXML_LIBS=-L/usr/lib -lxml2' 'XSL_CFLAGS=-I/usr/include/libxslt' 'XSL_LIBS=-L/usr/lib -lxslt'
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/Applications/XAMPP/xamppfiles/etc
Loaded Configuration File	/Applications/XAMPP/xamppfiles/etc/php.ini
Scan this dir for additional .ini files	(none)
Additional .ini files parsed	(none)
PHP API	20200930
PHP Extension	20200930
Zend Extension	420200930
Zend Extension Build	API420200930,NTS
PHP Extension Build	API20200930,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	enabled
Zend Memory Manager	enabled
Zend Multibyte Support	provided by mbstring
IPv6 Support	enabled

3) echo & print Statement

Php မှာoutputထုတ်ပြဖို့အတွက် နည်းလမ်းနှစ်ခုရှိပါတယ်။
Echoနဲ့printဖြစ်ပါတယ်။echo နဲ့ print ကတော်တော်များများတူပေမယ့်
ကွဲပြားစေတဲ့ အချက်ကတော့ echo က multiple parameters လက်ခံပြီးတော့ print
ကတော့ တခုပဲလက်ခံနိုင်ပါတယ်။
နောက်တချက်ကတော့ print ထက် echo က display ပြတာပိုမြန်ပါတယ်။

■ **Echo**

```
<? Php echo “I am Developer” ?>
```

သိုမဟုတ်

```
<?= “I am Developer” ?>
```

■ **Print**

```
<?php  print"I am Developer"; ?>
```

※Echoနဲ့print ဟာ arrayတွေကိုပြတဲ့အခါ warning message ပြပါတယ်။
Run ကြည့်တဲ့အခါ Array လို့ပဲပေါ်ပါတယ်။ဒါကြောင့် Array တွေကိုပြချင်တဲ့အခါ
Print-r()ကိုအသုံးပြုပါတယ်။

■ **print_r()**

Arrayတွေကိုပြချင်တဲ့အခါအထဲကinformationအတွေဖြစ်တဲ့ဘယ်အခန်းမှာဘယ်ke
yနဲ့ဘယ်valueပါရှိတယ်ဆိုတဲ့အတွင်းထဲထိပြပေးစေချင်တဲ့အခါအသုံးပြုပါတယ်။

```
<?php
        $programming = [ "fronted" => "Javascript", "backend" => "php"];
        print_r($programming);
 ?>
//Array ( [fronted] => Javascript [backend] => php )
```

■ **var_dump()**

Print_r() ဟာ array တွေထုတ်ပြဖို့အတွက်အသုံးပြုသလိုvar_dump() ဟာလည်းထုတ်ပြနိုင်ပါတယ်။var_dump ကတော့ Array တင်မက Variable တွေပါ detail info ထုတ်ပြပေးနိုင်ပါတယ်။

```
<?php
        $programming = [ "fronted" => "Javascript", "backend" => "php"];
        $name = "John";

        var_dump($programming);
        echo "<br/>";
        var_dump($name)
 ?>
//array(2) { ["fronted"]=> string(10) "Javascript" ["backend"]=> string(3)
"php" }
string(4) "John"
```

4) include & required Statement

Php တွေ တဖိုင်နဲ့တဖိုင်ချိတ်ဆက်ဖို့ သို့မဟုတ်တူညီတဲ့ function တွေ code တွေကို Common ထားပြီးသုံးချင်တဲ့အခါ သုံးပါတယ်။

■ Include

Example:

Header.php

```
<?php
   echo "This is header";
?>
```

Index.php

```
<?php
   include "header.php";
   echo "<br/>This is index.php";
?>
//This is header
//This is index.php
```

■ Required

Example:

Header.php

```
<?php
   echo "This is header";
?>
```

Index.php

```
<?php
   required "header.php";
   echo "<br/>This is index.php";
?>
//This is header
//This is index.php
```

■ **Include Vs Required**

Include နဲ့လုပ်ဆောင်ပုံခြင်းတူပေမယ့် မတူညီတဲ့အချက်ကတော့ Include ဟာ code မှာတခုခုerrorတတ်တဲ့အခါ warning message သာပြပြီးဆက်run ပါတယ်။Command line နဲ့ run ကြည့်တဲ့အခါတွေ့နိုင်ပါတယ်။required ကတော့တခုခုerrorတတ်တာနဲ့ error ပြပြီးဆက်အလုပ်မလုပ်တော့ပါဘူး။

■ **Include_once & required_once**

Include နဲ့ required statement တွေဟာ တခါခေါ်တိုင်း တခါrunနေမှာ ဖြစ်ပါတယ်။သို့ပေမယ် အကြိမ်ရေဘယ်လောက်ခေါ်ခေါ် တခါပဲ Run ချင်တဲ့အခါ include_once သို့မဟုတ် required_once ကို သုံးနိုင်ပါတယ်။

5) Comment

Comment မှာတော့ single line comment နဲ့ Multiline comment ဆိုပြီးနှစ်မျိုးရှိပါတယ်။

Single line

```
<? php
        // coding
?>
```

Multiline

```
<? php
        /*
        Coding
        Coding
        Coding
        */
?>
```

6) Datatype & Variables

DataType

Computer memory ကနေ data တွေကို ကိုယ်စားပြုနိုင်ဖို့Programming language တွေမှာ data type system တခုစီရှိကြပါတယ်။Data type တွေကို ကြေညာတဲ့အခါမှာလည်း နှစ်မျိုးသတ်မှတ်ထားပါတယ်။
Static typing နဲ့ dynamic typing ဖြစ်ပါတယ်။

■ Static type

Compile time မှာတင် datatype ကို check လုပ်ပြီး မှားခဲ့ရင် error တတ်ပြပါတယ်။ Example (Java,C,C++).

■ Dynamic type

Datatype ကို ကြိုတင်သတ်မှတ်မထားပဲ Assign ထည့်ပေးလိုက်တဲ့ Value ပေါ်မူတည်ပြီး type ကို auto ဆုံးဖြတ်ပါတယ်။
(JavaScript,python,PHP)
Php ဟာ Dynamic datatype ဖြစ်ပါတယ်။Dynamic type မှာပါဝင်တဲ့type နှစ်ခုကတော့Scalar နဲ့ Compound Data Type ဖြစ်ပါတယ်။

Scalar Data Types (အခြေခံဖြစ်တဲ့ DataType)

1. String
2. Integer
3. Boolean
4. Float

```
Example:
$name = “John”;
$age = 28;
$isMarried = false;
$height = 1.73;
```

Compound Data Types (series လိုက်သိမ်းတဲ့ DataType)

1. Array
2. Object

Example:
```
$airport = array("yangon", "mandalay", "naypyitaw");
$home = new homeClass();
```

■ Associative Array

Array ကို key value နဲ့ထည့်ပြီးဖန်တီးချင်တဲ့ခါအသုံးပြုပါတယ်။
ပုံမှန်Arrayခန်းတွေဟာ index 0,1,2ဖြစ်ပေမယ့် 0,1,2
အစားကိုယ့်သတ်မှတ်ထားတဲ့ key ထည့်ချင်တဲ့အခါ

```
$cars = array("Audi" => 133,"BMW" => 543,"Toyata" => 234);
var_dump ($cars);

//array(3) { ["Audi"]=> int(133) ["BMW"]=> int(543) ["Toyata"]=> int(234) }
```

Array Loop

```
$cars = array("Audi" => 133,"BMW" => 543,"Toyata" => 234);

foreach($cars as $key => $value){
   echo "My ".$key. " has ".$value. " km<br/>";
}

//My Audi has 133 km
//My BMW has 543 km
//My Toyata has 234 km
```

■ Multidimensional Array

Array ခန်းထဲမှာနောက်ထပ် Array ခန်းတွေ ထပ်ထည့်ထားတာ
ဖြစ်ပါတယ်။

```
$cars = array(
   "95 Petrol"=>array("Audi" ,"BMW" ,"Toyata"),
   "93 Petrol"=>array("Volvo" ,"Ford" ,"Hammer")
);
$cars[“95 Petrol”][0];
//Audi
```

array_push(array-name,value)

Arrayထဲနောက်ထပ်data တွေထပ်ထည့်ချင်တဲ့အခါသုံးပါတယ်။

```
Example:
 <?php
 $cars=array("Audi","BMW");
 array_push($cars,"Toyota");
 var_dump($cars);
 ?>
 //array(3) { [0]=> string(4) "Audi" [1]=> string(3) "BMW" [2]=>
 string(6) "Toyota" }
```

Special Data Types (သီးသန့် DataType)

1. NULL
2. Resource

```
Example:
$money = NULL;
$fp = fopen("index.php",'r');
```

Variable

Syntax

```
$variable_name = value;
```

Phpဟာ Dynamic type ဖြစ်တဲ့အတွက် variable တွေကို ကြေညာတဲ့အခါကြိုတင်သတ်မှတ်ထားစရာမလိုပါဘူး။Phpမှာ variable ကိုစတင်တဲ့အခါ $ နဲ့မဖြစ်မနေစတင်ရပါတယ်။

အပေါ်မှာလေ့လာခဲ့သလိုပဲvariable တွေဟာ Case sensitive ဖြစ်တဲ့အတွက် စာလုံးအကြီးအသေး Capital letter,small letter ကိုတူညီမှုရှိရပါတယ်။ ဥပမာ $car နဲ့ $Car နဲ့ တူညီမှုမရှိပါဘူး။သပ်သပ်စီဖြစ်ပါတယ်။

■ Isset(statement)

Variable တွေသတ်မှတ်ထားလားမထားဘူးဆိုတာကို စစ်ချင်တဲ့အခါ သုံးပါတယ်။

Example:
isset($name)

■ Constants defined

Syntax

```
define( Variable_name, value)
Or
const
```

Example:
define('PI',3.14)
echo PI;

7) Scope

Scope ဆိုတာကတော့ value တွေ၊process တွေ ကိုaccessible လုပ်နိုင်တဲ့ program ရဲ့area တခုဖြစ်ပါတယ်။php မှာတော့ scope 3 မျိုးခွဲခြားထားပါတယ်။

■ Global scope

Variable ကိုglobal scope(function တွေရဲ့အပြင်ဘက်)မှာကြေညာတဲ့အခါ function တွေရဲ့outside မှာပဲ access လုပ်နိုင်ပါတယ်။

Example:

```
<?php
 $a = 10;

   function show(){
      echo "Inside function $a<br>";
   }
   show();
   echo "Outside Function $a"
?>

//Inside function
//Outside Function 10
```

ဒီprogram မှာဆိုရင်တော့ $a ကို functionရဲ့အပြင်ဘက်မှာခေါ်တဲ့ပုံစံနဲ့ အတွင်းဘက်မှာခေါ်တဲ့ပုံစံကွဲပြားနေတာကိုပြထားတာဖြစ်ပါတယ်။ Function ရဲ့အတွင်းထဲမှာ $a ကိုခေါ်တဲ့ အခါ warning (command line တဲ့အခါတွေ့ရပါတယ်)ပြတာကို တွေ့ရပါတယ်။$a ကို global scope မဟုတ်တဲ့ function အတွင်းထဲ ခေါ်သုံးလိုမရနိုင်ပါဘူး။Function ရဲ့outside area မှာတော့global scope ဖြစ်တဲ့အတွက်ခေါ်သုံးနိုင်ပါတယ်။

global keyword

ဒါပေမယ့် ကိုယ့်program ဟာ Global scope မှာကြေညာထားတဲ့ variable ခေါ်သုံးဖို့လိုအပ်လာတဲ့အခါ global Keywordကို အသုံးပြုရပါတယ်။

```
Example:
<?php
 $a = 10;
  function show(){
     global $a;
     echo "Inside function $a<br>";
  }
  show();
  echo "Outside Function $a"
?>
//Inside function 10
//Outside Function 10
```

global keyword ကိုအသုံးလိုက်တဲ့အခါပုံမှန် access မရနိုင်တဲ့ $aကိုခေါ်သုံးလိုရသွားစေပါတယ်။output ကိုကြည့်မယ်ဆို တူညီတဲ့ valueတွေရနေတွေ့ရပါလိမ့်မယ်။

■ Local scope

Local scope ကတော့ function တွေရဲ့အတွင်းထဲမှာvariable ကြေညာထားတာကို local scopeလို့ခေါ်ပါတယ်။

Example:

```
<?php
 $a = 10;
  function show(){
     $a=20;
     echo "Inside function $a<br>";
  }
  show();
  echo "Outside Function $a"
?>
//Inside function 20
//Outside Function 10
```

အပေါ်ကprogramမှာဆိုရင်တော့function ထဲမှာ$a variableကိုကြေညာလိုက်တဲ့အခါ local scope ဖြစ်တဲ့အတွက် Function အတွင်းထဲဘယ်နေရာကယူသုံးသုံး ကြေညာထားတဲ့ value ကိုပဲရရှိမှာဖြစ်ပါတယ်။global scopeရဲ့$a နဲ့ သတ်ဆိုင်ခြင်းမရှိပါဘူး။

■ **Static**

```
<?php
function increase () {
  static $number = 0;
  $number++;
  return $number;
}
echo increase();
echo increase();
echo increase();
?>
```

Function ထဲမှာ variable တခုကိုကြေညာထားပြီး function ကို ထပ်ခါထပ်ခါ ခေါ်နေပေမယ့် အထဲက variable ရဲ့ value ကို တန်ဖိုးပျက်မသွားပဲ ကျန်ရှိနေစေချင်တဲ့အခါ staid keyword ကိုအသုံးပြုပါတယ်။Function အတွင်းထဲမှာသာ valid ဖြစ်ပါတယ်။

8) Operator

Operator ဆိုတာကတော့ processor တွေကို variable တွေကို ဘဉ်လိုaction တွေလုပ်ဆောင်မယ်ဆိုတာ ပြောပြပေးတဲ့ သင်္ကေတတခုဖြစ်ပါတယ်။

1. Arithmetic Operators
2. Assignment Operators
3. Comparison Operators
4. Error Control Operators
5. Execution Operators
6. Incrementing/Decrementing Operators
7. Logical Operators
8. String Operators
9. Array Operators

■ Arithmetic Operators

Operator	Description	Example
+	Addition	$a + $b
-	Subtraction	$a - $b
*	Multipy	$a * $b
/	Division	$a / $b
%	Modulus	$a % $b
**	Exponentiation	$a ** $b

■ Assignment Operators

Operator	Description	Example
=	Assigns values from right side operands to left side operand	$a = $b
+=	Adds right operand to the left operand and assign the result to left operand	$a += $b
-=	Subtracts right operand from the left operand and assign the result to left operand	$a -= $b
*=	Multiplies right operand with the left operand and assign the result to left operand	$a *= $b
/=	Divides left operand with the right operand and assign the result to left operand	$a /= $b
%=	Modulus using two operands and assign the result to left operand	$a %= $b

■ Comparison Operators

Operator	Description	Example
==	**true** if $a is equal to $b after type juggling	$a == $b
===	**true** if $a is equal to $b, and they are of the same type	$a === $b
!=	**true** if $a is not equal to $b after type juggling	$a != $b
<>	**true** if $a is not equal to $b after type juggling	$a <> $b
!==	**true** if $a is not equal to $b, or they are not of the same type	$a !== $b
<	**true** if $a is strictly less than $b	$a < $b
>	**true** if $a is strictly greater than $b	$a > $b
<=	**true** if $a is less than or equal to $b	$a <= $b
>=	**true** if $a is greater than or equal to $b	$a >= $b
<=> (Spaceship)	An int less than, equal to, or greater than zero when $a is less than, equal to, or greater than $b, respectively	$a <=> $b

■ Error Control Operators

@

@opeator ကတော့ statementရဲ့ရှေ့ဆုံးမှားထည့်ပြီးသုံးရတာဖြစ်ပါတယ်။ သူကိုသုံးလိုက်တဲ့အကျိုးကတော့ ရေးလိုက်တဲ့ statement ဟာ error ရှိနေရင်တောင် ပြမှာမဟုတ်ပဲ ignore လုပ်ပေးမှာဖြစ်ပါတယ်။

■ Execution Operators

backticks (``)

```
<?php
$output = `ls -al`;
echo "<p>$output</p>";
?>
```

Backtick နဲ့ရေးလိုက်တဲ့ statement တွေဟာ shell အနေနဲ့ execute လုပ်ပါတယ်။ ဒါပေမယ့် shell_exec() ကို disabled လုပ်ထားရင်တော့ရမှာမဟုတ်ပါဘူး။

■ Incrementing/Decrementing Operators

Operator	Description	Example
++(pre)	Increments $a by one, then returns $a	++$a
(post)++	Returns $a, then increments $a by one	$a++
--(pre)	Decrements $a by one, then returns $a	--$a
(post)--	Returns $a, then decrements $a by one	$a--

■ Logical Operators

Operator	Description	Example
and	**True** if both $a and $b are true	$a and $b
or	**True** if either $a or $b is true	$a or $b
xor	**True** if either $a or $b is true, but not both	$a xor $b
&&	**True** if both $a and $b are true	$a && $b
||	**True** if either $a or $b is true	$a || $b
!	**True** if $a is not true	!$a

■ String Operators

Operator	Description	Example
.	Concatenation	$a . $b
.=	Concatenation assignment	$a .= $b

■ Array Operators

Operator	Description	Example
+	Union of $a and $b	$a + $b
==	**true** if $a and $b have the same key/value pairs	$a === $b
===	**true** if $a and $b have the same key/value pairs in the same order and of the same types.	$a === $b
!=	**true** if $a is not equal to $b	$a != $b
<>	**true** if $a is not equal to $b	$a <> $b
!==	**true** if $a is not identical to $b	$a !== $b

■ Other Conditional Operators

Operator	Description	Example
?:	Ternary	$a = Exp ? Exp1 : Exp2
??	Null coalescing	$a = Exp1 ?? Exp2

9) Control Structure (loop,condition)

If…else

Syntax:

```
if (condition) {
  //  block of code to be executed if the condition is true
}else{
//  block of code to be executed if the condition is false
}
```

conditionတွေမှန်မမှန်စစ်တဲ့အခါဖြစ်လာနိုင်တဲ့အမျိုးအစားသိပ်မများတဲ့အခါ ifကိုအသုံးများပါတယ်။

Example:

```
<?php
 $age = 19;
 if($age > 18){
    echo "You can drink Beer.";
 }else{
    echo "You are not allowed to drink Beer.";
 }
?>
```

Switch

Syntax:

```
switch(expression) {
  case x:
    // code block
    break;
  case y:
    // code block
    break;
  default:
    // code block
}
```

Switch ကတော့ multiple case တွေစစ်ချင်တဲ့အခါသုံးနိုင်ပြီးတော့ ဖြစ်လာနိုင်တဲ့ case တွေဟာ အများကြီးရှိနိုင်တယ်ဆိုရင်တော့ switchကိုသုံးပါတယ်။ Break;ကတော့ရှေ့ဆက်မသွားတော့ပဲ switch ထဲထွက်မယ်ဆိုတဲ့ keyword ဖြစ်ပါတယ်။

```
Example:
<?php
 $phone = "Apple";
 switch ($phone) {
      case 'Apple':
        echo "Cost $1300";
        break;
      case 'Sony':
        echo "Cost $1100";
        break;
      case 'Samsung':
        echo "Cost $1000";
        break;
      case 'Nokia':
        echo "Cost $800";
        break;
    default:
        echo "Wrong Chioce";
        break;
 }
?>
```

For

Syntax:

```
for (statement 1; statement 2; statement 3) {
  // code block to be executed
}
```

အကြိမ်ရေတကြိမ်ထက်မကရှိနေတဲ့ process တွေကိုရေးချင်တဲ့အခါfor loop ကိုသုံးနိုင်ပါတယ်။

Example:

```
<?php
 for ($i=0; $i < 9; $i++) {
    echo "Number is".$i ."<br>";
 }
?>
```

Foreach

Array တွေကို ထုတ်ပြတဲ့အခါ အသုံးများပါတယ်။

Syntax

```
foreach ($variable as $key => $value) {
    # code...
 }
```

Example:

```
<?php
$array = array("Audi"=> 3000,"BMW" => 3500,"Toyota" => 2000);
 foreach ($array as $key => $value) {
    echo $key. " is $".$value."<br>";
 }
?>
```

While

Syntax

```
while (condition) {
 // code block to be executed
}
```

While ကတော့ပေးထားတဲ့condition မှန်နေသ၍loopingဖြစ်နေမှာပဲဖြစ်ပါတယ်။ အရေးကြီးတာကတော့ while loop ကိုရပ်တန့်ဖို့Condition ပေးရပါတယ်။

Example:

```
<?php
$a= 1;

while ($a <= 15) {
   echo $a. "<br>";
   $a++;
}
?>
```

DoWhile

Syntax

```
do {
  // code block to be executed
}
while (condition);
```

While နဲ့ဆင်တူပါတယ်။အဓိကကွာခြားချက်ကတော့ do whileဟာ ဘာပဲဖြစ်ဖြစ်တကြိမ်အလုပ်လုပ်ပါတယ်။ပြီးမှconditionကိုစစ်ပြီး ဆက်လုပ်မလုပ်ဆုံးဖြတ်ပါတယ်။

Example:

```
<?php
$a= 10;

do{
   echo $a;
}while($a>10)
?>
```

10) Function

Function ဆိုတာက machine တခုနဲ့ဆင်တူပါတယ်။input ဝင်လာတဲ့value ပေါ်မူတည်ပြီး outputကိုထုတ်ပေးတာဖြစ်ပါတယ်။Function ကို Name ပေးတဲ့အခါ သတ်မှတ်တဲ့တဲ့ပုံစံတွေရှိပါတယ်။

Camel

myFunction()

Lower case

my_function()

Pascal

MyFunction()

Function ကိုကြေညာတဲ့အခါ JavaScript လိုမျိုး ကြေညာနိုင်ပါတယ်။

Example:

```
<?php
  function showMessage(){
     echo "Messaage";
  }
  // function call
  showMessage();
?>
```

Php7 နောက်ပိုင်းမှာတော့ parameter တွေကို datatype တွေသတ်မှတ်ပြီး ရေးလာနိုင်ပါတယ်။

Example:

```
<?php
  function showMessage(string $name,int $age){
     echo "My Name is ".$name." and age is ".$age;
  }
  // function call with name and age
  showMessage("John",24);
?>
```

Parameter ကိုdefault value သတ်မှတ်ပြီး optional အဖြစ်ထားနိုင်ပါတယ်။ Optional ဆိုတာကတော့ function call တဲ့အခါ အဲ့ဒီparameter မပါလည်း function call နိုင်ပါတယ်။

Example:

```
<?php
   function showMessage(string $name,int $age,string $address =
"Yangon"){
      echo "My Name is ".$name." age is ".$age." and I live in
".$address;
   }
   // function call with name and age
   showMessage("John",24);
   echo "<br>";
   showMessage("John",24,"Mandalay");
?>
//My Name is John age is 24 and I live in Yangon
//My Name is John age is 24 and I live in Mandalay
```

11) PHP Super-global

Superglobal ဆိုတာကတော့ php မှာdefault ပါလာပြီးသား variable ဖြစ်ပါတယ်။ ကြေညာထားစရာမလိုပဲ ခေါ်သုံးလို့ရတဲ့ variable တွေဖြစ်ပါတယ်။Pageတခုကနဲ့ ရလာတဲ့data တွေကို storeလုပ်တာ၊ရယူတာစတဲ့နေရာတွေအတွက်အသုံးပြုပါတယ်။ Client ကနဲ့ပို့လိုက်တဲ့data တွေကို လက်ခံတာ ထုတ်ပြတာ စီမံတာ စတဲ့ client ကနဲ့ရတဲ့data တွေကို ဖမ်းပေးတဲ့အနေနဲ့သုံးပါတယ်။Superglobal ၉မျိုးရှိပါတယ်။

1. $GLOBALS
2. $_SERVER
3. $_REQUEST
4. $_POST
5. $_GET
6. $_FILES
7. $_COOKIES
8. $_SESSION
9. $_ENV

အသုံးများတဲ့ Superglobal ၆ခုကတော့

■ **$GLOBALS**

Variableကို$GLOBALSထဲထည့်ပြီး phpရဲ့နေရာတိုင်းကနေခေါ်သုံးနိုင်ပါတယ်။

Example:

```
<?php
$a = 100;
function sum3(){
   $GLOBALS['b']= $GLOBALS['a']+3;
}
sum3();
echo $b;
?>
```

■ **$_GET**

Clientကနေform tagမှာget methodနဲ့ပို့လိုက်တဲ့dataတွေကိုထုတ်ကြည့်တဲ့အခါ သုံးပါတယ်။Get နဲ့ပို့လိုက်တဲ့ user data တွေဟာ browserရဲ့address bar မှာ ပေါ်နေမှာဖြစ်လို့ security အရ weak ဖြစ်ပါတယ်။Client ဘက်က form tag နဲ့ပို့လိုက်ပြီး$_Get နဲ့ ပြန်ထုတ်ကြည့်တဲ့အခါinput,select,textarea စတဲ့ tagတွေရဲ့nameတွေနဲ့ပြန်ထုတ်ကြည့်ပါတယ်။ဒါကြောင့်nameတွေပေးခဲ့ဖို့ အရေးကြီးပါတယ်။

Example:

Form.html

```
<!DOCTYPE html>
<html lang="en">
 <head>
     <title>Document</title>
 </head>
 <body>
  <form action="post.php" method="get">
   <input type="text" name="name" />
   <input type="text" name="age" />
   <button type="submit">Send</button>
  </form>
 </body>
</html>
```

Post.php

```
<?php
 if(isset($_GET)){
    echo "Your Name :". $_GET["name"]."<br>";
    echo "Your Address :".$_GET["age"]."<br>";
    var_dump($_GET);
  }
?>
```

■ **$_POST**

Client ဆီကpost method နဲ့ server ဆီပေးပို့လိုက်တဲ့data တွေကို server ဘက်ကပြန်ယူတဲ့အခါအသုံးပြုပါတယ်။Get လိုပဲ့ tag တွေမှာname တွေပေးခဲ့ဖို့အရေးကြီးပါတယ်။

Get နဲ့ပို့လိုက်တဲ့ dataတွေဟာ address bar မှာပေါ်နေတဲ့အတွက် Security အရလုံခြုံမှုအားနည်းတာကြောင့်အရေးမကြီးတဲ့Dataတွေ transfer လုပ်တဲ့အခါ မှာပဲ get ကိုသုံးကြပါတယ်။Post နဲ့ ပို့လိုက်တဲ့data တွေဟာ browserရဲ့Address bar မှာ parameters အနေနဲ့ပြနေမှာမဟုတ်ပါဘူး။

Example:

Form.html

```
<!DOCTYPE html>
<html lang="en">
 <head>
     <title>Document</title>
 </head>
 <body>
  <form action="post.php" method="post">
   <input type="text" name="name" />
   <input type="text" name="age" />
   <button type="submit">Send</button>
  </form>
 </body>
</html>
```

Post.php

```
<?php
 if(isset($_POST)){
    echo "Your Name :". $_POST["name"]."<br>";
    echo "Your Address :".$_POST["age"]."<br>";
    var_dump($_POST);
 }
?>
```

■ **$_FILES**

Client ကနေ file upload လုက်လိုက်တဲ့ file တွေကို server ဘက်ကဖမ်းယူတဲ့အခါသုံးပါတယ်။File upload လုပ်မည့်html form tag မှာတော့ method post ဖြစ်ပြီးတော့File ကို encrypt လုပ်ပြီးပို့မှာဖြစ်တဲ့ အတွက် enctype="multipart/form-data"ကိုမဖြစ်မနေထည့်ပေးရပါတယ်။

Example:

Form.html

```
<!DOCTYPE html>
<html lang="en">
 <head>
    <title>Document</title>
 </head>
 <body>
  <form action="upload.php" method="post"
enctype="multipart/form-data">
    <input type="file" name="fileupload" />
    <button type="submit">Upload</button>
  </form>
 </body>
</html>
```

Upload.php

```
<?php
$file = $_FILES['fileupload']['name'];
$location =  $_FILES['fileupload']['tmp_name'];
$extension = pathinfo($file)['extension'];
$filename = pathinfo($file)['filename'];

if(move_uploaded_file($location,"upload/".$file.".".$extension)){
 echo 'File is successfully uploaded.';
}
else{
 echo 'There was some error moving the file to upload directory.';
}
?>
```

Advance file Upload Example

```
<?php
if (isset($_POST['submit'])) {
   $file = $_FILES['file'];
   $name = $_FILES['file']['name']; //Find file name
   $tmp_name = $_FILES['file']['tmp_name']; //Temp loc
   $size = $_FILES['file']['size']; //Find file size
   $error = $_FILES['file']['error']; //Find errors

   //Explode from punctuation mark
   $tempExtension = explode('.', $name);

   $fileExtension = strtolower(end($tempExtension));

   //Allowed extensions
   $isAllowed = array('jpg', 'jpeg', 'png', 'pdf');

   // 0 = no error - 1 = error
   if (in_array($fileExtension, $isAllowed)) {
      if ($error === 0) {
         if ($size < 100000) {
            $newFileName = uniqid('', true) . "." . $fileExtension;
            $fileDestination = "uploads/" . $newFileName;
            move_uploaded_file($tmp_name, $fileDestination);
            header("Location: files.php?uploadedsuccess");
         } else {
            echo "Sorry, your file size is too big!";
         }
      } else {
         echo "Sorry, there was an erro! Try it again";
      }
   } else {
      echo "Sorry, your file type is not accepted";
   }

}
?>
```

■ $_COOKIES

ကိုယ်ပိုင် cookie တွေကိုတည်ဆောက်လို့ရပါတယ်။client ရဲ့computerထဲမှာ သိမ်းထားနိုင်တဲ့သေးငယ်တဲ့file တခုဖြစ်ပါတယ်။ဥပမာ usernameလိုမျိုး server side ကို request လုပ်တဲ့အခါ user က input box ထဲမှာထပ်ရိုက်နေစရာ မလိုအောင် cookies ထဲမှာထည့်ထားပြီး ပြန်ထုတ်ပြတဲ့အခါမျိုးမှာသုံးပါတယ်။ Cookies မှာparameter ၇ခုပါဝင်ပါတယ်။

Syntax

```
setcookie(name, value, expire, path, domain, secure, httponly);
```

Nameနဲ့valueသာမဖြစ်မနေထည့်ပေးရမယ့်parameterဖြစ်ပါတယ်။ကျန်တာတွေကတော့ optional ဖြစ်ပြီးမထည့်လည်းရပါတယ်။

Name : သတ်မှတ်ထားမယ့်cookie name

Value : cookie ရဲ့value။

Example:

```
<?php
 $time = time() + 86400 * 30;
 setcookie("name","John",$time);
 print_r($_COOKIE);
?>
//Array ( [Phpstorm-3272c4a3] => 59ae072a-3b92-4167-a [name] => John )
```

Cookies ကိုပြန်ဖျက်ချင်တဲ့အခါ အချိန်ကို past time ပေးတာ သို့မဟုတ် Valueကို “”ပြန်ပေးပြီးဖျက်လို့ရပါတယ်။

■ **$_SESSION**

Cookies မှာတော့ user related ဖြစ်တဲ့data တွေကိုသာမှတ်ထားကြပါတယ်။ hacker တွေဟာ cookies information တွေကို tracking လုပ်ပြီး attacking လုပ်နိုင်တာကြောင့် အရေးကြီးတဲ့ information တွေဖြစ်တဲ့ useridၤ Password စတဲ့ information တွေကိုတော့ Cookies နဲ့မသုံးပဲSession နဲ့သုံးကြပါတယ်။ Session ဟာ server မှာstore လုပ်ထားတာကြောင့်browser ပိတ်လိုက်တာနဲ့ information တွေ ကိုဖျက်ပေးမှာဖြစ်ပါတယ်။

Session အသုံးပြုချင်တဲ့အခါ

session_start() Method နဲ့စတင်ရပါတယ်။အသုံးပြုတဲ့ပုံစံကတော့

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

    // set the session variable
    $_SESSION["username"] = "John ";
    $_SESSION["age"] = 24;
?>
```

Session ဖျက်ချင်တဲ့အခါ **session_destroy();** ကိုအသုံးပြုပါတယ်။

12) JSON

PHP မှာ json ကနေ object သို့မဟုတ် array၊ array ကနေ json ကို convert လုပ်နိုင်ဖို့ Build in method ပါရှိပါတယ်။

■ json_encode($value)

Resource ကလွဲတဲ့ကျန် datatype တွေကို json အနေနဲ့convert လုပ်ပေးပါတယ်။Array ကနေ json ကို convert လုပ်ပြီးသုံးတာများပါတယ်။

Example:

```
<?php
$arr = array('a' => 1, 'b' => 2, 'c' => 3, 'd' => 4, 'e' => 5);

echo json_encode($arr);
?>
//{"a":1,"b":2,"c":3,"d":4,"e":5}
```

■ json_decode($jsonString,$associative,$depth,$flags)

Json string ကနေ array or object ကို convert လုပ်ပါတယ်။**$associative** ရဲ့default value ကတော့ false ဖြစ်ပါတယ်။Object နဲ့ရချင်တဲ့အခါdefault အတိုင်းသုံးရပြီးတော့ array အနေနဲ့ convert လုပ်ချင်တဲ့အခါ true ကိုထည့်ပေးရပါတယ်။

Example:

```
<?php
$json = '{"a":1,"b":2,"c":3,"d":4,"e":5}';
var_dump(json_decode($json));
echo "<br/>";
var_dump(json_decode($json, true));
?>
//object(stdClass)#1 (5) { ["a"]=> int(1) ["b"]=> int(2) ["c"]=> int(3)
["d"]=> int(4) ["e"]=> int(5) }
array(5) { ["a"]=> int(1) ["b"]=> int(2) ["c"]=> int(3) ["d"]=> int(4)
["e"]=> int(5) }
```

13) File read write

Php မှာ client က upload တင်လိုက်တဲ့ file သို့မဟုတ် server ပေါ်file ကို read Write လုပ်နိုင်ဖို့ build in function ပါဝင်ပါတယ်။

fopen(directory,mode)

File တခုကို စတင်တည်ဆောက်ခင်တဲ့အခါ fopen method ကိုသုံးရပါတယ်။ Parameterတွေအနေနဲ့ကတော့directoryfileတည်ရှိနေမယ်pathလမ်းကြောင်း၊ Mode ကတော့ file ကို read or write လုပ်ချင်တဲ့ mode ကိုရေးပေးရပါတယ်။

```
Example:
<?php
  $myFile = fopen("upload/file.txt","w");
  echo $myFile;
?>
```

fwrite($file,$txt)

ရှိနေတဲ့ file ကို ဖွင့်ပြီး ရေးချင်တဲ့အခါ fwrite ကို အသုံးပြုပါတယ်။file ကိုwrite ပြီးတဲ့အခါ **fclose()**နဲ့ ပြန်ပိတ်ပေးလိုပါတယ်။

```
Example:
<?php
  $myFile = fopen("upload/file.txt","w");
  $txt = "Hay,This is my first writing";

  fwrite($myFile,$txt);
  fclose($myFile);

?>
```

Upload/File.txt မှာ ရေးလိုက်တဲ့စာသား “Hay,This is my first writing” ပေါ်နေတာတွေ့ရပါလိမ့်မယ်။

file_get_contents($filepath)

Fileထဲမှာရေးထားတဲ့ text တွေကိုပြန်ဖတ်ချင်တဲ့အခါ အသုံးပြုပါတယ်။

Example:

```
<?php
  $filePath = "upload/file.txt";
  $line= file_get_contents($filePath);

  echo $line;
?>
```

Write text with form tag

```
<html>
<head>
   <meta charset="UTF-8">
   <title>Document</title>
</head>
<body>
<?php

if (isset($_POST['submit'])) {
   $myFile = fopen("uploads/file.txt", "a");

   $txt = "My age is " . $_POST['age'] . "\n";

   fwrite($myFile, $txt);

   fclose($myFile);
}
?>

<form action="fileput.php" method="post">
   <input type="text" name="age">
   <input type="submit" name="submit">
</form>

</body>
</html>
```

14) Build_in Method & Function

String Function

- **nl2br($string)**

```
$str = "Today, we learn PHP.
   So Easy.
   Keep Going.";
echo nl2br($str);

//Today, we learn PHP.
//So Easy.
//Keep Going.
```

- **explode(separator,$string)**

```
$str = "BMW Toyota Ford Suzuki";
print_r(explode(" ", $str));

//Array ( [0] => BMW [1] => Toyata [2] => Ford [3] => Suzuki )
```

- **trim($string)**

```
$str = "      Tomorrow will be hot.       ";
echo trim($str);

//Tomorrow will be hot.
```

- **ltirm($string)**

```
$str = "      Tomorrow will be hot.       ";
echo ltrim($str);

//Tomorrow will be hot.
```

- **rtrim($string)**

```
$str = "      Tomorrow will be hot.       ";
echo rtrim($str);

//Tomorrow will be hot.
```

■ **strlent($string)**

```
$str = "Tomorrow will be hot.";
echo strlen($str);

//21
```

■ **strrev($string)**

```
$str = "Tomorrow will be hot.";
echo strrev($str);

//.toh eb lliw worromoT
```

■ **str_word_count($string)**

```
$str = "Tomorrow will be hot.";
echo str_word_count($str);

//4
```

■ **strtoupper($string)**

```
$str = "Tomorrow will be hot.";
echo strtoupper ($str);

//TOMORROW WILL BE HOT.
```

■ **strtolower($string)**

```
$str = "Tomorrow will be hot.";
echo strtolower($str);

//tomorrow will be hot.
```

■ **ucfirst($string)**

```
$str = "tomorrow";
echo ucfirst($str);

//Tomorrow
```

■ **ucwords($string)**

```
$str = "tomorrow will be hot";
echo ucwords($str);

// Tomorrow Will Be Hot
```

■ **strpos($string,'find_words')**

```
$str = "tomorrow will be hot";
echo strpos($str, "be");

//14
```

■ **stripos($string,'find_words')**

```
$str = "tomorrow will be hot";
echo stripos($str, "be");

//14
```

■ **str_replace($strold,$strnew,$string)**

```
$str = "tomorrow will be hot";
echo str_replace("hot", "cool", $str);

//tomorrow will be cool
```

■ **str_ireplace($strold,$strnew,$string)**

```
$str = "tomorrow will be hot";
echo str_ireplace("HOT", "cool", $str);

//tomorrow will be cool
```

■ **number_format($stringORnumber,decimal, decimalseparator,separator)**

```
$str = "3450.345";
echo number_format($str, 2);

//3,450.35
```

Number Function

- **ceil($number)**

```
echo(ceil(5.1));     //6
```

- **floor($number)**

```
echo(floor(5.1));    //5
```

- **abs($number)**

```
echo(abs(-14));     //14
```

- **pow($number)**

```
echo(pow(3,4));    //81
```

- **mt_rand($min,$max)**

```
echo(mt_rand(10,1000));
```

- **min($number1,$number2)**

```
echo(min(15,18));   //15
```

- **max($number1,$number2)**

```
echo(max(15,18));  //18
```

Array Function

- **count($array)**

```
$array = [“BMW”,”Toyata”,”Suzuki”,”Ford”];
Echo count($array);    //4
```

- **implode(“separator”, $array)**

```
$array = ["BMW", "Toyata", "Suzuki", "Ford"];
    echo implode(",", $array);

//BMW,Toyata,Suzuki,Ford
```

■ **array_push($array,$value)**

```
$array = ["BMW", "Toyata", "Suzuki", "Ford"];
    array_push($array, "KIA");
    var_dump($array);

    //array(5) {
     [0]=>
     string(3) "BMW"
     [1]=>
     string(6) "Toyata"
     [2]=>
     string(6) "Suzuki"
     [3]=>
     string(4) "Ford"
     [4]=>
     string(3) "KIA"
    }
```

■ **array_pop($array)**

```
    $array = ["BMW", "Toyata", "Suzuki", "Ford"];
    array_pop($array, "KIA");
    var_dump($array);

    //array(3) {
     [0]=>
     string(3) "BMW"
     [1]=>
     string(6) "Toyata"
     [2]=>
     string(6) "Suzuki"
    }
```

■ **in_array($value,$array)**

```
    $array = ["BMW", "Toyata", "Suzuki", "Ford"];
    var_dump(in_array("Ford", $array));

    //bool(true)
```

■ **array_unshift($array)**

```
$array = ["BMW", "Toyata", "Suzuki", "Ford"];
array_unshift($array, "KIA");
var_dump($array);

//array(5) {
 [0]=>
 string(3) "KIA"
 [1]=>
 string(3) "BMW"
 [2]=>
 string(6) "Toyata"
 [3]=>
 string(6) "Suzuki"
 [4]=>
 string(4) "Ford"
}
```

■ **array_shift($array)**

```
$array = ["BMW", "Toyata", "Suzuki", "Ford"];
array_unshift($array, "KIA");
var_dump($array);

array(3) {
 [0]=>
 string(6) "Toyata"
 [1]=>
 string(6) "Suzuki"
 [2]=>
 string(4) "Ford"
}
```

■ **array_search($value,$array)**

```
$array = ["BMW", "Toyata", "Suzuki", "Ford"];
var_dump(array_search("Ford", $array));

//int(3)
```

■ **array_keys($array)**

```
$array = ["BMW" => 1113, "Toyata" => 335, "Suzuki" => 2324, "Ford" =>
244];
var_dump(array_keys($array));

array(4) {
 [0]=>
 string(3) "BMW"
 [1]=>
 string(6) "Toyata"
 [2]=>
 string(6) "Suzuki"
 [3]=>
 string(4) "Ford"
}
```

■ **array_values($array)**

```
$array = ["BMW" => 1113, "Toyata" => 335, "Suzuki" => 2324, "Ford" =>
244];
var_dump(array_values($array));

array(4) {
 [0]=>
 int(1113)
 [1]=>
 int(335)
 [2]=>
 int(2324)
 [3]=>
 int(244)
}
```

■ **array_merage($array1,$array2)**

```
$array1 = ["BMW", "Toyata", "Suzuki", "Ford"];
    $array2 = ["Hummer", "Jeep"];
    var_dump(array_merge($array1, $array2));

    array(6) {
     [0]=>
     string(3) "BMW"
     [1]=>
     string(6) "Toyata"
     [2]=>
     string(6) "Suzuki"
     [3]=>
     string(4) "Ford"
     [4]=>
     string(6) "Hummer"
     [5]=>
     string(4) "Jeep"
           }
```

■ **sort($array)**

```
    $array = ["BMW", "Toyata", "Suzuki", "Ford"];
    sort($array);
    var_dump(($array));

    array(4) {
     [0]=>
     string(3) "BMW"
     [1]=>
     string(4) "Ford"
     [2]=>
     string(6) "Suzuki"
     [3]=>
     string(6) "Toyata"
    }
```

■ **rsort($array1)**

```
$array = ["BMW", "Toyata", "Suzuki", "Ford"];
rsort($array);
var_dump(($array));
```

```
array(4) {
  [0]=>
  string(6) "Toyata"
  [1]=>
  string(6) "Suzuki"
  [2]=>
  string(4) "Ford"
  [3]=>
  string(3) "BMW"
}
```

■ **aort($array)**

```
$array = ["BMW" => 1113, "Toyata" => 335, "Suzuki" => 2324];
asort($array);
var_dump($array);
```

```
array(3) {
  ["Toyata"]=>
  int(335)
  ["BMW"]=>
  int(1113)
  ["Suzuki"]=>
  int(2324)
}
```

■ **ksort($array1)**

```
$array = ["BMW" => 1113, "Toyata" => 335, "Suzuki" => 2324];
asort($array);
var_dump($array);

array(3) {
 ["BMW"]=>
 int(1113)
 ["Suzuki"]=>
 int(2324)
 ["Toyata"]=>
 int(335)}
```

Date Function

■ **date('Format')**

```
$today = date("F j, Y, g:i a");                 // March 10, 2001, 5:16 pm
$today = date("m.d.y");                         // 03.10.01
$today = date("j, n, Y");                       // 10, 3, 2001
$today = date("Ymd");                           // 20010310
$today = date("D M j G:i:s T Y");               // Sat Mar 10 17:16:18 MST 2001
$today = date("H:i:s");                         // 17:16:18
$today = date("Y-m-d H:i:s");                   // 2001-03-10 17:16:18
```

File Function

■ **mkdir('foldername')**

```
mkdir("img");
```

■ **rmdir('folder')**

```
rmdir("image");
```

■ **rename('old','new')**

```
rename("img", "image");
```

■ **rename('old','new')**

```
scandir("./");
```

15) OOP (Object-Orient Programming)

Object oriented programming ဆိုတာကတော့ programming တခုရဲ့paradigm တခုဖြစ်ပြီး class တွေ object တွေနဲ့တည်ဆောက်ထားပါတယ်။
Class ဆိုတာကတော့ အစုအဖွဲ့တခုဖြစ်ပါတယ်။Php မှာ Class တွေအသုံးပြုပြီး Object တွေတည်ဆောက်နိုင်ပါတယ်။

Syntax

```
class name{

}
```

Access Modifier

Class ထဲမှာရေးသားတဲ့ properties နဲ့ MethodတွေကိုAccess Modifierလို့ခေါ်ပါတယ်။Class ထဲမှာ properties နဲ့ methodတွေကြေညာတဲ့အခါ သုံးမျိုးရှိပါတယ်။

Public

နေရာတိုင်းကခေါ်သုံးနိုင်တဲ့ Access Modifier ဖြစ်ပါတယ်။

```
class Person
{
   public $name;
   public $age;

   public function show()
   {
      echo "Name:" . $this->name . " and Age:" . $this->age;
   }
}

$p = new Person();
$p->name = "John";
$p->age = 34;
$p->show();
```

Private

ကြေညာထားတဲ့ class အတွင်းမှာပဲအသုံးပြုနိုင်ပါတယ်။

```
class Person
{
   private $name;
   private $age;

   public function show()
   {
      echo "Name:" . $this->name . " and Age:" . $this->age;
   }

   public function __construct($name, $age)
   {
      $this->name = $name;
      $this->age = $age;
   }
}
$p = new Person("Brad", 16);
$p->show();
```

Protected

Main class နဲ့ဆက်ဆက်နေတဲ့ classအတွင်းထဲမှာပဲအသုံးပြုနိုင်ပါတယ်။

```
class Person
{
   protected $firstName;
   protected $age;

   public function __construct($name, $age)
   {
      $this->firstName = $name;
      $this->age = $age;
   }
}
```

```
class Employee extends Person
{
   public function getInfo()
   {
      echo "Employee Name:$this->firstName and $this->age";
   }
}


$emp = new Employee("John", 23);
$emp->getInfo();
```

Constructor and Destructor

Constructor ဆိုတာကတော့ class ကိုခေါ်လိုက်တဲ့အချိန်မှာ စတင်အလုပ်လုပ်တဲ့ function ဖြစ်ပါတယ်။ Constructor တည်ဆောက်တဲ့အခါ __ နဲ့စတင်ရပါတယ်။ Destructorကတော့ class ကိုခေါ် ပြီးနောက်ဆုံး အလုပ်လုပ်တဲ့function ဖြစ်ပါတယ်။

```
class Person
{
   protected $firstName;
   protected $age;

   public function __construct($name, $age)
   {
      $this->firstName = $name;
      $this->age = $age;
   }
   public function __destruct()
   {
      echo "Name : $this->firstName , Age: $this->age";
   }
}
$person = new Person("John", 23);
//Name : John , Age: 23
```

Unset($object)

Object ကို destroy သို့မဟုတ် delete လုပ်ချင်တဲ့အခါ အသုံးပြုပါတယ်။

Constants

Class ရဲ့properties မှာပြောင်းလဲလို့မရတဲ့ properties သတ်မှတ်ချင်တဲ့အခါ Constant keywordကို အသုံးပြုပါတယ်။

Classထဲက ခေါ်တဲ့အခါ self:: $properties နဲ့ခေါ်ရပြီးတော့ Public ကနေ ခေါ်တဲ့အခါ ClassName ::$properties နဲ့ ခေါ် နိုင်ပါတယ်။

```
class Greetings{
  const MESSAGE = "Hello! Have a nice day.";

public function say() {
    echo self::MESSAGE;
  }
}

echo Greetings::MESSAGE;

$g = new Greetings();
$g->say();
```

Static Modifier

Class တခုကို call လိုက်တိုင်းdefault တန်ဖိုးထည့်ပြီးသားဖြစ်ချင်တဲ့အခါPublic နဲ့ ကြေညာထားရင်လည်း constructor မပါပဲ တိုက်ရိုက်Access မလုပ်နိုင်အောင် သတ်မှတ်ထားချင်တဲ့အခါလည်း အသုံးပြုပါတယ်။

Classထဲက ခေါ်တဲ့အခါ self:: $properties နဲ့ခေါ်ရပြီးတော့ Public ကနေ ခေါ်တဲ့အခါ ClassName ::$properties နဲ့ ခေါ် နိုင်ပါတယ်။

```
class Person
{
   private $name;
   private $age;
   public static $drinkAge = 18;

   public function show()
   {
      echo "Name:" . $this->name . " and Age:" .
         $this->age . "Drinking Age is" . self::$drinkAge;
   }

}

echo Person::$drinkAge;
```

Php 7.4 နဲ့အထက် မှာတော့ datatype ကိုသတ်မှတ်နိုင်ပါပြီ။

```
class Person
{
   private string $name;
   private int $age;
}
```

Extends Class

Class တခုကို အမွေဆက်ခံပြီး parent class မှာပါတဲ့အတိုင်းextendsလုပ်ချင်တဲ့အခါ သုံးပါတယ်။

Person.php

```
class Person
{
   private ?string $name;
   private int $age;

   public static $drinkAge = 18;

   public function show()
   {
      echo "Name:" . $this->name . " and Age:" .
         $this->age;
   }

   public function __construct($name, $age)
   {
      $this->name = $name;
      $this->age = $age;
   }
}
```

Employee.php

```
class Employee extends Person
{
   public $id;

   public function __construct($name, $age, $id)
   {
      $this->id = $id;
      Parent::__construct($name, $age);
   }
}
```

Interface Class

Extends လုပ်ထားတဲ့Class တွေမှာ လုပ်ဆောင်ရမယ် ခေါင်းစဉ်ပြီး logic တွေကွဲပြားတာတွေရှိနိုင်ပါတယ်။ Apple Class နဲ့ orange Class ရှိတယ်လိုဆိုကြပါစို့ သူတို့ဟာ fruits ဖြစ်ပေမယ့် အရသာတွေမတူညီကြပါဘူး။

```
class Apple
{
   public function taste()
   {
      echo "Apple is sweet";
   }
}
class Orange
{
   public function taste()
   {
      echo "Orange is Sour";
   }
}

function getTaste(Apple $fruit)
{
   $fruit->taste();
}

$apple = new Apple();
$orange = new Orange();

getTaste($apple);
getTaste($orange);
```

အပေါ်က Code အတိုင်းဆို orange Class ကို getTaste function ထဲထည့်ပေးလိုက်တဲ့အခါ Error တတ်ပါလိမ့်မယ်။ Function မှာ Apple ကိုသာ လက်ခံမယ်လို့ရေးသားထားလို့ပါ။ဒီလိုပြဿနာတွေဖြေရှင်းဖို့အတွက် Interface ကို အသုံးပြုနိုင်ပါတယ်။

```
interface Fruit
{
   public function taste();
}

class Apple implements Fruit
{
   public function taste()
   {
      echo "Apple is sweet";
   }
}

class Orange implements Fruit
{
   public function taste()
   {
      echo "Orange is Sour";
   }
}

function getTaste(Fruit $fruit)
{
   $fruit->taste();
}

$apple = new Apple();
$orange = new Orange();

getTaste($apple);
getTaste($orange);
```

Fruit ဆိုတဲ့ interface ကြားခံClass တခုခံပြီး Class တွေမတူညီပေယ့် Function name တူတူထားပြီး ခေါ်သုံးလို့ရနိုင်ပါတယ်။ getTaste function မှာ interface class ကို parameter အနေနဲ့ ခေါ်ထားပြီး တူညီတဲ့ function name ခေါ် ပြီးဖြေရှင်းနိုင်ပါတယ်။ဒီလိုနေရာ မျိုးတွေမှာ interface ကို သုံးပါတယ်။

Abstract Class

Extends classတွေထဲမှာတူညီတဲ့ function nameနဲ့ ကွဲပြားတဲ့ business logic တွေတည်ရှိနေတဲ့အခါ parent class မှာ function name ကို common ထားပြီး Extends Class တွေမှာကိုယ်ပိုင် logic တွေနဲ့ override ပြန်ရေးတဲ့အခါမျိုးမှာ အသုံးပြုပါတယ်။ abstract နဲ့ရေးသားတဲ့အခါ Class မှာ abstract ထည့်ပေးရပြီး၊ Abstract function တခုကြေညာထားရပါတယ်။Extends လုပ်ထားတဲ့ Class တွေမှာတော့ Parent မှာကြေညာထားတဲ့ Abstract Function name နဲ့တူညီတဲ့ Name ပေးပြီး ကိုယ်ပိုင် Business logic တွေကိုရေးသားနိုင်ပါတယ်။

```
abstract class Person
{
   protected $name;
   protected $age;

   public function __construct($name, $age)
   {
      $this->name = $name;
      $this->age = $age;
   }

   abstract public function getInfo();
}

class Employee extends Person
{
   public function getInfo()
   {
      echo "$this->name is a employee and age is $this->age";
   }
}
```

```
class Customer extends Person
{
   public function getInfo()
   {
      echo "$this->name is a customer and age is $this->age";
   }
}

$emp = new Employee("John", 23);
$emp->getInfo();

echo "<br>";

$cus = new Customer("Mary", 45);
$cus->getInfo();

//John is a employee and age is 23
//Mary is a customer and age is 45
```

Abstract VS interface

1. Interface က function ကိုdeclare only ပဲလုပ်လို့ရပါတယ်။ Abstract ကတော့ properties ကို define လုပ်နိုင်သလို function တွေလည်း declare လုပ်နိုင်ပါတယ်။
2. Interface က multiple extends လုပ်နိုင်ပြီး Abstract ကတော့ single class ပဲextends လုပ်နိုင်ပါတယ်။
3. Interface မှာ properties သတ်မှတ်လို့မရနိုင်ပါဘူး။Abstract ကတော့ သတ်မှတ်လို့ရနိုင်ပါတယ်။
4. Interface ကတော့ public function ပဲရနိုင်ပြီး Abstract ကတော့ Acess Modifier တွေအားလုံးသတ်မှတ်နိုင်ပါတယ်။

Anonymous Class

Regular Class လိုမျိုး Class name သေချာမပေးထားပဲ one time use သုံးချင်တဲ့အခါ မျိုးမှာသုံးကြပါတယ်။Regular Class တွေအတိုင်းသုံးနိုင်ပါတယ်။

```
$obj=new class {
  public function sayhello(){
    echo "Hello World";
  }
};
$obj->sayhello();
?>
```

Trait

Php မှာ multiple classကို extend လုပ်ခွင့်ပေးမထားပါဘူး။ဒါကြောင့် classတွေကို extendsလုပ်ချင်တဲ့ အခါ trait ကို သုံးပါတယ်။

```
trait weakup
{
   public function alarm1()
   {
      echo "Werak Up Alarm at 6:00 AM";
   }
}

trait sleep
{
   public function alarm2()
   {
      echo "Sleep Time Alarm at 11:00 PM";
   }
}
```

```
class DailyWork
{
   use weakup, sleep;
}


$day = new DailyWork();
$day->alarm1();
echo "<br>";
$day->alarm2();

//Werak Up Alarm at 6:00 AM
//Sleep Time Alarm at 11:00 PM
```

16) cURL

တခြားserver သို့မဟုတ် same domain ကနေ data တွေကိုရယူတဲ့အခါ cURL ကိုအသုံးပြုပါတယ်။

```
$curl = curl_init();

curl_setopt($curl, CURLOPT_URL, "http://..");
curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);

$response = curl_exec($curl);

echo "<pre>";
print_r($response);
echo "</pre>";
```

17) MySQL

MySQL ဆိုတာကတော့ Database တခုဖြစ်ပါတယ်။Database ဆိုတာကတော့ data တွေကိုသိမ်းထားနိုင်တဲ့ နေရာတခုဖြစ်ပါတယ်။သူ့ထဲမှာတော့ data တွေတည်ရှိပြီး ရှိနေတဲ့ data တွေကို ဆွဲထုတ်နိုင်သလို အသစ်ထည့်တာတွေ၊ data update လုပ်တာတွေ၊ဖျက်သိမ်းတာတွေ လုပ်နိုင်တဲ့ နေရာတခုဖြစ်ပါတယ်။

MySQL လိုမျိုးတခြားdatabase များစွားရှိပါတယ်။

Php မှာ MySQL database နဲ့ ချိတ်ဆက်နိုင်ဖို့build in Class တွေရှိပါတယ်။PDO Class ကတော့database နဲ့ချိတ်ဆက်တဲ့အခါ သုံးရတာလွယ်စေတဲ့အတွက် PDOနဲ့ ချိတ်ဆက်တာများကြပါတယ်။

MySQL Query အကြောင်းကို နောက်သင်ခန်းစာမှာအသေးစိတ်လေ့လာနိုင်ပါတယ်။

Initial Connection

```
$pdo = new PDO('mysql:host=$hostname;port=$port;dbname=$dbname, $username, $password);
```

Error Exception တွေဖမ်းနိုင်ဖို့အတွက်

```
$pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
```

Query တွေကို Execute မလုပ်ခင် prepare

```
$sql = $pdo->prepare($query);
```

Execute

```
$sql->execute();
```

ရလာတဲ့ data တွေကို associated arrayနဲ့ ပြောင်းယူ

```
$result = $sql->fetchAll(PDO::FETCH_ASSOC);
```

SQL Injection တွေ protect လုပ်ဖို့အတွက် **bindValue** ကိုသုံးနိုင်ပါတယ်။

Example

```
$sql = $pdo->prepare("SELECT * FROM product WHERE code=:code");
$sql->bindValue(":code", value );
```

18) Namespace

Class ရဲ့ nameတွေ သို့မဟုတ် class တွေထဲမှာရှိတဲ့ method name တွေတူညီနေပြီး တခြား php file ကနေ required နဲ့ခေါ်သုံးဖို့လိုတဲ့အခါ name တွေတူညီနေတဲ့အတွက် Already use ဆိုတဲ့ error messange တွေပြပါတယ်။

ဒီပြဿနာကို ဖြေရှင်းဖို့အတွက် namespace ကို အသုံးပြုနိုင်ပါတယ်။

ဒီလိုတည်ဆောက်ထားတဲ့ Class တွေအတွက်

Employee.php

```
class payment
{
   public function pay()
   {
      echo "Pay with CB Bank";
   }
}
```

Customer.php

```
class payment
{
   public function pay()
   {
      echo "Pay with Kpay";
   }
}
```

Index.php ကနေ ခေါ်တဲ့အခါ

```
require "Employee.php";
require "Customer.php";

$empPay = new payment();
$empPay->pay();

echo "<br>";

$cusPay = new payment();
$cusPay->pay();
```

Fetal Error တတ်ပါတယ်။

Fatal error: Cannot declare class payment, because the name is already in use in /Applications/XAMPP/xamppfiles/htdocs/Customer.php on line 3

Namespace သုံးပြီးခေါ်တဲ့အခါမှာတော့ Class တွေရဲ့ declaration တွေအပေါ်မှာ

```
namespace empPayment;
class payment
{
   public function pay()
   {
      echo "Pay with CB Bank";
   }
}
namespace cusPayment;
class payment
{
   public function pay()
   {
      echo "Pay with Kpay";
   }
}
```

Index.php မှာ namespaceနဲ့သတ်မှတ်ထားတဲ့ Classကိုခေါ်တဲ့အခါ \ ခံပြီးခေါ်သုံးရပါတယ်။

```
require "Employee.php";
require "Customer.php";

$empPay = new empPayment\payment();
$empPay->pay();
echo "<br>";
$cusPay = new cusPayment\payment();
$cusPay->pay();
```

Browser result မှာတော့သတ်မှတ်ထားတဲ့ အတိုင်းပေါ်လာမှာဖြစ်ပါတယ်။

```
//Pay with CB Bank
//Pay with Kpay
```