PHP

for Web Developers

NATIONAL CYBER CITY
Nay Win Hlaing

First Edition

မာတိကာ

အခန်း(၁) - PHP ဆိုတာဘာလဲ

PHP ဆိုတာ server-side scripting language တစ်ခုဖြစ်ပါတယ်။ PHP ကို Web Applications တွေ Static Websites တွေ Dynamic Websites တွေ develop လုပ်ရာမှာ အသုံးပြုပါတယ်။ PHP က PHP: HyperText Preprocessor ဖြစ်ပြီးတော့ အရင်တုန်းက “Personal Home Pages” လို့ လည်း ခေါ်ပါတယ်။

Scripting language ဆိုတာဘာလဲ

Scripting Language ဆိုတာ ဘာလဲမပြောခင် scripts တွေ အလုပ်လုပ်ပုံကို ပြောပြပါ့မယ်။ script တစ်ခုဆိုတာ က program တစ်ခုရဲ့ runtime မှာ အလုပ်လုပ်ပေးတဲ့ a set of programming instructions တစ်ခုဖြစ်ပါတယ်။ ဒါကြောင့် Scripting Launguage ဆိုတာ program တစ်ခုရဲ့ runtime မှာ အလုပ်လုပ်ပေးတဲ့ scripts တွေ ဖြစ်ပါတယ်။ Scripts တွေကို များသောအားဖြင့် Software

environments တွေမှာ မြှုပ်ရေးလေ့ရှိကြပါတယ်။ Scripts တွေကို အသုံးပြုတဲ့ ရည်ရွယ်ချက်က application တွေရဲ့ performance တွေကို မြင့်မားဖို့ ပြီးတော့ tasks တွေကို routine လုပ်ဖို့အတွက်ပါ။ routine ဆိုတာ function သို့မဟုတ် procedure တစ်ခု ဒါမှမဟုတ် subprogram တစ်ခုကိုဆိုလိုတာပါ။ သူ့ကို program တစ်ခု ရဲ့ လိုအပ်တဲ့နေရာတိုင်းမှာ called လုပ်ပြီးတော့ အသုံးပြုပါတယ်။ ဥပမာ အချိန်တွေပြေပေးတဲ့ လုပ်ဆောင်မှု တစ်ခုကို နေရာတိုင်းမှာ လိုက်ပြီး မရေးတော့ပဲ နေရာတိုင်းမှာ အသုံးပြုလို့ ရအောင် tasks ကို လိုအပ်တဲ့အချိန်တိုင်းမှာ အသုံးပြုလို့ရအောင် rotine လုပ်ခြင်းဖြစ်ပါတယ်။ Client Application က နေ client side scripts တွေ အလုပ်လုပ်တဲ့အချိန်မှာ server side scripts ပြန်ပြီး အလုပ်လုပ်ပေးပါတယ်။ ဥပမာ client side က နေ data or something တစ်ခုခု request လုပ်တာမျိုးကို server side ကနေ respond ပြန်တာမျိုးပါ။ PHP က server side script ဖြစ်တာကြောင့် client browser ကနေလာတဲ့ client side script ဥပမာ JavaScript ကနေ အလုပ်လုပ်တဲ့အချိန်မှာ server side script ဖြစ်တဲ့ PHP က အလုပ်ပြန်လုပ်ပေးမှာဖြစ်ပါတယ်။

Programming Language နဲ့ Scripting Language ခြားနားချက်

- Programming Language တစ်ခုမှာ application တစ်ခုလုံးကို complete ဖြစ်အောင် develop လုပ်ဖို့အတွက် လိုအပ်တဲ့ features တွေအကုန်လုံးပါပါတယ်။ Scripting Language ကိုကျတော့ tasks routine တွေ အတွက် အသုံးပြုပါတယ်။
- Programming Language တစ်ခုမှာ Program တစ်ခု ပြီးအောင် လုပ်ဖို့ အရင်ဆုံး compile လုပ်ပြီးမှ execute ဖြစ်ပါတယ်။ Scripting Language မှာကျ codes တွေက များသောအားဖြင့် complie မလုပ်ဘဲ execute ဖြစ်ပါတယ်။
- Programming Language တစ်ခုမှာ အလုပ်လုပ်ဖို့ အခြားသော programming language တွေခြားထဲ မြှုပ်ရေးစရာမလိုပါဘူး။ Scripting Language ကိုတော့ အခြားသော software environments တွေမှာ မြှုပ် ရေးရပါတယ်။

PHP လေ့လာဖို့ သိထားသင့်တဲ့ Technologies များ

Php files တစ်ခုမှာဆိုရင် များသောအားဖြင့် HTML tags တွေ client side scripts အဖြစ် အသုံးများတဲ့ JavaScript တွေပါပါတယ်။

ဒါကြောင့် PHP language ကို လေ့လာဖို့အတွက် HTML ကို မသိမဖြစ် သိထားရပါမယ်။ HTML ကို သေခြာမသိဘဲ PHP ကိုလေ့လာလို့ရပေမဲ့ HTML ကိုသိတဲ့အခါ လေ့လာတဲ့အခါမှာ များစွာ အထောက်ကူပြုပါတယ်။

နောက်ပြီး applications တွေ ပိုမို powerful ဖြစ်အောင် Database Management System (DBMS) ကိုလည်း သိထားရပါမယ်။

Applications တွေ မှာ web services တွေ ပိုပြီး interactive ဖြစ်အောင် dynamic ဖြစ်အောင် JavaScript ကိုလည်း လေ့လာထားဖို့လိုပါတယ်။

PHP ကို ဘာကြောင့်လေ့လာသင့်သလဲ?

အခြားသော programming languages တွေရှိတဲ့ထဲကမှ Web Developing အတွက် PHP ကိုလေ့လာသင့်တာ ဒီအချက်တွေကြောင့်ပါ။

- PHP က open-source ဖြစ်ပြီးတော့ free အသုံးပြုလို့ရပါတယ်။
- PHP က လေ့လာရလွယ်ကူပြီး short learning curve ဖြစ်ပါတယ်။
- Web Hosting အားလုံးနီးပါက PHP ကို suppose ပေးပါတယ်။ ဒါကြောင့်လည်း PHP ကရွေးခြယ်စရာ Language တစ်ခုဖြစ်ပါတယ်။
- PHP က အခြားသော latest technology trends တွေနဲ့အပြိုင် ပုံမှန် up to date ဖြစ်ပါတယ်။
- PHP က server-side scripting language ဖြစ်တာကြောင့် server ဖက်မှာပဲ PHP ကို install လုပ်ဖို့လိုပြီး request လုပ်မဲ့ client-side ဖက်မှာ ထပ်ပြီး install လုပ်ဖို့မလိုဘဲ web browser တစ်ခုရှိဖို့ပဲ လိုပါတယ်။

- MySQL နဲ့ တွဲပြီး develop လုပ်ဖို့အတွက် built-in support တွေများစွာပါပါတယ်။ MySQL အတွက် support တွေများစွာပါတယ် ဆိုပေမဲ့ အခြားသော DBMS တွေဖြစ်တဲ့ Postgres, Oracle စတဲ့ DBMS တွေနဲ့ လည်း develop လုပ်လို့ရပါတယ်။

- PHP က cross-platform ဖြစ်ပါတယ်။ ဆိုလိုတာက applications တွေကို Linux, Windows, Mac OS စတဲ့ ကိုယ်နှစ်သက်တဲ့ Operating Systems ပေါ်မှာ အသုံးပြုလို့ရပါတယ်။

PHP ကို ဘယ်တွေမှာ အသုံးပြုသလဲ?

Internet ပေါ်မှာ ရှိတဲ့ Websites တွေအားလုံးရဲ့ 78% မှာ PHP ကိုအသုံးပြုနေကြပါတယ်။ ဆိုလိုတာက internet ပေါ်မှာ ရှိတဲ့ Websites 10 ခုမှာ 8 ခုက PHP ကို အသုံးပြုပါတယ်။
PHP ကို အသုံးပြုထားတဲ့ Websites တွေဖြစ်တဲ့ Social Sites တွေမှာ ဆိုရင် Facebook ဖြစ်ပါတယ်။ Facebook က သူတို့ ကိုယ်တိုင် create လုပ်ထားတဲ့ PHP(HHVM) ကိုအသုံးပြုပါတယ်။
CMS Blogs Sites တွေမှာ ဆိုရင် နာမည်ကြီးတဲ့ Wordpress မှာလည်း PHP ကို အသုံးပြုပါတယ်။
နောက်ပြီး Designs တွေဖန်တီးရာမှာ နာမည်ကြီးတဲ့ Canva မှာလည်း PHP ကိုအသုံးပြုပါတယ်။
Email Marketings Site ဖြစ်တဲ့ MailChimp မှာလည်း PHP ကိုအသုံးပြုပါတယ်။

PHP File Extensions နဲ့ PHP tag

File extension နဲ့ tags တွေက server ကို ရေးလိုက်တဲ့ file က PHP ဖြစ်ကြောင်းကို identify လုပ်ပေးပါတယ်။ PHP file တစ်ခုကို save တဲ့အခါ ".php" extension နဲ့ save ရပါမယ်။ အရင်တုန်းက old version တွေမှာဆိုရင်

- .phtml
- .php3
- .php4

- .php5
- .phps

ဆိုပြီး save ကြပါတယ်။ PHP က HTML နဲ့ တွဲပြီး အလုပ်လုပ်တာဖြစ်တဲ့အတွက် PHP codes တွေကို HTML tags တွေကြားမှာ ထည့်ရေးပါတယ်။

```
<HTML> <PHP CODE> </HTML>
```

HTML codes တွေမပါဘဲနဲ့ PHP codes only ရေးလို့ရပါတယ်။ အဲ့ဒီလို file ကို PHP pure file လို့ ခေါ်ပါတယ်။ Server က PHP codes တွေကို ဖတ်ပြီးတဲ့အခါမှာ result အဖြစ် web browser ပေါ်မှာ HTML code တွေနဲ့ output ထုတ်ပေးပါတယ်။ HTML tags တွေကြားမှာ ရှိတဲ့ PHP codes တွေကို server က သိဖို့ဆိုရင် PHP codes တွေကို PHP tag အတွင်းမှာသေခြာရေးဖို့လိုပါတယ်။ PHP tag ကဒီလိုပါ။

```
<?PHP . . . ?>
```

PHP က case-sensitive ဖြစ်တဲ့ Language ဖြစ်ပါတယ်။ သဘောက PHP မှာ “VAR” နဲ့ “var” နဲ့မတူပါဘူး။ PHP က သူကိုယ်တိုင်က case-sensitive မဖြစ်ပေမဲ့ အသေးနဲ့ရေးတာ ပိုကောင်းပါတယ်။ နောက်ပြီး မှတ်စရာ တစ်ခုက PHP tags အတွင်းမှာ ရေးတဲ့ statement တစ်ကြောင်းပြီးတိုင်း “;” ခံရပါတယ်။ statement တစ်ကြောင်းထဲဆို “;” မခံရပေမဲ့ တစ်ကြောင်းပြီးတိုင်း “;” ခံတာပိုကောင်းပါတယ်။ ဒါကြောင့် PHP scripts က server ပေါ်မှာ execute ဖြစ်ပြီးတော့ HTML အနေနဲ့ output ပြပါတယ်။

အခန်း(၂) - PHP Data Types and Variables

PHP Data Types

- PHP data types တွေကို အမျိုးစား ခွဲကြည့်မယ်ဆိုရင် alphanumeric characters ကို strings data type လို့ခေါ်ပါတယ်။
- numbers တွေကို အကုန်လုံးကို integer data typeလို့ခေါ်ပါတယ်။
- decimal points တွေ ပါတဲ့ numbers တွေကို double data type လို့ခေါ်ပါတယ်။
- True or false values တွေကို တော့ boolean data type လို့ခေါ်ပါတယ်။

PHP က loosely type language ဖြစ်ပါတယ်။ ဆိုလိုတာက data type တစ်ခု အသုံးပြုတဲ့အခါ data type က ဘာဖြစ်ပါတယ်ဆိုတာ ကြေငြာ သတ်မှတ်ပေးစရာမလိုပါဘူး။ PHP က variable ရဲ့ value ကို ကြည့်ပြီးတော့ ဘာ data type ဖြစ်ကြောင်း auto သတ်မှတ်ပါတယ်။

PHP Variable

Variable ဆိုတာ memory ထဲမှာ data ကို store လုပ်ဖို့ သတ်မှတ်ထားတဲ့ name တစ်ခုဖြစ်ပါတယ်။ PHP မှာ ဆိုရင် variable type နှစ်မျိုးရှိပါတယ်။ global variable နဲ့ local variable ဖြစ်ပါတယ်။ global variable ကို application ရဲ့ scripts တွေ အကုန်လုံးကနေ အသုံးပြုလို့ရပါတယ်။ local variable ကို ကြတော့ သူ့ကို သတ်မှတ် ကြေညာ ထားတဲ့ script အတွင်းမှာ ပဲ အသုံးပြုလို့ရပါတယ်။ နောက်ပိုင်းမှာ ပိုရှင်း ပြပါ့မယ်။

Variable တစ်ခုသတ်မှတ်မယ်ဆိုရင် အရင်ဆုံး $ ပြီးမှာ variable name လိုက်ပါတယ်။ နောက်ပြီး value ကို '=' နောက် ရေးပါတယ်။

```
$variable_name = value;
```

အပေါ်မှာ ဖော်ပြခဲ့တဲ့အတိုင်း PHP က case-sensitive ဖြစ်ပါတယ်။ $name နဲ့ $NAME နဲ့ မတူပါဘူး ။ variable နှစ်ခုဖြစ်ပါတယ်။
Variable တစ်ခုတည်ဆောက်တော့မည်ဆိုရင် သိထားသင့်တာလေးတွေရှိပါတယ် ။
variable တွေကို သတ်မှတ်တဲ့အခါ $ နောက်မှာ letter ကနေစရပါမယ်။ number တွေနဲ့စလို့မရပါဘူး။
ဥပမာ -

```
$name = “mg mg” // variable

$1name = “mg mg” // not a variable
```

နောက်ပြီး variable သတ်မှတ်တဲ့အခါ space “ ” ပါလို့မရပါဘူး။ space အစား (_) underscore နဲ့ သတ်မှတ်လို့ရပါတယ်။
ဥပမာ -

```
$first name = “John”// not a variable
```

```
$first_name = “John” // a variable
```

Echo Statement

data တွေကို screen ပေါ်မှာ output အနေနဲ့ ပြချင်တယ်ဆိုရင် echo statement ကို အသုံးပြုပါတယ်။

```
<?php
$my_var = “Hello World”;
echo $my_var; // “Hello World”
?>
```

ဒီမှာဆိုရင် my_var ဆိုတဲ့ variable လေးထဲကို “Hello World” ဆိုတဲ့ string လေးကို store လုပ်ထားပါတယ်။ အဲ့ဒီ variable ကို output ထုတ်ချင်တဲ့အခါ echo statement ကို သုံးပြီးတော့ output ထုတ်ထားပါတယ်။ ဒါကြောင့် screen ပေါ်မှာ Hello World ဆိုတဲ့ output လေးမြင်ရမှာ ဖြစ်ပါတယ်။ ဒီမှာ မှတ်ထား ရမှာ variable တစ်ခု ကြေညာတဲ့အခါ value assign ပါ တစ်ခါတည်းထည့်ပေးရပါတယ်။ အကယ်၍ value assign မရှိတဲ့ variable ကို output ထုတ်တဲ့အခါ PHP Notice: Undefined variable: ဆိုပြီး error တက်ပါလိမ့်မယ်။

Comment

Comment ကို အသုံးမပြုခင် comment တွေရဲ့ အရေးကြီးပုံကို ပြောပြချင်ပါတယ်။ ဥပမာဆိုပါစို့ တို့က company တစ်ခုခုမှာ အလုပ်လုပ်နေတယ်ဆိုပါစို့ ။ company မှာဆိုရင် teams တွေနဲ့ အလုပ်လုပ်ရတယ်။ teams ထဲမှာဆိုရင် အခြား programmers တွေလည်းပါမယ်။ project

တစ်ခုခုရေးတော့မယ်ဆိုရင် team အလိုက်ရေးရတယ်။ ရေးတဲ့ အချိန်မှာ function တွေ logics တွေအကြောင်းကို ဘယ် function က ဘာလုပ်တယ် ဒီ logics ကို ဘယ်လိုဆိုတာလေးကို comments တွေ ခံခဲ့မယ်။ အဲ့ဒီလို comments တွေ ခံခဲ့တဲ့အခါ codes ကို team members တွေဖြစ်တဲ့ အခြား programmers အတွက် များစွာအထောက်ကူပြုတယ်။ ရေးတဲ့ code တွေကို comment လေးတွေခံခဲ့အတွက် သူတို့လည်း အလွယ်တကူနားလည်စေတယ်။

နောက်တစ်ချက်က ရေးထားတဲ့ codes တွေကို နောက်ကြမှ ပြန်ကြည့်တဲအခါ ဘယ် function က ဘာအလုပ်လုပ်တယ်ဆိုတာကို comment မခံခဲ့ရင် အဲ့ဒီ function ကဘာအလုပ်လုပ်တယ်ဆိုတာကို ပြန်တွေး ပြန်စမ်းတဲ့အခါ အချိန်တွေကုန်ပါတယ်။ ဒါကြောင့် codes တွေရေးပြီးတိုင်မှာ ဘာ function က ဘာအလုပ်လုပ်တယ် ဒီ logic က ဘယ်လိုဆိုတာကို comment ခံခဲ့ဖို့ အကြံပြုချင်ပါတယ်။

PHP Data Types

အထက်မှာ PHP data types တွေအကြောင်း အကြမ်းဖျဉ်း ဖော်ပြထားပါတယ်။ ဒီမှာ တစ်ခုချင်းဆီ details ပြန် ရှင်းပါမယ်။ PHP က data types အားဖြင့် primitive data types (၈) မျိုးရှိပါတယ်။ Integer, Floating point number or Float, String, Booleans, Array, Object, resource နဲ့ NULL တို့ပဲဖြစ်ပါတယ်။

PHP Integers

Decimal points တွေမပါတဲ့ numbers တွေ အားလုံးကို ဥပမာ(...,-2,-1 , 0 , 1 , 2 ,...) တွေ အကုန်လုံးကို ကို integers လို့ခေါ်ပါတယ်။ integers တွေကို ပြန် ခွဲကြည့်မယ်ဆိုရင် သူ့မှာ decimal (base 10) , hexadecimal (base 16 - prefixed with 0x) or octal (base 8 - prefixed with 0) notation နဲ့ + , - sign တွေပါတဲ့ numbers တွေ ပါပါတယ်။
Example တွေနဲ့ ကြည့် ရအောင် ။

```
<?php
        $a = 123; // decimal number
        var_dump($a);
        echo "<br>";
        $b = -123; // a negative number
        var_dump($b);
        echo "<br>";
        $c = 0x1A; // hexadecimal number
        var_dump($c);
        echo "<br>";
        $d = 0123; // octal number
        var_dump($d);
?>
//output
int(123)
int(-123)
int(26)
int(83)
```

ဒီမှာဆိုရင် output ထုတ်ဖို့ အတွက် PHP built-in function ဖြစ်တဲ့ var_dump() ကို သုံးထားပါတယ်။ var_dump() က output တွေကို array data ပုံစံနဲ့ ပြပြီးတော့ ဘာ data type ဖြစ်ကြောင်း length ဖြစ်ကြောင်း ကို ထပ်ပြီး ပြပေးပါတယ်။ first variable ဖြစ်တဲ့ $a ကို ထုတ်တဲ့အခါ int(123) ဆိုပြီး ထွက်ပါလိမ့်မယ်။ second variable ဖြစ်တဲ့ $b က negative sign variable ဖြစ်ပြီး integer ဖြစ်တဲ့ အတွက် int(-123) ဆိုပြီး ရပါတယ်။ thrid variable မှာ မှတ်စရာ တစ်ခု ရှိပါတယ်။ PHP က numbers တွေကို အကုန်လုံးကို decimal အဖြစ် ပြောင်းပါတယ်။ ဒါကြောင့် hexadecimal number ကို decimal

value အဖြစ် int(26) ဆိုပြီး ရပါတယ်။ fourth variable မှာ octal number က လည်း integer ဖြစ်တဲ့ အတွက် int(83) ဆိုပြီးတော့ ရပါတယ်။

PHP version 5.4 အထက်မှာဆိုရင် integers in binary (base 2) notation ကိုပါ support ပေးပါတယ်။ ဥပမာ ($e = 011110111;) ။

နောက်ထပ် Integer Data type နဲ့ပတ်သတ်ပြီး သိသင့်တာလေးတွေရှိပါတယ်။ Integer

PHP Floating Points Numbers or Doubles

Float ponts numbers တွေကို float , double or real numbers တွေဖြစ်တဲ့ decimal points တွေပါတဲ့ number တွေ fractional numbers တွေ exponential form တွေ ပါပါတယ်။

```
<?php
        $a = 1.234;
        var_dump($a);
        echo "<br>";
        $b = 10.2e3;
        var_dump($b);
        echo "<br>";
?>
float(1.234)
float(10200)PHP Strings
```

Letters တွေ numbers တွေ characters တွေ ပေါင်းထားတဲ့ စာစဉ် လေးကို string လို့ ခေါ်ပါတယ်။ String ကို quote mark (‘ ’) တွေကြားထည့် ပြီး create လုပ် ပါတယ်။

```
        $my_string = 'Hello World';
```

Single quote (‘’) ထဲအပြင် double quote(“ ”) နဲ့ လည်း create လုပ်လို့ ရပါတယ်။ ဒါ ပေမဲ့ single quote နဲ့ double quote တွေက အလုပ်လုပ်ပုံမတူပါဘူး။

```
<?PHP
        $my_str = 'World';
        echo "Hello, $my_str!<br>";  // Displays: Hello World!
        echo 'Hello, $my_str!<br>';  // Displays:Hello, $my_str!
?>
```

ဒီမှာဆို ပထမတစ်ကြောင်း ကို double quote (“”) နဲ့ output ထုတ်တဲ့ အခါ variable $my_str က သူရဲ့ value အနေနဲ့ output ထွက်လာပါတယ်။ ဒုတိယ တစ်ကြောင်း single quote(‘’) နဲ့ output ထုတ်တဲ့ အခါ string အတိုင်းပဲ output ထွက်လာပါတယ်။ $my_str ကို variable အနေနဲ့ မသိပဲ string အနေနဲ့ပဲ သိပါတယ်။

Escape-sequence replacements

PHP strings ရေးတဲ့အခါ အထောက်ကူပြုတဲ့ escape-sequence တွေရှိပါတယ်။ သူတို့ကို အရှေ့မှာ ‘\’ ခံပြီး ရေးပါတယ်။

- \n ကို နောက်တစ်ကြောင်း ဆင်းချင်တဲ့အခါ အသုံးပြုပါတယ်။
- \r ကို Carriage Return ပေးချင်တဲ့အခါ သုံးပါတယ်။ \n နဲ့ဆင်ပါတယ်။
- \t ကို tab character ထည့်ချင်တဲ့အခါ အသုံးပြုပါတယ်။
- \$ ကို $ (dollar sign) ထည့်ချင်တဲ့အခါ အသုံးပြုပါတယ်။
- \" ကို single double-quote (") ထည့်ချင်တဲ့အခါ အသုံးပြုပါတယ်။
- \\ ကို single backslash (\) ထည့်ချင်တဲ့အခါ အသုံးပြုပါတယ်။

```
<?php
        echo '<pre>Hello\t World!</pre>';
```

```
        echo "<pre>Hello\t World!</pre>";
        echo "<pre>Hello\r World!</pre>";
        echo "<pre>Hello\$World!</pre>";
        echo 'I\'ll be back';
?>
//output
Hello\tWorld!
Hello   World!
Hello
World!

Hello$World!
I'll be back
```

ဒီမှာဆိုရင်လည်း မှတ်စရာလေး ရှိပါတယ်။ escape-sequence replacements တွေက single quote(‘ ’) နဲ့ ရေးတဲ့အခါ အလုပ် မလုပ်ပါဘူး။ ဒါပေမဲ့ single quote ထဲမှာ single quote(‘ ’) escape-sequence replacement က အလုပ်လုပ်ပါတယ်။

Manipulating PHP Strings

Strings ကို ကျွမ်းကျွမ်းကျင်ကျင် အသုံးပြုဖို့အတွက် PHP က built-in function ပေါင်း များစွာ support ပေးပါတယ်။ ဥပမာ string ရဲ့ length ကို တွက်ထုတ်တာ ၊ string ထဲက မှာ substring ကို ရှာတာ၊ ဒါမဟုတ် character တစ်ခုခုကိုရှာတာ ၊ substring တစ်ခု သို့မဟုတ် character တစ်ခြား တစ်ခုနဲ့ replace လုပ်တာ စတဲ့ အလုပ်လုပ်တွေ လုပ်ရပါတယ်။ အဲ့ဒီ ထဲက အသုံးများတဲ့ functions တွေ ကို ဖော်ပြပါမယ်။

strlen() - string ရဲ့ length ကို တွက်ချင်တဲ့အခါ သုံးပါတယ်။

```
<?php
	$my_str = 'Learn about PHP for Web Developers';

	// Outputs: 34
	echo strlen($my_str);
?>
```

ဒီမှာဆို string ရဲ့ အလုံး အရေတွက်ကို ထုတ်ပေးပါလိမ့်မယ်။

str_word_count() - string မှာ ရှိတဲ့ words အရေတွက်ကို သိချင်တဲ့ အခါ သုံးပါတယ်။

```
<?php
	$my_str = 'The quick brown fox jumps over the lazy dog.';

	// Outputs: 9
	echo str_word_count($my_str);
?>
```

ဒီမှာဆို string မှာ ရှိတဲ့ words အရေအတွက်ကို ထုတ်ပေးပါလိမ့်မယ်။

str_replace() - string မှာ ရှိတဲ့ ကိုပြောင်းချင်တဲ့ စာကို အခြားတစ်ခုနဲ့ ပြောင်းချင်တဲ့ အခါ နောက်ပြီး ပြောင်းလိုက်တဲ့အရေအတွက်ကို သိချင်တဲ့အခါ အသုံးပြုပါတယ်။

```
<?php
	$my_str = 'If the facts do not fit the theory, change the facts.';

	// Display replaced string
```

```
        echo str_replace("facts", "truth", $my_str);
?>
```

ဒီမှာဆို str_replace() က arguments သုံးခု တောင်းပါတယ်။ ပထမ တစ်ခုက string ထဲက ကိုပြောင်းချင်တဲ့ substring or word ။ ဒုတိယ တစ်ခုက ကို အစားထိုး replace လုပ်ချင်တဲ့ substring or word ။ တတိယ တစ်ခုက ကိုပြောင်းချင်တဲ့ string ဖြစ်ပါတယ်။ ဒါကြောင့် output အနေနဲ့ facts တွေ နေရာမှာ truth တွေ replace ဖြစ်ပြီးတော့ “If the truth does not fit the theory, change the truth.” output အဖြစ်ထွက်လာပါတယ်။

```
<?php
        $my_str = 'If the facts do not fit the theory, change the facts.';

        // Perform string replacement
        str_replace("facts", "truth", $my_str, $count);

        // Display number of replacements performed
        echo "The text was replaced $count times.";
?>
```

ဒီမှာဆိုရင် str_replace() က argument လေးခု ယူထားပါတယ်။ နောက်ဆုံး argument က string ထဲက ဘယ် နှစ်ကြိမ် replace လုပ်လဲဆိုတာ ကို return ပြန်ပေးပါတယ်။ ဒါကြောင့် $count ကို output ထုတ်ကြည့်တဲ့အခါ replace နှစ်ခါ လုပ်ခဲ့ရတာ ဖြစ်တဲ့အတွက် 2 ဆိုပြီး ရပါလိမ့်မယ်။
strpos() – string မှာ word or letter ရဲ့ postion ကို သိချင်တဲ့အခါ အသုံးပြုပါတယ်။

```
<?php
        echo strpos("Hello world!", "world"); // outputs 6
?>
```

နောက်ပြီး PHP 7 မှာ ဆိုရင် string function တွေ များစွာ ထပ်ပြီး ရှိလာပါတယ်။ အဲ့ထဲက အသုံးများတဲ့ functions တွေကို ဖော်ပြပါ့မယ်။

addslashes() - string ထဲက အထူးဖော်ပြထားတဲ့ စကားလုံးတွေမှာ “\” ထည့်တာဖြစ်ပါတယ်။

```
<?php
        $str = addslashes('What does "Mental" mean?');

        //output : What does \"Mental\" mean?
        echo($str);
?>
```

ဒီမှာဆို double quote(“”) တွေရှေ့မှာ “\” တွေ ထပ်ထည့်တာတွေရပါလိမ့်မယ်။

htmlspecialchars() - string ထဲမှာ ရှိတဲ့ special characters တွေကို html entities တွေ အဖြစ် ပြောင်းလဲပေးပါတယ်။

```
<?php
        $str = htmlspecialchars(“<script>alert(2)</script>”);

        //output : <script>alert(2)</script>
        echo($str);
?>
```

ဒီတိုင်းဆိုရင် ပုံမှန် 2 ဆိုပြီး alert box နဲ့ ပေါ်လာတက်ပါတယ်။ htmlspecialchars() က special characters တွေဖြစ်တဲ့ < or ( ) or > တို့ကို HTML entities တွေပြောင်းပစ်တာကြောင့် JavaScript code က အလုပ်မလုပ်တော့တာဖြစ်ပါတယ်။ htmlspecialchars() ကို search box တွေ input box တွေမှာ ဖြစ်နိုင်တဲ့ problem တွေကို prevent လုပ်ဖို့အတွက် အသုံးများပါတယ်။

PHP Booleans

Booleans တွေ့ switch နဲ့ တူပြီးတော့ သူ့မှာ value နစ်ခု ပဲ ရှိပါတယ် ။ 1 (true) or 0 (false) ဖြစ်ပါတယ်။ သူ့ကို conditional statement တွေမှာ အသုံးပြုများပါတယ်။

```
<?php
        $show_error = true;
        var_dump($show_error);
?>
```

ဒီမှာဆိုရင် variable ထဲကို boolean value true ကို assign လုပ်လိုက်ပါတယ်။ ဒါကြောင့် output က bool(true) ဆိုပြီးတော့ ထွက်လာပါလိမ့်မယ်။

PHP NULL

NULL က special data type ဖြစ်ပြီးတော့ သူ့မှာ NULL ဆိုတဲ့ value တစ်ခုပဲရှိပါတယ်။ variable တစ်ခု မှာ value assign မရှိဘူးဆိုရင် အဲ့ဒီ variable က NULL data type ဖြစ်ပါတယ်။ ဒါကြောင့် variable တစ်ခု declare လုပ်ပြီး value assign မရှိရင် အဲ့ဒီ variable NULL value ရှိနေမှာဖြစ်ပါတယ်။

```
<?php
        $a = NULL;
        var_dump($a);
        echo "<br>";

        $b;
        var_dump($b);
?>
```

ဒီမှာဆိုရင် output က နှစ်ခုလုံး NULL ထွက် နေပါလိမ့်မယ်။ value assign မရှိတဲ့ variable $b မှာ NULL value ဝင်နေပါလိမ့်မယ်။

PHP Arrays

Arrays ရဲ့ အရေးကြီးပုံလေး ပြောပြချင်ပါတယ်။ programming langauge တော်တော်များများမှာ array ကအရေးကြီးတဲ့ data type တစ်မျိုးဖြစ်ပါတယ်။ arrays နဲ့ သက်ဆိုင်တဲ့ function များစွာရှိပါတယ်။ နောက်ပြီး PHP developer တိုင်းက array function တွေကို အသုံးပြုပြီးတော့ code တွေကို readable ဖြစ်အောင် short ဖြစ်အောင် ဘယ်လိုရေးရမယ်ဆိုတာ သိထားသင့်ပါတယ်။ ဥပမာဆိုပါစို့ ကျောင်းသားတွေရဲ့ အမှတ် စာရင်းတွေရှိတယ်ဆိုပါစို့။ အဲ့ဒီအမှတ်စရင်းတွေ data အချက်အလက်တွေကို api က နေယူမယ်။ api အကြောင်းကို နောက်ပိုင်းမှာပြောပြမယ်။ api ကနေရလာတဲ့ JSON data ကို ပြန် decode လုပ် array ပြောင်း ။ ပြောင်းပြီးတော့ အောင်တဲ့သူက ဘယ်လောက်။ ကျတဲ့သူက ဘယ်လောက်။ ပြီးတော့ ဂုဏ်ထူးထွက်တဲ့သူကဘယ်လောက်။ အဲ့ဒီ ဂုဏ်ထူးထွက်တဲ့သူထဲကမှ မြန်မာစာ ဂုဏ်ထူးထွက်တဲ့သူက ဘယ်လောက်ပေါ့။ အောင်တဲ့သူတွေ ဂုဏ်ထူးထွက်တဲ့သူတွေကို array function တွေနဲ့ ပြန်ခွဲပြီး တွက်ထုတ်နိုင်ပါတယ်။

arrays က PHP မှာ data structure တစ်ခုဖြစ်ပြီးတော့ Multiple elements တွေကို variable တစ်ခု array တစ်ခုထဲမှာ သိမ်းထားနိုင်ပါတယ်။ ဥပမာဆိုပါစို့ number ၅ ခု တစ်ဆောက်ချင်တယ်ဆိုပါစို့။ ဒါဆို variable ၅ ခု တည်ဆောက်ရပါမယ်။ array နဲ့ ကျ number ၅ ခုလုံးကို array variable တစ်ခုထဲမှာ သိမ်းထားနိုင်ပါတယ်။ အဲ့ဒီလို တူညီတဲ့ data type မှမဟုတ်ဘူး မတူညီတဲ့ data type တွေကိုလည်း တစ်စုတစ်စီးထဲ သိမ်းထားနိုင်ပါတယ်။

Array က data values တွေကို index လေးတွေနဲ့ သိမ်းပါတယ်။ index ဆိုတာ number or name လေးတွေဖြစ်ပြီးတော့ array elements တွေရဲ့ အခန်းနံပါတ် သို့မဟုတ်position တွေဖြစ်ပါတယ်။ array တစ်ခုကိုတည်ဆောက်တဲ့အခါ များသော အားဖြစ် array() ဆိုတဲ့ function နဲ့တည်ဆောက်လို့ ရသလို [ ] (“Square brackets or simply brackets”) နဲ့လည်းတည်ဆောက်လို့ရပါတယ်။

Types of arrays

PHP မှာ array type သုံးမျိုး ရှိပါတယ်။ indexed array, associative array နဲ့ multidimensional array တွေဖြစ်ပါတယ်။

Indexed Arrays

Array ရဲ့ elements တွေကို numeric index တွေနဲ့ store လုပ်တဲ့ array ကို index array လို့ခေါ်ပါတယ်။

```
<?php
        $colors = array(“Red” , “Green” , “Blue”);
        print_r($colors);
?>
```

ဒီမှာ ဆိုရင် colors ဆိုတဲ့ array variable တစ်ခုတည်ဆောက်လိုက်ပါတယ်။ array ထဲမှာ element or values တစ်ခုထက်ပိုချင်တာဖြစ်တဲ့အတွက် values တွေကို “,” နဲ့ ခြားပြီး ထည့်ရပါတယ်။ other data types တွေကို echo statement သုံးပြီး output ထုတ်ပြလို့ရပါတယ်။ array variable တစ်ခုကို echo နဲ့ output ထုတ်တဲ့အခါ မှာ PHP Notice: Array to string conversion ဆိုပြီး error တက်ပါတယ်။ ဒါကြောင့် array တစ်ခုလုံး ကို output ပြချင်တဲ့အခါ var_dump() or print_r() function ကိုအသုံးပြုကြပါတယ်။

```
print_r( var_name , return_output) ;
```

print_r() က array variable ကို array format ဖြစ်တဲ့ key-value အတွဲနဲ့ output ထုတ်ပေးပါတယ်။ print_r() က parameter နှစ်ခုရှိပါတယ်။ ပထမက ကိုထုတ်ချင်တဲ့ array name ပါ ။ ဒုတိယ တစ်ခုက

optional parameters ပါ သူ့ကို boolean value ပေးရပါတယ်။ default အနေနဲ့ false ဖြစ်ပါတယ်။ များသော အားဖြင့် parameter တစ်ခုနဲ့ ပဲ အသုံးများပါတယ်။
အပေါ်က example ကို ပြန်ဆက်ရမယ်ဆိုရင် colors ဆိုတဲ့ array variable ကို print_r() function သုံးပြီး output ထုတ်ထားတဲ့ဖြစ်တာကြောင့် key-value အတွဲပုံစံနဲ့ output ပုံစံရပါတယ်။

```
Array ( [0] => Red [1] => Green [2] => Blue )
```

ဒီလိုလေးပါ ဒီမှာဆိုရင် “Red” ကို 0 ဆိုတဲ့ numeric index number နဲ့ တွဲသိမ်းထားတာဖြစ်ပါတယ်။ numeric array or indexed array မှာဆိုရင် index က 0 based ဖြစ်တဲ့ကြောင့် array ရဲ့ ပထမဆုံး element ကို 0 ကနေစပါတယ်။ ဒါကြောင့် array elements ငါးခု ရှိရင် index က 0 to 4 ဖြစ်ပါတယ်။ Manually ရေးမယ်ဆို ဒီလိုပါ။

```
<?php
        $colors[0] = "Red";
        $colors[1] = "Green";
        $colors[2] = "Blue";
?>
```

ဒါက လည်း တူတူပါပဲ ။ output ထုတ်မယ်ဆို key value အတွဲပုံစံလေးနဲ့ ထွက်လာပါလိမ့်မယ် ။

Associative Arrays

Keys(array index) တွေကို ကိုယ်တိုင် string value တွေနဲ့ assign ချထားတဲ့ array တွေကို associative array လို့ ခေါ်ပါတယ်။

```
<?php
```

```
        $ages = array(“Peter” => 22 , “Clark”=> 32 , “John”=>28);
        print_r($ages);
?>
(or)
<?php
        $ages = [“Peter” => 22 , “Clark”=> 32 , “John”=>28];
        print_r($ages);
?>
```

ဒီမှာဆိုရင် output က

Array([“Peter”] => 22 [“Clark”] => 32 [“John”] => 28) ရပါတယ်။ ဒီမှာဆိုရင် array index တွေက 0 based တွေမဖြစ်တော့ဘဲ ကိုယ်တိုင် specified လုပ်ထားတဲ့ name တွေနဲ့ key value ပုံစံ ပြန်ရပါတယ်။

```
<?php
        $ages["Peter"] = "22";
        $ages["Clark"] = "32";
        $ages["John"] = "28";
?>
```

ဒီလိုလည်း associative array ကို တည်ဆောက်လို့ရပါတယ်။

Working with Keys and Values

Array keys နဲ့ values တွေနဲ့ တွဲပြီးအလုပ်လုပ်ဖို့အတွက် PHP ရဲ့ basic array function တွေရှိပါတယ်။ အဲ့ဒီ ထဲကမှ array_combine() function က arrays နှစ် ခုကို

တစ်ခုထဲအဖြစ်ပြောင်းပြီးတော့ array တစ်ခုရဲ့ values တွေကို keys တွေအဖြစ်ပြောင်းလိုက်ပြီး နောက် array ရဲ့ values တွေကို စောနက keys တွေနဲ့ တွဲပြီး values လုပ်လိုက်ပါတယ်။ ဒီလိုပါ။

```
<?php
	$keys = ['sky', 'grass', 'orange'];
	$values = ['blue', 'green', 'orange'];
	$array = array_combine($keys, $values);
	print_r($array);
	/* Array
	(
	[sky] => blue
	 [grass] => green
	[orange] => orange
	) */
?>
```

ဒီမှာဆိုရင် $keys array နဲ့ $values array တို့ကို array_combine() function သုံးပြီးတော့ $array array ထဲကို ထည့်လိုက်တယ်။ဒါကြောင့် array_combine() function က $keys array နဲ့ $values နဲ့ကို ပေါင်းပြီးတော့ $keys ထဲက array ထဲက values တွေကို $array ရဲ့ keys တွေဖြစ်သွားပြီး $values ထဲက values တွေက $array ရဲ့ keys ရဲ့ values တွေဖြစ်သွားပါတယ်။
နောက် ထပ် သိသင့်တဲ့ array function က array_values() ပါ ။ သူက array ထဲက value တွေကြည့် သပ်သပ်စုပြီး array ဖန်တီးချင်တဲ့အခါသုံးပါတယ်။

```
<?php
	print_r(array_values($array));// ['blue', 'green', 'orange']
?>
```

နောက်ထပ် သိသင့်တဲ့ array function က array_keys() ပါ။ သူက array ထဲက keys တွေကြည့် သပ်သပ်စုပြီး array အသစ် ဖန်တီးချင်တဲ့အခါ သုံးပါတယ်။

```
<?php
        print_r(array_keys($array));//['sky','grass', 'orange']
?>
```

နောက်ထပ် array function တစ်ခုက array_flip() function ပါ။ သူက array ထဲက keys နဲ့ values ကို နေရာချိန်းပြစ်တာပါ။

```
<?php
        print_r(array_flip($array));
        /* Array
         (
         [blue] => sky
         [green] => grass
        [orange] => orange
         ) */
?>
```

Array destructuring in PHP

PHP မှာဆိုရင် array တည်ဆောက်တဲ့ [ ] ရှိပါတယ်။သူက array() function လိုမျိုး array တည်ဆောက်တာဆိုပေမဲ့ function မဟုတ်ပါဘူး။ language construct တစ်ခုပါပဲ။ သူနဲ့အတူနောက်တစ်ခု ရှိပါတယ်။ list() ဆိုတဲ့ language construct လေးပါ။ သူ့ကို array shorthand

syntax လို့လဲပြောပါတယ်။ သူက array ထဲက elements တွေကို variables တွေအဖြစ် ဆွဲထုတ်ပါတယ် ။ တစ်နည်းအားဖြင့် array ကို variables တွေအဖြစ် destructure လုပ်ပါတယ်။

```
<?php
        $array = [1, 2, 3];
        // Using the list syntax:
        list($a, $b, $c) = $array;
        // Or the shorthand syntax:
        [$a, $b, $c] = $array;
        // $a = 1
        // $b = 2
        // $c = 3
?>
```

ဒီမှာဆို array ကို list() နဲ့ variables တွေ split လုပ်လိုက်ပါတယ်။ ပြီးတော့ သူ့ကို အပေါ်မှာ ပြောတဲ့အတိုင်း shorthand ပုံစံနဲ့ [ ] နဲ့ ပြန်ပြထားပါတယ်။ ဒီမှာဆို list() နဲ့ [ ] ကတူတူပါပဲ ။ ဒါကြောင့် သူတို့ နှစ်ခုကို function လို့မသတ်မှတ်ဘဲ language construct လို့ဆိုရခြင်းပါ။

List နဲ့ နောက်ထပ် လုပ်နိုင်တာတစ်ခုက array element တွေကို skip လုပ်တာပါ။ တစ်နည်းအားဖြင့်ကျော်တာ**ပါ။**

```
<?php
        [, , $c] = $array;
        // $c = 3
```

ဒီမှာဆို array ရဲ့ ရှေ့က element နှစ်ခုကို skip လုပ်ခဲ့ပြီး နောက်ဆုံးတစ်ခုကိုပဲ ယူပြီး $c ထဲထည့်ထားပါတယ်။

နောက်ပြီး list က array တွေထဲက values တွေကို စယူရင် index 0 ကနေစပြီး ယူပါတယ်။ အကယ်၍ index 0 value မရှိရင် သူက variable ထဲကို Null ဆိုပြီး ထည့်ပါတယ်။

```
<?php
        $array = [
         1 => 'a',
        2 => 'b',
         3 => 'c',
        ];
?>
```

ဒီမှာဆို values တွေရဲ့ Index က 1 က စတယ်။ ဒါကို က list နဲ့ variable တွေကို ပြန်ယူရင် ပထမဆုံး variable က 0 Index ကို ယူတဲ့အခါ Index 0 မရှိတဲ့အတွက် Undefined array key 0 ဆိုပြီး error တက်ပါလိမ့်မယ် ။ ဆိုလိုတာ က မရှိတဲ့ index ရဲ့ value ကို ယူတာဖြစ်တဲ့အတွက် error တက်တာဖြစ်ပါတယ်။

PHP7.1 နောက်ပိုင်းမှာ non-numerical keys နဲ့ မဟုတ်တဲ့ array ထဲက values တွေကို list နဲ့ယူလို့ရပါတယ် ။ ဒီလိုပါ။

```
<?php
        $array = [
         'a' => 1,
         'b' => 2,
        'c' => 3,
        ];
        ['c' => $c, 'a' => $a] = $array;
?>
```

ဒီမှာဆို ကိုလိုချင်တဲ့ value ကို skip လုပ်ပြီးလည်းယူလို့ရပါတယ်။

လက်တွေ့မှာဆို ဒီလို အသုံးပြုပါတယ်။ list ကို parse_url နဲ့ pathinfo တို့မှာ ဒီလိုအသုံးပြုပါတယ်။ ဘာကြောင့်လဲဆိုတော့ parse_url() နဲ့ pathinfo() တို့က array return ပြန်ပေးတာဖြစ်တဲ့အတွက် list နဲ့ တွဲသုံးတာဖြစ်ပါတယ်။

```
<?php
        [
         'basename' => $file,
        'dirname' => $directory,
        ] = pathinfo('/users/test/file.png');
?>
```

pathinfo() ကနေပုံမှန်တိုင်းဆို Array ([dirname] => /users/test [basename] => file.png [extension] => png [filename] => file) ဒီလို array ပုံစံ return ပြန်ပေးပါလိမ့်မယ်။ ဒီမှာ list နဲ့ လိုချင်တာကိုပဲ skip လုပ်ပြီးယူပါတယ်။

Multidimensional Arrays

Array တစ်ခုတည်းမှာ အဲ့ဒီ array ရဲ့ elements အဖြစ် array တဲ့ပါတဲ့ array ကို ဆိုလိုပါတယ်။ ဆိုလိုတာက array တစ်ခု ထဲမှာ နောက်ထပ် sub-arrayတွေ ပါတာကို ဆိုလိုပါတယ်။

```
<?php
        $multiArray = array(
                array( “Red”, “Green” , “Blue” ),
                array(
                        "name" => "John Smith",
                       "email" => "johnsmith@mail.com",
                 )
        );
?>
```

ဒီမှာဆိုရင် array တစ်ခုထဲမှာ သူ့ရဲ့ element အဖြစ် အခြား array တွေနဲ့ ဖြစ်ပါတယ်။

echo statement နဲ့က array တစ်ခုလုံးကို output ထုတ်လို့ မရပေမဲ့ array ရဲ့ element တစ်ခုချင်းကိုတော့ output ပြန်ထုတ်လို့ရပါတယ်။ ဒီမှာဆိုရင် multiArray ထဲကမှာ ဒုတိယ array ရဲ့ email ကို output ထုတ်ချင်တယ်ဆိုပါစို့။

```
<?php
        echo “John ‘s email is “. $multiArray[1][“email”] ;
?>
```

ဆိုပြီးတော့ ထုတ်လို့ရပါတယ်။ indexed array ဆိုရင် ကိုယ်ထုတ်ချင်တဲ့ array ရဲ့ element ရဲ့ index number နဲ့ ထုတ်လို့ရပြီး associative array ဆိုရင်တော့ ကိုသတ်မှတ်ပေးတဲ့ name နဲ့ထုတ်ရပါတယ်။

```
<?php
        $ages = array("Peter" => 22 , "Clark"=> 32 , "John"=>28);
        $colors = array("Red" , "Green" , "Blue");
        echo "the color is " .$colors[0] ." and John’age is " . $ages["John"];
?>
```

PHP Object

PHP က Object Oriented Language ဖြစ်ပါတယ်။ Class နဲ့ object က Object-Oriented Programming ရဲ့ အဓိက အစိတ်ပိုင်းတွေ ဖြစ်ပါတယ်။ Class ဆိုတာ Objects တွေ အတွက် template ဖြစ်ပါတယ်။ ဆိုလိုတာက object တည်ဆောက်ရင် class ကို မှီပြီး individual objects တွေ တည်ဆောက်ပါတယ်။ ဥပမာ Fruit ဆိုတဲ့ class တစ်ခု ရှိတယ်ဆိုပါစို့ ။ သူ့မှာ name , color , weight စတဲ့ properties တွေရှိတယ်ဆိုပါစို့။ အခဲ့မှာမှာ individual objects တွေ တည်ဆောက်မယ်ဆိုရင် ဒီ Fruit ဆိုတဲ့ class ကို မှီပြီးတည်ဆောက်ပါတယ်။ ဥပမာ apple လို့ individual object တစ်ခုတည်ဆောက်လိုက်ရင် Fruit ထဲက properties တွေကို access ရပြီး “apple” , “red” စတဲ့ မတူနဲ့

different value တွေကို တည်ဆောက်လို့ရပါတယ်။ နည်းနည်း နားလည်ထားရင် ရပါပြီ။ OOP သင်ခန်းစာမှာ ထပ်ပြီး ရှင်းပြပါ့မယ်။

```
<?php
        class Fruit {
         public $name;
         public $color;
        public function __construct($name, $color) {
        $this->name = $name
         $this->color = $color;
         }
         public function message() {
        return "Fruit is an " . $this->name . " and  " . $this->color . " color .";
         }
}

        $fruit1 = new Fruit("apple", "red");
        echo $fruit1 -> message();
        echo "<br>";
        $fruit2 = new Fruit("mango", "yellow");
        echo $fruit2 -> message();
?>
```

PHP Resources

PHP resource က special data type ဖြစ်ပြီးတော့ external resource တွေကို ရည်ညွှန်းပါတယ်။ ဆိုလိုတာက resource variable တွေက external data sources ဖြစ်တဲ့ third party external application တွေရဲ့ file တွေ database စတဲ့ data sources တွေရဲ့ reference တွေ အဖြစ် အလုပ်လုပ်ပါတယ်။ PHP က အဲ့ဒီ resource variable တွေကရတဲ့ data sources တွေကို အသုံးပြုပြီးတော့ သက်ဆိုင်ရာ function လုပ်ဆောက်ချက်တွေလုပ်ပါတယ်။ ဥပမာ အားဖြင့် fopen() နဲ့ files တွေကို read-write လုပ်တာ ၊ database နဲ့ ချိတ်ဆက်ပြီး file data တွေကို handling လုပ်တာတွေပါ။

```
<?php
        $conn= mysqli_connect(“localhost”,“root”,“”,“registration”);
?>
```

ဒီလိုဆိုရင် mysqli_connect() function က နေ resource type data ကို return ပြန်ပေးလိမ့်မယ်။ နောက်ပြီး အဲ့ data ကို $conn ထဲမှာ ထည့်လိုက် ပါတယ်။

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

ဒီမှာဆိုရင် test.txt ဆိုတဲ့ resource file ကို read-only mode နဲ့ အလုပ်လုပ်လိုက်တာ ဖြစ်ပါတယ်။ var_dump() သုံးပြီး output လုပ်တဲ့အခါ resource(5) of type (stream) ဆိုပြီး variable type နဲ့ သူဖတ်တဲ့ file type ပါရပါတယ်။

gettype() and concat sign (.)

Variable type တွေကိုစစ်ချင်တဲ့အခါ php ရဲ့ built-in function တစ်ခု ဖြစ်တဲ့ gettype() function ကိုသုံးပြီး variable ရဲ့ data type ကို စစ်လို့ရပါတယ်။

```
<?php
        $my_var = 3.142;
        echo gettype($my_var) .” “. $my_var;  //double 3.142;
?>
```

ဒီမှာဆိုရင် double 3.142 ဆိုပြီး ထွက်လာပါလိမ့်မယ်။ ဒီမှာ မှတ်စရာ တစ်ခုရှိပါတယ်။ စာ တစ်ခုနဲ့ တစ်ခုကို ဆက်ချင်တဲ့အခါ PHP ရဲ့ concat sign က dot(.)ဖြစ်ပါတယ်။ ဒီမှာ space(“ “) နဲ့ ဆက်ချင်တာ ဖြစ်တဲ့အတွက် “ “ ရှေ့ကနေdot(.) နဲ့ တစ်ခါဆက်ပြီး နောက် က output value နဲ့ ထပ်ဆက်ချင်တဲ့ အတွက် dot(.) နဲ့ တစ်ခါ ထပ်ဆက်ပါတယ်။

Display HTML code

Output တွေကို HTML code တွေနဲ့လည်း တွဲပြီး ထုတ်လို့ရပါတယ်။

```
<?php
        echo “<h1>This is a Heading </h1>”;
?>
<?php
        echo “<h3 style=’color:blue’>This is a Heading with style</h3>”;
?>
```

ဒီမှာ ဆိုရင် ပထမတစ်ကြောင်းက html header h1 ပုံစံနဲ့ output ထွက်လာပါလိမ့်မယ်။ ဒုတိယ တစ်ခုက တော့ html header h3 ပုံစံကို အပြာရောင်လေးနဲ့ ရပါလိမ့်မယ်။ html နဲ့တွဲ ရေးတဲ့အခါ ပုံမှန် html code ကို double quote(“”) ထဲမှာ ရေးပါတယ်။

```
<?php
        $name = “John Smith”;
        echo “<p> My name is <bold> $name </bold> </p>”;
?>
```

Constant in PHP

Constant ဆိုတာ variable တစ်မျိုးပဲဖြစ်ပါတယ်။ သူ့ကို value တစ်ခါသတ်မှတ်ပြီးရင် value ကို ပြန်ချိန်းလို့ မရပါဘူး ။ သူ့ကို PHP scripts တွေ run နေစဉ် အချိန်းအတွင်း value ပြောင်းလည်မသွား အောင် အသေသတ်မှတ်ချင်တဲ့အခါ သုံးပါတယ်။ example အနေနဲ့ဆိုရင် database နဲ့ ချိတ်တဲ့အခါ database name နဲ့ password တွေ api key တွေ website’s base URL တွေ company name တွေကို အသေသတ်မှာထားချင်တဲ့အခါ မျိုးမှာ constant ကို သုံးပါတယ်။ နောက် ပြီး php မှာ magic constant ဆိုတာရှိပါတယ် ။ နောက်ပိုင်းမှာ ရှင်းပြပါ့မယ်။

Constant တစ်ခုကို သတ်မှတ်တော့မယ်ဆိုရင် PHP ရဲ့ built-in function ဖြစ်တဲ့ define() function ကို အသုံးပြုရပါတယ်။

```
        define( variable_name , value );
```

define() function က argument နှစ်ခုယူပါတယ်။ ပထမက ကို အသေသတ်မှတ်ချင်တဲ့ constant variable name ပါ ။ နောက်တစ်ခုက value ပါ ။ သူ့ကို ပြန်ခေါ်ချင်တဲ့အခါ သူ့ရဲ့ name ကို ပြန်ခေါ်ပြီး သုံးပါတယ်။

```
<?php
define("FB_PAGE_URL", "https://www.facebook.com/nationalcybercity ");
        echo "Thank you for visiting - " . FB_PAGE_URL ;
?>
```

ဒီမှာဆိုရင် output အနေနဲ့

Thank you for visiting - https://www.facebook.com/nationalcybercity ဆိုပြီး ထွက်ပါလိမ့်မယ်။

Variable Type Casting

Type casting ဆိုတာ variable တစ်ခု value တစ်ခုကို data type ပုံစံတစ်ခုတည်းဖြစ်အောင် ပြောင်းလည်းပေးတာကို ဆိုလိုပါတယ်။ ဥပမာ အခြား programming language တွေမှာဆိုရင် variable တစ်ခု တည်ဆောက်ရင် အဲ့ဒီ variable က ဘာ data type ဖြစ်ကြောင်းပါ တစ်ခါတည်း ကြေငြာပေးရပါတယ်။ မတူတဲ့ data type မတူတာဆိုရင် ပေါင်းခြင်း ၊ နုတ်ခြင်း တွေ လုပ်လို့မရပါဘူး ။ ပေါင်းချင်တဲ့အခါ variable တွေကို type casting လုပ်ရပါတယ်။ PHP မှာ variable type casting ကို auto လုပ်ပေးပါတယ်။

```
<?php
        $var1 = 3; // integer data type
        $var2 = 2.3; // float data type
        $result = $var + $var2;
        echo $result;
?>
```

ဒီမှာဆိုရင် output အနေနဲ့ 5.3 ဆိုပြီးတော့ ရပါလိမ့်မယ်။ Integer data type နဲ့ float data type မတူတဲ့ data type နှစ်ခုကို ပေါင်းပေမဲ့ type casting auto လုပ်ပေးပါတယ်။ ဒါပေမဲ့ က output data type ကို integer data type အနေနဲ့ရချင်ရင် ကိုယ်ကိုတိုင် type casting လုပ် လို့ရပါတယ်။

```
<?php
        $var1 = 3; // integer data type
        $var2 = 2.3; // float data type
        $result = $var1 +  (int) $var2;
        echo $result;
?>
```

ဒီမှာဆိုရင် output အနေနဲ့ 5 ဆိုပြီး ရပါလိမ့်မယ် ။ var2 variable ရှေ့မှာ type casting integer အဖြစ် ပြောင်းမယ်ပြောထားတာကြောင့် 2.3 ကနေ 2 အဖြစ် integer data type အဖြစ် ပြောင်းသွားပါတယ်။ ဒါကြောင့် ရလာတဲ့ result output က integer data type ဖြစ်ပါတယ်။

PHP Print Statement

Print Statement ကလည်း echo လိုပါပဲ browser မှာ data တွေပြဖို့အတွက် သုံးပါတယ်။ မတူတဲ့အချက်က print ကို print or print() ဆိုပြီး parenthese () ပါတာ ဖြစ်ပါတယ်။ print ကော print() နှစ်ခုလုံးက browser မှာ data တွေ output ပြဖို့ သုံးပါတယ်။ PHP7 မှာတော့ echo() ဆိုတဲ့ function ပါလာပါပြီ။ နောက်ပြီး PHP မှာ data output တွေပြဖို့အတွက် methods တွေ အများကြီး ရှိပါတယ်။ နောက်ပိုင်းမှာ ရှင်းပြပေးပါမယ်။

```
<?php
        $my_var = “Hello World”;
        print $my_var; // “Hello World”
?>
```

ဒီမှာဆိုရင် my_var ဆိုတဲ့ variable လေးထဲကို “Hello World” ဆိုတဲ့ string လေးကို store လုပ်ထားပါတယ်။ အဲ့ဒီ variable ကို output ထုတ်ချင်တဲ့အခါ print statement ကို သုံးပြီးတော့ output ထုတ်ထားပါတယ်။ ဒါကြောင့် screen ပေါ်မှာ Hello World ဆိုတဲ့ output လေးမြင်ရမှာ ဖြစ်ပါတယ်။ print statement နဲ့ output တွေကို HTML code တွေနဲ့လည်း တွဲပြီး ထုတ်လို့ရပါတယ်။

```
<?php
        print “<h1>This is a Heading </h1>”;
?>
<?php
        print “<h3 style=’color:blue’>This is a Heading with style</h3>”;
?>
```

ဒီမှာ ဆိုရင် ပထမတစ်ကြောင်းက html header h1 ပုံစံနဲ့ output ထွက်လာပါလိမ့်မယ်။ ဒုတိယ တစ်ခုကတော့ html header h3 ပုံစံကို အပြာရောင်လေးနဲ့ ရပါလိမ့်မယ်။ html နဲ့တွဲ ရေးတဲ့အခါ ပုံမှန် html code ကို double quote(“”) ထဲမှာ ရေးပါတယ်။

အခန်း(၃) - PHP Operators

What are the Operators?

Operators ဆိုတာ PHP ရဲ့ တစ်ချို့သော actions ဖြစ်တဲ့ ပေါင်းခြင်း နုတ်ခြင်း ၊ variable နှစ်ခုရဲ့ value တွေကို နှိုင်းယှဉ်ခြင်း စတဲ့ action တွေကို လုပ်ဆောင်ပေးတဲ့ symbols တွေကို ဆိုလိုပါတယ်။ PHP မှာ operators အမျိုးအစား များစွာရှိပါတယ်။

- Arithmetic operators
- Assignment operators

- Comparison operators
- Increment/Decrement operators
- Logical operators
- String operators
- Array operators
- Conditional assignment operators

ဆိုပြီးတော့ ရှိပါတယ်။

PHP Arithmetic Operators

Arithmetic Operators က PHP ရဲ့ Arithmetical Operations ဖြစ်တဲ့ addition, subtraction, multiplication စတဲ့ အလုပ်လုပ် တွေ လုပ်ပါတယ်။

+ (Addition) - variables တွေ ပေါင်းခြင်း ကို လုပ်ဆောင်ပေးပါတယ်။

$x + $y = Sum of $x and $y

- (Subtraction) - variables တွေ ရဲ့ နုတ်ခြင်း ကို လုပ်ဆောင်ပေးပါတယ်။

$x - $y = Difference of $x and $y

* (Multiplication) - variables တွေရဲ့ မြှောက်ခြင်းကို လုပ်ဆောင်ပေးပါတယ်။

$x * $y = Product of $x and $y

/ (Division) - variables တွေရဲ့ စားခြင်းကို လုပ်ဆောင်ပေးပါတယ်။

$x / $y = Quotient of $x and $y

% (Modulus) - variables တွေရဲ့ remainder ကို တွက်တဲ့ အလုပ်လုပ်ပါအတယ်။

$x % $y = Remainder of $x divided by $y

```
<?php
        $x = 10;
        $y = 4;
        echo($x + $y); // outputs: 14
        echo($x - $y); // outputs: 6
        echo($x * $y); // 0utputs: 40
        echo($x / $y); // outputs: 2.5
        echo($x % $y); // outputs: 2
?>
```

PHP Assignment Operators

Variables ထဲကို values assign ထည့်တဲ့အခါ assignment oprerators တွေကို သုံးပါတယ်။

= (Assign) - $x = $y

$x ထဲကို $y assign ထည့်လိုက်တယ်လို့ဆိုလိုတာပါ။

+=(Add and Assign) - $x += $y

$x ထဲကို $x + $y ထည့်လိုက်တယ်လို့ဆိုလိုတာပါ။ $x +=$y က $x = $x+$y နဲ့ ညီတယ် ဆိုလိုတာပါ။

-= (Subtract and Assign) - $x-=$y

$x ထဲကို $x - $y ထည့်လိုက်တယ်ဆိုလိုတာပါ။ $x -=$y က $x = $x-$y နဲ့ ညီတယ်လို့ဆိုလိုတာပါ။

*= (Multiply and Assign) - $x *= $y

$x ထဲကို $x*$y ထည့်လိုက်တယ်ဆိုလိုတာပါ။ $x *=$y က $x = $x*$y နဲ့ ညီတယ်လို့ဆိုလိုတာပါ။

/= (Divide and assign quotient) - $x /= $y

$x ထဲကို $x/$y ရဲ့ စားလဒ်ကို assign ထည့်လိုက်တယ်ဆိုလိုတာပါ ။ $x /= $y က $x = $x/$y နဲ့ ညီတယ်လို့ဆိုလိုတာပါ။

%= (Divide and assign modulus) - $x %=$y

$x ထဲကို $x% $y ကို ထည့်လိုက်တယ်ဆိုလိုတာပါ။ $x %=$y က $x = $x % $y နဲ့ ညီတယ် ဆိုလိုတာပါ။

```
<?php
        $x = 20;
        $x += 30;
        echo $x; // Outputs: 50

        $x = 50;
        $x /= 10;
        echo $x; // Outputs: 5

        $x = 100;
        $x %= 15;
        echo $x; // Outputs: 10
?>
```

PHP Comparison Operators

Comparison Operators တွေကValues နှစ်ခုကို comparison လုပ်ပေးပြီး boolean value ပြန်ပေးပါတယ်။

== (Equal) - $x == $y

$x နဲ့ $y ကတူလား စစ်ပြီး တူရင် true ပြန်ပေးပြီး မတူရင် false ပြန်ပေးပါတယ်။

=== (Identical) - $x === $y

$x နဲ့ $y က value ကော type ကောစစ်ပြီး တူရင် true ပြန်ပေးပြီး မတူရင် false ပြန်ပေးပါတယ်။

!= (Not equal) - $x !=$y

$x နဲ့ $y က မတူကြောင်း စစ်ပြီး မတူရင် true ပြန်ပေးပြီး တူရင် false ပြန်ပေးပါတယ်။

<> (Not equal) - $x <> $y

$x နဲ့ $y က မတူကြောင်း စစ်ပြီး မတူရင် true ပြန်ပေးပြီး တူရင် false ပြန်ပေးပါတယ်။

!== (Not identical) - $x !== $y

$x နဲ့ $y က value ကော type ကောမတူကြောင်း စစ်ပြီး မတူခဲ့ရင် true ပြန်ပေးပြီး တူရင် false ပြန်ပေးပါတယ်။

< (less than) - $x < $y

$x က $y ထက် ငယ် လား စစ်ပြီး ငယ်ရင် true ပြန်ပေးပြီး မငယ်ရင် false ပြန်ပေးပါတယ်။

> (greater than) - $x > $y

$x က $y ထက် ကြီး လား စစ်ပြီး ကြီးရင် true ပြန်ပေးပြီး မကြီးရင် false ပြန်ပေးပါတယ်။

>= (greater than or equal) - $x >= $y

$x က $y ထက်ကြီးလား (သို့) ညီလား စစ်ပြီး ကြီးခဲ့ရင် (သို့) ညီခဲ့ရင် true ပြန်ပေးပြီး မကြီးခဲ့ရင် (သို့) မညီခဲ့ရင် false ပြန်ပေးပါတယ်။ ဆိုလိုတာက တစ်ခုမဟုတ် တစ်ခု ဖြစ်ခဲ့ရင် true ပြန်ပေးတယ် ဆိုလိုတာပါ။

<= (less than or equal) - $x <= $y

$x က $y ထက်ငယ် လား (သို့) ညီလား စစ်ပြီး ငယ်ခဲ့ရင် (သို့) ညီခဲ့ရင် true ပြန်ပေးပြီး မငယ်ခဲ့ရင် (သို့) မညီခဲ့ရင် false ပြန်ပေးပါတယ်။ ဆိုလိုတာက တစ်ခုမဟုတ် တစ်ခု ဖြစ်ခဲ့ရင် true ပြန်ပေးတယ်ဆိုလိုတာပါ။

```
<?php
        $x = 25;
        $y = 35;
        $z = "25";
        var_dump($x == $z);  // Outputs: boolean true
        var_dump($x === $z); // Outputs: boolean false
        var_dump($x != $y);  // Outputs: boolean true
        var_dump($x !== $z); // Outputs: boolean true
        var_dump($x < $y);   // Outputs: boolean true
        var_dump($x > $y);   // Outputs: boolean false
        var_dump($x <= $y);  // Outputs: boolean true
        var_dump($x >= $y);  // Outputs: boolean false
?>
```

PHP Increment/decrement Operators

Increment/decrement operators ကို variable တွေ values တွေကို တိုးချင် လျော့ချင်တဲ့အခါ သုံးပါတယ်။

++$x (Pre-increment) - ++$x : $x ကို return မပြန်ခင် တစ် တိုးချင်တဲ့အခါ သုံးပါတယ်။

$x++ (Post-increment) - $x++ : $xကို return ပြန်ပြီး တစ် တိုးချင်တဲ့အခါ သုံးပါတယ်။

--$x (Pre-decrement) - --$x : $x ကို return မပြန်ခင် တစ် လျော့ချင်တဲ့အခါ သုံးပါတယ်။

$x-- (Post-decrement) - $x-- : $x ကို return ပြန်ပြီး တစ် လျော့ချင်တဲ့အခါ သုံးပါတယ်။

```
<?php
        $x = 10;
        echo ++$x; // Outputs: 11
        echo $x; // Outputs: 11
        $x = 10;
        echo $x--; // Outputs: 10
        echo $x; // Outputs: 9
?>
```

PHP Logical Operators

Logical operators တွေကို များသောအားဖြင့် conditional statement တွေမှာသုံးပါတယ်။

and (and) - $x and $y

$x နဲ့ $y ဖြစ်ခဲ့ရင် true ဖြစ်ခဲ့ရင် true ပြန်ပေးပြီး $x (သို့) $y တစ်ခုခုမှားခဲ့ရင် false ပြန်ပါတယ်။

or (or) - $x or $y

$x (သို့) $y တစ်ခု (သို့) တစ်ခု ဖြစ်ခဲ့ရင် တစ်ခု (သို့) တစ်ခု true ဖြစ်ခဲ့ရင် true ပြန်ပေးပြီး နှစ်ခုလုံး မှားရင် false ပြန်ပေးပါတယ်။

xor (Xor) - $x xor $y

$x (သို့) $y တစ်ခုမှ မဟုတ်ခဲ့ရင် true ပြန်ပေးပြီး $x (သို့) $y တစ်ခု (သို့) တစ်ခု ဖြစ်ခဲ့ရင် true ဖြစ်နေရင် false ပြန်ပေးပါတယ်။

&& (and) - $x && $y

$x နဲ့ $y ဖြစ်ခဲ့ရင် true ဖြစ်ခဲ့ရင် true ပြန်ပေးပြီး $x (သို့) $y ထဲက တစ်ခုခု မှားခဲ့ရင် false ပြန်ပါတယ်။

|| (or) - $x || $y

$x (သို့) $y တစ်ခု (သို့) တစ်ခု ဖြစ်ခဲ့ရင် တစ်ခု (သို့) တစ်ခု true ဖြစ်ခဲ့ရင် true ပြန်ပေးပြီး နှစ်ခုလုံး မှားခဲ့ရင် false ပြန်ပေးပါတယ်။

! (not) - !$x

$x သာမှားခဲ့ရင် true ပြန်ပေးပြီး မှန်ခဲ့ရင် false ပြန်ပေးပါတယ်။

```
<?php
        $x = 100;
        $y = 50;

        if ($x == 100 and $y == 50) {
        echo "Hello world!";
        }
        if ($x == 100 or $y == 80) {
        echo "Hello world!";
        }
        if ($x == 100 xor $y == 80) {
        echo "Hello world!";
        }
        if ($y !== 90) {
        echo "Hello world!";
        }
?>
```

PHP String Operators

String နှစ်ခုကို ဆက်ချင်တဲ့အခါ append လုပ်ချင်တဲ့အခါ သုံးပါတယ်။

```
. (Concatenation) - $str1 . $str2
```

အထက်မှာ ဖော်ပြထားပြီးပါပြီ။ string နှစ်ခုကို ဆက်ချင်တဲ့အခါ သုံးပါတယ်။

```
.= ( concatenation assign ) - $str1 .= $str2
```

$str1 ကို $str2 က append လုပ်ချင်တဲ့အခါ သုံးပါတယ်။

```
<?php
	$x = "Good";
	$y = $x ." Morning";
	// now $y contains "Good Morning "
	echo $y;
?>
```

ဒီမှာဆိုရင် $x မှာ “Good” ဆိုတယ် string ရှိတယ်။ နောက်ပြီး $y ထဲကို $x နဲ့ “Morning” နဲ့ ကို concat လုပ်လိုက်တယ်။ ဒါကြောင့် $y ထဲမှာ “Good Morning”ဆိုပြီး ဝင်သွားတယ်။ အဲ့ဒီလို မလုပ်ဘဲနဲ့ ဒီ $y ကို ထပ်မတည်ဆောက်တော့ပဲနဲ့ $x ကို ပဲ concat and reassign လုပ်ရင် ရပါတယ်။ အဲ့ဒီလို လုပ်ဖို့ဆိုရင် ဒီ .= (concat and assign) ဆိုတဲ့ သင်္ကေတလေးကို သုံးရပါမယ်။ အသုံးပြုပုံက ဒီလိုပါ။

```
<?php
	$x = "Good";
	$x.= " Morning";
	echo $x;
?>
```

ဒီလိုဆိုရင် လည်း output ကတူတူပါပဲ ။ “Good Morning” ဆိုပြီးတော့ရ ပါတယ် ။ နောက်ထပ် variable တစ်ခု ထပ်မလိုတော့သလို ရေးရတာလည်း ပိုတိုသွားပါတယ်။

ကျနော် အရှေ့ သင်ခန်းစာတွေမှာ ဆိုရင် integer variable တွေကို “” (double quote) ထဲမှာထည့်ပြီး echo နဲ့ output ထုတ်ပြပါတယ်။ အဲ့ဒီလိုပဲ string variable ကိုလည်း “” အတွင်းမှာ output တွဲထုတ်လို့ရပါတယ်။ အဲ့ဒီလို string data ထဲမှာ variable adding လုပ်ခြင်းကို variable inteerpolation လုပ်တယ်လို့ခေါ်ပါတယ်။

```
//Variable Interpolation
echo “Welcome $name!”;
```

နောက်ထပ် variable ကို output ထုတ်တဲ့ ပုံစံဖြစ်တဲ့ {} နဲ့ လည်း output ထုတ်လို့ရပါတယ်။

```
//using curly brackets {}
echo “Welcome {$name}!”;
```

နောက်ပြီးတော့ သူကို concat sign တွေနဲ့လည်း output ထုတ်လို့ ရတယ်။ ဒီလိုပါ။

```
//with concatenations
echo "Welcome ".$name."!";
```

ဟုတ်ပြီ နောက်တစ်ခုအနေနဲ့ ရှေ့ပိုင်း သင်ခန်းစာတွေမှာ ပြော ခဲ့တယ် print နဲ့ echo နဲ့ ကွားခြားပုံတွေပြောခဲ့ပါတယ်။ အဲ့ထဲက မှ echo နဲ့ string variable တွေကို , နဲ့ခြားပြီး multiple string variable တွေကို output ထုတ်လို့ရပါတယ်။ ဆိုလိုချင်တာက concatenations တွေကို ရှောင်ပြီး ထုတ်တဲ့ပုံစံပေါ့။

```
// avoiding concatenations
echo "Welcome ", $name, "!";
```

PHP Array Operators

Array operators တွေကို arrays တွေ compare လုပ်ချင်တဲ့အခါ သုံးပါတယ်။

+(Union) - Array + Array

Array တွေကို ပေါင်းချင်တဲ့အခါ union လုပ်ချင်တဲ့အခါ သုံးပါတယ်။

== (Equality) - $x == $y

$x array ရဲ့ key value အတွဲတွေနဲ့ $y array ရဲ့ key value အတွဲတွေ တူတဲ့အခါ true ပြန်ပေးပါတယ်။

=== (Identity) - $x === $y

$x array နဲ့ $y array တို့ရဲ့ key value အတွဲတွေ ၊ order of type တွေ တစ်ထပ်ထည်းကျတဲ့အခါ true ပြန်ပေးပါတယ်။

!= (Inequality) - $x != $y

$x array နဲ့ $y array တို့ မတူတဲ့အခါ true ပြန်ပေးပါတယ်။

<> (Inequality) - $x <> $y

$x array နဲ့ $y array တို့ မတူတဲ့အခါ true ပြန်ပေးပါတယ်။

!== (Non-Identity) - $x!==$y

$x array နဲ့ $y array တို့ရဲ့ key value အတွဲတွေ ၊ order of type တွေ တစ်ထပ်ထည်းမကျတဲ့အခါ true ပြန်ပေးပါတယ်။

```
<?php
        $a = array("a" => "apple", "b" => "banana");
        $b = array("a" => "red", "b" => "green", "c" => "orange");
```

```
        $c = $a + $b; // Union of $a and $b
        echo "Union of \$a and \$b : <br />";
        var_dump($c);
        $c = $b + $a; // Union of $b and $a
        echo "<br />Union of \$b and \$a : <br />";
        var_dump($c);
        ?>


<?php
        $x = array("a" => "Red", "b" => "Green", "c" => "Blue");
        $y = array("u" => "Yellow", "v" => "Orange", "w" => "Pink");
        $a = array("a" => "apple", "b" => "banana");
        $z = $x + $y; // Union of $x and $y
        var_dump($z);
        var_dump($x == $y);   // Outputs: boolean false
        var_dump($x <> $y);   // Outputs: boolean true
        var_dump($x !== $y);  // Outputs: boolean true
?>
```

Operators တွေကို ဒီလိုလည်းတွဲသုံးလို့ရပါတယ်။

```
array equality (==) and identity(===) operators
```

```
<?php
        $a = array("0" => "banana","1" => "apple", );
        $b = array( "banana", "apple");
        var_dump($a == $b);
```

```
        var_dump($a === $b);
?>
```

PHP Spaceship Operator

Spaceship operator (<=>) ကို expression နှစ်ခုကို compare လုပ်ချင်တဲ့အခါ သုံးပါတယ်။ သူ့ကို combined comparison operator လို့လည်း ခေါ်ပါတယ်။

Spaceship operator ကနှိုင်းယှဉ်ချင်တဲ့နှစ်ခု operands နှစ်ခု တူတဲ့အခါ 0 return ပြန်ပါတယ်။ operands နှစ်ခုယှဉ်တဲ့အခါ ဘယ်ဖက်က ကြီးနေရင် 1 return ပြန်ပါ ပြီး ညာဖက်က ကြီး နေရင် -1 return ပြန်ပါတယ်။

```
<?php
        // Comparing Integers
        echo 1 <=> 1; // Outputs: 0
        echo 1 <=> 2; // Outputs: -1
        echo 2 <=> 1; // Outputs: 1
        // Comparing Strings
        echo "x" <=> "x"; // Outputs: 0
        echo "x" <=> "y"; // Outputs: -1
        echo "eb" <=> "ag"; // Outputs: 1
?>
```

ဒီမှာဆိုရင် မှတ်စရာတစ်ခုရှိပါတယ်။ stringတွေကို compare လုပ်တဲ့အခါ left to right လုပ်ပြီးတော့ string ရဲ့ ပထဆုံး character ကို ပဲကြည့်ပါတယ်။ ဒါကြောင့် left to right သွားရင် “y” က “x” ထက် ကြီးတာကြောင့် "x" <=> "y" ယှဉ်တဲ့အခါ -1 ရပြီး “eb” နဲ့ “ag” က 1 ပြန်ပေးပါတယ်။

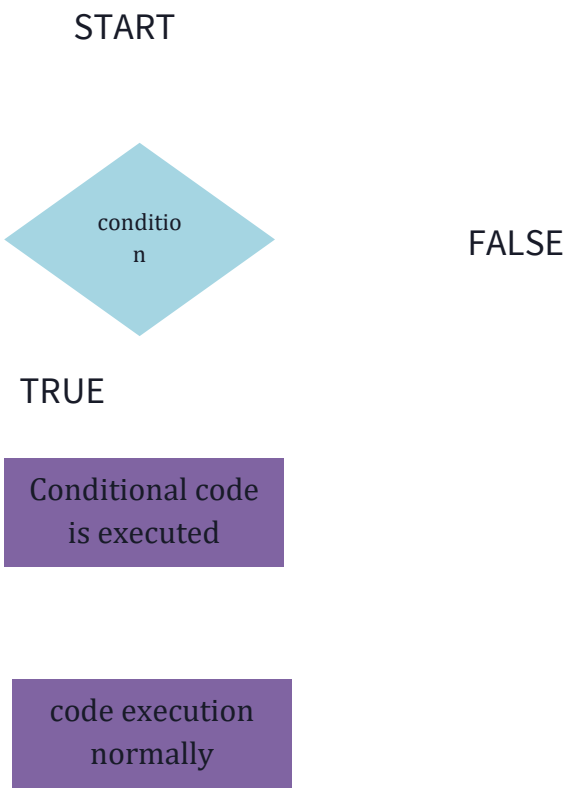
အခန်း(၄) - PHP Control Structure

What is the control structure?

လွယ်လွယ်ပြောရရင် control structure ဆိုတာက application ရဲ့ code execution flow ကို ထိန်းချုပ်တာခြင်းဖြစ်ပါတယ်။ ပုံမှန်အားဖြင့်ဆိုရင် program တစ်ခုက တစ်လိုင်းပြီး တစ်လိုင်း အလုပ်လုပ်ပါတယ်။ အဲ့ ဒီ လို အလုပ်လုပ်နေတာကို control structure တစ်ခုနဲ့ conditions တွေပေါ် မူတည်ပြီးတော့ flow ကို ပြောင်းလိုက်လို့ရပါတယ်။

Control structure က php ရဲ့ အဓိက core feature တစ်ခုဖြစ်ပြီးတော့ မတူညီတဲ့ situations တွေ input တွေ ပေါ် မူတည်ပြီးတော့ အလုပ်တွေလုပ်ပါတယ်။

Control structure ကို ပုံလေးနဲ့ ရှင်း ပြပါ့မယ်။



ဒီ diagram မှာဆိုရင် ပထမဆုံး condition စစ်ပါတယ်။ အကယ်၍ condition က true ဖြစ်ခဲ့ရင် condition အရ အလုပ်လုပ်မဲ့ code တွေ အလုပ်လုပ်ပါတယ် ။ ဒီမှာ မှတ်ထားစရာတစ်ခုက condtion စစ်ပြီးလို့ condition အရ အလုပ်လုပ်ပြီးတဲ့အချိန်မှာ code တွေ က ပုံမှန်အတိုင်းပဲ အလုပ်လုပ်သွားပါတယ် ။

PHP မှာ control structures တွေ အများကြီး support လုပ်ပါတယ်။

သူတို့ကို ထပ်ပိုင်းရင် decisions making statements နဲ့ loops ဆိုပြီး ရှိပါမယ်။

အဲ့ဒီထဲကမှ decisions making statements တွေအကြောင်း အရင်ပြောပြပါ့မယ်။

Decisions making statements

if...else and switch...case

အခြားသော programming languages တွေလိုပဲ PHP မှာလည်း condition တွေအရ ထွက်လာတဲ့ result value အရ action တွေ လုပ်ဆောင်တာမျိုးရှိပါတယ်။ ဆိုလိုတာ က condition တစ်ခုဖန်တီးပြီး condition ကရလာတဲ့ result values true or false အရ actions တွေလုပ်တာမျိုးကိုဆိုလိုတာပါ။

PHP မှာ decisions making statements တွေ များစွာရှိပါတယ်။

- if Statement
- if...else Statement
- if...elseif...else Statement
- switch...case Statement တို့ဖစ်ပါတယ်။

if Statement

if statement က condition စစ်လို့ true ဖြစ်တဲ့အခါ အလုပ်လုပ်ရန် အသုံးပြုပါတယ်။

```
if(condition){
        // Code to be executed
}
```

သူရဲ့ syntax က ဒီလိုပါ။ condition နေရာမှာ စစ်ပြီးတော့ true ဖြစ်ခဲ့ရင် if statement အတွင်းက code က executed ဖြစ်မှာပါ။

```
<?php
        $d = date("D");
        if($d == "Fri"){
          echo "Have a nice weekend!";
        }
?>
```

ဒီမှာဆိုရင် php ရဲ့ built-in function ဖြစ်တဲ့ date() function ကိုအသုံးပြုထားပါတယ်။ day ကို လိုချင်တာဖြစ်တဲ့အတွက် “D” ဆိုပြီး argument ပေးထားပါတယ်။ ဒါကြောင့် day ရဲ့ အတိုကောက်ထွက် လာပါလိမ့်မယ်။ day က “Fri” နဲ့သာ ညီခဲ့ရင် condition true ဖြစ်ခဲ့ရင် if statement အတွင်းက code ဖြစ်တဲ့ “Have a nice weekend!” ဆိုတဲ့ output ရမှာပါ ။ condition true မဖြစ်ခဲ့ရင် if statement က အလုပ်မလုပ်တော့ပဲ skip ဖြစ်သွားပါတယ်။

if...else Statement

else statement က if statement ရဲ့ alternative choice ဖြစ်ပါတယ်။ ဆိုလိုတာက if statement က condition true ဖြစ်ရင် အလုပ်လုပ်မှာ ဖြစ်ပြီး else statement က condition false ဖြစ်ခဲ့ရင် အလုပ်လုပ်မယ်ဆိုလိုတာပါ။ ဒါကြောင့် if...else statement က condition ပေါ် အခြေခံပြီး အလုပ်တစ်ခု လုပ်ပါတယ်။

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

သူရဲ့ syntax က ဒီလိုပါ။ else က if condition ပေါ် အခြေခံပြီး false ဖြစ်ရင် လုပ်မှာ ဖြစ်တဲ့အတွက် သူ့မှာ condition ထပ်စစ်စရာ မလိုပါဘူး။ ထပ်မစစ်ရပါဘူး။

```
<?php
        $d = date("D");
        if($d == "Fri"){
                echo "Have a nice weekend!";
        } else{
                echo "Have a nice day!";
        }
?>
```

ဒီမှာဆိုရင် $d က condition အရ “Fri” ဖြစ်လို့ true ဖြစ်ခဲ့ရင် “Have a nice weekend” ဆိုပြီး output ပြမှာဖြစ်ပြီး condition က “Fri” မဖြစ်ပဲ အခြား အခြေနေတစ်ခုခုဖြစ်ခဲ့လို့ false ဖြစ်တာနဲ့ “Have a nice day!” ဆိုပြီး else statement က အလုပ်လုပ်မှာပါ။

if...elseif...else Statement

if...elseif..else statement က if..else statement တွေ ပေါင်းထားတာပါ။

```
        if(condition1){
                // Code to be executed if condition1 is true
        } elseif(condition2){
                // Code to be executed if the condition1 is false and condition2 is true
        } else{
                // Code to be executed if both condition1 and condition2 are false
```

```
        }
```

သူ့ရဲ့ syntax က ဒီလိုပါ။ ဆိုလိုတာက အရင် if_else statement လို if condition false ဖြစ်တာနဲ့ else ကို မသွားသေးဘဲ နောက်ထပ် တစ်ဆင့် elseif() နဲ့ ထပ်စစ်တာဖြစ်ပါတယ်။ ဒါကြောင့် if condition false ဖြစ်ခဲ့ရင် elseif() ကထပ်စစ်ပါတယ်။ အကယ်၍ elseif() က ပါ false ဖြစ်မှသာ else က အလုပ် လုပ်မှာ ဖြစ်ပါတယ်။ elseif() က တစ်ခါ ပဲ စစ်ရတာ မဟုတ်ပါဘူး။ အကြိမ်များစွာ စစ်လို့ရပါတယ်။

```
<?php
        $age = 50;
        if ($age < 30){
                echo "Your age is less than 30!";
        }
        elseif ($age > 30 && $age < 40){
                echo "Your age is between 30 and 40!";
        }
        elseif ($age > 40 && $age < 50){
                echo "Your age is between 40 and 50!";
        }
        else{
                echo "Your age is greater than 50!";
        }
?>
<?php
        /* Correct Method: */
        if ($a > $b):
                echo $a." is greater than ".$b;
        elseif ($a == $b): // Note the combination of the words.
```

```
                echo $a." equals ".$b;
        else:
                 echo $a." is neither greater than or equal to ".$b;
        endif;
?>
```

switch..case statement

switch..case statement က if...elseif..else statement နဲ့ အလုပ်လုပ်ပုံခြင်း တူပါတယ်။ switch..case မှာလည်း variable တစ်ခုကို case တွေ အရ condition စစ်ပြီး true ဖြစ်တဲ့ case statement တစ်ခုက ပဲ အလုပ်လုပ် ပြီး break ဖြစ်သွားပါတယ်။

```
switch(n){
   case label1:
      // Code to be executed if n=label1
   break;
   case label2:
      // Code to be executed if n=label2
   break;
   ...
   default:
      // Code to be executed if n is different from all labels
}
```

ဒါက switch...case ရဲ့ syntax ဖြစ်ပါတယ်။ n နေရာမှာ condition စစ်မဲ့ variable ထည့်ရမှာ ကို စစ်ချင်တဲ့ case တွေနဲ့ စစ် မှာ ဖြစ်ပါတယ်။ ဥပမာ n က label1 နဲ့ ညီခဲ့ရင် label1 နဲ့ စစ်ထားတဲ့ case statement က code တွေ အလုပ်လုပ်မှာ ဖြစ်ပါတယ်။ case နောက်မှာ ကိုစစ်ချင်တဲ့ variable က string

ဒါမမဟုတ် integer ကိုစစ်ချင်တဲ့ data type ရေးလို့ရပါတယ်။ string ဆိုရင် double quote(“”) နဲ့ ရေးရန်သတိပြုပါ။

```
<?php
        $today = date("D");
        switch($today){
                case "Mon":
                echo "Today is Monday. Clean your house.";
                break;
        case "Tue":
                echo "Today is Tuesday. Buy some food.";
                break;
        . . .
        case "Sat":
                echo "Today is Saturday. It’s movie time.";
                break;
        case "Sun":
                echo "Today is Sunday. Do some rest.";
                break;
        default:
                echo "No information available for that day.";
                break;
        }
?>
```

ဒီမှာဆို ကိုစစ်ချင်တဲ့ variable ကို switch() ထဲမှာ စစ်ပါတယ်။ အကယ်၍ $today က “Mon” နဲ့ ညီရင် “Mon” နဲ့ညီရင် လုပ်မဲ့ case က code က အလုပ်လုပ်ပြီး switch..case statement က နေ break

ဖြစ်သွားပါတယ်။ အကယ်၍ “Mon” နဲ့ မညီ ခဲ့ရင် နောက် ထပ် case တစ်ခုနဲ့ ထပ်စစ်မှာ ဖြစ်ပါတယ်။ case အားလုံးနဲ့ မညီတဲ့အခါ သူက default ကို အလုပ်လုပ်မှာဖြစ်ပါတယ်။ အကယ်၍ default statement မပါခဲ့ခင် case တစ်ခုနဲ့မှာ မမှန်ခဲ့ရင် switch case statement က အလုပ်လုပ်တော့မှာ မဟုတ်ပါဘူး။

ဒီမှာ မှတ်စရာ တစ်ခုရှိပါတယ်။ အကယ်၍ case တစ်ခုပြီးတိုင်း break မလုပ်ခဲ့ရင် case တစ်ခုမှာ မှန်ခဲ့ရင်လည်း အောက်က case တွေနဲ့ ထပ်စစ်နေမှာ ဖြစ်ပါတယ်။ ဒါက if...elseif...else နဲ့ မတူတဲ့ အချက်ပါ။ ဒါကြောင့် case တစ်ခုပြီးတိုင်း break လုပ်ခဲ့ပါ။

Ternary Operator

Ternary Operator က conditions တွေ comparisons တွေ စစ်တဲ့ conditional operator တစ်ခုပါ။ ပြီးတော့ သူက if..else statement ရဲ့ alternative ပါ ။ ဆိုလိုတာက if..else statement နဲ့ အလုပ်လုပ်ပုံခြင်းတူပါတယ်။ ဒါကြောင့် သူ့ကို if...else statement ရဲ့ shorthand လို့လည်းဆိုပါတယ်။ operators က left to right အလုပ်လုပ်ပါတယ်။

```
(Condition) ? (Statement1) : (Statement2);
```

Syntax က ဒီလိုပါ။ သူ့မှာ operands သုံးခုရှိပါတယ်။ ပထမတစ်ခုက condition ဖြစ်ပါတယ်။ အဲ့ ဒီ condition နေ boolean value ပြန်ပေးပါတယ်။ အကယ်၍ true ဖြစ်ခဲ့လျှင် Statement1 က အလုပ်လုပ်ပါတယ်။ false ဖြစ်ခဲ့လျှင် Statement2 က အလုပ်လုပ်ပါတယ်။

if...else statement

```
<?php
        if($age < 18){
                echo 'Child';
        } else{
        echo 'Adult';
```

```
        }
?>
```

ternary operator

```
<?php echo ($age < 18) ? 'Child' : 'Adult'; ?>
```

ဒီမှာဆိုရင် နှစ်ခုလုံးက output တူတူပဲရပါလိမ့်မယ်။

Null Coalescing Operator

PHP မှာ ပေးထားတဲ့ variable က NULL ဟုတ်မဟုတ်ကို isset() function ကိုသုံးပြီး စစ်ပါတယ်။ null coalescing operator ဆိုတာ ternary operator နဲ့ isset() function ကို ဆက်ထားတယ်လို့ဆိုလို့ရပါတယ်။

```
        (Condition)?(Statement1)?(Statement2);
```

Syntax က ဒီလိုပါ။ ternary operator လိုပါပဲ။ ဒါပေမဲ့ ဒီလိုအသုံးပြုပါတယ်။

```
        <?php
                $user= $_GET['user'] ?? 'nobody';
        ?>
```

First operands က variable က null ဟုတ်မဟုတ်စစ်ပြီး null မဟုတ်ရင် user ရဲ့ value ကို ယူပြီး null ဆိုရင် second operands ဖြစ်တဲ့ “nobody” ကို $user ရဲ့ value အဖြစ်ထည့်ပေးလိုက်ပါတယ်။ သူတကယ် အလုပ်လုပ်ပုံက ဒီလိုပါ။

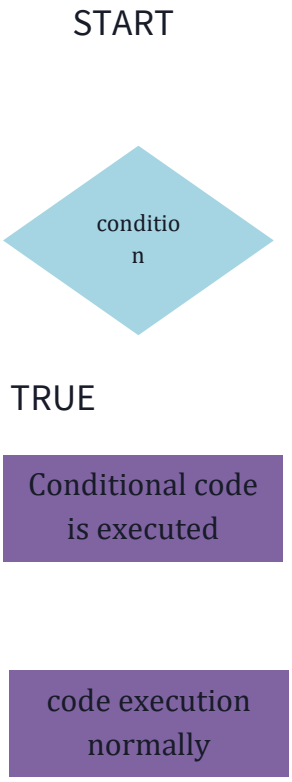
```
<?php
```

```
        $user= isset($_GET['user']) ? $_GET['user'] :'nobody';
?>
```

isset() နဲ့ user က null ဟုတ် မဟုတ် စစ်ပါတယ်။ value ရှိခဲ့တယ်ဆိုလျှင် $_GET[‘user’] က ရတဲ့ value အတိုင်း ယူမှာဖြစ်ပြီး isset() နဲ့ စစ် လို့ null ဖြစ်ခဲ့မယ်ဆိုရင် ‘nobody’ ကို ယူမှာဖြစ်ပါတယ်။ ဒါကြောင့် ရလာတဲ့ result ကအတူတူပါပဲ။ရေးရတာ code shorter ဖြစ်ပြီး ရိုးရှင်းပါတယ်။ ဒီ operators တွေ က အရမ်းအသုံးများပါတယ်။

Loops

တစ်ခါတစ်လေမှာ code တွေကို ထပ်ခါထပ်ခါ အလုပ်လုပ်စေချင်တဲ့အခါမျိုးတွေရှိပါတယ်။ အဲ့ဒီလို block of codes ကို အခြေနေတစ်ခု မှန်နေသေးသ၍ ထပ်ခါထပ်ထပ်ခါ ပြန် run ချင်တဲ့အခါ မျိုးမှာ loop တွေကို အသုံးပြုပါတယ်။ ဆိုလိုတာက တစ်ချို့ codes တွေကို condition က false ဖြစ်သည်အထိ အလုပ်ကိုဆက်တိုက်လုပ်စေတာပါ။



ဒီမှာဆိုရင် loop condition လေးတစ်ခုရှိပါတယ်။ အကယ်၍ condition က true ဖြစ်ခဲ့ရင် conditional code က အလုပ်လုပ်မှာဖြစ်ပါတယ်။ ဒီမှာ မှတ်စရာ တစ်ခုက conditional code ပြီးတဲ့အချိန်မှာ control က loop condition ကိုပြန်သွားပါတယ်။ ဒီလိုမျိုး loop ပတ်ပြီး conditional code တွေလုပ်ပြီး condition က false ဖြစ်တဲ့အချိန်မှ code flow တိုင်း အလုပ်ပြန်လုပ်ပါတယ်။

Different Types of Loops in PHP

PHP မှာ loops types လေးမျိုးရှိပါတယ်။

- while - condition မှန်နေသေးသ၍ loop က ထပ်ခါ ထပ်ခါ ပတ်နေပါတယ် ။
- do...while - condition တစ်ခါစစ်ပြီးတိုင်းမှာ block of codes အလုပ်တစ်ခါ လုပ်ပါတယ်။ condition တစ်ခုက လည်း true ဖြစ် ကို သတ်မှတ်ထားတဲ့ condition ကလည်း true ဖြစ်တဲ့အခါ ထပ်ခါထပ်ခါ အလုပ်လုပ်ပါတယ်။ loop ပတ်ပါတယ်။
- for - counter က ကိုယ်သတ်မှတ်ထားတဲ့ number ရောက်တဲ့အထိ loop ပတ်ပါတယ်။
- foreach - array ရဲ့ element တွေကို loops ပတ်တဲ့အခါ အသုံးများပါတယ်။

while loop

သတ်မှတ်ထားတဲ့ condition က true ဖြစ်နေသ၍ block of codes ကို အလုပ်လုပ်ချင်တဲ့အခါ while loop ကို အသုံးပြုပါတယ်။

```
while (condition is true) {
    code to be executed;
}
```

Syntax က ဒီလိုပါတယ်။ while() ထဲက condition true ဖြစ်နေသ၍ while statement က loop ပတ် အလုပ်လုပ်နေမှာပါ။

```
<?php
    $i = 1;
    while ($i < 6){
        echo $i;
```

```
        $i++;
}
```

ဒီမှာဆိုရင် output က 12345 ဆိုပြီး 6 အောက် ငယ်တဲ့ထိ output ထွက်ပါလိမ့်မယ်။ code ကို ပြန်ရှင်းရင် $i က initialize ပါ ။ loop ကို ဘယ် လောက်ကနေ စ ပတ်မယ်ကို ဆိုလိုတာပါ။ $i < 6 က while loop ကို ဘယ်ချိန်ထိ ပတ်မယ်ကို ဆိုလိုတာပါ။ ဒီမှာ less than 6 ဖြစ်သည့်အတွက် while loop က 5 ထိ ပတ်ပါတယ်။ $i++ က loop တစ်ခါပတ်ပြီး တိုင်း iteration တစ်ခါလုပ်ပြီးတိုင်း 1 တိုးမယ်ဆိုလိုတာပါ။ ဒါကြောင့် loop တစ်က တစ် ကနေ စ ပတ်ပြီးတော့ iteration တစ်ခါပြီးတိုင်း 1 တိုး နောက်ဆုံး 6 အောက် ငယ်တဲ့ 5 ကို ရောက်တဲ့အခါ မှာ ရပ်သွားပါတယ်။

while loop နဲ့ Fibonacci ကိန်းစဉ်တန်းလေးတွက်ကြည့်ရအောင်။

Fibonacci ကိန်းစဉ်တန်းအကြောင်း မှတ်မိမယ်ထင်ပါတယ်။

ဒီမှာ တစ်ဆယ်ထိ တွက်ကြည့်ရအောင် ။

```
{0,1,1,2,3,5,8,13,21,34,55,……}
<?php
        $max = 0;
        $first = 0;
        $second = 1;
        $result=0;
        echo $i," ," , $j," ,";
        while ($max < 10 )
        {
                $result = $first + $second;
                $first = $second;
                $second = $result;
                $max++;
                echo $result," ," ;
```

```
        }
?>
```

do...while loop

do...while loop က while loop နဲ့တော်တော်လေးဆင်တူပါတယ်။ သူရဲ့ block အတွင်းမှာ ရှိတဲ့ code ကို အရင်ဆုံး တစ်ကြိမ် အလုပ်လုပ်ပါတယ်။ ပြီးမှာ condition ကို check လုပ်တယ်။ codition မှန်နေသ၍ loop ဆက် ပတ်ပါတယ်။

```
do {
        code to be executed;
} while (condition is true);
```

Syntax က ဒီလိုပါ။ ပထမဆုံး အကြိမ်တော့ အမြဲတန်း အလုပ်လုပ်တယ်။ ပြီးမှ condition မှန် နေသ၍ loop ထပ် ပတ်ပါတယ်။

```
<?php
        $x = 1;
        do {
                echo "The number is: $x <br>";
                $x++;
        } while ($x <= 5);
?>
```

ဒီမှာဆိုရင် output က ပထမဆုံး 1 ကြိမ် အလုပ်လုပ်ပြီး “The number is: 1” ဆိုပြီး ထွက်ပါတယ်။ ပြီးမှ 1 ပေါင်းတယ် ။ ပြီး မှ condtion ကို စစ်ပါတယ်။ ဒါကြောင့် 5 နဲ့ ညီ သို့ မဟုတ် ငယ်တဲ့ထိ loop ပတ်ပြီး iterate လုပ် နေပေးပါတယ်။

ဒါကြောင့် သူက while loop နဲ့ ဆင်ပါတယ်။ မတူတဲ့ အချက် က while loop က condition ကို စစ်ပါတယ်။ condition true ဖြစ်မှ iteration စလုပ်ပါတယ်။ condition က false ဖြစ်ခဲ့ရင် while loop တစ်ခုလုံး အလုပ်မလုပ်တော့ပါဘူး။

do...while loop က အမြဲတန်း execute တစ်ခါ အရင် လုပ်ပါတယ်။ ဆိုလိုတာက condition ကို မစစ်ခင် block ထဲက code တွေက တစ်ခါ execute ဖြစ်ပြီးပါပြီ။ condition မှန်လျှင် အလုပ်ဆက်လုပ်ပြီး loop ပတ်ပြီး condition မမှန်ခဲ့ တစ်ခါ execute ဖြစ်ပြီးတာနဲ့ ပြီး သွားပါတယ်။

for loop

ကိုသတ်မှတ်တဲ့ number of times သတ်မှတ်ထားတဲ့ အကြိမ်အရေအတွက် ရောက်တဲ့အထိ loop ကို ပတ်ချင်တဲ့အခါ for loop ကိုအသုံးပြုပါတယ်။

```
for(initialization; condition; increment){
   // Code to be executed
}
```

Syntax က ဒီလိုပါ။ initialization က ကိုသတ်မှတ် ချင်တဲ့ counter variable ကို ဆိုလိုတာပါ။ iteration တစ်ခါလုပ်ပြီး တိုင်း အဲ့ဒီ variable ရဲ့ value က ပြောင်းလည်းနေမှာပါ။ condition က condition စစ်တာကို ဆိုလိုတာပါ။ condition true ဖြစ်ခဲ့လျှင် loop ဆက် ပတ်ပြီး condition false ဖြစ်တဲ့အခါ loop က ရပ်သွားမှာ ဖြစ်ပါတယ်။ နောက်ဆုံး expression ကတော့ variable ရဲ့ value ကို ပြောင်းလည်းတာပါ။ ဆိုလိုတာက increase လုပ်ချင်တာပဲဖြစ်ဖြစ် decrease လုပ်ချင်တာပဲဖြစ်ဖြစ် လုပ်ပါတယ်။ example လေးနဲ့ ပြပါမယ်။

```
<?php
        for ($x = 0; $x <= 5; $x++) {
                echo "The number is: $x <br>";
        }
```

```
?>
//output
The number is: 0
The number is: 1
The number is: 2
The number is: 3
The number is: 4
The number is: 5
<?php
        for ($i=1; $i<=10; ++$i){
                echo sprintf("The square of %d is %d.</br>", $i, $i*$i);
        }
?>
```

ဒီမှာဆိုရင် $x = 0 ဆိုပြီးတော့ counter ကို 0 က စမယ် ဆိုပြီး initialize လုပ် ပါတယ်။ ပြီး တော့ $x <=5 ဆိုပြီး 5 အောက် ငယ် မငယ် သို့ ကြီး မကြီး condition စစ်ပါတယ်။ နောက်ပြီး $x++ ဆိုပြီးတော့ iteration တစ်ခါပြီး တိုင်း $x ရဲ့ value ကို တစ် တိုးထားပါတယ်။ ဒါကြောင့် 0 က စမယ် လို့ initialize လုပ်ထားတာကြောင့် output က “The number is : 0” ဆိုတာကနေစပါတယ်။ ပြီးမှ condition စစ်ပြီးတော့ condition အရ output က 5 အောက် ငယ် သို 5 နဲ့ ညီ တဲ့ အထိ loop ပတ်ပြီး output ထွက်ပါတယ်။ ဒါကြောင့် for loop က initialize စလုပ်တဲ့ နေရာက နေ စပါတယ်။ ပြီးနောက် condition စစ်ပါတယ်။ condition တစ်ခါစစ်ပြီး တိုင်း true ဖြစ်တိုင်း iteration တစ်ခါ လုပ်ပြီး တိုင်း increment က အလုပ်လုပ်ပါတယ်။ အဲ့ဒီချိန် counter variable က value အသစ် ဖြစ်သွားပါတယ်။ အဲ့ဒီအချိန် condition နောက်တစ်ခါ ထပ်စစ်ပါတယ်။ condition false ဖြစ်ခဲ့လျှင် for loop က ရပ်သွားပါတယ်။

foreach Loop

foreach loop ကို array elements တွေကို iterate လုပ်တဲ့နေရာမှာ အသုံးပြုပြီးတော့ key/value ပုံစံနဲ့ loop ပတ်ပါတယ်။

```
foreach ($array as $value) {
        code to be executed;
}
```

ဒါက first syntax ပုံစံပါ။ $array နေရာမှာ ကို iterate လုပ်ချင်တဲ့ array ကို ထည့်ရပါတယ်။ အဲ့လို ထည့်လိုက်တဲ့အခါ as $value က array ထဲမှာ ရှိတဲ့ element တစ်ခုချင်းဆီကို value ထဲကို ထည့်ပါတယ်။ ပထတစ်ကြိမ် iterate လုပ်ပြီးတဲ့အခါ $value ထဲက array ရဲ့ နောက် element ကို ထပ်ထည့်ပြီး array element ကုန်တဲ့အထိ loop ပတ်သွားပါတယ်။

ဥပမာ -

```
<?php
        $colors = array("red", "green", "blue", "yellow");
        foreach ($colors as $value) {
                echo "$value <br>";
        }
        $fruits = array('apple', 'banana', 'orange', 'grapes');
        foreach ($fruits as $fruit){
                echo $fruit;
                echo "<br/>";
        }
?>
```

ဒီမှာဆိုရင် $colors ဆိုတဲ့ color array လေး ရှိပါတယ်။ ပထမအကြိမ် iterate လုပ်တဲအချိန်မှာ $value ထဲမှာ “red” ဆိုတဲ့ array ရဲ့ ပထမဆုံး element’s value ရှိပြီး ပထမအကြိမ် iterate လုပ်ပြီးတဲ့အချိန်မှာ $value ရဲ့ value က ပြောင်းသွားပါတယ်။ အဲ့ဒီလိုနဲ့ array ထဲက element’s value ကုန်သည်အထိ iterate လုပ် loop ပတ်သွားပါတယ်။

ဒုတိယ syntax က ဒီလိုပါ။

```
foreach($array as $key => $value){
   Code to be executed
}
```

သူက key/value အတွဲ ပုံစံနဲ့ array element တွေကို loop ပတ်ပါတယ်။

```
<?php
        $age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
        foreach($age as $x => $val) {
        echo "$x = $val<br>";
        }
?>
```

ဒီမှာဆိုရင် $age ဆိုတဲ့ key/value အတွဲ နဲ့ Associative Array ရှိပါတယ်။ array ရဲ့ element တွေကို key/value အတွဲ ပုံစံနဲ့ iterate လုပ်ချင်တဲ့အခါမျိုးမှာ ဒီပုံစံကို သုံးပါတယ်။

Breaking Out of the Loop

တစ်ခါတစ်လေမှာ loop မပြီးခင် ပြီးအောင်မပတ်ခင်မှာ loop ကို ရပ်ပစ်ချင်တာမျိုး ရှိပါတယ်။ အဲ့ဒီလို loop ပတ်နေရင်း ရပ်ဖို့ဆိုရင် break keyword ကို အသုံးပြုလို့ရပါတယ်။ break keyword ကို switch case statement မှာ သုံးပြပြီးဖြစ်ပါတယ်။ ဒီ loopတွေဖြစ်တဲ့ while, do-while, for, for တွေ မှာ အသုံးပြုလို့ရပါတယ်။

```
<?php
   $i = 0;
   while( $i < 10) {
      $i++;
      if( $i == 3 )break;
   }
   echo ("Loop stopped at i = $i" );
?>
```

ဒီမှာဆိုရင် loop ကို 3 ကြိမ်ပြီးတဲ့အခါမှာ break လုပ်လိုက်ပါတယ်။

```
<?php
for ($x = 0; $x < 10; $x++) {
  if ($x == 4) {
    break;
  }
  echo "The number is: $x <br>";
}
?>
```

ဒီမှာဆိုရင် loop ကို 4 ကြိမ်ပြီးတဲ့အခါ loop ကို break လုပ်လိုက်ပါတယ်။ အပေါ်ကနဲ့ အလုပ်လုပ်ပုံခြင်းတူပေမဲ့ ရေးပုံကွဲပါတယ်။

```
<?php
        $array = array( 1, 2, 3, 4, 5);
        foreach( $array as $value ) {
        if( $value == 3 )continue;
        echo "Value is $value <br />";
}
?>
<?php
          $x = 0;
          while($x < 10) {
          if ($x == 4) {
                    $x++;
          continue;
 }
           echo "The number is: $x <br>";
          $x++;
          }
?>
```

Continue က break out တစ်မျိုးပါပဲ။ ဒါပေမဲ့ သူရဲ့ အလုပ်လုပ်ပုံက break နဲ့ အလုပ်လုပ်ပုံ မတူပါဘူး။ loop ကို $x က 4 နဲ့ ညီတဲ့အချိန်မှာ continue က အဲ့ဒီ ညီတဲ့နေရာကို ကျော်တယ် break လုပ်ပါတယ်။ သို့ပေမဲ့ မရပ်ပဲ ဆက်လုပ်ပါတယ်။ ဒါကြောင့် ညီတဲ့နေရာကို ကျော်ပြီး loop ဆက်ဖြစ်ပါတယ်။

အခန်း(၅) - PHP Functions

What is a Function?

PHP function ဆိုတာ သီးခြားအလုပ်တစ်ခုခြင်းဆီကို လုပ်ဆောင်တဲ့ block of code တစ်ခုဖြစ်ပါတယ် ။ Functions တွေကို အဓိကသုံးရတဲ့ အကျိုးကျေးဇူးကတော့ code တွေကို reuse ပြန်လုပ်နိုင်ခြင်း ဖြစ်ပါတယ်။ Code ကိုတစ်ခါသတ်မှတ်ပြီးလျှင် အကြိမ်ကြိမ်ပြန်သုံးနိုင်ခြင်း ကို ဆိုလိုတာပါ။

Code တစ်ခုထဲကိုပဲအကြိမ်ကြိမ် ပြန်သုံးပေမဲ့ arguments တွေကို မတူအောင်ထည့်ပြီးတော့ ကွဲပြားတဲ့ result တွေကို အကြိမ်ကြိမ်ထုတ်နိုင်ပါတယ်။ PHP ရဲ့ အားသာချက်က built-in functions ပေါင်းများစွာပါပါတယ်။ php built-in function ဆိုတာ PHP script ထဲမှာ တိုက်ရိုက်ခေါ်နိုင်ပြီးတော့ ကိုလုပ်ချင်တဲ့အလုပ်တွေ လုပ်လို့ရပါတယ်။

Why Do We Use Functions

Function ကို အသုံးပြုခြင်းအားဖြင့် code duplication ကို လျော့ချပေးနိုင်ပါတယ် ။ နောက်ပြီး complex problems တွေကို ရိုးရှင်းတဲ့ pieces တွေ အဖြစ် decompsing လုပ်ပြီးလို့ မြင်လွယ်အောင် ရှင်းလို့ရပါတယ်။ နောက်ပြီး coding လုပ်တဲ့ အရည်အသွေးလည်း တိုးတက်လာနိုင်ပါတယ်။ နောက် ပြီး ကျနော် အပေါ်မှာ ပြောခဲ့သလိုပဲ code တွေက reusable ဖြစ်ပါတယ်။
အခြေခံအားဖြင့်ဆိုရင် functions type နှစ်မျိုးရှိပါတယ်။ built-in function နဲ့ user defined function ဆိုပြီးတော့ပါ။ built-in functions တွေဆိုတာက PHP language ရဲ့ အစိတ်ပိုင်းတွေလို့လည်းပြောလို့ရပါတယ်။ အပေါ်မှာ built-in function အကြောင်းကို အနည်းငယ် ရှင်းပြခဲ့ပြီး ဖြစ်ပါတယ်။ ဥပမာ - count() , strlen() , ...
User defined function ဆိုတာက ကိုယ်တိုင် ကိုယ်ပိုင်တည်ဆောက်ပြီး အသုံးပြုတဲ့ function တွေကို ဆိုလိုတာပါ။

PHP User-Defined Functions

PHP မှာ built-in functions ပေါင်း တစ်ထောင် ကျော်ပါသလို ကိုတိုင်လည်း custom functions တွေ တည်ဆောက်လို့ရပါတယ်။ function တစ်ခုသတ်မှတ်တော့ မည်ဆိုလျှင် function ဆိုတဲ့ keyword သုံ ပြီး သူ့ နောက်မှာ function name လိုက်ပါတယ် ။ ပြီးတော့ parentheses ဆိုတဲ့ () လေးလိုက်ပါတယ်။ နောက်ပြီး function ရဲ့ လုပ်ဆောင်ချက်တွေကို { } ထဲ ရေးပါတယ်။

```
function functionName(){
        //code to be executed
}
```

function တစ်ခု အလုပ်လုပ်ဖို့ execute ဖြစ်ဖို့ဆိုရင် ပြန် ခေါ် ဖို့လိုပါတယ် ။ function ကိုပြန်ခေါ် ဖို့ ဆိုရင် function name ကို စပြီး ရေးရပါတယ် ။ ပြီးမှ argument တွေ ကို parentheses ထဲ ထည့်ရေးရပါတယ်။

```
<?php
    function message() {
        echo "Hello world!";
    }
    message(); // call the function
?>
```

ဒီမှာဆို “Hello World” ဆိုပြီး ထွက်လာပါလိမ့်မယ်။
မှတ်စရာရှိပါတယ်။ function name ကို letter or underscore နဲ့ စလို့ရပါတယ်။ number or အခြား characters တွေနဲ့ စလို့မရပါဘူး။ နောက်ပြီး function names တွေက case-insensitive ပါ။

Functions with Parameters

Parameters ဆိုတာ name listed တွေ ဖြစ်ပြီး function ရဲ့ runtime မှာထည့်ပေးရတဲ့ input values တွေပါ။

```
function myFunc($oneParameter, $anotherParameter){
  // Code to be executed
}
```

parameter ဆိုတာတကယ်တော့ variables တွေလိုပါပဲ name လေးတွေပေးပြီး parentheses () ထဲမှာ ‘,’ ခြားပြီး ထည့်ရပါတယ်။ အဲ့ဒီလို parameters တွေကို name တွေနဲ့ သတ်မှတ်ပြီးရင် သူတို့ကို function အထဲမှာ ပြန်ထည့်သုံးနိုင်ပါတယ်။ parameters တွေကို အများကြီး ထည့်ပေးလို့ရပါတယ်။ အရေးကြီးတာက function call လုပ်တဲ့အချိန်မှာ ကို သတ်မှတ်ထားတဲ့ parameters တွေကို values (arguments) တွေ အဖြစ် ပြန်ထည့်ပေးဖို့ပဲလိုပါတယ်။

```
<?php
        // Defining function
        function getSum($num1, $num2){
        $sum = $num1 + $num2;
        echo "Sum of $num1 and $num2 is : $sum";
        }
        // Calling function
        getSum(10, 20);
?>
```

Arguments ဆိုတာ values တွေဖြစ်ပြီးတော့ function တည်ဆောက်တုန်းက function ရဲ့ parameters တွေဖြစ်ပါတယ်။ function ကို ပြန် call လုပ်တဲ့အချိန်မှာ parameters နေရာမှာ ထည့် ပေးရတဲ့ value ကို arguments လို့ ဆိုလိုတာပါ။ ဒါက usage terms တွေပဲ ဖြစ်ပါတယ်။ ဒါကြောင့် arguments ကလည်း parameters ၊ parameters က လည်း arguments တွေဖြစ်ပါတယ်။

အထက်မှာ ဖော်ပြခဲ့သည့်အတိုင်း PHP က Loosely Typed Language ဖြစ်တာကြောင့် variable ရဲ့ data type တွေကို value အရ auto associate လုပ်ပါတယ်။ data types တွေက strict sense မဟုတ်တာကြောင့် string နဲ့ integer ပေါင်းလို့ရပြီး error မတက်မပါဘူး။

PHP 7 မှာ types declarations ပါလာပါပြီ ။ function တွေ declaration လုပ်တဲ့အချိန်မှာ strict declarations ပေါင်းထည့်ခဲ့မယ်ဆိုလျှင် data types မတူတာပေါင်းတဲအခါ data type mismatches တွေဖြစ်ပြီး “Fatal Error” ပြပါလိမ့်မယ်။

```
<?php
        function addNumbers(int $a, int $b) {
          return $a + $b;
        }
        echo addNumbers(5, "5 days");
?>
```

ဒီမှာ strict declaration မလုပ်ထားတဲ့အတွက် 5 နဲ့ “5 days” နဲ့ပေါင်းတဲ့အခါ “5 days” က 5 ပြောင်းသွားပြီး ပေါင်း လဒ် 10 ရပါတယ်။ နောက်ထပ် ဥပမာ တစ်ခုထပ်ပြပါ့မယ်။

```
<?php declare(strict_types=1); // strict requirement
        function addNumbers(int $a, int $b) {
                return $a + $b;
        }
        echo addNumbers(5, "5 days");
?>
```

ဒီမှာဆိုရင် strict ဖြစ်ဖို့အတွက် PHP script ရဲ့ အပေါ်ဆုံးမှာ declare(strict_types=1) လို့ ရေးထားပါတယ်။ ဒါကြောင့် $b ရှေ့မှာ int လို့ strict type declaration လုပ်ထားတာကြောင့် “5 days” ဆိုပြီး argument ထည့်လိုက်တဲ့အခါ “5 days” က integer data type မဟုတ်တာကြောင့် Fatal Error တက်ပါတယ်။

Returning Values from a Function

Functions တွေက return statement ကို အသုံးပြုပြီးတော့ values တွေကို return ပြန်ပေးနိုင်ပါတယ်။ data types အမျိုးမျိုးကို return ပြန်နိုင်ပါတယ်။ ဆိုလိုတာက integer, string မှ မဟုတ် array, objects စတာတွေလည်း return ပြန်နိုင်ပါတယ်။

```
<?php
        function getSum($num1, $num2){
                $total = $num1 + $num2;
                return $total;
        }
        echo getSum(5, 10); // Outputs: 15
?>
```

Function ကို return statement ထည့်တဲ့အခါ function ကို ပြန် call ရင် function က data type တစ်ခုအနေပဲရပါတယ်။ ဒါကြောင့် function ကြောင့် ရတဲ့ retrun value ကို echo နဲ့ output ပြန်ထုတ်ထားတာဖြစ်ပါတယ်။

PHP 7 မှာ function argument တွေကို strict requirement လုပ်ပြီး type declaration လုပ်သလို function return ကို လည်း type declaration ထည်းလို့ရပါတယ်။ ဆိုလိုတာက return ပြန်ပေးမည့် value က string or integer လားဆိုတာကို သတ်မှတ်ပေးတာပါ။

```
<?php
        declare(strict_types=1); // strict requirement
        function addNumbers(float $a, float $b) : float {
                return $a + $b;
        }
        echo addNumbers(1.2, 5.2);
```

```
?>
```

ဒီမှာဆို ပေါင်းလို့ ရတဲ့ result ကို float data type နဲ့ return ပြန်မည် type declaration လုပ်ထားတာကြောင့် 6.4 ဆိုပြီး float value ပြန်ရပါတယ်။ အကယ်၍ float value နှစ်ခုကို ပေါင်းပြီး int ဆိုပြီး type declartion လုပ်ထားရင် integer အဖြစ်ပြန်ပါတယ်။

```
<?php
	declare(strict_types=1); // strict requirement
	function addNumbers(float $a, float $b) : int {
		return $a + $b;
	}
	echo addNumbers(1.2, 5.2);
?>
```

ဒီမှာ float values နှစ်ခုပေါင်းသော်လည်း return ပြန်မည့် type ကို int လို့ပြောထားတကြောင့် ပေါင်းလဒ် က 6 ဆိုပြီး integer value ပဲရပါတယ်။

Passing Arguments by Reference

Function တွေကို argument နှစ်မျိုးထည့်ပေးလို့ရပါတယ်။ value ရယ် referenc ရယ် ပါ။ များသောအားဖြင် function တွေမှာ argument ထည့်ပေးရင် values တွေပဲထည့်ပေးတာ များပါတယ်။ အဲ့ဒီလို values တွေထည့်ပေးတဲ့အခါ အဲ့ဒီ value ကို copy ယူပြီးတော့ function ထဲမှာ အလုပ်လုပ်ပါတယ်။ ဆိုလိုတာ က variable တစ်ခု function အပြင်မှာ တည်ဆောက်ထားတဲ့ ဆိုပါစို့။ အဲ့ဒီ variable ကို functon ထဲမှာ argument အနေနဲ့ ထည့်ပေးတဲ့အခါ မှာ အဲ့ဒီ variable ကို copy ယူပြီး function ထဲမှာ အလုပ်လုပ်ပါတယ်။ function outside က variable က ပြောင်းလည်းမှုမရှိတာကို ဆိုလိုတာပါ။

အကယ်၍ function ရဲ့ argument အဖြစ် reference ကို ထည့်ပေးမယ်ဆိုရင် ထည့်ပေးခြင်းခံရတဲ့ variable က function အတွင်းမှာ ပြောင်းရင် အပြင်မှာ လည်းပြောင်းသွားပါတယ်။ function argument ကို reference အနေနဲ့ပြောင်းထည့်ပေးစေချင်တဲ့အခါ argument အရှေ့မှာ ampersand (&) ထည့်ပေးရပါတယ်။

```
<?php
        function selfMultiply(&$number){
                $number *= $number;
                return $number;
        }
        $mynum = 5;
        echo $mynum; // Outputs: 5
        selfMultiply($mynum);
        echo $mynum; // Outputs: 25
?>
```

ဒီမှာဆိုရင် $mynum ကို function အပြင်ဖက်မှာ 5 လို့ value assign ထားပြီး function ကို argument အနေနဲ့ reference အနေနဲ့ ထည့်ပေးလိုက်တာပါတယ်။ ဒါကြောင့် function အတွင်းမှာ variable ကို ပြောင်းတဲ့အခါ argument အနေနဲ့ ထည့်ပေးလိုက်တဲ့ variable က အပြင်မှာလည်း ပြောင်းသွားပါတယ်။

Variable functions

PHP Documentation က တော့ ဒီလိုဆိုထားပါတယ် ။

PHP supports the concept of variable functions.

ဒါက ဘာကိုဆိုလိုတာလဲဆိုတော့ variable တစ်ခုကို parentheses (()) တပ်ပြီးတော့ function call ပုံစံပြန်သုံးလို့ရတယ်လို့ဆိုလိုတာပါ။ ဒီပုံစံကို သုံးနေ ကျမဟုတ်ပေမဲ့ variable ရဲ့ value assign ပေါ် အခြေခံတဲ့ dynamic function call တွေအတွက် များစွာ အထောက်ကူပြုပါတယ်။
ဥပမာတွေနဲ့ ရှင်းပြ ပါ့မယ်။

```
<?php
	// Variable Functions
	function red() {
		echo "color is red";
	}
	function blue() {
		echo "color is blue";
	}
	$color = "red";
	$color();
	$color = "blue";
	$color();
?>
```

ဒီမှာဆိုရင် အသေးစိတ်လေးကြည့်ရအောင်။

ပထမဆုံး PHP က red() function နဲ့ blue() function နဲ့ declarations ကို အရင် တွေ့ပါလိမ့်မယ်။ နောက်တစ်ခုအနေနဲ့ $color ထဲမှာ red ဆိုတဲ့ value assign ထည့်လိုက်တယ်လို့ပဲသိမှာပါ။ နောက်တစ်ကြောင်းမှာ $color ကို parentheses append လုပ်လိုက်ပါတယ် ။

အခဲ့ဒီ အချိန်မှာ PHP က red() function ကို $color() ထဲ replace လုပ်လိုက်ပါတယ်။ ဒါကြောင့် $color() ကို function ပုံစံ call() လုပ်တဲ့အချိန်မှာ red() fuction ရဲ့ အလုပ်ကို လုပ်ပေးပါတယ်။ နောက်တစ်ခု အနေနဲ့လည်း $color ကို blue ဆိုတဲ့ value assign ထည့်လိုက်တယ်လို့ပဲ မြင်မှာပါ။ နောက်တစ်ကြောင်းမှာ $color ကို function ပုံစံ parentheses () နဲ့ call လုပ်လိုက်တဲအချိန်မှာ blue() function ရဲ့ အလုပ်တွေက $color() ထဲကို replace ဖြစ်သွားပြီး function ရဲ့ အလုပ်လုပ်ကို လုပ်ပါတယ်။

Anonymous Functions

Anonymous functions သို့မဟုတ် Lambda က PHP ရဲ့ features တစ်ခုဖြစ်ပြီးတော့ သူကို အမြဲတန်းအသုံးမပြုပေမဲ့ တစ်ချို့အခြေနေတွေမှာ အရမ်း အသုံးဝင်ပါတယ် ။ဥပမာ function တစ်ခုကို အခြား function တစ်ခု ရဲ့ parameters တစ်ခုအဖြစ် ထည့်လိုက်တာမျိုး function scope အပြင်ဖက်က variable ကို access လုပ်တာမျိုး တွေမှာ အရမ်းအသုံးဝင်ပါတယ်။

Anonymous function တွေကို ဘာကြောင့်အသုံးပြုသင့်သလဲ

Anonymous function တွေကို တစ်ကြိမ်တစ်ခါ အသုံးပြုရန်သုံးတဲ့အခါ မှာ အရမ်းအသုံးဝင်ပါတယ် ။ တစ်ခါတစ်လေမှာ function နဲ့ အလုပ်လုပ်ချင်တယ် ဒါပေမဲ့ အခဲ့ဒီ function ကို function ကြီး တစ်ခုအနေနဲ့လည်း သက်သက်မထားချင်ဘူး အမြဲတန်းကြီးလည်း အသုံးမပြုချင်ဘူး ။ အခဲ့ဒီလိုအခြေနေတွေမှာ function တစ်ခုကို တည်ဆောက်မဲ့အစား တစ်ခါပဲ အသုံးပြုဖို့အတွက် anonymous function or lambda ကို အသုံးပြုလို့ရပါတယ်။

Anonymous function က regular function တွေနဲ့ ဆင်ပါတယ်။ regular function တွေလိုပဲ code တွေကို curly brackets ထဲမှာ ရေးမယ်။ arguments တွေ pass လုပ်မယ်။ values တွေ return ပြန်ပေးမယ်။ ပုံမှန် function တစ်ခု လုပ်သလို အကုန်လုပ်လို့ရပါတယ်။

သူတို့ နှစ်ခုရဲ့ အဓိက ကွာခြားချက်က anonymous function က သူ့မှာ name မရှိပါဘူး ။ ဒီ anonymous function example လေးကို ကြည့်ကြည့်ပါ။

```
function($argument1, $argument2){
//anonymous function definition goes here
};
```

သေခြာကြည့်ကြည့်မယ်ဆိုရင် အဓိက ကွာခြားချက် နှစ်ချက်ရှိပါတယ်။ anonymous function က name မရှိဘူး။ ပြီးမှာ anonymous function defined လုပ်ပြီးတဲ့ အချိန်မှာ semicolon(;) ထည့်ရပါတယ်။ ဘာကြောင့်လဲဆိုတော့ anonymous function defination က expression ဖြစ်ပေမဲ့ regular function ကတော့ code constructs ပုံစံဖြစ်တာကြောင့် ဖြစ်ပါတယ်။
အပေါ်က example အတိုင်း anonymous function ကို defined လုပ်ထားပေမဲ့ သူ့ကို အသုံးပြုလို့ မရသေးပါဘူး။ အဓိကက သူ့မှာ name မရှိအတွက်ကြောင့် function call လုပ်လို့မရသေးပါဘူး။
Anonymous function နဲ့အတူ ဒါတွေ လုပ်နိုင်ပါတယ်။
ပထမ အချက်က သူ့ကို variable တစ်ခုထဲကို assign ထည့်ပြီး အဲ့ဒီ variable name ကို function call ပုံစံခေါ် ပြီး အသုံးပြုပါတယ်။
နောက်တစ်ချက်က သူ့ကို အခြား function ထဲမှာ function parameters အဖြစ် အသုံးပြုလို့ရပါတယ်။ အဲ့ဒီလို function တစ်ခုကို function တစ်ခုရဲ့ parameter အဖြစ်သုံးတဲ့ပုံစံကို callback လို့ခေါ်ပါတယ်။
နောက်ထပ်တစ်ချက်က ကျနော်အပေါ်မှာ ဖော်ပြခဲ့တဲ့အတိုင်းပဲ function scope ထဲကနေ outer variable ကို access လုပ်နိုင်ပြီးတော့ အဲ့ဒီ function ကနေပဲ return ပြန်ပေးနိုင်ပါတယ်။ အဲ့ဒီလို ပုံစံကို closure လို့ခေါ်ပါတယ်။

Anonymous function တစ်ခုကို assign လုပ်ခြင်း

Anonymous function တစ်ခု create လုပ်တဲ့အချိန်မှာ သူ့ကို variable တစ်ခုထဲမှာ အခြား value တွေလိုပဲ assign လုပ်လို့ရပါတယ်။

```
$addition=function($arg1,$arg2){
return 'sum = '.$arg1+$arg2;
};
```

ဒီလို assign လုပ်ပြီးတဲ့အချိန်မှာ variable ကို function call လုပ်သလိုခေါ်ကြည့်ပါ။

```
echo $addition(20,50); // sum = 70
```

ဒီလိုဆို output က sum= 70 ဆိုပြီးထွက်ပါလိမ့်မယ်။

နောက်ပြီး သူ့ကို multiple anonymous functions တွေကို array elements လုပ်ပြီး အသုံးပြုလို့ရပါတယ်။

```
$myarray = array(
            function(){
                    echo "this is 0th index function";
            },
            function(){
                    echo "this is 1st index function";
            },
            function(){
                    echo "this is 2nd index function";
            },
            function(){
                    echo "this is 3rd index function";
            },
            function(){
                    echo "this is 4th index function";
             }
);
```

ဒီလိုဆို array ထဲမှာ anonymous function တွေကို ထည့်ထားလိုက်ပါပြီ ။ သူတို့ကို ပြန်ခေါ်သုံးချင်တဲ့အခါမှာ ဒီလိုပြန်သုံးပါတယ်။

```
echo $myarray[3]() //output- this is 3rd index function
```

Anonymous function ကို Callback အဖြစ် အသုံးပြုခြင်း

Anonymous function ကို အများဆုံးအသုံးပြုတဲ့ပုံစံက inline callback function ပုံစံနဲ့ အသုံးပြုတာများပါတယ်။ callback function ဆိုတာက function တစ်ခုဖြစ်ပြီးတော့ အခြား function တစ်ခုရဲ့ argument အဖြစ် pass လုပ်ပါတယ်။ အဲ့ဒီ argument အဖြစ်လက်ခံတဲ့ function က callback function ကို access လုပ်ပြီး function အတွင်းမှာ ပြန်အသုံးပြုလို့ရပါတယ်။

PHP ရဲ့ built-in function တော်တော်များများက callback function ကို access လုပ်နိုင်ပြီးတော့ callback function တွေကို အသုံးပြုပြီးတော့ ကိုယ်ပိုင် function တွေအဖြစ်ဖန်တီးလို့ရပါတယ်။

ဒီမှာ array_map() function ကိုအသုံးပြုပြီးတော့ array ရဲ့ element တစ်ခုချင်းဆီကို callback function နဲ့ run ပါမယ်။

array_map() function အသုံးမပြုခင်မှာ array_map() function အကြောင်း နည်းနည်းပြောပြ ချင်ပါတယ်။ပုံမှန်အားဖြင့် array_map() function က arguments နှစ်ခုယူပါတယ်။ ပထမတစ်ခုက callback function ဖြစ်ပြီးတော့ ဒုတိယတစ်ခုက array ပါ။ array ရဲ့ element တွေကို iterate လုပ်တိုင်းမှာ callback function က array ရဲ့ element တစ်ခုချင်းဆီကို အကျိုးသက်ရောက်မှုရှိပါတယ်။ callback function ကသာ return ပြန်ရမယ်ဆိုရင် Modified လုပ်ထားတဲ့ array ရဲ့ value တွေကို return value တွေနဲ့ replace လုပ်ပါလိမ့်မယ်။ ဒီမှာ မှတ်စရာတစ်ခုရှိတာက array_map() က array ကို အကျိုးသက်ရောက်မှု မရှိပါဘူး။

```
//without the callback function
$num_array = array(1,2,3,4,5);
foreach ($num_array as $key => $value) {
$new_array[$key]=$value*$value;
}
print_r($new_array);
//Array([0] => 1 [1] => 4 [2] => 9 [3] => 16 [4] => 25)
```

ဒီ အပေါ်က code ကို run လိုက်ရင် ဒီ အတိုင်း အဖြေရပါလိမ့်မယ်။ ဒါက ရိုးရိုးအသုံးပြုပုံပါ။

```
//with regular callback function
function square($num){
return $num*$num;
}

$new_array=array_map('square', $num_array);
print_r($new_array);
```

ဒီမှာဆို ရိုးရိုး regular function ကိုအသုံးပြုပြီးတော့ ရေးထားပါတယ်။

```
$new_array = array_map(function($num){
return $num*$num;
}, $num_array);
print_r($new_array);
print_r($num_array);
```

ဒီမှာဆိုရင် anonymous function ကို callback အနေနဲ့ အသုံးပြုထားတဲ့ပုံစံပဲဖြစ်ပါတယ်။ ရလဒ်တွေအားလုံးတူတူပါပဲ ။ အသုံးပြုပုံရေးပုံကွဲသွားပါတယ်။ ပို့ရှင်းသွားပါတယ်။

အခန်း(၆) - Variable Scope

functions တွေကို အသုံးမပြုဘူးဆိုရင် variables တွေကို page ရဲ့ ကြိုက်တဲ့နေရာမှာ အသုံးပြုလို့ရပါတယ်။ functions တွေ ပါလာပြီးဆိုရင်တော့ အဲ့ဒါက မမှန်နိုင်တော့ ပါဘူး။ functions တွေကို သူတို့ရဲ့ variables တွေကို တစ်ခြားနေရာတွေက နေယူသုံးခွင့်မပြုပါဘူး။ ဆိုလိုတာက function တစ်ခုအတွင်းမှာ vairables တွေ တည်ဆောက်မယ် ဆိုရင် အဲ့ဒီ variable ကို function အပြင်ဖက်မှာ အသုံးပြုလို့မရပါဘူး။

နော်ကတစ်ခုအနေနဲ့ function outside မှာ ကြေညာထားတဲ့ variable ကိုလည်း function ထဲမှာ အသုံးပြုလို့ မရပါဘူး။

```
<?php
$a = 3;
function foo()
{
        $a += 2;
}
foo();
echo $a;
?>
```

ဒီမှာဆိုရင် foo() function အတွင်းမှာ ရှိတဲ့ $a variable က function အပြင်ဖက်မှာ ရှိတဲ့ $a variable နဲ့ မတူပါဘူး။ ဒီမှာဆိုရင် function foo() က add and assign operator ကို အသုံးပြုထားပေမဲ့ foo() function အပြင်ဖက်မှာ ရှိတဲ့ $a variable က value က 3 အဖြစ်ပဲကျန်နေပြီးတော့ function အတွင်းမှာ ရှိတဲ့ $a က value က 2 ဖြစ်နေပါလိမ့်မယ်။
ဒီမှာ ဘာကြောင့် အဲ့လို ဖြစ်တာလည်းဆိုတော့ variable ရဲ့ scope ကွာတာပါ။
Variable တစ်ခုရဲ့ scope ဆိုတာ variable ရဲ့ declaration လုပ်ထားတဲ့နေရာအရ သူ့ရဲ့ location အရ သူ့ကို ဘယ်သူကပဲ သုံးခွင့်ရှိတယ် ဘယ်သူကပဲ access လုပ်နိုင် တယ်ဆိုတာကို ဆုံးဖြတ်ပေးတာကို ဆိုလိုပါတယ်။ PHP မှာဆိုရင် variable scope လေးမျိုးရှိပါတယ်။ local,global,static နဲ့ function parameters ဆိုပြီးတော့ပါ။

Local scope

Function တစ်ခုအတွင်းမှာ declare လုပ်တဲ့ variable က သူ့ရဲ့ local က အဲ့ဒီ function အတွင်းပဲဖြစ်ပါတယ်။ ဆိုလိုတာက အဲ့ဒီ variable ကို အဲ့ဒီ function အတွင်းမှာ ပဲ အသုံးပြုလို့ရမယ် function အပြင်ဖက်ကနေ သူ့ကို access လုပ်လို့မရပါဘူး။ နောက်ထပ်အနေနဲ့ default အနေနဲ့ function အပြင်မှာ declare လုပ်တဲ့ variable ကို global variable လို့ခေါ်ပြီးတော့သူ့ကို function

အတွင်းမှာ အသုံးပြုလို့မရပါဘူး။

```
<?php
function updateCounter()
{
        $counter++;
}
$counter = 10;
updateCounter();
echo $counter;
?>
```

Function အတွင်းမှာ declare လုပ်ထားတဲ့ $counter က local ဖြစ်ပါတယ်။ သူ့ကို နေရာတိုင်းမှာအသုံးပြုလို့မရပါဘူး။ function က $counter ++ လုပ်ပြီးတော့ increment လုပ်ထားပါတယ်။ ဒါပေမဲ့ အပြင်မှာ ရှိတဲ့ ဆိုလိုတာက function outside က global variable $counter က တော့ 10 ပဲ ရှိနေပါလိမ့်မယ်။ ဘာကြောင့်လဲဆိုတော့ function က local scope ပဲ ဖန်တီးနိုင်လို့ပါ။ အခြား launguage တွေနဲ့ မတူတာက PHP မှာ loop တွေ condition တွေထဲသွားပြီးတော့ variable ရဲ့ scope သွားဖန်တီးလို့မရပါဘူး။

Global Scope

Variable ကို function outside မှာကြေညာရင် global ဖြစ်ပါတယ်။ ဆိုလိုတာက သူ့ကို program ရဲ့ တစ်ချို့နေရာတွေကနေ access လုပ်နိုင်တယ်ကို ဆိုလိုတာပါ။ ဒါပေမဲ့ default အနေနဲ့ သူ့တို့ကို function အတွင်းမှာ ပြန်သုံးလို့မရပါဘူး။ အကယ်၍ function အတွင်းမှာ global variable တွေကို access လုပ်ချင်တယ်ဆိုရင် function အတွင်းမှာ variable declare လုပ်တဲ့ အခါ global

keyword ကို အသုံးပြုပြီး declare လုပ်ပေးရပါတယ်။ ဒီလိုပါ။

```
<?php
function updateCounter()
{
        global $counter;
        $counter++;
}
$counter = 10;
updateCounter();
echo $counter; //11
?>
```

ဒီမှာဆိုရင် function က $counter variable ကို access လုပ်နိုင်ပြီးတော့ increment လုပ်လို့ရသွားပါတယ်။

နောက်ထပ်တစ်နည်း အနေနဲ့ variable ကို တိုက်ရိုက် access လုပ်မဲ့အစားPHP ရဲ့ $GLOBALS array ကို အသုံးပြုပြီးတော့ global variable အဖြစ် အသုံးပြုလို့ရပါတယ်။

```
<?php
function updateCounter()
{
$GLOBALS['counter']++;
}
$counter = 10;
updateCounter();
echo $counter; //11
?>
```

Static Variables

Static variable တစ်ခုက function call ခေါ်နေစဉ် အတွင်းမှာ သူ့ရဲ့ value ကို retains လုပ်ပါတယ်။ retains လုပ်တယ်ဆိုတာက variable တစ်ခုက သူ့ရဲ့ value ပြောင်းလည်မှုကို ထိန်းသိမ်းပြီး ထပ် reassign လုပ်တယ်ကိုဆိုလိုတာပါ။ static vairable ကို function အတွင်းမှာပဲ အသုံးပြုလို့ရပါတယ်။ static variable တစ်ခု ဖန်တီးခြင်းရင် static keyword ကိုအသုံးပြုရပါတယ်။

```
<?php
function updateCounter()
{
        static $counter = 0;
        $counter++;
        echo "Static counter is now {$counter}<br/>";
}
$counter = 10;
updateCounter();
updateCounter();
echo "Global counter is {$counter}";
Static counter is now 1
Static counter is now 2
Global counter is 10
?>
```

ဒီမှာဆိုရင် static variable $counter က function တစ်ခါခေါ်တိုင်း တစ်တိုးသွားပါတယ်။ function တစ်ခါ call ပြီးတိုင် သူ့ရဲ့ value က reset မဖြစ်ဘဲ အလုပ်ထပ်လုပ်ပါတယ်။ ဒါကို retain လုပ်တယ်လို့ခေါ်ပါတယ်။

Function parameters

```
function greet($name)
{
        echo "Hello, {$name}";
}
greet("Janet"); //Hello, Janet
```

Function ရဲ့ parameters တွေကလည်း local ပါပဲ ။ သူတို့ကို function အတွင်းမှာပဲ အသုံးပြုလို့ရပါတယ်။ ဒီမှာဆိုရင် $name ကို function အပြင်ဖက်ကနေ access လုပ်လို့မရပါဘူး။

Closure

Closure ဆိုတာ lambda နဲ့တူ ပါတယ်။ ပြီးတော့ closure က anonymous fuction တစ်ခုရဲ့ object representation လို့ဆိုလို့ရပါတယ်။

```
<?php
    $string = function(){
                    return "This is an anonymous function";
                };
        echo $string();
?>
//output: This is an anonymous function.
```

ဒီမှာဆိုရင် anonymous function တစ်ကယ် return ပြန်ပေးလိုက်တာက closure object တစ်ခုပါ။ အဲ့ဒါကို မှာ $string ထဲကို assign လုပ်ပြီး $string() ကို ပြန်ခေါ်လိုက်တာပါ။ ဒါကြောင့် closure က anonymous function နဲ့ ဆင်တူတယ် ။ anonymous function ကို object orient way နဲ့ အသုံးပြုခြင်းလို့ဆိုနိုင်ပါတယ်။

```
<?php
$anony = function () {
 return 'lambda';
};
//Closure
echo get_class($anony);
//object
echo gettype($anony);
?>
```

သူက လက်တွေ့မှာ function အပြင်ဖက် က variable သို့မဟုတ် သူ့ရဲ့ parameter မှာ မပါတဲ့ variable တွေကို access လုပ်ဖို့အတွက် အသုံးပြုပါတယ်။
ဥပမာ လေးနဲ့ ရှင်းပြပါ့မယ်။

```
//define a regular variable
$user='Peter';
//create a closure using an anonymous function
$message=function() use ($user){
echo 'hello'. $user;
}
$message();//output - hello Peter
```

ဒီမှာဆိုရင် $user က function အပြင်ဖက် မှာ declare လုပ်ထားပါတယ်။ ဒါကြောင့် ပုံမှအတိုင်းဆို ကို function scope အတွင်းကနေ access လုပ်လို့မရပါဘူး။
နောက်ထပ် နည်းနည်းလေးရှုပ်တဲ့ ပုံစံနဲ့ ရှင်းပြပါ့မယ်။

```
<?php
$fruit = [
    ['name' => 'Apple', 'color' => 'red'],
    ['name' => 'Banana', 'color' => 'yellow'],
    ['name' => 'Orange', 'color' => 'orange'],
];
$fruit_names = array_column($fruit, 'name');
print_r($fruit_names); //Array ( [0] => Apple [1] => Banana [2] => Orange )
?>
```

ဒီမှာဆိုရင် $fruit ဆိုတဲ့ multidimensional array ရှိတဲ့ သူ့ထဲက name array ကြည့် လိုချင်တဲ့အခါ array_columm()ဆို တဲ့ array function ကိုအသုံးပြုထားပါတယ်။ အကယ်၍ color ကြည့်လိုချင်တဲ့အခါ color ဆိုပြီး array_column() ထဲမှာ ထည့်လိုက်ရုံပါပဲ။

အဲ့ဒီလိုရဖို့အတွက် ကိုယ်ပိုင် function လေးတည်ဆောက်ကြည့်ရအောင် ။

```
<?php
$fruit = [
    ['name' => 'Apple', 'color' => 'red'],
    ['name' => 'Banana', 'color' => 'yellow'],
    ['name' => 'Orange', 'color' => 'orange'],
];
function taste($arr, $key)
{
    $result = array_map('', $arr);
    return $result;
}
$fruit_names= taste($fruit, 'name');
print_r($fruit_names);
```

ဒီမှာဆိုရင် taste ဆိုပြီး ကိုယ်ပိုင် function တစ်ခုတည်ဆောက်လိုက်မယ်။ အဲ့ထဲမှာမှ array_map() function သုံးပြီးတော့ taste() function ကရလာတဲ့ parameter or variable ဖြစ်တဲ့ $arr or array ကို loop ပတ်လိုက်မယ်။ ပြီးတော့ ရလာတဲ့ result ကို $result ထဲထည့်ပြီး return ပြန်လိုက်မယ်။ နောက်ပြီး သူ့ကို $fruit_names ထဲကို ထည့်လိုက်မယ်။ ဒါဆို taste() ကရလာတဲ့ return value က $fruit_names ကရပြီး output ထုတ်လိုက်မယ်။ ဒါက လုပ်ချင်တဲ့ပုံစံပါ။ ဒါက run ကြည့်ရင် အဖြေမထွက်သေးပါဘူး။

```
function taste($arr, $key)
{
    $result = array_map("return_name", $arr);
    return $result;
}
function return_name($item)
{
    return $item['name'];
}
$fruit_names= taste($fruit, 'name');
print_r($fruit_names);
```

taste() function အောက်မှာ နောက်ထပ် return_name() ဆိုတဲ့ function တည်ဆောက်လိုက်မယ်။ ပြီးတော့ သူ့ထဲကိုparameter တစ်ခု pass လိုက်မယ် ။ ပြီး တော့ အဲ့ ဒီ parameter ရဲ့ 'name' ကို ပဲ return ပြန်ပေးမယ်။

နောက်ပြီး အဲ့ဒီ return_name() ဆိုတဲ့ function ကို array_map() ထဲမှာ callback function အနေနဲ့ ထည့်လိုက်မယ်။ array_map() ရဲ့ first parameter မှာ သူပေးထားတဲ့ algo တွေနဲ့ အလုပ်လုပ်လို့ရသလို own function or callback function တွေခေါ်ပြီးတော့လည်း အလုပ်လုပ်လို့ရပါတယ်။ ဒီမှာဆို return_name() ဆိုတဲ့ function ကို callback အနေနဲ့ ခေါ်ပြီး အလုပ်လုပ်ထားပါတယ်။ သူလုပ်တဲ့ အလုပ်က ရလာတဲ့ arrray ထဲကမှာ array ရဲ့ name တွေကြည့် စုပြီး array တစ်ခု return ပြန်ပေးဖို့ပါ။ ဒီမှာဆိုရင် name တွေကြည့် စုထားတဲ့ array output ရပါလိမ့်မယ်။

ဒီမှာဆိုရင် taste() က parameter နှစ်ခုက တစ်ခုကိုပဲ အလုပ်လုပ်ထားပါတယ်။ $key ကို အလုပ်လုပ်မလုပ်ရသေးပါဘူး။ taste() ကို call ပြီး သူ့နေရာမှာ 'name'လို့ pass ပေးပေမဲ့ သူက တကယ် အလုပ် မလုပ်သေးပါဘူး။ တကယ်က သူ့နေရာမှာ 'color'လို့ ပေးရင် color array လိုချင်တာပါ။ အဲ့ဒီလို လုပ်လို့ရအောင် code ကို ဒီလိုလေးပြန်ပြင်လိုက်ပါမယ်။

```
function taste($arr, $key)
{
    $result = array_map(function ($item) {
        return $item['name'];
    }, $arr);
    return $result;
}
```

ဒီမှာဆိုရင် return_name() function ကို array_map() အတွင်းမှာ callback function အနေနဲ့ မခေါ်တဲ့ဘဲ တိုက်ရိုက် anonymous function အနေနဲ့ ပြန်ခေါ်လိုက်ပါတယ်။ anonymous function ဆိုတာ function name မရှိတဲ့ function ကိုဆိုလိုတာပါ။ ဒီတိုင်းရေးရင်လည်း ပုံမှန်တိုင်း အလုပ်လုပ်နေပြီးတော့ $key parameter က လွတ်နေပါအုံးမယ်။
$key ကို အသုံးပြု ဖို့အတွက် code ကို ဒီလိုလေးထပ်ပြင်ပါမယ်။

```
function taste($arr, $key)
{
    $result = array_map(function ($item) {
        return $item[$key];
    }, $arr);
    return $result;
}
```

ဒီမှာဆိုရင် anonymous function က $key အရ array output ပြန်ထုတ်ပေးမာဖြစ်ပါတယ်။

ဆိုလိုတာက $key နေရာမှာ 'name'ဆိုရင် name တွေကြည့်စုထားတဲ့ array ပြန်ပေးမှာဖြစ်ပြီး 'color' ဆိုရင် color တွေကြည့် စုထားတဲ့ array ပြန်ပေးမှာ ဖြစ်ပါတယ်။ ဒါပေမဲ့ ဒီလိုရေးပြီး run မယ်ဆိုရင် undefined variable:key and undefined index ဆိုပြီး error ပဲ ထွက် နေပါလိမ့်မယ်။ ဒီ ကျနော့်တို့ အသုံးပြုလိုက်တဲ့ $key က taste() function ရဲ့ scope အတွင်းမှာ မရှိဘဲနဲ့ anonymous function ရဲ့ scope အတွင်းမှာ ရှိနေပါတယ်။ ဒါဆိုရင် မေးစရာရှိပါတယ်။ သူ့ကို global scope variable ပြောင်းကြည့်မယ်။ output ထွက်နိုင်လား။

```
function taste($arr, $key)
{
    $result = array_map(function ($item) {
        global $key;
        return $item[$key];
    }, $arr);
    return $result;
}
```

ဒီလိုဆိုရင်လည်း undefined index ဆိုပြီး error တက်နေပါလိမ့်အုံးမယ်။ ဒါဆိုရင် အဲ့ဒီ သုံးချင်တဲ့ $key နဲ့ anonymous function ရဲ့ scope တူအောင် $key ကို import လုပ်ဖို့လိုပါတယ်။ ဒီလိုပါ။

```
function taste($arr, $key)
{
    $result = array_map(function ($item) use($key) {
        return $item[$key];
    }, $arr);
    return $result;
}
```

ဒီမှာဆိုရင် $key ကို အသုံးပြုချင်တဲ့အတွက် anonymous function နဲ့ $key နဲ့ scope တူအောင် $key

ကို အသုံးပြုလို့ရအောင် use() ကို သုံးပြီး import လုပ်ထားပါတယ်။ ဒီလိုဆိုရင်တော့ output ထွက်ပါလိမ့်မယ်။ name လို့ပေးရင် name array ရပြီး color လို့ပေးရင် color array ရပါမယ်။ code အပြည့်စုံက ဒီလိုပါ။

```
function taste($arr, $key)
{
    $result = array_map(function ($item) use ($key) {
        return $item[$key];
    }, $arr);
    return $result;
}

$fruit_name = taste($fruit, 'name');
print_r($fruit_name);
```

ဒါကြောင့် scope က သေခြာတွေးကြည့်ရင် နည်းနည်းလေး ရှုပ်ပါတယ်။ variable တစ်ခုကို local scope အတွင်းမှာ လိုချင်တဲ့အခါ အဲ့ဒီ vairable က ကိုယ်လိုချင်တဲ့ local scope အတွင်းမှာ မရှိတဲ့အခါ တစ်းနည်းအားဖြင့် scope မတူတဲ့အခါ အသုံးပြုလို့ရအောင် တစ်နည်းနည်းနဲ့ import လုပ်ရပါတယ်။ global variable ဖြစ်တဲ့အခါ ၊ အဲ့ ဒီ global variable ကို အသုံးပြုချင်တဲ့အခါ function အတွင်းမှာ global ဆိုတဲ့ keyword နဲ့ global ဖြစ်ကြောင်း ကြေညာပေးရုံပါပဲ။ ဒါပေမဲ့ global variable မဖြစ်တဲ့အခါ အပေါ်က taste() function မှာကြည့်မယ်ဆိုရင် $key နဲ့ anonymous function က scope level တူတာကြောင့် $key ကို anonymous function ထဲမှာအသုံးပြုချင်တဲ့အတွက် import လုပ်ရတယ်။ $key က နဂိုကတည်းက global variable မဟုတ်တာကြောင့် global keyword ကို အသုံးပြုလို့မရပါဘူး ။

Arrow functions in PHP

Array function က PHP 7.4 မှာ ပါလာတဲ့ feature တစ်ခုဖြစ်ပါတယ်။ သူက anonymous function နဲ့ shorthand လို့ ပြောပေမဲ့ ကွားခြားချက်တွေရှိပါတယ်။ anonymous function ကော arrow function ကော နှစ်ခုလုံးက closure နဲ့ implement ဖြစ်ပါတယ်။

Array function ရဲ့ syntax က ဒီလိုပါ။

```
fn (argument_list) => expr
```

```
$posts = [ /* … */ ];
$ids = array_map(function ($post) {
    return $post->id;
}, $posts);
```

ပုံမှန်တိုင်း array ရဲ့ id တွေ လိုချင်တဲ့အခါ မှာ ဒီလိုရေးပါတယ်။ ဒါက arrow function နဲ့ ရေးကြည့်ပါမယ်။

```
        $ids = array_map(fn($post) => $post->id, $posts);
```

ဒီလိုဖြစ်သွားပါတယ်။

သူ့ကို စပြီး define လုပ်ရင် fn keyword နဲ့ defined လုပ်ပါတယ်။ မှတ်ထားရမှာတစ်ခုက anonymous function နဲ့ မတူတာက သူ့မှာ expression တစ်ခုပဲပါပါတယ်။ အဲ့ဒီ expression ကလည်း return statement ဖြစ်ပါတယ်။ ဒါပေမဲ့ return ပြန်ပါတယ်ဆိုပြီး return keyword ကို အသုံးပြုစရာမလိုပါဘူး။ နောက်ထပ် closure ပုံစံမှာ anonymous function နဲ့ မတူတဲ့အချက်ကို ထပ်ပြောပြပါ့မယ်။

```
<?php
        $factor = 10;
        $numbers = array_map(function($value) use ($factor){
                return $value * $factor;
```

```
        }, [1, 2, 3]);
        print_r($numbers);
        //Array([0] => 10 [1] => 20 [2] => 30)
?>
```

ဒီမှာ function က closure ဖြစ်ဖို့ use() clause ကိုအသုံးပြုထားပါတယ်။ အဲ့ဒီအခါမှာမှ scope ရဲ့ outer က variable ကို access လုပ်လို့ရပါတယ်။

Arrow syntax နဲ့ကျ ဒီလိုဖြစ်သွားပါတယ်။

```
<?php
$factor = 10;
$numbers = array_map(fn($value) => $value * $factor, [1, 2, 3]);
print_r($numbers);
//Array([0] => 10 [1] => 20 [2] => 30)
?>
```

ဒီမှာဆို arrow function က closure ဖြစ်ဖို့အတွက် $factor ကို ဒီတိုင်း ထည့်သုံးလို့ရပါတယ်။ ဆိုလိုတာက arrow function က scope နဲ့ binding မဖြစ်ပါဘူး ။ ဆိုလာတာက scope ကန်သန့်ချက် မရှိပါဘူး။

အခန်း(၇) - Web Techniques

PHP က web-scripting language အဖြစ် design လုပ်ထားတဲ့ language ဖြစ်ပါတယ်။ သို့ပေမဲ့သူ့ကို command-line နဲ့ GUI scripts တွေကနေလည်း အသုံးပြုလို့ရပါတယ်။ ပုံမှန် အားဖြင့်ဆိုရင် dynamic website တွေမှာ forms တွေ sessions တွေ redriection တွေ ပါတက်ပါတယ်။ အဲ့ဒီ elements တွေကို PHP နဲ့ဘယ်လို implements လုပ်မလဲဆိုတာကို ဒီ အပိုင်းက နေ စပြီးတော့ရှင်းပြပါ့မယ်။ ဒီအပိုင်းမှာ PHP ကနေ form paramters တွေ access လုပ်ပုံ files တွေကို uploaded လုပ်ပုံ နောက်ပြီး cookies တွေ ဘယ်လို set လုပ်ရတယ် ဘယ်လို send ပြီးတော့ web browsers တွေကို redirect လုပ်ရမယ်ဆိုတဲ့ အကြောင်းတွေ ပါပါတယ်။ နောက်တစ်ခုအနေနဲ့ PHP sessions ကို ဘယ်လို အသုံးပြုရမလဲဆိုတာပါ ပါပါတယ်။

Variables

Form parameters တွေနဲ့ cookies တွေပါတဲ့ server configuration နဲ့ request information တွေကို PHP scripts ကနေ access လုပ်လို့ နည်းသုံးနည်းရှိပါတယ်။ အဲ့ဒါကို ချုံလိုက်မယ်ဆိုရင် EGPCS(environment, GET, POST, cookies, and server) ဆိုပြီး ရပါတယ်။
ဒါတွေကို PHP ရဲ့ six global array လို့ ခေါ်ပါတယ်။

$_ENV(environment)

ဒီမှာ environment variables ရဲ့ value တွေပါဝင်ပါတယ်။ environment variables ရဲ့ name တွေက array ရဲ့ keys တွေဖြစ်ပါတယ်။

$_GET(GET)

GET request ရဲ့ အပိုင်းဖြစ်တဲ့ parameters တွေပါပါတယ်။ ဒီမှာ array ရဲ့ keys တွေက form parameters ရဲ့ name တွေဖြစ်ပါတယ်။

$_COOKIE

ဒီမှာ request ရဲ့ အပိုင်းဖြစ်တဲ့ passed cookie values တွေပါပါတယ်။ ဒီမှာ array ရဲ့ keys တွေက cookies ရဲ့ names တွေဖြစ်ပါတယ်။

$_POST(POST)

ဒီမှာ POST request ရဲ့ အပိုင်းဖြစ်တဲ့ parameters တွေပါပါတယ်။ ဒီမှာ array ရဲ့ keys တွေက form parameters ရဲ့ names တွေဖြစ်ပါတယ်။

$_SERVER

ဒီမှာ web server ရဲ့ အသုံးဝင်တဲ့ information တွေပါပါတယ်။

$_FILES

ဒီမှာ files uploaded ထားတဲ့ information တွေပါပါတယ်။
ဒီ variable တွေအကုန်လုံးက global သက်သက် မဟုတ်ပါဘူး။ သူ့ကို function definitions ထဲမှာပါ

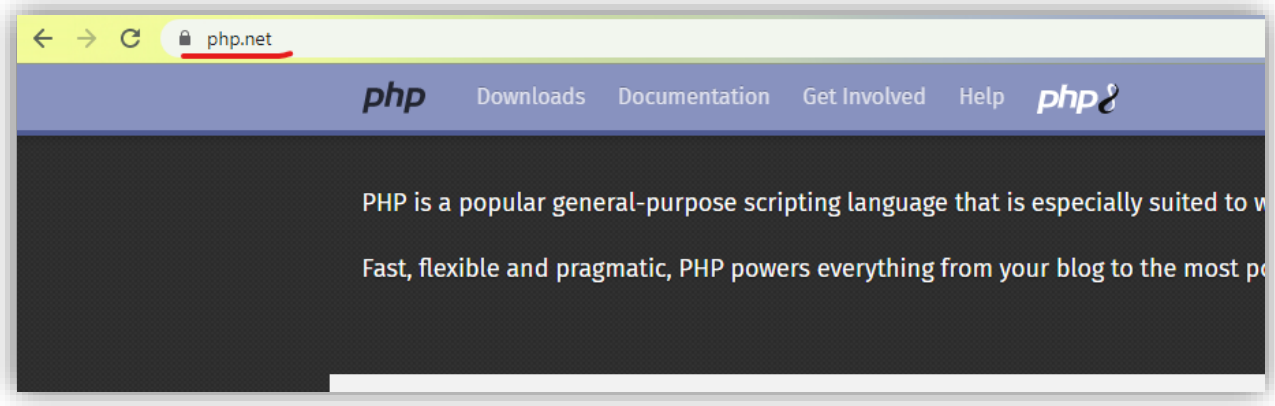
အသုံးပြုလို့ရပါတယ်။ နောက်ပြီး PHP က $_GET တို့ $_POST တို့ အလုပ်လုပ်ရင် auto-create လုပ်တဲ့ array ရှိပါတယ်။ $_REQUEST array ဖြစ်ပါတယ်။ သူ့ထဲမှာ $_GET,$_POST နဲ့ $_COOKIE elements တို့ ပါဝင်ပါတယ်။ ဒါတွေကိုလည်း သေခြာသိထားဖို့လိုပါတယ်။ နောက်ပိုင်းတွေမှာ ဒီ variables တွေကို အသုံးပြုပြီး အလုပ်လုပ်ရမှာ ဖြစ်ပါတယ်။

အခန်း(၈) - Request

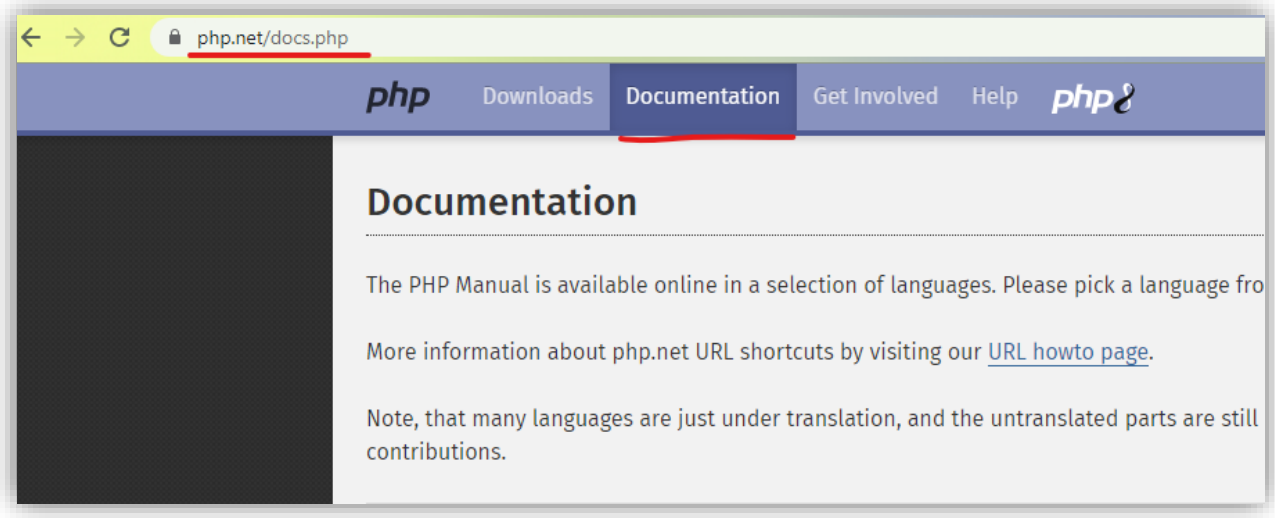
Web application တွေ သာမက application တွေတော်တော်များများ မှာ အရေးကြီးတဲ့ အရာတစ်ခုက user input ကို handling လုပ်ခြင်းပါပဲ။ ဘာကြောင့်လဲဆိုတော့ application က ဘာကို ဆိုလိုတယ်ဆိုတာ သိစေလို့ပါ။ ကျနော် တို့ က front end strong တဲ့ website တစ်ခုရေးတယ်ဆိုရင် users တွေကို correct products တွေကို သေခြာပြဖို့အတွက် user input တွေကို သေခြာကိုင်တွယ်နိုင်ဖို့လိုပါတယ်။ blogs site တစ်ခုရေးတယ်ဆိုပါစို့ ဒါဆိုရင် blogs နဲ့ ဆိုင်တဲ့ etries တွေ things တွေကို သေခြာပြဖို့အတွက် user inputs တွေကို သေခြာ handle လုပ်နိုင်ရပါမယ်။ Form input fields တွေကလာတဲ့ တစ်နည်းအာဖြင့် form parameters တွေကို handle လုပ်ဖို့ လိုပါတယ်။ form parameters တွေ applications တွေကို handling လုပ်တယ်ဆိုတာကို request တွေကို handling လုပ်နိုင်တာကို ဆိုလိုတာပါ။

GET request

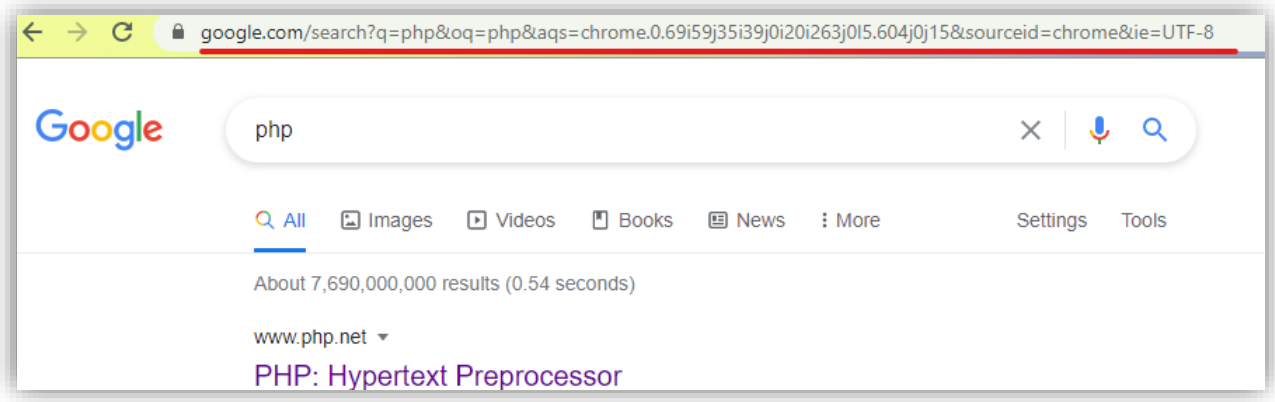
GET request က web applications တွေမှာ အများဆုံးအသုံးပြုတဲ့ common requests ဖြစ်ပါတယ်။ browser ထဲမှာ လုပ်တာမှန်သမျှက GET request လုပ်တာနေပါပဲ။ ဒါကြောင့် address bar မှာ တစ်ခုခုရိုက်ပြီး Enter နှိပ်လိုက်ရင် GET request တစ်ခု လုပ်လိုက်တာပါပဲ။ နောက်ပြီး GET request တကယ်လုပ်တာက လိုချင်တဲ့ information တွေကို ထုတ်ပေးတာပါပဲ။ ဒါဆိုရင် php.net ကို သွားပြီး request လုပ်ကြည့်ရအောင် ။



browser ကနေ php.net ကို request လုပ်လိုက်တဲ့အခါ server ကနေ ပုံအတိုင်း respond ပြန်ပေးပါတယ်။



ဒီမှာဆိုရင်လည်း နော်ထပ် request တစ်ခုအနေနဲ့ documentaion ကို request လုပ်လိုက်ပါတယ်။ ဒါကလည်း GET request တစ်ခု လုပ်လိုက်တာပါပဲ။ အဲ့ဒီအခါ server ကနေ လုပ်တဲ့ request ကို respond ပြန်ပါတယ်။ ဒါကြောင့် GET requests က အရမ်း ရိုးရှင်းပါတယ်။ GET request ရဲ့ အဓိက အပိုင်းကတော့ query string ကနေ information တွေရယူတဲ့အပိုင်းပါပဲ။



ဒါဆို ကနော်တို့ google ကိုသွားပြီး PHP ဆိုပြီး ရှာကြည့်ရအောင် ။ အဲ့ဒီမှာ address bar ကို ကြည့်မယ်ဆိုရင် ပထမဆုံး www.google.com ကိုတွေ့လိမ့်မယ်။ ပြီးတော့ / search ဆိုတဲ့ လုပ်လိုက်တဲ့ request လုပ်လိုက်တဲ့ resource ကို ပြပေးလိမ့်မယ်။ ပြီးတော့ ? ကိုတွေ့ပါလိမ့်မယ် ။ အဲ့ဒီ ? နောက်က အားလုံးကို query string လို့ခေါ် ပါတယ်။ query string မှာဆိုရင် တော်တော်များများက web application က လုပ်တဲ့ key value အတွဲတွေ ပါတက်ပါတယ်။ ဒီမှာဆို source က hp ပြီးတော့ q = php ။ q ဆိုတာက ဒီ Google's web serve ဒီ q ဆိုတဲ့ query parameter ကို search လုပ်တာဖြစ်ပါတယ်။ အဲ့ဒီ search result ကို မြင်နေရတာဖြစ်ပါတယ်။ ဒါကြောင့် Google' web server က ဘာကို ရှာရမယ်သိသွားတာပေါ့။ ဆိုလိုတာက ဒီ q ကို သူတို့ရဲ့ servers ကိုပို့လိုက်ပါတယ်။ ဒါကြောင့် q မှာ ပါတဲ့ PHP ကို ရှာပေးတာဖြစ်ပါတယ်။ application တွေက query parameters တွေကို default အနေနဲ့ မရှာပါဘူး ။

ဒါဆို query string ကနေ GET request နဲ့ data ယူကြည့်ရအောင်။

```php
<?php
$categoryid = $_GET['categoryid'];
?>
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>GET request</title>
</head>

<body>
  <div>
    <h1>GET request</h1>
    <p>
      <?=$categoryid;?>
    </p>
  </div>
</body>

</html>
```

Php file ထဲမှာ ဒီလိုရေးထားပါမယ် ။ ပြီးတော့ သူ့ကို run လိုက်မယ် ။ error တက်ပါလိမ့်မယ်။ အစမှာ query string မှာ မထည့်ထားတဲ့ categoryid ကို request ယူထားလို့ပါ။ ဒီ code လေးကို ပြန်ရှင်းမယ်ဆိုရင် query string ကလာမဲ့ categoryid ကို ကို get request နဲ့ ယူမှာဖြစ်တဲ့အတွက် $_GET[] ဆိုတဲ့ super global array variable ကို သုံးထားပါတယ်။ ပြီးတော့ HTML ထဲမှာ output ပြန်ထုတ်ထားပါတယ်။ ဒါကြောင့် url bar မှာ ဒီလိုပေးတဲ့အခါမှာ

```
http://localhost:4000/part_6/getRequest.php?categoryid=10
```

10 ဆိုပြီး output ရပါလိမ့်မယ်။ $_GET[] မှာ url bar ကနေပေးတဲ့ query string ရဲ့ key အားလုံးရှိပါတယ်။ ဒီမှာ လိုချင်တာ categoryid မလို့ categoryid ကို ယူထားတာဖြစ်ပါတယ်။ နောက်ပြီး မှတ်စရာတစ်ခုက query ထဲမှ ရေးသမျှ key တွေက အကုန်အသေးပါပဲ။ နောက်ပြီး query string ထဲမှာ နောက်ထပ် second query parameter ရေးခြင်တဲ့အခါ & ကို အသုံးပြုပါတယ်။ ဒီလိုပါ။

```
http://localhost:4000/part_6/getRequest.php?categoryid=10&limit=30
```

ဒါဆိုရင် limit ကိုကော အသုံးပြုလို့ရပါပြီ။

Form Handlings

PHP နဲ့ HTML form ကို handlings လုပ်နိုင်ခြင်းက dynamic website တိုင်းရဲ့ အရေးကြီးတဲ့ process တစ်ခုဖြစ်ပါတယ်။ website တစ်ခု သို့မဟုတ် web application တစ်ခုခုကို develop လုပ်မယ်ဆိုရင် users တွေကို Login form တို့ registration form တွေကို ကနေ input တွေက ပေးပါတယ်။ အဲ့ဒီ လို form parameters တွေကို access လုပ်ပြီး process လုပ်ရပါမယ်။ အဲ့ဒီလို form handlings လုပ်ဖို့အတွက် methods တွေအကြောင်းသိရပါမယ်။ သူတို့က form data တွေ form parameters တွေကို submit လုပ်ပြီး ဘယ်လို အသုံးပြုမလဲပေါ့။
Form တစ်ခု ဖန်တီးဖို့အတွက် form tags ကို အသုံးပြု ဖန်တီးပြီး input တွေအတွက် elements တွေ တည်ဆောက်လို့ရပါတယ်။ ဒီလိုလေးပါ။

```
<form action="script.php" method="get" >
</form>
```

ဒီ FROM တစ်ခုမှာဆိုရင် အရေးကြီးဆုံး form tag က action ဖြစ်ပါတယ်။ အဲ့ဒီ action က form data တွေကို ဘယ် page ကို sent မယ်ဆိုတာကို ညွှန်ပြပေးပါတယ်။ နောက်ထပ် အရေးကြီး တဲ့ form tag က method ဖြစ်ပါတယ်။ form data တွေကို page ကို ဘယ်လိုပို့မယ်ဆိုတာကို ဆောင်ရွက်ပေးပါတယ်။ အဲ့ဒီ လိုပို့တဲ့အခါမှာ method နှစ်မျိုးရှိပါတယ်။ get နဲ့ post ဆိုပြီးတော့ HTTP(Hypertext Transfer Protocol) method တွေဖြစ်ပါတယ်။ GET method က submitted data တွေကို လက်ခံမဲ့ page ကို ပို့တဲ့အခါမှာ name-value pairs ပုံစံနဲ့ URL bar မှာပို့ပေးပါတယ်။

POST Methods ($_POST[''])

ရှေ့မှာ URL bar သို့မဟုတ် query string ကနေ information တွေကို retrieve လုပ်ချင်တဲ့အခါ GET ကို အသုံးပြုပါတယ်။ ကျနော် ပြောခဲ့သလိုပဲ GET က information တွေ retrieve လုပ်တဲ့နေရာမှာ အသုံးများပါတယ်။ Application ရဲ့ state ကို change ချင်တာ Database မှာ တစ်ခုခု

အပြောင်းလည်းတွေလုပ်ချင်တဲ့အခါမျိုးမှာကျ GET request ကို အသုံးမပြုကျပါဘူး။ passowrd လိုမျိုး sensitive information တွေပါလာတဲ့အခါ GET ကို အသုံးပြုလို့မရပါဘူး။ URL မှာ ဒီတိုင်းကြီး ပေါ်နေပါလိမ့်မယ်။ ဒါကြောင့် sensitive information တွေပို့ချင်တဲ့အခါ Datastore လုပ်တာပတ်သတ်ပြီး တစ်ခုခုလုပ်ချင်တဲ့အခါ application ရဲ့ state တွေကို တစ်ခုခုချိန်းချင်တဲ့အခါ POST requst ကိုအသုံးပြုပါတယ်။ POST request က post နဲ့ ပို့ လိုက်တဲ့ information တွေကို $_POST[] ဆိုတဲ့ super global array ကို အသုံးပြုပါတယ်။ နောက်ပိုင်းမှာ example တွေနဲ့ တွဲပြီးရှင်းပြပါ့မယ်။

File Handlings

File တစ်ခုကို အောင်မြင်အောင် uploads လုပ်ဖို့ အတွက် PHP settings ကို configure လုပ်စရာလေးတွေရှိပါတယ်။ PHP file upload လုပ်ဖို့အတွက် အရေးကြီးတဲ့ option လေးတွေရှိပါတယ်။ အဲ့ဒီ option တွေကို php.ini file ထဲမှာ သွားပြီးတော့ configured သွားလုပ်ရပါတယ်။
php.ini file ကိုရှာမတွေ့ရင်, php_ini_loaded_file() ကို သုံးပြီးတော့ ဘယ်နေရာမှာ ရှိလည်း ရှာလို့ရပါတယ်။ ဒီလိုမျိုးပါ။

```
<?php echo php_ini_loaded_file(); ?>
```

ဒီလို run လိုရင် php.ini file ရှိတဲ့ location ရပါလိမ့်မယ်။

```
; Whether to allow HTTP file uploads.
; http://php.net/file-uploads
file_uploads=On


; Temporary directory for HTTP uploaded files (will use system default if not
; specified).
; http://php.net/upload-tmp-dir;
```

```
upload_tmp_dir="C:\xampp\tmp"
; Maximum allowed size for uploaded files.
; http://php.net/upload-max-filesize
upload_max_filesize=40M

; Maximum number of files that can be uploaded via a single request
max_file_uploads=20
```

ဒီ key settings မှာ ပထမဆုံးတွေ့ရတဲ့ fie_uploads ကို ကျနော် တို့ On တယ်ဆိုတာက file တွေ uploads လုပ်ဖို့ ခွင့်ပြုတယ်လို့ဆိုလိုတာပါ။ သူ့ရဲ့ Default value က On ထားတာဖြစ်ပါတယ်။

Upload_max_filesize က ကျနော်တို့ upload file ရဲ့ အမြင့်ဆုံး လက်ခံနိုင်တဲ့ maximum size ကိုဆိုလိုပါတယ်။ file exceeds upload_max_filesize ဆိုတဲ့ error တက်ရင် ဒီ value ကိုတိုးပေးရပါမယ်။ အဲ့လိုတိုးတဲ့အချိန်သေခြာအောင် post_max_size ကိုလည်း တိုးပေးရပါတယ်။

Upload_tmp_dir ဆိုတာက file တစ်ခုခုကို store လုပ်တဲ့ အခါ temporary directory အနေနဲ့ အသုံးပြုဖို့ဖြစ်ပါတယ်။ ပုံမှန်အားဖြင့်ဆို သူ့ကို value ထားဖို့မလိုပါဘူး။

post_max_size က POST method နဲ့ ပို့တဲ့ data ရဲ့ maximum size ကို ဆိုလိုတာပါ။ file upload လုပ်တဲ့အခါ post request နဲ့ပဲလုပ်တာများတဲ့အတွက် ဒီ post_max_size ကို value set ထားရပါမယ်။ အကယ်၍ upload_max_filesize ကို 10 MB လို့ထားရင် post_max_size ကို ၁6 Mb လောက်ထားသင့်ပါတယ်။

max_file_uploads ဆိုတာက file တွေကို တစ်ခါ Upload လုပ်ရင် ဘယ်လောက် upload လုပ်လို့ရတယ်ဆိုတာကိုဆိုလိုတာပါ။ default အနေနဲ့ တစ်ခါ upload ကို file 20 uploads လုပ်လို့ရပါတယ်။ ဒီ လောက်သိထားရင် လုံလောက်ပါပြီ။

PHP မှာ file uploads လုပ်တဲ့အခါ single or multiple files upload လုပ်လို့ရပါတယ်။

File information တွေကို လက်ခံယူတဲ့အခါ PHP ရဲ့ global $_FILES နဲ့ယူပါတယ်။ $_FILES ထဲမှာ file နဲ့ သက်ဆိုင်တဲ့ file name, file type, file size, temp file name နဲ့ file errors တွေကို ရနိုင်ပါတယ်။

file တစ်ခုကို upload လုပ်လိုက်တဲ့အခါ

$_FILES[‘filename’][‘name’] က file name ကို return ပြန်ပေးပါတယ်။ $_FILES[‘filename’][‘type’]

file ရဲ့ MIME type ကို return ပြန်ပေးပါတယ်။ $_FILES['filename']['size'] က file ရဲ့ size ကို bytes format နဲ့ return ပြန်ပေးပါတယ်။ $_FILES['filename']['tmp_name'] က file အတွက် temporary file name တစ်ခု ဖန်တီပြီးတော့ သတ်မှတ်ထားတဲ့ folder ကို မရွေ့ခင် server ထဲမှာ store လုပ်ထားပါတယ်။ $_FILES['filename']['error'] က file upload လုပ်တဲ့အခါ တက်တဲ့ error ကို return ပြန်ပေးပါတယ်။ error မရှိခဲ့ရင် 0 လို့ return ပြန်ပါလိမ့်မယ် ။ Error ရှိလား စစ်ချင်ရင် error > 0 နဲ့ စစ်လို့ရပါတယ်။

```
<form action="" method="post" enctype="multipart/form-data">
    Select File:
    <input type="file" name="fileToUpload"/>
    <input type="submit" value="Upload Image" name="submit"/>
</form>
```

File upload လုပ်ဖို့ဆိုရင် form element မှာ enctype = “multipart/form-data” ကို ထည့်ပေးရပါမယ်။ enctype attribute က MIME data တွေဖြစ်တဲ့ Image, audio, video တွေကို Upload ensures ဖြစ်တယ်လို့ဆိုလိုတာပါ။

```
<?php
if($_FILES["photo"]["error"] > 0){
    echo "Error: ". $_FILES["photo"]["error"] . "<br>";
} else{
    echo "File Name: ". $_FILES["photo"]["name"] . "<br>";
    echo "File Type: ". $_FILES["photo"]["type"] . "<br>";
    echo "File Size: ". ($_FILES["photo"]["size"] / 1024) . " KB<br>";
    echo "Stored in: ". $_FILES["photo"]["tmp_name"];
}
?>
```

ဒီလို ရေးပြီးတော့ file ရဲ့ information တွေကို output ထုတ်ကြည့်နိုင်ပါတယ်။
နောက်ပြီး file upload လုပ်တဲ့အခါမှာ အရေးကြီးတဲ့ function တစ်ခုရှိပါတယ်။ move_uploaded_file() function ဖြစ်ပါတယ်။ သူက uploaded file ကို location တစ်ခုဆီကို file ကို move လုပ်ပေးပါတယ်။ သူက POST request က လာတဲ့ file ကို check လုပ်ပြီးတော့ သတ်မှတ်ထားတဲ့ location ကို ရွှေ့ ပေးပါတယ်။ ဒီလိုပါ။

```
move_uploaded_file (string $from, string $to) : bool
```

Function ကို အသုံးပြုလိုက်ပြီးဆိုရင် file က valid ဖြစ်ရင် $from ကို $to နေရာကို ရွှေ့ပေးပါတယ်။ ဆိုလိုတာက temporary နေရာက file ကို ကို သတ်မှတ်ထားတဲ့ file ထဲကို ရွှေ့မယ်ဆိုလိုတာပါ။ ဒီလိုပါ။

```
move_uploaded_file ( string $filename , string $destination )
```

ကို ရွှေ့ချင်တဲ့ file ကို $filename နေရာမှာ ရေးပြီးတော့ ဘယ်နေရာကို ရွှေ့မှာ လဲဆိုတာကို $destination နေရာမှာ ရေးပါတယ်။

```
<?php
$target_path = "e:/";
$target_path = $target_path.basename( $_FILES['fileToUpload']['name']);

if(move_uploaded_file($_FILES['fileToUpload']['tmp_name'], $target_path)) {
    echo "File uploaded successfully!";
} else{
    echo "Sorry, file not uploaded, please try again!";
}
?>
```

ဒီမှာဆို ကျနော် တို့ file upload လုပ်လိုက်တုန်း က tempory file အနေနဲ့ server ထဲမှာ ရှိတဲ့ file ကို ကို သတ်မှတ်ထားတဲ့ folder ထဲကို ရွှေ့လိုက်ပါတယ်။ အဲ့ဒီလို ရွှေ့တဲ့အခါမှာ “ folder / filename ” ပုံစံနဲ့

ရွေ့ရပါတယ်။ ဒီလိုဆို ကိုသတ်မှတ်ထားတဲ့ folder ထဲမှာ ကို upload လုပ်လိုက်တဲ့ file ရောက်သွားပါလိမ့်မယ်။

Server Information

$_SERVER array မှာ web server အတွက် အသုံးဝင်တဲ့ information တွေများစွာ ပါဝင်ပါတယ်။ ဒီ information တွေတော်တော်များများက (CGI) လို့ ခေါ်တဲ့ Common Gateway Interface လို့ခေါ်တဲ့ environment variables တွေကနေလာတာ ဖြစ်ပါတယ်။ အဲ့ဒီ ထဲက မှာ အသုံးများတဲ့ entries တွေကို ဖော်ပြပါ့မယ်။

PHP_SELF

Document root နဲ့ သက်ဆိုင်တဲ့ current script ရဲ့ name ဖြစ်ပါတယ်။ ဒီ variable က self-referencing scripts ကို create လုပ်တဲ့နေရာမှာ အသုံးဝင်ပါတယ်။

SERVER_PORT

Request sent တဲ့ server port number ဖြစ်ပါတယ်။ example (80).

REQUEST_METHOD

The document ကို fetch လုပ်တဲ့ client ရဲ့ method ဖြစ်ပါတယ်။

QUERY_STRING

URL ရဲ့? နောက်က အကုန်လုံးကို Query_STRING လို့ခေါ် ပါတယ်။

Include and require files

Include statement and require statement နဲ့က text တွေ code တွေ ပါတဲ့ file တွေကို copy လုပ်ပြီးတော့ တစ်ခြား PHP file မှာ အလုပ်လုပ်စေချင်တဲ့ အခါမှာ အသုံးပြုပါတယ်။ တူညီတဲ့ PHP HTML tags ပါတဲ့ multiple pages တွေကို include လုပ်ချင်တဲ့အခါ မှာ files တွေကို including လုပ်ခြင်းက

အရမ်းအသုံးဝင်ပါတယ်။ တစ်ကယ် လက်တွေ့မှာ တော့ဘာ တူ ညီတဲ့ block of code တွေကို seprate လုပ်ပြီးတော့ ပြန်လိုချင်တဲ့အခါမှာ include statement or require statement သုံးပြီးတော့ လိုချင်တဲ့ နေရာမှာ အကြိမ်ကြိမ်ပြန်သုံးလို့ရပါတယ်။

Include Statement (include and include_once)

File ကို အခြား file အတွင်းမှာ ပြန် သုံးချင်တဲ့အခါ include ကို အသုံးပြုပါတယ်။ include ကို အသုံးပြုတဲ့အခါ တစ်ခါ ခေါ်ရင် တစ်ခါ အလုပ်လုပ်ပါတယ်။ ဆိုလိုတာက နှစ်ခါ ဆက်တိုက်ခေါ်ရင် နှစ်ခါ အလုပ်လုပ်ပါတယ်။ ဒီလိုပါ။

include("path/to/filename"); -Or- include "path/to/filename";

Include.php

```
<?php
        echo “Hello Include <br>”;
```

Index.php

```
<?php
    include “include.php”;
    include “include.php”;
?>
    <h1>Include Statement</h1>
```

ဒီမှာဆိုရင် include.php ကို index.php မှာ include လုပ်ထားတဲ့အတွက် index.php ကို run တဲ့အခါ Hello Include ဆိုပြီးတော့ တွေ့ရပါလိမ့်မယ် ။ နှစ်ခါ ခေါ်ထားတဲ့အတွက် နှစ်ခါ ထွက်နေပါလိမ့်မယ်။ ဒီလိုပါ။

```
Hello Include
Hello Include
Include Statement
```

Include_once ကို တစ်ခါပဲ ခေါ်ချင်တဲ့အခါမှာ အသုံးပြုပါတယ်။ သူ့ကို နှစ်ခါ ခေါ်ရင် လည်း တစ်ခါ

ပဲအလုပ်လုပ်ပါတယ်။

```
Include.php
        <?php
                echo “Hello Include <br>”;
Index.php
        <?php
                include_once “include.php”;
                   include_once “include.php”;
        ?>
        <h1>Include Statement</h1>
Hello Include
Include Statement
```

ဆိုပြီးတော့ပဲ ရပါလိမ့်မယ်။

Require Statement

Require statement မှာလည်း require and require_once ဆိုပြီးတော့ ရှိပါတယ်။ အထက်မှာ ဖော်ပြခဲ့တဲ့ include statement တွေဖြစ်တဲ့ include, include_once တို့နဲ့ အလုပ်လုပ်ပုံခြင်းတူပါတယ်။ ဒါပေမဲ့ include statement နဲ့ require statement မှာ ကွာခြားပုံလေးတွေရှိပါတယ်။ ဒီလိုပါ။

Require statement or include statement ကို အသုံးပြုတဲ့အခါ error တစ်စုံတစ်ခုတက်တဲ့အခါ Require statement က a fatal error (E_COMPILE_ERROR) တက်ပြီးတော့ script က ရပ်သွားပါတယ်။ဆိုလိုတာက require statement မှာ အသုံးပြုမှားရင် error တက်ပါတယ်။

Include statement က warning (E_Warning) ပဲတက်ပြီးတော့ script က ဆက်ပြီး အလုပ်လုပ်ပါတယ်။ include statement မှားသော်လည်း warning ပဲပြပြီး အလုပ်ဆက်လုပ်ပါတယ်။ ဒါကြောင့် include လုပ်တဲ့ file က missing ဖြစ်သော်လည်း user ကို output ပြမယ် execution ကိုဆက်လုပ်ခြင်တယ်ဆိုရင် include statement ကိုအသုံးပြုပါ။ တစ်ချို့အခြေနေတွေမှာ ဥပမာ CMS

လို content management system တွေမှာဆိုရင် execution တစ်ခုလုံးပြီးဖို့အတွက် ပါမှရမဲ့ file တွေပါလာပြီးဆိုရင် require statement ကိုအသုံးပြုရပါမယ်။ file including လုပ်တာက အရမ်းကို အသုံးဝင်ပြီးတော့ အလုပ်အများကြီးပြီးစေပါတယ်။ ဆိုလိုတာက ကျနော်တို့ application ရဲ့ page တိုင်းမှာ ပါရမဲ့ header, footer, or menu တို့ ကို standard ဆောက်ထားလို့ရပါတယ်။ အဲဒီလိုဆိုရင် header မှာ တစ်ခုခုပြင်ချင်တဲ့အခါ web pages တိုင်းကို ပြင်စရာ မလိုတော့ပဲ standard ဆောက်ထားတဲ့ header မှာ ပြင်ပြီး file including ပြန်လုပ်ရုံပါပဲ။

Redirection

Redirect mechanism ကို hyperlinks အကူညီ မပါဘဲ hyperlinks တွေကို click မလုပ်ဘဲနဲ့ user ကို page တစ်ခုကနေ အခြား page တစ်ခုကို ညွှန်ပေးတာကို ဆိုလိုပါတယ်။ တစ်ချို့အခြေနေတွေမှာ background မှာ အခြေနေပေါ် မူတည်ပြီး redirection လုပ်ခြင်းက အသုံးဝင်ပါတယ်။

အဲ့ဒီလို redirection လုပ် ဖို့အတွက် PHP မှာ header() လို့ခေါ်တဲ့ predefined function ရှိပါတယ်။ သူ့ကို common အနေနဲ့ redirection လုပ်ရာမှာ အသုံးများပါတယ်။

```
PHP Redirect Syntax
header(“Location: target-url”);
```

ဒီ PHP redirect မှာ အမှန်တကယ်ရှိတဲ့ URL ကို replace လုပ်ပြီး redirect လုပ်ရပါတယ်။ အဲ့ဒီလို redirect မလုပ်ခင်မှာ redirect လုပ်ချင်တဲ့ URL or page ရှိမရှိ စစ်ဖို့လိုပါတယ်။ စစ်ချင်ရင် ဒီလိုလေးရေးလို့ရပါတယ်။

```
 echo "PHP Redirect";
header("Location: phppot.com");
```

ဒီလိုဆို သွားချင်တဲ့ URL ကမမှန်တဲ့အခါ
Warning: Cannot modify header information - headers already sent by (... ဆိုပြီး warning

error တက်ပါလိမ့်မယ်။

နောက်ပြီး redirect လုပ်တဲ့အခါမှာ safely ဖြစ်အောင် exit() ဆိုတဲ့function ပါ ထပ်သုံးပါတယ်။

အခန်း(၉) - Regular Expressions

Regular expression ဆိုတာ sequence of characters လေးဖြစ်ပြီးတော့ search pattern အဖြစ် အလုပ်လုပ်ပါတယ်။ သူကို့ regex or RegExp လို့လည်း အသိများပါတယ်။ သူကို့ user input field တွေဖြစ်တဲ့ name, email, phone number စတဲ့ input တွေကို data format ကျ မကျ verify လုပ်ချင်တဲ့အခါမှာ အသုံးများပါတယ်။

Regular Expression Syntax

```
/food/
```

ဒီမှာဆိုရင် foot ကို forward slashes(/) နှစ်ခုထဲမှာ ထည့်ထားပါတယ်။ သူတို့ကို delimiters လို့လည်း ခေါ်ပါတယ်။ forward slashes(/) အစား hash sign(#), plus(+), percentage(%) သို့ မဟုတ် တစ်ခြားသော delimiters တွေကို လည်း အသုံးပြုလို့ရပါတယ်။ တစ်ခြားသော delimiters တွေသုံးမယ်ဆိုရင် escaping လုပ်ဖို့လိုတာတွေရှိလာနိုင်ပါတယ်။ ဥပမာ URL အတွက် pattern လုပ်မယ်ဆိုရင် URL မှာ slashes တွေပါတဲ့အတွက် slashes delimiters အစား အခြား delimiters ကို အဆင်ပြေသလို သုံးလို့ရပါတယ်။

```
/http:\/\/somedomain.com\//
#http://somedomain.com/#
```

```
+hello world+
%hello world%
~hello world~
```

Delimiters ကို ဒီလိုတွေလည်း အသုံးပြုလို့ရပါတယ်။

Words တွေကို အတိအကျတူဖို့ matching လုပ်မဲ့အစား multiple words matching လုပ်ဖို့အတွက် qualifiers တွေကို အသုံးပြုလို့ရပါတယ်။

```
/fo+/
```

+ qualifier က f နောက်မှာ one or more o လိုက်ရင် pattern နဲ့ match ဖြစ်မယ်လိုဆိုလိုတာပါ။ ဒါကြောင့် food, fool, fo4 ဆိုရင် matching ဖြစ်ပါတယ်။ တစ်ခြားတစ်ခုအနေနဲ့ * qualifier ကို အသုံးပြုလို့ရပါတယ်။ သူက f နောက်မှာ o က zero or more o လိုက်လို့ရတယ်လို့ဆိုလိုတာပါ။

```
/fo*/
```

ဒီလိုဆိုရင် food, fool, fo4 တို့လည်း match ဖြစ်သလို fast, fine, … စတာတွေနဲ့လည်း matching ဖြစ်ပါတယ်။ ဒီ qualifiers နှစ်ခုလုံးက f နောက်မှာ characters ဘယ်နှလုံး လိုက်ရမယ်လို့ limits မထားထားတာပါဘူး။ အဲ့ဒီ လိုဆိုမျိုး မိမိလိုချင်တဲ့ characters အရေအတွက်ပဲ limits လုပ်လို့ရပါတယ်။

```
/fo{2,4}/
```

ဒီလိုဆို f နောက်မှာ o က 2 လုံး ကနေ 4 လုံး အထိ ပါလို့ရပါတယ်။ ဒါကြောင့် fool, fooool,football တို့နဲ့ matching ဖြစ်ပါတယ်။ ဆိုလိုတာ က f နောက်မှာ o ဟာ အနည်းဆုံး 2 ရှိရပြီး အများဆုံး 4 လုံး လိုက်ရမယ်ဆိုလိုတာပါ။

ဒီ ဥပမာတွေမှာ ဆိုရင် pattern က f နဲ့ စပြီး o က one or more , 0 or more နဲ့ 2 ကနေ 4 လုံးလိုက်ရမှာ ဖြစ်ပါတယ်။ ဒီ pattern နောက်က သို့ မဟုတ် အရှေ့က ဟာ တွေ matching ဖြစ်ဖြစ် စစ်ခြင်းမရှိပါဘူး။

Modifiers

Regular expression ကို ထပ်ပြီး modify လုပ်ခြင်တဲ့အခါ modifiers တွေကို အသုံးပြုပါတယ်။

i - က case-insensitive search ဖြစ်ပါတယ်။

m - string ကို serval အဖြစ်အလုပ်လုပ်ပါတယ်။ m ဆိုတာ multiple ကို ဆိုလိုတာပါ။ default အနေနဲ့ ^ နဲ့ $ characters တွေကို string ရဲ့ အစနဲ့ အဆုံးမှာ ထားပြီး ရှာလေ့ရှိပါတယ်။အဲ့ ဒီ m modifier က string ထဲမှာ ^ နဲ့ $ အတွင်းမှာ ရှိတဲ့ line တိုင်းကို matching လုပ်ပေးပါတယ်။

s - က single line အဖြစ် လုပ်ဆောင်ပြီး အဲဒီ အထဲမှာ newline characters တွေပါလည်း ignore လုပ်ပါတယ်။

ဒီ modifiers တွေကို regular expression ရဲ့ အဆုံးမှာ ထားရပါတယ်။

ဥပမာ

```
/string/i
```

/wmd/i က WMD , wMD , WMd , wmd တွေနဲ့ matching ဖြစ်ပါတယ်။

တစ်ခြားသော language တွေမှာ global modifier(g) ကို support ပေးပါတယ် ။ PHP မှာ အဲ့ဒီ အတွက် preg_match() နဲ့ preg_match_all() ဆိုပြီး function နှစ်ခုခွဲထားပါတယ်။

Metacharacters

Regular expressions က သူတို့ရဲ့ searches ပိုပြီး filter လုပ်ချင်တဲ့အခါမှာ metacharacters ကို အသုံးပြုပါတယ်။ metacharacters တွေက simple character တွေဖြစ်ပြီးတော့ သူတို့ တစ်ခုခြင်းစီမှာ symboliz special meaning တွေရှိပါတယ်။ အသုံးများတဲ့ metacharacters တွေကို ဖော်ပြ ပါ့မယ်။

\A - က string ရဲ့ အစနဲ့ Matches ဖြစ်ရမယ်လို့ဆိုတာပါ။

\b - က a word boundary ပါ။

\B - က အကုန်လုံး နဲ့ match ဖြစ်ပါတယ်။ ဒါပေမဲ့ word boundary နဲ့ match မဖြစ်ရပါဘူး။

\d - က digit character နဲ့ match ဖြစ်ရပါမယ်။ သူက [0 - 9] နဲ့ တူတူပါပဲ။

\D - က non-digit character နဲ့ match ဖြစ်ရပါတယ်။

\s - က whitespace character နဲ့ matches ဖြစ်ရမယ်ဆိုလိုတာပါ။

\S - က non-whitespace character နဲ့ match ဖြစ်ရမယ်ဆိုလိုတာပါ။

[ ] - က character class တစ်ခုဖန်တီးတာဖြစ်ပါတယ်။

( ) - က character grouping လုပ်တာဖြစ်ပါတယ်။

$ - က line ရဲ့ အဆုံးသတ်ဖြစ်ပါတယ်။

^ - က string ကို သတ်မှတ်ထားတဲ့ characters နဲ့ စရမယ်ဆိုလိုတာပါ။

. - က newline ဆင်းတဲ့ character တွေနဲ့ matches မဖြစ်ဘူးဆိုလိုတာပါ။

\w - က underscore နဲ့ alphanumeric characters တွေ ပါတဲ့ string တွေနဲ့ matches ဖြစ်တယ်ဆိုလိုတာပါ။

\W - က string မှာ underscore နဲ့ alphanumeric characters တွေ ပါရင် match မဖြစ်ဘူးလို့ဆိုလိုတာပါ။

ဥပမာလေးတွေနဲ့ ပြပါ့မယ်။

```
/sa\b/
```

ဆိုရင် string က pisa တို့ lisa တို့ ဆိုရင် match ဖြစ်ပါတယ်။ sand တော့မရပါဘူး။

```
/\blinux\b/i
```

ဆိုရင် string က case-insensitive ဖြစ်ပြီးတော့ linux ဖြစ်ရင် match ဖြစ်ပါတယ်။

```
/sa\B/i
```

ဆိုရင် string က sa နဲ့ဆုံးလို့မရပါဘူး။ sand တို့ Sally တို့နဲ့ match ဖြစ်ပြီးတော့ Melissa နဲ့ တော့ match မဖြစ်ပါဘူး။

Regular Expression Functions

PHP ရဲ့ version 5.3 နဲ့ အထက်မှာ Perl style regular expressions လို့ခေါ်တဲ့ preg_ function family ကို support ပေးပါတယ်။

preg_match() - regular expression match ဖြစ်ရင် 1 return ပြန်တယ် match မဖြစ်ရင် 0 return ပြန်ပေးပါတယ်။

preg_match_all() - string တစ်ခုလုံးကို pattern နဲ့ စစ်ပြီးတော့ match ဖြစ်တဲ့ အချိန်တွေကို return ပြန်ပေးပါတယ်။ သူ့ကို global regular expression လို့လည်း ခေါ်ပါတယ်။

preg_replace() - regular expression match ဖြစ်တဲ့ နေရာကို replace လုပ်ပါတယ်။

Character Classes

Characters patten တစ်ခုကို square brackets နဲ့ ထည့်ထားတဲ့ပုံကို character class လို့ခေါ်ပါတယ်။ e.g.[abc]. ဒီလိုဆိုရင် patten က abc တစ်လုံးလုံးနဲ့ ညီ ရမယ်လို့ဆိုလိုတာပါ။ ဒါကို negated character clases လို့လည်းခေါ်ပါတယ်။ ဆိုလိုတာက except ကိုဆိုလိုတာပါ။ e.g.[^abc] ဆိုရင် abc က လွဲလို့ ကျန်တဲ့ characters နဲ့ စရမယ်လို့ဆိုလိုတာပါ။

Characters range တွေကို define လုပ်ချင်တယ်ဆို characters class ထဲမှာ hyphen (-) character ကို အသုံးပြုပါတယ်။ [0-9] ဆိုရင် 0 ကနေ 9 ထိ numbers တွေနဲ့ match ဖြစ်ရမယ် ဆိုလိုပါတယ်။

[abc] - a, b, c တစ်ခုခုနဲ့ match ဖြစ်ရမယ်လို့ဆိုလိုတာပါ။

[^abc] - a, b, c ကလွဲရင် တစ်ခြား characters တွေနဲ့ match ဖြစ်တယ်လို့ဆိုလိုတာပါ။

[a-z] - lowercase a to lowercase z ထဲက characters တွေနဲ့ match ဖြစ်တယ်လို့ ဆိုလိုတာပါ။

[A-Z] - uppercase a to uppercase z ထဲက characters တွေနဲ့ match ဖြစ်တယ်လို့ဆိုတာပါ။

[a-Z] - lowercase a to uppercase Z ထဲ က characters တွေနဲ့ match ဖြစ်တယ်လို့ ဆိုလိုတာပါ။

[0-9] - 0 နဲ့ 9 ထိ digital တစ်ခုခုနဲ့ match ဖြစ်မယ်လို့ဆိုတာပါ။

[a-z0-9] - ဒီလိုဆိုရင်တော့ a to z ကနေ 0 to 9 ထဲ က တစ်ခုခုနဲ့ match ဖြစ်တယ်လို့ဆိုတာပါ။

```
$pattern = "/ca[kf]e/";
$text = "He was eating cake in the cafe.";
if(preg_match($pattern, $text)){
   echo "Match found!";
} else{
   echo "Match not found.";
}
```

ဒီမှာဆို character class ([]) တစ်ခုဖန်တီးထားပါတယ်။ ca ရဲ့ နောက်မှာ k or a တစ်လုံးလုံး လိုက်ရမယ်။ ပြီးရင် e လိုက်ရမယ်လို့ဆိုလိုတာပါ။ ဒီမှာဆိုရင် preg_match() ကိုအသုံးပြုထားပါတယ်။ ဒါကြောင့် ပထမဆုံးတွေ့မယ့် case or expression ကိုပဲစစ်ပါတယ်။ ဒီမှာဆို cake ကို တွေ့တဲ့အခါ match ဖြစ်ပါတာကြောင့် 1 return ပြန်ပါမယ်။ ဒါကြောင့် “Match found!” ဖြစ်ပါတယ်။

```
<?php
$regex = '/^[a-zA-Z ]*$/';
$nameString = 'Sharukh khan';
if(preg_match($regex, $nameString)) {
   echo("Name string matching with regular expression");
} else {
   echo("Only letters and white space allowed in name string");
}
```

ဒီမှာဆိုရင် character class([]) ထဲမှာ a-zA-Z လို့ရေးထားတဲ့အတွက် a ကနေ Z အတွင်း ကြိုက်တဲ့ စကားလုံးဖြစ်လို့ရပါတယ်။ character class အစ အပြင်ဖက်မှာ ^ ပါတဲ့အတွက် a ကနေ Z အတွင်း စကားလုံးနဲ့ပဲ စရပါမယ်။ character class အပြင်ဖက်မှာ $ နဲ့ဆုံးတဲ့အတွက် a ကနေ Z အတွင်း စကားလုံးနဲ့ပဲဆုံးရပါမယ်။ ဒါကြောင့် match ဖြစ်ပါတယ်။

```
<?php
$pattern = "/ca[kf]e/";
$text = "He was eating cake in the cafe.";
$matches = preg_match_all($pattern, $text, $array);
echo $matches . " matches were found.";
```

ဒီမှာဆိုရင် preg_match_all() ကိုသုံးထားတဲ့အတွက် string ထဲမှာရှိတဲ့ ca[kf]e တွေကို အကုန်လုံးကို စစ်ပါတယ်။ ဒီမှာဆို cake နဲ့ cafe ဆိုပြီး နှစ်ခုလုံးက expression နဲ့ match ဖြစ်ပါတယ်။ ဒါကြောင့် $matches ထဲကို 2 ဝင်သွားပါလိမ့်မယ်။ ဒါကြောင့် “2 matches were found.” ဖြစ်ပါတယ်။

```
<?php
$str = 'Hello World';
echo preg_replace('/hello/', 'Hi', $str);
// outputs "Hello World"
echo '<br>';
echo preg_replace('/hello/i', 'Hi', $str);
// outputs "Hi World"
```

ဒီမှာဆို ပထမ တစ်ခုမှာ string ကနေ “hello” ရှိတဲ့ နေရာတွေကို “Hi” နဲ့ replace လုပ်ခိုင်းထားပါတယ်။ ဒါပေမဲ့ “hello” ကို မတွေ့ပဲ “Hello” ကိုပဲတွေ့တာကြောင့် repalce မဖြစ်သွားပါဘူး။
ဒုတိရတစ်ခုစစ်တဲ့အချိန်မှာ expression မှာ “i”(case-insensitive search)ကို အသုံးပြုထားတဲ့အတွက် “hello” or “Hello” တွေ့တဲ့အတွက် replace ဖြစ်သွားပါတယ်။

Other String Functions

String ရဲ့ Length ကို လိုချင်တဲ့အခါ မှာ PHP ရဲ့ strlen() ဆိုတဲ့ function ကို အသုံးပြုလို့ရပါတယ်။ function က string ရဲ့ length ကို return ပြန်ပေးပါတယ်။

```
Int strlen(string str)
<?php
$pswd = "secretpswd";
if (strlen($pswd) < 10)
echo "Password is too short!";
else
echo "Password is valid!";
?>
```

strcmp() - String နှစ်ခုကို compare လုပ်ချင်တဲ့အခါမှာ PHP မှာ strcmp() , strcasecmp() strspn() , strcspn() ဆိုပြီးရှိပါတယ် ။ ဒီထဲမှာ string နှစ်ခုရဲ့ case-sensitive မှာ စစ်ချင်တယ်ဆိုရင် strcmp() ကို အသုံးပြုပါတယ်။

int strcmp(string str1,string str2)

strcmp() functions က compare လုပ်တဲ့ string တွေအရ possible values သုံးခု return ပြန်ပါတယ်။

Str1 နဲ့ str2 နဲ့ တူရင် 0 return ပြန်ပါတယ်။

Str1 က str2 ထက်ငယ်ရင် -1 return ပြန်ပါတယ်။

Str2 က str1 ထက်ငယ်ရင် 1 return ပြန်ပါတယ်။

```
<?php
$pswd = "supersecret";
$pswd2 = "supersecret2";
if (strcmp($pswd, $pswd2) != 0) {
echo "Passwords do not match!";
```

```
} else {
echo "Passwords match!";
}
?>
```

strcmp() function နဲ့ စစ်ရင် စာလုံးအကြီးအသေးက အစ တူရပါမယ် မတူရင် flase 0 return ပြန်ပါတယ်။

strcmp() function နဲ့ စစ်သလို equality operator == နဲ့ စစ်လို့မ ရပါတယ်။

```
if ($str1 == $str2)
```

ဒီလို စစ်ရ strcmp() function နဲ့ အလုပ်လုပ်ပုံချင်းတူပါတယ်။

```
<?php
$email1 = "admin@example.com";
$email2 = "ADMIN@example.com";
if ($email1 == $email2)
echo "The email addresses are Identical";
?>
```

ဒီလိုဆိုရင် True ထွက်ပါလိမ့်မယ်။

trim() - String ထဲကနေ white space တွေ remove လုပ်ချင်တဲ့အခါ string ရဲ့ site နှစ်ဖက်လုံးကနေ white space တွေ ဖယ်ချင်တဲ့အခါ အသုံးပြုပါတယ်။

```
<?php
$string_name=' Welcome to PHP ';
$new_string=trim($string_name);
echo $new_string;
?>
```

strtoupper() - string တစ်ခုလုံးကို uppercase ပြောင်းချင်တဲ့အခါ အသုံးပြုပါတယ်။

```
<?php
echo strtoupper(“welcome to php”);// It will convert all letters of string into uppercase
//output
WELCOME TO PHP
?>
```

strtolower() - string တစ်ခုလုံးကို lowercase ပြောင်းချင်တဲ့အခါ အသုံးပြုပါတယ်။

```
<?php
echo strtolower(“Welcome To PHP”);// It will convert all letters of string into uppercase
//output
welcome to php
?>
```

အခန်း(၁၀) - PHP Sessions

Sessions ဆိုတာ information တွေ ကို web pages တိုင်းမှာ အသုံးပြုနိုင်ရန် သိမ်းဆီပေးတဲ့ အလုပ်လုပ်ပါတယ်။ sessions တွေကို website တွေမှာ ဘာကြောင့် လိုအပ်သလဲဆိုတော့ HTTP protocol တွေ အလုပ်လုပ်တဲ့ ပုံစံကြောင့်ပါ။

HTTP protocol က stateless protocol ဖြစ်ပါတယ်။ stateless protocol ဆိုတာက အကယ်၍ user တစ်ယောက် က webpage တစ်ခုမှာ action တစ်ခုခုလုပ်ရင် web server က ဘယ် user က ဘာ actions တွေလုပ်သွားတယ်ဆိုတာကို မမှတ်မိပါဘူး။ server က request ပေါ် မူတည်ပြီး requested page ကို ပဲ respond ပြန်ပေးပါတယ်။

ဥပမာ ဆိုပါစို့ ။ user တစ်ယောက် က facebook ကို သူ့ရဲ့ email address နဲ့ password နဲ့ ဝင်တယ်။ အဲ့ဒီ ဝင်နေတဲ့ အချိန်မှာ server က သူဘယ်သူလဲဆိုတာသိပြီးတော့ သူနဲ့ သက်ဆိုတဲ့ posts တွေ သူ့ရဲ့

friends posts တွေမြင်ရမယ်။ နောက်ပြီး သူ့ရဲ့ profile ကို update လုပ်ချင်တာ တစ်ယောက်ယောက်ကို message ပို့တာတို့ လုပ်နိုင်ပါတယ်။ ဒါကလည်း session ရဲ့ သဘောတရားပါပဲ။

User တစ်ယောက်က သူ့ရဲ့ account နဲ့ web application တစ်ခုကို login ဝင်မယ် ။ အဲ့ဒီ အချိန်မှာ အဲ့ဒီ user အတွက် session တစ်ခုဖန်တီးလိုက်မယ်။ အဲ့ဒီ session ထဲမှာ userame သို့မဟုတ် userid နဲ့ အခြား user နဲ့ ပတ်သတ်တဲ့ unique data တွေကို store လုပ်ထားလိုက်မယ်။ အဲ့ဒီ data တွေကို အသုံးပြုပြီးတော့ user နဲ့ ပတ်သတ်တဲ့ information တွေပြမယ်။ session က website ရဲ့ pages တွေတိုင်းမှာ အသုံးပြုနိုင်ဖို့ information တွေကို sharing လုပ်ပေးပြီး state ကို maintain လုပ်ထားပါတယ်။ အဲ့ဒီအခါ session တွေက server ကို request တွေအကုန်လုံးက user တစ်ယောက်ထဲလုပ်နေသိနိုင်ပြီးတော့ သူ လုပ်တဲ့ request နဲ့ ပတ်သတ်ပြီးတဲ့ information တွေ peferences တွေကို ပြပေးနိုင်မှာ ဖြစ်ပါတယ်။ logout လုပ်လိုက်ရင်တော့ session လဲ destory ဖြစ်သွားပါတယ်။

Session တစ်ခုဖန်တီးပြီး Information တွေသိမ်းဖို့အတွက် ဘယ်လောက် limit ထားရမယ်လို့ ကန့်သတ်ချက်မရှိပါဘူး။

Start a Session in PHP

Session တစ်ခုကို ဖန်တီးတော့မယ်ဆိုရင် PHP မှာ session_start() function နဲ့ ဖန်တီးလို့ရပါတယ်။

```
<?php
// start a session
session_start();
?>
```

ဒီမှာ အရေးကြီးတာတစ်ခုရှိပါတယ်။ session_start() function ကို PHP script ရဲ့ ပထမဆုံး အကြောင်းမှာ called လုပ်ရပါမယ်။ အဲ့ဒီလို မလုပ်ခင် output တစ်ခုခု sent လိုက်ရင် Header are

already sent ဆိုပြီး error တက်ပါတယ်။

Automatically Start a Session

အကယ်၍ application က သုံးတဲ့တစ်လျှောက်လုံး session လိုတယ်ဆိုရင် application စကတည်းက session ကို session_start function ကို မသုံးဘဲ auto-start လုပ်ချင်တယ်ဆိုရင် PHP ရဲ့ php.ini မှာ option ကို configuration လုပ်လို့ရပါတယ်။ အဲ့ဒီလိုဆိုရင် request တိုင်းမှာ session ကို auto-start လုပ်သွားမှာပါ။ php.ini file ရဲ့ session.auto_start က ပုံမှန်တိုင်းဆိုရင် 0 ဖြစ်နေပြီး auto startup လုပ်ဖို့ဆိုရင် သူ့ကို 1 ပေးထားရပါမယ်။

```
session.auto_start = 1
```

အထက်မှာ ဖော် ပြခဲ့တဲ့အတိုင်း session တစ်ခုcreate လုပ်တိုင်း unique number တစ်ခုကို server က create လုပ်ပါတယ်။ သူ့ကို session id လို့ခေါ်ပါတယ်။ အဲ့ဒီ session_id ကို အသုံးပြုချင်ရင် session_id() ဆိုတဲ့ function နဲ့ ရယူနိုင်ပါတယ်။

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

ဒီလိုဆိုရင် current session id ကို ရပါတယ်။ session_id function ကို အသုံးပြုပြီးတော့ system ရဲ့ generated session ကို ကိုယ်ပိုင် id နဲ့ replace လုပ်လို့ရပါတယ်။

```
<?php
session_id(YOUR_SESSION_ID);
session_start();
```

```
?>
```

session_id() က argument တစ်ခုယူပါတယ်။ အဲ့ဒီ argument နေရာမှာ ရဲ့ own id ထည့်ရမှာ ဖြစ်ပါတယ်။ ဒီနေရာမှာ မှတ်စရာတစ်ခုက session_id() function ကို session_start() function ရဲ့ session id ကို generate မလုပ်ခင်မှာ အသုံးပြုရပါမယ်။

How to Create Session Variables

Session ကို start လုပ်လိုက်တဲ့အချိန်မှာ session information တွေကို initialized လုပ်ဖို့ $_SESSION[] ဆိုတဲ့ super-global array ကို လည်း တစ်ခါတည်း တည်ဆောက်လိုက်ပါတယ်။ ပုံမှန်အားဖြင့်ဆိုရင် array အလွတ် တစ်ခု ဖန်တီးပြီးတော့ information တွေသိမ်းချင်တဲ့အခါမှာ key-value pair ပုံစံနဲ့ store လုပ်လို့ရပါတယ်။ ဒီလိုပါ။

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

// set the session variable
$_SESSION["username"] = "iamjonny";
$_SESSION["userid"] = "1";
?>

<html>
   <body>
```

```
    <?php
    echo "Session variable is set.";
    ?>

    <a href="second_page.php">Go to Second Page</a>

    </body>
</html>
```

ဒီမှာ ဆိုရင် session ကို start လုပ်ဖို့အတွက် script ရဲ့ very first line မှာ session_start() function ကို အသုံးပြုထားပါတယ်။ session ထဲမှာ information တွေသိမ်းချင်တဲ့အတွက် $_SESSION ဆိုတဲ့ session variable ထဲမှာ သိမ်းပါတယ်။

Getting PHP Session Variable Values

အထက်ပါ example မှာဆိုရင် session က start လုပ်ထားပြီး session variables လေးတွေ create လုပ်ထားပါတယ်။ အပေါ်မှာ ဖော်ပြခဲ့တဲ့အတိုင်း session information တွေကို requests တွေတိုင်းမှာ အသုံးပြုလို့ရပါတယ်။ ဆိုလိုတာက session variables တွေကို page
တစ်ခု မှာ initialize လုပ်ပြီးရင် session ရှိတဲ့ page တိုင်းမှာအသုံးပြုလို့ရပါတယ်။

seconde_page.php

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

// get the session variable values
```

```
$username = $_SESSION["username"];
$userid = $_SESSION["userid"];
?>

<html>
   <body>

   <?php
   echo "Username is: ".$username."<br/>";
   echo "User id is: ".$userid;
   ?>

   </body>
</html>
```

session_start() function called ပြီးတာနဲ့ session variables တွေ access လုပ်ပြီးတော့ အသုံးပြုလို့ရပါတယ်။ တစ်ခါလေ session variables တွေ များစွာတည်ဆောက်တဲ့အခါ session တွေ ကို သိချင်တဲ့အခါ $_SESSION global array ကို တိုက်ရိုက် output ထုတ်ကြည့်လို့ရပါတယ်။

text.php

```
<?php
  print_r($_SESSION);
?>
```

ဒီလိုဆို session variable အားလုံးကြည့်လို့ရပါတယ်။

How to Modify and Delete Session Variables

တည်ဆောက်ထားတဲ့ session variable တွေကို modify ပြန်လုပ်လို့ရသလို delete ပြန်လုပ်လို့လည်းရပါတယ်။

```
session.php
<?php
// start the session
session_start();
// update the session variable values
$_SESSION["userid"] = "1111";
?>
<html>
   <body>
   <?php
   echo "Username is: ".$username."<br/>";
   echo "User id is: ".$userid;
   ?>
   </body>
</html>
```

ဒီလိုဆိုရင် session variable userid က update ဖြစ်သွားပါလိမ့်မယ် ။ နောက်ထပ် example တစ်ခုထပ်ပြပါ့မယ်။

```
other.php
<?php
session_start();
if (!isset($_SESSION['count']))
```

```
{
  $_SESSION['count'] = 1;
}
else
{
  ++$_SESSION['count'];
}
echo $_SESSION['count'];
?>
```

ဒီ example မှာ ဆိုရင် $_SESSION[‘count’] variable ကို မရှိရင် set လုပ်ထားပါတယ်။ ပြီးတော့ 1 ကို assign လုပ်ထားပါတယ်။ အကယ်၍ ရှိတယ်ဆိုရင် increment 1 လုပ်ထားပါတယ်။ ဒါကြောင့် page ကို refresh လုပ်တိုင်းမှာ $_SESSION[‘count’] variable မှာ 1 တိုးသွားပါလိမ့်မယ်။
Session variable တွေကို delete လုပ်ချင်တဲ့အခါမှာ unset function နဲ့ delete လုပ်လို့ရပါတယ်။

```
session_delete.php
<?php
// start a session
session_start();

// initialize a session variable
$_SESSION['username'] = 'imjonny';

// unset a session variable
unset($_SESSION['username']);
?>
```

ဒီလိုဆို $_SESSION['username'] ကို ထပ်ပြီးသုံးလို့မရတော့ပါဘူး။ ဒါက session variable တွေ delete လုပ်ချင်တဲ့အခါ အသုံးပြုပါတယ်။

How to Destroy a Session

အထက်မှာ ဖော်ပြ ခဲ့တဲ့အတိုင်း ကိုယ် delete လုပ်ချင်တဲ့ session variable တွေကို unset() နဲ့ delete လုပ်ပါတယ်။ အကယ်၍ session variable တွေအပါအဝင် session တစ်ခုလုံးကို delete လုပ်ချင်တဲ့အခါ session_unset() နဲ့ session_destroy() function ကို အသုံးပြုလို့ရပါတယ်။ Session_destroy() function က session တစ်ခုလုံးကို destroy လုပ်လိုက်ပါတယ်။

```
<?php
// start the session
session_start();
?>
<html>
<body>
        <?php
        // clean the session variable
        session_unset();
        // destroy the session
        session_destroy();
        ?>
</body>
</html>
```

ဒီလိုဆိုရင် session တစ်ခုလုံး destroy ဖြစ်သွားပြီး session ကော session variable တွေကို

ထပ်အသုံးပြုလို့မရတော့ပါဘူး။ ဒီ function တွေကို website တွေရဲ့ logout or checkout pages တွေမှာ အသုံးပြုပါတယ်။

အခန်း(၁၁) - PHP Cookies

cookie ဆိုတာက small text file လေးတစ်ခုဖြစ်ပြီးတော့ 4kb အများဆုံးရှိတဲ့ data ကို user computer ပေါ်မှာ store လုပ်တဲ့ အလုပ်လုပ်ပါတယ်။ ပုံမှန်အားဖြင့် cookie တွေကို information တွေကို tracking လုပ်ဖို့အတွက် အသုံးများပါတယ်။ ဥပမာ user တစ်ယောက် website ကို visit လုပ်ထားရင် နောက်တစ်ခါ အဲ့ဒီ website ကို ပြန်လာတဲ့အခါ psersonalize လုပ်ဖို့အတွက် username ကိုသော်လည်းကောင်း retrive ပြန်လုပ်ပြီး ဒီ user ပဲဖြစ်ကြောင်းကို tracking လုပ်ဖို့အတွက် အသုံးပြုပါတယ်။

Browser ကနေ page တစ်ခုကို request လုပ်တိုင်းမှာ server ကို cookie ထဲမှာ ရှိတဲ့ data တွေကို request နဲ့အတူ ပို့ဆောင်ပါတယ်။ ဒါကြောင့် sensitive ဖြစ်တဲ့ data တွေကို cookies ထဲမှာ မထားသင့်ပါဘူး ။ ဒီ issues တွေကြောင့် session ကို အသုံးများတာပါ။

Cookie တစ်ခုဖန်တီးချင်တယ်ဆိုရင် setcookie() ဆိုတဲ့ function ကိုအသုံး ပြုလို့ရပါတယ်။

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

ဒီမှာဆိုရင် name နဲ့ value က လိုအပ်ပါတယ်။ ကျန်တဲ့ parameters တွေက optional ပါ။
Name က ကို ပြုလုပ်ချင်တဲ့ cookie ရဲ့ name ပါ။
Value က cookie ရဲ့ value ပါ။ ဒီမှာ sensitive information တွေ ထားခြင်းကို ရှောင်ကျဉ်ပါ။
Expires က cookie ရဲ့ expiry date ပါ။ expiry date ကို UNIX timestamp format နဲ့ထားရမယ်။ default value က 0 ပါ။
Path က cookie ကို access လုပ်မဲ့ server ရဲ့ path ကို ထားပါတယ်။ pathတိုင်းမှာ access ဖြစ်ချင်ရင် '/' ပဲထားပါတယ်။
Domain က cookie ရနိုင်မဲ့ domain ကိုထည်းပေးပါတယ်။ e.g www.example.com
Secure က HTTPS ဖြစ်တဲ့ secure connection ကိုပဲ cookie send မယ်လို့ဆိုလိုတာပါ။
httponly က htttponly ဟုတ်လား မဟုတ်လား ထည့်ပေးရပါမယ်။

```
<?php
// Setting a cookie
setcookie("username", "John Carter", time()+30*24*60*60);
?>
```

Cookie value တွေကို ပြန်ပြီး access လုပ်ချင်တယ်ဆိုရင် $_COOKIE superglobal array ကို အသုံးပြုပါတယ်။

```
<?php
// Verifying whether a cookie is set or not
if(isset($_COOKIE["username"])){
    echo "Hi " . $_COOKIE["username"];
} else{
```

```
    echo "Welcome Guest!";
}
?>
```

Cookie value ကို ပြန်ပြီး delete လုပ်ချင်ရင် setcookie() function မှာ username value ကို “” ထားခဲ့ပြီးတော့ expire date ကို past ဖြစ် အောင်လုပ်ရပါတယ်။ ဒီလိုပါ။

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

<?php
echo "Cookie 'user' is deleted.";
?>
</body>
</html>
```

ဒီလိုဆို cookie value က ပျက်သွားပါတယ်။

အခန်း(၁၂) - JSON

JSON မသွားခင် Object အကြောင်းလေးရှင်းပြပါအုံးမယ်။ Object ဆိုတာ Class ရဲ့ instance တစ်ခုဖြစ်ပါတယ်။ object တစ်ခုက key-value ပုံစံနဲ့ data collection တစ်ခုဖြစ်ပါတယ်။ data တွေကို curly bracket {} တွေနဲ့ store လုပ်ပါတယ်။

Multiple data တွေကို, လေးတွေနဲ့ collect လုပ်တယ်။

Object တစ်ခုကို တိုက်ရိုက်တည်ဆောက်လို့မရပါဘူး။ ဒါကြောင့် array ကို object ပုံစံပြောင်းပြီး အသုံးပြုလို့ရပါတယ်။ ဒီမှာပြန်ရှင်းတဲ့အကြောင်းက JSON ကိုသွားဖို့ Object အကြောင်း သေခြာသိရပါမယ်။

```
$object = (object) ['property' => 'value'];
```

ဒီလိုဆို array ကနေ object ပုံစံပြောင်းသွားပါလိမ့်မယ်။

```
$marks=(object)array("Peter"=>65,"Harry"=>80,"John"=>78,"Clark"=>90);
```

ဒီလိုဆို $mark က object ဖြစ်သွားပါလိမ့်မယ်။ object ရဲ့ value တွေရယူချင်တဲ့အခါမှာ properties name တွေနဲ့ ယူရပါတယ်။
$marks->Peter ဆိုရင် Peter ရဲ့ value ကိုရပါလိမ့်မယ်။
နောက်တစ်နည်း အနေနဲ့ new stdClass() ကို သုံးပြီးတော့လည်း တည်ဆောက်လို့ရပါတယ်။
$object = new stdClass();
ဒီလိုဆိုလည်း $object တစ်ခုတည်ဆောက်တာဖြစ်ပါတယ်။ သူရဲ့ propery နဲ့ value တွေထည့်ချင်တဲ့အခါ
$object->”Peter” = 65 ဆိုပြီး ထည့်ရပါတယ်။

JSON ဆိုတာ lightweight data-interchange format တစ်ခုဖြစ်ပါတယ်။ သူရဲ့ အရှည်ခေါက် JavaScript Object Notation ဖြစ်ပါတယ်။ JSON က standard text-based data format ဖြစ်ပြီးတော့ JavaScript object syntax ကနေ ဆင်းသက်လာတာဖြစ်ပါတယ်။ ဒါကြောင့် သူ့ရဲ့ data တွေ key-values pairs ပုံစံ ဖြစ်ပါတယ်။ JSON ကို web clients နဲ့ web servers ကြားမှာ data တွေကို exchange လုပ်ဖို့အတွက် အသုံးပြုပါတယ်။

```
// JSON syntax
{
   "name": "John",
   "age": 22,
   "gender": "male",
}
```

သူရဲ့ syntax က ဒီလိုပါ။ JSON format ကို အခြား programming language တွေက access လုပ်ပြီး create လုပ်နိုင်တာကြောင့် မတူညီတဲ့ platforms တွေပေါ်မှာ data တွေ ကို send ပြီး receive လုပ်ဖို့

အသုံးပြုကျပါတယ်။ အထက်မှာ ဖော်ပြခဲ့တဲ့ data-interchange format ဆိုတာက test format ဖြစ်ပြီးတော့ မတူညီတဲ့ platforms နဲ့ operating systems တွေကြားမှာ data တွေကို interchange or exchange လုပ် ဖို့အတွက အသုံးပြုပါတယ်။ ဒါကြောင့် lightweight data-interchange format ဖြစ်တဲ့ JSON က popular အဖြစ်ဆုံးဖြစ်ပါတယ်။ JSON က JavaScript object properties နဲ့ ဆင်တာကြောင့် သူ့ကို ရေးရင်လည်း key-value ပုံစံလေးနဲ့ ရေးပါတယ်။

```
// JSON data
"name": "John"
```

ဒီနေရာမှာ မှတ်ထားရမှာ တစ်ခုက key ကို double quote နဲ့ ရေးရပါတယ်။ JSON data က Object နဲ့ Array structure နစ်ခုပေါ် အခြေခံပါတယ်။
အပေါ် မှာ JSON ရဲ့ key တွေက double quote ထဲမှာ ရေးတဲ့ အတွက် string တွေဖြစ်ပါတယ်။ value ကတော့ string , number ,boolean , null , object နဲ့ array ကြိုက်တာ ဖြစ်လို့ရပါတယ်။ ဒီလိုပါ။

```
{
   "book": {
      "name": "Harry Potter and the Goblet of Fire",
      "author": "J. K. Rowling",
      "year": 2000,
      "genre": "Fantasy Fiction",
      "bestseller": true
   }
}
```

```
{
   "fruits": [
      "Apple",
      "Banana",
```

```
    "Strawberry",
    "Mango"
  ]
}
```

Parsing JSON with PHP

JSON data structure က PHP arrays နဲ့ အတော်လေးဆင်တူပါတယ်။ JSON data တွေကို encode and decode လုပ်ဖို့အတွက်PHP မှာ built-in functions တွေရှိပါတယ်။ json_encode() နဲ့ json_decode() ဖြစ်ပါတယ်။ ဒီ function နှစ်ခုလုံးက UTF-8 encoded string data နှစ်ခုလုံးပေါ်မှာ အလုပ်လုပ်ပါတယ်။

Encoding JSON Data in PHP

json_encode() က value တစ်ခုကို JSON format encode လုပ်တဲ့နေရာမှာ အသုံးပြုပါတယ်။ data တွေအကုန်လုံးကို resource data type မှာ လွဲ၍ encode လုပ်လို့ရပါတယ်။

```
<?php
// An associative array
$marks = array("Peter"=>65, "Harry"=>80, "John"=>78, "Clark"=>90);

echo json_encode($marks);
?>
```

ဒါဆိုရင် $marks ကို json အဖြစ် encode လုပ်ပြီးတော့ output ထုတ်ကြည့်ရင် ဒီလိုရပါလိမ့်မယ်။

{"Peter":65,"Harry":80,"John":78,"Clark":90}

ဒါက associative array တစ်ခုကို encode လုပ်တာပါ။

```
<?php
// An indexed array
$colors = array("Red", "Green", "Blue", "Orange", "Yellow");

echo json_encode($colors);
?>
```

ဒီလို index array တစ်ခုကို encode လုပ်ပြီး output ထုတ်ကြည့်တဲ့အခါ
["Red","Green","Blue","Orange","Yellow"] ဒီလိုရပါတယ်။
Index array ကို associative array ပုံစံ encode လုပ်ချင်တဲ့အခါ JSON_FORCE_OBJECT ကို အသုံးပြုလို့ရပါတယ်။

```
<?php
// An indexed array
$colors = array("Red", "Green", "Blue", "Orange", "Yellow");

echo json_encode($colors, JSON_FORCE_OBJECT);
?>
```

ဒီလိုဆို {"0":"Red","1":"Green","2":"Blue","3":"Orange"} ဆိုပြီး ရပါလိမ့်မယ်။

Decoding JSON Data in PHP

JSON data ကို decode လုပ်တာလည်း encoding လုပ်တာနဲ့ဆင်ပါတယ်။ PHP ရဲ့ json_decode() function က encode လုပ်ထားတဲ့ JSON data တွေကို PHP ရဲ့ သက်ဆိုင်ရာ PHP

data type တွေအဖြစ် ပြောင်းပေးပါတယ်။ ပုံမှန်တိုင်းဆိုရင် JSON object တွေကို PHP Object အဖြစ် ပြောင်းပေးပါတယ်။

```
<?php
// Store JSON data in a PHP variable
$json = '{"Peter":65,"Harry":80,"John":78,"Clark":90}';


var_dump(json_decode($json));
?>
```

ဒါဆိုရင် output က

object(stdClass)#1 (4) { ["Peter"]=> int(65) ["Harry"]=> int(80) ["John"]=> int(78) ["Clark"]=> int(90) } ဒီလိုရနေပါလိမ့်မယ်။ default အားဖြင်ဆိုရင် json_decode() က PHP ရဲ့ Object ပုံစံပဲ return ပြန်တာကြောင့် associative array ပုံစံ return ပြန်စေချင်တဲ့အခါ json_decode($json,true) ဆိုပြီး အသုံးပြုပါတယ်။

```
<?php
// Store JSON data in a PHP variable
$json = '{"Peter":65,"Harry":80,"John":78,"Clark":90}';


var_dump(json_decode($json,true ));
?>
```

ဒီလိုဆို

array(4) { ["Peter"]=> int(65) ["Harry"]=> int(80) ["John"]=> int(78) ["Clark"]=> int(90) } associative array ပုံစံ return ပြန်ပါလိမ့်မယ်။

```
<?php
// Assign JSON encoded string to a PHP variable
$json = '{"Peter":65,"Harry":80,"John":78,"Clark":90}';

// Decode JSON data to PHP associative array
$arr = json_decode($json, true);

// Loop through the associative array
foreach($arr as $key=>$value){
    echo $key . "=>" . $value . "<br>";
}
echo "<hr>";
// Decode JSON data to PHP object
$obj = json_decode($json);

// Loop through the object
foreach($obj as $key=>$value){
    echo $key . "=>" . $value . "<br>";
}
?>
```

How To Read A JSON File Using PHP

JSON ဆို အထက်မှာ ဖော်ပြခဲ့တဲ့ အတိုင်းပဲ သူက file format တစ်ခုဖြစ်ပါတယ်။ သူ့ရဲ့ extension က .json ဖြစ်ပါတယ်။ e.g. data.json. အဲ့ဒီ json file တစ်ခုမှာ ဆိုရင် JavaScript Object Notation style နဲ့ data တွေပါပါတယ်။

```
data.json
[
{"id":"12","name":"John","salary":"54000","age":"65"},
{"id":"13","name":"Harry","salary":"12350","age":"29"},
{"id":"14","name":"Smith","salary":"90000","age":"18"}
]
```

ဒီ data.json ထဲက data တွေကို အသုံးပြုချင်တဲ့အခါ PHP မှာ file ကနေ string အဖြစ် ပြောင်းလည်းဖတ်ပေးတဲ့ file_get_contents() ဆိုတဲ့ built-in function ရှိပါတယ်။ file_get_contents() function က file တင်မကပဲ API URLs ကနေ data တွေ ရယူပြီး variable အဖြစ် store လုပ်ချင်တဲ့အခါမှာလည်း အသုံးပြုလို့ရပါတယ်။ json file ကနေ data တွေရယူချင်တဲ့အခါမှာ ဒီလိုအသုံးပြုလို့ရပါတယ်။

```
<?php
// Read JSON file
$json_data = file_get_contents($api_url);
// Decode JSON data into PHP array
$response_data = json_decode($json_data);
// Cut long data into small & select only first 10 records
$user_data = array_slice($user_data, 0, 9);
// Print data if need to debug
//print_r($user_data);
// Traverse array and display user data
foreach ($user_data as $user) {
        echo "name: ".$user->employee_name;
        echo "<br />";
        echo "name: ".$user->employee_age;
```

```
        echo "<br /> <br />";
}
?>
```

ဒီမှာဆိုရင် file_get_contents() function က json file က data တွေ fetching လုပ်ပေးပါတယ်။ ရလာတဲ့ data တွေကို PHP array အဖြစ်ပြောင်းဖို့ json_decode() နဲ့ decoding လုပ်လိုက်ပါတယ်။ decode လုပ်ပြီး ရလာတဲ့ array ကို foreach နဲ့ iterate လုပ်ထားပါတယ်။ API URLs ကနေ ဖတ်ချင်တဲ့အခါ ဒီလိုရေးပါတယ်။

```
<?php

$api_url = 'https://jsonplaceholder.typicode.com/users';

// Read JSON file
$json_data = file_get_contents($api_url);

// Decode JSON data into PHP array
$response_data = json_decode($json_data);
. . .
. . .
```

file_get_contents() function က API or json file တွေက data တွေကို read ပဲ လုပ်နိုင်ပါတယ်။ write, update, delete စတဲ့ operation တွေ လုပ်လို့မရပါဘူး ။

အခန်း(၁၃) - PHP MySQL

What is MySQL

MySQL က PHP မှာ အသုံးပြုများတဲ့ database system တစ်ခုဖြစ်ပါတယ်။ PHP မှာ MySQL နဲ့ ဆိုင်တဲ့ functions ပေါင်းများစွာ support ထားပါတယ်။ website တော်တော်များများကို PHP နဲ့ MySQL နဲ့ ပေါင်းပြီးရေးကျပါတယ်။ MySQL က free ဖြစ်ပြီးတော့ လွယ်လွယ်ကူကူ install လုပ်လို့ရပါတယ်။ အကယ်၍ wampserver or xampp ကို အသုံးပြုတယ်ဆိုရင် MySQL က တွဲပါလာပြီးသားဖြစ်ပါတယ်။ MySQL က Relational Database Management System(RDMS) တစ်ခုဖြစ်ပြီးတော့ SQL (Structured Query Language) ကို အသုံးပြုပါတယ်။ MySQL က အသုံးပြုရလွယ်ကူပြီးတော့ secure ဖြစ်တယ် ၊ scalable လည်းဖြစ်ပါတယ်။ MySQL က Linux, Windows, Mac OS X စတဲ့ operating

systems တွေတော်တော်များများမှာ runs နိုင်ပါတယ်။ နောက်ပြီး Applications တော်တော်များများရဲ့ database အဖြစ် MySQL ကို အသုံးပြုပါတယ်။ MySQL မှာ sensitive data တွေကို protect လုပ်ဖို့အတွက် data security layers များစွာပါဝင်ပါတယ်။ MySQL database က အခြား relational database တွေလိုပဲ data တွေကို tables တွေ အဖြစ် store လုပ်ပါတယ်။ table ဆိုတာက data collection တစ်ခုဖြစ်ပြီးတော့ rows တွေ columns တွေ နဲ့ ဖွဲ့စည်းထားပါတယ် ။ table ရဲ့ row တစ်ခုတိုင်းဆီက data record တစ်ခု အဖြစ် ကိုယ်စားပြုပါတယ်။ rows တွေက တစ်ခုနဲ့ တစ်ခု ဆက်စပ်နေပါတယ်။ columns တွေကတော့ field တွေဖြစ်တဲ့ id , first_name , last_name , email စတာတွေကို ကိုယ်စားပြုပါတယ်။ ဒီလိုပုံစံလေးပါ။

```
+----+------------+-----------+---------------------+
| id | first_name | last_name | email               |
+----+------------+-----------+---------------------+
|  1 | Peter      | Parker    | peterparker@mail.com |
|  2 | John       | Rambo     | johnrambo@mail.com   |
|  3 | Clark      | Kent      | clarkkent@mail.com   |

+----+------------+-----------+---------------------+
```

MySQL Databases with SQL

SQL ဆိုတာ Structured Query Language ဖြစ်ပြီးတော့ MySQL လို relational databases တွေနဲ့ တွဲအလုပ်လုပ်တဲ့ standardized language ဖြစ်ပါတယ်။ SQL က database နဲ့ သက်ဆိုင်တဲ့ task ဖြစ်တဲ့ database နဲ့ tables တွေ create လုပ်တာ ၊ database tables ထဲမှာ data တွေကို သိမ်းတာ ၊ database tables ထဲက data တွေကို access လုပ်ပြီး delete ၊ update တွေ လုပ်ပါတယ်။

Connect to MySQL Server

MySQL database ထဲမှာ data တွေ ထည့်ဖို့ data တွေ access လုပ်ဖို့ဆိုရင် MySQL Server နဲ့ ချိန်ဆက်ရပါ့မယ်။ MySQL server ကို ချိတ်ဆက်ဖို့အတွက် PHP မှာ နည်းနှစ်နည်းရှိပါတယ်။ MySQLi (i mean improved MySQL) နဲ့ PDO(PHP Data Objects) extensions တွေပါ။

PDO extension က portable ပိုဖြစ်တယ်၊ databases 12 မှာ support တယ်ဆိုပေမဲ့ MySQLi extension က MySQL database မှာပဲ အသုံးပြုလို့ရပါတယ်။ databases ကို ချိတ်ဆက်တဲ့ နေရာမှာ

object-oriented way သို့မဟုတ် procedural way ဆိုပြီး ချိတ်ဆက်ပုံနှစ်မျိုးရှိပါတယ်။ PDO extension or MysQLi extension နှစ်ခုလုံးက နည်းနှစ်နည်းလုံးကို သုံးပြီး ချိတ်ဆက်လို့ရပေမဲ့ MySQLi extension က MySQL database server နဲ့ ချိတ်ဆက်ရလွယ်ကူပြီးတော့ beginner တွေအနေနဲ့ ပိုနားလည်လွယ်ပါတယ်။ ဒီမှာ MySQLi extension အသုံးပြုပြီးတော့ procedural way နဲ့ database ကို ချိတ်ဆက်အသုံးပြုပါမယ်။

PHP မှာ MySQL database ကို ချိတ်ဆက်ဖို့အတွက် mysqli_connect() ဆိုတဲ့ function ရှိပါတယ်။ mysqli_connect() နဲ့ PHP နဲ့ MySQL database ကြား connection တစ်ခုဖန်တီးပြီး database နဲ့သက်ဆိုင်တဲ့အလုပ်တွေအကုန်လုံးက အဲ့ဒီ connection ကိုဖြတ်ပြီး အလုပ်လုပ်ပါတယ်။

```
<?php
$mysqli_link = mysqli_connect("{HOST_NAME}", "{DATABASE_USERNAME}", "{DATABASE_PASSWORD}", "{DATABASE_NAME}");

if (mysqli_connect_errno())
{
   printf("MySQL connection failed with the error: %s", mysqli_connect_error());
    exit;
}
```

Or

```
$link = mysqli_connect("hostname", "username", "password", "database");

// Check connection
if($link === false){
    die("ERROR: Could not connect. " . mysqli_connect_error());
}
```

ဒီနေရာမှာ hostname က ကျနော်တို့ MySQL server's ရဲ့ host-name or IP address ကိုဆိုလိုပါတယ်။ ဒီမှာ က MySQL server နဲ့ PHP ကို system တစ်ခုထဲမှာ အတူဖြစ်တဲ့အတွက်localhost or 127.0.0.1 ကိုအသုံးပြုလို့ရပါတယ်။ MySQL server ကို externally host အဖြစ်အသုံးပြုတဲ့ဆိုရင် သက်ဆိုင်တဲ့ host-name or IP address ကို အသုံးပြုပေးရပါမယ်။

Database_username ဆိုတာက MySQL ကို အသုံးပြုတဲ့ username ကို ဆိုလိုတာပါ။ ပုံမှန်အားဖြင့်ဆို root ဖြစ်ပါတယ်။ ဒီ username နဲ့ MySQL server ကို ချိတ်ဆက်မှာဖြစ်ပါတယ်။

Database_password ကို MySQL server အသုံးပြုတဲ့ user ရဲ့ password ဖြစ်ပါတယ်။ ဒီ username နဲ့ password က MySQL server ချိတ်ဆက်တိုင်း အသုံးပြုနေရမှာဖြစ်ပါတယ်။

Databasename က ချိတ်ဆက်အသုံးပြုချင်တဲ့ MySQL database name ဖြစ်ပါတယ်။ connection တစ်ခုကို ဖန်တီးပြီးလျှင် databaseနဲ့သက်ဆိုင်တဲ့ operations တွေ query တွေ စလုပ်လို့ရပြီဖြစ်ပါတယ်။

Creating Tables in the database

Data တွေကို database ထဲမှာ store လုပ်ချင်တယ်ဆိုရင် table တွေတည်ဆောက်ပြီး သိမ်းရပါတယ်။ table တွေကို row-column ပုံစံတွေနဲ့ ဖွဲ့စည်းထားပါတယ်။

Database ထဲမှာ table တစ်ခုတည်ဆောက်ချင်တယ်ဆိုရင် SQL ရဲ့ CREATE TABLE statement ကို အသုံးပြုရပါတယ်။ CREATE TABLE statement အသုံးပြုထားတဲ့ SQL query ကို mysqli_query() function ထဲကို passing လုပ်ပြီး table ကို တည်ဆောက်ရပါတယ်။

```
<?php
$link = mysqli_connect("localhost", "root", "", "demo");


// Check connection
if($link === false){
    die("ERROR: Could not connect. " . mysqli_connect_error());
```

```
}

// Attempt create table query execution
$sql = "CREATE TABLE IF NOT EXISTS persons(
    id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
    first_name VARCHAR(30) NOT NULL,
    last_name VARCHAR(30) NOT NULL,
    email VARCHAR(70) NOT NULL UNIQUE
)";
if(mysqli_query($link, $sql)){
    echo "Table created successfully.";
} else{
    echo "ERROR: $sql. " . mysqli_error($link);
}

// Close connection
mysqli_close($link);
?>
```

ဒါဆိုရင် id, first_name, last_name, email ဆိုတဲ့ four columns ပါဝင်တဲ့ persons ဆိုတဲ့ table တစ်ခုကို သတ်မှတ်ထားတဲ့ table ထဲမှာ တည်ဆောက်လိုက်ပြီ ဖြစ်ပါတယ်။ ဒီမှာ မှတ်ထားရမှာက field တိုင်းရဲ့ နောက်မှာ data type declaration လေးတွေပါပါတယ်။ဒါက ဘယ် column ဘယ် data တွေယူတယ် ဘာ data type တွေ ယူတယ်ဆိုတာကို ဖော်ပြပေးတာဖြစ်ပါတယ်။ နောက်ပြီး column တွေရဲ့ သက်ဆိုင်ရာ optional attributes လေးတွေပါပါတယ် ။

NOT NULL - ဆိုတာက column မှာ ထည့်မဲ့ row ရဲ့ data က null value မဖြစ်ရပါဘူး မပါမနေရလို့ဆိုလိုတာပါ။

AUTO_INCREMENT - က MySQL က row တစ်ခါ record တစ်ခါ add တိုင်းမှာ field ရဲ့ value ကို auto 1 တိုးအောင် ပြုလုပ်ပေးတာဖြစ်ပါတယ်။
PRIMARY KEY - က unique identity ဖြစ်အောင်လုပ်တာဖြစ်ပါတယ်။ များသောအားဖြစ်ဆိုရင် PRIMARY KEY ကို ID number column နဲ့ တွဲသုံးလေ့ရှိပါတယ်။ တစ်ခြားသော attributes တွေကိုလည်း လိုအပ်သလို အသုံးပြုလို့ရပါတယ်။

Inserting Data into a MySQL Database Table

Database မှာ table တည်ဆောက်ပြီးရင် data တွေ ထည့်ချင်တဲ့အခါမှာ INSERT INTO statement ကို အသုံးပြုရပါတယ်။
သက်ဆိုင်တဲ့ data တွေကို INSERT INTO statement နဲ့ SQL query အသုံးပြုပြီး mysqli_query() နဲ့ passing လုပ်တဲ့အခါမှာ table ထဲကို data တွေ insert လုပ်သွားမှာ ဖြစ်ပါတယ်။

```
<?php
$link = mysqli_connect("localhost", "root", "", "demo");
// Check connection
if($link === false){
    die("ERROR: Could not connect. " . mysqli_connect_error());
}

// Attempt insert query execution
$sql = "INSERT INTO persons (first_name, last_name, email) VALUES ('John', 'Smith', 'johnsmith@mail.com')";
if(mysqli_query($link, $sql)){
    echo "Records inserted successfully.";
} else{
```

```
    echo "ERROR: Could not able to execute $sql. " . mysqli_error($link);
}

// Close connection
mysqli_close($link);
?>
```

ဒီလိုဆို data တွေက table ထဲကို record တစ်ခုအနေနဲ့ row အလိုက်ဝင်သွားပါလိမ့်မယ်။ ဒီနေရာမှတ်ထားရမှာက သူနဲ့ သတ်ဆိုင်တဲ့ data တွေကို နေရာ အလိုက်ထည့်ပေးရပါ့မယ်။ နောက်ပြီး id နေရာမှာ ထည့်ပေးစရာ မလိုပါဘူး။ AUTO_INCREMENT လုပ်ထားတဲ့အတွက် row တစ်ခါ insert လုပ်တိုင်းမှာ MySQL က auto 1 တိုးနေပါလိမ့်မယ်။

Inserting Multiple Rows into a Table

Table ထဲကို တစ်ကြိမ်ထဲနဲ့ multiple rows data တွေထည့်လိုရပါတယ်။ ဒီလိုပါ။

```
<?php
$sql = "INSERT INTO persons (first_name, last_name, email) VALUES
        ('John', 'Rambo', 'johnrambo@mail.com'),
        ('Clark', 'Kent', 'clarkkent@mail.com'),
        ('John', 'Carter', 'johncarter@mail.com'),
        ('Harry', 'Potter', 'harrypotter@mail.com')";
if(mysqli_query($link, $sql)){
    echo "Records added successfully.";
} else{
    echo "ERROR: Could not able to execute $sql. " . mysqli_error($link);
}
```

```
?>
```

Multiple rows insert လုပ်တဲ့အခါမှာ row တစ်ခုချင်းဆီကို parentheses () ထဲ ရေးရပြီး comma (,) နဲ့ ခြားရပါတယ်။ database ရဲ့ table ကို ပြန်ကြည့်တဲ့အခါမှာ data တွေ rows အလိုက်ဝင်နေပြီး id နေရာက data က auto 1 increment ဖြစ်နေတာ တွေ့ရပါလိမ့်မယ် ။

Insert Data into a Database from an HTML Form

Data တွေကို SQL query ဖန်တီးပြီး တိုက်ရိုက် insert လုပ်လို့ရသလို user input form ကနေရတဲ့ input values တွေနဲ့ တွဲပြီးတော့လည်း insert လုပ်လို့ရပါတယ်။ ဆိုလိုတာက input form ကနေ database ထဲကို ထည့်ဖို့ data ရယူမှာ ဖြစ်ပါတယ်။

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Add Record Form</title>
</head>
<body>
<form action="insert.php" method="post">
   <p>
      <label for="firstName">First Name:</label>
      <input type="text" name="first_name" id="firstName">
   </p>
   <p>
      <label for="lastName">Last Name:</label>
      <input type="text" name="last_name" id="lastName">
```

```
    </p>
    <p>
        <label for="emailAddress">Email Address:</label>
        <input type="text" name="email" id="emailAddress">
    </p>
    <input type="submit" value="Submit">
</form>
</body>
</html>
```

Form ကနေ ပို့လိုက်တဲ့ data တွေ ရယူဖို့အတွက် POST method နဲ့ ပို့ရင် $_POST[] superglobal array နဲ့ ပြန်ပြီး retrieve လုပ်လို့ရသလို အထက်မှာ ဖော်ပြခဲ့သလို POST or GET method နဲ့ data တွေကို ပို့တိုင်းမှာ $_REQUEST[] ဆိုတဲ့ superglobal array ကို auto တည်ဆောက်တာဖြစ်တဲ့အတွက် $_REQUEST[] နဲ့လည်း retrieve ပြန်လုပ်လို့ရပါတယ်။

```
<?php

$link = mysqli_connect("localhost", "root", "", "demo");

// Check connection
if($link === false){
    die("ERROR: Could not connect. " . mysqli_connect_error());
}

// Escape user inputs for security
$first_name=mysqli_real_escape_string($link, $_REQUEST['first_name']);
$last_name = mysqli_real_escape_string($link, $_REQUEST['last_name']);
```

```
$email = mysqli_real_escape_string($link, $_REQUEST['email']);

// Attempt insert query execution
$sql = "INSERT INTO persons (first_name, last_name, email) VALUES ('$first_name', '$last_name', '$email')";
if(mysqli_query($link, $sql)){
    echo "Records added successfully.";
} else{
    echo "ERROR: Could not able to execute $sql. " . mysqli_error($link);
}

// Close connection
mysqli_close($link);
?>
```

ဒီနေရာမှာ data တွေ လက်ခံရယူတဲ့ အခါမှာ mysqli_real_escape_string() function ကို အသုံးပြုထားပါတယ်။ function က string ထဲမှာ ပါလာမဲ့ special characters တွေကို escapes လုပ်ပေးပြီးတော့ SQL injection ရန်ကနေ ကာကွယ်ပေးပါတယ်။ နောက်ပြီး data တွေကို SQL query ထဲကို passing လုပ်တဲ့အခါ string data type တွေကို single quote(‘’) သုံးပေးသင့်ပါတယ်။ တစ်ခါတစ်ရံမှာ error တက်တက်တာကြောင့်ပါ။

PHP MySQL Prepared Statements

Prepared statement ဆိုတာက actual parameter values တွေအစား SQL query မှာ template placeholder တွေနဲ့ ရေးနည်းကို ဆိုလိုပါတယ်။ သူ့ ကို parameterized statement လို့လည်းခေါ်ပါတယ်။ ဒီလိုပါ။

```
INSERT INTO persons (first_name, last_name, email) VALUES (?, ?, ?);
```

Value တွေအစား anonymous positional placeholder တွေနဲ့ အစားထိုးရေးလို့ရပါတယ်။ prepare statement ရဲ့ execution မှာ prepare and execute ဆိုပြီး statement နှစ်ခုပါဝင်ပါတယ်။

Prepare statement မှာ actual parameter value တွေ မပါဘဲ SQL statement template တစ်ခု create လုပ်ပြီး server ကိုပို့ပါတယ်။ အဲ့ဒီအခါ server က statement template ကို parsing ပြန်လုပ်ပြီး syntax မှန်မမှန်, query optimization မှန်မှန်စစ်ဆေးပြီး data တွေ လက်ခံရရှိတဲ့အခါ အသုံးပြုဖို့ ခနသိမ်းထားပါတယ်။

Execute statement မှာ parameter values တွေကို server စီ ပို့လိုက်ပါတယ်။ အဲ့ဒီ အခါ server က ရတဲ့ data တွေနဲ့ သိမ်းထားတဲ့ template နဲ့ ပေါင်းလိုက်ပြီး execution ဖြစ်သွားပါတယ်။

Prepared Statements က တစ်ချို့သော အခြေနေတွေဖြစ်တဲ့ statement တစ်ခုထဲနဲ့ different values တွေကို အကြိမ်ရေများစွာ လုပ်ချင်တဲ့အခါမှာ အရမ်းအသုံးဝင်ပါတယ်။ အသုံးဝင်တဲ့ အကြောင်းလေးထပ်ဖော်ပြပါ့မယ်။

Prepared Statements က တူညီတဲ့ statement တွေကို တစ်ခါတည်းနဲ့ လျှင်လျှင်မြန်မြန် execution လုပ်နိုင်ပါတယ်။ ဘာကြောင့်လဲဆိုတော့ statement တစ်ခု parse လုပ်ပြီးရင် အကြိမ်ကြိမ် execution လုပ်နိုင်လို့ပါ။ နောက်ပြီး execution ဖြစ်ဖို့အတွက် database တွေကို SQL statement တစ်ခုလုံးပို့စရာမလိုဘဲ placeholder values တွေကိုပဲ ပို့ပေးရတာဖြစ်တဲ့အတွက် bandwidth usage ကိုလည်း လျော့ချပေးပါတယ်။ Prepared statements ကို သုံးခြင်အားဖြင့် SQL injection ကို ကောင်းကောင်းကာကွယ်ပေးပါတယ်။ ဘာကြောင့်လဲဆိုတော့ parameter values တွေကို SQL query ထဲမှာ တိုက်ရိုက်ထည့်မထားခြင်းကြောင့် ဖြစ်ပါတယ်။ parameter values တွေကို ပို့တဲ့အခါ query string နဲ့ မပို့ပဲ သီးခြား protocol တစ်ခုနဲ့ သီးခြားပို့တာကြောင့် data တွေကို လွယ်လွယ်ကူကူ ဖျတ်ယူလို့မရပါဘူး။ Server ကလည်း data တွေကို execution point ရောက်မှ တိုက်ရိုက်သုံးတာဖြစ်တဲ့အတွက် error-prone နဲတာကြောင့် database security အတွက် အသုံးဝင်တဲ့ critical element တစ်ခုဖြစ်ပါတယ်။

```
// Prepare an insert statement
$sql = "INSERT INTO persons (first_name, last_name, email) VALUES (?, ?, ?)";
```

```
if($stmt = mysqli_prepare($link, $sql)){
    // Bind variables to the prepared statement as parameters
    mysqli_stmt_bind_param($stmt, "sss", $first_name, $last_name, $email);

    /* Set the parameters values and execute
    the statement again to insert another row */
    $first_name = "Hermione";
    $last_name = "Granger";
    $email = "hermionegranger@mail.com";
    mysqli_stmt_execute($stmt);

    /* Set the parameters values and execute
    the statement to insert a row */
    $first_name = "Ron";
    $last_name = "Weasley";
    $email = "ronweasley@mail.com";
    mysqli_stmt_execute($stmt);
    echo "Records inserted successfully.";
} else{
    echo "ERROR: Could not prepare query: $sql. " . mysqli_error($link);
}
// Close statement
mysqli_stmt_close($stmt);
```

ဒါက statement ကို အသုံးပြုပါ။

ဒီမှာ တစ်ကြောင်းချင်း ရှင်းရလျှင် ပထမဆုံး INSERT INTO statement နဲ့ first_name, last_name, email fields values တွေအစား? အသုံးပြုပြီး create လုပ်ထားပါတယ်။ ပြီးတော့ statement တစ်ခု

prepare လုပ်ပါတယ် ။ mysqli_stmt_bind_param() function က placeholder တွေအဖြစ်နဲ့သုံးထားခဲ့တဲ့? နေရာက values တွေအတွက် variables တွေကို binding လုပ်ပေးပါတယ်။ အဲ့ဒီ အခါ ? တွေက actual values တွေနဲ့ replace လုပ်ခံရပါတယ်။ function ရဲ့ the second argument အနေနဲ့ type definition ကို အသုံးပြုရပါတယ်။ “sss” ဆိုတာက bind လုပ်တဲ့ variable value တွေက string တွေ ဖြစ်ကြောင်း ဖော်ပြတာဖြစ်ပါတယ်။

Type definition အကြောင်း ပြောပြပါ့မယ်။

b - က binary ဖြစ်ပြီးတော့ image, PDF file စတာတွေကို ဆိုလိုတာပါ။

d - က double ဖြစ်ပြီးတော့ floating-point number တွေကို ဆိုလာတာပါ။

i - က integer ဖြစ်ပြီးတော့ number တွေကို ဆိုလိုတာပါ။

s - က string ဖြစ် ပြီးတော့ text တွေကို ဆိုလိုတာပါ။

Binding လုပ်တဲ့ variables တွေနဲ့ type defination ထားပုံနဲ့ တူရပါမယ် ။ပြီးတော့ SQL statement မှာထားတဲ့ placehoder အရေတွက်နဲ့လည်းတူရပါမယ်။ parmeters 4 လုံးလိုပြီး type defination ကို “sss” ဆိုပြီးထားလို့မရသလို integer လိုတဲ့နေရာမှာ “i” ဆိုပြီး ထားရပါမယ်။

Insert Data from an HTML Form with prepared statement

ရှေ့သင်ခန်းစာမှာ user input fields တွေကနေ data တွေ insert လုပ်ခဲ့ပါတယ်။ prepared statement ကို အသုံးပြုပြီးတော့ user input fields တွေ ကနေ data တွေကို insert လုပ်လို့ရပါတယ်။ HTML form ကို ရှေ့သင်ခန်းစာတိုင်း ပြန်ရေးလို့ရပါတယ်။ form action သွားမဲ့ file မှာ ဒီလိုရေးရပါမယ်။

```
// Prepare an insert statement
$sql = "INSERT INTO persons (first_name, last_name, email) VALUES (?, ?, ?)";


if($stmt = mysqli_prepare($link, $sql)){
    // Bind variables to the prepared statement as parameters
    mysqli_stmt_bind_param($stmt, "sss", $first_name, $last_name, $email);
```

```
    // Set parameters
    $first_name = $_REQUEST['first_name'];
    $last_name = $_REQUEST['last_name'];
    $email = $_REQUEST['email'];

    // Attempt to execute the prepared statement
    if(mysqli_stmt_execute($stmt)){
        echo "Records inserted successfully.";
    } else{
        echo "ERROR: Could not execute query: $sql. " . mysqli_error($link);
    }
} else{
    echo "ERROR: Could not prepare query: $sql. " . mysqli_error($link);
}
```

ဒီမှာဆိုရင် form ကနေ data တွေ လက်ခံတဲ့ အခါမှာ mysqli_real_escape_string() function ကို အသုံးပြုမထားပါဘူး။ Prepared statements ကို အသုံးပြုရင် inputs values တွေကို query ကနေ တိုက်ရိုက် submit လုပ်တာ မဟုတ်တဲ့အတွက် input fields တွေကို escaped လုပ်စရာ မလိုတော့ပါဘူး။

PHP MySQL SELECT Query

ရှေ့သင်ခန်းစာတွေမှာ data တွေကို database ရဲ့ table ထဲမှာ insert လုပ်ခဲ့ပါတယ်။ အဲ့ဒီ data တွေကို ပြန်အသုံးပြုဖို့အတွက် data တွေကို retrieve ပြန်လုပ်ရပါမယ်။ data တွေကို data တွေကို retrieve ပြန်လို့ဖို့အတွက် SQL ရဲ့ SELECT statement query ကို အသုံးပြုရပါမယ်။

သူ့ရဲ့ syntax က ဒီလိုပါ။

```
SELECT column1_name, column2_name, columnN_name FROM table_name;
```

Database table ထဲက data တွေကို SELECT statement နဲ့ column တစ်ခုချင်းဆီ retrieve ပြန်ထုတ်လို့ရသလို အကုန်လုံးကို ပြန် retrieve ပြန်ထုတ်ချင်တဲ့အခါ “*” ကို အသုံးပြုလို့ရပါတယ်။ ဒီလိုပါ။

```
$select_query = "SELECT * FROM persons";
$result = mysqli_query($mysqli_link, $select_query);
while ($row = mysqli_fetch_array($result, MYSQLI_ASSOC)) {
    echo "First Name:" . $row['first_name'] . "<br/>";
    echo "Last Name:" . $row['last_name'] . "<br/>";
    echo "Email:" . $row['email'] . "<br/>";
    echo "<br/>";
}
// close the db connection
mysqli_close($mysqli_link);
?>
```

ဒီမှာဆိုရင် SELECT လုပ်လို့ရလာတဲ့ data တွေအတွက် mysqli_fetch_array() function ကို MYSQLI_ASSOC option နဲ့ သုံးထားပါတယ်။ ရလာတဲ့ data တွေကို array အဖြစ် ပြန် fetching လုပ်ချင်တဲ့အတွက် mysqli_fetch_array() function ကို အသုံးပြုပြီး associative array ပုံစံလိုချင်တဲ့အခါ MYSQLI_ASSOC option ကို အသုံးပြုထားပါတယ်။ အခြားသော fetching function တွေလည်းရှိပါတယ်။

Mysqli_fetch_all - က rows တွေကို တစ်ကြိမ်ထဲနဲ့ တစ်ခါတည်း fetch လုပ်ချင်တဲ့အခါမှာ အသုံးပြုပါတယ်။ option အနေနဲ့ associative array ပုံစံထုတ်လို့ရသလို numeric array ပုံစံလည်း ထုတ်လို့ရပါတယ်။ နှစ်မျိုးလုံးသုံးချင်လည်း ရပါတယ်။

Mysqli_fetch_array - က တစ်ခါ retrieve လုပ်တိုင်း row တစ်ခု ရရန်အသုံးပြုပါတယ်။ records တွေအားလုံးကို ထုတ်ဖို့အတွက် while loop ကိုအသုံးပြုရပါမယ်။ options အနေနဲ့ associative array ပုံစံ၊ numeric array ပုံစံ သို့ မဟုတ် နှစ်မျိုးလုံးတစ်ခါတည်းသုံးချင်လည်း ရပါတယ်။

Mysqli_fetch_assoc - က row တစ်ခုချင်းဆီကို retrieve လုပ်တိုင်း associative array ပုံစံနဲ့ တစ်ခါတည်း retrieve လုပ်ချင်တဲ့အခါ အသုံးပြုပါတယ်။

Mysqli_fetch_object - က row တစ်ခုချင်းဆီကို object တစ်ခုချင်းပုံစံနဲ့ retrieve လုပ်ပါတယ်။

mysqli_fetch_assoc() function ကို အသုံးပြုရင် ဒီလိုပါ။

```
$sql = "SELECT id, first_name, last_name FROM persons";
$result = mysqli_query($conn, $sql);
if (mysqli_num_rows($result) > 0) {
  // output data of each row
  while($row = mysqli_fetch_assoc($result)) {
    echo "id: " . $row["id"]. " - Name: " . $row["firstname"]. " " . $row["lastname"]. "<br>";
  }
} else {
  echo "0 results";
}
```

ဒီမှာ ကို retrieve လုပ်ချင်တဲ့ table ထဲမှာ rows ရှိမရှိကို mysqli_num_rows() function ကို အသုံးပြုပြီးတော့ rows အရေအတွက်ကို စစ်လို့ရပါတယ်။ mysqli_fetch_assoc() function ကို အသုံးပြတဲ့အခါ associative array ပုံစံရမှာဖြစ်တဲ့အတွက် options သုံးစရာမလိုပါဘူး။

mysqli_fetch_object() function ကို အသုံးပြုရင်တော့ ဒီလိုပါ။

```
…
…
while ($row = mysqli_fetch_object($result)) {
    echo "First Name:" . $row->first_name . "<br/>";
```

```
    echo "Last Name:" . $row->last_name . "<br/>";
    echo "Email:" . $row->email . "<br/>";
    echo "<br/>";
}
...
...
```

ဒီမှာ object ပုံစံ fetching လုပ်တဲ့အတွက် သူ့ရဲ့ properties တွေရဲ့ value တွေလိုချင်တဲ့အခါ arrow key(->) နဲ့ object ပုံစံ ပြန်ယူရပါမယ်။ ဒီနည်းလမ်းတွေနဲ့ Mysql database table ထဲက data တွေကို retrieve လုပ်လို့ရပါတယ်။

MySQL WHERE Clause

WHERE clause ကို records တွေကို specify လုပ်ချင်တဲ့အခါ အသုံးပြုပါတယ်။ ဆိုလိုတာက records တွေကို condition အရ ကိုကြိုက် နှစ်သက်သလို specify လုပ်နိုင်တာကို ဆိုလိုတာပါ။

```
SELECT column_name(s) FROM table_name WHERE column_name operator value
```

SELECT statement နဲ့ တွဲသုံးတဲ့ syntax ပါ။

```
+----+------------+-----------+----------------------+------------+
| id | first_name | last_name | email                | age        |
+----+------------+-----------+----------------------+------------+
|  1 | Peter      | Parker    | peterparker@mail.com | 19         |
|  2 | John       | Rambo     | johnrambo@mail.com   | 20         |
|  3 | Clark      | Kent      | clarkkent@mail.com   | 30         |
|  4 | John       | Carter    | johncarter@mail.com  | 25         |
|  5 | Harry      | Potter    | harrypotter@mail.com | 21         |
+----+------------+-----------+----------------------+------------+
```

ဒီ table လေးကိုအသုံးပြုပြီးတော့ personရဲ့ အသက်20ကျော်တဲ့သူတွေကို select လုပ် ကြည့်ရအောင် ။

```
$sql = "SELECT id, firs_tname, last_name FROM persons WHERE age > 20";
```

```
$result = mysqli_query($conn, $sql);

if (mysqli_num_rows($result) > 0) {
  // output data of each row
  while($row = mysqli_fetch_assoc($result)) {
    echo "id: " . $row["id"]. " - Name: " . $row["firs_tname"]. " " . $row["last_name"]. "<br>";
  }
} else {
  echo "0 results";
}
```

ဒီလိုဆို persons table ထဲက age 20 ထက်ကြီးတဲ့ records တွေပဲ ထွက်လာတော့မှာ ဖြစ်ပါတယ်။ conditions နှစ်ခုစစ်ချင်တဲ့အခါ ဒီလိုစစ်ပါတယ်။

```
$sql = "SELECT id, firs_tname, last_name FROM persons WHERE age > 20 AND age < 30";
```

ဒီလိုဆို persons table ထဲက age 20 နဲ့ age 30 ကြားထဲက records တွေပဲ ထွက်လာတော့မှာ ဖြစ်ပါတယ်။

PHP MySQL LIMIT Clause

LIMIT Clause ကို SELECT STATEMENT က ရလာတဲ့ result rows တွေကို limit လုပ်ချင်တဲ့အခါ အသုံးပြုပါတယ်။ ဒီ feature က page ကို optimizing လုပ်တဲ့ နေရာမှာ အရမ်းအသုံးဝင်ပါတယ်။ ဆိုလိုတာက ရှိသမျှ data rows တွေကို အကုန် ပြီးတဲ့ထိ ပြဖို့အတွက်ဆို loading time များစွာပေးရပါလိမ့်မယ်။ website ရဲ့ readability ကို ထိန်းဖို့အတွက်ဆို ဒီလို Limitation clause တွေက များစွာအထောက်ကူပြုပါတယ်။ ဥပမာ များပြားတဲ့ data records တွေကို pagination သုံးပြီး multiple page အဖြစ် ပြောင်းလဲ အသုံးပြုတာဖြစ်ပါတယ်။ အဲ့ဒီအခါ page တိုင်းမှာ သူနဲ့ သက်ဆိုတဲ့ data records

တွေကို loading လုပ်ပြီး နောက် ထပ် data records page အတွက် pagination link လေးတွေ ဖန်တီးထားနိုင်တာဖြစ်ပါတယ်။

LIMIT ရဲ့ အသုံးပြုပုံ syntax က ဒီလိုပါ။

```
SELECT column_name(s) FROM table_name LIMIT row_offset, row_count;
```

LIMIT clause ကို အသုံးပြုတဲ့အခါ parameters ကို တစ်ခုထဲ ထားလို့ရသလို နှစ်ခုလည်း ထားလို့ရပါတယ်။

Parameters နှစ်ခုထားတဲ့အခါမှာ first parameter က first row ရဲ့ offset အဖြစ်လုပ်ဆောင်ပါတယ်။ သဘောအားဖြင့် စတင်မှတ်လို့လည်း ဆိုနိုင်ပါတယ်။ second parameter က ကိုယ် return ပြန်ချင်တဲ့ rows အရေအတွက်ကိုဆိုလိုပါတယ်။

```
+----+------------+-----------+----------------------+------------+
| id | first_name | last_name | email                | age        |
+----+------------+-----------+----------------------+------------+
|  1 | Peter      | Parker    | peterparker@mail.com | 19         |
|  2 | John       | Rambo     | johnrambo@mail.com   | 20         |
|  3 | Clark      | Kent      | clarkkent@mail.com   | 30         |
|  4 | John       | Carter    | johncarter@mail.com  | 25         |
|  5 | Harry      | Potter    | harrypotter@mail.com | 21         |
+----+------------+-----------+----------------------+------------+
```

ဒီ data tables ကို အသုံးပြု ပြီး LIMIT လုပ်ကြည့်ပါမယ်။

```
$select_query = "SELECT * FROM persons LIMIT 2,4";
$result = mysqli_query($conn, $select_query);
if (mysqli_num_rows($result) > 0) {
    while ($row = mysqli_fetch_object($result)) {
        echo "First Name :" . $row->first_name . "<br>";
        echo "Email : " . $row->email . "<br><hr>";
    }
}
```

```
+----+------------+-----------+----------------------+------------+
| id | first_name | last_name | email                | age        |
+----+------------+-----------+----------------------+------------+
|  2 | John       | Rambo     | johnrambo@mail.com   | 20         |
|  3 | Clark      | Kent      | clarkkent@mail.com   | 30         |
|  4 | John       | Carter    | johncarter@mail.com  | 25         |
|  5 | Harry      | Potter    | harrypotter@mail.com | 21         |
+----+------------+-----------+----------------------+------------+
```

ဒီလိုဆိုရင် Data rows တွေက 2 ကနေ စပြီး records 4 ခု ထွက်လာပါလိမ့်မယ်။

PHP MySQL ORDER BY Clause

ORDER BY Clause ကို SELECT statement နဲ့ တွဲပြီးတော့ data တွေကို သတ်မှတ်ချင်တဲ့ field အရ order အတိုင်း ရချင်တဲ့အခါ အသုံးပြုပါတယ်။ ဆိုလိုတာ က ကိုယ်သတ်မှတ်ထားတဲ့ field အရ sorting စီပြီး data တွေကို return ပြန်ပေးတာဖြစ်ပါတယ်။ အဲ့ဒီလို sorting စီတဲ့နေရာမှာ descending or ascending စီလို့ရပါတယ်။

အသုံးပြုပုံ syntax က ဒီလိုပါ။

```
SELECT column_name(s) FROM table_name ORDER BY column_name(s) ASC|DESC
```

```
$select_query = "SELECT * FROM persons ORDER BY age";
$result = mysqli_query($conn, $select_query);
if (mysqli_num_rows($result) > 0) {
    while ($row = mysqli_fetch_object($result)) {
        echo "First Name :" . $row->first_name . "<br>";
        echo "age :" . $row->age . "<br>";
        echo "Email : " . $row->email . "<br><hr>";
    }
}
```

နောက်မှာ ASC or DESC မပါခဲ့ရင် ငယ်စဉ်ကြီးလိုက်စီသွားပါတယ်။

```
$select_query = "SELECT * FROM persons ORDER BY age DESC ";
$result = mysqli_query($conn, $select_query);
if (mysqli_num_rows($result) > 0) {
    while ($row = mysqli_fetch_object($result)) {
        echo "First Name :" . $row->first_name . "<br>";
        echo "age :" . $row->age . "<br>";
        echo "Email : " . $row->email . "<br><hr>";
    }
}
```

ဒီလိုဆိုရင် တော့ data rows တွေက age အရ ကြီးစဉ်ငယ်လိုက် ရပါလိမ့်မယ်။

```
+----+------------+-----------+----------------------+------------+
| id | first_name | last_name | email                | age        |
+----+------------+-----------+----------------------+------------+
|  3 | Clark      | Kent      | clarkkent@mail.com   | 30         |
|  4 | John       | Carter    | johncarter@mail.com  | 25         |
|  5 | Harry      | Potter    | harrypotter@mail.com | 21         |
|  2 | John       | Rambo     | johnrambo@mail.com   | 20         |
+----+------------+-----------+----------------------+------------+
```

PHP MySQL UPDATE Query

UPDATE statement ကို database table ထဲမှာရှိတဲ့ records တွေကို change or modify လုပ်ချင်တဲ့အခါ အသုံးပြုပါတယ်။ UPDATE statement ကိုအသုံးပြုတဲ့အခါ WHERE clause နဲ့တွဲအသုံးပြုရပြီး WHERE condition အရ ပြောင်းလဲမှုက records ပေါ် သက်ရောက်တာဖြစ်ပါတယ်။ UPDATE statement ရဲ့ syntax က ဒီလိုဖြစ်ပါတယ်။

```
UPDATE table_name SET column1 = value, column2 = value2, ... WHERE column_name = some_value
```

```
+----+------------+-----------+----------------------+------------+
| id | first_name | last_name | email                | age        |
+----+------------+-----------+----------------------+------------+
|  1 | Peter      | Parker    | peterparker@mail.com | 19         |
|  2 | John       | Rambo     | johnrambo@mail.com   | 20         |
|  3 | Clark      | Kent      | clarkkent@mail.com   | 30         |
|  4 | John       | Carter    | johncarter@mail.com  | 25         |
|  5 | Harry      | Potter    | harrypotter@mail.com | 21         |
+----+------------+-----------+----------------------+------------+
```

ဒီ table ကို ပဲ အသုံးပြုပြီးတော့ id 1 ရှိတဲ့ person ရဲ့ data တွေကို ပြောင်းလည်းကြည့်ပါမယ်။ ပြန်ပြီး modify လုပ်ကြည့်ပါမယ်။

```
$sql = "UPDATE persons SET email='peterparker_new@mail.com' WHERE id=1";
if(mysqli_query($link, $sql)){
    echo "Records were updated successfully.";
} else {
    echo "ERROR: Could not able to execute $sql. " . mysqli_error($link);
}
```

ဒါက ရိုရိုရှင်းရှင်း sql query ကို အသုံးပြုပြီးတော့ id 1 ရှိတဲ့ person ရဲ့ email ကို “peterparker_new@mail.com” ဆိုပြီး ပြောင်းလဲလိုက်တာဖြစ်ပါတယ်။ prepare statement နဲ့ဆိုရင် ဒီလိုပါ။

```
$sql = "UPDATE persons SET email=? WHERE id=?";


if ($stmt = mysqli_prepare($conn, $sql)) {
    mysqli_stmt_bind_param($stmt, "si", $email, $id);
    $email = "harrypotter_new@mail.com";
```

```
    $id = 5;
    mysqli_stmt_execute($stmt);
    if (mysqli_stmt_affected_rows($stmt) > 0) {
        echo "Update Successfully!";
    } else {
        echo "Errors : " . mysqli_error($conn);
    }
} else {
    echo "Errors : " . mysqli_error($conn);
}
mysql_stmt_close($stmt);
```

ဒီလိုဆိုလည်း update ဖြစ်ပါတယ်။ statement binding လုပ်တဲ့နေရာမှာ id နေရာမှာ integer ဖြစ်တဲ့အတွက် “i” လို့သုံးထားတာ သတိပြုပါ။ နောက်ပြီး value တိုက်ရိုက်ထည့်ချင်းကို ရှောင်ပါ။ ဒီမှာ Update တကယ်ဖြစ်လား စစ်ချင်တဲ့အတွက် mysqli_stmt_affected_rows() function ကို အသုံးပြုထားပါတယ်။ mysqli_stmt_affected_rows() function ကို insert, update, delete စတဲ့ operations or statement တွေ လုပ်ဆောင်တဲ့အခါ data ပြောင်းလဲမှု ဖြစ်ပါတယ်။ အဲ့ဒီ ပြောင်းလည်းသွားတဲ့ data records အရည်အတွက် ကို သိချင်တဲ့အခါ အသုံးပြုပါတယ်။ ပြောင်းလဲတဲ့ rows အရေအတွက် number ကို return ပြန်ပေးတာဖြစ်တဲ့အတွက် 0 ထက်ကြီးပါက ပြောင်းလဲသွားတယ် update ဖြစ်သွားတယ်လို့ ယူဆလို့ရပါတယ်။

PHP MySQL DELETE Query

Database table ထဲကို data တွေ insert လုပ်လို့ရသလို table ထဲက data records တွေကိုလည်း DELETE statement ကိုအသုံးပြုပြီးတော့ ပြန်ဖျက်လို့ရပါတယ်။ DELETE statement ကို

WHERE clause နဲ့တွဲအသုံးပြုပြီးတော့ WHERE condition အရ data တွေကို delete လုပ်ပါတယ်။ သူ့ရဲ့ syntax ကဒီလိုပါ။

```
DELETE FROM table_name WHERE column_name=some_value
```

```
+----+------------+-----------+----------------------+------------+
| id | first_name | last_name | email                | age        |
+----+------------+-----------+----------------------+------------+
|  1 | Peter      | Parker    | peterparker@mail.com | 19         |
|  2 | John       | Rambo     | johnrambo@mail.com   | 20         |
|  3 | Clark      | Kent      | clarkkent@mail.com   | 30         |
|  4 | John       | Carter    | johncarter@mail.com  | 25         |
|  5 | Harry      | Potter    | harrypotter@mail.com | 21         |
+----+------------+-----------+----------------------+------------+
```

ဒီ table ကို ပဲ အသုံးပြုပြီးတော့ id 1 ရှိတဲ့ person ရဲ့ data တွေကို delete လုပ်ကြည့်ပါမယ်။

```
$sql = "DELETE FROM persons WHERE first_name='John'";
if(mysqli_query($link, $sql)){
    echo "Records were deleted successfully.";
} else{
    echo "ERROR: Could not able to execute $sql. " . mysqli_error($link);
}
```

ဒါက ရိုးရိုး DELETE query ကိုအသုံးပြုပြီးတော့ first_name “John” ဖြစ်တဲ့ data record တစ်ခုလုံးကို delete လုပ်တာဖြစ်ပါတယ် အကယ်၍ first_name “John” က နှစ်ယောက်ရှိတယ်ဆိုရင် data records နှစ်ခုလုံး delete ဖြစ်သွားမှာပါ။
Prepared statement နဲ့ ဒီလိုအသုံးပြုပါတယ်။

```
$sql = "DELETE FROM persons  WHERE first_name=?";

if ($stmt = mysqli_prepare($conn, $sql)) {
```

```
    mysqli_stmt_bind_param($stmt, "s", $first_name);
    $first_name = "Harry";
    mysqli_stmt_execute($stmt);

    if (mysqli_stmt_affected_rows($stmt) > 0) {
        echo "Delete Successfully!";
    } else {
        echo "Errors : " . mysqli_error($conn);
    }
} else {
    echo "Errors : " . mysqli_error($conn);
}
```

ဒီလိုဆိုရင် first_name “Harry” ရှိတဲ့ data records ဟာ database table ထဲကနေ delete ဖြစ်သွားပါတယ်။ အထက်မှာ ဖော်ပြခဲ့သလိုပဲ delete ဖြစ်လား သိဖို့အတွက် mysqli_stmt_affect_rows() function ကို အသုံးပြုထားပါတယ်။ ဒီမှာတော့ delete ဖြစ်သွားတဲ့ rows အရေအတွက်ကို return ပြန်ပေးမှာဖြစ်ပါတယ်။