

PHP

เนื้อหา

- PHP คืออะไร
- รูปแบบไวยากรณ์ (Syntax) ของภาษา PHP
- ตัวแปรและการประกาศตัวแปร
- การรับข้อมูลจากฟอร์ม
- ตัวดำเนินการ (Operators)
- คำสั่งเลือกเงื่อนไข
- คำสั่งวนซ้ำ
- อาร์เรย์ (Array)
- ฟังก์ชัน (Function)
- Global variables - Superglobals
- วันที่และเวลา
- การ Include File
- การใช้งาน Session
- PHP กับ JSON
- การใช้ฟังก์ชัน file_get_contents()
- การเขียนโปรแกรมเชิงวัตถุ
- การทำงานกับไฟล์
- การทำงานกับ Database

PHP คืออะไร

PHP คือภาษาคอมพิวเตอร์จำพวก scripting language สำหรับทำงานด้านฝั่งของเซิร์ฟเวอร์ (server-side scripting) ถูกออกแบบมาสำหรับการพัฒนาเว็บไซต์ แต่มันก็ยังสามารถใช้เขียนโปรแกรมเพื่อวัตถุประสงค์ทั่วไปได้ คำว่า PHP ย่อมาจาก Personal Home Page ซึ่งในปัจจุบันนี้หมายถึง PHP: Hypertext Preprocessor

PHP เป็นผลงานที่เติบโตมาจากกลุ่มของนักพัฒนาในเชิงเปิดเผยแพร่ต้นฉบับ หรือ OpenSource ดังนั้น PHP จึงมีการพัฒนาไปอย่างรวดเร็ว และแพร่หลายโดยเฉพาะอย่างยิ่งเมื่อใช้ร่วมกับ Apache Web server ระบบปฏิบัติการอย่างเช่น UNIX, Linux, CentOS รวมทั้ง Microsoft Windows



ลักษณะเด่นของ PHP

- ใช้งานได้ฟรี
- PHP เป็นโปรแกรมทำงานฝั่ง Sever มีขีดความสามารถไม่จำกัด
- PHP รองรับระบบปฏิบัติการหลากหลาย ไม่ว่าจะเป็น UNIX, Linux, Windows
- เรียนรู้ง่าย ใช้โครงสร้างและไวยากรณ์ภาษาที่ง่าย และสามารถแทรกเข้าไปใน HTML ได้
- มีความเร็วและมีประสิทธิภาพ
- ใช้ร่วมกับ XML ได้ทันที
- ใช้กับระบบแฟ้มข้อมูลได้
- ใช้กับข้อมูลตัวอักษรได้อย่างมีประสิทธิภาพ
- ใช้กับโครงสร้างข้อมูล แบบ Scalar, Array, Associative array
- ใช้กับการประมวลผลภาพได้

รูปแบบไวยากรณ์ (Syntax) ของภาษา PHP

การเขียนภาษา PHP มีรูปแบบไวยากรณ์ที่สำคัญดังนี้

- โค้ดของ PHP จะอยู่ในบล็อก `<?php ... ?>` หรือ `<? ... ?>`
- แต่ละคำสั่งจะต้องใส่ semicolon (;) ปิดท้าย เช่น `$a = $b * 2;`
- แต่ละบรรทัดสามารถเขียนได้มากกว่า 1 คำสั่ง เช่น `$a = 1; $b = 2; $x = 0;` แต่ไม่นิยมทำกัน เพราะจะทำให้ดูโค้ดยาก
- การเขียนคอมเมนต์ เพื่ออธิบายโปรแกรม หรือเพื่อปิดโค้ดในส่วนที่ไม่ต้องการให้ทำงาน จะใช้เครื่องหมาย # หรือ // นำหน้าข้อความคอมเมนต์ กรณีที่ต้องการคอมเมนต์หลายๆ บรรทัดจะใช้ `/*` ข้อความคอมเมนต์ `*/` เช่น

PHP Code

```
<?php
# ทักทายกันก่อน
echo "Hello PHP developer.";

// ประกาศตัวแปร
$a = 5;
$b = 2;
$c = $a * $b;    //หาผลคูณของ a กับ b
$d = $a + /* $b */ + $c;

/*
echo "<br>a = $a";
echo "<br>b = $b";
echo "<br>c = $c";
echo "<br>d = $d";
*/

echo "<br>a = $a<br>b = $b<br>c = $c<br>d = $d";
?>
```

- ไฟล์ PHP สามารถใส่ HTML แท็ก และโค้ด PHP รวมกันได้ ดังตัวอย่างนี้

PHP Code

```
<!DOCTYPE html>
<html>
<body>
    <h3>HTML</h3>

    <?php
    echo "PHP";
    ?>
</body>
</html>
```

- ชื่อตัวแปร ตัวอักษรเล็ก/ใหญ่ มีความสำคัญ (Case Sensitive) ถือว่าไม่ใช่ตัวเดียวกัน เช่น \$num = 5; \$Num = 10; ตัวแปร \$num กับ \$Num ไม่ใช่ตัวเดียวกัน
- คีย์เวิร์ด คำสั่ง ชื่อคลาส ชื่อฟังก์ชัน ตัวอักษรเล็ก/ใหญ่ ไม่สำคัญ (Not Case Sensitive)
- While space หรือช่องว่าง ไม่มีผลกับการทำงานโปรแกรม ดังตัวอย่างต่อไปนี้ ตัวแปรทั้ง 3 ตัวมีค่าเท่ากัน

PHP Code

```
<?php
$str1="PHP";
$str2 = "PHP";
$str3  = "PHP";
?>
```

ตัวแปรและการประกาศตัวแปร

ตัวแปร คือ สิ่งที่ใช้เก็บค่าของข้อมูลในหน่วยความจำ ตัวแปรจะประกอบไปด้วยชื่อของตัวแปร (identifier) ใช้เพื่ออ้างอิงหรือเข้าถึงค่าภายในตัวแปร การตั้งชื่อตัวแปรจะมีกฎในการตั้งชื่อดังนี้

- ต้องขึ้นต้นด้วย \$ แล้วตามด้วยชื่อตัวแปร เช่น \$name
- ชื่อตัวแปรต้องขึ้นต้นด้วยตัวอักษรหรือ underscore (_)
- ไม่สามารถตั้งชื่อตัวแปรที่ขึ้นต้นด้วยตัวเลขได้
- ชื่อตัวแปรต้องเป็นตัวอักษร ตัวเลข หรือ underscore (_) เท่านั้น
- ชื่อตัวแปรตัวเล็ก/ใหญ่ไม่เหมือนกัน เช่น \$total กับ \$Total ไม่ถือว่าเป็นตัวแปรเดียวกัน

การประกาศตัวแปร

ภาษา PHP สามารถประกาศตัวแปรโดยไม่ต้องกำหนดประเภทของตัวแปรได้เลย เช่น

PHP Code
<pre><?php \$name = "PHP"; \$PI = 3.14159265359; \$sum = 0; \$isOK = true; \$value = null; \$arr = array(); ?></pre>

การตรวจสอบตัวแปร และการยกเลิกตัวแปร

เราสามารถเช็คได้ว่า ตัวแปรที่จะใช้งานมีการประกาศมาแล้วหรือยัง โดยใช้ฟังก์ชัน `isset()` และสามารถยกเลิกการประกาศตัวแปรได้ ด้วยฟังก์ชัน `unset()` ดังตัวอย่างต่อไปนี้

PHP Code
<pre><?php \$str = "PHP"; if (isset(\$str)) { echo "Value of \\$str is \$str
"; unset(\$str); echo "Value of \\$str is \$str
"; } else { echo "Variables \\$str does not exists."; } ?></pre>
Output
<p>Value of \$str is PHP</p> <p>Warning: Undefined variable \$str in C:\webroot\variable.php on line 20</p> <p>Value of \$str is</p>

ประเภทของข้อมูล

ภาษา PHP จะมีข้อมูลแบ่งออกเป็น 8 ประเภท ดังนี้

1. **Integer:** คือ ประเภทข้อมูลแบบจำนวนเต็ม (จำนวนที่ไม่มีทศนิยม) สามารถเป็นได้จำนวนเต็มลบหรือจำนวนเต็มบวก มีค่าอยู่ในช่วง -2,147,483,648 and 2,147,483,647 ตัวอย่างการประกาศตัวแปรประเภทนี้ เช่น `$num = 65; $amount = 1250;` เป็นต้น
2. **Float:** คือ ประเภทข้อมูลที่เก็บข้อมูลในรูปแบบของจำนวนจริง ซึ่งมักจะใช้ในการเก็บตัวเลขที่มีค่าและความละเอียดมาก เช่น ข้อมูลการคำนวณทางวิทยาศาสตร์ หรือตัวเลขที่มีจุดทศนิยม ตัวอย่างการประกาศตัวแปรประเภทนี้ เช่น `$PI = 3.14159265359; $price = 29.5;`
3. **String:** คือ ประเภทข้อมูลที่เป็นข้อความ หรือตัวอักษร การประกาศตัวแปรประเภทนี้ เช่น `$gender = "Male"; $company = "WinWin Interactive";`
4. **Boolean:** คือ ประเภทข้อมูลทางตรรกศาสตร์ มีค่าที่เป็นไปได้เพียงสองค่าคือ จริง (true) หรือ เท็จ (false) ตัวอย่างการประกาศตัวแปร เช่น `$tested = false; $isOK = true;`
5. **Array:** คือ ประเภทข้อมูลแบบชุดซึ่งมีการเก็บของข้อมูลเป็นลำดับโดยมี Index ในการอ้างถึงค่าของสมาชิกในอาร์เรย์ เพื่อให้สะดวกในการจัดการกับข้อมูลในอาร์เรย์ ยกตัวอย่างเช่น เก็บคะแนนของนักเรียนทั้งหมดในห้องเรียนไว้ในอาร์เรย์ เพื่อให้ง่ายในการหาผลรวม ค่าเฉลี่ย ค่าต่ำสุด/สูงสุด เป็นต้น ตัวอย่างการประกาศตัวแปรประเภทนี้ เช่น `$score = array(67, 81, 59, 77, 54); $member = array("อนุชา","สุกัญญา","ภาณุวัฒน์","ฉัตรชัย","ไกรศร");` ข้อมูลในอาร์เรย์จะเป็นข้อมูลประเภทอะไรก็ได้ หรือเป็นอาร์เรย์เองก็ได้
6. **Object:** เป็นอินสแตนซ์ของคลาสที่กำหนดโดยโปรแกรมเมอร์ ซึ่งสามารถบรรจุทั้งค่าและฟังก์ชันประเภทอื่นๆ ที่เฉพาะเจาะจงสำหรับคลาส
7. **Resource:** คือ ประเภทข้อมูลพิเศษที่เก็บข้อมูลจากภายนอก โดยข้อมูลของตัวแปรประเภทนี้มักจะสร้างจากฟังก์ชันพิเศษ เช่น การอ่านข้อมูลจากฐานข้อมูล หรือการอ่านข้อมูลของรูปภาพ ข้อมูลเหล่านี้จะถูกเก็บไว้ในตัวแปร Resource ในรูปแบบของ Binary
8. **Null:** เป็นข้อมูลชนิดพิเศษในการบ่งบอกถึงว่าตัวแปรไม่ได้ถูกกำหนดค่า หรือไม่มีค่าอะไร ตัวอย่างการประกาศตัวแปรประเภทนี้ เช่น `$temp = null;`

ขอบเขตของตัวแปร (Variable Scope)

ตัวแปรที่สร้างขึ้นปกติจะเป็นตัวแปรแบบ local คือ สร้างขึ้นในระดับไหน ก็จะใช้งานได้เฉพาะในระดับนั้น เช่น สร้างขึ้นในฟังก์ชัน ก็ใช้งานได้แค่ในฟังก์ชัน นอกฟังก์ชันจะไม่รู้จัก ตรงกันข้าม ถ้าสร้างขึ้นนอกฟังก์ชัน ก็จะใช้งานในฟังก์ชันไม่ได้ ดังตัวอย่างต่อไปนี้

PHP Code

```
1  <!DOCTYPE html>
2  <html>
3  <body>
4
5  <?php
6  $rate = 1.5;
7  function calbonus($salary) {
8      $bonus = $salary * $rate;
9      echo "1. \$rate = " . $rate . ", \$bonus = " . $bonus;
10 }
11
12 calbonus(20000);
13 echo "<br><br>2. \$rate = " . $rate . ", \$bonus = " . $bonus;
14 echo "<br><br>3. Type of \$bonus: " . gettype($bonus);
15 ?>
16
17 </body>
18 </html>
```

Output

Warning: Undefined variable \$rate in D:\Game\web\variable_scope.php on line 8

Warning: Undefined variable \$rate in D:\Game\web\variable_scope.php on line 9

1. \$rate = , \$bonus = 0

Warning: Undefined variable \$bonus in D:\Game\web\variable_scope.php on line 13

2. \$rate = 1.5, \$bonus =

Warning: Undefined variable \$bonus in D:\Game\web\variable_scope.php on line 14

3. Type of \$bonus: NULL

ค่าคงที่

ค่าคงที่คือ ค่าของ Literal หรือค่า ใดๆ ที่มีการกำหนดให้กับตัวแปรค่าคงที่ เมื่อกำหนดค่าแล้ว จะไม่สามารถเปลี่ยนแปลงค่าได้อีก ในภาษา PHP เราสามารถประกาศค่าคงที่ได้สองแบบ โดยการใช้ฟังก์ชัน `define()` และคำสั่ง `Const` ดังตัวอย่างต่อไปนี้

PHP Code
<pre><?php define("NAME", "บริษัท วินวิน อินเตอร์แอคทีฟ จำกัด"); define("WEBSITE", "winwin.co.th"); const ADDRESS = "เลขที่ 446/67-71 อาคารปาร์คอเวนิว 2 ถนนสุขุมวิท 71 แขวงพระโขนงเหนือ เขตวัฒนา กรุงเทพฯ 10110"; const FAX = "02-381-5234"; //ตัวอย่างการใช้งาน echo NAME . "
ที่อยู่: " . ADDRESS . "
Website: " . WEBSITE . "
Fax: " . FAX; ?></pre>
Output
<pre>บริษัท วินวิน อินเตอร์แอคทีฟ จำกัด ที่อยู่: เลขที่ 446/67-71 อาคารปาร์คอเวนิว 2 ถนนสุขุมวิท 71 แขวงพระโขนงเหนือ เขตวัฒนา กรุงเทพฯ 10110 Website: winwin.co.th Fax: 02-381-5234</pre>

ข้อแตกต่างของฟังก์ชัน `define()` กับคำสั่ง `const`

- `Const` จะทำการกำหนดค่าคงที่ในเวลา Run-time ในขณะที่ `define()` กำหนดค่าคงที่ใน Compile
- `Const` ไม่สามารถใช้ในบล็อกของคำสั่ง เช่น `If For` หรือ `While` แต่ `define()` ใช้ได้
- `Const` จะเป็นแบบ case-sensitive ในขณะที่ `define()` สามารถกำหนดเป็น case-insensitive ได้
- `Const` สามารถกำหนดค่าแบบอาร์เรย์ได้ ในขณะที่ `define()` ไม่สามารถทำได้
- `Const` สามารถใช้ประกาศเป็นค่าคงที่ภายในคลาสหรือ Interface ได้

ค่าคงที่ในภาษา PHP

ในภาษา PHP มีค่าคงที่ที่เรียกว่า Predefined constant ซึ่งเป็นค่าคงที่ของภาษาที่เกิดขึ้นในขณะที่โปรแกรมทำงาน นั้นหมายความว่าค่าคงที่เหล่านี้จะถูกกำหนดในตอนแรกที่ Interpreter ทำงานเสร็จสิ้น และแต่ละตัวจะให้ผลลัพธ์ที่แตกต่างกันไปในการรันโปรแกรม ต่อไปนี้เป็นรายการของ Predefined constant ในภาษา PHP

Constant	Description
__CLASS__	ชื่อของคลาสปัจจุบันที่โปรแกรมทำงานอยู่
__DIR__	ชื่อของโฟลเดอร์ปัจจุบันที่ไฟล์โปรแกรมทำงานอยู่
__FILE__	ชื่อของไฟล์โปรแกรมทำงานอยู่
__FUNCTION__	ชื่อของฟังก์ชันที่โปรแกรมทำงานอยู่
__LINE__	หมายเลขบรรทัดที่โปรแกรมทำงานอยู่
__METHOD__	ชื่อของเมธอดที่โปรแกรมทำงานอยู่
__NAMESPACE__	ชื่อของเนมสเปซที่โปรแกรมทำงานอยู่

ตัวอย่างการใช้งาน

PHP Code
<pre><?php echo "DIR: " . __DIR__ . "
"; echo "FILE: " . __FILE__ . "
"; echo "LINE: " . __LINE__ . "
"; ?></pre>
Output
<pre>DIR: C:\webroot FILE: C:\webroot\const.php LINE: 4</pre>

โอเปอเรเตอร์ (Operators)

โอเปอเรเตอร์ คือ ตัวดำเนินการระหว่างค่า 2 ค่า หรือตัวแปร 2 ตัวแปร โอเปอเรเตอร์ในภาษา PHP แบ่งเป็นกลุ่มย่อยได้ดังนี้

โอเปอเรเตอร์ทางคณิตศาสตร์ (Arithmetic Operators)

โอเปอเรเตอร์	ชื่อ	ตัวอย่าง	ความหมาย
+	Addition	$\$a + \b	การบวกค่า \$a กับ \$b
-	Subtraction	$\$a - \b	การลบค่า \$b ออกจากค่า \$a
*	Multiplication	$\$a * \b	การคูณค่า \$a กับ \$b
/	Division	$\$a / \b	การหารค่า \$a ด้วย \$b
%	Modulus	$\$a \% \b	การหาค่าเศษที่เหลือจากการหารค่า \$a ด้วย \$b เช่น $\$a = 8; \$b = 5;$ $\$a \% \b จะได้ 3
**	Exponentiation	$\$a ** \b	การหาค่ายกกำลังของ \$a ยกกำลัง \$b

ตัวอย่างการใช้งาน

PHP Code
<pre><?php \$a = 9; \$b = 5; echo '\$a + \$b = ' . (\$a + \$b) . '
'; echo '\$a - \$b = ' . (\$a - \$b) . '
'; echo '\$a * \$b = ' . (\$a * \$b) . '
'; echo '\$a / \$b = ' . (\$a / \$b) . '
'; echo '\$a % \$b = ' . (\$a % \$b) . '
'; echo '\$a ** \$b = ' . (\$a ** \$b) . '
'; ?></pre>
Output
<pre>\$a + \$b = 14 \$a - \$b = 4 \$a * \$b = 45 \$a / \$b = 1.8 \$a % \$b = 4 \$a ** \$b = 59049</pre>

โอเปอเรเตอร์กำหนดค่า (Assignment Operators)

โอเปอเรเตอร์	ตัวอย่าง	ความหมาย
=	\$a = 5	กำหนดค่าให้ \$a มีค่าเป็น 5
+=	\$a += 5	กำหนดให้ตัวแปร \$a มีค่าเท่ากับ \$a บวก 5
-=	\$a -= 5	กำหนดให้ตัวแปร \$a มีค่าเท่ากับ \$a ลบ 5
*=	\$a *= 5	กำหนดให้ตัวแปร \$a มีค่าเท่ากับ \$a คูณ 5
/=	\$a /= 5	กำหนดให้ตัวแปร \$a มีค่าเท่ากับ \$aหาร 5
%=	\$a %= 5	กำหนดให้ตัวแปร \$a มีค่าเท่ากับเศษของ \$a หาร 5

ตัวอย่างการใช้งาน

PHP Code
<pre><?php \$a = 9; \$a += 2; echo '\$a = ' . \$a . '
'; \$b = 15; \$b -= 6; echo '\$b = ' . \$b . '
'; \$c = 8; \$c *= 3; echo '\$c = ' . \$c . '
'; \$d = 18; \$d /= 4; echo '\$d = ' . \$d . '
'; \$e = 11; \$e %= 5; echo '\$e = ' . \$e . '
'; ?></pre>
Output
<pre>\$a = 11 \$b = 9 \$c = 24 \$d = 4.5 \$e = 1</pre>

โอเปอเรเตอร์เกี่ยวกับข้อความ (String Operators)

โอเปอเรเตอร์	ชื่อ	ตัวอย่าง	ความหมาย
.	Concatenation	\$a . \$b	การนำข้อความในตัวแปร \$b มาต่อท้ายตัวแปร \$a
.=	Concatenation assignment	\$a .= \$b	การกำหนดค่าให้ตัวแปร \$a โดยเอาข้อความในตัวแปร \$b มาต่อท้ายตัวแปร \$a

ตัวอย่างการใช้งาน

PHP Code
<pre><?php \$a = "Big"; \$b = "Boss"; \$c = \$a . \$b; echo \$c . "
"; \$a .= \$b; //เหมือนกับ \$a = \$a . \$b; echo \$a; ?></pre>
Output
<pre>BigBoss BigBoss</pre>

โอเปอเรเตอร์การเปรียบเทียบ (Comparison Operators)

โอเปอเรเตอร์	ชื่อ	ตัวอย่าง	ความหมาย
==	Equal	\$a == \$b	เป็นจริง เมื่อ \$a เท่ากับ \$b
===	Identical	\$a === \$b	เป็นจริง เมื่อ \$a เท่ากับ \$b และต้องเป็นข้อมูลประเภทเดียวกัน
!=	Not equal	\$a != \$b	เป็นจริง เมื่อ \$a ไม่เท่ากับ \$b
<>	Not equal	\$a <> \$b	เป็นจริง เมื่อ \$a ไม่เท่ากับ \$b
!==	Not identical	\$a !== \$b	เป็นจริง เมื่อ \$a ไม่เท่ากับ \$b หรือมีประเภทข้อมูลไม่เหมือนกัน
<	Less than	\$a < \$b	เป็นจริง เมื่อ \$a น้อยกว่า \$b
>	Greater than	\$a > \$b	เป็นจริง เมื่อ \$a มากกว่า \$b

<=	Less than or equal to	\$a <= \$b	เป็นจริง เมื่อ \$a น้อยกว่าหรือเท่ากับ \$b
>=	Greater than or equal to	\$a >= \$b	เป็นจริง เมื่อ \$a มากกว่าหรือเท่ากับ \$b
<=>	Spaceship	\$a <=> \$b	เปรียบเทียบตัวแปร \$a และ \$b โดยจะได้ -1 ถ้า \$a น้อยกว่า \$b 0 ถ้า \$a เท่ากับ \$b 1 ถ้า \$a มากกว่า \$b

ตัวอย่างการใช้งาน

PHP Code
<pre><?php if (50 == "50") echo "แมว
"; if (50 === "50") echo "ผีเสื้อ
"; if ("ant" != "bee") echo "ม้า
"; if ("ant" <> "ANT") echo "กระต่าย
"; if (50 !== 50.0) echo "แพะ
"; if (100 < 120) echo "หมู
"; if (100 > 120) echo "ไก่
"; if (75 <= 50) echo "ควาย
"; if (75 >= 50) echo "ช้าง
"; echo (15 <=> 20); ?></pre>
Output
<pre>แมว ม้า กระต่าย แพะ หมู ช้าง -1</pre>

โอเปอเรเตอร์เพิ่มค่า/ลดค่า (Increment / Decrement Operators)

โอเปอเรเตอร์	ชื่อ	ความหมาย
++\$x	Pre-increment	เพิ่มค่า \$x ขึ้น 1 แล้วค่อย return ค่า
\$x++	Post-increment	Return ค่า \$x ก่อน แล้วค่อยเพิ่มค่า \$x ขึ้น 1
--\$x	Pre-decrement	ลดค่า \$x ลง 1 แล้วค่อย return ค่า
\$x--	Post-decrement	Return ค่า \$x ก่อน แล้วค่อยลดค่า \$x ลง 1

ตัวอย่างการใช้งาน

PHP Code
<pre><?php \$a = 5; \$b = 5; \$c = 5; \$d = 5; echo ++\$a.", ".\$b++.", ".\$c--.", ".\$d--."
"; echo \$a.", ".\$b.", ".\$c.", ".\$d."
"; ?></pre>
Output
<pre>6, 5, 4, 5 6, 6, 4, 4</pre>

โอเปอเรเตอร์ตรรกศาสตร์ (Logical operators)

โอเปอเรเตอร์	ชื่อ	ตัวอย่าง	ความหมาย
and หรือ &&	And	\$a and \$b	เป็นจริง ถ้า \$a และ \$b มีค่าเป็นจริง
or หรือ	Or	\$a or \$b	เป็นจริง ถ้า \$a หรือ \$b มีค่าเป็นจริง
xor	Xor	\$a xor \$b	เป็นจริง ถ้า \$a หรือ \$b มีค่าเป็นจริง แต่ต้องไม่เป็นจริง ทั้ง 2
!	Not	!\$a	เป็นจริง ถ้า \$a มีค่าเป็นเท็จ และเป็นเท็จ ถ้า \$a มีค่า เป็นจริง

ตัวอย่างการใช้งาน

PHP Code

```
<?php
$a = 100;
$b = 150;
$c = false;
if ($a == 100 and $b == 100) {
    echo "มะม่วง<br>";
} else {
    echo "ขนุน<br>";
};

if ($a == 100 or $b == 100) {
    echo "แดงโม<br>";
} else {
    echo "ส้มโ&#x0e4;<br>";
};

if ($a != $c xor $b != $c) {
    echo "มะพร้าว<br>";
} else {
    echo "ลำไย<br>";
}

var_dump(!$c);
?>
```

Output

ขนุน
แดงโม
ลำไย
bool(true)

โอเปอเรเตอร์เกี่ยวกับอาร์เรย์ (Array Operators)

โอเปอเรเตอร์	ชื่อ	ตัวอย่าง	ความหมาย
+	Union	$\$a + \b	เอาอาร์เรย์ $\$a$ กับ $\$b$ มารวมกัน

ตัวอย่างการใช้งาน

PHP Code
<pre><?php \$a = array("a" => "TOYOTA", "b" => "HONDA"); \$b = array("b" => "MAZDA", "c" => "NISSAN", "d" => "FORD"); \$c = \$a + \$b; print_r(\$a); echo "<hr>"; print_r(\$b); echo "<hr>"; print_r(\$c); ?></pre>
Output
<pre>Array ([a] => TOYOTA [b] => HONDA) Array ([b] => MAZDA [c] => NISSAN [d] => FORD) Array ([a] => TOYOTA [b] => HONDA [c] => NISSAN [d] => FORD)</pre>

จากตัวอย่างนี้จะเห็นว่าอาร์เรย์ $\$a$ และ $\$b$ มีคีย์ที่ซ้ำกัน เมื่อเอาอาร์เรย์ทั้ง 2 มารวมกัน คีย์ที่ซ้ำกัน จะไม่ถูกแทนที่

โอเปอเรเตอร์กำหนดค่าตามเงื่อนไข (Conditional Assignment Operators)

โอเปอเรเตอร์	ชื่อ	ตัวอย่าง	ความหมาย
?:	Ternary	<code>\$x = expr1 ? expr2 : expr3</code>	\$x จะมีค่าเท่ากับ expr2 ถ้า expr1 เป็นจริง แต่ถ้า expr1 ไม่เป็นจริง \$x จะมีค่าเท่ากับ expr3
??	Null coalescing	<code>\$x = expr1 ?? expr2</code>	\$x จะมีค่าเท่ากับ expr1 ถ้า expr1 มีค่า แต่ถ้า expr1 ไม่มีค่า \$x จะได้ค่าเท่ากับ expr2

ตัวอย่างการใช้งาน

PHP Code
<pre><?php \$total = 1525; \$discount = \$total >= 1000 ? \$total * 0.05 : 0; echo "Total: \$total, Discount: \$discount
"; \$user = \$_GET["user"] ?? "Anonymous"; \$class = \$class ?? "General"; echo "User: \$user, Class: \$class"; ?></pre>
Output
<pre>Total: 1525, Discount: 76.25 User: Anonymous, Class: General</pre>

จากตัวอย่างนี้ กำหนดให้ \$total มีค่า 1525 และหาส่วนลด โดยกำหนดเงื่อนไขว่า ถ้า \$total มากกว่าหรือเท่ากับ 1000 จะคำนวณส่วนลดให้ 5% แต่ถ้า \$total ไม่ถึง 1000 จะไม่ได้ส่วนลด

ส่วน \$user จะอ่านจาก parameter “user” แต่ถ้าไม่มี parameter “user” ตัวแปร \$user จะได้ค่าเป็น “Anonymous” และ \$class จะได้ค่าเป็น “General” เพราะตัวแปร \$class ไม่ได้ประกาศและกำหนดค่ามาก่อน

คำสั่งเลือกเงื่อนไข

คำสั่งเลือกเงื่อนไข ใช้เพื่อกำหนดให้โปรแกรมเลือกการทำงานตามเงื่อนไขใดอย่างหนึ่ง ภาษา PHP มีคำสั่งเลือกเงื่อนไข 2 ตัว คือ if และ switch ซึ่งจะมีรูปแบบดังนี้

คำสั่ง if

เป็นคำสั่งตรวจสอบเงื่อนไข เพื่อควบคุมการทำงานของโปรแกรมให้ทำงานตามเงื่อนไขที่กำหนด แต่ถ้าไม่เป็นไปตามเงื่อนไขก็จะไม่ทำงาน การเขียนคำสั่ง if มีรูปแบบดังนี้

```
if (condition) {  
    Code to be executed if condition is true;  
}
```

ตัวอย่างการใช้งาน

PHP Code
<pre><?php \$status = 0; if (\$status != 1) { echo "ขณะนี้ระบบ ปิดให้บริการชั่วคราว"; } ?></pre>
Output
ขณะนี้ระบบ ปิดให้บริการชั่วคราว

คำสั่ง if...else

เป็นคำสั่งตรวจสอบเงื่อนไข โดยจะทำตามคำสั่งในบล็อก if ถ้าเงื่อนไขเป็นจริง หรือทำตามคำสั่งในบล็อก else ถ้าเงื่อนไขเป็นเท็จ มีรูปแบบคำสั่งดังนี้

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

ตัวอย่างการใช้งาน

PHP Code
<pre><?php \$n = date("N"); if (\$n < 6) { echo "Today is working day"; } else { echo "Today is Weekend day"; } ?></pre>
Output
Today is working day

คำสั่ง if...elseif...else

เป็นคำสั่งตรวจสอบเงื่อนไข โดยสามารถตรวจสอบได้มากกว่า 1 เงื่อนไข โดยมีรูปแบบคำสั่งดังนี้

```
if (condition) {  
    Code to be executed if this condition is true;  
} elseif (condition) {  
    code to be executed if first condition is false and this condition is true;  
} else {  
    code to be executed if all conditions are false;  
}
```

ตัวอย่างการใช้งาน

PHP Code
<pre><?php \$score = rand(1,100); if (\$score < 50) { \$grade = "F"; } elseif (\$score < 60) { \$grade = "D"; } elseif (\$score < 70) { \$grade = "C"; } elseif (\$score < 80) { \$grade = "B"; } else { \$grade = "A"; } echo "คะแนนสุ่มได้ \$score คะแนน ได้เกรด \$grade"; ?></pre>
Output
คะแนนสุ่มได้ 87 คะแนน ได้เกรด A

* คะแนนและเกรดที่ได้ ขึ้นอยู่กับ \$score ที่สุ่มได้

คำสั่ง switch

คำสั่ง switch เป็นคำสั่งในการเลือกเงื่อนไขคล้ายกับคำสั่ง if...elseif แต่คำสั่ง switch จะใช้สำหรับเปรียบเทียบกับค่าคงที่โดยตรงที่ไม่ใช่ Expression โดยมีรูปแบบคำสั่งดังนี้

```
switch (n) {  
    case label1:  
        code to be executed if n=label1;  
        break;  
    case label2:  
        code to be executed if n=label2;  
        break;  
    case label3:  
        code to be executed if n=label3;  
        break;  
    ...  
    default:  
        code to be executed if n is different from all labels;  
}
```

ตัวอย่างการใช้งาน

PHP Code
<pre><?php \$code = "JPY"; echo "Code: \$code
"; switch (\$code) { case "THB": echo "Currency: Thai Baht"; break; case "USD": echo "Currency: U.S. dollar"; break; case "EUR": echo "Currency: Euro"; break; case "JPY": echo "Currency: Japanese yen"; break; default: echo "Unsupported currency"; } ?></pre>
Output
Code: JPY Currency: Japanese yen

คำสั่งวนซ้ำ (Loops)

คำสั่งวนซ้ำ (Loops) เป็นคำสั่งที่ใช้สั่งให้โปรแกรมวนซ้ำ เพื่อทำคำสั่งใน Loop จนกว่าเงื่อนไขจะเป็นเท็จ ใน ภาษา PHP จะมีคำสั่งวนซ้ำอยู่ 4 ตัว ดังนี้

- **while** โปรแกรมจะตรวจสอบเงื่อนไขก่อน ถ้าเงื่อนไขเป็นจริง ก็จะทำคำสั่งในบล็อกวนซ้ำเรื่อยๆ จนกว่าเงื่อนไขจะเป็นเท็จ
- **do...while** โปรแกรมจะทำคำสั่งในบล็อกก่อน 1 ครั้ง แล้วจึงตรวจสอบเงื่อนไข จะหยุดวน Loop ถ้าเงื่อนไขเป็นเท็จ
- **for** โปรแกรมจะวนซ้ำทำคำสั่งในบล็อกตามจำนวน หรือเงื่อนไขที่กำหนด
- **foreach** โปรแกรมจะวนซ้ำทำคำสั่งในบล็อกตามจำนวนสมาชิกของอาร์เรย์

คำสั่ง while

รูปแบบคำสั่ง

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

ตัวอย่างการใช้งาน

PHP Code	Output
<pre><?php \$n = 0; \$m = rand(2,12); while (\$n < 12) { \$n++; echo "\$m x \$n = " . (\$m * \$n) . "
"; } ?></pre>	<pre>6 x 1 = 6 6 x 2 = 12 6 x 3 = 18 6 x 4 = 24 6 x 5 = 30 6 x 6 = 36 6 x 7 = 42 6 x 8 = 48 6 x 9 = 54 6 x 10 = 60 6 x 11 = 66 6 x 12 = 72</pre>

* ผลลัพธ์ที่ได้ ขึ้นอยู่กับค่าตัวแปร \$m ที่สุ่มได้

คำสั่ง do...while

รูปแบบคำสั่ง

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

ตัวอย่างการใช้งาน

PHP Code	Output
<pre><?php do { \$n = rand(0,5); echo "Random number: \$n
"; } while (\$n > 0); ?></pre>	Random number: 3 Random number: 2 Random number: 4 Random number: 1 Random number: 3 Random number: 5 Random number: 0

คำสั่ง for

รูปแบบคำสั่ง

```
for (init counter; test counter; increment counter) {
    code to be executed;
}
```

ตัวอย่างการใช้งาน

PHP Code	Output
<pre><?php \$str = ""; for (\$i=0; \$i<=40; \$i+=5) { \$str .= (\$str == "" ? \$i : ", " . \$i); } echo \$str . "
"; \$str = ""; for (\$i=20; \$i>=-20; \$i-=5) { \$str .= (\$str == "" ? \$i : ", " . \$i); } echo \$str; ?></pre>	0, 5, 10, 15, 20, 25, 30, 35, 40 20, 15, 10, 5, 0, -5, -10, -15, -20

คำสั่ง foreach

รูปแบบคำสั่ง

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

หรือ

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

ตัวอย่างการใช้งาน

PHP Code

```
<?php  
$arr = array("TESLA","BYD","MG","GWM");  
foreach($arr as $x) {  
    echo "$x<br>";  
}  
echo "<hr>";  
  
$arr2 = array("ปากกา" => 25, "ดินสอ" => 5, "ยางลบ" => 12, "ไม้บรรทัด" => 20);  
foreach($arr2 as $item => $price) {  
    echo "$item ราคา $price บาท<br>";  
}  
?>
```

Output

TESLA
BYD
MG
GWM

ปากกา ราคา 25 บาท
ดินสอ ราคา 5 บาท
ยางลบ ราคา 12 บาท
ไม้บรรทัด ราคา 20 บาท

การใช้คำสั่ง break

คำสั่ง break เป็นคำสั่งที่ใช้สั่งให้โปรแกรมหยุดการวน Loop ดังตัวอย่างการใช้งานต่อไปนี้

PHP Code	Output
<pre><?php \$arr = array(8,51,24,72,31,95,0,44,65,17); foreach(\$arr as \$n) { if (\$n < 1) break; echo "\\$n = \$n
"; } ?></pre>	<pre>\$n = 8 \$n = 51 \$n = 24 \$n = 72 \$n = 31 \$n = 95</pre>

การใช้คำสั่ง continue

คำสั่ง continue เป็นคำสั่งที่ใช้สั่งให้โปรแกรมข้ามการทำงานในรอบนั้น แล้วไปทำงานในรอบต่อไป ดังตัวอย่างต่อไปนี้

PHP Code	Output
<pre><?php \$arr = array(8,51,24,72,31,95,0,44,65,17); foreach(\$arr as \$n) { if (\$n < 1) continue; echo "\\$n = \$n
"; } ?></pre>	<pre>\$n = 8 \$n = 51 \$n = 24 \$n = 72 \$n = 31 \$n = 95 \$n = 44 \$n = 65 \$n = 17</pre>

การใช้คำสั่ง for ซ้อน for

คำสั่งตรวจสอบเงื่อนไข คำสั่งวนซ้ำ ทุกอย่างสามารถใช้ซ้อนกันได้ จะซ้อนกี่รอบก็ได้ ดังตัวอย่างต่อไปนี้

PHP Code	Output
<pre><?php \$n = 10; for (\$a = 1; \$a <= \$n; \$a++) { for (\$b = 1; \$b <= \$a; \$b++) { echo (\$a % 5 == 0) ? "0" : "X"; } echo "
"; } ?></pre>	<pre>X XX XXX XXXX 00000 XXXXXXX XXXXXXXX XXXXXXXXXX XXXXXXXXXXX 0000000000</pre>

การรับข้อมูลจากฟอร์ม

การรับข้อมูลจากผู้ใช้ เราจะอาศัยฟอร์มของ HTML ในการรับข้อมูล แล้วให้ HTML ส่งข้อมูลมาให้ PHP เพื่อประมวลผล การรับข้อมูลจากฟอร์มจะมีวิธีรับข้อมูล 3 แบบ ดังนี้

1. การรับข้อมูลแบบ GET

การรับข้อมูลแบบ GET จะใช้กับฟอร์ม ที่ส่งข้อมูลด้วย method="GET" โดย HTML จะนำข้อมูลในฟอร์มส่งมาในรูปแบบ parameter ทาง URL การอ่านข้อมูลแบบ GET จะอ่านจากตัวแปร \$_GET ดังตัวอย่างต่อไปนี้

PHP Code

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>Test</title>
</head>

<?php
$username = null;
if ($_GET) {
    $username = $_GET["uname"];
}
?>

<body>
    <h2>HTML Form</h2>
    <form method="GET" action="">
        <label>กรุณาป้อนชื่อ</label><br>
        <input type="text" id="uname" name="uname" maxlength="25" value="<?= $username; ?>" required>
        <input type="submit" id="btn_ok" name="btn_ok" value="OK">
    </form>
    <?php
    if ($username) {
        echo "<p>สวัสดี $username ยินดีต้อนรับ</p>";
    }
    ?>
</body>
</html>
```

Output

← → ↻ 🏠 ⓘ localhost/form.php?uname=Nineboy&btn_ok=OK

HTML Form

กรุณาป้อนชื่อ

สวัสดี Nineboy ยินดีต้อนรับ

2. การรับข้อมูลแบบ POST

การรับข้อมูลแบบ POST จะใช้กับฟอร์ม ที่ส่งมาจะส่งด้วย method="POST" การอ่านข้อมูลแบบ POST จะอ่านจากตัวแปร \$_POST ดังตัวอย่างต่อไปนี้

PHP Code

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>Test</title>
</head>

<?php
$username = null;
if ($_POST) {
    $username = $_POST["username"];
}
?>

<body>
    <h2>HTML Form</h2>
    <form method="POST" action="">
        <label>กรุณาป้อนชื่อ</label><br>
        <input type="text" id="uname" name="uname" maxlength="25" value="<?= $username; ?>" required>
        <input type="submit" id="btn_ok" name="btn_ok" value="OK">
    </form>
    <?php
    if ($username) {
        echo "<p>สวัสดี $username ยินดีต้อนรับ</p>";
    }
    ?>
</body>
</html>
```

Output

← → ↻ 🏠 ⓘ localhost/form.php

HTML Form

กรุณาป้อนชื่อ

สวัสดี Mis-Boy ยินดีต้อนรับ

3. การรับข้อมูลแบบ REQUEST

การรับข้อมูลแบบ GET หรือ POST จะต้องใช้ให้ตรงกับ method ของฟอร์ม แต่ถ้าไม่ทราบวิธีการส่งข้อมูลของฟอร์ม สามารถใช้ตัวแปร \$_REQUEST รับก็ได้ ซึ่งจะรับได้ทั้งฟอร์มที่ใช้ method GET หรือ POST ดังตัวอย่างต่อไปนี้

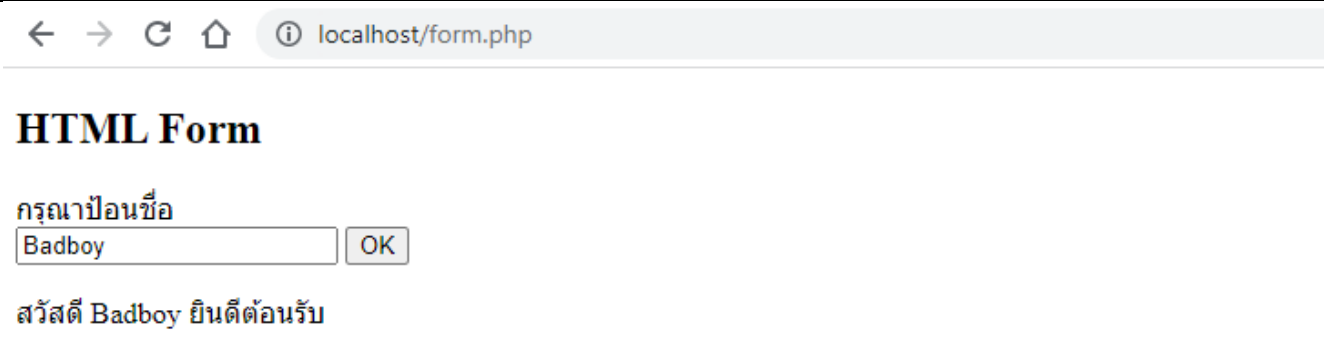
PHP Code

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>Test</title>
</head>

<?php
$username = null;
if ($_REQUEST) {
    $username = $_REQUEST["username"];
}
?>

<body>
    <h2>HTML Form</h2>
    <form method="POST" action="">
        <label>กรุณาป้อนชื่อ</label><br>
        <input type="text" id="username" name="username" maxlength="25" value="<?= $username; ?>" required>
        <input type="submit" id="btn_ok" name="btn_ok" value="OK">
    </form>
    <?php
    if ($username) {
        echo "<p>สวัสดี $username ยินดีต้อนรับ</p>";
    }
    ?>
</body>
</html>
```

Output



* ข้อควรระวังในการรับค่าจากฟอร์ม ต้องพิมพ์ชื่อแท็กให้ถูกต้อง แบบ case-sensitive ด้วย ถ้าไม่ตรงกันจะรับค่าจากฟอร์มไม่ได้

อาร์เรย์ (Array)

อาร์เรย์ (Array) คือประเภทข้อมูลที่เก็บข้อมูลเป็นชุดลำดับเรียงต่อกันในหน่วยความจำ อาร์เรย์เป็นตัวแปรประเภทหนึ่งที่สามารถเก็บข้อมูลได้มากกว่าหนึ่งค่า อาร์เรย์ช่วยอำนวยความสะดวกในกรณีที่เรากำลังจัดการข้อมูลประเภทเดียวกันเป็นจำนวนมาก การประกาศตัวแปรอาร์เรย์ทำได้ 2 แบบ คือใช้ฟังก์ชัน `array()` หรือใช้เครื่องหมาย `[]` ดังตัวอย่างต่อไปนี้

PHP Code

```
<?php
//ใช้ฟังก์ชัน array()
$arr1 = array(); //อาเรย์เปล่า
$arr2 = array(5,8,2,4,6,7,3);
$arr3 = array("Audi","Benz","BMW","Ferrari","Porsche","Volvo");
$arr4 = array("ปากกา" => 25, "ดินสอ" => 5, "ยางลบ" => 12, "ไม้บรรทัด" => 20);

//ใช้ []
$arr5 = [5,8,2,4,6,7,3];
$arr6 = ["Audi","Benz","BMW","Ferrari","Porsche","Volvo"];
$arr7[] = "HTML";
$arr7[] = "CSS";
$arr7[] = "JavaScript";
$arr8["ปากกา"] = 25;
$arr8["ดินสอ"] = 5;
$arr8["ยางลบ"] = 12;
$arr8["ไม้บรรทัด"] = 20;
?>
```

Index ของอาร์เรย์

ตัวแปรอาร์เรย์ที่สร้างขึ้นมา จะมี index กำกับอยู่ทุกตัว โดย index ของอาร์เรย์จะเริ่มต้นที่ 0 เราสามารถแสดงข้อมูลและ index ของอาร์เรย์ได้ โดยใช้ฟังก์ชัน `print_r()` ดังตัวอย่างต่อไปนี้

PHP Code
<pre><?php \$arr2 = array(5,8,2,4,6,7,3); \$arr3 = array("Audi","Benz","BMW","Ferrari","Porsche","Volvo"); \$arr4 = array("ปากกา" => 25, "ดินสอ" => 5, "ยางลบ" => 12, "ไม้บรรทัด" => 20); print_r(\$arr2); echo "<hr>"; print_r(\$arr3); echo "<hr>"; print_r(\$arr4); ?></pre>
Output
<pre>Array ([0] => 5 [1] => 8 [2] => 2 [3] => 4 [4] => 6 [5] => 7 [6] => 3)</pre> <hr/>
<pre>Array ([0] => Audi [1] => Benz [2] => BMW [3] => Ferrari [4] => Porsche [5] => Volvo)</pre> <hr/>
<pre>Array ([ปากกา] => 25 [ดินสอ] => 5 [ยางลบ] => 12 [ไม้บรรทัด] => 20)</pre>

การเข้าถึงข้อมูลในอาร์เรย์ นอกจากการใช้คำสั่ง `foreach` แล้ว ยังสามารถกำหนด index ของอาร์เรย์ที่ต้องการข้อมูลโดยตรงก็ได้ ดังตัวอย่างต่อไปนี้

PHP Code
<pre><?php \$arr = array("Audi","Benz","BMW","Ferrari","Porsche","Volvo"); echo "ข้อมูลใน \\$arr[1] = " . \$arr[1] . "
"; echo "ข้อมูลใน \\$arr[3] = " . \$arr[3] . "
"; echo "ข้อมูลใน \\$arr[5] = " . \$arr[5] . "
"; ?></pre>
Output
<pre>ข้อมูลใน \$arr[1] = Benz ข้อมูลใน \$arr[3] = Ferrari ข้อมูลใน \$arr[5] = Volvo</pre>

การใช้คำสั่งวนซ้ำกับอาร์เรย์

นอกจากคำสั่ง foreach แล้ว คำสั่งวนซ้ำอื่นๆ ไม่ว่าจะเป็น for, while, do...while ก็สามารถใช้กับอาร์เรย์ได้เช่นกัน แต่ต้องระบุ index ของข้อมูลที่ต้องการ ดังตัวอย่างต่อไปนี้

PHP Code
<pre><?php \$arr = array("Audi","Benz","BMW","Ferrari","Porsche","Volvo"); for(\$i=0; \$i<=3; \$i++){ echo "\\$arr[\$i] = " . \$arr[\$i] . "
"; } echo "<hr>"; for(\$i=count(\$arr)-1; \$i>=count(\$arr)-4; \$i--) { echo "\\$arr[\$i] = " . \$arr[\$i] . "
"; } ?></pre>
Output
<pre>\$arr[0] = Audi \$arr[1] = Benz \$arr[2] = BMW \$arr[3] = Ferrari <hr/> \$arr[5] = Volvo \$arr[4] = Porsche \$arr[3] = Ferrari \$arr[2] = BMW</pre>

ตัวอย่างนี้ใช้คำสั่ง for ในการดึงข้อมูลจากอาร์เรย์มาแสดง โดยรอบแรก จะดึงข้อมูลในอาร์เรย์ 4 ตัวแรก และรอบที่สอง จะดึงข้อมูลในอาร์เรย์ 4 ตัวท้ายมาแสดง โดยเรียงลำดับจากตัวท้ายก่อน

อาร์เรย์หลายมิติ

จากตัวอย่างที่ผ่านมา เป็นการใช้งานอาร์เรย์หนึ่งมิติ หากเราต้องการเก็บข้อมูลที่มีหลากหลาย สามารถสร้างอาร์เรย์ไว้ในอาร์เรย์ได้ ดังตัวอย่างต่อไปนี้ จะเป็นตัวอย่างอาร์เรย์ 2 มิติ

PHP Code
<pre><?php \$arr = array(array("อนุชา", 81, "A"), array("สุภาพร", 74, "B"), array("ประเทือง", 68, "C"), array("รวิวรรณ", 90, "A"),); echo "ผลการทดสอบผู้เข้าอบรม
"; for (\$i = 0; \$i < count(\$arr); \$i++) { echo \$arr[\$i][0] . " ได้ " . \$arr[\$i][1] . " คะแนน ได้เกรด " . \$arr[\$i][2] . "
"; } ?></pre>
Output
ผลการทดสอบผู้เข้าอบรม อนุชา ได้ 81 คะแนน ได้เกรด A สุภาพร ได้ 74 คะแนน ได้เกรด B ประเทือง ได้ 68 คะแนน ได้เกรด C รวิวรรณ ได้ 90 คะแนน ได้เกรด A

การระบุข้อมูลในอาร์เรย์หลายมิติ จะใช้การระบุ index ตามจำนวนมิติของแอเรย์ เหมือนตัวอย่าง

การใช้อาร์เรย์หลายมิติที่มีคีย์

PHP Code
<pre><?php \$arr = array("day" => array("จันทร์", "อังคาร", "พุธ", "พฤหัสบดี", "ศุกร์", "เสาร์", "อาทิตย์"), "color" => array("เหลือง", "ชมพู", "เขียว", "ส้ม", "ฟ้า", "ม่วง", "แดง")); for(\$i=0; \$i<7; \$i++){ echo "วัน".\$arr["day"][\$i]." สี".\$arr["color"][\$i]."
"; } ?></pre>
Output
วันจันทร์ สีเหลือง วันอังคาร สีชมพู วันพุธ สีเขียว วันพฤหัสบดี สีส้ม วันศุกร์ สีฟ้า วันเสาร์ สีม่วง วันอาทิตย์ สีแดง

การค้นหาข้อมูลในอาร์เรย์

หากต้องการค้นหาข้อมูลในอาร์เรย์ สามารถใช้ฟังก์ชัน `in_array()` ในการค้นหาได้ ฟังก์ชันนี้จะได้ค่าเป็น `true` ถ้ามีข้อมูลที่ค้นหาในอาร์เรย์ แต่ถ้าไม่มีจะได้ค่าเป็น `false` ดังตัวอย่างต่อไปนี้

PHP Code
<pre><?php \$mypet = array("แมว", "หมา", "นก", "กระต่าย"); if (in_array("เต่า", \$mypet)) { echo "ฉันมีเต่าเป็นสัตว์เลี้ยง"; } else { echo "ฉันไม่มีเต่าเป็นสัตว์เลี้ยง"; } ?></pre>
Output
ฉันไม่มีเต่าเป็นสัตว์เลี้ยง

การจัดเรียงข้อมูลในอาร์เรย์

PHP มีฟังก์ชันสำหรับจัดเรียงข้อมูลในอาร์เรย์ให้ใช้งานหลายฟังก์ชัน แต่ละฟังก์ชันก็จะมีวิธีการจัดเรียงที่แตกต่างกันไป ดังนี้

ฟังก์ชัน `sort()` เป็นฟังก์ชันสำหรับจัดเรียงข้อมูลจากน้อยไปมาก ดังตัวอย่างการใช้งานต่อไปนี้

PHP Code
<pre><?php \$arr = array(5,8,2,4,6,7,3); sort(\$arr); foreach(\$arr as \$n){ echo \$n . " "; } ?></pre>
Output
2 3 4 5 6 7 8

ฟังก์ชัน `rsort()` เป็นฟังก์ชันในการจัดเรียงข้อมูลจากมากไปน้อย ดังตัวอย่างต่อไปนี้

PHP Code
<pre><?php \$arr = array(5,8,2,4,6,7,3); rsort(\$arr); foreach(\$arr as \$n){ echo \$n . " "; } ?></pre>
Output
8 7 6 5 4 3 2

ฟังก์ชัน `asort()` เป็นฟังก์ชันในการจัดเรียงข้อมูล โดยจะเรียงตามค่า (Value) จากน้อยไปมาก ดังตัวอย่างต่อไปนี้

PHP Code
<pre><?php \$arr = array("Cutter" => 55, "Pencil" => 5, "Eraser" => 12, "Ruler" => 20); asort(\$arr); foreach(\$arr as \$key => \$value) { echo \$key . ": " . \$value . "
"; } ?></pre>
Output
<pre>Pencil: 5 Eraser: 12 Ruler: 20 Cutter: 55</pre>

ฟังก์ชัน `arsort()` เป็นฟังก์ชันในการจัดเรียงข้อมูล โดยจะเรียงตามค่า (Value) จากมากไปน้อย ดังตัวอย่างต่อไปนี้

PHP Code
<pre><?php \$arr = array("Cutter" => 55, "Pencil" => 5, "Eraser" => 12, "Ruler" => 20); arsort(\$arr); foreach(\$arr as \$key => \$value) { echo \$key . ": " . \$value . "
"; } ?></pre>
Output
<pre>Cutter: 55 Ruler: 20 Eraser: 12 Pencil: 5</pre>

ฟังก์ชัน `ksort()` เป็นฟังก์ชันในการจัดเรียงข้อมูล โดยจะเรียงตามคีย์ (Key) จากน้อยไปมาก ดังตัวอย่างต่อไปนี้

PHP Code
<pre><?php \$arr = array("Cutter" => 55, "Pencil" => 5, "Eraser" => 12, "Ruler" => 20); ksort(\$arr); foreach(\$arr as \$key => \$value) { echo \$key . ": " . \$value . "
"; } ?></pre>
Output
Cutter: 55 Eraser: 12 Pencil: 5 Ruler: 20

ฟังก์ชัน `krsort()` เป็นฟังก์ชันในการจัดเรียงข้อมูล โดยจะเรียงตามคีย์ (Key) จากมากไปน้อย ดังตัวอย่างต่อไปนี้

PHP Code
<pre><?php \$arr = array("Cutter" => 55, "Pencil" => 5, "Eraser" => 12, "Ruler" => 20); krsort(\$arr); foreach(\$arr as \$key => \$value) { echo \$key . ": " . \$value . "
"; } ?></pre>
Output
Ruler: 20 Pencil: 5 Eraser: 12 Cutter: 55

การนับจำนวน หาผลรวม ข้อมูลในอาร์เรย์

เราสามารถให้ฟังก์ชัน count() เพื่อหาจำนวนข้อมูลในอาร์เรย์ และให้ฟังก์ชัน array_sum() เพื่อหาผลรวมของข้อมูลในอาร์เรย์ได้ ดังตัวอย่างต่อไปนี้

PHP Code
<pre><?php \$score = array(6,4,5,2,7,9,8); echo "อาเรย์ \\$score มีข้อมูลอยู่ " . count(\$score) . " ตัว
"; echo "ผลรวมของข้อมูลในอาเรย์ \\$score เท่ากับ " . array_sum(\$score); ?></pre>
Output
อาเรย์ \$score มีข้อมูลอยู่ 7 ตัว ผลรวมของข้อมูลในอาเรย์ \$score เท่ากับ 41

การเพิ่มข้อมูลในอาร์เรย์

การเพิ่มข้อมูลในอาร์เรย์ จะให้ฟังก์ชัน array_push() เพื่อเพิ่มข้อมูลต่อท้ายในอาร์เรย์ และให้ฟังก์ชัน array_unshift() เพื่อแทรกข้อมูลในอาร์เรย์ด้านหน้า ดังตัวอย่างต่อไปนี้

PHP Code
<pre><?php \$cars = array("Benz","BMW","Ferrari","Porsche"); array_push(\$cars, "Volvo"); foreach(\$cars as \$b) { echo "\$b "; } echo "
"; array_unshift(\$cars, "Audi"); foreach(\$cars as \$b) { echo "\$b "; } ?></pre>
Output
Benz BMW Ferrari Porsche Volvo Audi Benz BMW Ferrari Porsche Volvo

การลบข้อมูลออกจากอาร์เรย์

การลบข้อมูลออกจากอาร์เรย์ จะใช้ฟังก์ชัน `array_pop()` เพื่อลบข้อมูลตัวสุดท้ายออก และฟังก์ชัน `array_shift()` ใช้สำหรับลบข้อมูลตัวแรกออก ดังตัวอย่างต่อไปนี้

PHP Code
<pre><?php \$color = array("Red","Green","Blue","Pink","Yellow"); array_pop(\$color); foreach(\$color as \$c) { echo "\$c "; } echo "
"; array_shift(\$color); foreach(\$color as \$c) { echo "\$c "; } ?></pre>
Output
<pre>Red Green Blue Pink Green Blue Pink</pre>

ฟังก์ชัน (Function)

ฟังก์ชัน (Functions) คือส่วนของโปรแกรมหรือซอสโค้ดที่ใช้สำหรับจัดการกับงานที่เฉพาะเจาะจง เราจะแยกชุดของซอสโค้ดที่มีการทำงานบ่อย หรือใช้งานมากกว่า 1 ครั้ง เป็นฟังก์ชัน เพื่อให้ง่ายต่อการแก้ไข และเขียนโปรแกรมได้เร็วขึ้น ภาษา PHP มีฟังก์ชันอยู่ 2 ประเภท คือ Build-in function เป็นฟังก์ชันพื้นฐานที่ PHP เตรียมไว้ให้เราใช้งาน เช่น ฟังก์ชัน `gettype()`, `print_r()`, `date()`, `rand()`, `trim()`, `substr()`, `number_format()` เป็นต้น ประเภทที่ 2 คือ User-defined function เป็นฟังก์ชันที่เราสร้างขึ้นเองเพื่อวัตถุประสงค์บางอย่าง โดยใช้คำสั่ง `function` ในการสร้างโดยมีรูปแบบดังนี้

```
function function_name (arguments) {
    code to be executed;
    return value;
}
```

การสร้างฟังก์ชัน จะมี arguments และ return value หรือไม่ได้ การตั้งชื่อฟังก์ชันจะมีกฎเหมือนกับการตั้งชื่อตัวแปร และต้องไม่ซ้ำกับคำสงวนของ PHP ดังตัวอย่างต่อไปนี้

PHP Code

```
<?php
// ฟังก์ชัน แบบไม่มี argument และ return value
function date_time() {
    echo date("Y-m-d H:i:s");
}

// ฟังก์ชัน แบบมี parameter แต่ไม่มี return value
function title($str, $h1v) {
    echo "<$h1v>$str</$h1v>";
}

// ฟังก์ชัน แบบมี argument และ return value
function calculate($num1, $num2, $opr) {
    $result = 0;
    if ($opr == "+") $result = $num1 + $num2;
    if ($opr == "-") $result = $num1 - $num2;
    if ($opr == "*") $result = $num1 * $num2;
    if ($opr == "/") $result = $num1 / $num2;
    return $result;
}

// ตัวอย่างการใช้งาน
echo "เวลาปัจจุบัน: ";
date_time();

title("PHP User-defined function", "h2");

echo "9 + 5 = " . calculate(9,5,"+") . "<br>";
echo "9 - 5 = " . calculate(9,5,"-") . "<br>";
echo "9 * 5 = " . calculate(9,5,"*") . "<br>";
echo "9 / 5 = " . calculate(9,5,"/") . "<br>";
?>
```

Output

เวลาปัจจุบัน: 2023-03-02 10:30:25

PHP User-defined function

9 + 5 = 14

9 - 5 = 4

9 * 5 = 45

9 / 5 = 1.8

การกำหนดค่าเริ่มต้นให้ argument

ฟังก์ชันที่มี argument ปกติจะต้องส่งค่าให้ครบตามจำนวน argument ที่กำหนดด้วย ในภาษา PHP เราสามารถกำหนดค่าเริ่มต้นของ argument ได้ ตอนเรียกใช้งานฟังก์ชัน เราสามารถที่จะใส่ค่า หรือไม่ใส่ค่า argument ที่มีการกำหนดค่าเริ่มต้นไว้แล้วก็ได้ ดังตัวอย่างต่อไปนี้

PHP Code

```
<?php
function dayName($dayNo, $lang = "th") {
    $lang = strtolower($lang);
    if (!is_numeric($dayNo)) return "Invalid input dayNo.";
    if ($dayNo < 1 || $dayNo > 7) return "Out of range dayNo.";
    if (!in_array($lang, array("th","en"))) return "Unsupported language.";

    $arr_day = array(
        "th" => array("จันทร์", "อังคาร", "พุธ", "พฤหัสบดี", "ศุกร์", "เสาร์", "อาทิตย์"),
        "en" => array("Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"),
    );

    $ix = $dayNo - 1;
    return $arr_day[$lang][$ix];
}

echo dayName(1) . "<br>";
echo dayName(3, "th") . "<br>";
echo dayName(5, "en") . "<br>";
echo dayName(6, "EN") . "<br>";
?>
```

Output

จันทร์
พุธ
Friday
Saturday

การส่งค่าตัวแปร

การส่งค่าไปยังฟังก์ชัน โดยปกติแล้ว PHP จะทำการคัดลอกค่านั้นไปใส่ในตัวแปรใหม่ เรียกการส่งค่าแบบนี้ว่า Pass by value การส่งค่าแบบนี้ หากมีการเปลี่ยนแปลงค่าของตัวแปรในฟังก์ชัน จะไม่มีผลกับตัวแปรนอกฟังก์ชัน ภาษา PHP ยังมีการส่งค่าอีกแบบ คือ การส่งค่าแบบอ้างอิง (Passing by reference) ซึ่งเป็นการส่งตำแหน่งที่อยู่ของข้อมูลเข้าไปแทน การเปลี่ยนแปลงค่าของตัวแปรที่ส่งแบบนี้ในฟังก์ชัน จะมีผลกับตัวแปรนอกฟังก์ชัน วิธีส่งค่าแบบอ้างอิง จะใส่เครื่องหมาย & ไว้หน้าตัวแปรที่ต้องการ ดังตัวอย่างต่อไปนี้

PHP Code
<pre><?php function reverse_upper(&\$a, \$b) { \$a = strtoupper(strrev(\$a)); \$b = strtoupper(strrev(\$b)); echo "2. \\$a = \$a, \\$b = \$b
"; } \$a = "Ferrari"; \$b = "Porsche"; echo "1. \\$a = \$a, \\$b = \$b
"; reverse_upper(\$a, \$b); echo "3. \\$a = \$a, \\$b = \$b"; ?></pre>
Output
<pre>1. \$a = Ferrari, \$b = Porsche 2. \$a = IRARREF, \$b = EHCSROP 3. \$a = IRARREF, \$b = Porsche</pre>

ฟังก์ชันไม่จำกัด argument

ฟังก์ชันปกติ จะมีการกำหนด argument ที่สามารถส่งข้อมูลเข้าไป แต่ในบางครั้ง ข้อมูลที่ต้องการประมวลผล อาจจะมีจำนวนที่ไม่แน่นอน ทำให้ไม่สามารถกำหนด argument ที่แน่นอนได้ ตัวอย่างนี้จะเป็นตัวอย่างการสร้างฟังก์ชันแบบไม่จำกัด argument

PHP Code
<pre><?php function sum() { if (func_num_args() == 0) { echo "<p>Warning: No arguments.</p>"; return 0; } \$args = func_get_args(); \$sum = 0; foreach(\$args as \$num) { \$sum += \$num; } return \$sum; } \$a = sum(); \$b = sum(9, 14, 11, 6, 21, 15); \$c = sum(12, 5, 10, 8); echo "\\$a = \$a, \\$b = \$b, \\$c = \$c"; ?></pre>
Output
<pre>Warning: No arguments. \$a = 0, \$b = 76, \$c = 35</pre>

ตัวอย่างนี้มีการสร้างฟังก์ชัน sum() ขึ้นมา โดยไม่ได้กำหนด argument แต่จะอาศัยฟังก์ชัน func_get_args() เพื่ออ่าน arguments ทั้งหมดที่ส่งเข้ามาแทน ฟังก์ชัน func_get_args() จะ return ข้อมูลในรูปแบบอาร์เรย์ นอกจากนี้ ยังมีการใช้ฟังก์ชัน func_num_args() เพื่อเช็คว่ามี argument มากี่ตัว ถ้าไม่ส่ง argument มาเลยจะแสดงข้อความ "Warning: No arguments." และ return ค่า 0 กลับ

ฟังก์ชันที่ return ค่า มากกว่าหนึ่งค่า

หากต้องการ return ค่ามากกว่า 1 ค่า ลักษณะนี้ต้อง return ค่าเป็นอาร์เรย์ ดังตัวอย่างต่อไปนี้

PHP Code
<pre><?php function calGrade(\$cs, \$ms, \$fs) { \$total = \$cs + \$ms + \$fs; if (\$total < 0 \$total > 100) return array(\$total, "F"); if (\$total >= 80) return array(\$total, "A"); if (\$total >= 70) return array(\$total, "B"); if (\$total >= 60) return array(\$total, "C"); if (\$total >= 50) return array(\$total, "D"); return array(\$total, "F"); } \$result = calGrade(16,29,33); echo "คะแนนรวม: " . \$result[0] . "
"; echo "เกรดที่ได้: " . \$result[1]; ?></pre>
Output
คะแนนรวม: 78 เกรดที่ได้: B

Global Variables – Superglobals

Superglobals คือ ตัวแปรที่ PHP กำหนดไว้ ตัวแปรเหล่านี้สามารถเข้าถึงได้ตลอด โดยไม่ต้องคำนึงถึงขอบเขต เราสามารถเข้าถึงได้จากฟังก์ชัน คลาส หรือไฟล์ใดๆ โดยไม่ต้องทำอะไรเป็นพิเศษ ตัวแปร Superglobals ใน PHP มีอยู่หลายตัว ดังนี้

- \$GLOBALS: ตัวแปรอาร์เรย์ที่เก็บค่าของตัวแปร Global ทั้งหมดในสคริปต์ที่กำลังใช้งาน
- \$_SERVER: ตัวแปรที่เก็บค่าที่ถูกกำหนดโดยเว็บเซิร์ฟเวอร์ หรือค่าที่ได้จากการทำงานของสคริปต์
- \$_REQUEST: ตัวแปรที่เก็บค่าที่ได้จากการ GET, POST หรือการใช้คุกกี้
- \$_POST: ตัวแปรที่เก็บค่าที่ได้จากการ POST ข้อมูลจาก HTML Form
- \$_GET: ตัวแปรที่เก็บค่า query string หรือ parameter ที่ต่อท้าย URL
- \$_FILES: ตัวแปรอาร์เรย์ใช้เก็บไฟล์ที่ถูกอัปโหลดผ่าน HTTP
- \$_ENV: ตัวแปรอาร์เรย์ที่เก็บค่าตัวแปรสภาพแวดล้อม (Environment variables)
- \$_COOKIE: ตัวแปรที่เก็บค่าตัวแปรคุกกี้
- \$_SESSION: ตัวแปรที่เก็บค่าตัวแปร session ที่สร้างขึ้นในโปรแกรม

ตัวแปรเหล่านี้ บางตัวเราได้พูดถึงไปแล้วในหัวข้อ การรับข้อมูลจากฟอร์ม ในหัวข้อนี้จะพูดถึงตัวแปร Superglobals ที่น่าสนใจอีก 2 ตัว คือ \$GLOBALS และ \$_SERVER ดังนี้

\$GLOBALS

เป็นตัวแปรอาร์เรย์ที่เก็บค่าของตัวแปร Global ทั้งหมดในสคริปต์ที่กำลังทำงานอยู่ไว้ เราสามารถสร้าง เข้าถึง และดูค่าของตัวแปร Global ทั้งหมดได้โดยอาศัยตัวแปร \$GLOBALS ดังตัวอย่างต่อไปนี้

PHP Code
<pre><?php function addition() { \$GLOBALS['c'] = \$GLOBALS['a'] + \$GLOBALS['b']; } \$a = 45; \$b = 52; addition(); print_r(\$GLOBALS); echo "
\\$c = \$c"; ?></pre>
Output
Array ([_GET] => Array () [_POST] => Array () [_COOKIE] => Array () [_FILES] => Array () [a] => 45 [b] => 52 [c] => 97) \$c = 97

การสร้างตัวแปร Global สามารถใช้คำสั่ง global สร้างก็ได้เช่นกัน ดังตัวอย่างต่อไปนี้

PHP Code
<pre><?php function addition() { global \$a, \$b, \$c; \$c = \$a + \$b; } \$a = 45; \$b = 52; addition(); print_r(\$GLOBALS); echo "
\\$c = \$c"; ?></pre>
Output
Array ([_GET] => Array () [_POST] => Array () [_COOKIE] => Array () [_FILES] => Array () [a] => 45 [b] => 52 [c] => 97) \$c = 97

\$_SERVER

เป็นตัวแปรที่เก็บค่าที่ถูกกำหนดโดยเว็บเซิร์ฟเวอร์ หรือค่าที่ได้จากการทำงานของสคริปต์ในขณะนั้น
Element ที่สำคัญในตัวแปร \$_SERVER จะมีดังนี้

Element	Description
\$_SERVER['PHP_SELF']	Return ชื่อไฟล์สคริปต์ปัจจุบัน เช่น /server.php
\$_SERVER['GATEWAY_INTERFACE']	Return เวอร์ชันของ CGI ที่เซิร์ฟเวอร์ใช้งาน
\$_SERVER['SERVER_ADDR']	Return IP ของเครื่องเซิร์ฟเวอร์
\$_SERVER['SERVER_NAME']	Return ชื่อของ host server เช่น test.winwin.co.th
\$_SERVER['SERVER_SOFTWARE']	Return ข้อมูลซอฟต์แวร์ของเซิร์ฟเวอร์
\$_SERVER['SERVER_PROTOCOL']	Return ชื่อและเวอร์ชันของ Protocol
\$_SERVER['REQUEST_METHOD']	Return วิธีการ (Method) ในเข้าถึงเว็บเพจ
\$_SERVER['REQUEST_TIME']	Return Timestamp ที่มีการเข้าสู่เว็บเพจ เช่น 1680322397
\$_SERVER['QUERY_STRING']	Return Parameters ที่ระบุต่อท้าย URL (ถ้ามี)
\$_SERVER['HTTP_ACCEPT']	Return header ที่สามารถใช้ได้
\$_SERVER['HTTP_ACCEPT_CHARSET']	Return header charset ที่สามารถใช้ได้ เช่น utf-8, ISO-8859-1
\$_SERVER['HTTP_HOST']	Return host header เช่น test.winwin.co.th
\$_SERVER['HTTP_REFERER']	Return URL ก่อนหน้าที่เรียกมาที่เพจปัจจุบัน
\$_SERVER['HTTPS']	Return on ถ้าเว็บไซต์ปัจจุบันเป็น secure HTTP หรือ https
\$_SERVER['REMOTE_ADDR']	Return IP ของเครื่อง client ที่เรียกดูเว็บ
\$_SERVER['REMOTE_HOST']	Return ชื่อ Host ของเครื่อง client ที่เรียกมาที่เว็บ
\$_SERVER['REMOTE_PORT']	Return port ของเครื่อง client ที่เรียกมาที่เว็บ
\$_SERVER['SCRIPT_FILENAME']	Return pathname ของสคริปต์ที่ทำงานอยู่ เช่น /home/www/test/server.php
\$_SERVER['SERVER_ADMIN']	Return admin ของเซิร์ฟเวอร์ที่กำหนดไว้ในไฟล์คอนฟิก
\$_SERVER['SERVER_PORT']	Return port ของเซิร์ฟเวอร์ที่ใช้อยู่ เช่น 80, 443
\$_SERVER['SCRIPT_NAME']	Return ชื่อไฟล์สคริปต์ปัจจุบัน เช่น /server.php

ตัวอย่างการใช้งาน

PHP Code	Output
<pre><?php echo \$_SERVER['SERVER_NAME']; echo "
"; echo \$_SERVER['SERVER_PROTOCOL']; echo "
"; echo \$_SERVER['REQUEST_METHOD']; ?></pre>	<pre>localhost HTTP/1.1 GET</pre>

วันที่และเวลา

PHP มีฟังก์ชัน `date()` สำหรับจัดรูปแบบของวันที่และเวลา โดยมีรูปแบบการใช้งานดังนี้

`date(format, timestamp)`

โดยที่ `format` คือรูปแบบของวันที่และเวลา และ `timestamp` คือวันที่และเวลาที่ต้องการกำหนด ซึ่งจะไม่กำหนดก็ได้ ถ้าไม่กำหนดฟังก์ชันจะใช้วันที่และเวลาปัจจุบัน

Format ทัวไปสำหรับฟังก์ชัน `date()`

ต่อไปนี้เป็นตัวอย่างตัวอักษรทั่วไปที่ใช้กำหนด `format`

ตัวอักษร	คำอธิบาย
d	แทนเลขวันที่ของแต่ละเดือน เช่น เดือน ม.ค. จะมีค่าอยู่ในช่วง 01 ถึง 31
j	แทนเลขวันที่ของแต่ละเดือน คล้ายกับ d แต่จะไม่มี 0 นำหน้า จะได้ค่าอยู่ในช่วง 1 ถึง 31
m	แทนเลขเดือน ตั้งแต่ 01 ถึง 12
n	แทนเลขเดือนเหมือน m แต่จะไม่มี 0 นำหน้า จะได้ค่าอยู่ในช่วง 1 ถึง 12
Y	แทนเลขปี (ค.ศ.) 4 หลัก เช่น 2023
y	แทนเลขปี (ค.ศ.) 2 หลัก เช่น 23
l	แทนชื่อวันในแต่ละอาทิตย์
N	แทนเลขวันในแต่ละอาทิตย์ โดยที่ 1 คือ วันจันทร์, 2 คือ วันอังคาร,..., 7 คือวันอาทิตย์
H	แทนเลขชั่วโมง ตั้งแต่ 00 ถึง 23
h	แทนเลขชั่วโมง ตั้งแต่ 01 ถึง 12
i	แทนเลขนาที ตั้งแต่ 00 ถึง 59
s	แทนเลขวินาที ตั้งแต่ 00 ถึง 59

* ดูเพิ่มเติมได้ที่ https://www.w3schools.com/php/func_date_date.asp

การแสดงผลวันที่ และเวลา ปัจจุบัน

การแสดงผลวันที่ และเวลา ปัจจุบัน จะใช้ฟังก์ชัน `date()` ดังตัวอย่างต่อไปนี้

PHP Code	Output
<pre><?php echo "Today is " . date("Y-m-d H:i:s") . "
"; echo "Today is " . date("d/m/Y H:i:s") . "
"; echo "Today is " . date("l") . "
"; ?></pre>	<pre>Today is 2023-04-01 15:46:44 Today is 01/04/2023 15:46:44 Today is Saturday</pre>

การสร้างวันที่จากข้อความ

การสร้างวันที่ หรือ timestamp ใน PHP สามารถใช้ฟังก์ชัน `strtotime()` ในการสร้าง ดังตัวอย่างต่อไปนี้

PHP Code
<pre><?php echo "Current date: " . date("Y-m-d") . "
"; echo date("Y-m-d", strtotime("tomorrow")) . "
"; echo date("Y-m-d", strtotime("yesterday")) . "
"; echo date("Y-m-d", strtotime("next Saturday")) . "
"; echo date("Y-m-d", strtotime("+4 days")) . "
"; echo date("Y-m-d", strtotime("-3 days")) . "
"; echo date("Y-m-d", strtotime("+2 weeks ")) . "
"; echo date("Y-m-d", strtotime("+1 months ")) . "
"; \$c = time(); \$d = strtotime("2023-03-02"); echo "
Current Timestamp is \$c
"; echo "Timestamp of 2023-03-02 is \$d
"; echo date("Y-m-d", \$d) . "
"; ?></pre>
Output
<pre>Current date: 2023-04-01 2023-04-02 2023-03-31 2023-04-08 2023-04-05 2023-03-29 2023-04-15 2023-05-01 Current Timestamp is 1680340850 Timestamp of 2023-03-02 is 1677690000 2023-03-02</pre>

การสร้างวันที่ด้วยฟังก์ชัน `mktime()`

การสร้างวันที่หรือ timestamp ด้วยฟังก์ชัน `strtotime()` จะสร้าง timestamp จากข้อความ แต่ฟังก์ชัน `mktime()` จะสร้าง timestamp จากตัวเลข โดยมีรูปแบบการใช้งานดังนี้

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

ตัวอย่างการใช้งาน

PHP Code	Output
<pre><?php \$d = mktime(10,15,30,3,2,2023); echo date("Y-m-d H:i:s", \$d); ?></pre>	<pre>2023-03-02 10:15:30</pre>

การ Include ไฟล์

เราสามารถเขียนโค้ด ตัวแปร ค่าคงที่ หรือฟังก์ชัน แยกไว้เป็นไฟล์ต่างหาก เมื่อต้องการใช้งาน ก็ทำการ include ไฟล์นั้นเข้ามา ทำให้เราสามารถใส่ตัวแปร ค่าคงที่ หรือฟังก์ชันที่อยู่ในไฟล์ได้ เหมือนกับการเขียนไว้ในไฟล์ที่ทำงานอยู่ ซึ่งจะช่วยให้ประหยัดเวลาการเขียนโปรแกรม และยังง่ายต่อการแก้ไข

การ include ไฟล์จะมีคำสั่ง ให้ใช้งาน 2 คำสั่ง คือ include กับ require ซึ่งทำหน้าที่เหมือนกัน ต่างกันที่การจัดการ กรณีที่เกิดข้อผิดพลาด คำสั่ง include จะแสดง WARNING และทำงานต่อ แต่คำสั่ง require จะแสดง ERROR และหยุดการทำงาน ดังตัวอย่างต่อไปนี้

PHP Code	
file: common.php	
<pre><?php \$office_ip = array("180.183.250.244","180.183.248.19","49.0.80.115","49.0.82.21"); function get_ip(){ \$ip = \$_SERVER["REMOTE_ADDR"]; return substr(\$ip, 0, 50); } ?></pre>	
file: include.php	
<pre><?php include "common.php"; \$client_ip = get_ip(); echo "<p>IP Address: \$client_ip</p>"; if (!in_array(\$client_ip, \$office_ip)) { die("Sorry access denied!"); } else { echo "Welcome back."; } ?></pre>	
Output	
IP Address: ::1	IP Address: 180.183.250.244
Sorry access denied!	Welcome back.

นอกคำสั่ง include กับ require แล้ว ยังมีอีก 2 คำสั่ง คือ include_once และ require_once ซึ่งทำหน้าที่เหมือนกับ include และ require แต่คำสั่ง include_once และ require_once จะตรวจสอบไฟล์ที่ include ก่อนว่าเคย include มาก่อนแล้วหรือยัง ถ้าเคยแล้วจะไม่ include ซ้ำอีก และ 2 คำสั่งนี้จะประมวลผลช้ากว่า เพราะต้องเสียเวลาในการตรวจสอบไฟล์ที่ include เข้ามา

การใช้งาน Session

ตัวแปร Session เป็นตัวแปรหน่วยความจำ ที่สามารถเรียกใช้งานได้ตลอดเวลา ทั้งในหรือนอก class, function เราสามารถเก็บข้อมูลบางอย่างไว้ใน session เพื่อใช้งานในเพจต่างๆ เช่น เก็บ username ลูกค้าที่ Login เข้าสู่ระบบ เป็นต้น โดยปกติแล้วค่าในตัวแปร session จะมีอายุอยู่ได้ประมาณ 20 นาที หากไม่ได้ใช้งานเกินกว่าเวลาที่กำหนด หรือผู้ใช้ปิด browser ตัวแปร session ก็หายไปด้วย การใช้งาน session มีฟังก์ชันที่เกี่ยวข้องดังนี้

ฟังก์ชัน	คำอธิบาย
session_start()	ฟังก์ชันสำหรับเริ่มใช้งาน session (ต้องใช้ฟังก์ชันนี้ทุกครั้งที่จะใช้ตัวแปร Session)
session_id()	ฟังก์ชันนี้จะให้เลข session กลับมา ซึ่งแต่ละเครื่อง หรือแต่ละ browser จะมีเลขที่แตกต่างกัน
session_destroy()	ฟังก์ชันสำหรับทำลายตัวแปร session
session_unset()	ฟังก์ชันสำหรับยกเลิกตัวแปร session ทั้งหมด
unset()	ฟังก์ชันสำหรับยกเลิกตัวแปรที่กำหนด

ตัวอย่างการใช้งาน

PHP Code
<pre><?php session_start(); \$_SESSION["code"] = "IT241"; \$_SESSION["name"] = "Sayan"; \$_SESSION["type"] = 2; echo "<p>รหัสพนักงาน: ".\$_SESSION["code"]."
"; echo "Session ID: ".session_id()."</p>"; print_r(\$_SESSION); echo "<hr>"; unset(\$_SESSION["type"]); print_r(\$_SESSION); echo "<hr>"; session_unset(); session_destroy(); print_r(\$_SESSION); ?></pre>
Output
<pre>รหัสพนักงาน: IT241 Session ID: 3fksmpn2scr2q6d6h317noc3uq Array ([code] => IT241 [name] => Sayan [type] => 2) Array ([code] => IT241 [name] => Sayan) Array ()</pre>

PHP กับ JSON

JSON ย่อมาจาก JavaScript Object Notation เป็นไวยากรณ์สำหรับจัดเก็บและแลกเปลี่ยนข้อมูล มีรูปแบบเป็นข้อความ ตัวอย่างเช่น {"code":"000","message":"success","id":"9105","point":"22500"} ภาษา PHP มีฟังก์ชันสำหรับทำงานกับ JSON 2 ตัว คือ json_encode() และ json_decode()

json_encode()

ฟังก์ชัน json_encode() ใช้สำหรับแปลงข้อมูลให้อยู่ในรูปแบบ JSON ตัวอย่างการใช้งาน

PHP Code
<pre><?php \$cars = array("Audi","Benz","BMW","Ferrari","Porsche","Volvo"); \$product = array("Cutter" => 55, "Pencil" => 5, "Eraser" => 12, "Ruler" => 20); echo json_encode(\$cars); echo "
"; echo json_encode(\$product); ?></pre>
Output
<pre>["Audi","Benz","BMW","Ferrari","Porsche","Volvo"] {"Cutter":55,"Pencil":5,"Eraser":12,"Ruler":20}</pre>

json_decode()

ฟังก์ชัน json_decode() ใช้สำหรับแปลง JSON ให้เป็น PHP Object หรือ Array

ตัวอย่างการใช้งาน 1 แปลง JSON เป็น Object

PHP Code
<pre><?php \$jsonObj = '{"Cutter":55,"Pencil":5,"Eraser":12,"Ruler":20}'; \$obj = json_decode(\$jsonObj); echo \$obj->Cutter . "
"; echo \$obj->Ruler . "<hr>"; foreach(\$obj as \$k => \$v) { echo "\$k ราคา \$v บาท
"; } ?></pre>

Output
55 20 <hr/> Cutter ราคา 55 บาท Pencil ราคา 5 บาท Eraser ราคา 12 บาท Ruler ราคา 20 บาท

ตัวอย่างการใช้งาน 1 แปลง JSON เป็น Array

PHP Code
<pre><?php \$jsonObj = '{"Cutter":55,"Pencil":5,"Eraser":12,"Ruler":20}'; \$arr = json_decode(\$jsonObj, true); echo \$arr["Pencil"] . "
"; echo \$arr["Eraser"] . "<hr>"; foreach(\$arr as \$k => \$v) { echo "\$k ราคา \$v บาท
"; } ?></pre>
Output
5 12 <hr/> Cutter ราคา 55 บาท Pencil ราคา 5 บาท Eraser ราคา 12 บาท Ruler ราคา 20 บาท

การใช้ฟังก์ชัน file_get_contents()

ฟังก์ชัน file_get_contents() เป็นฟังก์ชันสำหรับอ่านข้อมูลจากไฟล์ หรือจากเว็บมาใช้งานในโปรแกรม

ตัวอย่างการใช้งาน 1 อ่านข้อมูลจากไฟล์

PHP Code
<pre><?php \$str = file_get_contents("assets/txt/fsp3.txt"); echo \$str; ?></pre>

ตัวอย่างการใช้งาน 2 อ่านข้อมูลจากเว็บ

PHP Code
<pre><?php \$api_url = "http://test.winwin.co.th/api/getproduct.php?sortkey=qty"; \$response = @file_get_contents(\$api_url); if (!\$response) { die("อ่านข้อมูลจาก API ไม่สำเร็จ"); } \$json = json_decode(\$response); if (\$json->code != "000") { die("เกิดข้อผิดพลาด " . \$json->code . ": " . \$json->message); } echo "สินค้าในสต็อก:"; echo ""; foreach(\$json->products as \$p) { echo "" . \$p->name . " ราคา " . \$p->price . " บาท/" . \$p->unit . " ยอดคงเหลือ " . \$p->qty . " " . \$p->unit . ""; } echo ""; ?></pre>
Output
<p>สินค้าในสต็อก:</p> <ol style="list-style-type: none">1. ดินสอ ราคา 45 บาท/แพ็ค ยอดคงเหลือ 59 แพ็ค2. ไม้บรรทัด ราคา 20 บาท/ชิ้น ยอดคงเหลือ 77 ชิ้น3. ยางลบ ราคา 10 บาท/ก้อน ยอดคงเหลือ 142 ก้อน4. ปากกา ราคา 15 บาท/ด้าม ยอดคงเหลือ 160 ด้าม

การเขียนโปรแกรมเชิงวัตถุ

การเขียนโปรแกรมเชิงวัตถุ (Object-oriented programming: OOP) นั้นเป็นการเขียนโปรแกรมโดยมองทุกอย่างเหมือนเป็นเหมือนออบเจกต์ (Object) ซึ่งจะประกอบไปด้วยข้อมูล ในรูปแบบของคุณสมบัติ (Property) และกระบวนการทำงาน (Method) คุณสมบัตินี้ที่สำคัญของออบเจกต์ คือ Method จะทำงานและจัดการกับ Property ของออบเจกต์นั้น ในภาษา PHP นั้นออบเจกต์ถูกสร้างมาจากคลาส (Class) ซึ่งหมายความว่าออบเจกต์เป็นตัวแปรของคลาส (Class instance)

ออบเจกต์และคลาส

- **คลาส** เปรียบเสมือน ต้นแบบ หรือ พิมพ์เขียว ที่ใช้สร้างออบเจกต์ คลาสแต่คลาสสามารถสร้างเป็นออบเจกต์ได้ไม่จำกัด
- **ออบเจกต์** คือ ตัวแปรชนิดหนึ่งที่ถูกสร้างมาจากคลาส หรือ Class Instance

ในการเขียนโปรแกรมเชิงวัตถุ มีคุณสมบัติที่สำคัญที่ทำให้การเขียนโปรแกรมมีประสิทธิภาพมากขึ้น ช่วยให้โค้ดมีความยืดหยุ่น ปลอดภัย และสามารถใช้งานโค้ดเดิมซ้ำๆ โดยไม่ต้องเขียนขึ้นใหม่ ซึ่งจะช่วยลดเวลาในการพัฒนาโปรแกรมลงได้เป็นอย่างมาก

การสร้างคลาส

การสร้างคลาสจะใช้คำสั่ง class ตามด้วยชื่อคลาส และเครื่องหมาย {} ดังตัวอย่างนี้

PHP Code

```
<?php
class Car {
    // Properties
    public $model;
    public $color;

    // Method
    function set_model($model) {
        $this->model = $model;
    }
    function get_model() {
        return $this->model;
    }

    function set_color($color) {
        $this->color = $color;
    }
    function get_color() {
        return $this->color;
    }
}
```

การสร้างออบเจ็ค

คลาสจะไม่มีประโยชน์อะไร ถ้าไม่มีการสร้างออบเจ็ค เราสามารถสร้างออบเจ็คหลายๆ อันจากคลาสเดียว โดยที่แต่ละออบเจ็คจะมี property และ method เหมือนกับคลาสที่กำหนด การสร้างออบเจ็คจะใช้คีย์เวิร์ด new ตัวอย่างต่อไปนี้ จะสร้างออบเจ็ค \$car1 และ \$car2 จากคลาส Car

PHP Code	Output
<pre><?php class Car { // Properties public \$model; public \$color; // Method function set_model(\$model) { \$this->model = \$model; } function get_model() { return \$this->model; } function set_color(\$color) { \$this->color = \$color; } function get_color() { return \$this->color; } } \$car1 = new Car(); \$car2 = new Car(); \$car1->set_model("Toyota Yaris"); \$car1->set_color("White"); \$car2->set_model("Honda City"); \$car2->set_color("Red"); echo \$car1->get_model() . " is " . \$car1->get_color(); echo "
"; echo \$car2->get_model() . " is " . \$car2->get_color(); ?></pre>	<pre>Toyota Yaris is White Honda City is Red</pre>

การใช้คีย์เวิร์ด \$this

คีย์เวิร์ด \$this ใช้สำหรับอ้างถึงออบเจกต์ที่อยู่ภายในคลาส ไม่ว่าจะเป็น property หรือ method ดังจะเห็นจากตัวอย่างก่อนหน้านี้ มีการใช้คีย์เวิร์ด \$this เพื่ออ้างถึงตัวแปร \$model และ \$color กรณีที่ต้องการอ้างถึงออบเจกต์จากภายนอกคลาส ก็สามารทำได้ ดังตัวอย่างนี้

PHP Code	Output
<pre><?php class myCar { public \$model; public \$color; } \$mycar = new myCar(); \$mycar->model = "Toyota Vios"; \$mycar->color = "สีดำ"; echo "ฉันใช้ " . \$mycar->model . " สี " . \$mycar->color; ?></pre>	ฉันใช้ Toyota Vios สีดำ

การใช้คีย์เวิร์ด instanceof

เราสามารถห้คีย์เวิร์ด instanceof เพื่อตรวจสอบว่าออบเจกต์ที่สร้างเป็น instance ของคลาสที่กำหนดหรือไม่ได้ ดังอตัวอย่างต่อไปนี้

PHP Code	Output
<pre><?php class myCar { public \$model; public \$color; } class book { public \$name; } \$obj = new myCar(); var_dump(\$obj instanceof myCar); echo "
"; var_dump(\$obj instanceof book); ?></pre>	bool(true) bool(false)

คอนสตรัคเตอร์และดีสตรัคเตอร์

คอนสตรัคเตอร์ (Constructor) คือ ฟังก์ชันที่จะถูกเรียกขึ้นมาทำงานทันทีที่มีการสร้างออบเจกต์จากคลาส ส่วนดีสตรัคเตอร์ (Destructor) เป็นฟังก์ชันที่จะถูกเรียกขึ้นมาทำงานก่อนที่ออบเจกต์ที่สร้างจะถูกทำลายและคืนหน่วยความจำให้แก่ระบบ หรือตอนที่สคริปต์สิ้นสุดลง ตัวอย่างการใช้งาน

PHP Code

```
<?php
class Car {
    public $model;
    public $color;

    function __construct($model, $color) {
        $this->model = $model;
        $this->color = $color;
        echo "<p>สร้างออบเจกต์ Car ขึ้นมา<br>";
        echo "รถของคุณคือ " . $this->model . " สี" . $this->color . "</p>";
    }
    function __destruct() {
        echo "<p>ออบเจกต์ถูกทำลายแล้ว<br>";
        echo "รถของคุณคือ " . $this->model . " สี" . $this->color . "</p>";
    }
}

$car = new Car("Honda Civic", "ขาวแพลทินัม");
$car->color = "เทาโซนิค";
?>
```

Output

```
สร้างออบเจกต์ Car ขึ้นมา
รถของคุณคือ Honda Civic สีขาวแพลทินัม

ออบเจกต์ถูกทำลายแล้ว
รถของคุณคือ Honda Civic สีเทาโซนิค
```

การจำกัดการเข้าถึง

เราสามารถจำกัดการเข้าถึงตัวแปรหรือฟังก์ชันต่างๆ ในคลาสได้ โดยเพิ่มคีย์เวิร์ด Public, Private หรือ Protected หน้าตัวแปร หรือฟังก์ชัน โดยปกติ ถ้าไม่มีคีย์เวิร์ดเหล่านี้ PHP จะกำหนดให้เป็น Public คีย์เวิร์ดแต่ละตัวมีความแตกต่างกัน ดังนี้

- public ตัวแปรหรือฟังก์ชันที่สามารถเข้าถึงได้ทุกที่ในโปรแกรม
- private ตัวแปรหรือฟังก์ชันที่สามารถเข้าถึงได้ภายในคลาสเท่านั้น
- protected ตัวแปรหรือฟังก์ชันที่สามารถเข้าถึงได้จากในคลาสเดียวกันหรือคลาสที่ได้รับการสืบทอด (Inheritance class)

ตัวอย่างการใช้งาน

PHP Code

```
<?php
class Book {
    public $name;
    private $editor;
    protected $price;

    public function set_name($name) {
        $this->name = $name;
    }
    private function set_editor($ename) {
        $this->editor = $ename;
    }
    protected function set_price($price) {
        $this->price = $price;
    }
}

$book = new Book();
$book->name = "คู่มือการเขียนโปรแกรมภาษา PHP";
$book->editor = "ธนากร วงเจริญกิจ";
$book->price = 299;

$book->set_name("คู่มือการเขียนโปรแกรมภาษา PHP");
$book->set_editor("ธนากร วงเจริญกิจ");
$book->set_price(299);
?>
```

จากตัวอย่างจะเห็นว่า ตัวแปร \$editor, \$price และฟังก์ชัน set_editor(), set_price() ไม่สามารถเข้าถึงจากภายนอกคลาสได้

การสืบทอด (Inheritance)

คลาสลูกที่ถูกสร้างขึ้นมาจะมีคุณสมบัติและ method เหมือนคลาสแม่ทุกอย่าง สามารถกำหนดตัวแปรหรือฟังก์ชันเพิ่มเติมเพื่อให้คลาสลูก หรือปรับเปลี่ยนการทำงานของฟังก์ชันให้มีความแตกต่างจากคลาสแม่ได้ ซึ่งจะช่วยให้การพัฒนาระบบทำได้สะดวกและรวดเร็วขึ้น เพราะไม่ต้องเขียนโค้ดใหม่ทั้งหมด และยังช่วยลดข้อผิดพลาดในการเขียนโค้ดใหม่ การสร้างคลาสลูก จะใช้คีย์เวิร์ด extends เพิ่มเติม ตอนสร้างคลาส ดังตัวอย่างต่อไปนี้

PHP Code

```
<?php
class book1 {
    public $name;
    public $price;

    public function setBook($name, $price) {
        $this->name = $name;
        $this->price = $price;
    }
    public function getBook() {
        return array($this->name, $this->price);
    }
}

class book2 extends book1 {
    public $qty;

    public function setQty($qty){
        $this->qty = $qty;
    }
    public function getBook() {
        return array($this->name, $this->price, $this->qty);
    }
}

$b = new book2();
$b->setBook("Full Stack Programmer", 299);
$b->setQty(45);
echo "หนังสือ " . $b->getBook()[0] . "<br>";
echo "ราคา " . $b->getBook()[1] . " บาท<br>";
echo "คงเหลือ: " . $b->getBook()[2] . " เล่ม";
?>
```

Output

หนังสือ Full Stack Programmer
ราคา 299 บาท
คงเหลือ: 45 เล่ม

ค่าคงที่ของคลาส

เราสามารถประกาศค่าคงที่ในคลาสได้ โดยใช้คำสั่ง `const` เหมือนการประกาศค่าคงที่ปกติ หลังจากประกาศค่าคงที่แล้ว สามารถเข้าถึงค่าคงที่ที่ประกาศไว้จากภายในคลาสได้ โดยใช้คีย์เวิร์ด `self` และใช้โอเปอเรเตอร์ `::` สำหรับการเข้าถึงค่าคงที่จากภายนอกคลาส ดังตัวอย่างต่อไปนี้

PHP Code

```
<?php
class test {
    const MESSAGE = "Hello";
    const POSITION = "Developer";
    public function sayHi() {
        echo self::MESSAGE;
    }
}

$obj = new test();
$obj->sayHi();
echo "<br>" . test::POSITION;
?>
```

Output

```
Hello
Developer
```

Abstract Class

Abstract Class คือคลาสที่ยังไม่สมบูรณ์ โดยจะสร้าง Abstract Function ไว้ (Abstract Function คือ ฟังก์ชันที่ไม่มีการเขียนคำสั่งในฟังก์ชัน) คลาสลูกที่มาสืบทอดต้องเขียน Abstract Function ต่อเอาเอง Abstract Class มีไว้สำหรับเป็นแม่แบบให้คนที่มาใช้ ทำการพัฒนาได้ที่กำหนดไว้ต่อ Abstract Class ไม่สามารถนำมาสร้างเป็นออบเจกต์ได้ การสร้าง Abstract Class จะใช้คีย์เวิร์ด **abstract** หน้าคำสั่ง class ดังตัวอย่างนี้

PHP Code

```
<?php
abstract class item {
    public $name, $price, $unit;
    public function __construct($name, $price, $unit) {
        $this->name = $name;
        $this->price = $price;
        $this->unit = $unit;
    }
    abstract public function showItem();
}

class fruit extends item {
    public function showItem() {
        echo "<p>" . $this->name . " ราคา " . $this->price . " บาท/" . $this->unit . "</p>";
    }
}

class book extends item {
    public function showItem() {
        echo "<p>หนังสือ: " . $this->name . " ราคา " . $this->price . " บาท/" . $this->unit . "</p>";
    }
}

$a = new fruit("องุ่น", 80, "กิโลกรัม");
$b = new book("Advance PHP", 395, "เล่ม");
$a->showItem();
$b->showItem();
?>
```

Output

องุ่น ราคา 80 บาท/กิโลกรัม

หนังสือ: Advance PHP ราคา 395 บาท/เล่ม

Interfaces

อินเตอร์เฟส (Interfaces) คือ Abstract class ที่ประกอบด้วย Abstract function ทั้งหมด ในอินเตอร์เฟสจะมีเพียงการประกาศฟังก์ชันเท่านั้น ไม่สามารถกำหนด Properties หรือประกาศตัวแปรได้ ฟังก์ชันทั้งหมดในอินเตอร์เฟส จะถือเป็น Abstract function โดยปริยาย โดยไม่จำเป็นต้องใส่คีย์เวิร์ด **abstract** ไว้ตอนประกาศฟังก์ชัน และต้องกำหนดให้เป็น **public** เท่านั้น ไม่สามารถกำหนดเป็น **private** หรือ **protected** ได้

การสร้างคลาสใหม่ให้สืบทอดจากอินเตอร์เฟซ จะใช้คีย์เวิร์ด implements คลาสลูกที่ทำการสืบทอดจากอินเตอร์เฟซ ต้องสร้างฟังก์ชันให้ครบตามที่ประกาศอยู่ในอินเตอร์เฟซ และต้องเป็น public เหมือนอินเตอร์เฟซด้วย

ตัวอย่างการใช้งานอินเตอร์เฟซ

PHP Code
<pre><?php interface Shape { public function area(\$var1, \$var2); } class Rectangle implements Shape { public function area(\$w, \$l) { echo "<p>สี่เหลี่ยมผืนผ้า กว้าง \$w ซม. ยาว \$l ซม." . "
มีพื้นที่เท่ากับ " . (\$w * \$l) . " ซม.</p>"; } } class Triangle implements Shape { public function area(\$w, \$h) { echo "<p>สามเหลี่ยม ฐาน \$w ซม. สูง \$h ซม." . "
มีพื้นที่เท่ากับ " . (0.5 * \$w * \$h) . " ซม.</p>"; } } \$shape1 = new Rectangle(); \$shape2 = new Triangle(); \$shape1->area(5, 6); \$shape2->area(6, 7); ?></pre>
Output
<p>สี่เหลี่ยมผืนผ้า กว้าง 5 ซม. ยาว 6 ซม. มีพื้นที่เท่ากับ 30 ซม.</p> <p>สามเหลี่ยม ฐาน 6 ซม. สูง 7 ซม. มีพื้นที่เท่ากับ 21 ซม.</p>

ความแตกต่างของ Interface และ Abstract Class

- อินเตอร์เฟซไม่สามารถกำหนด properties หรือตัวแปรได้ แต่ Abstract Class ได้
- ฟังก์ชันในอินเตอร์เฟซต้องเป็น public เท่านั้น แต่ Abstract Class ใช้ public หรือ protected
- ฟังก์ชันในอินเตอร์เฟซทั้งหมดต้องเป็น Abstract Function แต่ Abstract Class ไม่จำเป็น
- การสืบทอดของคลาสลูกอินเตอร์เฟซใช้คีย์เวิร์ด implements แต่ Abstract Class ใช้ extends

Static Methods และ Static Properties

การกำหนด static ให้กับตัวแปร หรือฟังก์ชันในคลาส จะเป็นการอนุญาตให้สามารถเรียกใช้ตัวแปรหรือฟังก์ชันนั้นจากในคลาสได้ โดยไม่ต้องสร้างเป็นออบเจกต์ก่อน โดยใช้โอเปอเรเตอร์ :: ดังตัวอย่างต่อไปนี้

PHP Code
<pre><?php class Book { public static \$name = "Advanced PHP"; public static \$price = 299; public static function Price() { return Book::\$price; } } echo "<p>หนังสือ " . Book::\$name . " ราคา " . Book::Price() . " บาท</p>"; ?></pre>
Output
หนังสือ Advanced PHP ราคา 299 บาท

การเข้าถึงตัวแปร และฟังก์ชันแบบ static จากฟังก์ชันปกติ จะใช้คีย์เวิร์ด self และโอเปอเรเตอร์ :: ดังตัวอย่างนี้

PHP Code
<pre><?php class Book { public static \$name; public static \$price; public static function echobook() { echo "<p>เพิ่มหนังสือ " . Book::\$name . " เรียบร้อยแล้ว</p>"; } public function setBook(\$name, \$price) { self::\$name = \$name; self::\$price = \$price; self::echobook(); } public function getBook() { return array(self::\$name, self::\$price); } } \$book = new Book(); \$book->setBook("Full Stack Programmer", 499); \$arr = \$book->getBook(); echo "<p>หนังสือ " . \$arr[0] . " ราคา " . \$arr[1] . " บาท</p>"; ?></pre>
Output
เพิ่มหนังสือ Full Stack Programmer เรียบร้อยแล้ว
หนังสือ Full Stack Programmer ราคา 499 บาท

การเข้าถึงตัวแปร และฟังก์ชันแบบ static จากคลาสลูก จะใช้คีย์เวิร์ด parent ดังตัวอย่างนี้

PHP Code
<pre><?php class Item { public static \$name; public static \$price; public static function getItem() { return array(self::\$name, self::\$price); } } class Book extends Item { public function __construct(\$name, \$price) { parent::\$name = \$name; parent::\$price = \$price; } public function showBook() { \$arr = parent::getItem(); echo "<p>หนังสือ " . \$arr[0] . " ราคา " . \$arr[1] . " บาท</p>"; } } \$b = new Book("Basic CSS", 199); \$arr = \$b->showBook(); ?></pre>
Output
หนังสือ Basic CSS ราคา 199 บาท

Final Methods

Final Methods คือฟังก์ชันในคลาสแม่ ที่ป้องกันไม่ให้คลาสลูกสามารถ Override ได้ โดยจะใช้คีย์เวิร์ด final นำหน้าฟังก์ชันที่ต้องการประกาศเป็น Final Methods ดังตัวอย่างนี้

PHP Code
<pre><?php class Shape { final function circle_area(\$rad) { return pi() * \$rad * \$rad; } } class Circle extends Shape { public function circle_area(\$r) { return pi() * \$r * \$r; } } ?></pre>

จากตัวอย่างนี้ จะเกิด Fatal error: Cannot override final method Shape::circle_area()

Namespace

Namespaces คือสัญลักษณ์ที่ใช้ในการจัดเก็บกลุ่มของคลาสและเมธอดให้เป็นกลุ่มเดียวกัน แนวคิดของ namespaces ในการเขียนโปรแกรมนั้นเหมือนกับไฟล์ระบบในคอมพิวเตอร์ namespaces จะเป็นเหมือนโฟลเดอร์สำหรับแยกข้อมูลออกเป็นกลุ่มหรือคนละประเภท และข้อมูลในโฟลเดอร์นั้นเปรียบได้กับขอบเขตใน namespaces

Namespaces ยังช่วยในการแก้ไขปัญหา ชื่อคลาสซ้ำซ้อน โดยปกติในการเขียนโปรแกรม เราจะไม่สามารถสร้างคลาสที่มีชื่อเดียวกันได้ ตัวอย่างเช่นโปรแกรมเมอร์ 2 คน ต้องการเอาโค้ดมารวมกัน แต่ชื่อคลาสที่ทั้ง 2 คนใช้เป็นชื่อเดียวกัน ทำให้ไม่สามารถเอาโค้ดมารวมกันได้ กรณีนี้สามารถใช้ Namespaces ช่วยแก้ปัญหาได้ โดยการกำหนด Namespaces ของแต่ละคนแยกกัน

การสร้างและใช้งาน Namespaces

การสร้าง Namespaces จะใช้คีย์เวิร์ด namespace แล้วตามด้วยชื่อ Namespace ดังตัวอย่างต่อไปนี้

PHP Code

```
<?php
//File: namespace.php

namespace NameSpace1;

class Person {
    public $fname;
    public $lname;

    function __construct($firstName, $lastName) {
        $this->fname = $firstName;
        $this->lname = $lastName;
    }

    public function getName() {
        return $this->fname . " " . $this->lname;
    }
}
?>
```


ตัวอย่างการใช้งาน Namespace

PHP Code
<pre><?php include_once("namespace.php"); use Namespace1\Person; \$person1 = new Person("นพกร", "ไชยลังกา"); echo \$person1->getName(); ?></pre>
Output
นพกร ไชยลังกา

ตัวอย่างการสร้าง Multiple Namespaces

PHP Code
<pre><?php //File: namespace.php namespace Namespace1; class Person { public \$fname; public \$lname; function __construct(\$firstName, \$lastName) { \$this->fname = \$firstName; \$this->lname = \$lastName; } public function getName() { return \$this->fname . " " . \$this->lname; } }</pre>

```
namespace NameSpace2;

class Person {
    public $fname;
    public $lname;
    public $nname;

    function __construct($firstName, $lastName, $nickName) {
        $this->fname = $firstName;
        $this->lname = $lastName;
        $this->nname = $nickName;
    }

    public function getName() {
        return $this->fname . " " . $this->lname . " (" . $this->nname . ")";
    }
}

?>
```

ตัวอย่างการใช้ Multiple Namespaces

PHP Code

```
<?php
include_once("namespace.php");

$person1 = new NameSpace1\Person("นพกร","ไชยลังกา");
$person2 = new NameSpace2\Person("นพกร","ไชยลังกา","บอสส์");
echo $person1->getName();
echo "<br>";
echo $person2->getName();
?>
```

Output

นพกร ไชยลังกา
นพกร ไชยลังกา (บอสส์)

PHP Code

```
<?php
$servername = "localhost";
$username = "root";
$password = "xxxxx";
$dbname = "xxxxx";

$conn = new mysqli($servername, $username, $password, $dbname);

if ($conn->connect_error) {
    die("การเชื่อมต่อล้มเหลว: " . $conn->connect_error);
}
echo "เชื่อมต่อสำเร็จ";

$sql = "SELECT id FROM account";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    while ($row = $result->fetch_assoc()) {
        echo "id: " . $row["id"] . "<br>";
    }
} else {
    echo "ไม่มีข้อมูล";
}

$conn->close();
```

ใส่ชื่อ host,username,password,db เพื่อ เชื่อมต่อกับ db
หลังจากเชื่อมต่อได้แล้วจะโชว์ข้อมูล table แค่ id