Function

Web Programming (PHP)

Contents

ฟังก์ชันคืออะไร สำคัญอย่างไร

- การเขียนคำสั่งย่อยๆ ไว้ในส่วนหนึ่งของคำสั่ง เพื่อให้ทำงานตามคำสั่งนั้นๆ ให้เสร็จ แล้วสามารถนำไปใช้ทำงานร่วมกับคำสั่งหรือฟังก์ชันอื่นๆ ภายในโปรแกรมนั้นได้
- บางโปรแกรมมีความจำเป็นต้องทำงานในลักษณะที่ซ้ำๆ กันอยู่ในแต่ละส่วนของโปรแกรม
- ชุดคำสั่งที่ซ้ำๆ กันนั้น ทำให้ต้องเสียเวลาในการพิมพ์คำสั่งทุกครั้งที่จะสั่งทำงาน มีโอกาสที่จะทำให้เกิดข้อผิดพลาดขึ้น
- สามารถแยกชุดคำสั่งที่ใช้บ่อยๆ หรือซ้ำๆ กันนี้ออกมาและสร้างขึ้นเองเป็นฟังก์ชั่น (user-defined functions) ทำให้ไม่จำเป็นต้องเขียนชุดคำสั่งนั้นซ้ำๆ กัน

ฟังก์ชันคืออะไร สำคัญอย่างไร

- ฟังก์ชันใน php แบ่งออกเป็น 2 ประเภทคือ
  - ฟังก์ชันที่โปรแกรมเมอร์สร้างขึ้นเอง (user defined function) เพื่อต้องการใช้งานตามความต้องการของตนเอง
  - ฟังก์ชันที่ php สร้างขึ้นมาให้สำหรับการทำงานทั่วไป (built-in function) ซึ่งเป็นฟังก์ชันที่มีอยู่แล้ว เช่น echo, chr, substr

ส่วนประกอบของฟังก์ชัน

- ***การประกาศฟังก์ชัน*** ประกอบด้วย
  - ชื่อฟังก์ชัน เพื่อใช้สำหรับเวลาที่ต้องการเรียกใช้ฟังก์ชัน
  - Parameter หรือตัวแปรส่ง เป็นตัวแปรที่ฟังก์ชันสร้างไว้เพื่อรับค่าที่จะส่งเข้ามาให้ฟังก์ชัน
  - คำสั่งที่สั่งให้ฟังก์ชันทำงาน
  - การคืนค่าของฟังก์ชัน (มีหรือไม่มีก็ได้)
- ***การเรียกใช้ฟังก์ชัน*** ประกอบด้วย
  - Argument หรือตัวแปรรับ เป็นตัวแปรที่ต้องการส่งค่าให้กับฟังก์ชัน

การสร้างและการเรียกใช้ฟังก์ชัน

- ***ชื่อฟังก์ชัน*** มีหลักการในการตั้งชื่อดังนี้
  - ควรจะสัมพันธ์กับการทำงานของฟังก์ชัน (หรือจะไม่เกี่ยวข้องก็ได้)
  - ชื่อฟังก์ชัน ต้องไม่ซ้ำกัน
  - การตั้งชื่อคล้ายกับตัวแปรคือ ตั้งชื่อเป็นตัวอักษร ตัวเลข เส้นขีดล่าง (_) ได้ แต่ห้ามขึ้นต้นด้วยตัวเลข

การสร้างและการเรียกใช้ฟังก์ชัน

```
function ชื่อฟังก์ชัน() {
        คำสั่งในฟังก์ชัน ;
}
```

ตัวอย่าง การสร้างฟังก์ชันชื่อ say_yes (แบบไม่มีการกำหนด parameter)

```
function say_yes() {
   echo “Yes, sir.....”;
}
```

เวลาเรียกใช้งานฟังก์ชัน ให้เขียนว่า

say_yes();

Web browser แสดงผลเป็น Yes, sir.....

การสร้างและการเรียกใช้ฟังก์ชัน

- ***Parameter (ตัวแปรส่ง)*** เป็นตัวแปรที่ฟังก์ชันสร้างขึ้นเพื่อรับค่าที่จะส่งเข้ามาให้กับฟังก์ชัน แล้วนำค่านั้นไปประมวลผลในฟังก์ชัน

```
function ชื่อฟังก์ชัน($parameter1, $parameter2,... , $parametern) {
    คำสั่งที่ต้องการให้ทำงาน ;
    การคืนค่าของฟังก์ชัน (มีหรือไม่ก็ได้) ;
}
```

- ***Argument (ตัวแปรรับ)*** เป็นตัวแปรที่ต้องการ**ส่งค่ามาให้กับฟังก์ชัน** ซึ่งจะเรียกใช้งานฟังก์ชัน

```
ชื่อฟังก์ชัน(Argument1, Argument2,... , Argumentn) ;
```

การสร้างและการเรียกใช้ฟังก์ชัน

```
function ชื่อฟังก์ชัน($parameter1, $parameter2,...) {
    คำสั่งในฟังก์ชัน ;
}
```

ฟังก์ชันจะรับ parameter ซึ่งเป็นตัวแปรที่เก็บค่าที่ส่งมาให้กับฟังก์ชัน

ตัวอย่าง

```
function italic($str) {
   echo “<i>$str</i>”;
}
```

เวลาเรียกใช้งานฟังก์ชัน ให้เขียนว่า

italic(“เอียง”);

Web browser แสดงผลเป็น *เอียง*

```
function summ($a,$b) {
    return $a+$b;
}
```

เวลาเรียกใช้งานฟังก์ชัน ให้เขียนว่า

```
echo summ(1,9);
```

Web browser แสดงผลเป็น 10 (เป็นผลบวกของ 1 และ 9)

การสร้างและการเรียกใช้ฟังก์ชัน

ส่งผ่านด้วยค่าตัวแปร

การเปลี่ยนแปลงค่าในฟังก์ชันไม่มีผลต่อค่าที่อยู่นอกฟังก์ชัน

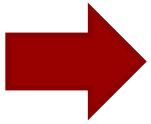
ส่งผ่านด้วยค่าการอ้างอิง

การเปลี่ยนแปลงของค่าตัวแปรในฟังก์ชัน ทำให้ค่าตัวแปรนอกฟังก์ชันเปลี่ยนแปลงไปด้วย

การส่งผ่านค่าตัวแปรให้ฟังก์ชัน

- *การส่งผ่านด้วยค่าตัวแปร (Pass by Value)*
  - เป็นการส่งผ่านค่า Argument
  - เป็นการคัดลอกค่า แล้วส่งค่าที่คัดลอกไปยังตัวแปรภายในฟังก์ชัน แล้วทำงาน
  - เป็นค่าตัวแปร คนละค่า ของค่าตัวแปรภายนอกฟังก์ชัน
  - ในฟังก์ชันมีการเปลี่ยนแปลงค่าที่ผ่านไปให้ ไม่มีผล ต่อค่าที่อยู่ภายนอกฟังก์ชัน

```
<php
$a=100;
echo “เริ่มต้น a มีค่าเท่ากับ “.$a.”<br />”;
AddS($a);
echo “ภายหลัง a มีค่าเท่ากับ “.$a.”<br />”;
function AddS($a) {
$a+=20;
echo “ภายในฟังก์ชัน AddS a มีค่าเท่ากับ”. $a;
return;
}
?>
```

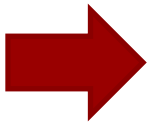


```
เริ่มต้น a มีค่าเท่ากับ 100
ภายในฟังก์ชัน AddS a มีค่าเท่ากับ 120
ภายหลัง a มีค่าเท่ากับ 100
```

การส่งผ่านค่าการอ้างอิง

- *การส่งผ่านด้วยค่าการอ้างอิง (Pass by Reference)*
  - เป็นการส่งผ่านค่า Argument
  - เป็นการคัดลอกค่าส่งต่อไปที่ตัวแปรภายในฟังก์ชัน แล้วทำงาน
  - โดยค่าตัวแปร Argument เดิม เปลี่ยนแปลงไปด้วย
  - การส่งค่าแบบนี้ ต้องใส่เครื่องหมาย & ไว้หน้าตัวแปร Argument ที่ส่งให้ฟังก์ชัน

```
<php
$a=100;
echo “เริ่มต้น a มีค่าเท่ากับ “.$a.”<br />”;
AddS($a);
echo “ภายหลัง a มีค่าเท่ากับ “.$a.”<br />”;
function AddS(&$a) {
$a+=20;
echo “ภายในฟังก์ชัน AddS a มีค่าเท่ากับ”. $a;
return;
}
?>
```

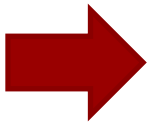


```
เริ่มต้น a มีค่าเท่ากับ 100
ภายในฟังก์ชัน AddS a มีค่าเท่ากับ 120
ภายหลัง a มีค่าเท่ากับ 120
```

การส่งค่าแบบการกำหนดค่าปริยาย

- *การส่งค่าแบบการกำหนดค่าปริยาย (Default Argument Values)*
  - Default = ค่าเริ่มต้น หรือค่าปริยาย
  - การเรียกใช้ฟังก์ชันโดยไม่ส่งค่า Argument ใดๆ เลยไปที่ฟังก์ชัน ฟังก์ชันจะทำงานโดยเลือกเอาค่าปริยายที่มีการกำหนดในส่วนของ Argument
  - หากมีการการเรียกใช้ฟังก์ชันโดยส่ง Argument มาที่ฟังก์ชัน ฟังก์ชันจะทำงานโดยใช้ค่าที่ถูกส่งมาแทนค่าปริยายที่มีการกำหนดขึ้น

```
<php
function eating ($x="sandwich") {
      return "I am eating $x.<br>";
}
echo eating();
$abc="hamburger";
echo eating ($abc);
echo eating();
?>
```



I am eating sandwich.

I am eating hamburger.

I am eating sandwich

การคืนค่าจากฟังก์ชัน

ฟังก์ชันที่ไม่มีการคืนค่า

การเปลี่ยนแปลงค่าในฟังก์ชันไม่มีผลต่อค่าที่อยู่นอกฟังก์ชัน

ฟังก์ชันที่มีการคืนค่า

การเปลี่ยนแปลงของค่าตัวแปรในฟังก์ชัน ทำให้ค่าตัวแปรนอกฟังก์ชันเปลี่ยนแปลงไปด้วย

การคืนค่าจากฟังก์ชัน

- *ฟังก์ชันที่ไม่มีการคืนค่า*
  - ใช้คำสั่ง return ที่จุดที่ต้องการให้ หยุดการทำงาน เพื่อจบการทำงานภายในฟังก์ชันออกไปที่จุดเรียกใช้ฟังก์ชัน เพื่อทำงานต่อไป

การคืนค่าจากฟังก์ชัน

ฟังก์ชันที่ไม่มีการคืนค่า

```
<?php
$num = $_POST['no'];
OddOrEven($num);
function OddOrEven($nx) {
    if ($nx%2 == 0) {
        echo "<strong><font color=blue> เป็นเลขคู่ </font></strong>";
        return;
    }
    else {
        echo "<strong><font color=red> เป็นเลขคี่ </font></strong>";
        return;
    }
}
?>
```

ค่าที่รับจากจาก ‘no’ เก็บไว้ที่ตัวแปรชื่อ $num

เมื่อเรียกใช้งานฟังก์ชัน OddOrEven

ถ้า $num เป็นเลขคู่ (เช่น 2) จะแสดงผลเป็น **เป็นเลขคู่**

ถ้า $num เป็นเลขคี่ (เช่น 3) จะแสดงผลเป็น **เป็นเลขคี่**

การคืนค่าจากฟังก์ชัน

- *ฟังก์ชันที่มีการคืนค่า*
  - ใช้คำสั่ง return ตามด้วยค่าที่ต้องการส่งกลับ
  - โดยค่าที่ส่งกลับออกไป จะเป็นตัวแปร ค่าคงที่ หรือฟังก์ชันก็ได้

การคืนค่าจากฟังก์ชัน

- *ฟังก์ชันที่มีการคืนค่า*

```
<?php
    $price = $_POST['price'];
    $cus = $_POST['customer'];
    $disprice = CheckPrice($price, $cus);
    echo "ราคาลดแล้วเหลือ = " . $disprice . " บาท";

    function CheckPrice($pr, $cs)     {
            switch ($cs) {
                case 1:
                    return $pr;                         // ลูกค้าทั่วไปไม่ได้ลด
                case 2:
                    return $pr * 0.95; // ลูกค้าสมาชิกลดให้ 5%
                case 3:
                    return $pr * 0.90; // ลูกค้า VIP ลดให้ 10%
            }
    }
?>
```

ค่าที่รับจากจาก ‘price’ เก็บไว้ที่ตัวแปรชื่อ $price
เป็นค่าของราคาสินค้าที่ลูกค้าซื้อ
ค่าที่รับจากจาก ‘customer’ เก็บไว้ที่ตัวแปรชื่อ $cus
เป็นประเภทลูกค้าที่มีการลดราคาตามประเภทลูกค้า
เรียกใช้งานฟังก์ชันชื่อ CheckPrice