Written by Thapanapong Rukkanchanunt

PHP Loop

Loop คืออะไร

- เมื่อเราต้องเขียนเว็บที่มีรายละเอียดมากขึ้น เราจะพบกับปัญหาการเขียนโค้ดที่มีลักษณะคล้ายกันจำนวนมาก เช่นถ้าเราต้องการสร้างตารางขนาด 7 x 7 เราต้องพิมพ์ <td></td> ถึง 49 ครั้ง นอกจากนี้ถ้าเราต้องการเพิ่มหรือลดขนาดตารางเราต้องแก้ไขหลายจุด
- Loop เป็นโครงสร้างที่ทำคำสั่งที่มีลักษณะคล้ายกันตามจำนวนครั้งที่เรากำหนดให้

Code without Loop

- สร้าง List ตัวเลข

```
echo “<ul>”;
echo “<li>1</li>”;
echo “<li>2</li>”;
echo “<li>3</li>”;
echo “<li>4</li>”;
echo “</ul>”;
```

- 1
- 2
- 3
- 4

Code with Loop

- สร้าง List ตัวเลข

```
echo “<ul>”;
for ($i=1;$i<=4;$i++) {
        echo “<li>$i</li>”;
}
echo “</ul>”;
```

- 1
- 2
- 3
- 4

โครงสร้าง Loop

- ในภาษา PHP โครงสร้าง Loop จะแบ่งออกเป็น 4 ประเภทได้แก่
  - For Loop
  - Foreach Loop
  - While Loop
  - Do While Loop

For Loop

- พิจารณาโค้ดต่อไปนี้ ซึ่งพิมพ์ตัวเลขตั้งแต่ 0 ถึง 9

```
1       for ($i = 0;$i < 10; $i++) {
2               echo $i;
3       }
```

- โครงสร้าง Loop จะเริ่มต้นจากคำว่า for แล้วตามด้วยวงเล็บ
- ภายในวงเล็บ เป็นคำสั่งสามคำสั่งคั่นกันด้วยเครื่องหมาย ;

For Loop (2)

```
1       for ($i = 0;$i < 10; $i++) {
2               echo $i;
3       }
```

- คำสั่งแรกบอกว่าจะเริ่ม Loop จากตรงไหน
- คำสั่งที่สองบอกว่าจะจบ Loop ณ จุดใด
- คำสั่งที่สามบอกว่าจะทำอะไรเมื่อจบหนึ่งรอบของ Loop

For Loop (3)

```
1       for ($i = 0;$i < 10; $i++) {
2               echo $i;
3       }
```

- Loop จะเริ่มต้นโดยกำหนดให้ตัวแปร $i มีค่า 0
- Loop จะทำงานรอบต่อไปก็ต่อเมื่อเงื่อนไข $i < 10 เป็นจริง
- เมื่อจบ Loop หนึ่งรอบแล้วตัวแปร $i จะมีค่าเพิ่มขึ้น 1
  - $i++ คือการเขียน $i = $i + 1; แบบย่อ

For Loop (4)

```
1       for ($i = 0;$i < 10; $i++) {
2               echo $i;
3       }
```

- หลังจากสามคำสั่งในวงเล็บ คำสั่งที่อยู่ในวงเล็บ { } จะเป็นคำสั่งที่ทำในหนึ่งรอบของ Loop ในตัวอย่างคำสั่งที่จะทำในหนึ่งรอบของ Loop คือ echo $i;

Loop in Action

```
1       for ($i = 0;$i < 10; $i++) {
2               echo $i;
3       }
```

- Loop เริ่มต้น โดยเรากำหนดค่า $i เป็น 0

Loop in Action (2)

```
1       for ($i = 0;$i < 10; $i++) {
2               echo $i;
3       }
```

- ตรวจสอบว่าเงื่อนไข $i < 10 เป็นจริงหรือไม่
- เนื่องจากตอนนี้ $i มีค่าเป็น 0 เงื่อนไขจึงเป็นจริง

Loop in Action (3)

```
1       for ($i = 0;$i < 10; $i++) {
2               echo $i;
3       }
```

- เมื่อเงื่อนไขเป็นจริง โปรแกรมจะทำคำสั่งในวงเล็บ {}
- คำสั่ง echo $i; พิมพ์ค่า $i ออกทางหน้าจอ

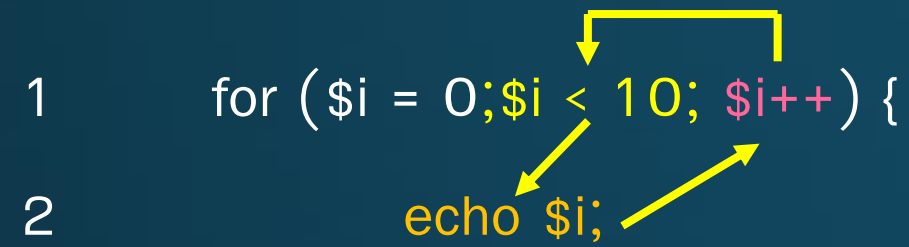
Loop in Action (4)

```
1       for ($i = 0;$i < 10; $i++) {
2               echo $i;
3       }
```

- เมื่อทำคำสั่งในเว็บลง { } เสร็จสิ้นหมดแล้ว เราจะเพิ่มค่า $i มาหนึ่ง
- การกระทำทั้งหมดนี้คือหนึ่งรอบใน Loop

0

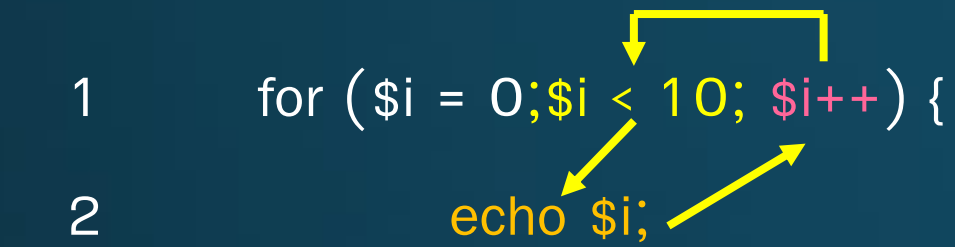
Loop in Action (5)



- ในรอบถัด ๆ ไปเราจะตามขั้นตอนเดิมตั้งแต่การตรวจสอบเงื่อนไข

01

Loop in Action (6)



```
1       for ($i = 0;$i < 10; $i++) {
2               echo $i;
3       }
```

$i	10

- เมื่อ $i มีค่า 10 เงื่อนไขกลายเป็นเท็จ Loop จึงจบตรงนี้

```
0123456789
```

Foreach Loop

- หลังจากที่เรารู้โครงสร้าง For Loop แล้ว เราจะพบว่าเมื่อเรารู้จำนวนครั้งที่ต้องทำ เราสามารถใช้ความรู้นั้นมาสร้างเงื่อนไขให้ Loop ทำตามจำนวนครั้งที่เราต้องการได้
- Foreach Loop เป็น Loop ที่มีจำนวนครั้งในการทำซ้ำที่แน่นอน ซึ่งจะนำมาใช้กับการทำซ้ำของค่าที่อยู่ใน Array
- เราสามารถมอง Foreach Loop ว่าเป็นการดึงค่าจาก Array ทีละตัวออกมาคำนวณหรือแสดงผลออกทางหน้าจอ

Foreach Loop

- ตัวอย่างข้างล่างเป็นแสดงค่าทุกค่าใน Array ในรูปแบบของ List

```
echo “<ul>”;
$fruits = array(“Apple”, “Banana”, “Orange”, “Grape”, “Pineapple”);
foreach ($fruits as $fruit) {
        echo “<li>$fruit</li>”;
}
echo “</ul>”;
```

- Apple
- Banana
- Orange
- Grape
- Pineapple

Foreach Loop Syntax

```
foreach ($fruits as $fruit) {
        echo “<li>$fruit</li>”;
}
```

- Foreach Loop จะเริ่มต้นจากคำว่า foreach แล้วตามด้วยวงเล็บ
- ภายในวงเล็บจะเป็นการบอก PHP ว่าจะให้ Loop บนตัวแปร Array ใด และจะให้เก็บค่านั้นไว้ชั่วคราวที่ตัวแปรใด ในรูปแบบ [ตัวแปร Array] as [ตัวแปรเก็บค่าชั่วคราว]
- ในตัวอย่าง PHP จะวน Loop บนตัวแปร $fruits และใช้ตัวแปร $fruit เก็บค่าชั่วคราว

Foreach Loop Syntax (2)

```
foreach ($fruits as $fruit) {
        echo “<li>$fruit</li>”;
}
```

- ในแต่ละรอบของ Loop โปรแกรมจะทำตามคำสั่งที่อยู่ภายในวงเล็บ { } โดยค่าของตัวแปร $fruit จะมีการเปลี่ยนแปลงทุกรอบ

Foreach Loop in Action

```
$fruits = array("Apple", "Banana", "Orange", "Grape", "Pineapple");
foreach ($fruits as $fruit) {
        echo "<li>$fruit</li>";
}
```

- ในรอบแรกของ Loop ค่าของตัวแปร $fruit จะเป็นคำว่า Apple

- Apple

Foreach Loop in Action

```
$fruits = array("Apple", "Banana", "Orange", "Grape", "Pineapple");
foreach ($fruits as $fruit) {
        echo "<li>$fruit</li>";
}
```

- Apple
- Banana

- ในรอบที่สองของ Loop ค่าของตัวแปร $fruit จะเป็นคำว่า Banana
- ดังนั้น foreach นี้จะทำซ้ำทั้งหมด 5 รอบ ซึ่งเท่ากับจำนวนสิ่งของใน Array

For และ Foreach

- เราจะพบว่าโครงสร้างทั้งสองคล้ายกันตรงที่จำนวนรอบในการทำซ้ำแน่นอนตั้งแต่ตอนเขียนโปรแกรม
- For Loop ใช้เงื่อนไขเป็นตัวกำหนดจำนวนรอบ
- Foreach Loop ใช้จำนวนค่าใน Array เป็นตัวกำหนดรอบ
- ถ้าหากเราไม่ทราบจำนวนรอบที่แน่นอนในบางสถานการณ์ (อาจจะเกี่ยวข้องกับดวง) For และ Foreach จึงไม่เพียงพอต่อความต้องการ

While Loop

- While Loop เป็นโครงสร้างที่ทำคำสั่งซ้ำ ๆ โดยมีเงื่อนไขที่จะตรวจสอบก่อนเริ่มแต่ละรอบ ถ้าเงื่อนไขเป็นจริงถึงจะทำตามคำสั่งที่ให้ไว้
- สิ่งที่ทำให้แตกต่างจาก For และ Foreach คืออาจจะไม่มีการเปลี่ยนแปลงค่าที่แน่นอน ใน For ส่วนที่สามในวงเล็บ ( ) จะระบุว่าตัวแปรจะมีการเปลี่ยนอย่างไร ส่วนใน Foreach ค่าของตัวแปรชั่วคราวจะมีการเปลี่ยนแปลงที่ทำนายได้

While Loop Syntax

```
$sum = 0;
while ($sum < 50) {
        $sum = $sum + rand(1,10);
        echo “<p>$sum</p>”;
}
```

- โครงสร้าง While จะเริ่มจากคำว่า while แล้วตามด้วยวงเล็บ ( )
- ในวงเล็บจะเป็นเงื่อนไขที่จะถูกตรวจสอบทุกครั้งก่อนจะเริ่มแต่ละรอบ

While Loop Syntax

```
$sum = 0;
while ($sum < 50) {
        $sum = $sum + rand(1,10);
        echo “<p>$sum</p>”;
}
```

- หลังจากตรวจสอบเงื่อนไขแล้ว PHP จะทำคำสั่งในวงเล็บ { }

While Loop Example

```
$sum = 0;
while ($sum < 50) {
        $sum = $sum + rand(1,10);
        echo “<p>$sum</p>”;
}
```

- จากตัวอย่างข้างต้น PHP จะตรวจสอบเงื่อนไข $sum < 50 ในแต่ละรอบ เริ่มแรก $sum มีค่าเป็น 0 ในแต่ละรอบใน Loop ค่าของ $sum จะเพิ่มขึ้นตามเลขสุ่มจาก 1 ถึง 10 ดังนั้นเราจะไม่ทราบเลยว่าต้องบวกกี่ครั้งถึงจะมีค่าอย่างน้อย 50

ข้อควรระวังใน While Loop

- เนื่องจาก While Loop จะมีการตรวจสอบเงื่อนไขทุกครั้ง เราจะต้องทำให้แน่ใจว่า ณ จุดหนึ่ง เงื่อนไขดังกล่าวต้องเป็นเท็จเพื่อออกจาก Loop ไม่เช่นนั้นจะเกิดเหตุการณ์ Infinite Loop
- เหตุการณ์ Infinite Loop ในภาษาต่าง ๆ มักจะทำให้โปรแกรมค้างและปิดตัวเองลง ดังนั้นอาการโปรแกรมค้าง แปลว่าโปรแกรมไม่หลุดออกจาก Loop
- ในภาษา PHP จะมีเวลาจำกัดไว้ ถ้าประมวลเกิน 2 นาทีจะหยุดการกระทำทั้งหมด ฉะนั้นขณะที่เราทำงาน ถ้าต้องรอนาน (ไอคอนโหลดหมุนนาน ๆ) ควรคาดเดาไว้ว่าเกิด Infinite Loop แน่นอน

Do While Loop

- Loop ประเภทสุดท้ายคือ Do While ซึ่งคล้ายกับ While แต่แตกต่างกันที่ Do While จะตรวจสอบเงื่อนไขหลังจากทำคำสั่งในวงเล็บ { } เสร็จสิ้นแล้ว
- ดังนั้นคำสั่งในวงเล็บ { } จะถูกคำนวณอย่างน้อยหนึ่งครั้งเสมอใน Do While

Do While Loop Syntax

```
$sum = 0;
do {
        $sum = $sum + rand(1,10);
        echo “<p>$sum</p>”;
} while ($sum < 50);
```

- โครงสร้าง Do While เริ่มจากคำว่า do แล้วตามด้วยคำสั่งที่ต้องการทำซ้ำในวงเล็บ { }

Do While Loop Syntax (2)

```
$sum = 0;
do {
        $sum = $sum + rand(1,10);
        echo “<p>$sum</p>”;
} while ($sum < 50);
```

- หลังจากวงเล็บ จะเป็นคำว่า while แล้วตามด้วยเงื่อนไขในวงเล็บ ( )
- สังเกตว่าหลังวงเล็บ ( ) จะต้องมีเครื่องหมาย ; เสมอ

Do While Loop Example

```
$sum = 0;
do {
        $sum = $sum + rand(1,10);
        echo “<p>$sum</p>”;
} while ($sum < 50);
```

- ตัวอย่างข้างต้นจะได้ผลลัพธ์ในลักษณะเดียวกับตัวอย่าง While