PHPの文法おさらい

この章では簡単なプログラムを動かしてみながらPHPの文法をおさらいします。この章に出てくるソースコードはhtmlフォルダにindex.phpファイルを作って、そこにかかれているものとします。

HTMLを拡張する言語

index.phpに以下のように書いて、ブラウザでlocalhos:8000/index.phpにアクセスしてみてください。

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>サンプルページ</title>
</head>
<body>
    <h1>サンプルです</h1>
    <p>サンプル文章</p>
</body>
</html>
```

見てわかるとおり、これはhtmlです。<!DOCTYPE html>でhtmlの文書であることを示し、<html>タグで囲まれた場所がhtmlです。そして<head>タグで囲まれた場所で、タイトルや文字コードを示し、<body>タグで表示される内容が始まり、文章が書かれています。

ここに最初のPHPコードを書きましょう。phpのコードは<?php ではじまり、?>で終わります。bodyのところを次のように書き換えてみましょう。

```
<body>
    <h1>サンプルです</h1>
    <p><?php echo date("Y年m月d日");?></p>
</body>
```

現在の日付が表示されたと思います。echoは文字列を出力してくれる機能です。厳密には関数ではありませんが、関数だと思ってもらっても構いません。そしてdateは指定されたフォーマットで日時の文字列を返す関数です。

そして、変数を使うこともできます。phpでは先頭に$を付けることで変数であることを示します。

```
<?php $a = 'サンプル文章';?>
<body>
    <h1>サンプルです</h1>
    <p><?php echo $a;?></p>
</body>
```

このようにプログラミングをしてHTMLを組み立てるのではなく、既にあるHTMLに埋め込むようにして書けることから、PHPはHTML埋め込み言語などと言われています。

型のおさらい

PHPは動的型付け言語であり、実行時に動的に型を付けるので、ある程度よしなにやってくれます。しかし、型に対して意識が無いと、文字列の数字("120"とか)に対して足し算をしようとしてエラーになったりします。なので、PHPにはどのような型があるのかを理解しましょう。

数値

PHPには整数(integer)と浮動小数点(float もしくは double)という二種類の型があります。

```
$var1 = 100;
echo gettype($var1);
// integer

$var2 = 1.2345;
echo gettype($var2);
// double
```

割り算をしたときなどはうまく割り算をしてくれますが、浮動小数点が我々が普段使う小数とは実体が異なることに注意が必要です。以下のコードではfalseが出力されます。

```
$var1 = 0.1;
$var2 = 0.2;
var_export($var1+$var2==0.3)
```

浮動小数点は二進数で表現されるので、十進数の小数と同じように計算してもわずかに誤差が生じます。これは我々が1/3を小数で正しく表せないのと同じです。なので、小数を比較するときは注意しましょう。

数値型は以下のような演算ができます

```
echo 1 + 2;  // 3
echo 4 - 3;  // 1
echo 5 * 6;  // 30
echo 10 / 2; // 5
echo 1 / 2;  // 0.5 割り切れない時は小数になります
echo 10 % 3; // 1 割った時の余りが出てきます
echo 3 ** 3; // 27 3の3乗です。累乗と言います
```

またPHPには加算子と減算子と呼ばれるものがあります。すこし挙動がややこしいので注意してください。

```
$num = 10;
echo $num++; // 100 (100と表示された後、加算されて101になります)
```

```
echo $num;   // 101
echo $num--; // 101 (101と表示された後、減算されて100になります)
echo $num;   // 100
echo ++$num; // 101 (先に加算されてから表示されます)
echo $num;   // 101
echo --$num; // 100
```

文字列

PHPではstringという型で文字列を扱います。

```
$var1 = 'Hello!';

echo gettype($var1);
// string
```

シングルクォーテーションで囲ってもダブルクォーテーションで囲っても文字列が出力されますが、ダブルクォーテーションの時は変数を埋め込むことができます。

```
$var1 = 'World';

echo "Hello!{$var1}";
// Hello!World
```

文字列はピリオド（.）で結合できます

```
$var1 = 'Hello';
$var2 = 'World';

echo $var1 . ' ' . $var2;
// Hello World
```

PHPが遅いのに気にしなくていい理由

PHPは動的型付け言語であり、インタープリタ言語なので遅い言語です。GCがなく、静的型付けでコンパイルもするc言語と比較すると100倍近く遅いと言われています。また、Laravelなどの大きなフレームワークを使うとさらに5倍くらい遅くなります。この時あなたは「え？！100倍も遅かったらサーバー代が100倍になるじゃん！大丈夫なの？！」と思うでしょう。

しかしPHPの動作速度の遅さが支障になるケースはあまりありません。PHPがユーザーから受け付けて処理するデータはサイズがしれています。一方で、データベースは10万件100万件もあるデータから必要なデータを検索して取り出す必要があります。なので多くの場合データベースをチューニングする方が先決で、PHPの動作速度がボトルネックになることはあまりないのです。

また、ネットワークの遅さもボトルネックになります。いまのCPUはとても高性能なので1GBのファイルを展開するのはすぐですが、1GBのファイルを転送するにはかなり時間がかかります。わざわざZipファイルにしてダウンロードさせるのはネットワークの遅さの方が致命的だからです。

よくPHPがほかの言語と比較して遅いことを指摘する人がいますが、このような現状を理解して指摘している人は少ない印象があります。仮に画像処理のようなPHPの遅さがボトルネックになる場合でも、そこだけc言語などで書き直すべきで、速さだけを理由にして中途半端に早い言語を使うべきではありません。

論理値

論理値はtrue/falseを取り扱う型です。booleanともいいます。

```
$var1 = true;
echo gettypr($var1);
// boolean
```

比較は二つの値を比較して論理値を返します。if文はこれを利用して実現します。

```
$var1 = 'a';
$var2 = 'a';

if ($var1 == $var2){
    echo '同じです';
}
```

演算子	説明
==	型のキャストをした後で等しいときにtrue
===	内容が同じでかつ型が同じ時 true
!=	型のキャストをした後で等しくない時true
!==	内容が違うか型が違うときにtrue
<	右側が大きければtrue
>	左側が大きければtrue
<=	右側が大きいか等しいときにtrue
>=	左側が大きいか等しいときにtrue
<=>	左側が大きいとき1,同じ時0,右側が大きいとき-1 (PHP7から)

==などは暗黙の型変換があるので注意が必要です。たとえば以下はtrueになります

```
"1" == "01" // 1 === 1 あつかいになり true
```

このように、意図しない比較になるので、できるだけ===を使うようにしましょう

配列

複数の値を取り扱うときは配列を使います。

```
$lamp = ['Linux', 'Apche', 'MySQL', 'PHP'];
echo gettype($lamp);
// array
```

PHPの配列はキーと値のペアで構成される連想配列です。これをvar_dumpで確認します。var_dumpはとても便利なので、覚えておきましょう。

```
$lamp = ['Linux', 'Apche', 'MySQL', 'PHP'];
var_dump($lamp);
```

```
array(4) {
    [0]=>
        string(5) "Linux"
    [1]=>
        string(5) "Apche"
    [2]=>
        string(5) "MySQL"
    [3]=>
        string(3) "PHP"
}
```

値だけ定義した場合、キーに整数が自動で割り当てられます。当然自分でキーを設定することもできます。キーは数値である必要はありません。

```
$lamp = ['Linux'=>10, 'Apche'=>20, 'MySQL'=>30, 'PHP'=>40];
var_dump($lamp);
```

```
array(4) {
    ["Linux"]=>
        int(10)
    ["Apche"]=>
        int(20)
    ["MySQL"]=>
        int(30)
    ["PHP"]=>
        int(40)
}
```

PHPはこのようにキーが整数の添字配列も連想配列であるという特徴があります。

配列はarray()か短縮構文([])で書けます。

```
$lamp = ['Linux', 'Apche', 'MySQL', 'PHP'];
$lamp2 = array('Linux', 'Apche', 'MySQL', 'PHP');
```

配列にデータを追加したい場合はarray_push()かブラケットを用います。

```
$lamp = ['Linux', 'Apche'];
array_push($lamp, 'MySQL');
$lamp[] = 'PHP';

var_dump($lamp);  // LinuxからPHPまでふくんだ配列になる
```

キーを指定して値を追加することもできます。

```
$lamp = ['Linux'=>10, 'Apche'=>20, 'MySQL'->30];
$lamp['PHP'] = 40;
var_dump($lamp);
```

配列に配列をくっつけることができます。array_merge()か+演算子でできます。

```
$lamp = ['Linux'=>10, 'Apche'=>20];
$lamp = array_merge($lamp, ['MySQL'->30, 'PHP'=>40]);
var_dump($lamp);

$lamp = ['Linux'=>10, 'Apche'=>20,'MySQL'=>30];
$lamp = $lamp + ['PHP'=>40]; // 要素が一個だけの配列もあります
var_dump($lamp);
```

要素の削除はunset()で行えます。

```
$lamp = ['Linux'=>10, 'Apche'=>20, 'MySQL'->30, 'PHP'=>40, 'Windows'=>50];
unset($lamp['Windows']);
var_dump($lamp);
```

制御構文と関数

プログラミングをするうえ重要な制御構文と関数です。

if文

条件を満たしたらブロック内を実行します。ifで条件を記述します。elseifで後ろにつなげていくことができ、最後まで条件を満たさなかったときの動作をelseで記述します。elseifやelseは使わない場合は省略しても構いません。

```
$count = 1;

if($count == 0){
    echo 'count は 0';
}elseif($count == 1){
    echo 'count は 1';
}elseif($count == 2){
    echo 'count は 2';
}else{
    echo '意図しない値です';
}
```

switch文

評価する変数が一つであるような場合はSwitchが便利です。ただし、前述した==のような緩い評価を行うので気を付けましょう。

```
$count = 1;

switch ($count){
    case 0:
        echo 'count は 0';
        break;
    case 1;
        echo 'count は 1';
        break;
    case 2:
        echo 'count は 2';
        break;
    default:
        echo '意図しない値です';
        break;
}
```

while文

条件がtrueであるあいだ繰り返し処理が実行されます。処理の途中で抜ける場合はbreak;をつかいます

```
$count = 0;
while(true){
    echo $count.'<br>';
```

```
    $cout++;
    if($count == 4){
        break;
    }
}
```

for文

for文は指定した回数だけ繰り返し処理が実行されます。最初にカウンター変数を初期化しと増減処理をまとめて記述します。

```
for($i = 0; $i <= 10; $i++){
    echo $i;
}
// 12345678910
```

foreach文

foreach文は配列から要素を取り出して処理を行っていきます。

```
$lamps = ['Linux', 'Apche', 'MySQL', 'PHP'];
foreach($lamps as $lamp){
    echo $lamp . '<br>';
}

$lamps2 = ['L'=>'Linux', 'A'=>'Apche','M'=> 'MySQL','P'=> 'PHP'];
foreach($lamps2 as $alphabet => $name){
    echo $alphabet . 'は' . $name . 'です。<br>';
}
```

関数

関数は引数の返り値をもち、何らかの処理を一つにまとめることができます。以下の処理では関数を呼び出したときに関数内で「自己紹介：」が表示され、返された「私の名前はPiffettです」をもとにechoが文字列を表示しています。

```
function introduce($name){
    echo '自己紹介:';
    return '私の名前は' . $name . 'です';
}

echo introduce('Piffett');
```

これまで紹介したarray_merge()などの関数は全て組み込み関数です。PHPは組み込み関数が非常に多いことが特徴です。また、PHPのバージョンが上がる際に非推奨になったり使用不可になる関数もあるため、公式

ドキュメントを読む癖を付けましょう。

http://php.net/manual/ja