

16 PHPの基礎知識

第15章では、PHPで「ようこそ!」の文字をブラウザに表示することができました。ここでは、PHPでプログラムを作るための基礎を学びます。

とはいうものの、PHPは奥が深く、関数だけとっても1000以上と膨大な数です。本書では「MySQLを利用する上でのPHPの知識」に絞り、MySQLを制御するのに最低限必要な「変数」「文字列」「関数」「比較演算子」「条件分岐」「繰り返し」「配列」を解説します。

変数

PHPの変数とは

まずは**変数**です。MySQLでも登場しましたが(→P.275)、変数は値を入れておく「箱」のようなものです。PHPがMySQLと連携する上で、変数は欠かすことのできない大切な役割を果たします。

さっそく使ってみましょう。まず、List16-01のスクリプトをP.344と同じ要領で入力してみてください。

LIST ■ 16-01 hensu.php

```
<?php
$a= "ようこそ!";
print $a;
?>
```

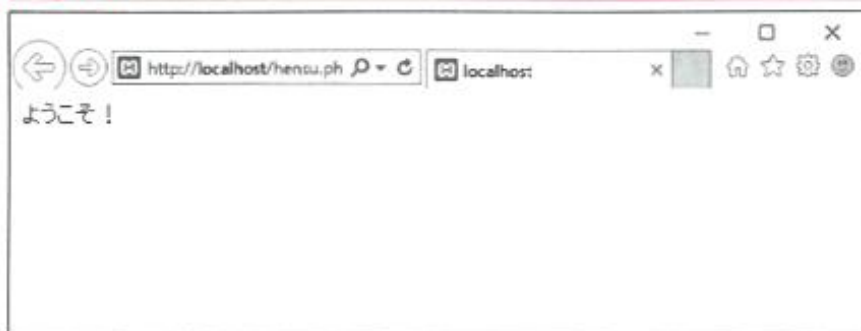
もう一度ミスがないのを確認したら、これを「hensu.php」というファイル名で、Webサーバーによって公開されるフォルダに保存してください。くどいようですが、保存時の文字エンコーディングをUTF-8にするのを忘れないようにしましょう。Webサーバーによって公開されるフォルダは異なりますが、本書のようにMAMPをインストールした場合、「C:\MAMP\htdocs」です(→P.344)。

そして、Apacheが動作していることを確認(→P.18)したら、Webブラウザのアドレスバーに次のように入力します。

```
http://localhost/hensu.php
```

実行すると、次のように表示されるはずです。

FIG 16-01 実行結果



もし表示されなかったら、P.345を見てチェックしてみましょう。

さて、スクリプトの内容を見ていきましょう。まず、PHPでは変数の名前の最初に「\$」を付けます。上の例の変数は「a」としたので、「\$a」になります。そして、次の構文がポイントです。

```
$a="ようこそ!";
```

プログラム言語が初めての方は、この式を見て不思議に思うかもしれません。これは、変数「\$a」に、「ようこそ!」という文字列を「代入」しています。

プログラムの世界では、「=」は、「=の右側の値を、左側に代入する」ということを意味します。つまり、これで「\$a」の中身が「ようこそ!」になったのです。

FIG 16-02 変数



ちなみに、MySQLの比較演算子で登場した「等しい」を示す「=」は、PHPの場合「==」と2つ続けて記述します(→P.368)。

前述のとおり、「print」は文字列を書き出す命令ですね。よって、次で「\$a」の値、すなわち「ようこそ!」の文字列が書き出されます。

```
print $a;
```

P.344の例を、変数を使って表しただけですね。このように「値を入れる箱」のようなものが「変数」なのです。また、ずっと入れっぱなしというわけではなく、出し入れ自由という便利な箱です。

変数名の規則

変数の名前の設定には、いろいろな規則があります。PHPの場合には、次のような規則があります。

- 最初に「\$」を付ける
- 大文字・小文字を区別する
- 文字や数字や「_」(アンダースコア)で構成される
- 「\$」の次を数字で始めることはできない

・・・▶ 「\$」を付ける

PHPの変数は、頭に「\$」を付けるのが特徴です。たとえば「\$nenrei」「\$TEL」のようになります。

・・・▶ 文字、数字、「_」で構成されるが、数字で始めることはできない

変数の先頭は、文字または「_」(アンダースコア)です。「\$30nendo」はダメですが「\$_30nendo」はよい、ということになります。

・・・▶ 大文字・小文字を区別する

大文字と小文字を区別する、ということは、たとえば「\$tel」「\$Tel」「\$TEL」はそれぞれ別の変数として扱われることになります。スクリプトを書いていると、ついつい「\$x」と「\$X」が入り乱れてしまったりするので注意しましょう。本書では、変数名はすべて小文字で表しています。

定義済み定数

PHPには、**定義済み定数**というユーザーが設定しなくても最初から定義されている値があります。たとえば、次のような円周率やPHPのバージョンを表す定数は、そのまま利用することができます。

▶ PHPの定数の例

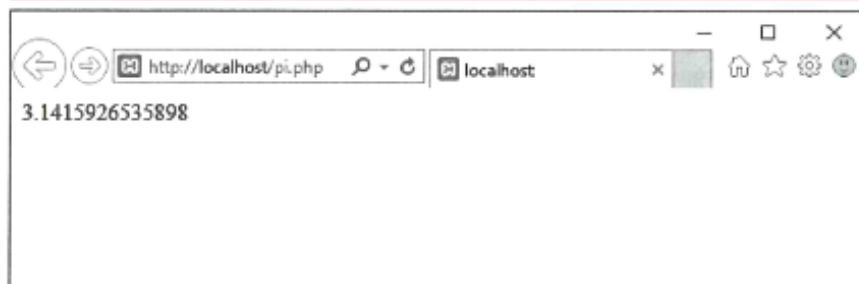
M_PI	円周率
PHP_VERSION	PHPのバージョン
PHP_OS	動作しているOS

たとえばList16-02のようにすると、円周率が表示されます。

LIST ■ 16-02 pi.php

```
<?php
print M_PI;
?>
```

FIG | 16-03 実行結果



変数のデータ型

次に、変数の「データ型」です。PHPの場合、データ型を定義しなくても変数を使えるという特徴があります。面倒な定義を必要とせず、「文字列を代入すれば文字列型」「整数を入れれば整数型」というように、代入したときにPHPが自動的にその型を決めてくれます。

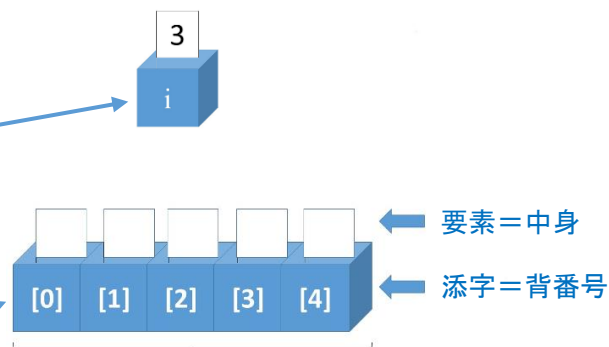
MySQLでテーブルを作るときは、最初からフィールドのデータ型を決めておく必要がありました(→P.65)。数値型のカラムに文字列を入れることは、基本的にはできませんでした。また、ストアドファンクションで使う変数は、最初に「DECLARE」でその型を宣言する必要がありました(→P.275)。

これに対してPHPは実におおらかで、「入れちゃえば、あとは型を合わせるよ」という仕様になっています。

次は、PHPで使うことのできる主なデータ型です。

TABLE ■ 16-01 PHPで使える主なデータ型

内容	データ型
整数	integer
浮動小数点	float, double
文字列	string
論理値	boolean
オブジェクト	object
配列	array
リソース型	resource
ヌル(空の値)	NULL



配列とは、複数の箱をセットにして名前を付けたもの
通常は”0”から始まる番号がそれぞれの箱に振られる

$\$i(0)$ 、 $\$i(1)$ 、 $\$i(2)$...のように使われる

代入すれば勝手にデータ型を設定してくれるので、私たちが学んでいる内容のレベルでは、特に気にする必要はありません。

文字列

PHPの文字列の扱いについてです。

文字列の結合

文字列をつなげる方法です。慣れないうちはちょっと戸惑うかもしれませんが、文字列を結合するときには、PHPでは「`.`」を使います。文字列データは「`"`」や「`'`」で囲むので、「西沢」と「夢路」を結合して表示するときは、「`"西沢"."`夢路」`"`」とします。

次のList16-03は、変数「`$a`」に「こんにちは」、変数「`$b`」に「ようこそ！SQLカフェへ」の文字列を代入し、この2つを結合して表示させる例です。

LIST 16-03 join.php

```
<?php
$a= "こんにちは";
$b="ようこそ！SQLカフェへ";
print $a.$b;
?>
```

1

「`$a.$b`」で、それぞれの変数に入った文字列を結合しています (1)。

「`"`」と「`'`」の使い方

文字列は「`"`」(ダブルクォーテーション)や「`'`」(シングルクォーテーション)で囲んで表します。

ただし、「`"`」で囲った文字の中に「`"`」の文字を入れたり、「`'`」で囲った文字の中に「`'`」の文字を入れたりすることはできません。これは、PHPは「`"`」や「`'`」があると、それを文字列の始点あるいは終点だと判断するため、結果として「`'`」や「`"`」が足りない、あるいは多いと処理されてしまうからです。MySQLのSQL文と同様です。

たとえば、次はエラーになります。

```
print "文字列は""で囲ってください";
```

「`"`」の中に「`"`」がある

...▶ エスケープ処理

このような場合、「"」や「'」の直前に「\」を付けることで、エラーを回避します。このような処理を、「エスケープ処理」といいました(→P.68)。

FIG 16-04 エスケープ処理

```
print "文字列は\" \"で囲ってください";
```

「"」に「\」を付ける(エスケープ処理)

...▶ 「'」「"」を使う

あるいは別の方法として、「"」を文字として扱うときは「'」で囲む、あるいは「'」を文字として扱うときは「"」で囲う、という方法があります。たとえば、次はエラーにはなりません。

```
print "文字列は' 'で囲んでください";
```

以降の章では、MySQLによるWebアプリケーションを作成するとき、PHPスクリプトによってSQL文を発行します。この場合の記述は、SQL文を文字列としてqueryというメソッドの引数に指定します(→P.425)。

queryやメソッドの意味については、また後で詳しく触れますが、今はとりあえず「query(ここに文字列としてのSQL文)」というように記述する方法だけを覚えてください。

たとえば「INSERT INTO tbl VALUES ('A101',...)」というSQL文を発行するときは、どのように記述すればよいでしょうか？ SQL文は文字列データなので「'」で囲むとします。「'」が使われているSQL文をさらに「'」で囲ったとすれば、これはエラーになってしまいます。

```
query('INSERT INTO tbl VALUES('A101',...)')
```

「'」を「'」で囲むとエラー

この場合「\」でエスケープするか、あるいは次のように、内側の「'」を文字として扱うために外側を「"」で囲むなどの対策が必要となります。

```
query("INSERT INTO tbl VALUES('A101',...)")
```

「'」を「"」で囲めばOK

あるいは、内側の「"」を文字として扱うために外側を「'」で囲むこともできます。PHP+MySQLの利用時に最初につまずくのは、こうした「'」や「"」の使い方であることも珍しくありません。queryメソッドについては以降で勉強しますが、ここでは「'」

の文字を「'」で、あるいは「"」の文字を「"」で囲うのは御法度！ということ覚えておいてください。

変数を「" "」と「' '」で囲むときの違い

文字列データは「" "」と「' '」のどちらで囲んでもよいのですが、変数を囲むときには処理結果に違いが生じます。

List16-04のように「\$a=123」とした場合、「print "\$a"」では「" "」の中の変数が展開されて、その値である「123」が表示されます(1)。

LIST ■ 16-04 double_quotation.php

```
<?php
$a=123;
print "$a";
?>
```

FIG | 16-05 実行結果

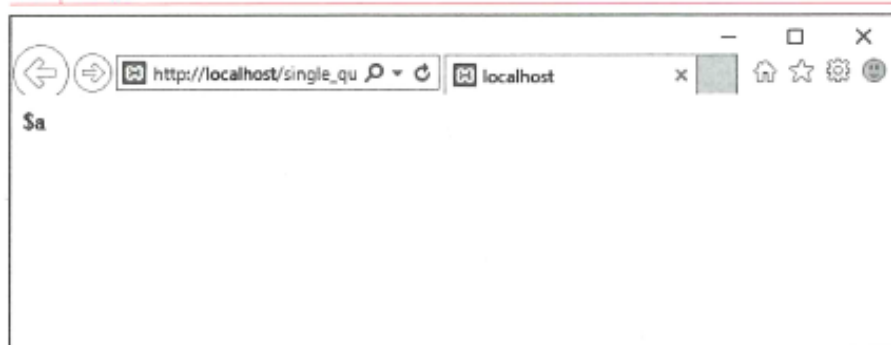


ところが、List16-05のように「print '\$a'」とすると、「' '」の中の変数は展開されずにそのままの文字列として扱われます(2)。つまり「\$a」という、そのままの文字列が表示されることになります。

LIST ■ 16-05 single_quotation.php

```
<?php
$a=123;
print '$a';
?>
```

FIG 16-06 実行結果



このように、PHPで発行するSQL文に変数がある場合は注意が必要です。

配列

配列とは何か

ここまで使ってきた変数は、「`$i=1000`」のように、1つの値だけを格納していました。実は、「見かけは1つでも、たくさんの値を格納できる」という、便利な変数もあります。

これを**配列**といいます。

「普通の変数ではなく、配列変数だ」と宣言する方法はいくつかありますが、最初に「array」という命令を使った方法を紹介します。なお、配列変数の名前も、P.357と同じ規則で付けられます。

たとえば、「\$m」という変数を配列にして、この中にたくさん格納するためには次のようにします。

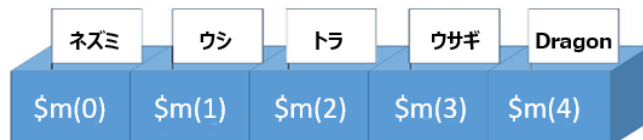
```
$m=array("ネズミ","ウシ","トラ","ウサギ","Dragon");
```

これで「\$m」という1つの変数に、「ネズミ」「ウシ」「トラ」「ウサギ」「Dragon」という5つの文字列が格納されました。PHPでは、「データを代入すれば、勝手にそのデータ型を設定」(→P.358)してくれるので、これで「\$m」は「文字列」型の配列変数となります。

次に、配列からそれぞれの値を取り出す方法です。たとえば上の例の値(文字列)は、左側からそれぞれに、最初は\$m[0]、2番目が\$m[1]、3番目が\$m[2]…という名前を付けられて格納されています。取り出すときは、この名前を指定することで取り出せます。つまり、「print \$m[2];」を実行すると「トラ」が書き出されます。「ウシ」ではありませんのでご注意ください。

TABLE ■ 16-06 格納された値と対応する配列変数

配列変数	格納された値
\$m[0]	ネズミ
\$m[1]	ウシ
\$m[2]	トラ
\$m[3]	ウサギ
\$m[4]	Dragon



[]の中の数字で、「何番目の値か」ということを判断しています。この[]の中に書く数字を、**添え字**といいます。添え字は「0」から付けられます。添え字が「1」からでなく「0」から始まることを不思議に思うかもしれませんが、しかし、コンピュータの世界では、数字の序列は「0」から始まることが多いのです。ですから、配列の最後の添え字は個数より1だけ小さくなるので注意してください。

たとえば上の例なら、5番目の「Dragon」は\$m[5]ではなく、\$m[4]となります。

配列に値を入れる方法

array命令を使わずに配列変数に値を入れる方法には、次のようなものがあります。

・・・▶ 直接代入する

```
$m[4]="Dragon";  
$m[0]="ネズミ";  
$m[2]="トラ";  
$m[1]="ウシ";  
$m[3]="ウサギ";
```

この場合、別に順番に関係なく入れることができます。また、いきなり「\$m[777]=「学問」」のような突飛な添え字を使ってもかまいません。

・・・▶ 順番に代入する

```
$m[]="ネズミ";  
$m[]="ウシ";  
$m[]="トラ";  
$m[]="ウサギ";  
$m[]="Dragon";
```

この場合、最初に代入したものから「\$m[0]」、「\$m[1]」…となります。for (→P.368) やwhile (→P.371) を使って繰り返して代入するときなどに便利です。

添え字の使い方は、慣れないうちは間違えやすいものです。使う際にはよく注意する必要があります。

・・・▶ 配列を使った例

List16-21は、P.377のList16-18を、配列を使って改良した例です。配列変数「\$m」に「眠くありませんか」「おはようございます」「こんにちは」「こんばんは」を代入し、0時から「眠くありませんか」、6時から「おはようございます」、9時から「こんにちは」、18時から「こんばんは」と表示されます。

LIST 16-21 hairetu.php

```

<?php
$m=array("眠くありませんか","おはようございます","こんにちは","こんばんは");
if (date("G")>=18){
    print $m[3];
}elseif(date("G")>=9){
    print $m[2];
}elseif(date("G")>=6){
    print $m[1];
}else{
    print $m[0];
}
?>

```

1の「眠くありませんか」は「\$m[0]」、「おはようございます」は「\$m[1]」、「こんにちは」は「\$m[2]」、「こんばんは」は「\$m[3]」に代入されます。

COLUMN ▶ 添え字の数値設定はほどほどに

数個の添え字しか使わないのに「\$m[777] = "学問"」などと離れた数値に設定することは、メモリを無駄に消費することになります。なるべく必要に応じた値にする方が無難でしょう。

連想配列

添え字は、順番を表す数値以外にも、ユーザーが決めた文字を使うこともできます。これを**連想配列**といいます。

たとえばテストの点を格納する配列変数で、英語なら「\$t["eigo"]」、国語なら「\$t["kokugo"]」…のように、次のように直観的にわかりやすい添え字を設定することができます。

List16-22は、連想配列を使った例です。「合計点は：225」と表示されます。

LIST 16-22 rensou.php

```

<?php
$t["eigo"]=73;
$t["suugaku"]=84;
$t["kokugo"]=68;
print "合計点は:";
print $t["eigo"]+$t["suugaku"]+$t["kokugo"];
?>

```

73	84	68
\$t["eigo"]	\$t["suugaku"]	\$t["kokugo"]

COLUMN ▶**連想配列の添え字が「"」や「'」なしでも動作する!?**

連想配列の添え字は、「"」か「'」で囲むのが正式です。しかし、たとえば「\$m[eigo]」のように連想配列の添え字を「"」や「'」で囲まなくても、多くの場合問題なく動作してしまいます。

▶動作するけど、正式ではない例(rensou_no.php)

```
<?php
$m[hoge]=100;
$m[piyo]=200;
print $m[hoge];
print "<br>";
print $m[piyo];
?>
```

上の例は、一般的には「100」「200」と正しく動作します。ところが厳密には誤りで、正しくは「\$m["hoge"]」と「\$m["piyo"]」とします。「\$m[hoge]」や「\$m[piyo]」としても通常ではエラー表示がされませんが、実は低いレベルのエラーが発生しています。しかしPHPには、「できるだけ頑張って解釈するよ!」というところがあるのです。

いずれにしても連想配列の添え字は、必ず「"」か、「'」で囲うものと考えたほうが無難です。