# セッションを使ったWebアプリをつくろう

## ゲームの基本形をつくる

数当てゲームをつくります。

ここの部分は「クッキーを使ったWebアプリをつくろう」と同じです。

## セッションの使用を開始する

今回は、セッションでサーバーとクライアントの通信をおこないます。

最初に kazuate.html を kazuate.php とファイル名を変更し、

最初の部分に以下の記述を加えます。

<?php session\_start(); ?>

これは、<!doctype>宣言よりも前に記述しないといけません。

この記述があると、まだセッションが開始されていない場合はセッションが開始されます。

具体的には、セッションIDが発行されて、ブラウザにヘッダ情報を通じて、セッションIDが通知されます。

次に「ゲーム開始」ボタンをつくって、コンピュータに数を決めさせます。

ゲーム開始ボタンは、リンクでつくりましょう。

<p><a href=”start.php”>ゲーム開始</a></p>

次は、start.phpです。

セッションを使いますから、まず先頭にそのための記述を加えます。

session\_start();

この場合、もうすでにセッションは開始されていますから、

この記述があると、発行されたセッションIDを引き継ぎます。

つまり同一セッションであると認識されるわけです。

rand関数で、数をセットすることとします。

$com = rand(1, 10);

今回はセッションを使ってますので、セッション変数が使えます。

セッション変数「$\_SESSION」に「num」の項目を作って、

そこにコンピュータの考えた数を覚えさせることにします。

$\_SESSION['num'] = $com;

まとめて書くと、以下のようになります。

session\_start();

$com = rand(1, 10);

$\_SESSION['num'] = $com;

今までのところのコードを以下に示します。

kazuate.php

<?php session\_start(); ?>

<!DOCTYPE html>

<html lang="ja">

<head>

<meta charset="utf-8">

<title>数当てゲーム</title>

<link rel="stylesheet" href="kazuate.css">

</head>

<body>

<header>

<h1>数当てゲーム</h1>

</header>

<article>

<p>私と数当てゲームをしましょう。<br>

私が1から10までで何か数字を思い浮かべますから、<br>

その数字を当ててください。</p>

<p><a href=”start.php”>ゲーム開始</a></p>

<form action="hantei.php" method="post">

数を入力：

<input type="number" min="1" max="10" name="kazu">

<input type="submit">

</form>

</article>

<hr>

<footer>

<small>&copy; 2017 Seiichi Nukayama</small>

</footer>

</body>

</html>

start.php

?php

session\_start();

$com = rand(1, 10);

$\_SESSION['num'] = $com;

?>

<!DOCTYPE html>

<html lang="ja">

<head>

<meta charset="utf-8">

<title>数当てゲーム</title>

<link rel="stylesheet" href="kazuate.css">

</head>

<body>

<header>

<h1>数当てゲーム</h1>

</header>

<article>

<p>数を決めました。いつでもどうぞ。</p>

<a href="kazuate.php">もどる</a>

</article>

<hr>

<footer>

<small>&copy; 2017 Seiichi Nukayama</small>

</footer>

</body>

</html>

## セッションについて

セッション変数は連想配列になってます。

start.phpでは、セッションの連想配列を以下のように使っています。

$com = rand(1, 10);

$\_SESSION['num'] = $com;

この$\_SESSION['num']は、サーバーにのみ保存されます。

つまり、大事なデータ（この場合は、コンピューターの考えた数）は、ネットワーク上を流れないということです。

ただ、ネットワーク上には、セッションIDは流れます。

サーバーが相手のコンピュータ（ブラウザ）を特定するために、セッションIDを使っているのです。

セッションがスタートしたとき、サーバーからユーザーのブラウザに向かって、セッションIDが送られます。

サーバーから送られたWebページを見て、ユーザーは何かしらの処理をおこない、送信ボタンを押します。

すると、このセッションIDは、ユーザー側のコンピュータ（ブラウザ）からサーバーに向かって、ヘッダ情報の中にクッキー情報として記述されて送られます。

サーバーはユーザー側から送られてきたセッションIDを見て、サーバーの中にあるセッションIDと照合し、合致したら、データをそのユーザーのデータであると認識します。

そして、そのセッションIDのもとに保存されていた $\_SESSION['num']とユーザーから送られてきた $\_POST['kazu'] を比べて、判定処理をおこなうというわけです。

ですから、ネットワーク上を流れるのはセッションIDだけです。

（このセッションIDの乗っ取りをどう防ぐかは、別課題です。session\_regenerate\_id関数を使います。）

さて、セッションが開始されて、ゲームがスタートしました。

サーバー側でrand関数でセットされた数は、$\_SESSION['num']で保存されてます。

処理が kazuate.php に戻ってきました。

ユーザーはここで数を入力して、hantei.phpにPOST情報として送ります。

hantei.php でもセッションを引き継ぎます。

session\_start();

そして、サーバーの中に保存してあった$\_SESSION['num']をとりだし、

$com = $\_SESSION['num'];

$com に代入して、以下、$manと比較します。

hantei.php の全部です。

<?php

session\_start();

$com = $\_SESSION['num'];

$man = $\_POST['kazu'];

if ($man < $com)

$message = "小さすぎます。";

else if ($man > $com)

$message = "大きすぎます。";

else if ($man == $com)

$message = "大正解！";

else

$message = "変ですね〜";

?>

<!DOCTYPE html>

<html lang="ja">

<head>

<meta charset="utf-8">

<title>数当てゲーム</title>

<link rel="stylesheet" href="kazuate.css">

</head>

<body>

<header>

<h1>数当てゲーム</h1>

</header>

<article>

<p><?php print $message; ?></p>

<a href="kazuate.php">もどる</a>

</article>

<hr>

<footer>

<small>&copy; 2017 Seiichi Nukayama</small>

</footer>

</body>

</html>

## セッションを終了する

セッションは開始されると、ブラウザが閉じるまでセッションIDは保持されます。

（ブラウザが閉じてもセッションIDを保持できるように、有効期限を設定することはできます）

ただし、サーバーに保存されているユーザーのデータは破棄されていません。

ユーザーのデータも含めて破棄するには、

1) セッション変数を空にする。

　 今回の場合は、$\_SESSION['num'] を空にすることになります。

2) セッションクッキーが存在する場合は破棄する。

　 今回の場合は、セッションIDをクッキーとして使用しています。

3) セッションを破棄する。

　 サーバーの中のユーザーのデータを破棄します。

という処理が必要です。

今回は、bye.php 作成して、そこで処理をしてみます。

まず、最初に session\_start(); を記述します。

<?php session\_start(); ?>

そのあとに、以下の記述を加えてください。

// セッション変数を空にする。

$\_SESSION = array(); ------ (1)

// セッションクッキーが存在する場合は破棄する。

if (isset($\_COOKIE[session\_name()])) { ------ (2)

$params = session\_get\_cookie\_params(); ------ (3)

setcookie(session\_name(), '', time() - 42000, ---- (4)

$params["path"], $params["domain"],

$params["secure"], $params["httponly"]

);

}

// セッションを破棄する。

session\_destroy(); ------ (5)

?>

(1) --- これで、$\_SESSION['num']はなくなります。

(2) --- session\_name関数は、現在のセッション名を取得します。

今回の場合だと、「PHPSESSID」です。

ゆえに、この行は、$\_COOKIE['PHPSESSID']が存在していれば、TRUEとなります。

この部分、PHPマニュアルや「PHPの絵本」だと、

if (ini\_get("session.use\_cookies"))

となっています。

これは、"session.use\_cookies"の値が 1 と設定されているならばということで、「クライアント側にセッションIDを保存する際にクッキーを使用するかどうか」ということを設定しています。デフォルトは 1（YES） です。

ini\_get関数は、設定オプションの値を返します。

(3) --- session\_get\_cookie\_params関数は、現在のクッキー情報を返します。

今回の場合は、以下のようになっているはずです。

lifetime -- '0' -- クッキーの生存期間

path -- '/' -- 情報が保存されている場所（パス

domain -- （未設定） -- クッキーのドメイン= --

secure -- （未設定） -- クッキーはセキュアな接続のみ送信

httponly -- （未設定） -- クッキーはHTTPを通してのみアクセス可能

(4) --- lifetimeに過去の時間を設定しています。これで、クッキー情報は破棄されています。

(5) --- サーバー内のユーザーデータを破棄します。

bye.php

<?php

session\_start();

// セッション変数を空にする。

$\_SESSION = array();

// セッションクッキーが存在する場合は破棄する。

if (isset($\_COOKIE[session\_name()])) {

$params = session\_get\_cookie\_params();

setcookie(session\_name(), '', time() - 42000,

$params["path"], $params["domain"],

$params["secure"], $params["httponly"]

);

}

// セッションを破棄する。

session\_destroy();

?>

<!DOCTYPE html>

<html lang="ja">

<head>

<meta charset="utf-8">

<title>数当てゲーム</title>

<link rel="stylesheet" href="kazuate.css">

</head>

<body>

<header>

<h1>数当てゲーム</h1>

</header>

<article>

<p>お疲れ様でした。</p>

<p><a href="kazuate.php">もどる</a></p>

</article>

<hr>

<footer>

<small>&copy; 2017 Seiichi Nukayama</small>

</footer>

</body>

</html>

これで完成です。