◆新規会員登録機能を作成する（1/3）[メール認証][仮登録]
2016 年 2 月 26 日
https://noumenon-th.net/programming/2016/02/26/registration/

PHP
メール認証を利用した新規会員登録を、3 回の記事に渡って説明していきます。今回はその 1 回目で、全体像を把握していきます。

まずはデモ画面からご覧下さい。

デモ画面（※デモでは実際にメール等は送られません。）

新規登録機能の概要は以下となります。

1.メールアドレスを登録する。

2.登録したメールに登録ページである URL が届く。

3.URL をクリックすると、登録画面が開くので登録をおこなう。

コードに関しては、大きく分けて２つの機能で構成されています。

1.メールアドレスの登録（仮登録）

メール登録フォーム（registration_mail_form.php）

メール確認・送信（registration_mail_check.php）

2.新規会員の登録（本登録）

会員登録フォーム（registration_form.php）

登録確認（registration_check.php）

登録完了（registration_insert.php）

データベースには以下２つのテーブルを用意しました。

（仮登録用でメールアドレスを保存する）pre_member テーブル

（本登録用）member テーブル

以上を踏まえた上で、登録機能の動きを説明していきます。

(画像 新規会員登録概要.png)

1・2
ユーザがメールの入力を行います。正しいメールが登録されれば、トークン（ランダムの文字列）を生成します。そのトークンを日付等と合わせてデータベースに保存します。それと同時にそのトークンを含めた URL を生成し、その URL をユーザにメールで通知します。

3
ユーザが URL のリンクを辿って会員登録画面へと訪れて来るので、その URL からトークンを取得（GET）します。そして、そのトークンとデータベースに登録されたそのトークンの登録日等を調べます。もし、有効（24 時間以内/未登録）なトークンであるならば、会員登録のフォームを表示させます。

4
ユーザが適当なアカウント・パスワードを入力したらデータベースへと入力します。それと同時に、そのユーザが辿ってきた URL（トークン）を無効にします。

以上がメール認証を利用した新規会員登録の概要です。

次回（2 回目）は 1・2 のメール送信（仮登録）の実装を説明していきます。

＊新規会員登録機能を作成する（2/3）[メール認証][仮登録]
2016 年 2 月 27 日
https://noumenon-th.net/programming/2016/02/27/registration2/

PHP
メール認証を利用した新規会員登録機能を作成します。今回はその 2 回目で、前回の記事に引き継ぎメール登録部分を実装していきます。

前回記事
新規会員登録機能を作成する（1/3）
デモ画面（※デモでは実際にメール等は送られません。）

今回は上記のイラストにおいて、手順 1・2 のメール登録機能を実装していきます。

1 メール登録フォーム（registration_mail_form.php）
2 メール確認・送信（registration_mail_check.php）

まず、仮登録用の pre_member テーブルを準備します。

(画像 pre_member.png)

pre_member テーブル

MySQL
```
1 CREATE TABLE pre_member (
2 id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
3 urltoken VARCHAR(128) NOT NULL,
4 mail VARCHAR(50) NOT NULL,
5 date DATETIME NOT NULL,
6 flag TINYINT(1) NOT NULL DEFAULT 0
7 )ENGINE=InnoDB DEFAULT CHARACTER SET=utf8;
```

urltokine カラムには URL に含めるトークンが入力されます。date カラムにはメールアドレスが登録された日付けが入ります。flag カラムはデフォルトが 0 の状態で自動入力され、会員登録が完了した時に、値を 1 に置き換えます。

registration_mail_form.php（メール登録フォーム）

PHP
```
1 <?php
2 session_start();
3
3 header("Content-type: text/html; charset=utf-8");
5
```

```
6 //クロスサイトリクエストフォージェリ（CSRF）対策
7 $_SESSION['token'] = base64_encode(openssl_random_pseudo_bytes(32));
8 $token = $_SESSION['token'];
9
10 //クリックジャッキング対策
11 header('X-FRAME-OPTIONS: SAMEORIGIN');
12
13 ?>
14
15 <!DOCTYPE html>
16 <html>
17 <head>
18 <title>メール登録画面</title>
19 <meta charset="utf-8">
20 </head>
21 <body>
22 <h1>メール登録画面</h1>
23
24 <form action="registration_mail_check.php" method="post">
25
26 <p>メールアドレス：<input type="text" name="mail" size="50"></p>
27
28 <input type="hidden" name="token" value="<?=$token?>">
29 <input type="submit" value="登録する">
30
31 </form>
32
33 </body>
34 </html>
```

7・8 行目
クロスサイトリクエストフォージェリ（CSRF）については以下の関連ページをご参照下さい。

クロスサイトリクエストフォージェリ（CSRF）[トークン]

11 行目
クリックジャッキング対策については以下の関連ページをご参照下さい。
クリックジャッキング[X-FRAME-OPTIONS]

registration_mail_check.php（メール確認・送信）

PHP

```
1 <?php
2 session_start();
3
4 header("Content-type: text/html; charset=utf-8");
5
6 //クロスサイトリクエストフォージェリ（CSRF）対策のトークン判定
7 if ($_POST['token'] != $_SESSION['token']){
8         echo "不正アクセスの可能性あり";
9         exit();
10 }
11
12 //クリックジャッキング対策
13 header('X-FRAME-OPTIONS: SAMEORIGIN');
14
15 //データベース接続
16 require_once("db.php");
17 $dbh = db_connect();
18
19 //エラーメッセージの初期化
20 $errors = array();
21
22 if(empty($_POST)) {
23         header("Location: registration_mail_form.php");
24         exit();
25 }else{
26         //POST されたデータを変数に入れる
27         $mail = isset($_POST['mail']) ? $_POST['mail'] : NULL;
28
29         //メール入力判定
30         if ($mail == ''){
31                 $errors['mail'] = "メールが入力されていません。";
32         }else{
33                 if(!preg_match("/^([a-zA-Z0-9])+([a-zA-Z0-9¥._-])*@([a-zA-Z0-9_-])+([a-zA-Z34      0-9¥._-]+)+$/",
$mail)){
35                         $errors['mail_check'] = "メールアドレスの形式が正しくありません。";
36                 }
37
38                 /*
40                 ここで本登録用の member テーブルにすでに登録されている mail かどうかをチェック 441 する。
```

```
42                  $errors['member_check'] = "このメールアドレスはすでに利用されております。";43
44                  */
45          }
46 }
47
48 if (count($errors) === 0){
49
50          $urltoken = hash('sha256',uniqid(rand(),1));
51          $url = "http://○○○.co.jp/registration_form.php"."?urltoken=".$urltoken;
52
53          //ここでデータベースに登録する
54          try{
55                  //例外処理を投げる（スロー）ようにする
56                  $dbh->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
57
58                  $statement = $dbh->prepare("INSERT INTO pre_member (urltoken,mail,date) VAL59 UES
(:urltoken,:mail,now() )");
60
61                  //プレースホルダへ実際の値を設定する
62                  $statement->bindValue(':urltoken', $urltoken, PDO::PARAM_STR);
63                  $statement->bindValue(':mail', $mail, PDO::PARAM_STR);
64                  $statement->execute();
65
66                  //データベース接続切断
67                  $dbh = null;
68
69          }catch (PDOException $e){
70                  print('Error:'.$e->getMessage());
71                  die();
72          }
73
74          //メールの宛先
75          $mailTo = $mail;
76
77          //Return-Path に指定するメールアドレス
78          $returnMail = 'web@sample.com';
79
80          $name = "ウェブの葉ショップ";
81          $mail = 'web@sample.com';
82          $subject = "【ウェブの葉ショップ】会員登録用 URL のお知らせ";
83
84 $body = <<< EOM
```

```
85   24 時間以内に下記の URL からご登録下さい。
86 {$url}
87 EOM;
88
89         mb_language('ja');
90         mb_internal_encoding('UTF-8');
91
92         //From ヘッダーを作成
93         $header = 'From: ' . mb_encode_mimeheader($name). ' <' . $mail. '>';
94
95         if (mb_send_mail($mailTo, $subject, $body, $header, '-f'. $returnMail)) {
96
97                   //セッション変数を全て解除
98                   $_SESSION = array();
99
100                  //クッキーの削除
101                  if (isset($_COOKIE["PHPSESSID"])) {
102                            setcookie("PHPSESSID", '', time() - 1800, '/');
103                  }
104
105                  //セッションを破棄する
106                  session_destroy();
107
108                  $message = "メールをお送りしました。24 時間以内にメールに記載された URL からご登録下さ
い。";
109
110        } else {
111                  $errors['mail_error'] = "メールの送信に失敗しました。";
112        }
113 }
114
115 ?>
116
117 <!DOCTYPE html>
118 <html>
119 <head>
120 <title>メール確認画面</title>
121 <meta charset="utf-8">
123 </head>
124 <body>
125 <h1>メール確認画面</h1>
126
```

```
127 <?php if (count($errors) === 0): ?>
128
129 <p><?=$message?></p>
130
131 <p>↓この URL が記載されたメールが届きます。</p>
132 <a href="<?=$url?>"><?=$url?></a>
133
134 <?php elseif(count($errors) > 0): ?>
135
136 <?php
137 foreach($errors as $value){
138         echo "<p>".$value."</p>";
139 }
140 ?>
141
142 <input type="button" value="戻る" onClick="history.back()">
143
144 <?php endif; ?>
145
146 </body>
147 </html>
```

16・17 行目
データベースの接続を行っています。db.php は本ページの下記に記載しています。

38・39 行目
本来ならば、既に会員登録完了してあるメールアドレスかどうかを調べます。今回は実装していません。

46・47 行目
URL に含めるトークンを生成し、URL につなげています。「?urltoken=」とすることで GET メソッドによりトークンを取得できるようになります。

70～107 行目
メールアドレスをチェックしデータベースに登録されたら、ユーザにメールを通知しています。ユーザは通知された URL のリンクから本登録画面（イラストの 3 部分）へと遷移してきます。

pre_member テーブルへの仮登録

仮登録 flag0

db.php（データベース接続）

```
<?php

function db_connect(){
	$dsn = 'mysql:host=〇〇〇;dbname=〇〇〇;charset=utf8';
	$user = '〇〇〇';
	$password = '〇〇〇';

	try{
		$dbh = new PDO($dsn, $user, $password);
		return $dbh;
	}catch (PDOException $e){
		print('Error:'.$e->getMessage());
		die();
	}
}

?>
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
<?php
```

```
function db_connect(){
	$dsn = 'mysql:host=○○○;dbname=○○○;charset=utf8';
	$user = '○○○';
	$password = '○○○';

	try{
		$dbh = new PDO($dsn, $user, $password);
		return $dbh;
	}catch (PDOException $e){
		print('Error:'.$e->getMessage());
		die();
	}
}

?>
```

4～6 行目

環境に合わせて適当な値を設定して下さい。

次回（3 回目）はイラストの 3・4 の本登録の実装を説明していきます。

＊新規会員登録機能を作成する（3/3）[メール認証][本登録]
2016 年 2 月 28 日
https://noumenon-th.net/programming/2016/02/28/registration3/

PHP
メール認証を利用した新規会員登録機能を作成します。今回は最後の 3 回目で、前回の記事に引き継ぎ会員の本登録部分を実装していきます。

前回記事

新規会員登録機能を作成する（1/3）
新規会員登録機能を作成する（2/3）

デモ画面（※デモでは実際にメール等は送られません。）
新規会員登録概要

最終回は上記のイラストにおいて、手順 3・4 の会員本登録機能を実装していきます。

3 会員登録フォーム（registration_form.php）
　会員登録確認（registration_check.php）

4 会員登録完了（registration_insert.php）

まず、本登録用の member テーブルを準備します。

member テーブル

MySQL
CREATE TABLE member (
id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
account VARCHAR(50) NOT NULL,
mail VARCHAR(50) NOT NULL,
password VARCHAR(128) NOT NULL,
flag TINYINT(1) NOT NULL DEFAULT 1
)ENGINE=InnoDB DEFAULT CHARACTER SET=utf8;
1
2
3
4
5
6
7
CREATE TABLE member (
id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
account VARCHAR(50) NOT NULL,
mail VARCHAR(50) NOT NULL,
password VARCHAR(128) NOT NULL,
flag TINYINT(1) NOT NULL DEFAULT 1
)ENGINE=InnoDB DEFAULT CHARACTER SET=utf8;

member

アカウント・メールアドレス・パスワード（ハッシュ化したパスワード）を保存します。flag に関しては今回は特に利用しませんが、デフォルトで 1 が自動入力されます。

registration_form.php（会員登録フォーム）

PHP

```
<?php
session_start();

header("Content-type: text/html; charset=utf-8");

//クロスサイトリクエストフォージェリ（CSRF）対策
$_SESSION['token'] = base64_encode(openssl_random_pseudo_bytes(32));
$token = $_SESSION['token'];

//クリックジャッキング対策
header('X-FRAME-OPTIONS: SAMEORIGIN');

//データベース接続
require_once("db.php");
$dbh = db_connect();

//エラーメッセージの初期化
$errors = array();

if(empty($_GET)) {
	header("Location: registration_mail_form.php");
	exit();
}else{
	//GET データを変数に入れる
	$urltoken = isset($_GET[urltoken]) ? $_GET[urltoken] : NULL;
	//メール入力判定
	if ($urltoken == ''){
		$errors['urltoken'] = "もう一度登録をやりなおして下さい。";
	}else{
		try{
```

```
                        //例外処理を投げる（スロー）ようにする
                        $dbh->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

                        //flag が 0 の未登録者・仮登録日から 24 時間以内
                        $statement = $dbh->prepare("SELECT mail FROM pre_member WHERE urltoken=(:urltoken) AND flag =0 AND date > now() - interval 24 hour");
                        $statement->bindValue(':urltoken', $urltoken, PDO::PARAM_STR);
                        $statement->execute();

                        //レコード件数取得
                        $row_count = $statement->rowCount();

                        //24 時間以内に仮登録され、本登録されていないトークンの場合
                        if( $row_count ==1){
                                $mail_array = $statement->fetch();
                                $mail = $mail_array[mail];
                                $_SESSION['mail'] = $mail;
                        }else{
                                $errors['urltoken_timeover'] = "この URL はご利用できません。有効期限が過ぎた等の問題があります。もう一度登録をやりなおして下さい。";
                        }

                        //データベース接続切断
                        $dbh = null;

                }catch (PDOException $e){
                        print('Error:'.$e->getMessage());
                        die();
                }
        }
}

?>

<!DOCTYPE html>
<html>
<head>
<title>会員登録画面</title>
<meta charset="utf-8">
</head>
<body>
<h1>会員登録画面</h1>
```

```
<?php if (count($errors) === 0): ?>

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

<p>メールアドレス：<?=htmlspecialchars($mail, ENT_QUOTES, 'UTF-8')?></p>
<p>アカウント名：<input type="text" name="account"></p>
<p>パスワード：<input type="text" name="password"></p>

<input type="hidden" name="token" value="<?=$token?>">
<input type="submit" value="確認する">

</form>

<?php elseif(count($errors) > 0): ?>

<?php
foreach($errors as $value){
        echo "<p>".$value."</p>";
}
?>

<?php endif; ?>

</body>
</html>
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
```

```
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
<?php
session_start();

header("Content-type: text/html; charset=utf-8");
```

```
//クロスサイトリクエストフォージェリ（CSRF）対策
$_SESSION['token'] = base64_encode(openssl_random_pseudo_bytes(32));
$token = $_SESSION['token'];

//クリックジャッキング対策
header('X-FRAME-OPTIONS: SAMEORIGIN');

//データベース接続
require_once("db.php");
$dbh = db_connect();

//エラーメッセージの初期化
$errors = array();

if(empty($_GET)) {
	header("Location: registration_mail_form.php");
	exit();
}else{
	//GET データを変数に入れる
	$urltoken = isset($_GET[urltoken]) ? $_GET[urltoken] : NULL;
	//メール入力判定
	if ($urltoken == ''){
		$errors['urltoken'] = "もう一度登録をやりなおして下さい。";
	}else{
		try{
			//例外処理を投げる（スロー）ようにする
			$dbh->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

			//flag が 0 の未登録者・仮登録日から 24 時間以内
			$statement = $dbh->prepare("SELECT mail FROM pre_member WHERE
urltoken=(:urltoken) AND flag =0 AND date > now() - interval 24 hour");
			$statement->bindValue(':urltoken', $urltoken, PDO::PARAM_STR);
			$statement->execute();

			//レコード件数取得
			$row_count = $statement->rowCount();

			//24 時間以内に仮登録され、本登録されていないトークンの場合
			if( $row_count ==1){
				$mail_array = $statement->fetch();
				$mail = $mail_array[mail];
```

```
                                $_SESSION['mail'] = $mail;
                        }else{
                                $errors['urltoken_timeover'] = "この URL はご利用できません。有効期限が過ぎた等の問題があります。もう一度登録をやりなおして下さい。";
                        }

                        //データベース接続切断
                        $dbh = null;

                }catch (PDOException $e){
                        print('Error:'.$e->getMessage());
                        die();
                }
        }
}

?>

<!DOCTYPE html>
<html>
<head>
<title>会員登録画面</title>
<meta charset="utf-8">
</head>
<body>
<h1>会員登録画面</h1>

<?php if (count($errors) === 0): ?>

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

<p>メールアドレス：<?=htmlspecialchars($mail, ENT_QUOTES, 'UTF-8')?></p>
<p>アカウント名：<input type="text" name="account"></p>
<p>パスワード：<input type="text" name="password"></p>

<input type="hidden" name="token" value="<?=$token?>">
<input type="submit" value="確認する">

</form>

<?php elseif(count($errors) > 0): ?>
```

```
<?php
foreach($errors as $value){
        echo "<p>".$value."</p>";
}
?>

<?php endif; ?>

</body>
</html>
```

7・8 行目

クロスサイトリクエストフォージェリ（CSRF）については以下の関連ページをご参照下さい。

クロスサイトリクエストフォージェリ（CSRF）[トークン]

11 行目

クリックジャッキング対策については以下の関連ページをご参照下さい。

クリックジャッキング[X-FRAME-OPTIONS]

14・15 行目

データベースの接続を行っています。db.php は本ページの下記に記載しています。

35 行目

GET メソッドで取得したトークン（urltoken）を元に仮登録（pre_member テーブル）のデータを検索しています。
その際に、仮登録が 24 時間以内であり flag が 0（まだ本登録していない）の条件を指定しています。

registration_check.php（会員登録確認）

PHP

```
<?php
session_start();

header("Content-type: text/html; charset=utf-8");

//クロスサイトリクエストフォージェリ（CSRF）対策のトークン判定
if ($_POST['token'] != $_SESSION['token']){
	echo "不正アクセスの可能性あり";
	exit();
}

//クリックジャッキング対策
header('X-FRAME-OPTIONS: SAMEORIGIN');

//前後にある半角全角スペースを削除する関数
function spaceTrim ($str) {
	// 行頭
	$str = preg_replace('/^[　 ]+/u', '', $str);
	// 末尾
	$str = preg_replace('/[　 ]+$/u', '', $str);
	return $str;
}

//エラーメッセージの初期化
$errors = array();

if(empty($_POST)) {
	header("Location: registration_mail_form.php");
	exit();
}else{
	//POST されたデータを各変数に入れる
	$account = isset($_POST['account']) ? $_POST['account'] : NULL;
	$password = isset($_POST['password']) ? $_POST['password'] : NULL;

	//前後にある半角全角スペースを削除
	$account = spaceTrim($account);
	$password = spaceTrim($password);
```

```
        //アカウント入力判定
        if ($account == ''):
                $errors['account'] = "アカウントが入力されていません。";
        elseif(mb_strlen($account)>10):
                $errors['account_length'] = "アカウントは 10 文字以内で入力して下さい。";
        endif;

        //パスワード入力判定
        if ($password == ''):
                $errors['password'] = "パスワードが入力されていません。";
        elseif(!preg_match('/^[0-9a-zA-Z]{5,30}$/', $_POST["password"])):
                $errors['password_length'] = "パスワードは半角英数字の 5 文字以上 30 文字以下で入力して下さい。";
        else:
                $password_hide = str_repeat('*', strlen($password));
        endif;

}

//エラーが無ければセッションに登録
if(count($errors) === 0){
        $_SESSION['account'] = $account;
        $_SESSION['password'] = $password;
}

?>

<!DOCTYPE html>
<html>
<head>
<title>会員登録確認画面</title>
<meta charset="utf-8">
</head>
<body>
<h1>会員登録確認画面</h1>

<?php if (count($errors) === 0): ?>


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

<p>メールアドレス：<?=htmlspecialchars($_SESSION['mail'], ENT_QUOTES)?></p>
```

```
<p>アカウント名：<?=htmlspecialchars($account, ENT_QUOTES)?></p>
<p>パスワード：<?=$password_hide?></p>

<input type="button" value="戻る" onClick="history.back()">
<input type="hidden" name="token" value="<?=$_POST['token']?>">
<input type="submit" value="登録する">

</form>

<?php elseif(count($errors) > 0): ?>

<?php
foreach($errors as $value){
        echo "<p>".$value."</p>";
}
?>

<input type="button" value="戻る" onClick="history.back()">

<?php endif; ?>

</body>
</html>
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
```

62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
<?php

```
session_start();

header("Content-type: text/html; charset=utf-8");

//クロスサイトリクエストフォージェリ（CSRF）対策のトークン判定
if ($_POST['token'] != $_SESSION['token']){
	echo "不正アクセスの可能性あり";
	exit();
}

//クリックジャッキング対策
header('X-FRAME-OPTIONS: SAMEORIGIN');

//前後にある半角全角スペースを削除する関数
function spaceTrim ($str) {
	// 行頭
	$str = preg_replace('/^[　 ]+/u', '', $str);
	// 末尾
	$str = preg_replace('/[　 ]+$/u', '', $str);
	return $str;
}

//エラーメッセージの初期化
$errors = array();

if(empty($_POST)) {
	header("Location: registration_mail_form.php");
	exit();
}else{
	//POST されたデータを各変数に入れる
	$account = isset($_POST['account']) ? $_POST['account'] : NULL;
	$password = isset($_POST['password']) ? $_POST['password'] : NULL;

	//前後にある半角全角スペースを削除
	$account = spaceTrim($account);
	$password = spaceTrim($password);

	//アカウント入力判定
	if ($account == ''):
		$errors['account'] = "アカウントが入力されていません。";
	elseif(mb_strlen($account)>10):
		$errors['account_length'] = "アカウントは 10 文字以内で入力して下さい。";
```

```
        endif;

        //パスワード入力判定
        if ($password == ''):
                $errors['password'] = "パスワードが入力されていません。";
        elseif(!preg_match('/^[0-9a-zA-Z]{5,30}$/', $_POST["password"])):
                $errors['password_length'] = "パスワードは半角英数字の 5 文字以上 30 文字以下で入力して下さい。";
        else:
                $password_hide = str_repeat('*', strlen($password));
        endif;

}

//エラーが無ければセッションに登録
if(count($errors) === 0){
        $_SESSION['account'] = $account;
        $_SESSION['password'] = $password;
}

?>

<!DOCTYPE html>
<html>
<head>
<title>会員登録確認画面</title>
<meta charset="utf-8">
</head>
<body>
<h1>会員登録確認画面</h1>

<?php if (count($errors) === 0): ?>


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

<p>メールアドレス：<?=htmlspecialchars($_SESSION['mail'], ENT_QUOTES)?></p>
<p>アカウント名：<?=htmlspecialchars($account, ENT_QUOTES)?></p>
<p>パスワード：<?=$password_hide?></p>

<input type="button" value="戻る" onClick="history.back()">
<input type="hidden" name="token" value="<?=$_POST['token']?>">
```

```
<input type="submit" value="登録する">

</form>

<?php elseif(count($errors) > 0): ?>

<?php
foreach($errors as $value){
        echo "<p>".$value."</p>";
}
?>

<input type="button" value="戻る" onClick="history.back()">

<?php endif; ?>

</body>
</html>
```

※今回は、同じアカウント名でも登録できるようになっています。もし同じアカウント名をはじく場合は、アカウント入力判定の部分で member テーブルを検索する必要があります。

registration_insert.php（会員登録完了）

```
<?php
session_start();

header("Content-type: text/html; charset=utf-8");

//クロスサイトリクエストフォージェリ（CSRF）対策のトークン判定
if ($_POST['token'] != $_SESSION['token']){
        echo "不正アクセスの可能性あり";
        exit();
}

//クリックジャッキング対策
header('X-FRAME-OPTIONS: SAMEORIGIN');
```

```
//データベース接続
require_once("db.php");
$dbh = db_connect();

//エラーメッセージの初期化
$errors = array();

if(empty($_POST)) {
	header("Location: registration_mail_form.php");
	exit();
}

$mail = $_SESSION['mail'];
$account = $_SESSION['account'];

//パスワードのハッシュ化
$password_hash =  password_hash($_SESSION['password'], PASSWORD_DEFAULT);

//ここでデータベースに登録する
try{
	//例外処理を投げる（スロー）ようにする
	$dbh->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

	//トランザクション開始
	$dbh->beginTransaction();

	//member テーブルに本登録する
	$statement = $dbh->prepare("INSERT INTO member (account,mail,password) VALUES (:account,:mail,:password_hash)");
	//プレースホルダへ実際の値を設定する
	$statement->bindValue(':account', $account, PDO::PARAM_STR);
	$statement->bindValue(':mail', $mail, PDO::PARAM_STR);
	$statement->bindValue(':password_hash', $password_hash, PDO::PARAM_STR);
	$statement->execute();

	//pre_member の flag を 1 にする
	$statement = $dbh->prepare("UPDATE pre_member SET flag=1 WHERE mail=(:mail)");
	//プレースホルダへ実際の値を設定する
	$statement->bindValue(':mail', $mail, PDO::PARAM_STR);
	$statement->execute();
```

```
        // トランザクション完了（コミット）
        $dbh->commit();

        //データベース接続切断
        $dbh = null;

        //セッション変数を全て解除
        $_SESSION = array();

        //セッションクッキーの削除・sessionid との関係を探れ。つまりはじめの sesssionid を名前でやる
        if (isset($_COOKIE["PHPSESSID"])) {
                setcookie("PHPSESSID", '', time() - 1800, '/');
        }

        //セッションを破棄する
        session_destroy();

        /*
        登録完了のメールを送信
        */

}catch (PDOException $e){
        //トランザクション取り消し（ロールバック）
        $dbh->rollBack();
        $errors['error'] = "もう一度やりなおして下さい。";
        print('Error:'.$e->getMessage());
}

?>

<!DOCTYPE html>
<html>
<head>
<title>会員登録完了画面</title>
<meta charset="utf-8">
</head>
<body>

<?php if (count($errors) === 0): ?>
<h1>会員登録完了画面</h1>

<p>登録完了いたしました。ログイン画面からどうぞ。</p>
```

```
<p><a href="">ログイン画面（未リンク）</a></p>

<?php elseif(count($errors) > 0): ?>

<?php
foreach($errors as $value){
        echo "<p>".$value."</p>";
}
?>

<?php endif; ?>

</body>
</html>
```

71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110

```
<?php
session_start();
```

```
header("Content-type: text/html; charset=utf-8");

//クロスサイトリクエストフォージェリ（CSRF）対策のトークン判定
if ($_POST['token'] != $_SESSION['token']){
	echo "不正アクセスの可能性あり";
	exit();
}

//クリックジャッキング対策
header('X-FRAME-OPTIONS: SAMEORIGIN');

//データベース接続
require_once("db.php");
$dbh = db_connect();

//エラーメッセージの初期化
$errors = array();

if(empty($_POST)) {
	header("Location: registration_mail_form.php");
	exit();
}

$mail = $_SESSION['mail'];
$account = $_SESSION['account'];

//パスワードのハッシュ化
$password_hash =  password_hash($_SESSION['password'], PASSWORD_DEFAULT);

//ここでデータベースに登録する
try{
	//例外処理を投げる（スロー）ようにする
	$dbh->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

	//トランザクション開始
	$dbh->beginTransaction();

	//member テーブルに本登録する
	$statement = $dbh->prepare("INSERT INTO member (account,mail,password) VALUES (:account,:mail,:password_hash)");
	//プレースホルダへ実際の値を設定する
```

```
        $statement->bindValue(':account', $account, PDO::PARAM_STR);
        $statement->bindValue(':mail', $mail, PDO::PARAM_STR);
        $statement->bindValue(':password_hash', $password_hash, PDO::PARAM_STR);
        $statement->execute();

        //pre_member の flag を 1 にする
        $statement = $dbh->prepare("UPDATE pre_member SET flag=1 WHERE mail=(:mail)");
        //プレースホルダへ実際の値を設定する
        $statement->bindValue(':mail', $mail, PDO::PARAM_STR);
        $statement->execute();

        //  トランザクション完了（コミット）
        $dbh->commit();

        //データベース接続切断
        $dbh = null;

        //セッション変数を全て解除
        $_SESSION = array();

        //セッションクッキーの削除・sessionid との関係を探れ。つまりはじめの sesssionid を名前でやる
        if (isset($_COOKIE["PHPSESSID"])) {
                setcookie("PHPSESSID", '', time() - 1800, '/');
        }

        //セッションを破棄する
        session_destroy();

        /*
        登録完了のメールを送信
        */

}catch (PDOException $e){
        //トランザクション取り消し（ロールバック）
        $dbh->rollBack();
        $errors['error'] = "もう一度やりなおして下さい。";
        print('Error:'.$e->getMessage());
}

?>

<!DOCTYPE html>
```

```
<html>
<head>
<title>会員登録完了画面</title>
<meta charset="utf-8">
</head>
<body>

<?php if (count($errors) === 0): ?>
<h1>会員登録完了画面</h1>

<p>登録完了いたしました。ログイン画面からどうぞ。</p>
<p><a href="">ログイン画面（未リンク）</a></p>

<?php elseif(count($errors) > 0): ?>

<?php
foreach($errors as $value){
        echo "<p>".$value."</p>";
}
?>

<?php endif; ?>

</body>
</html>
```

31 行目

パスワードをハッシュ化してデータベースに保存します。ハッシュ化については以下の関連ページをご参照下さい。

ハッシュ関数について/password_hash()を利用する

34～81 行目

会員データを本登録である member テーブルに入力し、それと同時に仮登録である pre_member テーブルの flag を 1 にしています。この二つの操作はトランザクションによって制御されています。トランザクションについて

は以下の関連ページをご参加下さい。

PDO でトランザクション処理を行う[beginTransaction]

pre_member テーブルで flag を 1 にする。

仮登録 flag1

member テーブル

本登録

73 行目

本登録が完了したら、完了通知のメールをユーザに送りますが、今回は実装していません。

db.php（データベース接続）

PHP

```
<?php

function db_connect(){
	$dsn = 'mysql:host=○○○;dbname=○○○;charset=utf8';
	$user = '○○○';
	$password = '○○○';

	try{
		$dbh = new PDO($dsn, $user, $password);
		return $dbh;
	}catch (PDOException $e){
```

```
                print('Error:'.$e->getMessage());
                die();
        }
}
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
<?php

function db_connect(){
        $dsn = 'mysql:host=○○○;dbname=○○○;charset=utf8';
        $user = '○○○';
        $password = '○○○';

        try{
                $dbh = new PDO($dsn, $user, $password);
                return $dbh;
        }catch (PDOException $e){
                print('Error:'.$e->getMessage());
                die();
        }
}
```

4～6 行目

環境に合わせて適当な値を設定して下さい。