第５回　ファイルアップロード

５－１．ファイルのアップロード

ファイルをアップロードするには、**formタグにenctypeプロパティの設定を行う必要があります。**

**この設定がないと、アップロードしたファイル名のみが文字列として扱われる**こととなります。

<form>開始タグ

■**enctype**プロパティ

|  |  |
| --- | --- |
| 設定値 | 意味 |
| application/x-www-form-urlencoded | URLエンコードを指定 （初期値） |
| multipart/form-data | マルチパートを指定。**フォーム内でファイルを送信する場合は、この形式を指定**しておく必要がある |

■SampleCode（HTML）

**※inputタグの設定にも注意。「type=”text”」ではうまくいきません。【HTMLの復習】**

**テキストボックスは「**name="upfile"**」にて設定。**

<form action="sample01\_6.php" method="POST" enctype="multipart/form-data">

<div id="frame" class="p-5 border-info rounded" style="border:1px dashed;">

<!-- ファイル大きさ制限数の保持（hidden） -->

<input type="hidden" name="MAX\_FILE\_SIZE" value="2000000">

<!-- ファイル入力箇所 -->

<div id="file" class="input-group">

<input type="file" id="inputFile"

class="form-control form-control-lg" name="upfile">

  　</div>

</div>

<div class="p-5 d-grid gap-2 d-md-flex justify-content-md-end">

<button type="submit" class="btn btn-danger btn-lg">アップロード</button>

</div>

</form>

■Sample（入力画面）　ファイル選択

※[ファイル選択]ボタン押下にて、ファイルを選択すると、選択したファイル名が右側に表示される。

５－２．FILEデータの受け取り

■FILEデータを取得できるスーパーグローバル変数

|  |  |
| --- | --- |
| $\_FILES | HTTP ファイルアップロード変数  POSTで送信された**ファイルの情報が連想配列で取得**できます。 |

■SampleCode（入力画面のHTML**テキストボックスは「**name="upfile"**」にて設定。**）

<?php

// 値チェックなどを行うので、変数に格納して使用が一般的

$upFile = $\_FILES["upfile"];

?>

■$\_FILES[userfile]の内容

※userfileには<input type=”file” >のname属性名が入る。

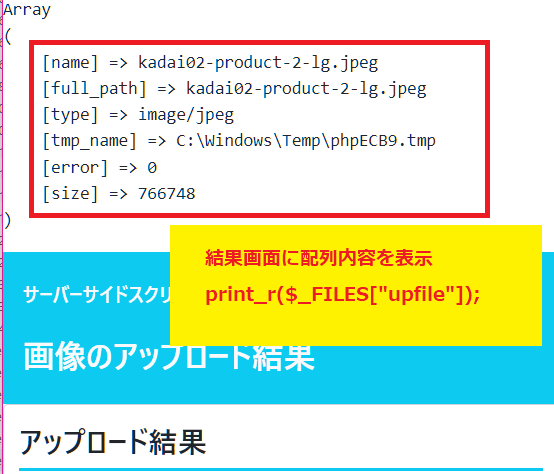
SampleCodeでは name=”upfile” なので、$\_FILES["upfile"]　で取得可能。

　※表は次のページにも続きます。

|  |  |
| --- | --- |
| キー | 内容 |
| $\_FILES[userfile]**[“name”]** | クライアントPC上のファイル名 |
| $\_FILES[userfile]**[“full\_path”]** | ファイルのフルパス。  この値は実際のディレクトリ構造を反映しているとは必ずしも言えないため、 信用できません。 PHP 8.1.0 以降で利用可能です。 |
| $\_FILES[userfile]**[“type”]** | ファイルのMIMEタイプ |
| $\_FILES[userfile]**[“tmp\_name”]** | 仮ディレクトリ内のテンポラリファイルの名前 |
| $\_FILES[userfile]**[“error”]** | エラー種別を表す数値   |  |  |  | | --- | --- | --- | | 定数 | 値 | 内容 | | UPLOAD\_ERR\_OK | 0 | 成功 | | UPLOAD\_ERR\_INI\_SIZE | 1 | ファイルは、  php.iniの  upload\_max\_filesize  ディレクティブの値を超えています。 | | UPLOAD\_ERR\_FORM\_SIZE | 2 | ファイルは、HTML フォームで指定された MAX\_FILE\_SIZE を超えています。 | | UPLOAD\_ERR\_PARTIAL | 3 | ファイルは一部のみしかアップロードされていません。 | | UPLOAD\_ERR\_NO\_FILE | 4 | ファイルはアップロードされませんでした。(ファイルがないときはここ) | | UPLOAD\_ERR\_NO\_TMP\_DIR | 6 | テンポラリフォルダがありません。 | | UPLOAD\_ERR\_CANT\_WRITE | 7 | ディスクへの書き込みに失敗しました。 | | UPLOAD\_ERR\_EXTENSION | 8 | PHP の拡張モジュールがファイルのアップロードを中止しました。 どの拡張モジュールがファイルアップロードを中止させたのかを突き止めることはできません。 | |
| $\_FILES[userfile]**[“size”]** | ファイルのサイズ（バイト単位） |

SampleCodeで $\_FILES["upfile"]　の内容を表示してみます。

■Sample画面（ $\_FILES["upfile"] の内容を表示）



HTMLでinputタグのfile属性を設定するだけで、上記のようにサーバにファイルはアップされています。ただし、このファイルは、PHPのテンポラリフォルダ（一時的な保存をしておくためのフォルダ）内にあるため、このまま何も処理をしないと、画面を閉じると消えてしまいます。

なので、テンポラリフォルダのファイルを、保存用のフォルダ（今回は、asset/storageフォルダ）に保存しなおす必要があります。

ファイルアップロード（move\_uploaded\_file関数）



■引数

|  |  |
| --- | --- |
| from | 移動元ファイルパス  ※$\_FILES[userfile][“tmp\_name”]に保存されている仮ディレクトリ |
| to | 移動先ファイルパス |

■SampleCode（ファイル移動）　※Sampleのため、入力値チェックは未。

<?php

//変数の定義

$message = "";//エラーメッセージ

$status = true;//アップロード成功かどうか

$newfile="";//新しいファイル名

$extension="";//拡張子

if ($\_FILES["upfile"]["error"] === UPLOAD\_ERR\_OK) {

//アップロードはできているので、画像保存処理

if(!move\_uploaded\_file($\_FILES["upfile"]["tmp\_name"],"./asset/storage/".$\_FILES["upfile"]["name"])) {

//保存できなかった場合は、エラーメッセージを出力

$status = false;

$message = "画像ファイルを保存できませんでした";

}

}

}else{

//ファイルアップロードエラー

$status = false;

switch ($\_FILES["upfile"]["error"]) {

case UPLOAD\_ERR\_INI\_SIZE:

case UPLOAD\_ERR\_FORM\_SIZE:

$message = "ファイルのサイズが大きすぎます";

break;

・・・・・・・以下略・・・・・・・・・・・

}

}?>

■SampleCode（HTML　一部抜粋）

HTML内でｉｆ文を使用して、アップロードが成功した場合は、その画像ファイルに表示させるようにして、失敗した場合は、エラーメッセージを表示するようにする。

<!-- ファイル結果表示箇所 -->

<div class="text-center">

  <?php if ($status) : ?>

    <figure class="d-inline-block me-1 mt-1 mb-5">

<img class="img-fluid" src="./asset/storage/<?= ファイル名 ?>">

</figure>

  <?php else : ?>

    <p class="text-danger"><?= $message ?></p>

  <?php endif ?>

</div>

■Sample（入力画面）　ファイル選択

・入力（ファイル選択後）　　　　　　　　　　　　　　　　　　・結果

５－３．その他　関数など

ファイル情報取得とファイルアップロードは５－２までで完了しました。

ここからは、セキュリティ的なロジック追加です。

**<<ファイル名のリネームについて>>**

特に、ユーザーが自由にファイルのアップロードをする場合、様々な問題が起こる可能性があります。

1. 別のユーザーが同じファイル名を付けてアップロードする可能性がある。（上書きされてしまう）
2. ファイル名に2バイト文字または半角カナなどの文字が使用されている。（文字化け等の可能性がある）
3. 悪意のあるユーザが「../../」などパスを含んだ名前を付けている可能性がある。（Directory Traversal攻撃　重要なファイルが壊されるかも）
4. 悪意のあるユーザが、画像ファイルを装って、そうではないファイルをアップロードしてくる可能性がある。（悪意のあるPGが実行されてしまうかも）

そのような不具合を回避するためにもアップロードされたファイルはリネーム（名前を付け直し）するのが一般的です。ここでの注意点は、リネームの際に名前が被らない（ユニークな）ようにすることです。

**<<画像ファイルかどうかの判定について>>**

アップロードされたファイルが画像データかどうかを判定するにはいくつか方法があるが、今回は、$\_FILESの["type"]の値（MIME）で判定する。

|  |  |  |
| --- | --- | --- |
|  | **拡張子** | **MIME** |
| GIF | **.gif** | image/gif |
| JPEG | **.jpg　.jpeg** | image/jpeg |
| PNG | **.png** | image/png |

赤字は、付け直し後の拡張子 （<https://www.tagindex.com/html/basic/mimetype.html>）

MIMEタイプは改ざんすることができてしまうため、拡張子を付け直すこととする。

さらに念を入れるために、アップロードされたファイルの実行権限をなくす（chmod　６４４）こともしておいたほうが良いが、Linuxの話になってしまうので今回は省略する。

<?php

// 画像ファイルかどうかMIME形式で判定

switch($\_FILES["upfile"]["type"]){

case "image/gif":

$extension = ".gif";//拡張子の設定

break;

・・・・・・・・・・・・・・・・・・以下略・・・・・・・・・・・・・・・・・・・・・・・・・・・

}

//画像ファイルだった場合、ファイル名を変更

if($status){

// リネーム処理

$newfile = uniqid().$extension;

・・・・・・・・・・・・・・・・・・・・・・・・・以下ファイル保存処理…………………………………

?>

●uniqid関数

一意な ID を生成する

uniqid()

戻り値：string（ユニークな文字列）

（https://www.php.net/manual/ja/function.uniqid.php）

**<<$\_FILESがないまたは改ざんされている場合>>**

POST送信されているのに何らかの理由（悪意も含めて）で$\_FILES["upfile"]が存在しなかった、または、配列の構成が改ざんされていた場合（ファイルを指定せずに送信する場合は、$\_FILES["upfile"]は存在するのでこの例には当たらない）、プログラムを終了する。

（この処理の検証は通常処理からでは難しいので、とりあえず、以下のような記述を書いておくこと。）

<?php

// $\_FILES['upfile']が存在しない場合は強制終了

if(!isset($\_FILES['upfile']['error'])|| !is\_int($\_FILES['upfile']['error'])){

exit("エラー");

}

?>

POSTでデータが送信されたか確認

課題５\_２は、課題5\_1から実行されていることを前提にプログラムされています。なので、いきなり課題5\_2を呼び出すと不具合が起こります。したがって、いきなり課題5\_2が実行された（GET送信された）場合は、課題5\_1に強制的に画面遷移をさせる（リダイレクト）ようにします。

■Webクライアントの情報を取得できるスーパーグローバル変数

|  |  |
| --- | --- |
| $\_SERVER | サーバー情報および実行時の環境情報  **Webクライアントの情報が連想配列で取得**できます。 |

■$\_SERVER変数の主なキー

※一部抜粋

|  |  |
| --- | --- |
| キー | 内容 |
| PHP\_SELF | URLドキュメントルート以下 |
| SERVER\_ADDR | WebサーバーのIPアドレス |
| SERVER\_NAME | Webサーバーのホスト名 |
| **REQUEST\_METHOD** | **ページにアクセスする際に使用されたリクエストのメソッド名**  **'GET', 'HEAD', 'POST', 'PUT' など。** |
| HTTP\_USER\_AGENT | クライアントのブラウザ種別を表すUser-Agent文字列 |
| HTTPS | http://接続時は空欄、https://接続時は何らかの文字列 |
| SCRIPT\_FILENAME | アクセスされているPHPファイルの絶対パス |
| SCRIPT\_NAME | アクセスされているPHPのURLのパス |

■SampleCode（結果画面sample01\_6.php）

※POST形式のリクエストがなければ、入力画面sample01\_5.phpに遷移する。

<?php

// POST形式でなければ

if ($\_SERVER["REQUEST\_METHOD"] !== "POST") {

  　header("Location: kadai05\_1.php"); //kadai05\_1.phpへ画面遷移

  　exit; // 処理を終了させる

}

?>

リダイレクト処理（header関数）

header( "Location: /\* 遷移先のURL \*/" );

※他に引数を指定できますが、今回は概要のみ。

※「　Location:　」と使用することで、指定したURLへリダイレクト（自動的に転送）します。

※header関数の前にechoやprint、HTMLなどを指定しないこと。画面遷移できなくなります。

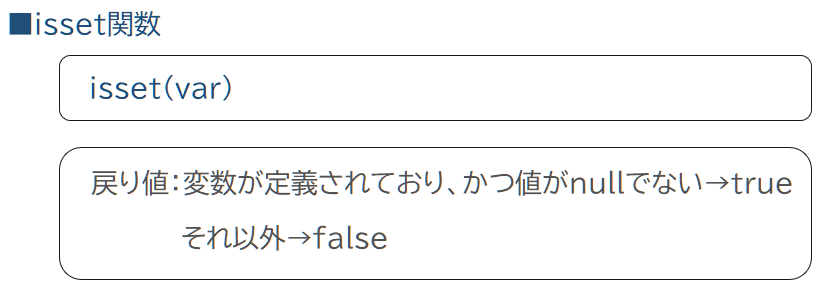
処理終了（exit）

　　　exit;

※以降の処理は行われません。

続いて、ファイルアップロードを行うより前に、データのチェックを行います。

値チェック（isset関数）



■引数

|  |  |
| --- | --- |
| var | 調べたい変数 |