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。

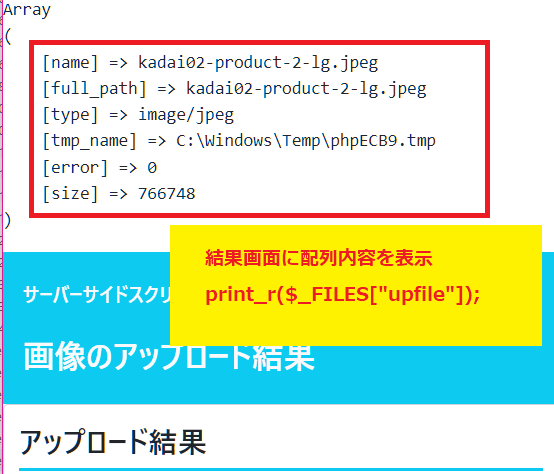
SampleCodeでは name=”upfile” なので、$\_FILES["upfile"]　で取得可能。

　※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。 | | 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）



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

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

//↓結果格納用の連想配列

$result = [

  　"status"  => true,　　//結果フラグ用

  　"message" => null, 　//結果メッセージ用

  　"result"  => null,　 //結果のファイルパス用

];

//↓移動先のディレクトリ参照（「\_DIR\_」は自分自身のディレクトリを示す）

$moveFilePath = \_\_DIR\_\_ . "/asset/storage/{$upFile["name"]}";

  if (move\_uploaded\_file($upFile["tmp\_name"], $moveFilePath)) {

    $result["message"] = "ファイルのアップロードに成功しました";

//↓HTMLのsrcプロパティに設定するパス用なので、フォルダパスではなく、URLで指定

$result["result"] =

"http://localhost/php1\_2023\_配布用/asset/storage/{$upFile["name"]}";

  } else {

    $result["status"] = false;

    $result["message"] = "ファイルのアップロードに失敗しました";

  }

?>

■SampleCode（HTML　一部抜粋）

<!-- ファイル結果表示箇所 -->

<div class="text-center">

  <?php if ($result["status"]) : ?>

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

<img class="img-fluid" src="<?= $result["result"] ?>">

</figure>

  <?php else : ?>

    <p class="text-danger"><?= $result["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。ただし、入力画面から送信されたデータ形式や値の確認はまだしていません。

ここからは、データのチェックに必要な関数の一例を学びましょう。

POSTでデータが送信されたか確認

■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: sample01\_5.php"); //sample01\_5.phpへ画面遷移

  　exit; // 処理を終了させる

}

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

?>

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。

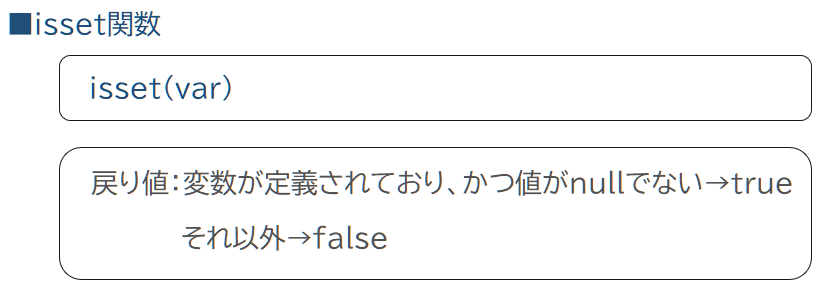
処理終了（exit）

　　　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 |

■SampleCode（結果画面sample01\_6.php）

※$\_FILES[“upfile”]が定義されていない、またはnullである場合、結果用変数にエラーとして値をセット。

<?php

　// ～（省略）～

　　// 定義されていない、またはnullである場合

if (!isset($\_FILES["upfile"])) {

  　$result["status"] = false; // エラーのときfalseとする

  　$result["message"] = "ファイルのアップロードに失敗しました"; // 画面表示用

}

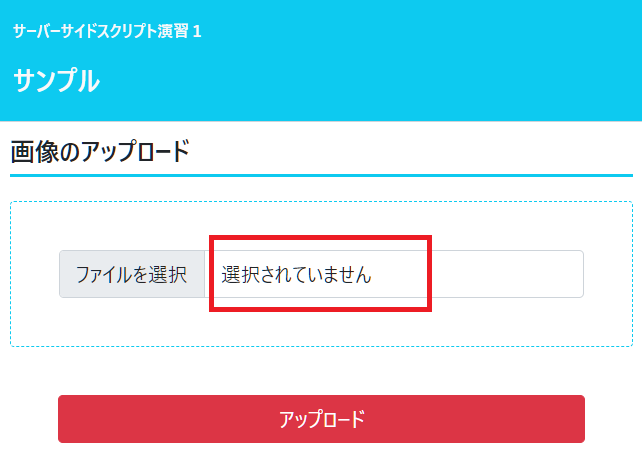
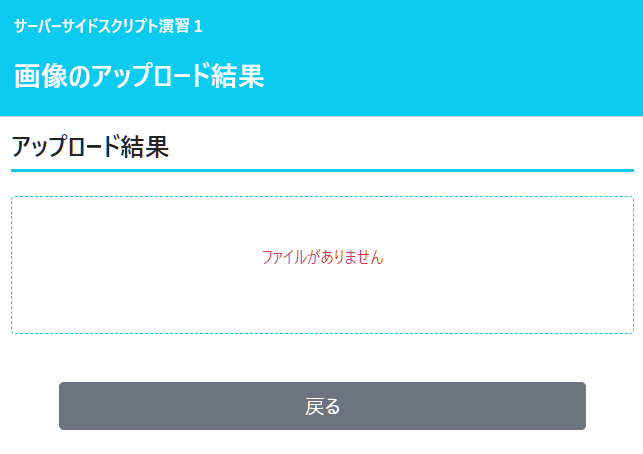
$upFile = $\_FILES["upfile"];

// ～（省略）～

?>

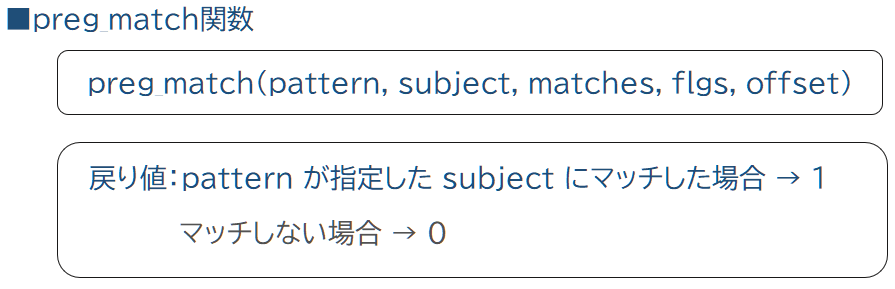
■Sample画面

・入力（ファイル選択なし）　　　　　　　　　　　　　　　　　　　・結果（issetのエラーチェックに該当）

続いて、ファイルの種別が画像であるかを調べます。

パターンチェック（preg\_match関数）



■引数

|  |  |
| --- | --- |
| pettern | string型。検索するパターンを表す文字列。正規表現で記載。 |
| subject | string型。入力文字列 |
| matches | array型。省略可。初期値 null |
| flags | int型。省略可。初期値0 |
| offset | int型。省略可。初期値0 |

■SampleCode（ファイルが画像であるかパターンチェック）

**※ファイル種別は $upFile[“type”] に格納される。**

**画像の場合は、「image/種別」となるので、「image」の文字列が入っているかを確認。**

<?php

　// ～（省略）～

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

//文字列imageを含まないとき＝画像ではないとき、エラーとして連想配列に値を格納

//※文字列のパターンは「’/image/’」（ / スラッシュで囲む）

//※条件の前の「!」は否定（ＮＯＴ）　初回の復習

if (!preg\_match('/image/', $upFile["type"])) {

  　$result["status"] = false;

  　$result["message"] = "画像ファイル以外はアップロードできません";

}

// ～（省略）～

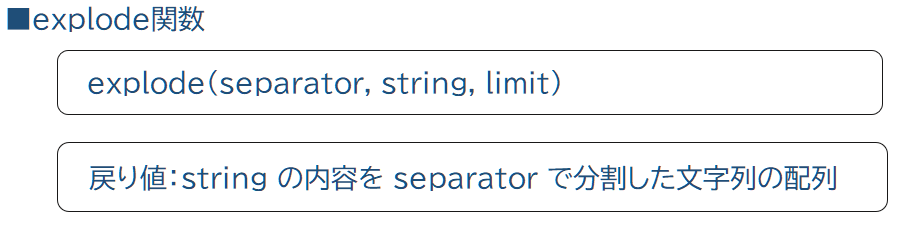
?>

■Sample画面

・入力（wordファイルを選択）　　　　　　　　　　　　　　　・結果（パターンチェックでエラー扱い）

文字列を文字列により分割する（explode関数）



■引数

|  |  |
| --- | --- |
| separator | 区切り文字列。 |
| string | 入力文字列。 |
| limit | int型。省略可。最大要素数。 |

■SampleCode１

**※ファイル種別は $upFile[“name”] に格納される。**

**たとえば、ファイルが「kadai02-product-8-lg.jpeg」であるとき、**

**ファイル名を「.」で分割してみます。**

<?php

　// ～（省略）～

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

　// ～（省略※エラーチェックなど）～

　//拡張子つきの画像ファイル名を「.」前後で分割。ファイル名と拡張子に分ける

$ext = explode(".", $upFile["name"]);

　//確認用に画面に連想配列の値を表示

echo "<pre>";

print\_r($ext);

echo "</pre>";

// ～（省略）～

?>

■Sample画面

・入力（画像ファイルを選択）　　　　　　　　　　　　　　　・結果

■SampleCode２

**※たとえば、ファイル名が「kadai02-product-8-lg.jpeg」であり、**

**ファイル名を「日時.拡張子」に変更したい場合、**

**名前は日時で上書きすればいいが、拡張子はアップロードしたファイルによって変わるため、**

**拡張子のみ別に保存しておき、新たなファイル名の末尾に拡張子を追加する。**

**そうすれば拡張子が強制的に変更されてファイルが破損するなどのミスを防げる。**

<?php

　// ～（省略）～

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

　// ～（省略※エラーチェックなど）～

　//拡張子つきの画像ファイル名を「.」前後で分割。ファイル名と拡張子に分ける

  $reFileName = date("YmdHis");// date関数で日時を取得

  $ext = explode(".", $upFile["name"]); //ファイル名分割

  $ext = $ext[count($ext) - 1]; // 連想配列の末尾＝拡張子のみ取得

　//確認用に画面に連想配列の値を表示

echo "<pre>";

print\_r($reFileName.”.”.$ext); //日付のファイル名と拡張子を文字列連結して表示

echo "</pre>";

// ～（省略）～

?>

■Sample画面

・入力（画像ファイルを選択）　　　　　　　　　　　　　　　・結果

グラフィカル ユーザー インターフェイス, アプリケーション

自動的に生成された説明　