せめて避けたいコードと理由

がる

自己紹介

- エンジニアやってます
- 占い師もやってます
- イベンターもやってます
- Blogとかも書いてます
- よく「……ご職業は？」とか聞かれます(笑

今回はなににこだわるの？

「お仕事として」のプログラミング

- 「よいコード」「美しいコード」の定義は難しい
- とはいえ「明らかによろしくない」コードは、ある
- 流動する「ビジネス」を基点に考察する

基本的な進め方

- 問題のあるコードを、NDAに留意しつつ表示
- どんなところに問題を感じているか？
- 0から書くならどんな風に？
- リファクタせざるを得ないときは？

Globalもどき

```
if($_POST["action"]){$action = $_POST["action"];}elseif($_REQUEST["action"]){$action = $_REQUEST["action"];}elseif($_GET["action"]){$action =
$_GET["action"];}
if($_POST["mode"]){$mode = $_POST["mode"];}elseif($_REQUEST["mode"]){$mode = $_REQUEST["mode"];}elseif($_GET["mode"]){$mode =
$_GET["mode"];}// mode
if($_POST["kw"]){$kw = $_POST["kw"];}elseif($_REQUEST["kw"]){$kw = $_REQUEST["kw"];}elseif($_GET["kw"]){$kw = $_GET["kw"];}// キーワード検索
if($_POST["andor"]){$andor = $_POST["andor"];}elseif($_REQUEST["andor"]){$andor = $_REQUEST["andor"];}elseif($_GET["andor"]){$andor =
$_GET["andor"];}// キーワード検索
if($_POST["pline"]){$pline = $_POST["pline"];}elseif($_REQUEST["pline"]){$pline = $_REQUEST["pline"];}elseif($_GET["pline"]){$pline = $_GET["pline"];}// 改
ページ番号
if($_POST["result_cnt"]){$result_cnt = $_POST["result_cnt"];}elseif($_REQUEST["result_cnt"]){$result_cnt =
$_REQUEST["result_cnt"];}elseif($_GET["result_cnt"]){$result_cnt = $_GET["result_cnt"];}// 結果数
if($_POST["no"]){$no = $_POST["no"];}elseif($_REQUEST["no"]){$no = $_REQUEST["no"];}elseif($_GET["no"]){$no = $_GET["no"];}// 登録番号
if($_POST["co_no"]){$co_no = $_POST["co_no"];}elseif($_REQUEST["co_no"]){$co_no = $_REQUEST["co_no"];}elseif($_GET["co_no"]){$co_no =
$_GET["co_no"];}// 返信コメント用番号
if($_POST["amount"]){$amount = $_POST["amount"];}elseif($_REQUEST["amount"]){$amount = $_REQUEST["amount"];}elseif($_GET["amount"]){$amount =
$_GET["amount"];}// 購入個数
if($_POST["price"]){$price = $_POST["price"];}elseif($_REQUEST["price"]){$price = $_REQUEST["price"];}elseif($_GET["price"]){$price = $_GET["price"];}// 商
品単価
if($_POST["mid"]){$mid = $_POST["mid"];}elseif($_REQUEST["mid"]){$mid = $_REQUEST["mid"];}elseif($_GET["mid"]){$mid = $_GET["mid"];}
 ー中略ー
// テキスト
$text01 = $_POST["text01"];          $text02 = $_POST["text02"];          $text03 = $_POST["text03"];          $text04 = $_POST["text04"];
          $text05 = $_POST["text05"];
$text06 = $_POST["text06"];          $text07 = $_POST["text07"];          $text08 = $_POST["text08"];          $text09 = $_POST["text09"];
          $text10 = $_POST["text10"];
 ー中略ー
// テキストエリア
$message01 = $_POST["message01"];          $message02 = $_POST["message02"];          $message03 = $_POST["message03"];
          $message04 = $_POST["message04"];          $message05 = $_POST["message05"];
$message06 = $_POST["message06"];          $message07 = $_POST["message07"];          $message08 = $_POST["message08"];
          $message09 = $_POST["message09"];          $message10 = $_POST["message10"];
 ー中略ー
// 画像
$upfile01_name = $_FILES["upfile01"]["name"]; $upfile01 = $_FILES["upfile01"]["tmp_name"];          $delupfile01 = $_POST["delupfile01"];  $upfile01_type =
$_FILES["upfile01"]["type"];
$upfile02_name = $_FILES["upfile02"]["name"]; $upfile02 = $_FILES["upfile02"]["tmp_name"];          $delupfile02 = $_POST["delupfile02"];  $upfile02_type =
$_FILES["upfile02"]["type"];
 ー中略ー
if($_POST["a"]){$a = $_POST["a"];}elseif($_REQUEST["a"]){$a = $_REQUEST["a"];}elseif($_GET["a"]){$a = $_GET["a"];}
if($_POST["b"]){$b = $_POST["b"];}elseif($_REQUEST["b"]){$b = $_REQUEST["b"];}elseif($_GET["b"]){$b = $_GET["b"];}
if($_POST["c"]){$c = $_POST["c"];}elseif($_REQUEST["c"]){$c = $_REQUEST["c"];}elseif($_GET["c"]){$c = $_GET["c"];}
```

で…

- 何がまずい？
  - 割とPHP固有な発想(一部BASIC系の人にもある)
  - 厳密にはワーニングが出る記法(存在チェック)
  - register_globalsを意識している
  - 約300変数を毎回：メモリは？
- 新規なら？
  - 普通に「使う時に必要な情報」をゲトる
- リファクタの時は？
  - こまめに頑張る(苦笑)。目印やExcelを併用すると楽

「勤勉である」という怠惰

```
// ▼表示項目の設定
$this->get_conv()->monoDic("customer_id", security::sanitize_html($customer_data["customer_id"]));
$this->get_conv()->monoDic("name1", security::sanitize_html($customer_data["name1"]));
$this->get_conv()->monoDic("name2", security::sanitize_html($customer_data["name2"]));
$this->get_conv()->monoDic("name1_kana", security::sanitize_html($customer_data["name1_kana"]));
$this->get_conv()->monoDic("name2_kana", security::sanitize_html($customer_data["name2_kana"]));
$this->get_conv()->monoDic("email", security::sanitize_html($customer_data["email"]));
$this->get_conv()->monoDic("uid", security::sanitize_html($customer_data["uid"]));
$this->get_conv()->monoDic("sex_name", security::sanitize_html($sex_name));
$this->get_conv()->monoDic("birthday", security::sanitize_html($customer_data["birthday"]));
$this->get_conv()->monoDic("birthday_year", security::sanitize_html($birthday_year));
$this->get_conv()->monoDic("birthday_month", security::sanitize_html($birthday_month));
$this->get_conv()->monoDic("birthday_day", security::sanitize_html($birthday_day));
$this->get_conv()->monoDic("zip1", security::sanitize_html($customer_data["zip1"]));
$this->get_conv()->monoDic("zip2", security::sanitize_html($customer_data["zip2"]));
$this->get_conv()->monoDic("prefecture_name", security::sanitize_html($prefecture_name));
$this->get_conv()->monoDic("city", security::sanitize_html($customer_data["city"]));
$this->get_conv()->monoDic("address", security::sanitize_html($customer_data["address"]));
$this->get_conv()->monoDic("building", security::sanitize_html($customer_data["building"]));
$this->get_conv()->monoDic("point", security::sanitize_html($customer_data["point"]));
$this->get_conv()->monoDic("temp_point", security::sanitize_html($customer_data["temp_point"]));
$this->get_conv()->monoDic("memo", security::sanitize_html($customer_data["memo"]));
```

で…

- 何がまずい？
  - これ単体だと「微妙」なのだけど出来ればまとめたい(実際には、そっくりなコピペがソース内で4箇所ほど……
- 新規なら？
  - 配列でiteratorとかメソッド切り出すとかもう少しなにか
- リファクタの時は？
  - 最低限「そっくりなコピペ」は一箇所に集約する…ための場所をまず確保して、そこから、修正
  - その後で、その一箇所を適宜修正する

基本的なWeb系セキュリティ

```
/*              no              */
$no = $_REQUEST['no'];
/*       シリアルID  */
$serial_id = $_REQUEST['serial_id'];

- 中略 -
$query = "SELECT * FROM gyoumu_lot_usr WHERE 1nenbun_no = '".$no."'
AND serial_id = '".$serial_id."' AND adview_flg = '0'";
$sel->query($query);

- 中略 -
$url  = "http://foo.com/sub_victory.php?no=".$no."&serial_id=".$serial_id."";
```

で…

- 何がまずい？
  - PHP的には$REQUEST_も少々危険
  - SQL InjectionとXSSがナチュラルに存在
- 新規なら？
  - ちゃんとエスケープ処理用の集約点を作る＆する
- リファクタの時は？
  - まず「危ない箇所」を洗い出す(grep可能なコメント
  - (場当たりだけど)各個撃破。XSSなどは「テンプレートエンジン」使ってないと特に大変だけど、やる！！
  - せめて「関数化」で集約点つくるか、本質的には、全体的に設計をやりなおす

「ノーアイデア」ぢゃ困ります

- 「DBにクレジットカード情報を生入力」
- 糅てて加えて「管理画面でCSVダウンロード可」
- さらに「管理画面はBASIC認証のみ」
- トドメが「誰もログ監視その他をしていない」

で…

- 何がまずい？
  - BASIC認証は「生パスワードが毎回パケット上」
  - 漏れても気づけない orz
- 新規なら？
  - 素直にちゃんと「まともな」認証機構を(PHPのsession関数群は……
  - DBにクレジットカード情報なんか持たせない
- リファクタの時は？
  - とりあえずせめて音速でhttps＋Digest認証
  - パスワードも一端変更、かつ当面は「定期的にこまめに」変更
  - CSVダウンロード付近ふくめ「ログをとる」＆「ログをこまめに監視」
  - っつか出来れば当面は「アラートmail」をCSVダウンロードに
  - 近日「設計を根底から見直す」
  - DBデータの暗号化もある程度有効
  - ってか「情報を持たないですむ」方法を考える

鵜呑みにしないっていうか読め

参考サイト
http://www.stackasterisk.jp/tech/php/php03_06.jsp
＞＞
また、ここのページに書かれている内容はセッションハイジャックに対しては**無力です**。ここで言うセッションハイジャックとは、セッションIDを直接指定してアクセスする方法です。クラッカーは存在するであろうセッションIDを予想する／あてずっぽうで指定する、もしくは通信経路の途中で傍受する、ブラウザのキャッシュやクッキーなどを見る、などの方法でセッションIDを指定します。これをやられると、**セッション内部のページに簡単にアクセスできるほか、個人情報の漏洩、サイトの性質によっては金額的な損害も発生することがあります**。ここに書くことは、それらに対しても一切の責任を負わないことをご了承ください。
＜＜

で…

- 何がまずいか？
  - お金が絡むサイトですから。まずいなんてもんぢゃ…
- 新規なら？
  - 理解してから設計をやり直す
  - 「本来的には」言語やframeworkで用意されてるclassを用いるとよい……事が、多い、かも
- リファクタの時は？
  - いやなんていうか…説得して理解させるしか…
  - 説得できたら「処理を一箇所にまとめて」「改築」
  - 無理なら……「逃げて～」

熟考するか、でなければ考えるな

```
<input type="hidden" name="item0" value="氏名">
<input type="hidden" name="item0_check_need" value="1">
<input type="hidden" name="realname" value="item0">
<input type="hidden" name="item0_check_max" value="50">
<input type="hidden" name="item1" value="メールアドレス">
<input type="hidden" name="item1_check_need" value="1">
<input type="hidden" name="email" value="item1">
<input type="hidden" name="item1_check_max" value="255">
<input type="hidden" name="item1_check_valid" value="101">
<input type="hidden" name="item2" value="会社名">
<input type="hidden" name="item2_check_need" value="0">
<input type="hidden" name="item2_check_max" value="255">
<input type="hidden" name="item2_check_valid" value="0">
<input type="hidden" name="item2_check_invalid" value="">
<input type="hidden" name="item3" value="役職・担当">
<input type="hidden" name="item3_check_need" value="0">
```

その2

<input type="hidden" **name="recipient" value="local-part@domain-part"**>
<input type="hidden" **name="subject" value="ニュースレター配信申込み"**>
<input type="hidden" name="headermess" value="この度はニュースレター配信の登録を頂き誠にありがとうございました。
ニュースレターは毎月初めに配信されています。
どうぞお楽しみに！">
<input type="hidden" name="footermess" value="なお、ご登録頂きましたメールアドレスへ
自動返信メールをお送りしておりますので
ご確認頂けますよう、お願い申し上げます。

2日経っても自動返信メールが届いていない場合には
ご登録いただいたメールアドレスが間違えている可能性がございます。
その際にはお手数ですが、再度ご登録下さいますよう
宜しくお願い致します。">
<input type="hidden" name="return_link_title" value="トップページへ戻る">
<input type="hidden" **name="return_link_url"**
value="http://www.domain.com/reform/">

その3

<INPUT type="hidden" **name="table" value="user"**>
<INPUT type="hidden" **name="column" value="id, password, name"**>
<INPUT type="hidden" **name="where" value="id='明らかにID'"**>

その４

```
/************************************************************
*　　インジェクション対策
************************************************************/
function replaceData4SQL($data, $class){

	switch($class){
		case 1:// html
			$data = str_replace("<", "&lt;", $data);
			$data = str_replace(">", "&gt;", $data);
		break;
		case 2:// sql
			$data = str_replace("'", "''", $data);
			$data = str_replace("¥¥", "¥¥¥¥", $data);
			$data = str_replace(";", "", $data);
		break;
		case 3:// html + sql
			$data = str_replace("'", "''", $data);
			$data = str_replace("¥¥", "¥¥¥¥", $data);
			$data = str_replace(";", "", $data);

			$data = str_replace("<", "&lt;", $data);
			$data = str_replace(">", "&gt;", $data);
		break;
	}

	return $data;

}// function
```

で…

- 何がまずい？
  - header Injectionとかその他山盛り
  - spamの踏み台も簡単 orz
- 新規なら？
  - 設定なら、config系のファイルなどに
  - データの保持なら、セッション情報に
  - エスケープは「出力の直前」に「出力にあわせて」
- リファクタの時は？
  - まず説得と教育と理解
  - config、セッション情報などは「一点に集約」してから

総論

- むやみに勤勉なのはよくない(怠惰は美徳)
- 常に「集中点」を作る(がるパタ「随所で関所」
- 認証などの基本的な機能は下手に自作しない
- 自作するのであれば。アイデアはちゃんと基礎を踏まえつつ、有識者に相談してみる。特に「アタックしてくるクラッカーの視点」は重要
- ビジネスは水物。ある程度の頻度と速度と量の「修正」は、それを前提に(っていう角度の問題コード、今回出せなかったなぁ…)

おまけ:リファクタする時の手順一案

- ディレクトリを切って、空ファイルを作成し、ここを「共通関数置き場」にする。
- Documentor系(PHPDocとか)の準備をする
- 告知と周知と、場合によっては鞭と飴で理解させる
- まず「コピペ部分を潰す」&「コメントを書く」&「日記を書く」&「これらを教育、周知徹底」
- 手が付けにくい部分は「ログを仕込んで」「監視」
- 密結合をゆっくり粗結合にしてから、次を考える