

ふつうのpixivのつくりかた

An introduction to building pixiv for ordinary PHPers



pixiv Inc.
USAMI Kenta

pixiv

2023-07-26
PHP の今を語る！4社合同勉強会

お前誰よ



- うさみけんた (@tadsan) / Zonu.EXE / にゃんだーすわん
- ピクシブ株式会社 pixiv事業本部 Webエンジニアリングチーム PHPer
 - 2012年末から現職、APIとかCIとかいろいろなところを見つめてきました
 - 最近ではピクシブ百科事典(dic.pixiv.net)も開発しています
- Emacs PHP Modeを開発しています(2017年-)
- プログラミング言語にちょっとこだわりのある素人



PHPカンファレンス沖縄2023



LLイベント

Learn Languages



Learn Languages 2023



About

Learn Languagesは様々なプログラミング言語について学びたい技術者のためのイベントです。2003年からほぼ毎年開催しています。

([Open Developers Conference \(ODC\) 2023](#) のトラックとして開催します)

URL <https://ll.jus.or.jp/2023/>

ハッシュタグ [#ll2023jp](#)



Session

Perl, PHP, Python, Rubyの20年とこれから

2003年に本イベントの第1回で扱ったプログラミング言語はPerl, PHP, Python, Rubyの4つでした。

今回は再びこれら4つの言語を取り上げ、今までの20年を振り返りこれからの展望を語っていただきたいと思います。

タイムテーブル:

13:00 - 13:20 Perl
13:25 - 13:45 PHP
14:00 - 14:20 Python
14:25 - 14:45 Ruby
15:00 - 15:15 この20年を振り返る座談会



History

このイベントは2003年にLL Saturdayとして開始しました。LLはLightweight Languageの略で、LL Saturdayではスクリプト言語であるPerl, PHP, Python, Rubyについて取り上げました。

2017年には様々な言語を学ぼうという主旨で、Learn Languagesという名称にリニューアルしました。2020年は新型コロナウイルス感染拡大のため開催を断念しました。

今までのアーカイブは以下を参照してください。

[Learn Languages 2022](#)
[Learn Languages 2021](#)
[Learn Languages 2019](#)

pixivの紹介

pixiv.netとは

- “好き”に出会えるイラスト・マンガ・小説作品の投稿プラットフォーム
- 2007年9月10日に開始
 - 累計登録ユーザー数: 約9300万ID
 - 累計作品投稿作品数: 約1.2億作品（うち小説1900万作品）
 - イラストブックマーク: 約200億件
- Web・スマートフォンアプリで展開

ThePHPF

pixiv inside
pixivのなかの話



つくる



はたらく



ささえる

🔍 記事を検索

[トップ](#) > [ささえる](#) > The PHP Foundationへの寄付を開始しました



The PHP Foundationへの寄付を開始しました



Article by tadsan 2022-12-26

こんにちは、うさみ(@tadsan)です。標記の通り、ピクシブ株式会社はThe PHP Foundation([@ThePHPF](#))への継続的な財政支援を開始しました。

OSPO

General / ... / エンジニア職 / オープンソース互助会(#t-oss/pixiv ...)

Edited just now +15 Share



オープンソース互助会(#t-oss/pixiv OSPO)

社内でオープンソースをよりよく活用していく活動を行う技術互助会です。

世の中一般的な言い方としては、OpenOffice Program Office(OSPO)という分野の活動のため、対外的にはこの取組はpixiv OSPOと呼称していきます。

- [#t-oss](#)

📖 ギャラリービュー 📅 Calendar

議事録

ここまでのあらすじ

- 前回: [OSS互助会 @May 12, 2023](#)

最近Slackで話題になったこと

ここまでのあらすじ

- 前回: [OSS互助会 @June 9, 2023](#)

最近Slackで話題になったこと

CLAとは

多くの人の貢献によって成り立っている、フリーなコードやドキュメントの著作権をうまく扱う方法は3つあります。ひとつめは、(私はお勧めしません)が、著作権にまつわる問題をすべて無視して、

ここまでのあらすじ

@April 7, 2023

@nontan が libwebp というGoogleが開発しているOSS プロダクトへのパッチ提出を提案

ふつう #とは

フレームワークが
使われていない

ただのPHP

※諸説あります

諸説ありますが
難しい構造には
なっていない

メタプログラミングは
基本的に
あまりやらない

(こっそり使ってる
部分もありますが
大筋は追えるはず)

PHP詳しくない人でも
雰囲気に従って書けば
セキュリティの問題は
起こさないように

pixivのPHP

- PHPはApache+mod_php（一時fpmを使っていたこともある）
 - nginxからリバースプロキシしてApacheにリクエスト
 - PHPそのものは独自でビルドしてはいるがプレーン
- DocumentRootにある.phpファイルから処理を辿っていけば全部わかる
- LinuxとかMySQLとかよくあるLAMPスタックに載っている

よくある質問

- フレームワークは使っていないの？
 - 使ってません。マイクロフレームワークのようなものはいくつかある
 - ベンリユーティリティの集合でフレームワークに似ているが実態はラッパー
- フレームワーク／言語の移行やフルスクラッチしないの…？
 - 直近での予定はありません。過去に痛い目にも遭ってるし

10年前のコードのイメージ

```
<?php // www.pixiv.net/htdocs/hoge.php
require_once __DIR__ . '/../inc/bootstrap.php';
include_once INC_PATH . '/Hoge/Fuga.php';
include_once INC_PATH . '/Hoge/Piyo.class.php';
try {
    display()
} catch (Exception $e) {
    error::exception_error($e);
}
```

include_onceいっぱい

エラーハンドリング
(してないページもあった)

```
function display() {
    // ...
```

ファイルローカルな
グローバル関数

この下にもいっぱい

2015年のコードのイメージ

```
<?php // www.pixiv.net/htdocs/hoge.php  
require_once __DIR__ . '/../inc/bootstrap.php';
```

include_once 一個だけ

```
AppRunner::execute(new Wwv_HogeController);
```

このクラスは
自動ロードされる

2018年のコードのイメージ

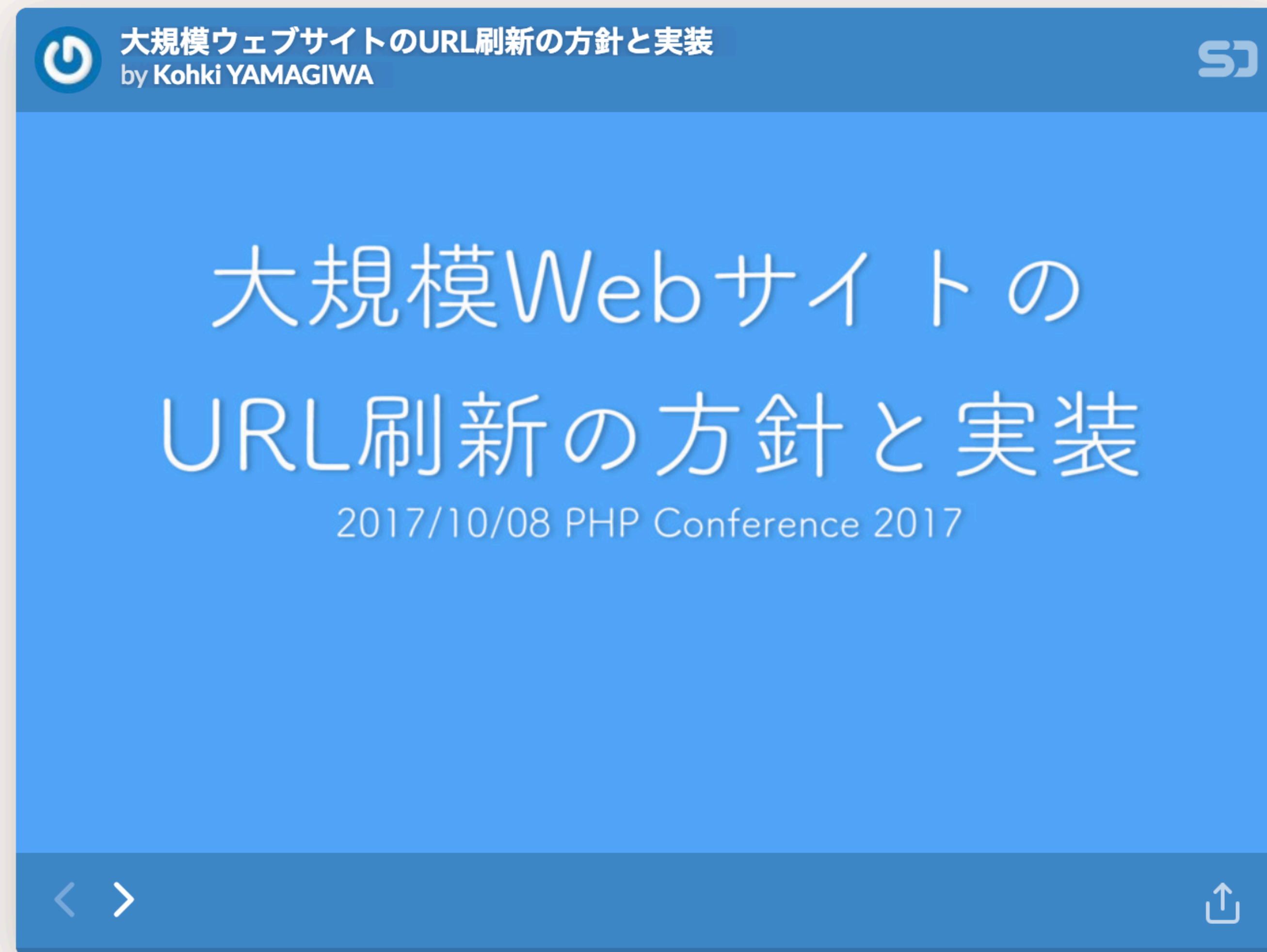
```
final class Controller_WwwRoutes {  
    public static function getRoutes() {  
        $route_map = [  
            '/' => [  
                'action' => function () {  
                    Wwv_IndexController::main();  
                },  
            ],  
            '/hoge.php' => [  
                'action' => function () {  
                    Wwv_HogeController::main();  
                },  
            ],  
        ],  
    }  
}
```

htdocs/hoge.phpは消滅

URLとファイルは切り離され
マップの一要素に

全URLが1ファイルに

技術基盤チームの山際はWebサイトのURL設計と刷新の方針について話しました。



pixivは2007年から開始されたWebサービスです。イラストを投稿するサービスという点では現在まで一貫していますが、現在までの間には大きくの変化がありました。発表された当初のURL設計と現在のページ構成には乖離があります。

PHPカンファレンス2017で発表

2023年こうなりたい

```
<?php // WwwRoutes.php
public static function getRoutes() {
    return [
        '/' => Www_View_IndexController::main(...),
        '/hoge' => Www_View_HogeController::main(...),
        '/.well-known/change-password' => fn() =>
            Util_Http::redirect_and_exit(
                ReverseRoute::fullAccountsPasswordChange([])
            ),
    ],
}
```

... ですっきり

コントローラをdispatchする処理

```
<?php
```

```
final class PCAppRunner {  
    public static function execute(Closure $action) {  
        try {  
            Controller_Util::turnOnWhoops();  
            Controller_Util::redirectToHttps();  
  
            $action();  
        } catch (\Throwable $exception) {  
            PCAppRunner::setHttpStatus(500);  
            Controller_Util::displayWhoops($exception);  
        }  
    }  
}
```

whoops

パソコン版

開発しやすさの ための取り組み

CI (Jenkins, GitLab CI)

- GitLabにブランチをpush、MergeRequestを作るとCIが走る
- PHPUnit、PHPStanと独自の正規表現ベースのlinterでチェック
 - PHPStanは2018年から運用開始
- 時間帯にもよるけどpushして10分かからずに結果がコメントされる

PHPで開発しにくいところ

- 外部入力(クエリパラメータ・フォーム・JSON)の取得・検証
- PDOの機能の貧弱さ
- テンプレートエンジンとURLの問題
- RailsやLaravelにあるようなカッコイイ機能がない
 - かわいいエラー表示
 - 対話環境/REPL (rails console)

クエリパラメータの問題

```
$id = $_GET['user_id'];  
if (is_numeric($id)) User_Common::getId($id);  
else error("不正な入力です");
```

- こういうコードを書いてはいけない
 - クエリパラメータに数字ではない値が入ってくる可能性
 - 入力が空、入力が任意の文字列、入力が不正な数字列、入力が配列
filter_input() をかけると安全にはなるが、それでも面倒

ParamHelper

```
$id = ParamHelper::get('id')->asPositiveInt();  
$user = User_Common::getById($id);
```

- 外部入力から値を取り出すヘルパー
 - クエリパラメータ・フォーム・URL・JSONなどに対応・拡張可能
- 未入力や不正な入力があると例外を投げる
- キャッチしてエラー画面を描画する

ParamHelper

```
$mode = ParamHelper::get('mode')->asStringInArray['hoge', 'fuga'];
```

- ?mode=hoge または ?mode=fuga のみを期待するようなパターン
- 不正な形式や ?mode=piyo のような期待しない入力で例外を投げる
- PHPStan拡張で 'hoge'|'fuga' という型をつけている
- 型安全と実行時安全の両立！（PHPカンファレンス沖縄で話します）

テンプレートとURL生成の問題

```
<a href="{ $smarty.const.SYSTEM_URL_WWW|escape}member_illust.php?id={ $user.id|escape}">{ $user.name|escape}</a>
```

- こういうコードを書くと事故が入り込みやすい
 - 純粋にtypoの危険性
 - idなどのパラメータに不正な値を入れるリスク

→ ReverseRoute

```
<a href="{reverse_route page='fullWwwMemberProfile' id=$user.id}">
{$user.name|escape}</a>
```

- あるページに名前をつけて、reverse_route関数に page 引数で渡す
- 生成結果は変更前と同じ
- ルーティングの逆関数にあたるので、一部のフレームワークは ReverseRoutingやURLヘルパーなどの名前をサポートしてる

→ ReverseRoute

```
/**
 * @route\example https://www.pixiv.net/member.php?id=12345 {id: 12345}
 * @route\example https://www.pixiv.net/member.php?id=12345&utm_source=xxxxxx
{id: 12345, utm_source: "xxxxxx"}
 */
public static function fullWwwMemberProfile(array $params)
{
    Util_Assert::num($params['id']);

    return ReverseRoute::buildUrl(SYSTEM_URL_WWW, '/member.php', ['id'],
$params);
}
```

→ Attribute最高！！

```
#[RouteExample('https://www.pixiv.net/member.php?id=12345', ['id'=> 12345])
#[RouteExample('https://www.pixiv.net/member.php?id=12345&utm_source=xxxxx',
['id' => 12345, 'utm_source' => 'xxxxx'])
public static function fullWwwMemberProfile(array $params)
{
    Util_Assert::num($params['id']);

    return ReverseRoute::buildUrl(SYSTEM_URL_WWW, '/member.php', ['id'],
$params);
}
```

pixiv inside [archive]



pixiv insideは移転しました！ » <https://inside.pixiv.blog/>

2016-10-25

DocCommentでPHPのユニットテストの書きやすさを劇的に改善する手法

44

B!ブックマーク

0

シェア

ツイート

こんにちは、pixivでPHPをやってる@tadsanです。好きなテストフレームワークは [PHPUnit](#) です！好きな某CALOIDは[テトさん](#)です！

みなさまはユニットテストを書いていますか？今回はDocCommentとPHPUnitのプロバイダーをうまく利用してテストの記述を省力化する手法を紹介いたします。

pixiv insideとは

<https://inside.pixiv.blog/> に移転しました。こちらは2016年末までのアーカイブです。

+ 読者になる

221

月別アーカイブ

▼ 2016 (52)

2016 / 12 (28)

2016 / 11 (6)

2016 / 10 (1)

blogに書きました

<https://devpixiv.hatenablog.com/entry/2016/10/25/093000>

PDOの問題

- SQLは自動生成じゃなくて手で書きたい…
でもPDOで複雑なクエリを書こうとすると文字列結合が避けられない…
- PDOで書きにくいクエリの代表例: id IN (1, 2, 3)
- もう ? を並べてインデックスをインクリメントしながらbindValueはやだ…
 - PDOのクエリのplaceholderに ? を書くか :hoge で書くか問題
 - 型チェックできるように書くのもちょっとめんどう
 - だからテンプレートエンジンを作ろう → PxxvSql

PxvSql

```
/**
 * @param int[] $novel_ids
 * @param int $limit
 * @param int $offset
 * @param string $order ASC|DESC
 */
public static function getByNovelIds(array $novel_ids, $limit, $offset, $order = 'ASC')
{
    $con = self::getConnection(DBConf::DB_NOVEL, DBConf::REP_READ);
    $stmt = PxvSql::build($con, NovelRankingDao::getByNovelId_sql, [
        ':novel_ids' => $novel_ids,
        ':limit' => $limit,
        ':offset' => $offset,
        ':order' => $order,
    ]);
    $stmt->execute();

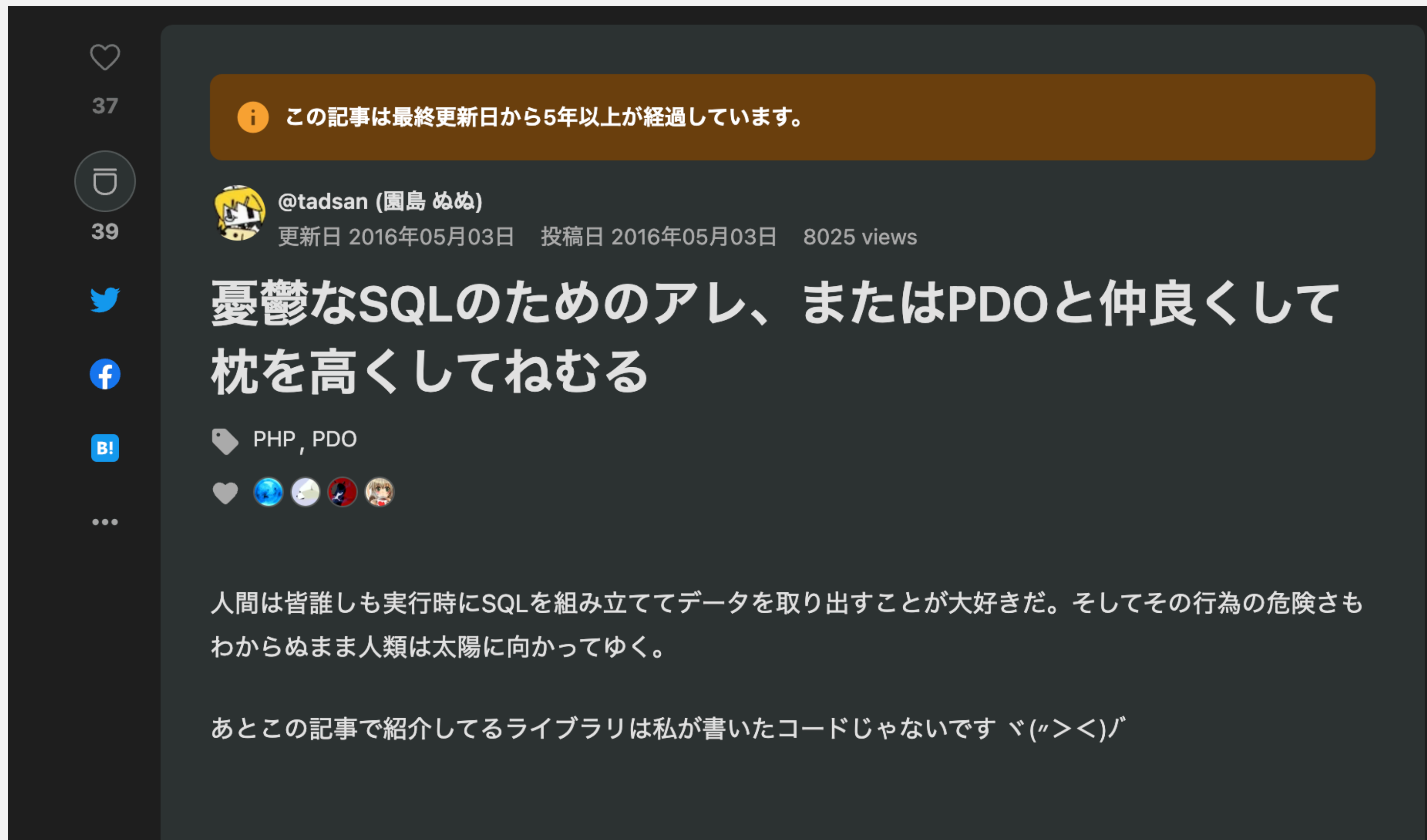
    return $stmt->fetchAll(PDO::FETCH_ASSOC);
}
const getByNovelId_sql = '
    SELECT `novel_id`, `mode`, `date`, `rank`
    FROM `pix_novel_ranking`
    WHERE `novel_id` IN ( :novel_ids@int[] )
    ORDER BY `date` :order@ascdesc
    LIMIT :limit@int
    OFFSET :offset@int
';
```

PxvSql

- 文字列結合を一切やらなくてもクエリが書けるようになった
- 実行時に型検査をするのでnullなどの不穏な値は入り込まない
- エラーチェックのボイラープレートがグッと減る SQLi(脆弱性)のリスクなし
- :hoge@int で整数の埋め込み、:fuga_ids@int[] で複数の整数を展開する %if や %for の記法で条件分岐や複数SETなども可能

PxvSql

- 問題意識はQiitaに書いた



TetoSQL

- tadsanが個人の責任で再実装したバージョン (PHP ≥ 5.4)
- <https://github.com/BaguettePHP/TetoSQL>
- 記法を自由に拡張できるように作られている
- 各社で必要な記法や型を柔軟に実装することも可能！

PDOのラッパー

- これは、少なくとも2011年以前からあった仕組み
 - 書き込み用(master)と読み込み用(slave)を明示的に分けて利用可能
 - slaveにINSERTやUPDATEなどのクエリを発行しようとするとは例外
- 全クエリにSQLのクエリ単一行にして、トレースをコメントにして埋め込む
 - スロークエリのログにどのコントローラに起因するか集計しやすいスロークエリを一覧できるビューアーもある
 - 最近ではDatadogのAPMを使えば不要になるかもしれない

新機能のリリース時にどうする？

- A: 新機能が実装されたブランチをリリースするときマージする
- B: リリース前にあらかじめマージして、実行されないようにしておく
- よくある問題
 - Aは機能の規模が大きいとビッグバンマージを引き起こし、予期せぬ問題を引き起こすことがある（コンフリクト解決が大変）
 - 理想としてはBで、一般ユーザー向けに実行されないとしても鮮度のよいうちにマージしておきたい（リリース時の変更を最小限に）

機能の有効化／無効化

```
if ( $_SERVER[ 'REMOTE_ADDR' ] === OFFICE_IP ) {  
    hogehoge ( ) ;  
}
```

- 新しい機能をサービスに実装して、社内だけで有効化したいとする
- 無造作にこんなコードを書かれると後から意味不明になるそもそも社外と社内での挙動が別物になるので嫌な予感しかない

フィーチャートグル

```
if (ABTest_DevToggle::isEnabledDevToggle('hogeFunc')) {  
    hogehoge();  
}
```

- 専用のコンソール画面でボタンを押すことで有効化／無効化される
- 設定を一行足すだけで一般ユーザーに対してリリースできる
 - リリース後に障害などが発生しないことが確認できたらifを消す
- 元はA/Bテストのための機構だった（一般ユーザーに対して確率で適用）

まとめ

何もない野原に秩序をつくる

- ベんり機能を導入すると、部分的に生産性が体感で数倍になったりする
 - 一からサービスを作るとしたら同じことを繰り返せるかは、悩む
 - いまならLaravelを入れるかもしれないが、それは未来人だからできる発言
 - サービスの価値はコードではなく、その結果のユーザー体験
 - フレームワークがなかったとしても、
pixivが2007年からユーザーに価値を提供してきた結果は変わらない

まとめ

- 今回の内容はPHPカンファレンス関西2018で発表したスライド
 - 2018年以来、良くも悪くもAPIが非常に安定している
- そろそろPHP 8.2に全面移行できるので、次のスタンダードを考えたい
 - AttributesはPHP 8に全面移行できていなくても使えるので先行導入
- pixivで開発者体験と安全性のフォーカスしたアプリケーションをいっしょに開発していきましょう！！！！

続きはWebで (あとWEB+DB PRESS)

- [pixiv FANBOX @tadsan](#)
- [pixiv inside](#)
 - [DocCommentでPHPのユニットテストの書きやすさを劇的に改善する手法](#)
 - [WEB+DB PRESS連載『PHP大規模開発入門』を振り返る](#)
 - [pixivの基盤ノウハウ大公開！PHPカンファレンス2017登壇レポート](#)

続きはWebで (あとWEB+DB PRESS)

- Qiita
 - 憂鬱なSQLのためのアレ、またはPDOと仲良くして枕を高くしてねむる
 - PSRの誤解
 - インスパイヤされて掲示板を作りたくなった シリーズ
 - includeって書きたくない僕たちのためのオートローディングとComposer
- GitHub
 - whoopsやエラー処理の例 <https://github.com/zonuxe/wdb-php-96-sample>

続きはWebで (あとWEB+DB PRESS)



- これ一冊を購入すれば歴代のPHP連載が全部読めるのでおすすめ
- まとめ <https://inside.pixiv.blog/tadsan/3991>
- 今回の内容に近いものだと
 - vol.81, vol.87, vol.91, vol.94, vol. 96
- 買って読んでね！ Webにサンプルコードもあるよ