PHPで PostgreSQLと PGroongaを使って 高速日本語全文検索！

須藤功平 クリアコード

第115回 PHP勉強会@東京
2017-06-28

PostgreSQLと全文検索

- LIKE：組込機能
- textsearch：組込機能
- pg_trgm：標準添付
  - アーカイブには含まれている
  - 別途インストールすれば使える

LIKE

- 少ないデータ
  - 十分実用的
  - 400文字×20万件くらいなら1秒とか
- 少なくないデータ
  - 性能問題アリ

textsearch

- インデックスを作るので速い
- 言語毎にモジュールが必要
  - 英語やフランス語などは組込
  - 日本語は別途必要
- 日本語用モジュール
  - 公式にはメンテナンスされていない
    forkして動くようにしている人はいる

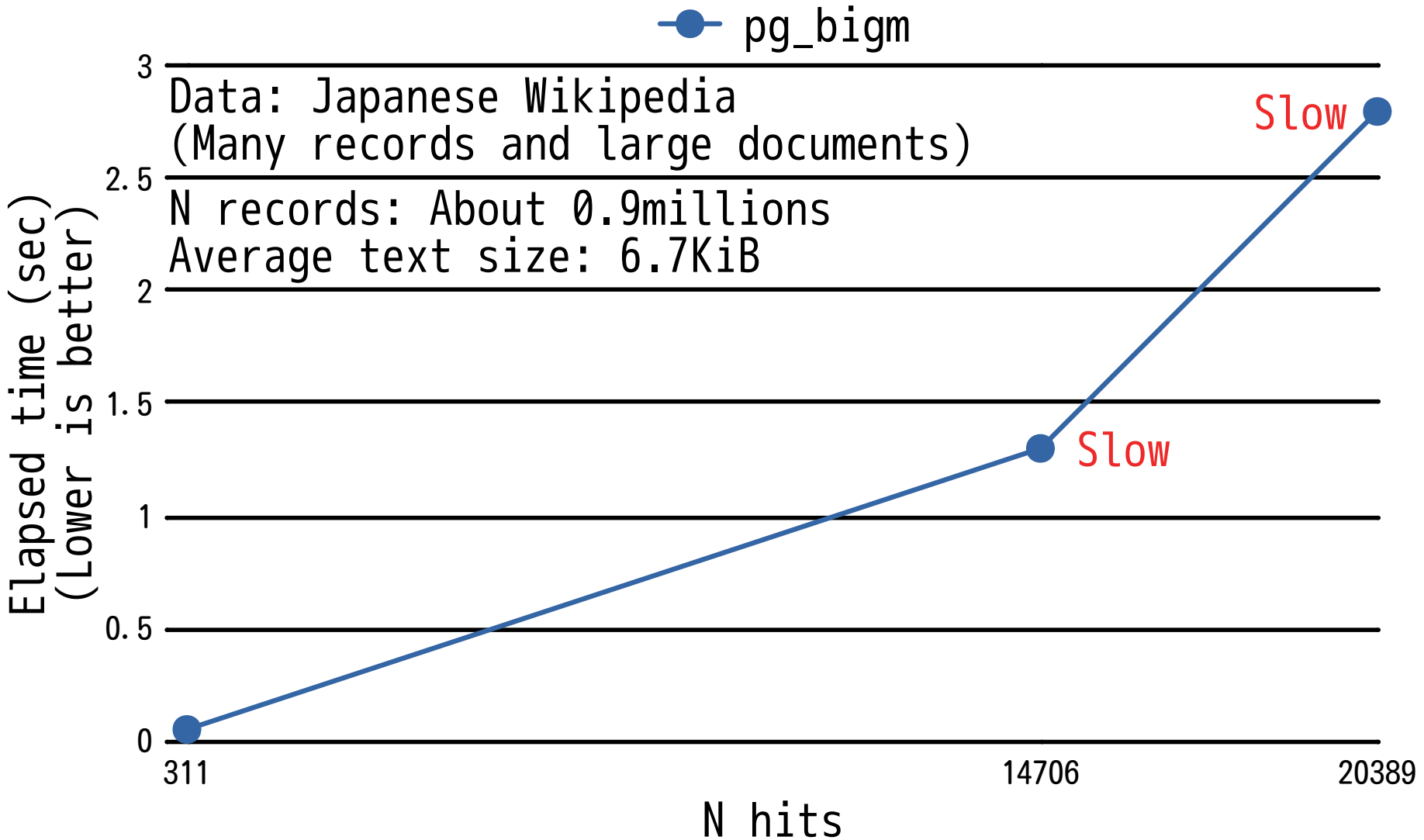
pg_trgm

- インデックスを作るので速い
  - 注：ヒット件数が増えると遅い
  - 注：テキスト量が多いと遅い
  - 注：1,2文字の検索は遅い（米・日本）
- 日本語を使うにはひと工夫必要
  - C.UTF-8を使う
  - ソースを変更してビルド

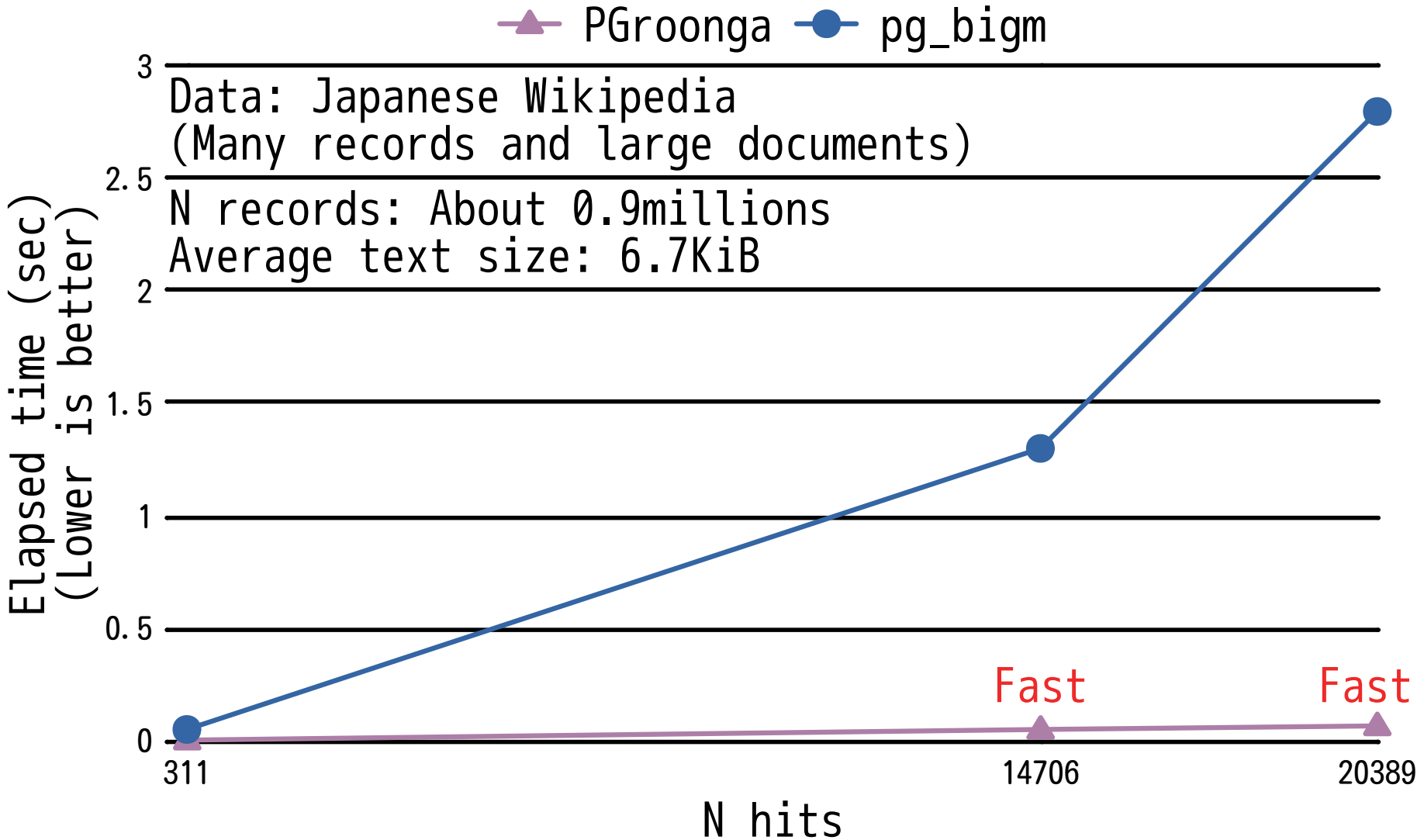
プラグイン

- pg_bigm
  - pg_trgmの日本語対応強化版
- PGroonga
  - 本気の全文検索エンジンを利用
  - 速いし日本語もバッチリ！

ベンチマーク：pg_bigm



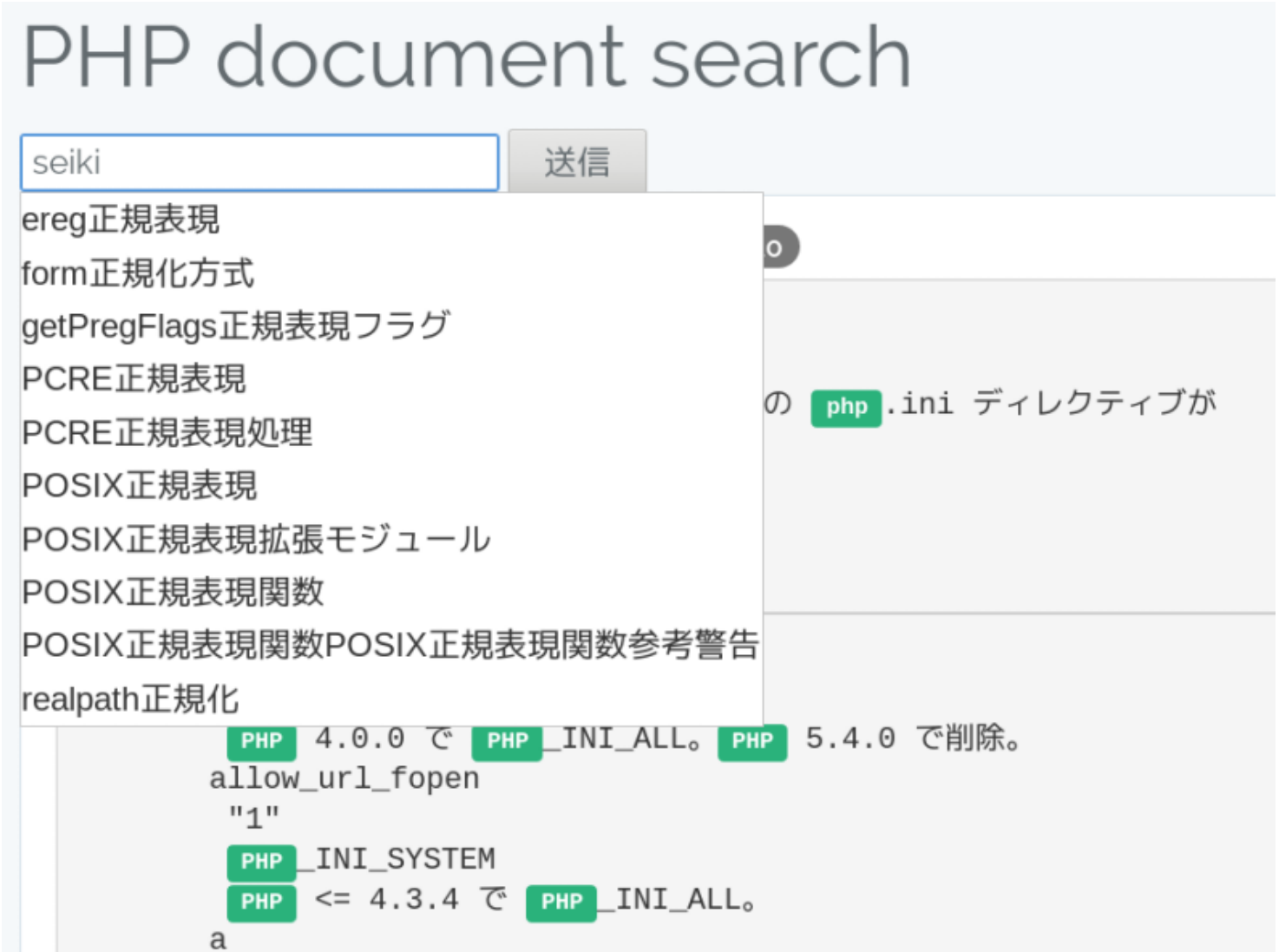
ベンチマーク：PGroonga



よし！

- PostgreSQLとPGroongaを使って
- 高速日本語全文検索サービスを
- PHPで作ろう！

PHP document search



機能

- 検索キーワードハイライト
- キーワード周辺テキスト表示
- オートコンプリート
  - ローマ字対応（seiki→正規表現）

作り方：ツール

- フレームワーク
  - Laravel
- RDBMS
  - PostgreSQL
- 高速日本語全文検索機能
  - PGroonga

作り方：インストール

- Laravel
  - 省略
- PostgreSQL
  - パッケージで
- PGroonga
  - パッケージで
    https://pgroonga.github.io/ja/install/

初期化：Laravel

```
% laravel new php-document-search
% cd php-document-search
% editor .env
```

初期化：データベース

```
% sudo -u postgres -H \
    createdb php_document_search
```

初期化：PGroonga

```
-- ↓を実行する必要がある
CREATE EXTENSION pgroonga;
```

初期化：PGroonga

マイグレーションファイル作成

```
% php artisan \
    make:migration enable_pgroonga
```

マイグレーション

```php
public function up()
{
  DB::statement("CREATE EXTENSION pgroonga;");
}

public function down()
{
  DB::statement("DROP EXTENSION pgroonga;");
}
```

モデル作成

- ドキュメントはモデル
  - 名前：Entry
  - 1ページ1インスタンス

モデル作成

```
% php artisan \
    make:model \
      --migration \
      --controller \
      --resource \
      Entry
```

マイグレーション

```
public function up() {
  Schema::create('entries', function ($table) {
    $table->increments('id');
    table->text('url');
    $table->text('title');
    $table->text('content');
    // PGroonga用インデックス。デフォルトで全文検索用。
    // 主キー（id）も入れるのが大事！スコアー取得に必要。
    $table->index(
       ['id', 'title', 'content'], null, 'pgroonga');
  });
}
```

データ登録

1. PHPのドキュメントを
   ローカルで生成
   - PHPのドキュメントの作り方
     http://doc.php.net/tutorial/
   - フィードバックチャンスがいろいろあったよ！（後述）
2. ページ毎にPostgreSQLに挿入

コマンド作成

```
% php artisan \
    make:command \
      --command=doc:register \
      RegisterDocuments
```

登録コマンド実装（一部）

```
public function handle()
{
  foreach (glob("public/doc/*.html") as $html_path) {
    $document = new \DOMDocument();
    @$document->loadHTMLFile($html_path);
    $xpath = new \DOMXPath($document);
    $entry = new Entry();
    $entry->url = "/doc/" . basename($html_path);
    // XPathでテキスト抽出
    $this->extract_title($entry, $xpath);
    $this->extract_content($entry, $xpath);
    $entry->save();
  }
}
```

登録

```
% php artisan doc:register
```

検索用コントローラー

```
public function index(Request $request)
{
  $query = $request['query'];
  $entries = Entry::query()
    // ↓はモデルに作る（後述）
    ->fullTextSearch($query)
    ->limit(10)
    ->get();
  return view('entry.search.index',
              [
                'entries' => $entries,
                'query' => $query,
              ]);
}
```

検索対象モデル

```
public function
  scopeFullTextSearch($query, $search_query)
{
  if ($search_query) {
    return ...; // クエリーがあったら検索
  } else {
    return ...; // なかったら適当に返す（省略）
  }
}
```

検索対象モデル：検索

```
return $query
  ->select('id', 'url')
  // 適合度をスコアーとして返す
  ->selectRaw('pgroonga.score(entries) AS score')
  // キーワードハイライト
  ->highlightHTML('title', $search_query)
  // キーワード周辺のテキスト（キーワードハイライト付き）
  ->snippetHTML('content', $search_query)
  // タイトルと本文を全文検索（後で補足）
  ->whereRaw('title @@ ? OR content @@ ?',
             [$search_query, $search_query])
  // それっぽい文書の順に返す
  ->orderBy('score', 'DESC');
```

キーワードハイライト

```
public function scopeHighlightHTML($query,
                                   $column,
                                   $search_query)
{
  return $query
    // PGroonga提供ハイライト関数
    ->selectRaw("pgroonga.highlight_html($column, " .
    // PGroonga提供クエリーからキーワードを抽出する関数
                "pgroonga.query_extract_keywords(?)) " .
                "AS highlighted_$column",
                [$search_query]);
}
```

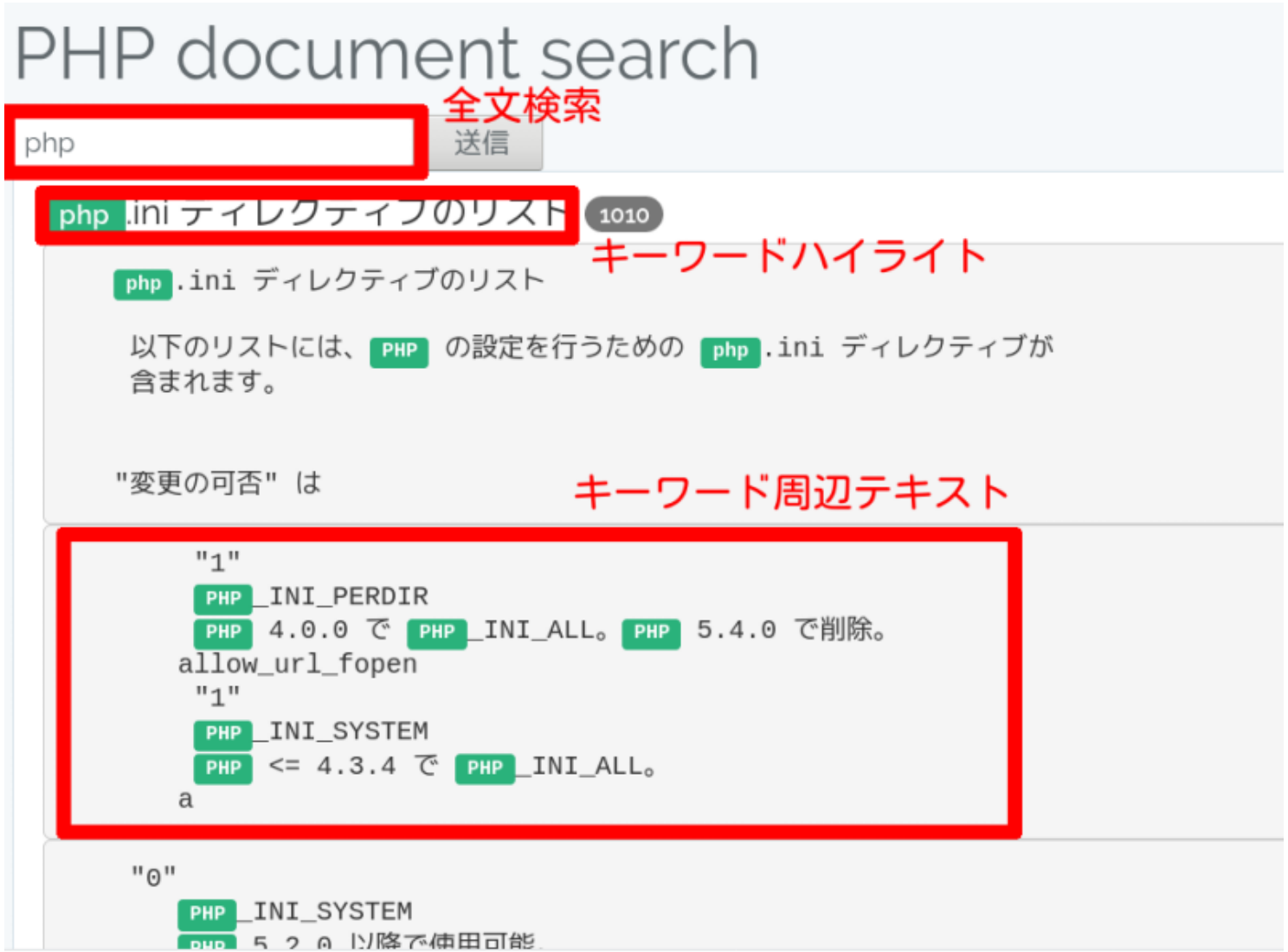
検索結果

```
<div class="entries">
  @foreach ($entries as $entry)
    <a href="{{ $entry->url }}">
      <h4>
        {{-- マークアップ済み！ --}}
        {!! $entry->highlighted_title !!}
        <span class="score">{{ $entry->score }}</span>
      </h4>
      {{-- 周辺テキストはtext[]（後で補足） --}}
      @foreach ($entry->content_snippets as $snippet)
        <pre class="snippet">{!! $snippet !!}</pre>
      @endforeach
    </a>
  @endforeach
</div>
```

検索対象モデル：配列

```
public function getContentSnippetsAttribute($value)
{ // PostgreSQLは配列をサポートしているがPDOは未サポート
  // '["...","..."]'という文字列になるのでそれを配列に変換
  return array_map(
    function ($e) {
      // 「"」が「\"」になっているので戻す
      return preg_replace('/\\\\(.)/', '$1', $e);
    },
    explode('","', substr($value, 2, -2)));
}
```

高速日本語全文検索！


PHP document search
全文検索
php
送信
php.ini ディレクティブのリスト
1010
キーワードハイライト
php.ini ディレクティブのリスト
以下のリストには、PHP の設定を行うための php.ini ディレクティブが含まれます。
"変更の可否" は
キーワード周辺テキスト
"1"
PHP_INI_PERDIR
PHP 4.0.0 で PHP_INI_ALL。PHP 5.4.0 で削除。
allow_url_fopen
"1"
PHP_INI_SYSTEM
PHP <= 4.3.4 で PHP_INI_ALL。
a
"0"
PHP_INI_SYSTEM

オートコンプリート

- 必要なもの
  - 候補用テーブル
  - 候補のヨミガナ（カタカナ）
  - PGroonga!!!

モデル作成

```
% php artisan \
    make:model \
        --migration \
        --controller \
        --resource \
        Term
```

マイグレーション：カラム

```
public function up()
{
  Schema::create('terms', function ($table) {
    $table->increments('id');
    $table->text('term');
    $table->text('label');
    $table->text('reading'); // 本当は配列にしたい
    $table->timestamps();
    // インデックス定義（後述）
  });
}
```

マイグレーション インデックス

```
$table->index([
  // 候補に対する前方一致検索用
  DB::raw('term pgroonga.text_term_search_ops_v2'),
  // ヨミガナに対する前方一致RK検索用
  DB::raw('reading pgroonga.text_term_search_ops_v2'),
], null, 'pgroonga');
// 候補に対する全文検索用（中間一致用）
$table->index([DB::raw('term')], null, 'pgroonga');
```

前方一致RK検索

- 日本語特化の前方一致検索
  - ローマ字・ひらがな・カタカナで
    カタカナを前方一致検索できる
  - gy→ギュウニュウ
  - ぎ→ギュウニュウ
  - ギ→ギュウニュウ

候補モデル：検索

```
public function
  scopeComplete($query, $search_query)
{
  return $query
    ->select("label")
    ->highlightHTML('label', $search_query)
    ->whereRaw("term &^ :query OR " . // 前方一致検索
               "reading &^~ :query OR " . // 前方一致RK検索
               "term @@ :query", // 全文検索
               ["query" => $search_query])
    ->orderBy("label")
    ->limit(10);
}
```

コントローラー

```php
public function index(Request $request)
{
  $query = $request["query"];
  // モデルに実装した検索処理を呼び出し
  $terms = Term::query()->complete($query);
  $data = [];
  foreach ($terms->get() as $term) {
    $data[] = [
      "value" => $term->label,
      "label" => $term->highlighted_label,
    ];
  }
  // JSONで候補を返す
  return response()->json($data);
}
```

UI

```
$('#query').autocomplete({
  source: function(request, response) {
    $.ajax({
      url: "/terms/", // コントローラー呼び出し
      dataType: "json",
      data: {query: this.term},
      success: response
    });
  }
}).autocomplete("instance")._renderItem = function(ul, item) {
  return $("<li>")
    .attr("data-value", item.value) // 候補には生データを使う
    .append(item.label) // ハイライトしたデータを表示
    .appendTo(ul);
};
```

オートコンプリート！

PHP document search

seiki

送信

ereg正規表現
form正規化方式
getPregFlags正規表現フラグ
PCRE正規表現
PCRE正規表現処理
POSIX正規表現
POSIX正規表現拡張モジュール
POSIX正規表現関数
POSIX正規表現関数POSIX正規表現関数参考警告
realpath正規化

の php.ini ディレクティブが

PHP 4.0.0 で PHP_INI_ALL。PHP 5.4.0 で削除。

allow_url_fopen
"1"
PHP_INI_SYSTEM
PHP <= 4.3.4 で PHP_INI_ALL。
a

まとめ

- PGroongaを使えば…
  - 高速日本語全文検索サービスを…
  - PHPで簡単に作れる！
- PHP document searchのソース
  - https://github.com/kou/php-document-search

その他（1）

- PHP+MySQL+Mroongaでも簡単！
- Groongaではじめる全文検索
  - https://grnbook-ja.tumblr.com/
  - 著者：北市真
  - PHP+Mroonga入門の電子書籍
  - 今はまだ無料！

その他（2）

- だれかPHP document searchをメンテナンスしませんか？
  - 普通に便利じゃないかと！
  - 複数バージョン対応とか
  - 複数言語対応とか

その他（3）

- PHPの開発に参加しませんか？
  - PDOのPostgreSQL対応強化とか
  - ドキュメントまわりとか
- やりたいけど自分はムリそう…
  - そんなことはないんですよ！

その他（4）

- OSS Gateワークショップ
  - OSS開発未経験者を経験者にするワークショップ
  - PHPもOSS！
  - 次回は7月29日
    https://oss-gate.doorkeeper.jp/events/upcoming

その他（5）

- PHPカンファレンス2017内で OSS Gateワークショップ開催はどうですか！？
  - PHP関連のOSSの開発に参加する人が増えるとうれしい？
  - うれしいならコラボできそう