【PHP on IIS スペシャルセミナー】
PHP5 時代のアプリケーション開発事情

PHP 5リリースから4年、PHP 5には移行した方が良いのか？
PHP 4→5の移行を検討されている方への注意事項、そろそろ見えてきた次期PHP 6の変更点を解説。
後半では、昨今のPHPアプリケーション開発事情として、統合開発環境、ライブラリ、フレームワークについて紹介します。

山田祥寛 (YAMADA, Yoshihiro)
yoshihiro@wings.msn.to
http://www.wings.msn.to/

WINGS More Progress, More Future

PHP 4、5の現状

- **PHP 5リリース（2005年7月）から4年経ったが…**
  - 2007年末の段階で、PHP 4ベースのサイトが全体の6割以上
  - 新規開発でのPHP 5導入はそれなりに進んだものの…
  - PHP 4で構築された稼働中サイトの移行は不十分
- **PHP 5への移行が思うように進まなかった理由**
  - PHP 5に移行する積極的な理由がない
    → 使いたい機能、ライブラリがどれだけあった？
    → （たとえば）例外処理は、そもそもライブラリが対応しなければ無意味
  - PHP 5は遅い、不安定（という、当初の悪印象）
  - 当初、PHP 5では動作しないアプリケーションもあった（Ex. PukiWikiなど）
  - PHP 4は十分に高機能であった
    → 今問題なく動いているならば、それでいい！的な空気

今こそPHP 5移行のとき

- **PHP 5.0リリースから4年で状況は変化**
  - PHP 5.1：Zend Engine 2の高速化（性能低下の問題は解消済み）
  - PHP 5.2：メモリマネージャの導入（安定性も大幅に向上）
  - データベース抽象化レイヤ、XML系の新機能、JSON関数などの追加
  - PHP 5の機能を活かしたライブラリ／フレームワークも増加（PEAR2、Zend Framework、symfonyなどなど）
  - PHP 5に移行しない理由が見つからない！
- **PHP 4サポートが遂に終了（2007/12/31）**
  - 開発者リソースの主力は既にPHP 6に移行
  - 致命的なセキュリティフィックスの対応だけを2008/8/8まで継続
  - PHP 6の一部機能をバックポートしたPHP 5.3も2008年末頃？にリリース予定
    → PHP 4→ 6に向かうよりも、PHP 5→ 6の方がカンタン

PHP 6の変更点（1）

- **基本的には...**
  - PHP 5との互換性を維持
  - ただし、大きな変更点もあり、プログラミングスタイルには自ずと影響も
- **Unicodeへの本格的な対応**
  - mbstringではなく、PHPコア（内部コード）としてのUnicode対応
  - 文字コード処理には、IBM開発のICU（International Components for Unicode）を利用
  - Unicodeの有効／無効を設定可能に
- **PHP6の新機能**
  - 名前空間のサポート（PHP 5ではいったん実装を見送られたもの）
  - gotoサポート（スパゲッティコードの原因。賛否あり）
  - APC（Alternative PHP Cache）コードキャッシュの標準バンドル
    ※コードキャッシュ：内部構造に変換されたコードを維持して、コードの繰り返し実行を効率化
  - 64ビット整数への対応

PHP 6の変更点（2）

- 既存機能の整理
  - register_globalsサポートの廃止
  - magic_quotes機能の廃止
  - セーフモードのサポートを終了
  - PHP 4（Zend Engine 1）互換モードの削除
    ※オブジェクト指向構文に関わる挙動を強制的にPHP 4互換に
    （zend.ze1_compatibility_mode）
  - 動的メソッド（非staticメソッド）の静的コールを禁止
  - ASPタグ（<%～%>）の廃止
  - PDO以外のデータベース拡張をPECLに移動

PHP 5の変更ポイント（1）

- オブジェクト関連の変更
  - オブジェクトは参照渡しが基本
    （PHP 4では値渡し）

```
class MyClass { public $x; }

$obj1 = new MyClass();
$obj1->x = 10;
$obj2 = $obj1;   // 参照渡し
$obj3 = clone $obj1;  // 値渡し
$obj1->x = 100;
print($obj2->x); // 100
print($obj3->x); // 10
```

  - オブジェクトの比較
    ==演算子：同じクラスのインスタンスで
    同じ属性値を持つか
    ===演算子：同じインスタンスか
    （PHP 4では双方とも同じ挙動）

```
class MyClass { public $x; }

$obj1 = new MyClass();
$obj2 = new MyClass();
var_dump($obj1 == $obj2);   // TRUE
var_dump($obj1 === $obj2); // FALSE
```

PHP 5の変更ポイント (2)

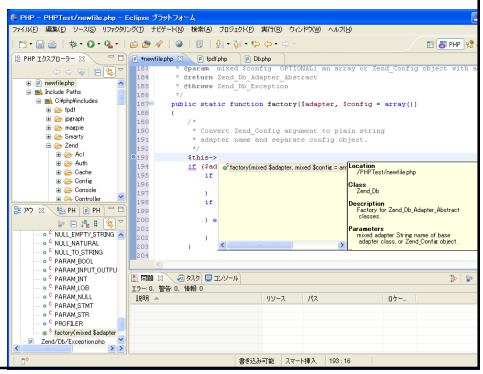
- 関数の変更点
  - get_class、get_parent_class、get_class_methods関数
    → PHP 4はすべて小文字で返す、PHP 5は定義されたままの名前
    ※PHP 5では取得した名前を大文字／小文字に変換するなどの対処
  - strrpos、strripos関数
    → PHP 4では検索文字 (列) を「先頭の1文字」のみ、PHP 5では「文字列」として認識
  - empty関数
    → PHP 4ではプロパティがないクラスを空、PHP 5では空でないと見なす
  - array_merge関数
    → PHP 4では配列以外の引数も認める、PHP 5では配列のみしか受け付けない
  - ip2long関数
    → IPアドレス変換失敗時にPHP 4では-1、PHP 5ではFALSEを返す
  - DOMXML関数 → DOM関数、XSLT関数 → XSL関数

PHP 5の変更ポイント (3)

- その他の追加点
  - オブジェクト指向構文 (コンストラクタ、デストラクタ、アクセス修飾子、抽象メソッド、インターフェイス、final／static修飾子、クラス内定数 (const))
  - マジックメソッド (__call、__get、__set、__autoload、__isset、__unset、__sleep、__wakeup、__toStringなど)
  - Stnadard PHP Class (SPL) 、Reflection APIの追加
  - タイプヒンティングなど
  - 予約語の追加と削除
- PHP 4 → 5の移行は大して難しくない
  - オブジェクト指向構文の変更ポイントと、下位互換性のない関数を中心に確認
    → どうしても移行できない場合は…暫定的な移行モードとしてPHP4互換モード
  - 古い記法をなくしていくことも重要 (クラスと同名のコンストラクタ、非staticメソッドの静的呼び出しなど)
    → エラー表示モードE_STRICT (下位互換のための機能に警告)

統合開発環境 (1)

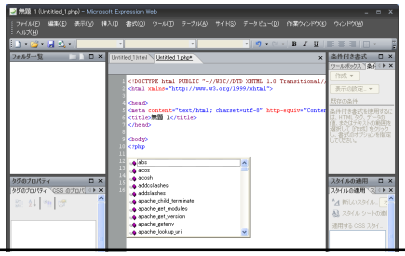
- **PHPで利用可能な統合開発環境も増えてきた**
  - .NET FrameworkでのVisual Studio、JavaでのEclipseなどIDEによる開発は一般的
  - ひと昔前はPHP開発はテキストエディタだった、が…
  - 有償／無償を問わず、さまざまな開発環境がPHPでも利用可能に
- **PHP Development Tool（http://www.eclipse.org/pdt/）**
  - EclipseベースのIDE（無償）
  - コード補完機能からアウトライン表示、ハイライト表示、デバッグ機能など
  - その他にもPHPエディタ（http://phpspot.net/php/）、Aptana Studio（http://www.aptana.com/studio）など



統合開発環境 (2)

- **有償製品ならばまた別の選択肢も**

製品名	概要
Zend Studio for Eclipse	WYSIWYGエディタ、リファクタリング機能、ユニットテスト機能など、Zend社が提供する標準的な開発環境
VS.php	Visual Studioをベースにした開発環境
Expression Web 2	マイクロソフト社提供のWebサイトデザインツール（2よりPHP開発にも対応）



▲Expression Web 2

PHPで利用可能なライブラリ

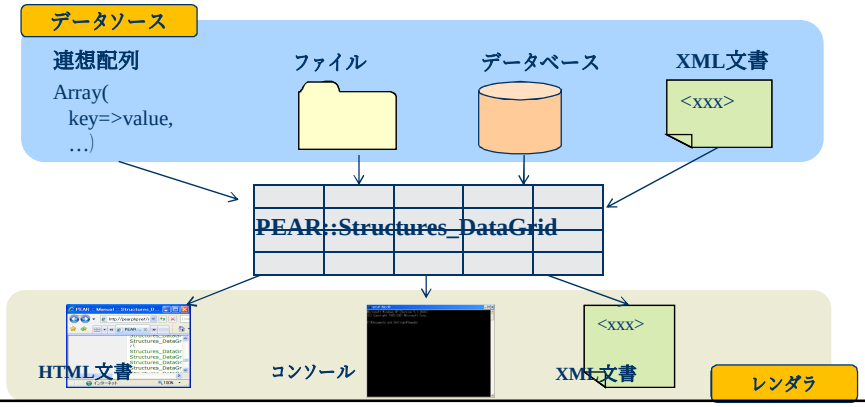
- **外部ライブラリにも目を向けよう！**
  - PHP標準の機能でできることはやはり限られる
    → 特定の目的に特化した機能では積極的にライブラリを提供しよう！
  - ライブラリ＝C言語ベース／PHPスクリプトベースのライブラリに大別できる

拡張ライブラリ
- C言語ベースのライブラリ
  - バンドルエクステンション
  - PECL（PHP Extension Code Library）
  - サードベンダから提供されるライブラリ
- PHPベースのライブラリ
  - PEAR（PHP Extension and Application Repository）
  - サードベンダから提供されるライブラリ

  - 今の主流はPHPスクリプトベースのライブラリ
  - 昨今ではZend Engineの速度向上に伴い、パフォーマンス上の制約が（それなりに）解消
  - ソースコードが確認しやすい、導入の容易さ（ソースコードをコピーするだけ）

PEAR::Structures_DataGrid

- **ASP.NETのGridViewコントロールライクなリッチなグリッド表を生成**
  - ページング、ソート機能つきの一覧表をシンプルなコードで作成
  - データソースやレンダラを自由に選択可能



サンプル） PEAR::Structures_DataGrid

- 「データバインド→カラム設定→レンダラによる出力」のシンプルなコーディング

```
$grid = new Structures_DataGrid(10);  // ページ辺りの行数を設定
            // 結果セットをグリッド表にバインド（データソースは自動認識）
$params = array('dsn' => 'mysqli://phpusr:phppass@localhost/phplibs');
$res = $grid->bind('SELECT isbn, title, price, publish FROM book', $params);
            // グリッドに表示するカラムの定義
$c_title = new Structures_DataGrid_Column('書名', 'title', 'title');
$c_title->setFormatter('createLink()');  // コールバック関数でリンクを生成
$c_price = new Structures_DataGrid_Column('価格', 'price', 'price');
$c_price->setFormatter('createCurrency()');
            ...中略...
 // 生成したカラムをグリッドに追加
$grid->addColumn($c_title);
            ...中略...
$s = new Smarty();
$grid->fill($s);  // グリッド情報を流し込み
$s->display('DataGrid.tpl');
```

書名	価格	出版社
ソースコードリーディングから学ぶJavaの設計と実装	2,604円	技術評論社
今日からつかえるASP.NET 2.0サンプル集	3,360円	秀和システム
Pocket詳解JSP&サーブレット辞典	2,310円	秀和システム
サーバサイドAjax入門	2,940円	翔泳社

1 | 2 | 3 >

FPDF

- フリーで利用可能なPDF生成ライブラリ
  - 画像取り込みや図形描画、表組み、リンクなど基本機能に対応
  - FPDIと連携することで、既存PDF文書をテンプレートとしてインポートが可能に
  - japanese.phpを利用することで日本語にも対応可能
    ※ただし、FPDIを利用する場合はコードの書き換えが必要
  - PDFLib/PDIのような有償ツールに比べると、やや処理速度に劣る
    → 負荷の高いサイトでなければ、それほど問題にはならない
  - 参考）リッチなコンテンツを作成するためのライブラリ
    PEAR::Spreadsheet_Excel_Writer、
    JpGraph、
    QRCode Perl CGI＆PHP scripts、
    PEAR::Image_Barcode、
    PEAR::Text_CAPTCHAなども

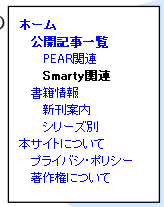
サンプル）FPDF

- 「テンプレートのインポート→適用」までを数行で記述可能

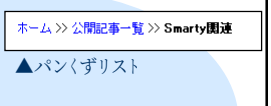
```
$pdf = new PDF_Japanese('P','mm','A4');
$pdf->AddSJISFont();

$pdf->setSourceFile('template.pdf');      // テンプレートの設定
$ipage = $pdf->importPage(1);             // テンプレートのインポート
$pdf->addPage();   // 新規ページの追加
$pdf->useTemplate($ipage, 0, 0);          // テンプレートの適用

$pdf->SetFont('SJIS','B',14);
$pdf->setXY(30,20);
$pdf->cell(80, 12, encode($_POST['item'.$i]), 1, 0, 'L');      // 入力値の出力
...中略...
$pdf->output();      // PDF文書の出力
```

PEAR::HTML_Menu

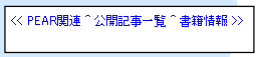
- さまざまな形式のメニューを一元的に管理＆生成
  - パラメータを変更するだけで、ツリー形式、パンくずリスト、前後リンク、サイトマップなどのメニューを自由に切り替え可能
  - メニュー情報はXML形式や連想配列、データベースなどから指定可能
  - 参考）ナビゲーションを作成するためのライブラリ
    PEAR::HTML_TreeMenu、
    PEAR::Pager、
    PEAR::HTML_TagCloudなども



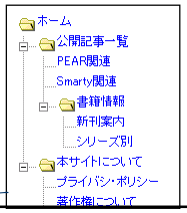
▲サイトメニュー／ツリーメニュー



▲パンくずリスト



▲前後リンク



▲開閉式ツリー

▲タグクラウド

サンプル）PEAR::HTML_Menu（1）

- メニュー情報は連想配列やXML文書、データベースなどで定義可能

```
<?xml version='1.0' encoding='UTF-8' ?>
<sitemap>
  <a1 title="ホーム" url="index.php">
    <sub>
      <a11 title="公開記事一覧" url="article.php">
        <sub>
          <a111 title="PEAR関連" url="article_pear.php" />
          <a112 title="Smarty関連" url="HTML_Menu.php" />
        </sub>
      </a11>
      ...中略...
</sitemap>
```

サンプル）PEAR::HTML_Menu（2）

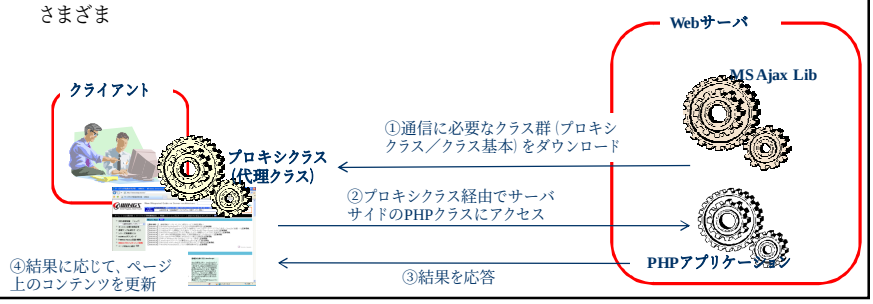
- メニュー情報は連想配列やXML文書、データベースなどで定義可能

```
$xml = new Xml_Unserializer($us_p);
$xml->unserialize('Sitemap.xml');
$data = $xml->getUnserializedData();  // サイトマップ情報を連想配列に変換

$menu = new HTML_Menu($data);      // HTML_Menuオブジェクトを生成
$menu->forceCurrentUrl(basename($_SERVER['SCRIPT_NAME']));
$renderer=new HTML_Menu_DirectRenderer(); // レンダラを生成
$renderer->setMenuTemplate('<table border="0">', '</table>');
$menu->render($renderer, 'tree');          // メニュー形式を指定
print($renderer->toHtml());   // メニューを出力
```

パラメータを変更するだけで出力するメニューを切り替えできる

PHP for Microsoft AJAX Library

- マイクロソフトが提供する標準のAJAXライブラリ（Microsoft AJAX Library）
  - クライアントサイドスクリプトから透過的にサーバクラスにアクセスするためのプロキシ機能を提供
  - PHP for Microsoft AJAX Library＝Microsoft AJAX LibraryをPHP向けに利用しやすく機能追加したラッパー
  - その他にも、PEAR::HTML_AJAX、SAJAX、AJASONなどAJAX対応ライブラリはさまざま



サンプル）PHP for Microsoft AJAX Library

- 特別なコーディングは最小限に、Ajax機能を実装可能

●サーバサイド

```
class MyClass extends MSAjaxService {
 function getInfos($keywd) {
   ...中略...
   return $result;
 }
}
 // プロキシクラス、またはメソッドの結果を生成
$clazz = new MyClass();
$clazz->ProcessRequest();
```

●クライアントサイド

```
<script type="text/javascript"
 src="./lib/MicrosoftAjax.js"></script>
<script type="text/javascript"
 src="MyClass.php/js"></script>
<script type="text/javascript">
function search() {
 MyClass.getInfos(
   $get('keywd').value,
   function(result) {
     // 通信成功時の処理
   });
}
</script>
```

PHPUnit（1）

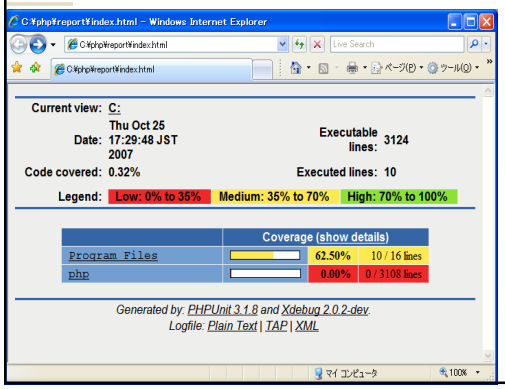
- **JUnitに影響を受けて開発されたテスティングフレームワーク**
  - JUnitによく似た使い勝手で、単体テストが可能
  - 日本語ドキュメントも充実
  - さまざまなケースでのテストに対応

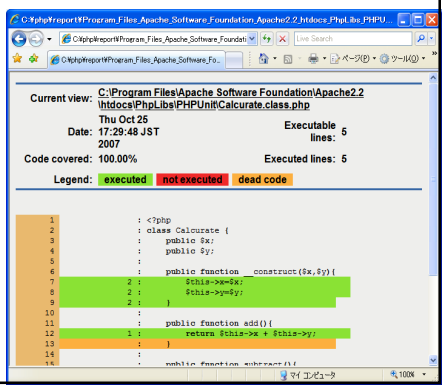
クラス名	概要
PHPUnit_Framework_TestCase	標準的なテストケース（すべてのテストケースの基底クラス）。戻り値の内容を確認
PHPUnit_Extensions_OutputTestCase	print命令による出力結果（出力バッファの内容）をチェック。 シンプルな文字列比較の他、正規表現によるチェックも可能
PHPUnit_Extensions_ExceptionTestCase	特定の処理で任意の例外が発生するかを確認
PHPUnit_Extensions_PerformanceTestCase	特定のメソッドが指定時間内で完了するかを確認（パフォーマンステスト）

  - （当然）複数のテストケースを束ねるテストスイートにも対応

PHPUnit（2）

- **XDebug拡張モジュール導入でコードカバレッジ解析も可能**
  - コードカバレッジ解析＝テストコードがコード全体の何割をカバーしているかを解析
  - 左図：総合レポート　右図：個々のスクリプトに対するカバレッジ解析結果





サンプル）PHPUnit

- JUnitやNUnitと同じ要領でテストコードの記述が可能

```
class CalcurateTest extends PHPUnit_Framework_TestCase {
  ...中略...
  public function setUp(){
    $this->cal = new Calcurate(5,10);      // テスト対象のクラスをインスタンス化
  }
    // Calcurate::addメソッドをテストするコード
  public function testAdd() {
    $result = $this->cal->add();  // addメソッドの実行
    $expected = 15;   // 期待する値
    $this->assertEquals($result, $expected); // $resultと$expectedの比較
  }
}
```

```
> phpunit CalcurateTest CalcurateTest.class.php
PHPUnit 3.1.9 by Sebastian Bergmann.
.F    →「.」は成功、「F」は失敗
Time: 0 seconds
```

アプリケーションフレームワーク（1）

- アプリケーションフレームワークとは？
  - アプリケーション開発のための共通的な土台
  - 汎用的な機能や、（多くの場合は）デザインパターンを提供
- フレームワークの必要性
  - システム要件の高度化
    → 餅は餅屋（それぞれの専門家が分業する必要性）
  - 少ない開発コストと短い納期
  - 日常的な改定要求に耐えられるしくみづくりの必要性
- PHPでも、これまでフレームワークが不要だったわけではないが…
  - 大規模システムにPHPが適用される機会がそれほどは多くなかった（全体からすれば）
  - オブジェクト指向構文の弱さから汎用的なフレームワークを用意しにくかった
  - PHP 5でオブジェクト指向構文が強化されたことで状況が変化しつつある
    ※Lightweight Language人気も要因？

アプリケーションフレームワーク (2)

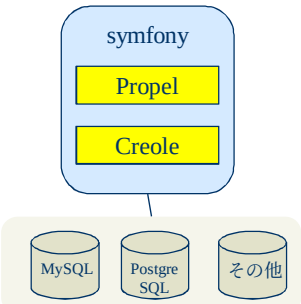
- フレームワーク導入のメリット
  - 開発コード／工数の削減 (問題領域に対する「設計分析」のノウハウを再利用できる)
  - コンカレントな開発でモジュールの品質を均一化できる
  - 保守性の向上 (問題追跡の効率化、修正時の影響箇所の局所化)
- フレームワーク導入のデメリット
  - フレームワークのルールを新たに学ぶ必要がある
  - PHPのフレームワークは百花繚乱の状態 (デファクトスタンダードの不在)

名称	対応	ソース生成	インストール	備考
Zend Framework	5	×	手動	Zend社による開発
symfony	5	○	PEAR	RoRの影響
CakePHP	4、5	○	手動	RoRの影響
Ethna	4、5	○	PEAR	国産。GREEで利用
Mojavi	4、5	×	手動	PHP F/Wの草分け

symfony (http://www.symfony-project.org/)

- Ruby on Railsの影響を色濃く受けたMVCフレームワーク
  - Scaffolding機能によるRAD (Rapid Application Development) 開発が可能
  - O/R (Object Relational) マッピング機能でSQLレスなデータベース開発が可能
    - Propel：O/Rマッパー (JavaのTorqueから移植されたもの)
    - Creole：データベースアクセスライブラリ
  - シンプルなヘルパーを数多く提供 (以下、主要)

名称	概要
Form	フォーム部品を出力
JavaScript	JavaScript／Ajax関連の処理
Cache	キャッシュの制御
I18N	国際化関連の機能を提供
Text	文字列関連の処理

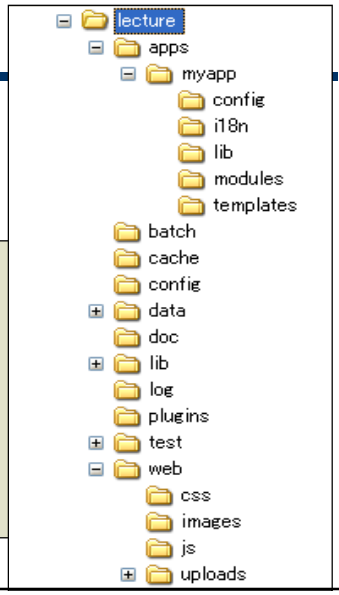


  - スマートURL (設定ファイルでURLの形式を操作可能)
  - 柔軟な拡張性と高い移植性

symfony（Scaffolding機能（1））

- コマンド実行だけで最低限の環境を構築
  - symfonyアプリケーションの基本構造は「プロジェクト」「アプリケーション」「モジュール」
  - symfonyコマンドでテンプレートを自動生成

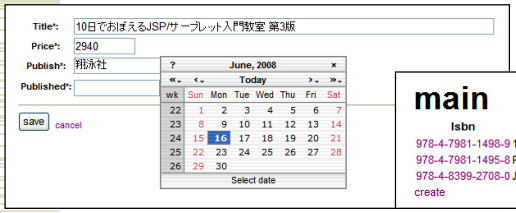
```
> pear channel-discover pear.symfony-project.com
  // PEARチャネルの設定
> pear install symfony/symfony
  // symfonyのインストール
> mkdir lecture  // lectureフォルダの作成
> cd lecture  // カレントフォルダの移動
> symfony init-project lecture
  // lectureプロジェクトの構築
> symfony init-app myapp
  // myappアプリケーションの構築
```



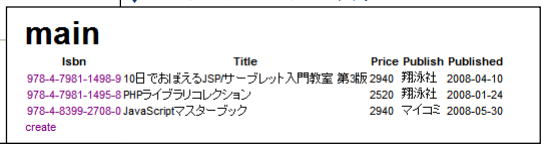
symfony（Scaffolding機能（2））

- symfonyコマンドでCRUD（Create/Read/Update/Delete）も自動生成
  - データベースのメタデータからスキーマ情報の取得、モデル、基本ビューの生成まで
  - スキーマファイルからデータベースの自動構築も可能

```
> symfony propel-build-schema  // スキーマ情報の取得
> symfony propel-build-model  // スキーマファイル→モデルの生成
> symfony propel-generate-crud myapp main Book
  // CRUDの自動生成（Bookクラスをもとに、myappアプリ－mainモジュールを生成）
> symfony cc  // キャッシュの消去
```



← 個別データの編集画面
↓bookテーブルのリスト表示



symfony（YAML）

- **symfonyの設定ファイルはYAML形式が基本**
  - YAML：YAML Ain‘t Markup Languageの再帰的略語
  - 記号とインデントで階層構造を表現
  - 設定ファイルで表現できるものはすべて設定ファイルで
    → ただし、規約で設定できるものは規約で（Convention rather than configuration）

```
# database.yml
all:
  propel:
    class: sfPropelDatabase
    param:
      phptype:  mysql
      host:     localhost
      database: test
      username: root
      password: 12345
```

```
# schema.yml
propel:
  _attributes:
    package: lib.model
  book:
    isbn:
      type: VARCHAR
      size: 50
      required: true
      primaryKey: true   ...後略...
```

symfony（Ajax対応）

- **JavaScriptライブラリであるprototype.js、script.aculo.usを統合**
  - JavaScript／AjaxヘルパーでJavaScriptを隠蔽（JavaScriptレスでAjax機能を実装）
    → Ex. ページの一部分をサーバからの処理に基づいて更新するコード

```
<?php print javascript_tag(
  remote_function(
    array(
      ‘update’ => ‘result’,   // 反映先の要素
      ‘url’ => ‘mymod/myact‘   // 呼び出し先のURL
)) ) ?>
```

  - その他にもさまざまなAjaxヘルパーが存在（主なものは以下）

ヘルパー名	概要
link_to_remote	リンク操作によるページの部分更新
observe_form／observe_field	フォーム（または要素）が更新されたタイミングでのAjax通信
sortable_element	ソート可能なリスト
input_auto_complete_tag	オートインクリメント機能を持つ入力ボックス

Zend Framework（http://framework.zend.com/）

- 互いに疎結合なコンポーネントが特徴のフレームワーク
  - Zend Technorogies社による開発（継続的なメンテナンスが期待できる）
  - symfonyは一体型のフレームワーク（個々のコンポーネントとしては使いにくい）
    ⇔ Zend Frameworkは個々のコンポーネントを単体のライブラリとして利用可能
    → Zend_Controller（フレームワーク機能）を利用しなくても構わない
  - PEAR的なライブラリ集（ただし、体系的に整理され、連携も企図されている）
  - 外部ライブラリに依存しない、BSDライセンス（再配布の容易さ）
    → ライセンス問題が発生にくい商用システムでの利用が容易
  - 豊富なコンポーネント（以下、主要抜粋。全部で約50種類）

名称	概要	名称	概要
Zend_Acl	アクセス制御	Zend_Db	データベース抽象化レイヤ
Zend_Auth	ユーザ認証	Zend_Form	検証機能付きフォーム
Zend_Cache	データ／ページキャッシュ	Zend_Layout	ツーステップビュー
Zend_Controller	MVCコントローラ	Zend_Log	ロギング
Zend_[Translate	ローカライズ機能	Zend_Rest	REST式Web APIへのアクセス

Zend Framework（Zend_Db）

- 高速かつ高機能なデータベース抽象化レイヤ
  - DB抽象化レイヤとは… データベースの仕様差を吸収するためのライブラリ
  - PHP 5.1～には標準的なPDO（PHP Data Objects）はあるものの、機能は少なめ
  - PEAR::MDB2は高機能だが、低速なのがネック
    Ex. 1000件のデータを繰り返し挿入／取得した場合のベンチマークテスト
    PDO：3秒 MDB2：15秒（Zend_Db：7秒）
  - そこで、Zend_Db。PEAR::MDB2よりも高速、PDOよりも高機能
    → SELECT命令の動的生成やプロファイリング機能、O/Rマッピング

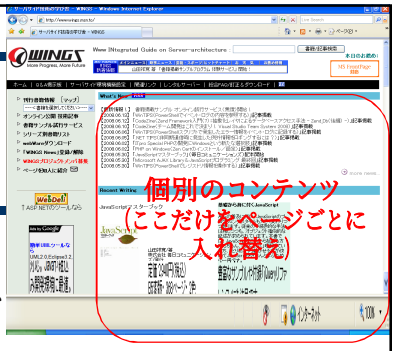
▼SELECT命令の動的生成

```
$sel = $db->select();
$sel->from('book', array('isbn', 'title', 'price'))
  ->joinInner('author', 'book.author_id
          = author.author_id', array('name'))
  ->order('published');
```

▼O/Rマッピング機能を利用した行の挿入

```
$bok = new Book();
$row = $bok->createRow();
$row->isbn = '978-4-8399-2708-0';
$row->title = 'JavaScriptマスターブック';
$row->save();
```

Zend Framework（Zend_Layout）



- ツーステップビューパターン
  - マスターなるテンプレート（マスタページ）の中に個別のテンプレート（コンテンツページ）を埋めこむためのしくみ

▼レイアウト機能を有効化

```
$layout = Zend_Layout::startMvc(
  array(
    'layoutPath' => './application/smarty/templates',
    'layout' => 'master',
    'contentKey' => 'content')
);
$layout->setViewSuffix('tpl');
```

▼マスターページ（master.tpl）

```
<body>
<img src="../image/wings.jpg" />
<hr />
{$content|smarty:nodefaults}
</body>
```

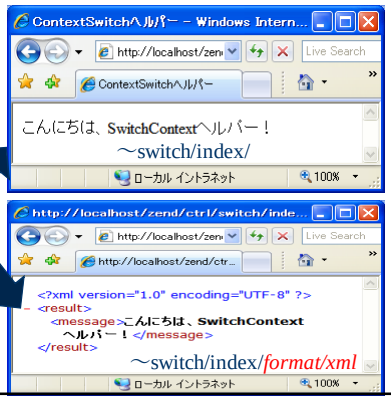
↑ 個別ページのコンテンツを埋込

- 余談）Zend Frameworkではビュー機能が弱いといわれるが…
  Smartyのような外部テンプレートエンジンを組み込むことも容易（もちろん、他のも）

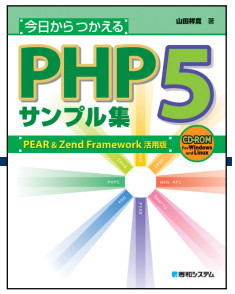
Zend Framework（ContextSwitchヘルパー）

- クエリ情報の切り替えで応答フォーマットを動的に切り替えるしくみ
  - 同一のコンテンツをHTML、XML、JSONなど形式だけを切り替えて出力したい！
  - アクションを個別に用意することなく、ビューだけを切り替えられる仕組み
    → フォーマットごとにテンプレートを用意すればOK！

```
public function init() {
  $context = $this->_helper->ContextSwitch;
  $context->addActionContext('index', array('xml', 'json')) // indexアクションの出力形式を指定
        ->initContext();
}
  // コンテンツを出力するためのindexアクション
public function indexAction() {
  $this->view->message = 'こんにちは、～！';
}
```



PHP関連書籍＆サイト

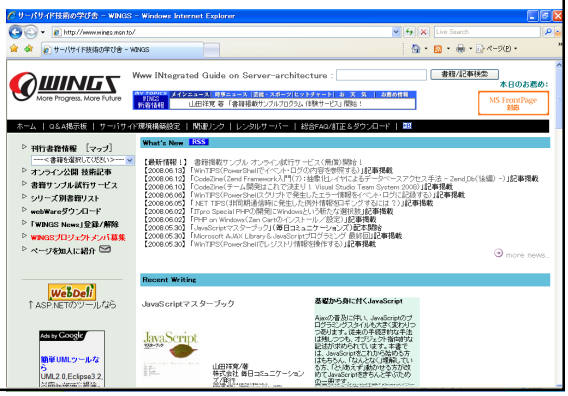


- クラスライブラリ／フレームワーク活用のための書籍
  - 今日からつかえるPHP5サンプル集
    PEAR＆Zend Framework活用版（秀和システム）
    ※*2008年6月26日（水）*刊行予定
  - PHPライブラリコレクション（翔泳社）
  - PEAR入門 – PHP標準ライブラリを極める！ -（翔泳社）
  - Smarty入門 - PHP＋テンプレート･エンジンでつくるMVCアプリケーション-（翔泳社）

- PEAR／フレームワーク関連のサイト記事
  - ［連載］Zend Framework入門
    （http://codezine.jp/a/article/aid/1824.aspx ）
  - ［連載］symfony入門
    （ http://codezine.jp/a/article/aid/704.aspx ）
  - ［連載］PEARライブラリ活用
    （ http://codezine.jp/a/article/aid/1816.aspx ）

ご静聴ありがとうございました

- 「サーバサイド技術の学び舎 - WINGS」
  - http://www.wings.msn.to/
  - サーバサイド技術に関する記事や書籍、関連サイトなど最新の情報を日々紹介
  - RSSフィードも配信中



サーバサイド技術の学び舎 - WINGS