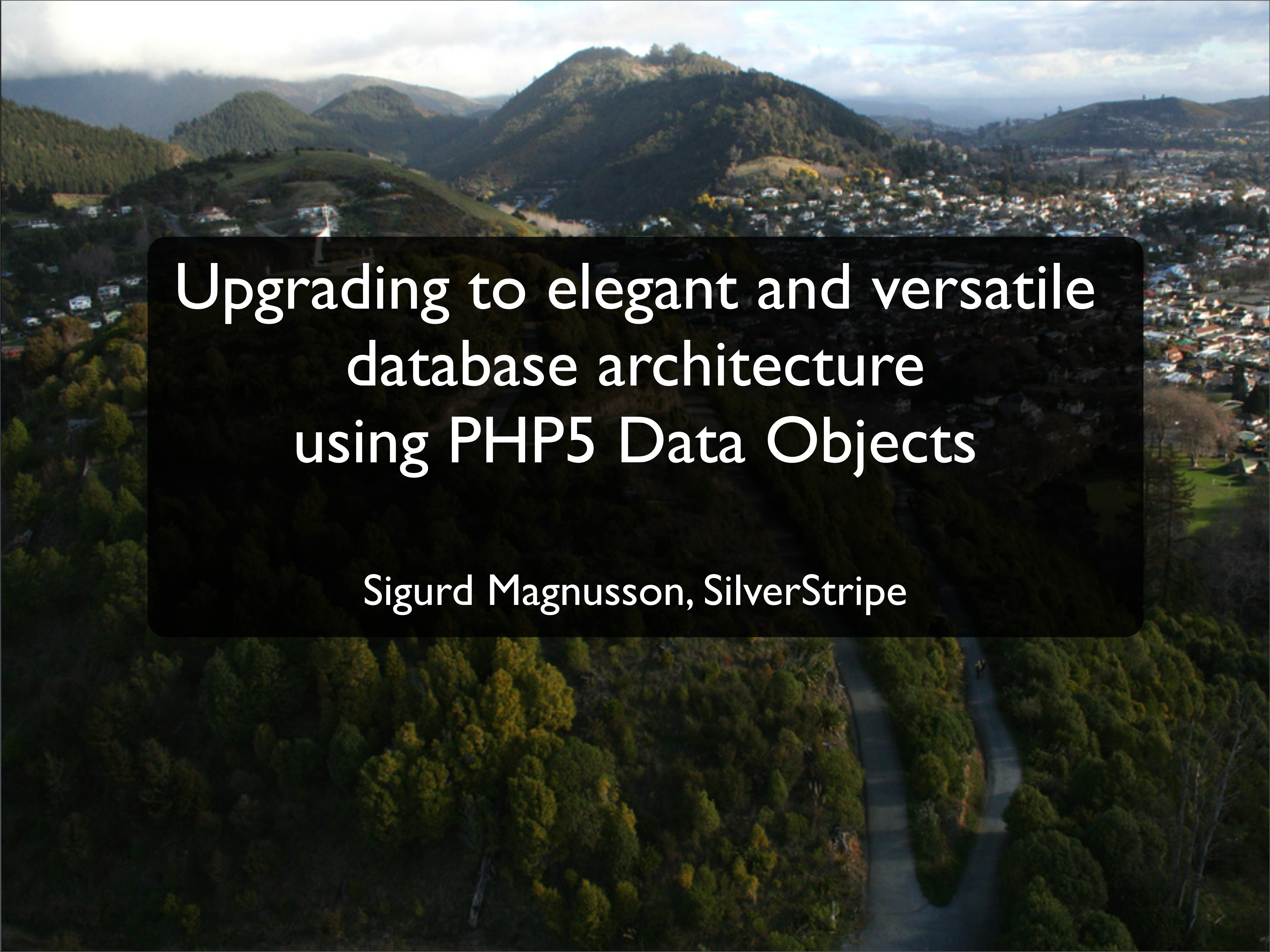
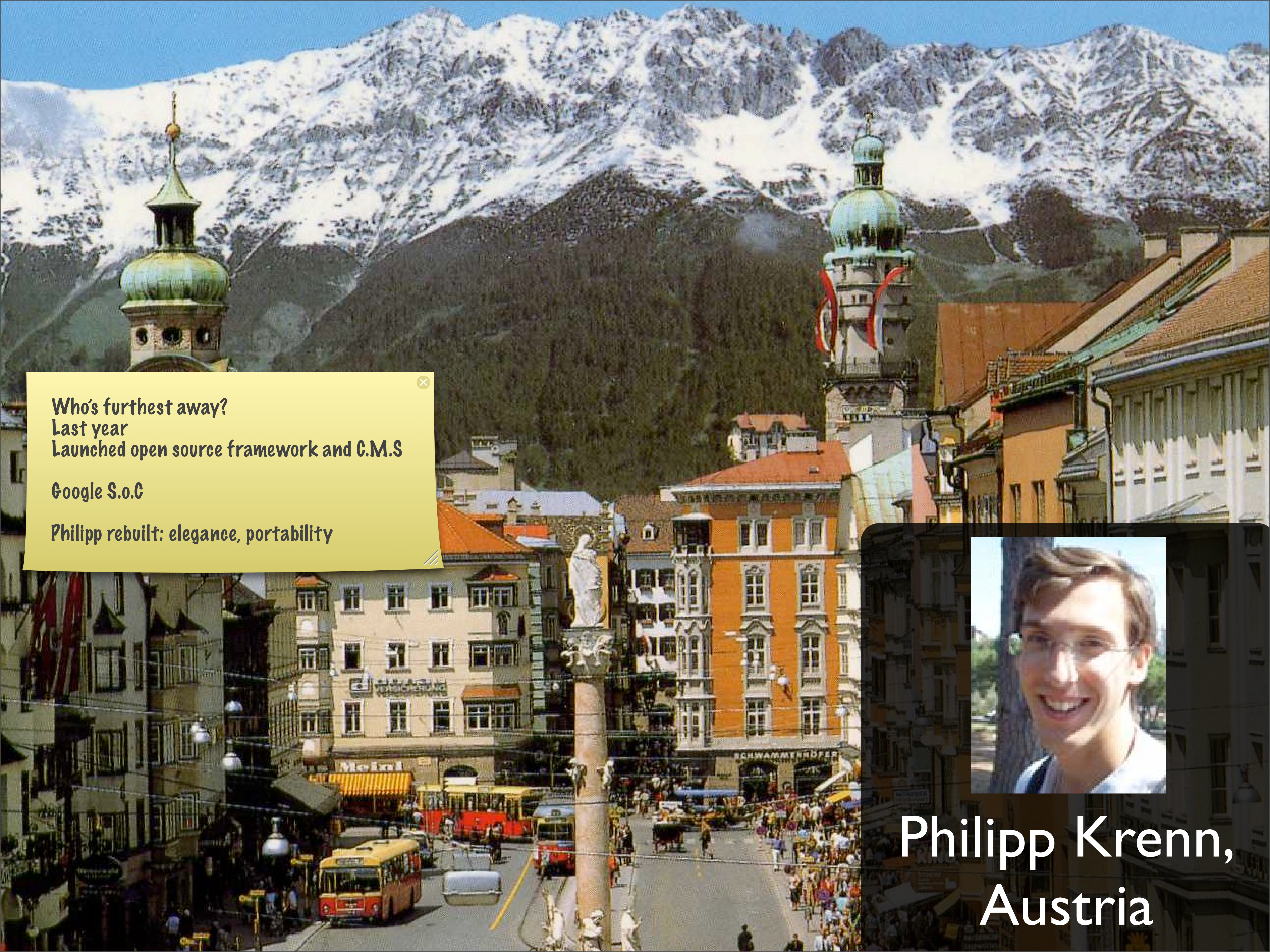


Upgrading to elegant and versatile database architecture using PHP5 Data Objects

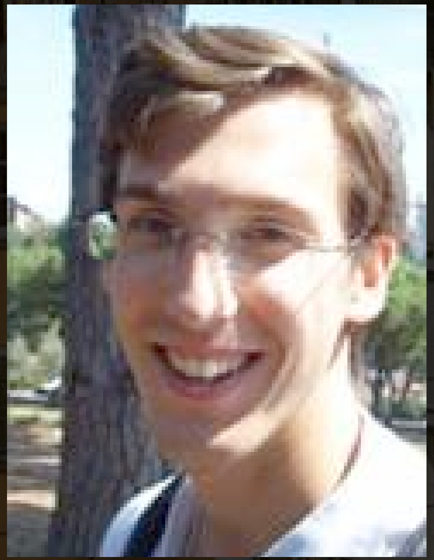
Sigurd Magnusson, SilverStripe



Who's furthest away?
Last year
Launched open source framework and C.M.S

Google S.o.C

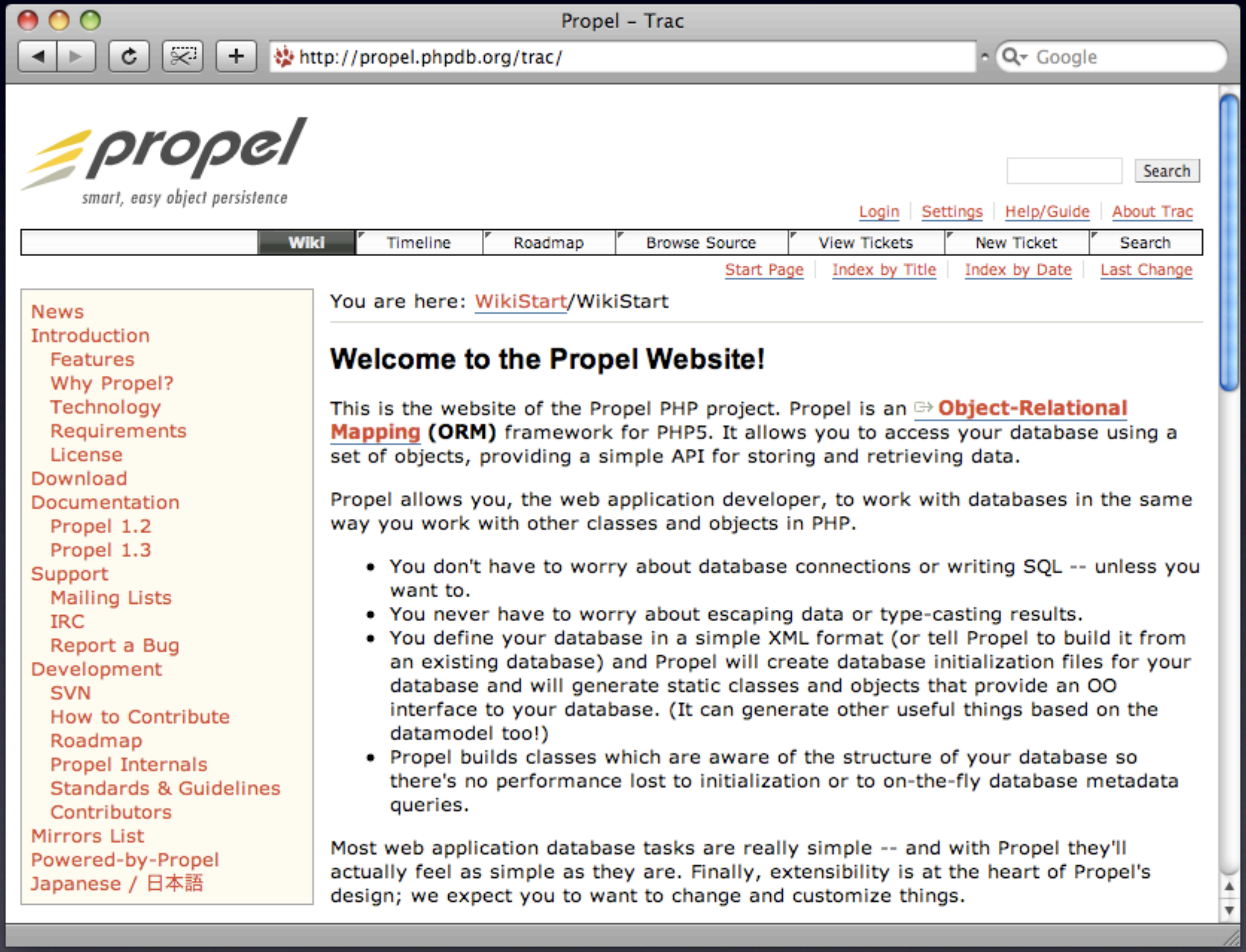
Philipp rebuilt: elegance, portability



Philipp Krenn,
Austria

What existed already?

Propel (and Creole)



You are here: WikiStart/WikiStart

Welcome to the Propel Website!

This is the website of the Propel PHP project. Propel is an **Object-Relational Mapping (ORM)** framework for PHP5. It allows you to access your database using a set of objects, providing a simple API for storing and retrieving data.

Propel allows you, the web application developer, to work with databases in the same way you work with other classes and objects in PHP.

- You don't have to worry about database connections or writing SQL -- unless you want to.
- You never have to worry about escaping data or type-casting results.
- You define your database in a simple XML format (or tell Propel to build it from an existing database) and Propel will create database initialization files for your database and will generate static classes and objects that provide an OO interface to your database. (It can generate other useful things based on the datamodel too!)
- Propel builds classes which are aware of the structure of your database so there's no performance lost to initialization or to on-the-fly database metadata queries.

Most web application database tasks are really simple -- and with Propel they'll actually feel as simple as they are. Finally, extensibility is at the heart of Propel's design; we expect you to want to change and customize things.

Propel and Creole

Write schema.xml files to build database and ORM.

```
<database>
    <table name="author" description="Author Table">
        <column name="author_id" type="integer" required="true" primaryKey="true"
             autoIncrement="true" description="Author Id"/>
        <column name="first_name" type="varchar" size="128" required="true"
             description="First Name"/>
        <column name="last_name" type="varchar" size="128" required="true"
            description="Last Name"/>
    </table>
</database>
```

Propel and Creole

PHP objects automatically created to manipulate database

```
<?php
// INSERT INTO author (first_name, last_name) VALUES ('Sigurd', 'Magnusson');

// ... initialize Propel ...

$author = new Author();

$author->setFirstName("Sigurd");
$author->setLastName("Magnusson");

$author->save();
?>
```

Propel and Creole

Gets complex with WHERE, JOIN, GROUP BY, ORDER, etc.

```
<?php
// SELECT * FROM author WHERE first_name = 'Sigurd’
                             AND last_name <> 'Magnussen';

// ... initialize Propel ...

$c = new Criteria();

$c->add(AuthorPeer::first_name "Sigurd");

$c->add(AuthorPeer::last_name, "Magnussen", Criteria::NOT_EQUAL);

$authors = AuthorPeer::doSelect($c);  // returns array of Author objects
?>
```

Propel for us...

PHP5 only, OO, exceptions, unit tests...

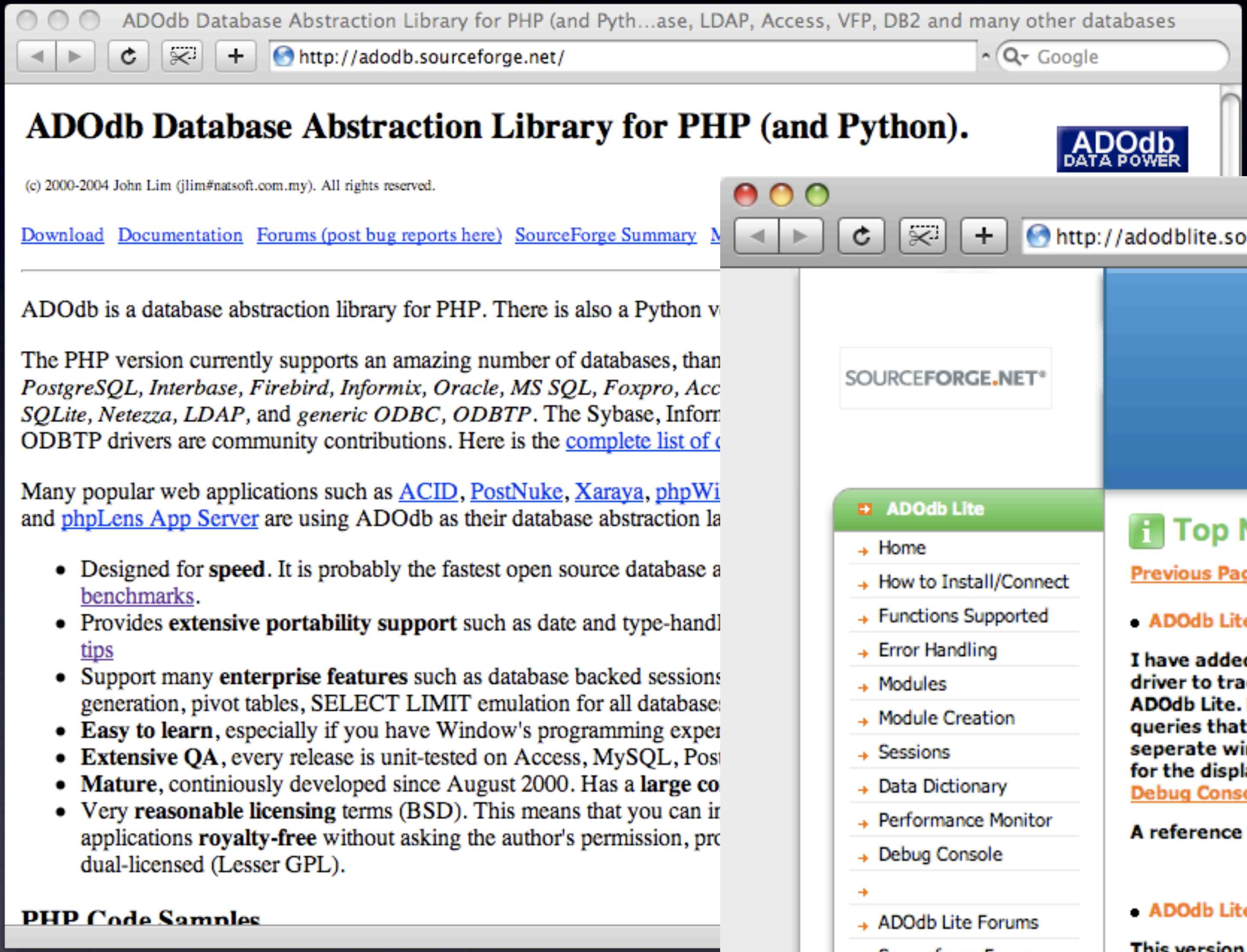
Docs, Active

LGPL

Abstraction *(MySQL, Postgres...)*

Propel is an ORM and we didn't need to replace the one we already had.

ADOdb, ADOdb Lite



ADOdb Database Abstraction Library for PHP (and Pyth...ase, LDAP, Access, VFP, DB2 and many other databases
http://adodb.sourceforge.net/
Google
ADOdb DATA POWER

ADOdb Database Abstraction Library for PHP (and Python).

(c) 2000-2004 John Lim (jlim#natsoft.com.my). All rights reserved.

Download Documentation Forums (post bug reports here) SourceForge Summary M

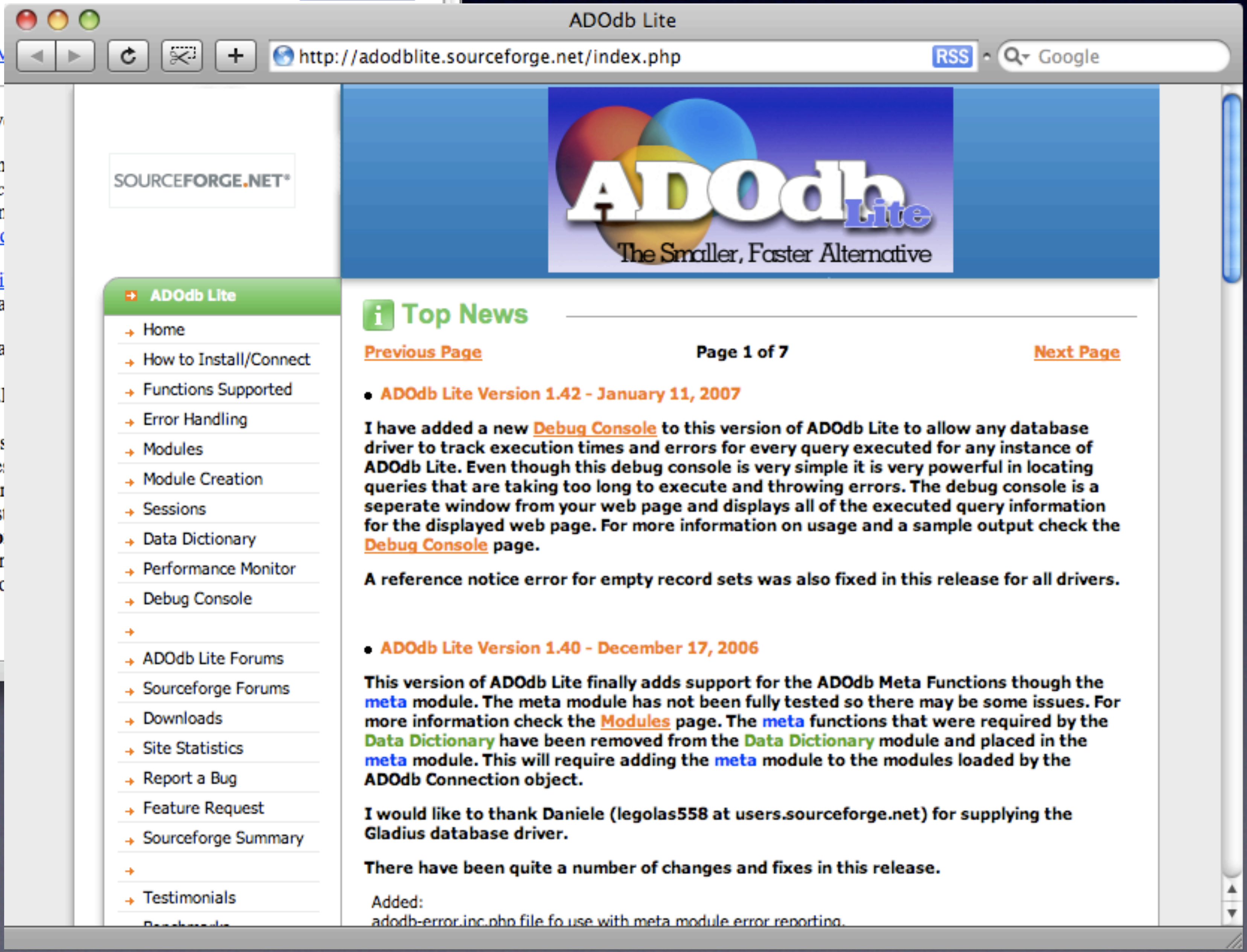
ADOdb is a database abstraction library for PHP. There is also a Python v

The PHP version currently supports an amazing number of databases, than *PostgreSQL, Interbase, Firebird, Informix, Oracle, MS SQL, Foxpro, Acc SQLite, Netezza, LDAP*, and *generic ODBC, ODBTP*. The Sybase, Inform ODBTP drivers are community contributions. Here is the complete list of d

Many popular web applications such as ACID, PostNuke, Xaraya, phpWi and phpLens App Server are using ADOdb as their database abstraction la

- Designed for **speed**. It is probably the fastest open source database a benchmarks.
- Provides **extensive portability support** such as date and type-handl tips
- Support many **enterprise features** such as database backed session generation, pivot tables, SELECT LIMIT emulation for all database
- **Easy to learn**, especially if you have Window's programming exper
- **Extensive QA**, every release is unit-tested on Access, MySQL, Post
- **Mature**, continiously developed since August 2000. Has a **large co**
- Very **reasonable licensing** terms (BSD). This means that you can i applications **royalty-free** without asking the author's permission, pro dual-licensed (Lesser GPL).

PHP Code Samples



Top News

Previous Page Page 1 of 7 Next Page

- **ADOdb Lite Version 1.42 - January 11, 2007**

I have added a new Debug Console to this version of ADOdb Lite to allow any database driver to track execution times and errors for every query executed for any instance of ADOdb Lite. Even though this debug console is very simple it is very powerful in locating queries that are taking too long to execute and throwing errors. The debug console is a seperate window from your web page and displays all of the executed query information for the displayed web page. For more information on usage and a sample output check the Debug Console page.

A reference notice error for empty record sets was also fixed in this release for all drivers.

- **ADOdb Lite Version 1.40 - December 17, 2006**

This version of ADOdb Lite finally adds support for the ADOdb Meta Functions though the meta module. The meta module has not been fully tested so there may be some issues. For more information check the Modules page. The meta functions that were required by the Data Dictionary have been removed from the Data Dictionary module and placed in the meta module. This will require adding the meta module to the modules loaded by the ADOdb Connection object.

I would like to thank Daniele (legolas558 at users.sourceforge.net) for supplying the Gladius database driver.

There have been quite a number of changes and fixes in this release.

Added:
adodb-error.inc.php file fo use with meta module error reporting.

Alternatite syntax for
richer, compatible, sql

ADOdb, ADOdb Lite

```
<?php

$DB = NewADOConnection("mysql://$user:$pwd@$server/$db?persist");

$row = $DB->GetRow("select col from table where key='John'");

?>
```

ADOdb, ADOdb Lite

Comprehensive

Abstraction ++
(MySQL, Postgres...)

Compiled add-on

Docs, Active

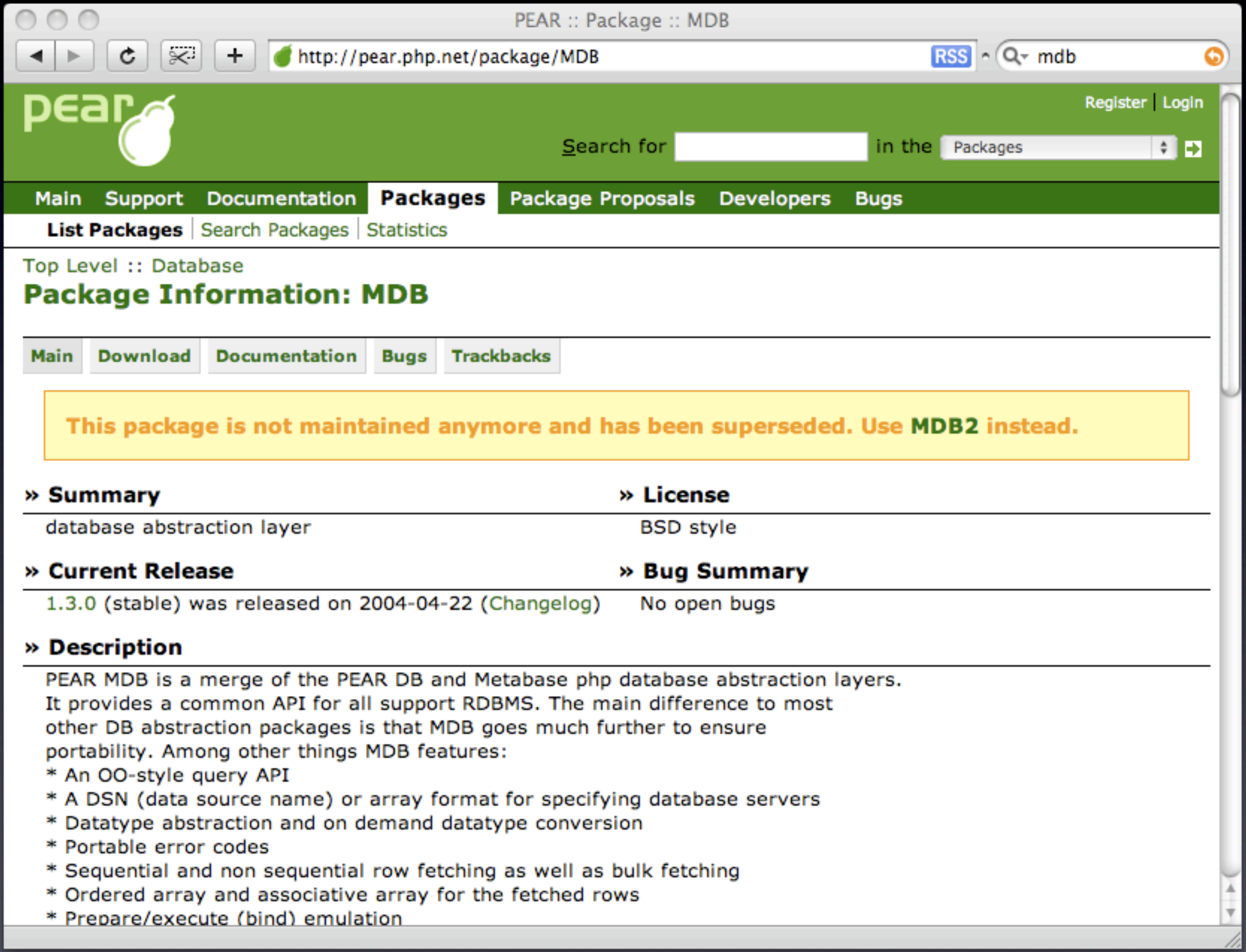
BSD

Just. Too. Much.
... Efficiency?

Hard to engage w/
our architecture.

(ADOdb lite dead?)

MDB



Top Level :: Database

Package Information: MDB

Main | Download | Documentation | Bugs | Trackbacks

This package is not maintained anymore and has been superseded. Use MDB2 instead.

» Summary
database abstraction layer

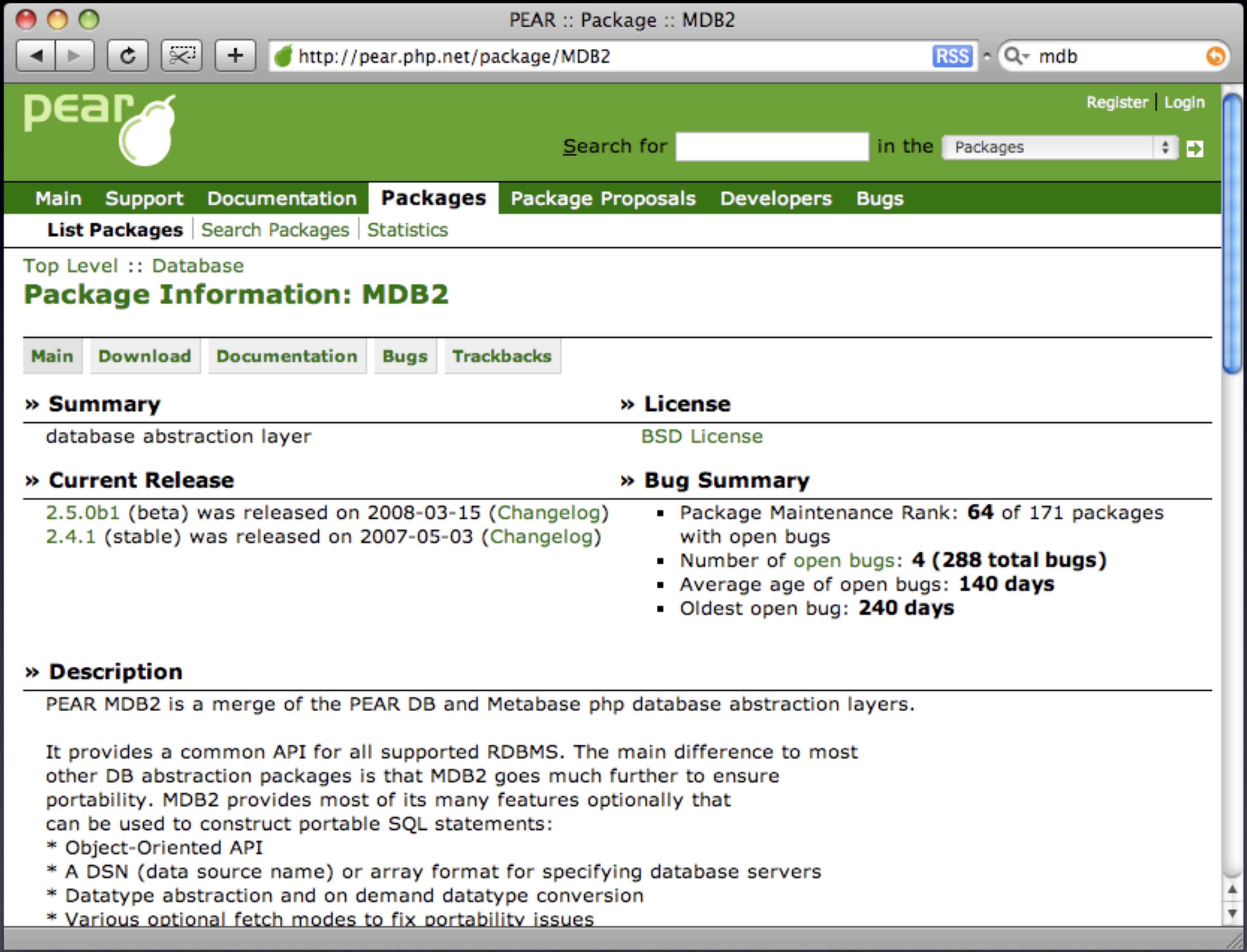
» License
BSD style

» Current Release
1.3.0 (stable) was released on 2004-04-22 (Changelog)

» Bug Summary
No open bugs

» Description
PEAR MDB is a merge of the PEAR DB and Metabase php database abstraction layers.
It provides a common API for all support RDBMS. The main difference to most
other DB abstraction packages is that MDB goes much further to ensure
portability. Among other things MDB features:
* An OO-style query API
* A DSN (data source name) or array format for specifying database servers
* Datatype abstraction and on demand datatype conversion
* Portable error codes
* Sequential and non sequential row fetching as well as bulk fetching
* Ordered array and associative array for the fetched rows
* Prepare/execute (bind) emulation

MDB 2



Top Level :: Database

Package Information: MDB2

Main | Download | Documentation | Bugs | Trackbacks

» Summary

database abstraction layer

» License

BSD License

» Current Release

2.5.0b1 (beta) was released on 2008-03-15 (Changelog)
2.4.1 (stable) was released on 2007-05-03 (Changelog)

» Bug Summary

- Package Maintenance Rank: **64** of 171 packages with open bugs
- Number of open bugs: **4 (288 total bugs)**
- Average age of open bugs: **140 days**
- Oldest open bug: **240 days**

» Description

PEAR MDB2 is a merge of the PEAR DB and Metabase php database abstraction layers.

It provides a common API for all supported RDBMS. The main difference to most
other DB abstraction packages is that MDB2 goes much further to ensure
portability. MDB2 provides most of its many features optionally that
can be used to construct portable SQL statements:
* Object-Oriented API
* A DSN (data source name) or array format for specifying database servers
* Datatype abstraction and on demand datatype conversion
* Various optional fetch modes to fix portability issues

MDB 2

```
<?php
require_once 'MDB2.php';

$mdb2 =& MDB2::connect('mysql://usr:pw@localhost/dbname');
if (PEAR::isError($mdb2)) die($mdb2->getMessage());

$res =& $mdb2->query('SELECT * FROM clients');
if (PEAR::isError($res)) {    die($res->getMessage());

...

?>
```

MDB 2

Abstraction
(MySQL, Postgres...)

Similar to ADOdb
but less ambitious

PEAR *(Conventions)*

Docs, Active?

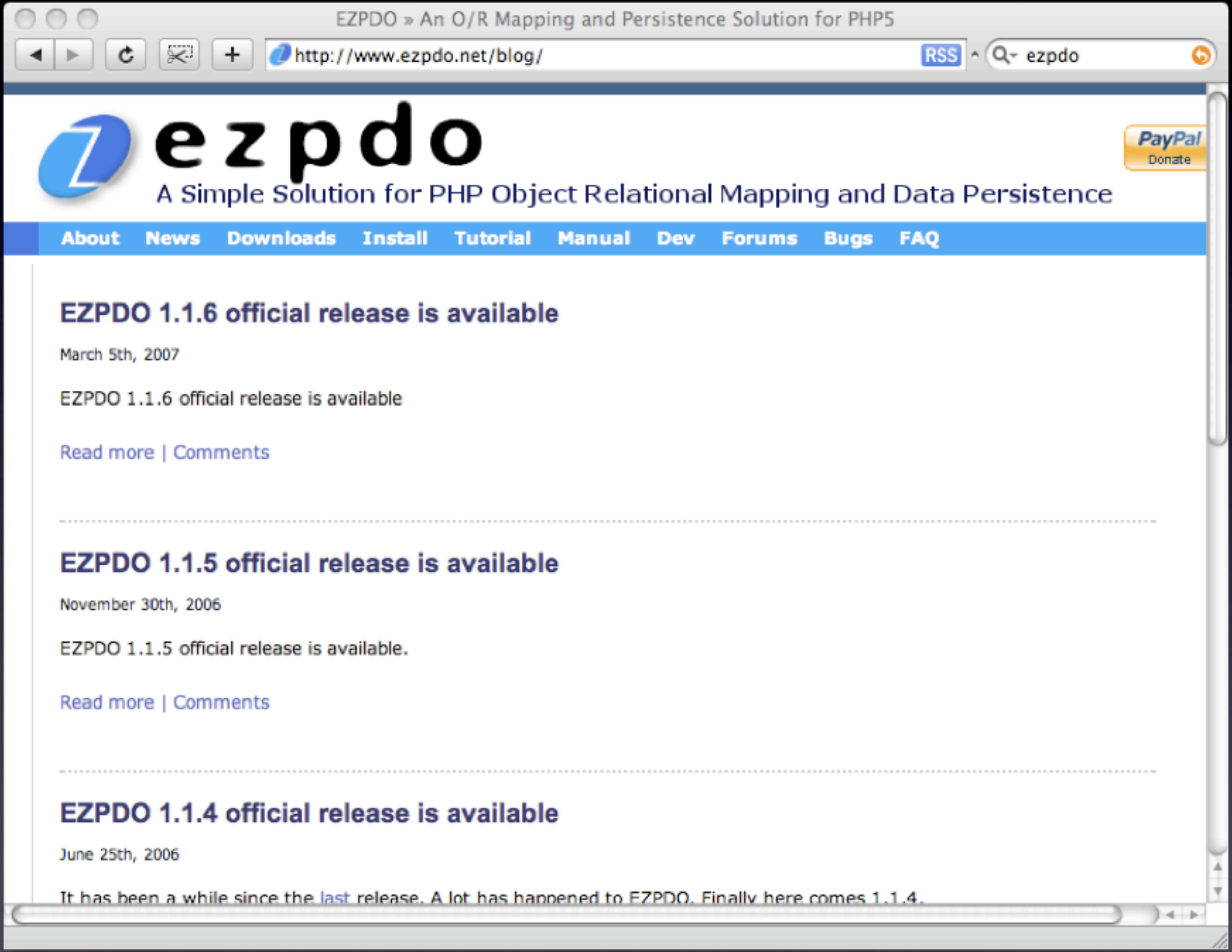
BSD

Didn't engage with
our architecture.
(PHP4 compatible).

Requires PEAR

Slowish dev?

EZPDO



EZPDO » An O/R Mapping and Persistence Solution for PHP5
http://www.ezpdo.net/blog/
ezpdo
ezpdo
A Simple Solution for PHP Object Relational Mapping and Data Persistence
PayPal Donate
About News Downloads Install Tutorial Manual Dev Forums Bugs FAQ

EZPDO 1.1.6 official release is available

March 5th, 2007

EZPDO 1.1.6 official release is available

Read more | Comments

EZPDO 1.1.5 official release is available

November 30th, 2006

EZPDO 1.1.5 official release is available.

Read more | Comments

EZPDO 1.1.4 official release is available

June 25th, 2006

It has been a while since the last release. A lot has happened to EZPDO. Finally here comes 1.1.4.

EZPDO

Code comments build database, and define ORM.

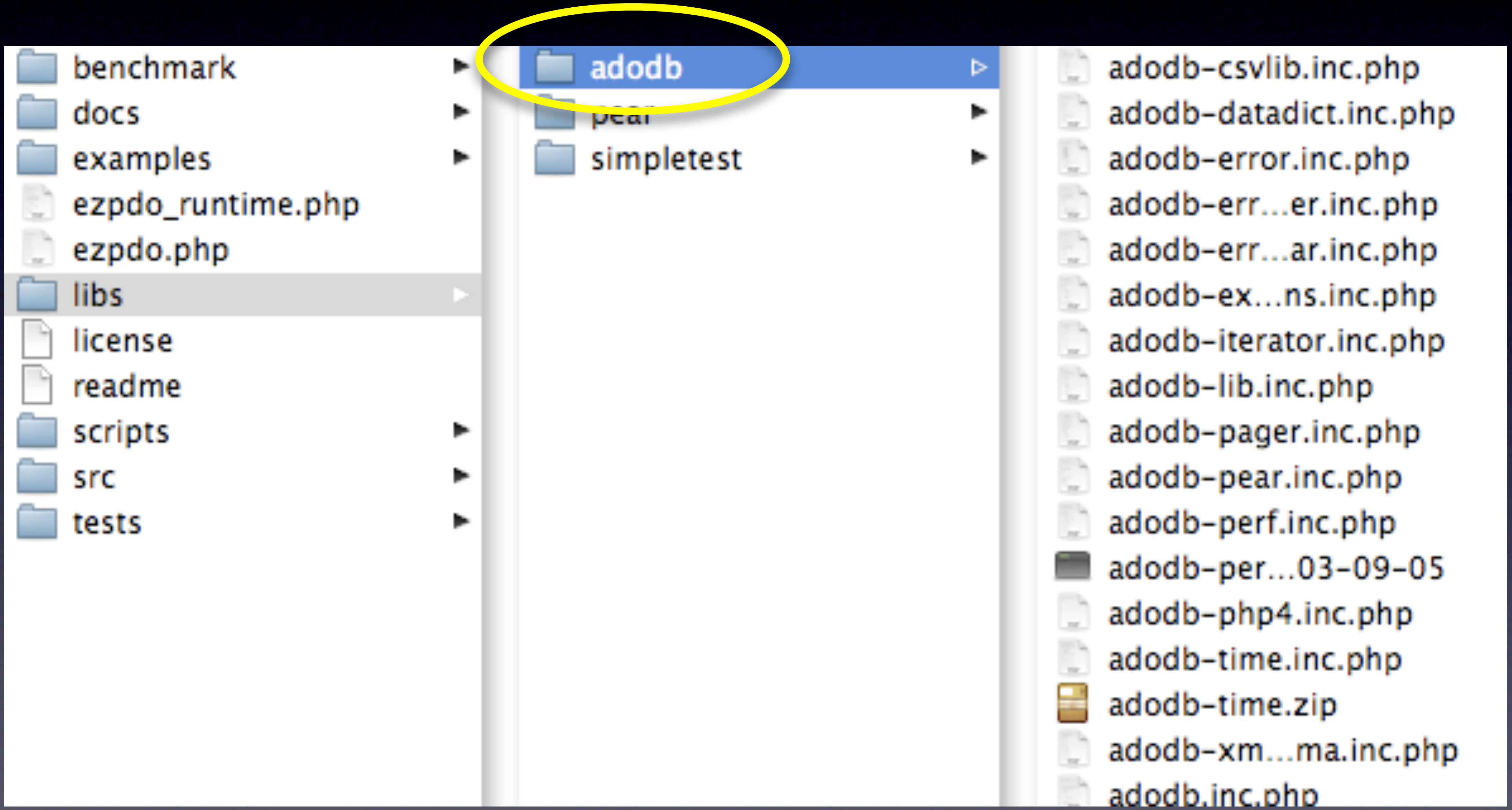
```
<?php
/**
 * Class of a book
 * @orm mysql://username:password@localhost/ezpdo
 */

class Book {
    /**
     * @orm char(64)
     */
    public $title;

    /**
     * @orm char(32)
     */
    public $author;

   // Your regular code for the class here.
}
?>
```

EZPDO



benchmark
docs
examples
ezpdo_runtime.php
ezpdo.php
libs
license
readme
scripts
src
tests

adodb
pear
simpletest

adodb-csvlib.inc.php
adodb-datadict.inc.php
adodb-error.inc.php
adodb-err...er.inc.php
adodb-err...ar.inc.php
adodb-ex...ns.inc.php
adodb-iterator.inc.php
adodb-lib.inc.php
adodb-pager.inc.php
adodb-pear.inc.php
adodb-perf.inc.php
adodb-per...03-09-05
adodb-php4.inc.php
adodb-time.inc.php
adodb-time.zip
adodb-xm...ma.inc.php
adodb.inc.php

EZPDO

PHP5 ORM
Very easy to learn

Sits on top of ADOdb or other abstraction layers

BSD

Provides an ORM and we didn’t need to replace the one we already had.

Comments as code?

Slow development?

PHP Data Objects...

What is PDO?

- Data *access* abstraction layer.
- Consistently named functions across different database drivers.
- Object oriented. PHP language clean-up.
- Tidier code through exceptions.
- Faster safer SQL with prepared stmts.

What is PDO?

Does small number of things really well

Didn’t overlap our code

Fast *(PHP5, compiled)*

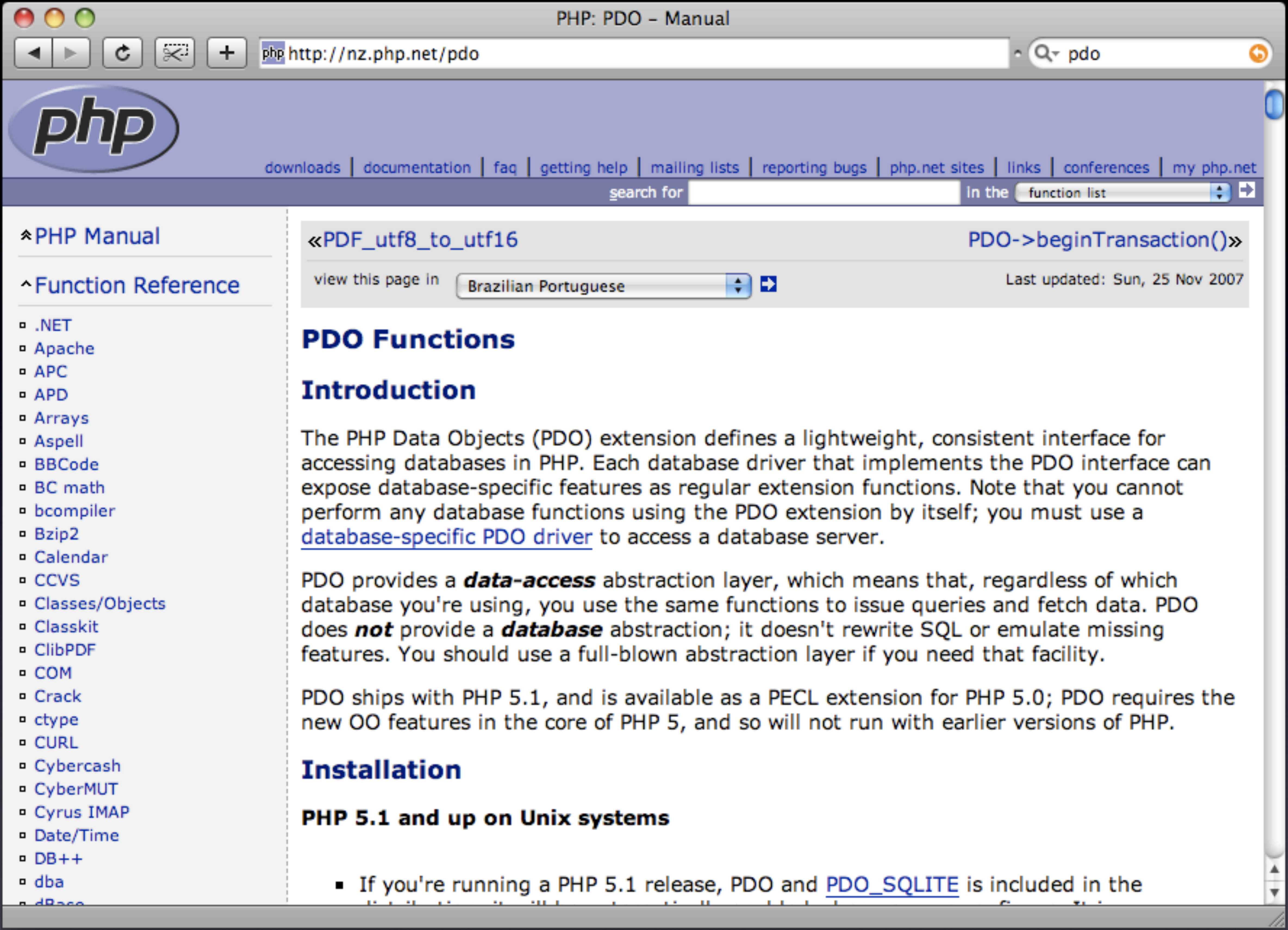
Official, evangelized

Modern architecture

Lacks SQL syntax compatibility filter

Lacks rich set of methods *(e.g. inspect/modify schema)*

Slow PHP5 adoption hurts



PHP: PDO – Manual
http://nz.php.net/pdo
pdo
downloads | documentation | faq | getting help | mailing lists | reporting bugs | php.net sites | links | conferences | my php.net
search for
in the function list

PHP Manual

Function Reference

- .NET
- Apache
- APC
- APD
- Arrays
- Aspell
- BBCode
- BC math
- bcompiler
- Bzip2
- Calendar
- CCVS
- Classes/Objects
- Classkit
- ClibPDF
- COM
- Crack
- ctype
- CURL
- Cybercash
- CyberMUT
- Cyrus IMAP
- Date/Time
- DB++
- dba

«PDF_utf8_to_utf16 PDO->beginTransaction()»

view this page in Brazilian Portuguese

Last updated: Sun, 25 Nov 2007

PDO Functions

Introduction

The PHP Data Objects (PDO) extension defines a lightweight, consistent interface for accessing databases in PHP. Each database driver that implements the PDO interface can expose database-specific features as regular extension functions. Note that you cannot perform any database functions using the PDO extension by itself; you must use a database-specific PDO driver to access a database server.

PDO provides a ***data-access*** abstraction layer, which means that, regardless of which database you're using, you use the same functions to issue queries and fetch data. PDO does ***not*** provide a ***database*** abstraction; it doesn't rewrite SQL or emulate missing features. You should use a full-blown abstraction layer if you need that facility.

PDO ships with PHP 5.1, and is available as a PECL extension for PHP 5.0; PDO requires the new OO features in the core of PHP 5, and so will not run with earlier versions of PHP.

Installation

PHP 5.1 and up on Unix systems

- If you're running a PHP 5.1 release, PDO and PDO_SQLITE is included in the

Old school

- mysql_connect($host, $user, $password);
  mysql_select_db($db);
- sqlite_open($db, 0666);
- pg_connect("host=$host dbname=$db user=$user password=$password");

New school

```
$dsn = “sqlite:/var/mydata.db”;
$user = “siggy”;
$pwd = “qwertyuiop”;



$conn = new PDO(“$dsn”, $user, $pwd);
```

Easy to switch...

```
~~$make = “Ford”;~~
$make = “Ford; DROP DATABASE users;”;      // exploit

$m = mysql_real_escape_string($make);
$m = $conn->quote($make);

$q =   mysql_query("SELECT sum(price) FROM cars WHERE make='$m'");
$q = $conn->query("SELECT sum(price) FROM cars WHERE make=’$m’");

while($r = mysql_fetch_assoc($q)) { echo $r['make']; }
while($r = $q->fetch(PDO::FETCH_ASSOC) ) { echo $r['make']; }

mysql_close();
$conn = null;
```

Prepared statements

```
$authors = array( ... ... ); // imagine 100 records each with several fields

foreach ($authors as $person)
{
    $conn->query("INSERT INTO authors(firstname, surname, email, phone)
                              VALUES ( " . $conn->quote($staff[firstname]) .", " .
                                           $conn->quote($staff[surname])   .", " .
                                             ... ...   .")" );
}

// or, with the Scout's motto...
$stmt = $conn->prepare('INSERT INTO authors(firstname, surname, email, phone)
                                              VALUES(?, ?, ?,?)');
foreach($authors as $person)
{
    $stmt->execute( $person['firstname'], $author['surname'], $author['email']), ... );
}
```

// Second example executes quicker, is safer, and has clearer code...

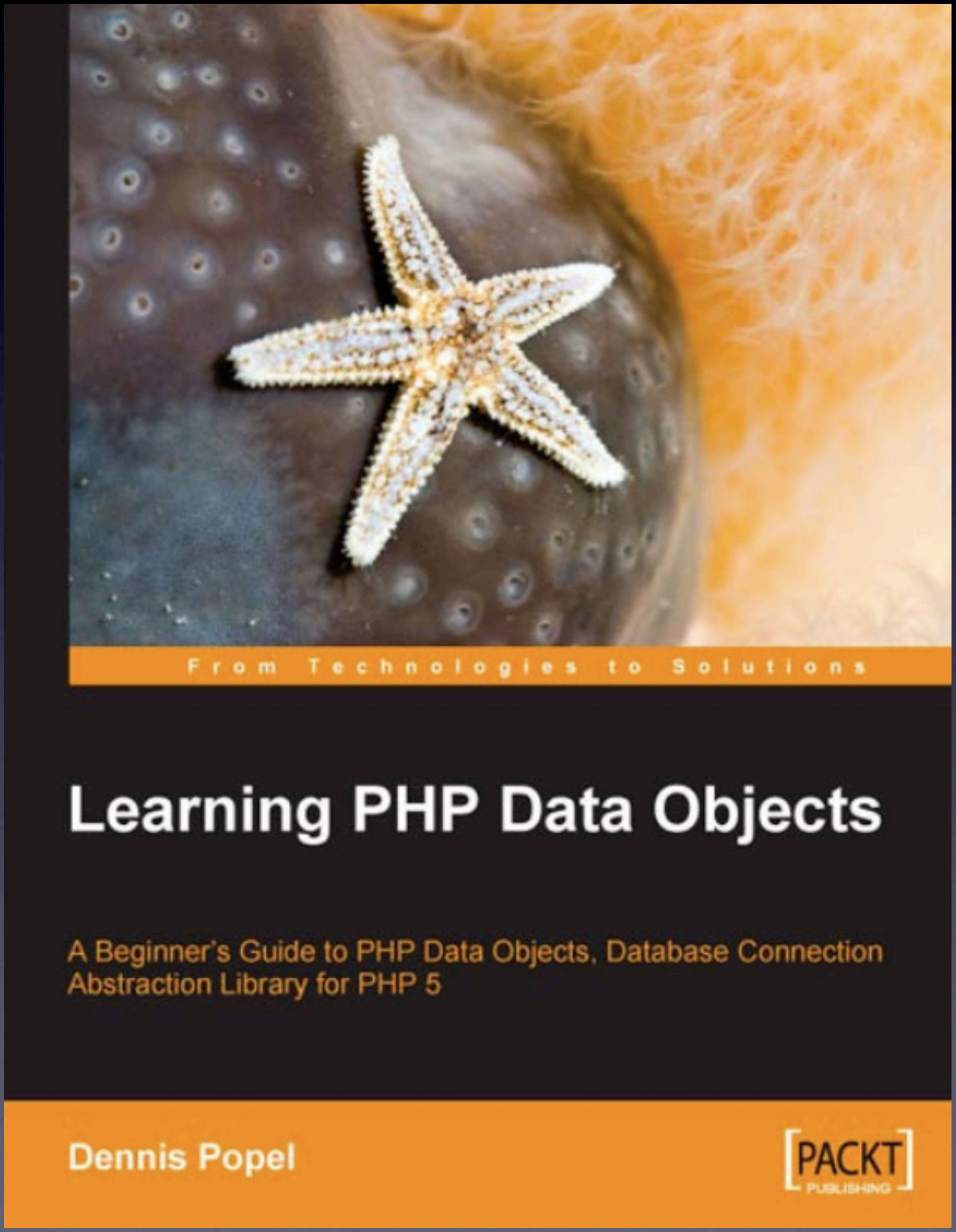
Exceptions & Transactions

```
$conn->beginTransaction();

try
{
    $conn->query("UPDATE accounts SET balance=balance-100  WHERE id=$me");
    $conn->query("UPDATE accounts SET balance=balance+100 WHERE id=$you");

    $conn->commit();
    echo "Funds transferred.";
}
catch(PDOException $e)
{
    $conn->rollBack();
    throw $e;
}
```

Learn more...



From Technologies to Solutions

Learning PHP Data Objects

A Beginner's Guide to PHP Data Objects, Database Connection Abstraction Library for PHP 5

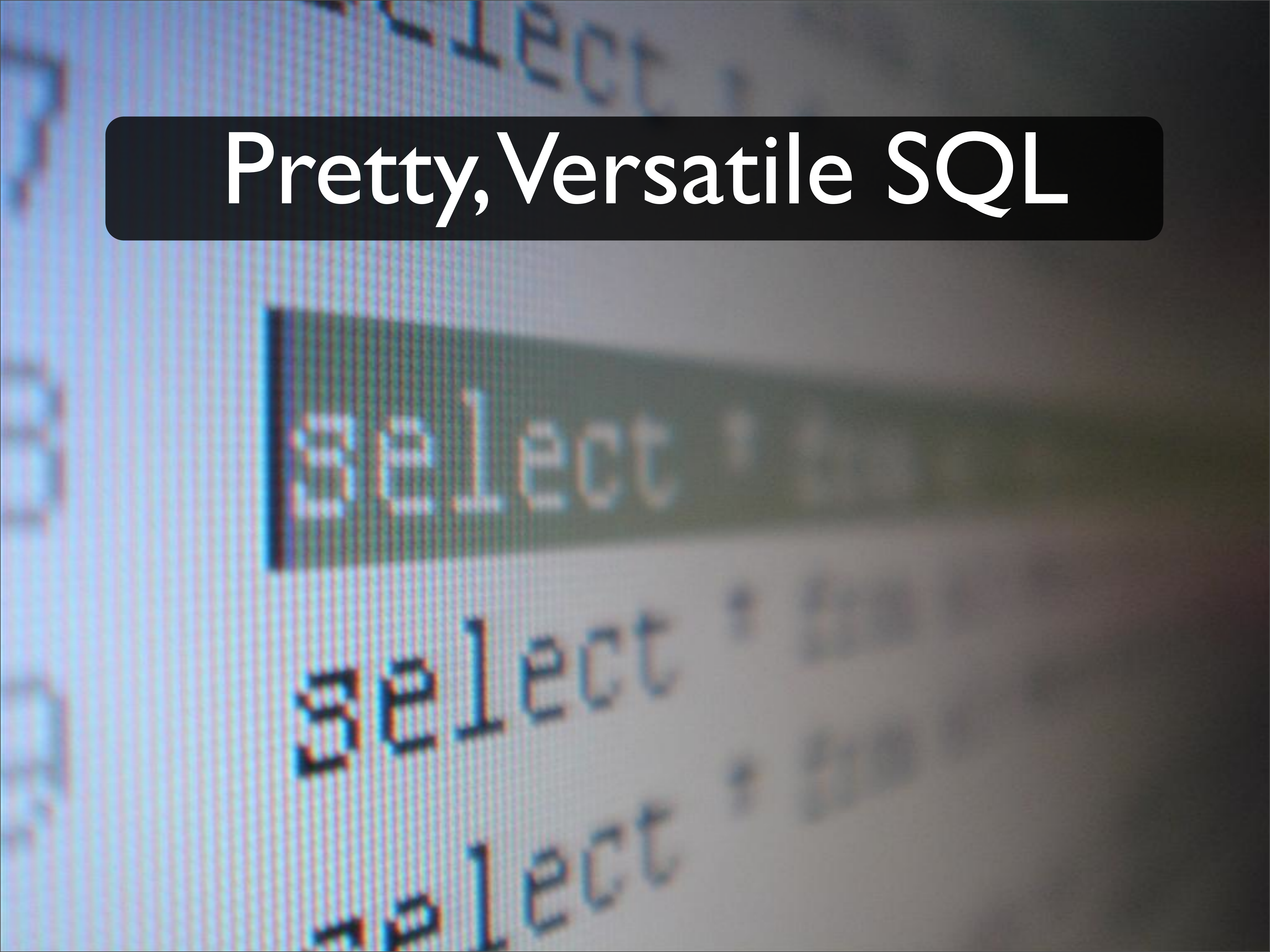
Dennis Popel

PACKT PUBLISHING

Migrating to PDO

Migrating safely

- Consider adding PDO support while keeping mysql_* support for compatibility.
- Unit tests.
- Check databases mirror.



Pretty, Versatile SQL

postgres is "

create table with "cars"
must select as "cars"

SELECT
`Authors`.`Firstname`,
`Authors`.`Surname`
FROM
`Database`.`Authors`
WHERE `` ORDER ``

```
INSERT INTO
CUSTOMERS
~~SET X=1, Y=2~~
(X, Y)
VALUES (1, 2)
```

Dealing with differences

The Norm.

```
if ( mysql_version > 5.2.88.28.8 {
    $sql = “INSERT INTO table SET a=1”;
}


elseif ( $mysql_version < 5.2.88.22.8 &&
         $mysql_version > 4.0.23 {
     $sql = “INSERT INTO table”;
}

else
{
     $sql = “INSER”;
}
```

Better?

```
$query = array(

        "mysql" => array(
            "6.0.0" => “SELECT * from cutting_edge”,
            "5.0.15" => ..., //MySQL 5.0.15 <= x < 6
            "default" => ... //Older...
        ),


        "pgsql" => “SELECT ...”,    // Any version
        "mssql" => ...
);



DB::switched_query($query);
```

Better?

```
$query = array(

        "mysql" => array(
            "6.0.0" => ...,  //MySQL alpha release
            ~~"5.0.15" => ..., //MySQL 5.0.15 and newer~~
            ~~"default" => ... //Older...~~
        ),


        "pgsql" => "SELECT ...",     // Any version
        "mssql" => ...
);



DB::switched_query($query);
```

WILL END IN SADNESS

```
class database {
    function query()
    function num_rows()

    class mysql {
        function connect()
        function insert_row()
        function build_table()
    }

    class pgsql {
        function connect()
        function insert_row()
        function build_table()
        function num_rows()
    }
}
```

```
$db = new mysql/pgsql();          //use factory class
$db->query("SELECT * FROM table");
print_r( $db->getrow() );
$db->insertRow();
```

Thanks!

Slides, links, resources at silverstripe.com/pdo-talk

Sigurd Magnusson, SilverStripe

sigurd@silverstripe.com