PHP Data Objects (PDO)

Reza Salehi

CLOUD CONSULTANT

@zaalion   linkedin.com/in/rezasalehi2008

Overview

An alternative to call relational databases

Introducing PDO (PHP Data Objects)

Configuring PDO

Using PDO for CRUD

Demo:

- Using PDO for MS SQL and MySQL

PDO (PHP Data Objects)

Working with Relational Databases in PHP

Database-specific

Using corresponding databases extensions/objects

Using PDO

Data access layer which uses a unified API

The PHP Data Objects (PDO) defines a lightweight, consistent interface for accessing databases in PHP.

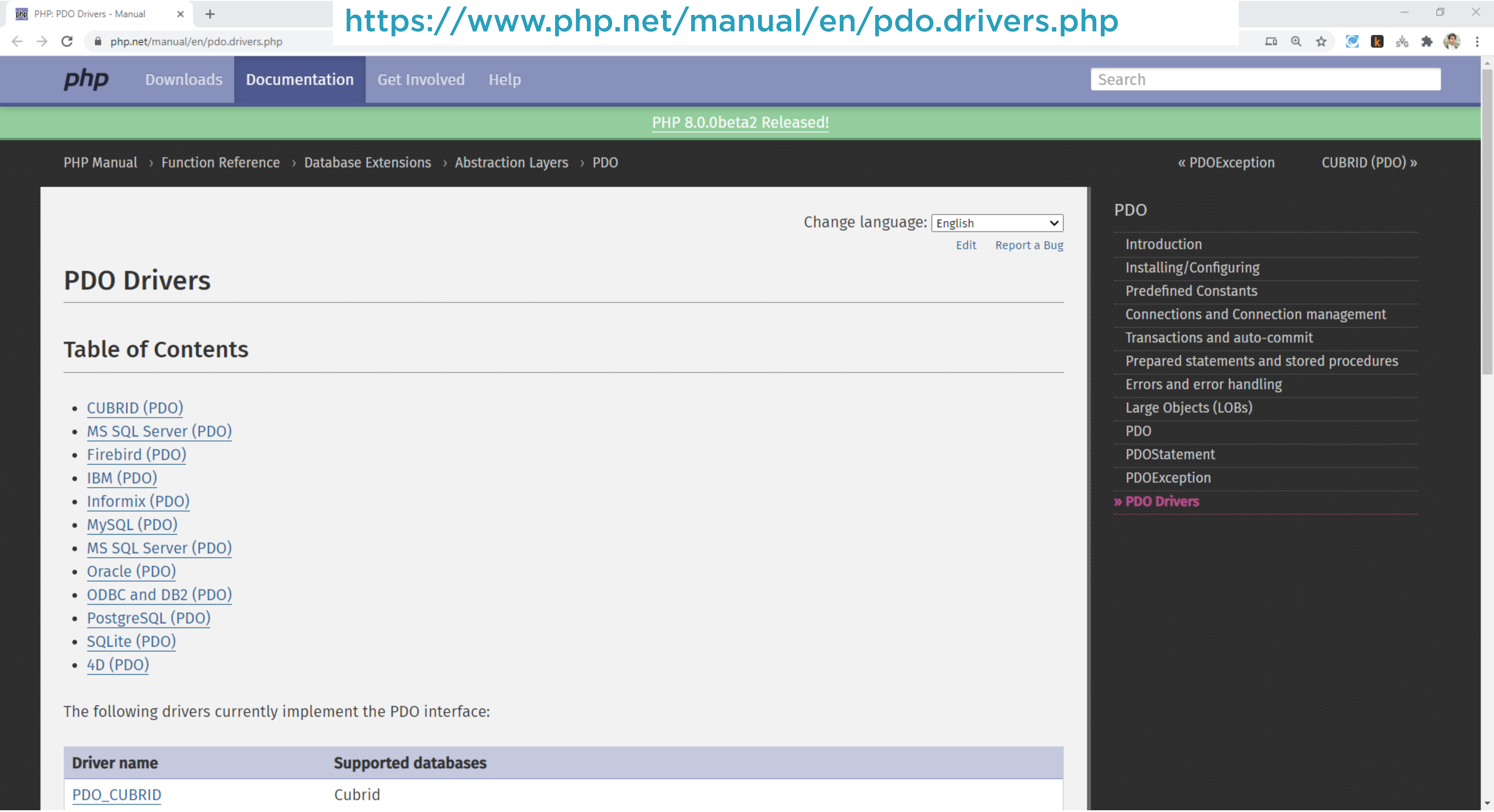
PDO

PDO provides a data-access abstraction layer for PHP

Use the same functions to talk to different relational databases (MySQL, MS SQL, etc.)

PDO ships with PHP 5.1

You must use a database-specific PDO driver to access a database server https://www.php.net/manual/en/pdo.drivers.php



PHP 8.0.0beta2 Released!

PHP Manual › Function Reference › Database Extensions › Abstraction Layers › PDO

« PDOException   CUBRID (PDO) »

Change language: English

Edit   Report a Bug

PDO Drivers

Table of Contents

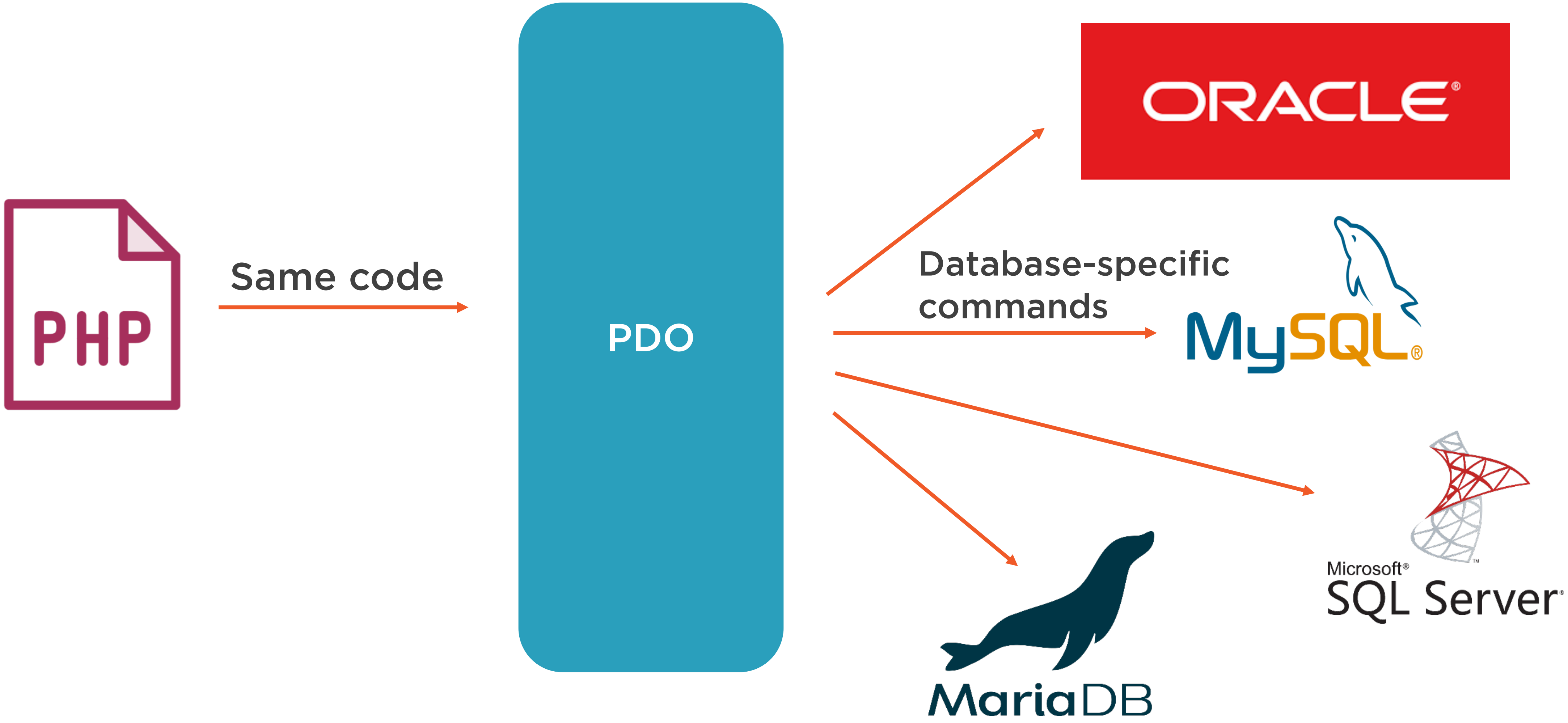
The following drivers currently implement the PDO interface:

Driver name	Supported databases
PDO_CUBRID	Cubrid

PDO

- Introduction
- Installing/Configuring
- Predefined Constants
- Connections and Connection management
- Transactions and auto-commit
- Prepared statements and stored procedures
- Errors and error handling
- Large Objects (LOBs)
- PDO
- PDOStatement
- PDOException
- » PDO Drivers

PDO Data-access Abstraction



Configuring PDO

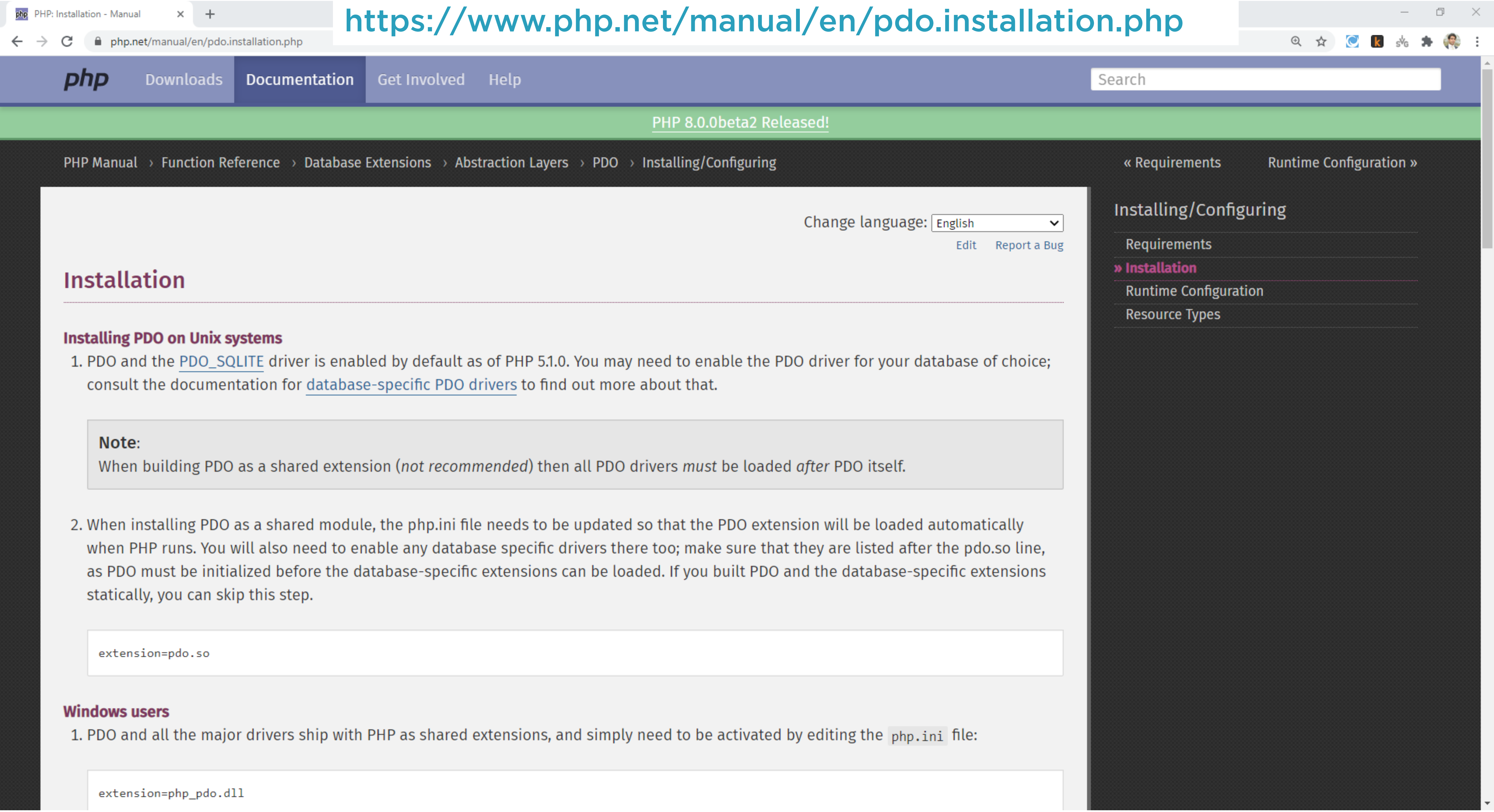
Configuring PDO

Enable PDO extension

Enable database-specific PDO extension

Update code with PDO syntax

https://www.php.net/manual/en/pdo.installation.php



Installation

Installing PDO on Unix systems

1. PDO and the PDO_SQLITE driver is enabled by default as of PHP 5.1.0. You may need to enable the PDO driver for your database of choice; consult the documentation for database-specific PDO drivers to find out more about that.

   > **Note**:
   > When building PDO as a shared extension (*not recommended*) then all PDO drivers *must* be loaded *after* PDO itself.

2. When installing PDO as a shared module, the php.ini file needs to be updated so that the PDO extension will be loaded automatically when PHP runs. You will also need to enable any database specific drivers there too; make sure that they are listed after the pdo.so line, as PDO must be initialized before the database-specific extensions can be loaded. If you built PDO and the database-specific extensions statically, you can skip this step.

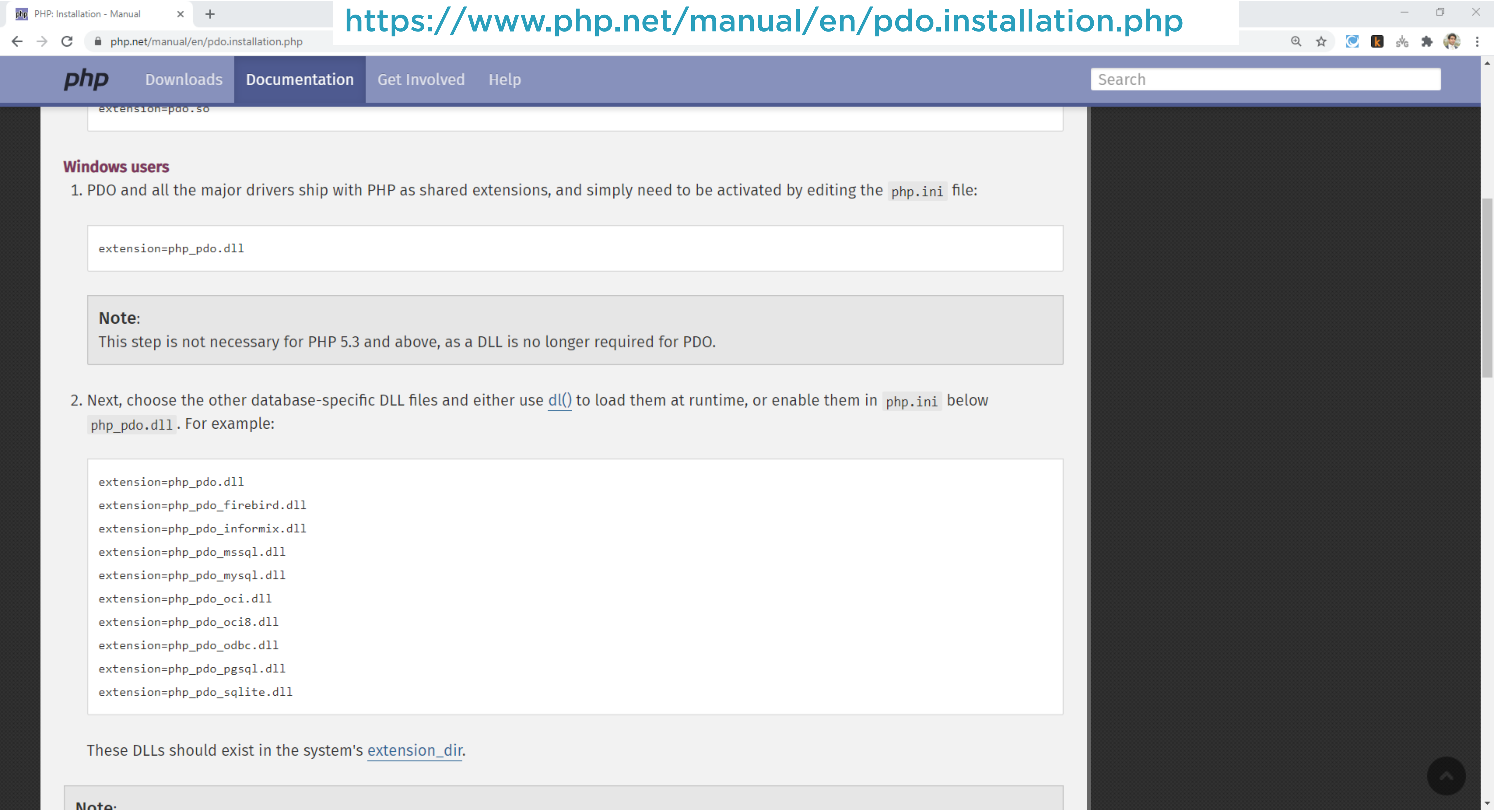
```
extension=pdo.so
```

Windows users

1. PDO and all the major drivers ship with PHP as shared extensions, and simply need to be activated by editing the `php.ini` file:

```
extension=php_pdo.dll
```

https://www.php.net/manual/en/pdo.installation.php


PHP: Installation - Manual
php.net/manual/en/pdo.installation.php
php
Downloads
Documentation
Get Involved
Help
Search

```
extension=pdo.so
```

Windows users

1. PDO and all the major drivers ship with PHP as shared extensions, and simply need to be activated by editing the `php.ini` file:

```
extension=php_pdo.dll
```

> **Note**:
> This step is not necessary for PHP 5.3 and above, as a DLL is no longer required for PDO.

2. Next, choose the other database-specific DLL files and either use dl() to load them at runtime, or enable them in `php.ini` below `php_pdo.dll`. For example:

```
extension=php_pdo.dll
extension=php_pdo_firebird.dll
extension=php_pdo_informix.dll
extension=php_pdo_mssql.dll
extension=php_pdo_mysql.dll
extension=php_pdo_oci.dll
extension=php_pdo_oci8.dll
extension=php_pdo_odbc.dll
extension=php_pdo_pgsql.dll
extension=php_pdo_sqlite.dll
```

These DLLs should exist in the system's extension_dir.

Note:

```php
<?php
$srvname = "localhost";
$usrname = "username";
$password = "password";


// Open MySQL connection
$conn = mysqli_connect($srvname, $usrname, $password);


// Check connection
if (!$conn) {
  die("Connection failed: " . mysqli_connect_error());
}
echo “Successfully connected to MySQL!";
?>
```

◂ MySQL Server name/IP

◂ Database credentials

◂ Open a MySQL connection (Procedural)

◂ Exit if failed, print the connection error

```
<?php
$servername = "localhost, 8080";
$connectionInfo = array( "Database"=>"dbName",
"UID"=>"userName",
"PWD"=>"password");

$conn = sqlsrv_connect( $serverName, $connectionInfo);

if( $conn ) {
     echo “Successfully connected to MS SQL!";
} else {
     echo "Connection failed!";
     die( print_r( sqlsrv_errors(), true));
}?>
```

- MS SQL Server name/IP, port
- Database credentials
- Open a MS SQL connection (Object Oriented)
- Exit if failed, print the connection error

```
<?php

$servername = "localhost"; $dbname = “MyDB”;

$username = "username";

$password = "password";



// Open PDO connection

$conn = new PDO(
"mysql:host=$servername;dbname=$dbname", $username,
$password);

$conn->setAttribute(PDO::ATTR_ERRMODE,
PDO::ERRMODE_EXCEPTION);



$sql = “INSERT INTO person (firstname, lastname)
VAULES (‘Reza’, ‘Salehi’)”;

$conn->exec($sql);

?>
```

◀ MySQL Server name/IP

◀ Database credentials

◀ Open a PDO connection

◀ Set exception attributes

◀ Run SQL command

sqlsrv:Server=

```
<?php

$servername = "localhost"; $dbname = “MyDB”;

$username = "username";

$password = "password";



// Open PDO connection

$conn = new PDO(
"sqlsrv:Server=$host;Database=$dbname", $username,
$password);

$conn->setAttribute(PDO::ATTR_ERRMODE,
PDO::ERRMODE_EXCEPTION);



$sql = “INSERT INTO person (firstname, lastname)
VAULES (‘Reza’, ‘Salehi’)”;

$conn->exec($sql);

?>
```

◂ MS SQL Server name/IP

◂ Database credentials

◂ Open a PDO connection

◂ Set exception attributes

◂ Run SQL command

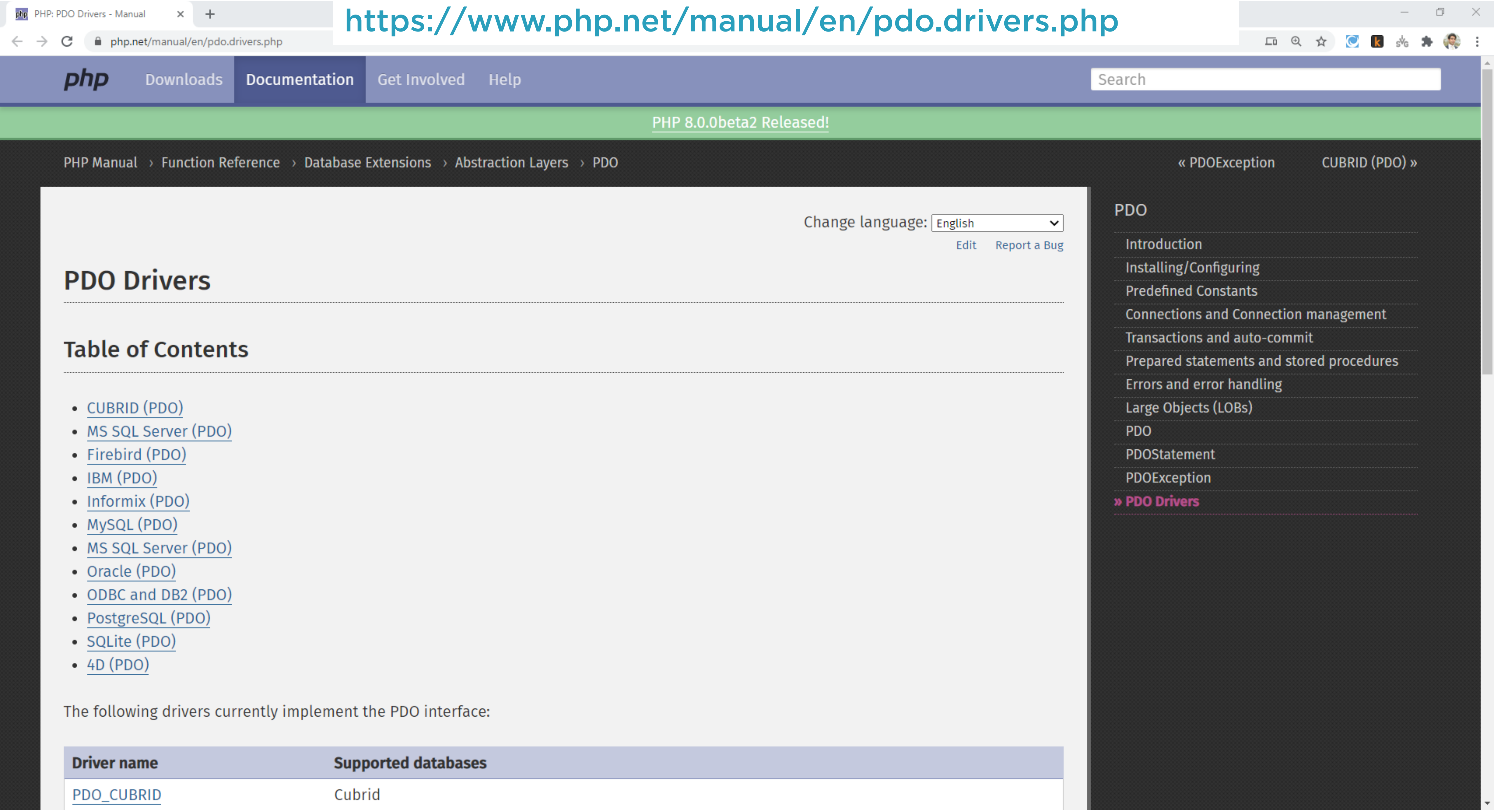
PDO Support

MySQL	MS SQL Server	SQLite
MariaDB	Oracle	4D

PDO Support

- CUBRID
- IBM DB2
- ODBC
- Firebird
- Informix
- PostgreSQL

https://www.php.net/manual/en/pdo.drivers.php



PHP: PDO Drivers - Manual
php.net/manual/en/pdo.drivers.php
php
Downloads
Documentation
Get Involved
Help
Search
PHP 8.0.0beta2 Released!
PHP Manual › Function Reference › Database Extensions › Abstraction Layers › PDO
« PDOException
CUBRID (PDO) »
Change language: English
Edit
Report a Bug
PDO
Introduction
Installing/Configuring
Predefined Constants
Connections and Connection management
Transactions and auto-commit
Prepared statements and stored procedures
Errors and error handling
Large Objects (LOBs)
PDO
PDOStatement
PDOException
» PDO Drivers

PDO Drivers

Table of Contents

The following drivers currently implement the PDO interface:

Driver name	Supported databases
PDO_CUBRID	Cubrid

Demo

Using PDO with MySQL

Demo

Using PDO with MS SQL

- Azure SQL Database

Summary

An alternative to call relational databases

- PDO (PHP Data Objects)

Configuring and using PDO

Demo: using PDO for MS SQL and MySQL