# PDO

## 1. Mi az a PDO?

A PHP Data Objects egy PHP 5 kiterjesztés neve, mely C kód formájában oldja meg az adatbázis absztrakció problémáját, olyan szolgáltatásokat nyújtva, mint a tranzakciók kezelése, előkészített SQL parancsok futtatása, paraméterezése, automatikus típus szerinti *escapelés* (az SQL Injection[11] támadások ellen) és így tovább. Ez az egységesített interfész lehetővé teszi például, hogy rendszerünket SQLite alapokról MySQL alapokra vigyük át egyetlen sor átírásával.

A PDO nem más, mint egy objektum az adatbázis kapcsolatok, lekérdezések kényelmes, hatékony, átlátható kezelésére.

## 2. Használata, legfontosabb metódusai

Kapcsolat inicializálása:

$db = new PDO('mysql:host=localhost;dbname=database', 'user', 'password');

A kapcsolat létrehozásakor példányosított objektum metódusain keresztül érjük el az adatbázisunkat.

A legfontosabb metódusok:

Egyszerű lekérdezés:

A *query* metódussal egyszerűen futtathatunk lekérdezéseket.

$db->query(" SELECT \* FROM `table` ");

Tranzakciók:

A tranzakciók lényege, hogy egy blokkban vagy mindegyik SQL utasítás végrehajtódik, vagy pedig egyik sem. A legjobban ezt egy banki átutalásos példával lehet szemléltetni, ahol az egyik lekérdezés leveszi "A" ügyfél számlájáról a pénzt, a második pedig rárakja "B" számlájára. Ez esetben azt szeretnénk, hogy ha nem sikerül az első lépés, akkor a második se hajtódjon végre.

try {

$db = new PDO('mysql:host=localhost;dbname=database', 'user', 'password');

$db->setAttribute(PDO::ATTR\_ERRMODE, PDO::ERRMODE\_EXCEPTION);

$db->beginTransaction();

$db->exec(" UPDATE `accounts` SET `account\_value`=`account\_value`-100 WHERE `account\_id`='1' ");

$db->exec(" UPDATE `accounts` SET `account\_value`=`account\_value`+100 WHERE `account\_id`='2' ");

$db->commit();

} catch (PDOException $e) {

$db->rollBack();

}

A példában használt metódusok:

setAttribute:

Az adatbázis kezelő objektum tulajdonságait tudjuk vele beállítani különböző, a PDO objektumban tárolt konstansokkal. Ez esetben az lett beállítva, hogy - az alapértelmezettől eltérően - kivételt dobjon hiba esetén.

beginTransaction: Jelzi, a tranzakció kezdetét.

commit: A tranzakció lefuttatása.

rollBack: A tranzakció visszavonása.

Előkészített lekérdezés:

Ennek előnye akkor mutatkozik, amikor egy lekérdezést többször kell futtatnunk különböző paraméterekkel. Ilyenkor a *prepare* metódussal előkészítjük a lekérdezést, a paraméterek helyére egy-egy tetszőleges nevet írva, majd az *execute* metódusnak megadjuk a paraméter név-érték párokat asszociatív tömbként.

$pre = $db->prepare(' INSERT INTO "attrs" ("id", "name") VALUES (:id, :name) ');

$pre->execute(array(':id' => '1', ':name' => '12'));

$pre->execute(array(':id' => '2', ':name' => '43'));

Kapcsolat lezárása:

$db = NULL;

## 3. Miért választottam?

A választásomat a PDO objektum orientáltsága, ezáltal gyors, könnyed kezelése, valamint hasonló képességű konkurens megoldás hiánya indokolja. Működési feltétele csupán a PHP PDO kiterjesztésének és a kiválasztott adatbázis motort támogató PHP kiterjesztés, esetemben SQLite támogatás telepítése.

<http://hu.php.net/pdo>

[11] <http://pezia.hu/content/2009/03/08/weboldalak_biztons%C3%A1ga_1_sql_injection>

<http://wezfurlong.org/blog/2004/may/first-steps-with-pdo/>

<http://deadlime.hu/2006/02/11/mi-is-az-a-pdo/>

<http://www.phpro.org/tutorials/Introduction-to-PHP-PDO.html>