CSIS 3280: Lecture 9

Review and Announcement

- Review
  - Relational Database
  - MySQL

- Announcement
  - Quiz 2 is next week (**You need a webcam**)
  - Material: **SQL**, **FORM**, **PDO**, ~~Advance OO (?)~~

Review and Announcement

- Midterm common mistake
  - Forget to use $this-> in the OO console app
  - Wrong variable name, parameter list
  - Copy and paste from my code without understanding of what's going on
- Please arrange an office hour time with me if you want to see where your mistakes in the midterm are.
- Midterm's free marks
  - I made the folder structure, you just need to rename the main file

Criteria	Grading
Files and folders are named and structured properly according to the naming convention.	1 point

  - You have the comments, and program structure ready. Unless you did not put anything in the config file and/or your code is messy…

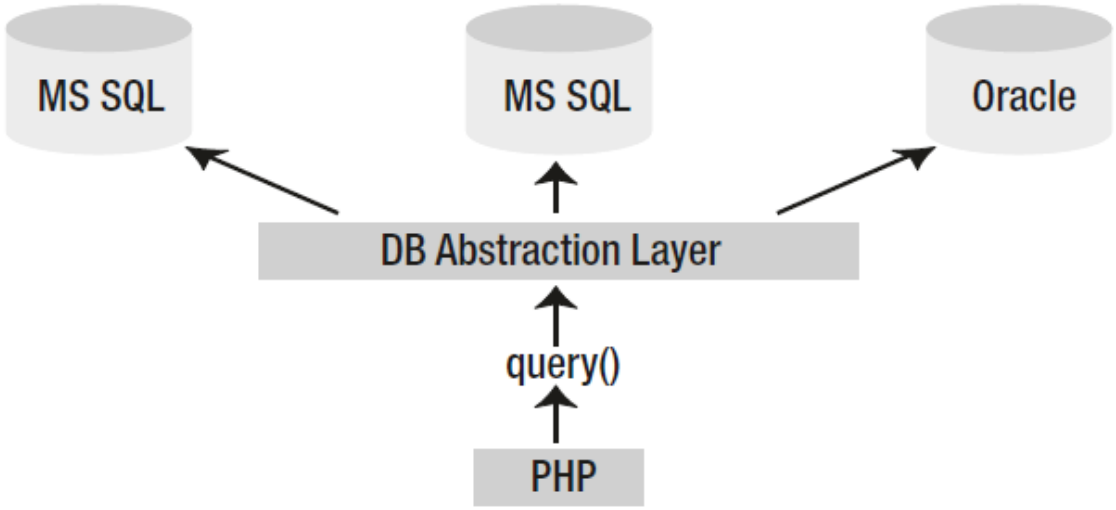
Good program structure is used, comments are used where applicable. Include files are used appropriately and proper separation of concerns is observed. Constants are defined in the config.inc.php.	3 points

Agenda

- Introducing PDO
- Connecting to database
- Executing Query
- Prepared Statements
- Data Access Object (DAO) and PDO

Introducing PDO

- PHP support different kind of databases



- PHP Data Object (PDO):
  - Coding consistency
  - Flexibility
  - OOP
  - Performance

Connecting to a database

- PDO constructor

```
PDO PDO::__construct(string DSN [, string username [, string password
                    [, array driver_opts]]])
```

  - DSN (data source name) specifies the **database driver**, e.g., mysql, the **host**, **database** name (and **port** number)
  - Be careful with the DSN. The default database port in WAMP is used by MariaDB. MariaDB also uses mysql driver.
  - You also need to specify the **username** and **password**
- The following statement instantiate a new PDO to connect to a MySQL database at localhost port 3308, on database chp28 and username/password of webuser/passcode
  - Notice the colon and semicolon at the DSN part

```
$dbh = new PDO('mysql:host=localhost;dbname=ch28;port=3308', 'webuser', 'passcode');
```

PDO connection options

- There are several options you can use in connecting with the database (see textbook page 669—670). We will be using PDO::ATTR_PERSISTENT and PDO::ATTR_ERRMODE :
  - We will set to use a persistent connection, and
  - We will ask PDO to report error on exception

PDO connection

- You can save the credential as constant in the config file

```php
$host = "localhost";
$port = 3308;
$dbName = "books";
$user = "root";
$password = "";

$dsn = "mysql:host=" . $host . ";dbname=" . $dbName . ";port=" . $port;
$options = array(
    PDO::ATTR_PERSISTENT => true,
    PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION
);

try {
    $dbh = new PDO($dsn,$user,$password, $options);
}
catch (PDOException $exception) {
    printf("Connection error: %s", $exception->getMessage());
}
```

Query

- Query with no result returned (INSERT, UPDATE, DELETE) → use exec()

```php
$query = "INSERT INTO books(ISBN, Author, Title, Price)
            VALUES('1-123-12345', 'Bambang', 'PHP Made Easy', '99.99') ";
$dbh->exec($query);


$query = "UPDATE books
            SET ISBN='1-123-12345-1'
            WHERE ISBN='1-123-12345'";
$affected = $dbh->exec($query);
echo "Total rows affected: $affected";
```

Query

- Query expecting result(s) (SELECT) → use query()

```php
$query = "SELECT * FROM books";
$dbh->query($query);

foreach ($dbh->query($query) AS $row){
    printf("%s by %s (%s) is \$%s \n",
        $row['Title'],$row['Author'],$row['ISBN'],$row['Price']);
}
```

Prepared Statements

- In order to avoid the SQL injection attack and to improve efficiency, use prepared statements
- You need to prepare(), bindParam() and execute() the query

```
PDOStatement PDO::prepare(string query [, array driver_options])
boolean PDOStatement::execute([array input_parameters])
boolean PDOStatement::bindParam(mixed parameter, mixed &variable [, int
datatype [, int length [, mixed driver_options]]])
```

- Below is an example of using prepare() and execute(). Notice the use of colon (:)

```
// Create and prepare the query
$query = "INSERT INTO products SET sku =:sku, title =:title";
$stmt = $dbh->prepare($query);

// Execute the query
$stmt->execute( [':sku' => 'MN873213', ':title' => 'Minty Mouthwash'] );

// Execute again
$stmt->execute( [':sku' => 'AB223234', ':title' => 'Lovable Lipstick'] );
```

Binding the parameter

- Binding the parameter can use the following options

`PDO::PARAM_BOOL`: SQL `BOOLEAN` datatype

`PDO::PARAM_INPUT_OUTPUT`: Used when the parameter is passed into a stored procedure and therefore could be changed after the procedure executes

`PDO::PARAM_INT`: SQL `INTEGER` datatype

`PDO::PARAM_NULL`: SQL `NULL` datatype

`PDO::PARAM_LOB`: SQL large object datatype

`PDO_PARAM_STMT`: `PDOStatement` object type; presently not operational

`PDO::PARAM_STR`: SQL string datatypes

Binding the parameter

- Use is_null(), is_int(), is_bool(), etc to check the data to simplify the binding process

```php
public function bind($param, $value, $type = null)  {
    if (is_null($type)) {
        switch (true) {
            case is_int($value):
                $type = PDO::PARAM_INT;
                break;
            case is_bool($value):
                $type = PDO::PARAM_BOOL;
                break;
            case is_null($value):
                $type = PDO::PARAM_NULL;
                break;
            default:
                $type = PDO::PARAM_STR;
                break;
        }
    }
    $this->stmt->bindValue($param, $value, $type);
}
```

Fetching Row(s)

- You can read the next arrow of result and fetch it in different format, e.g., array, associative arrays, **class**!

```
mixed PDOStatement::fetch([int fetch_style [, int cursor_orientation
                          [, int cursor_offset]]])
array PDOStatement::fetchAll([int fetch_style])
```

- fetchAll() accept an optional second argument. Therefore, we can fetch all the result into a specific class
- If we want to fetch() the result into a class, we need to use setFetchMode() before we perform fetch()

```php
public function singleResult()   {

    //Executethe statement
    $this->stmt->execute();
    //set fetch mode to return classes
    $this->stmt->setFetchMode(PDO::FETCH_CLASS, $this->className);
    return $this->stmt->fetch(PDO::FETCH_CLASS);
}
```

Data Access Object

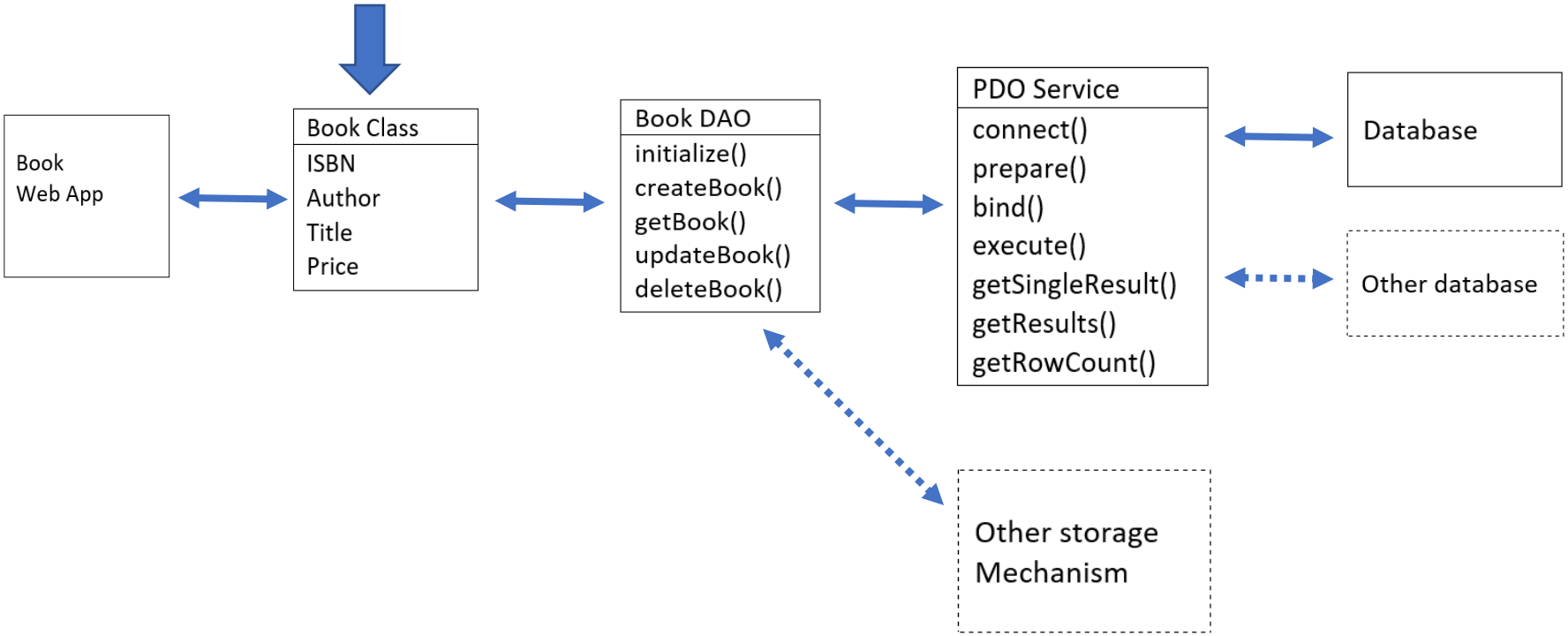
- DAO is a design pattern that provides an abstract interface to some type of database or other persistence mechanism (relational, OO, XML, etc)
  - Separates low-level data accessing operations from high level business services
  - Separates what data access the application needs from how the needs can be satisfied by the storage engine
  - The details of the database will not be exposed and can be changed/modified as needed
- For example, in our book application demo, we need to have the functionality to createBook(), getBooks(), deleteBook() and updateBook().

Web application with PDO and DAO

- The following are the files that we need to build
  - config.inc.php – stores the DB configuration
  - PDOAgent.class.php – a class that will manage all low-level connection to the database
    - Connection, preparing the statement, binding, executing, etc
    - Single row operations (to get detail on a specific entry, or edit)
    - Result set operations
  - Book.class.php – stores the Book class properties and methods
    - **The properties' name must match with the table columns' name**
  - BookDAO.class.php – the DAO of Book class. The layer between the Book class and PDO agent
  - Page.class.php – the class for displaying the HTML
  - The main controller file

Book application

```
mysql> desc books;
+--------+------------+------+-----+---------+-------+
| Field  | Type       | Null | Key | Default | Extra |
+--------+------------+------+-----+---------+-------+
| ISBN   | char(13)   | NO   | PRI | NULL    |       |
| Author | char(50)   | YES  |     | NULL    |       |
| Title  | char(100)  | YES  |     | NULL    |       |
| Price  | float(4,2) | YES  |     | NULL    |       |
+--------+------------+------+-----+---------+-------+
4 rows in set (0.02 sec)
```



Lab and next week

- Please download the Lab. You must submit the lab next week at 9:00 AM

- Next week:
  - Quiz 2
  - String