esprit
Se former autrement
HONORIS UNITED UNIVERSITIES

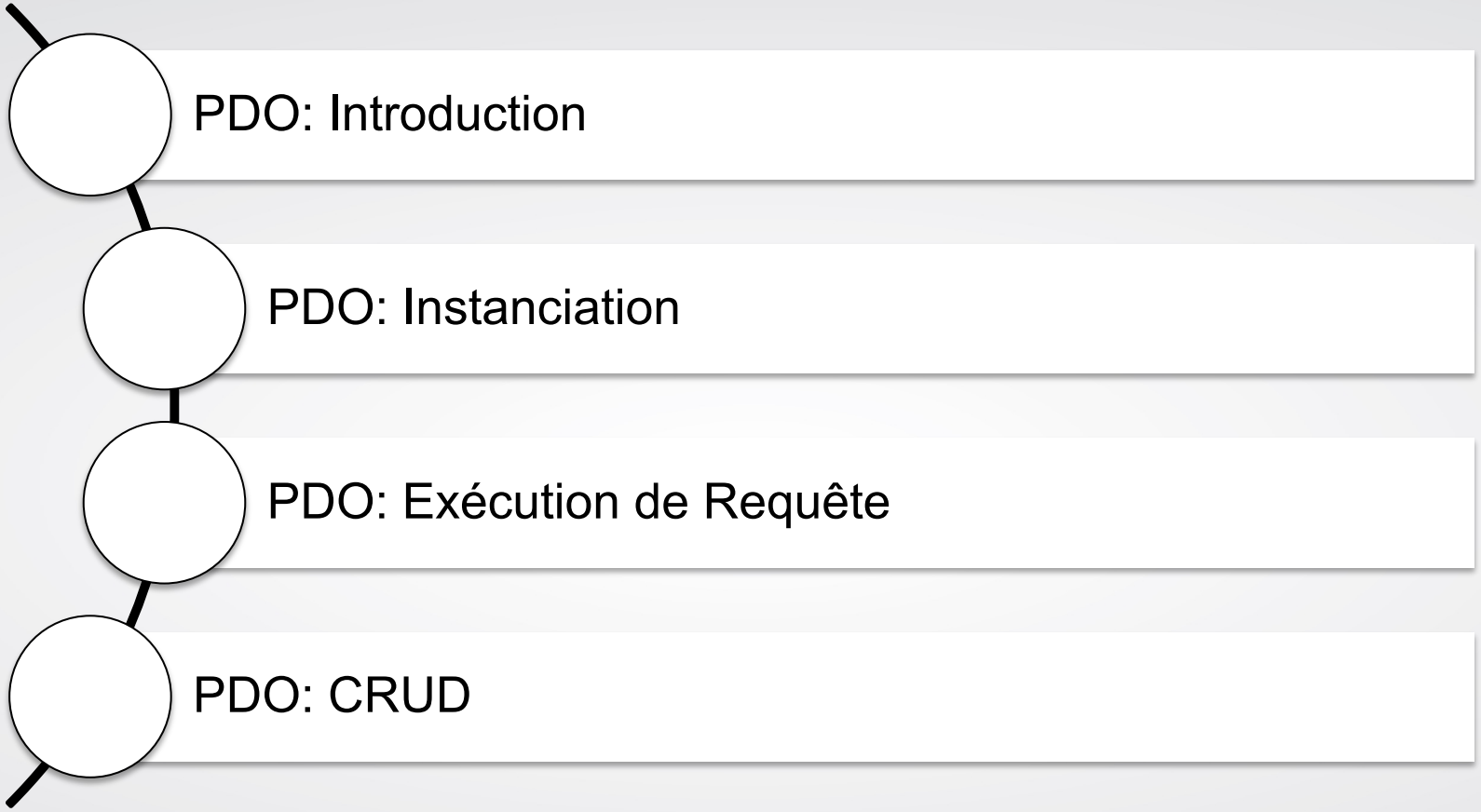
Chapitre 5: PHP

UP Web

AU: 2023/2024

United Nations Educational, Scientific and Cultural Organization
UNESCO Chair "Project-based learning" ESPRIT School of engineering, Tunisia
EUR-ACE® Délivrée par la Commission des Titres d'Ingénieur
CONFÉRENCE DES GRANDES ÉCOLES
cdio CONCEIVE DESIGN IMPLEMENT OPERATE

Plan


PDO: Introduction
PDO: Instanciation
PDO: Exécution de Requête
PDO: CRUD

Objectifs

- Les architectures du web
- Comprendre la syntaxe PHP
- Appréhender les notions de l'orientée objet
- Se connecter à une BD
- Manipuler les données d'une BD via PHP

Prérequis

- Langage HTML

Manipulation de la BD

PDO: PHP Data Objects

- PDO définit une interface pour accéder à une BD depuis PHP.
- PDO supporte plusieurs systèmes de bases de données:
  - MySQL
  - ODBC
  - SQLITE
  - OCI Oracle Call Interface
  - SQLite
  - …

Manipulation de la BD

PHP connexion à la BD

- Pour se connecter au serveur, on peut utiliser le fichier '**config.php**':

```php
<?php
class config
{   private static $pdo = null;
    public static function getConnexion()
    {
        if (!isset(self::$pdo)) {
            $servername="localhost";
            $username="root";
            $password ="password";
            $dbname="myDBName";
            try {
                self::$pdo = new PDO("mysql:host=$servername;dbname=$dbname",
                        $username,
                        $password,
                    [
                        PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,
                        PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC
                    ]
                );
                echo "connected successfully";
            } catch (Exception $e) {
                die('Erreur: ' . $e->getMessage());
            }
        }
        return self::$pdo;
    }
}
config::getConnexion();
?>
```

Manipulation de la BD

PHP connexion à la BD

Cette configuration permet de lancer une exception PDOException en cas d'erreur.

```php
<?php
class config
{   private static $pdo = null;
    public static function getConnexion()
    {
        if (!isset(self::$pdo)) {
            $servername="localhost";
            $username="root";
            $password ="password";
            $dbname="myDBName";
            try {
                self::$pdo = new PDO("mysql:host=$servername;dbname=$dbname",
                        $username,
                        $password,
                    [
                        PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,
                        PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC
                    ]
                );
                echo "connected successfully";
            } catch (Exception $e) {
                die('Erreur: ' . $e->getMessage());
            }
        }
        return self::$pdo;
    }
}
config::getConnexion();
?>
```

Manipulation de la BD

PHP connexion à la BD

Cette configuration permet de spécifier la méthode de récupération. Dans se cas, chaque ligne est retournées dans un tableau indexé par le nom des colonnes.

```php
<?php
class config
{   private static $pdo = null;
    public static function getConnexion()
    {
        if (!isset(self::$pdo)) {
            $servername="localhost";
            $username="root";
            $password ="password";
            $dbname="myDBName";
            try {
                self::$pdo = new PDO("mysql:host=$servername;dbname=$dbname",
                        $username,
                        $password,
                    [
                        PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,
                        PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC
                    ]
                );
                echo "connected successfully";
            } catch (Exception $e) {
                die('Erreur: ' . $e->getMessage());
            }
        }
        return self::$pdo;
    }
}
config::getConnexion();
```

Manipulation de la BD

PHP connexion à la BD

- Ou en utilisant la syntaxe suivante:

```php
<?php
class config
{   private static $pdo = null;
    public static function getConnexion()
    {
        if (!isset(self::$pdo)) {
            $servername="localhost";
            $username="root";
            $password ="password";
            $dbname="myDBName";
            try {
                self::$pdo = new PDO("mysql:host=$servername;dbname=$dbname",
                        $username,
                        $password

                );
                self::$pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
                self::$pdo->setAttribute(PDO::ATTR_DEFAULT_FETCH_MODE, PDO::FETCH_ASSOC);

                echo "connected successfully";
            } catch (Exception $e) {
                die('Erreur: ' . $e->getMessage());
            }
        }
        return self::$pdo;
    }
}
config::getConnexion();
?>
```

Manipulation de la BD

PHP connexion à la BD

- La première étape consiste à créer un objet PDO:

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

1. **dsn ou Data Source Name**: représente les informations nécessaires pour se connecter à la base.
2. **Username** (optionnel pour certains pilote tel que sqlite)
3. **Password**
4. **Options**: un tableau contenant les options spécifiques à la connexion. Ce paramètre est optionnel.

Remarque: La base de données (myDB) peut être créer en utilisant l'application phpMyAdmin.

Manipulation de la BD

PHP connexion à la BD

- La connexion sera fermée automatiquement lorsque le script se termine. Pour fermer la connexion avant, utilisez la commande suivante:

```
self::$pdo = null;
```

- Comment je peux inclure le fichier de « **connection.php** »?

Récrire le code connection.php, utiliser les fonctions :

- require ()
- require-once()
- include()
- include-once()

Manipulation de la BD

PHP connexion à la BD

- La différence entre include et include_once:
  - include_once inclut et évalue le fichier spécifié durant l'exécution du script. Le comportement est similaire à include, mais la différence est que si le code a déjà été inclus, il ne le sera pas une seconde fois, et include_once retourne TRUE.
  - La structure *include_once* est utilisée:
    - de préférence lorsque le fichier va être inclus ou évalué plusieurs fois dans un script,
    - ou bien lorsque vous voulez être sûr qu'il ne sera inclus qu'une seule fois, pour éviter des redéfinitions de fonctions ou de classes.

Manipulation de la BD

PHP connexion à la BD

- La différence entre include et require:
  - require est identique à include mise à part le fait que lorsqu'une erreur survient, il produit également une erreur fatale de niveau E_COMPILE_ERROR.
  - Il stoppera le script alors que include n'émettra qu'une alerte de niveau E_WARNING, ce qui permet au script de continuer.

Manipulation de la BD

PHP connexion à la BD

- La différence entre require et require_once?
  - L'instruction *require_once* est identique à require mis à part que PHP vérifie si le fichier a déjà été inclus, et si c'est le cas, ne l'inclut pas une deuxième fois.

=> Pour faire appel au fichier « config.php » il suffit:

```php
<?php

require '../config.php';
//.............................


?>
```

Manipulation de la BD

PDO: exec(), query(), execute(), prepare()

Fonction	Explication
PDO::exec(string $sql): int	Permet d'exécuter une requête. Renvoie le nombre de ligne affectées. À utiliser avec: Insert, Update et Delete.
PDO::query(string $sql): PDOStatement	Exécute une requête. Retourne le résultat en un objet PDOStatement
PDO::prepare(string $sql): PDOStatement	Prépare la requête sans l'exécuter. Renvoie un objet PDOStatement.
PDOStatement::execute([array $param]): bool	Exécute une requête préparée.

Manipulation de la BD

PDO: CRUD

CRUD: ‘Create’ des données

L'instruction INSERT INTO est utilisée pour ajouter de nouveaux enregistrements à une table.

```php
<?php
        require '../config.php';
        $sql = "INSERT INTO personne (Nom, Prenom, Age) VALUES (:nom, :prenom,:age)";
        $db = config::getConnexion();
        try {
            $query = $db->prepare($sql);
            $query->execute([
                'nom' => 'Ben Mohamed',
                'prenom' => 'John',
                'age' => 23
            ]);
        } catch (Exception $e) {
            die('Error:' . $e->getMessage());
        }

?>
```

Manipulation de la BD

PDO: CRUD

CRUD: ‘Create’ des données

L'instruction INSERT INTO est utilisée pour ajouter de nouveaux enregistrements à une table.

PDOStatement::bindValue(
$param,
$value,
[$data_type]
) : bool

```php
<?php
require '../config.php';
$sql = "INSERT INTO personne (Nom, Prenom, Age) VALUES (:nom, :prenom,:age)";
$db = config::getConnexion();
try {
    $query = $db->prepare($sql);
    $query->bindValue(':nom', 'Ben Mohamed');
    $query->bindValue(':prenom', 'John');
    $query->bindValue(':age', 23,PDO::PARAM_INT);
    $query->execute();
} catch (Exception $e) {
    die('Error:' . $e->getMessage());
}
?>
```

Manipulation de la BD

PDO: CRUD

CRUD: ‘Create’ des données

L'instruction INSERT INTO est utilisée pour ajouter de nouveaux enregistrements à une table.

PDOStatement::bindParam
(
$param,
$value,
[$data_type],
[$length]
) : bool

```php
<?php
require '../config.php';
$sql = "INSERT INTO personne (Nom, Prenom, Age) VALUES (?, ?, ?)";
$db = config::getConnexion();
try {
    $query = $db->prepare($sql);
    $nom = 'Ben Mohamed';
    $prenom = 'John';
    $age = 23;
    $query->bindParam(1, $nom, PDO::PARAM_STR);
    $query->bindParam(2, $prenom, PDO::PARAM_STR);
    $query->bindParam(3, $age, PDO::PARAM_INT);
    $query->execute();
} catch (Exception $e) {
    die('Error:' . $e->getMessage());
}
?>
```

Manipulation de la BD

PDO: CRUD

CRUD: ‘Read’ des données
L'instruction SELECT est utilisée pour sélectionner les données à partir d'une table.

```php
<?php
        require '../config.php';


        $sql = "SELECT * FROM personne";
        $db = config::getConnexion();
        try {
            $query = $db->prepare($sql);
            $query->execute();
            $result =$query->fetchAll();

        } catch (Exception $e) {
            die('Error:' . $e->getMessage());
        }
        foreach ($result as $row) {
            echo $row['Nom'].' '.$row['Prenom'];
        }

?>
```

Manipulation de la BD

PDO: CRUD

CRUD: 'Update' des données
L'instruction UPDATE est utilisée pour modifier les données d'une table.

```php
<?php
require '../config.php';
$sql = "UPDATE personne SET Nom = :nom, Prenom = :prenom, Age = :age WHERE id = :id";
$db = config::getConnexion();

try {
    $query = $db->prepare($sql);
    $query->bindParam(':nom', "Karim", PDO::PARAM_STR);
    $query->bindParam(':prenom', "Mohamed", PDO::PARAM_STR);
    $query->bindParam(':age', 24, PDO::PARAM_INT);
    $query->bindParam(':id', 1, PDO::PARAM_INT);
    $query->execute();
    echo $query->rowCount() . "records  updated successfully";
} catch (Exception $e) {
    die('Error:' . $e->getMessage());
}
?>
```

Manipulation de la BD

PDO: CRUD

CRU**D**: ‘Delete’ des données
L'instruction DELETE est utilisée pour supprimer des données d'une table.

```php
<?php
require '../config.php';
$sql = "DELETE FROM personne WHERE id = :id";
$db = config::getConnexion();
try {
    $query = $db->prepare($sql);
    $query->bindParam(1, $id, PDO::PARAM_INT);
    $query->execute();
    $rowCount = $query->rowCount();

    if ($rowCount > 0) {
        echo "Delete successful. $rowCount rows affected.";
    } else {
        echo "No rows deleted.";
    }
} catch (Exception $e) {
    die('Error:' . $e->getMessage());
}
?>
```

Merci de votre attention