CSE485 – WEB TECHNOLOGIES

PHP-MySQL, PHP-Ajax

Kieu Tuan Dung
Email: dungkt@tlu.edu.vn

Nội dung

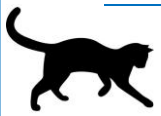
- PHP – Ajax:
  - jQuery Ajax
- PHP – PDO

1.jQuery Ajax: load()

- **load**(url, params, callback)
  - url: URL Tiếp nhận, xử lý và gửi lại dữ liệu.
  - Params: lưu giữ các biến cần gửi đi.
  - Callback: hàm mà nó sẽ gọi đến sau khi quá trình Ajax hoàn tất .

1.jQuery Ajax: load() …

```
1  $(document).ready(function( ){
2     $("div#result").load("result.php", {user: admin}, callback);
3  });
4  function callback(){
5     alert("Kết thúc quá trình.");
6  }
7  //Nội dung trong result.php
8  if(isset($_POST['user']))
9  {
10  echo 'Bạn đã gửi dữ liệu của người dùng = '.$_POST['user'].' thành công';
11 }else{
12  echo 'Không nhận được dữ liệu của người dùng nào';
13 }
```



1. jQuery Ajax: get()

- **get**(url, params, callback)
  - url: URL Tiếp nhận, xử lý và gửi lại dữ liệu.
  - Params: lưu giữ các biến cần gửi đi.
  - Callback: hàm mà nó sẽ gọi đến sau khi quá trình Ajax hoàn tất .

1. jQuery Ajax: get() …

```
1  $(document).ready(function( ){
2     $.get("result.php", {user: admin}, function(data){$("div#result").html(data);});
3  });
4  //Nội dung trong result.php
5  if(isset($_GET['user']))
6  {
7   echo 'Bạn đã gửi dữ liệu của người dùng = '.$_POST['user'].' thành công';
8  }else{
9   echo 'Không nhận được dữ liệu của người dùng nào';
10 }
```

1. jQuery Ajax: post()

- **post**(url, params, callback)
  - url: URL Tiếp nhận, xử lý và gửi lại dữ liệu.
  - Params: lưu giữ các biến cần gửi đi.
  - Callback: hàm mà nó sẽ gọi đến sau khi quá trình Ajax hoàn tất .

1. jQuery Ajax: post() …

```
$(document).ready(function( ){
   $.post("result.php", {user: admin}, function(data){$("div#result").html(data);});
});
//Nội dung trong result.php
if(isset($_POST['user']))
{
 echo 'Bạn đã gửi dữ liệu của người dùng = '.$_POST['user'].' thành công';
}else{
 echo 'Không nhận được dữ liệu của người dùng nào';
}
```

1. jQuery Ajax: ajax()

- Phương thức tổng quát phổ biến nhất
- Cho phép cấu hình và tùy chỉnh các thông số
  - $.ajax - fullfill send XMLHttpRequests
  - $.post – shorthand $.ajax
  - $.get - shorthand $.ajax
  - $.load is the same concept, but allows you to load the content into a selected element easily.

1. jQuery Ajax: ajax() …

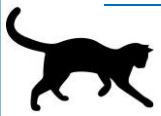
- var data = $('form#ID_form').serialize();

```
$.ajax({
  type: 'Loại gửi dữ liệu (POST hoặc GET)',
  url: 'URL Tiếp nhận, xử lý và gửi lại dữ liệu',
  data: { Các biến dữ liệu được gửi lên server.(ten_bien1:dữliệu,ten_bien2:dữliệu,...)
  dataType: Kiểu dữ liệu trả về ('html','text','json','xml'),
  success: function(data) {
    Nội dung sẽ được thực thi sau khi nhận được dữ liệu từ server
  },
  error: function() {
    Nội dung sẽ được thực thi khi có lỗi phát sinh
  }
});
```

2. PHP PDO

- So sánh PDO và MySQLi

	PDO	MySQLi
Database hỗ trợ	Hơn 12 loại	Chỉ hỗ trợ MySQL
API	Hướng đối tượng (OOP)	Hướng đối tượng (OOP) - Hướng thủ tục (Procedural)
Kết nối Database	Dễ dàng	Dễ dàng
Đặt tên tham số	Có	Không
Object Mapping	Có	Có
Prepared Statements	Có	Không
Hiệu năng	Cao	Cao
Stored Procedures	Có	Có



2. PHP PDO …

- **Kết nối MySQL**:
  - $conn = new PDO(\'mysql:host=localhost;dbname=music\', $username, $password);
- **Kết nối SQLite**:
  - $conn = new PDO("sqlite:your/database/path/music.db");
- **Hủy kết nối**: $conn=null

2. PHP PDO …

- **INSERT && UPDATE:**
  - **Prepare statement**: Chuẩn bị một câu lệnh SQL làm khung/mẫu được gọi là Prepared Statement với các Placeholder
  - **Bind params**: Gắn giá trị thực vào các placeholder
  - **Execute**: Thực thi câu lệnh.

2. PHP PDO …

- **Prepare statement: Unnamed Placeholder**

```
$stmt = $conn->prepare(\'INSERT INTO users (name, email, age) values (?, ?, ?)\');
//Gán các biến (lúc này chưa mang giá trị) vào các placeholder theo thứ tự tương ứng
$stmt->bindParam(1, $name);
$stmt->bindParam(2, $mail);
$stmt->bindParam(3, $age);

//Gán giá trị và thực thi
$name = “Nguyen Van A";
$mail = “anv@wru.vn";
$age = 20;
$stmt->execute();
```

2. PHP PDO …

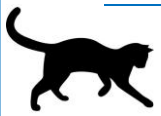
- **Prepare statement**: **Placeholder**

```
$stmt = $conn->prepare(\'INSERT INTO users (name, email, age) values (:name, :mail, :age)\');

//Gán các biến (lúc này chưa mang giá trị) vào các placeholder theo tên của chúng
$stmt->bindParam(\':name\', $name);
$stmt->bindParam(\':mail\', $mail);
$stmt->bindParam(\':age\', $age);

//Gán giá trị và thực thi
$name = “Nguyen Van A";
$mail = “anv@wru.vn";
$age = 20;
$stmt->execute();
```

2. PHP PDO …

- **Prepare statement: Sử dụng Mảng**

```
$stmt = $conn->prepare(\'INSERT INTO users (name, email, age) values (?, ?, ?)\');
$data = array(\'Nguyen Van A\', \'anv@wru.vn\', 22);

//Phương thức execute() dưới đây sẽ gán lần lượt giá trị trong mảng vào các Placeholder theo thứ tự
$stmt->execute($data);
```



2. PHP PDO …

- **Đọc dữ liệu từ Database**

```
$stmt = $conn->prepare(\'SELECT email, age from users WHERE name = :name\');

//Thiết lập kiểu dữ liệu trả về
$stmt->setFetchMode(PDO::FETCH_ASSOC);

//Gán giá trị và thực thi
$stmt->execute(array(\'name\' => \'a\'));

//Hiển thị kết quả, vòng lặp sau đây sẽ dừng lại khi đã duyệt qua toàn bộ kết quả
while($row = $stmt->fetch()) {
    echo $row[\'name\'] , \'\n\';
    echo $row[\'email\'] , \'\n\';
    echo $row[\'age\'] , \'\n\';
}
```

2. PHP PDO …

- **Xử lý ngoại lệ**

```
try {
    $stmt = new PDO(\'mysql:host=localhost;dbname=music\', \'root\', \'abcabc\');
    $stmt->setAttribute( PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION );

    //Sai cú pháp, FORM thay vì FROM
    $stmt->prepare(\'SELECT name FORM people\');
}
catch(PDOException $e) {
    echo "ERROR! Co loi xay ra voi PDO";
    file_put_contents(\'PDOErrors.txt\', $e->getMessage(), FILE_APPEND);
}
```

Hỏi / Đáp


THE END