Ajax 1: Ajax thuần

FDP 5.0

Nội dung

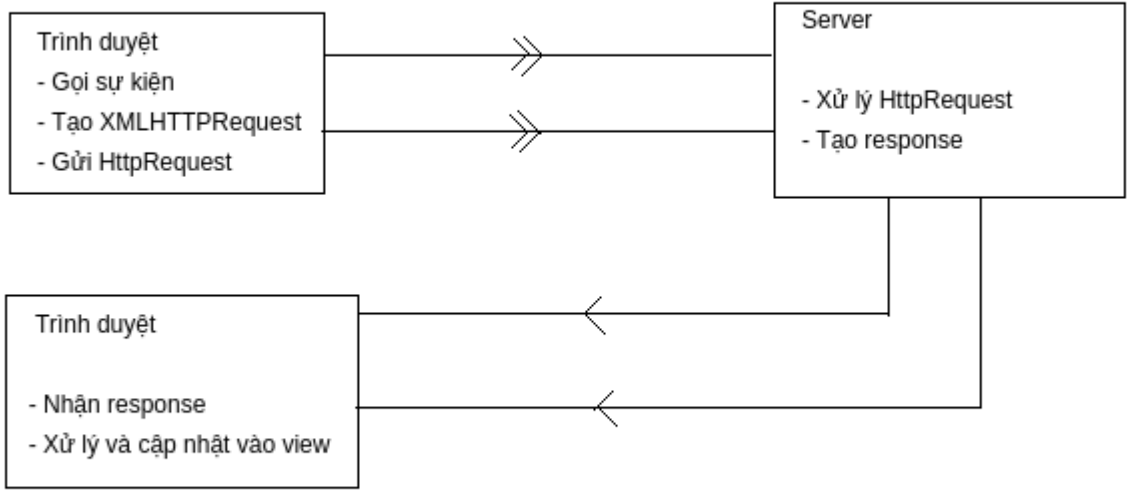
- Giới thiệu
- Tìm hiểu về XMLHttpRequest
- Tìm hiểu Ajax Request
- Tìm hiểu Ajax Response
- Các sự kiện trên XMLHttpRequest

GIỚI THIỆU

- AJAX không phải là một loại ngôn ngữ lập trình
- Là 1 kỹ thuật cho phép truy cập máy chủ lấy dữ liệu và xử lý phía client
- Viết tắt của Asynchronous JavaScript And XML
- Cho phép người dùng tương tác với server để lấy dữ liệu , thay đổi nội dung 1 thành phần trong trang mà không cần phải reload toàn bộ trang

GIỚI THIỆU

❖ Cách hoạt động của AJAX



GIỚI THIỆU

❖ Mô tả các thức hoạt động của AJAX

- Một sự kiện xảy ra trong một trang web (trang được tải, một nút được nhấp)
- Đối tượng XMLHttpRequest được tạo bởi JavaScript
- Đối tượng XMLHttpRequest gửi yêu cầu đến máy chủ web
- Máy chủ xử lý yêu cầu
- Máy chủ gửi phản hồi trở lại trang web
- Phản hồi được đọc bằng JavaScript
- Hành động thích hợp (như cập nhật trang) được thực hiện bởi JavaScript

Tìm hiểu về XMLHttpRequest

❖ Vì sao lại là XML ?

- Trước đây thời điểm AJAX được tạo ra thì việc xử dụng định dạng XML để nhận dữ liệu rất phổ biến.

❖ Đối tượng XMLHttpRequest

- Để sử dụng cần tạo ra đối tượng XMLHttpRequest
- Cú pháp ***variable*** **= new XMLHttpRequest();**

Tất cả các trình duyệt hiện đại (Chrome, Firefox, IE7 +, Edge, Safari Opera) đều có đối tượng XMLHttpRequest được tích hợp sẵn.

Tìm hiểu về XMLHttpRequest

Thí dụ: **var xhttp = new XMLHttpRequest();**

Các phiên bản cũ của Internet Explorer (IE5 và IE6) sử dụng đối tượng ActiveX thay vì đối tượng XMLHttpRequest:
***variable* = new ActiveXObject("Microsoft.XMLHTTP");**
Thí dụ: *var*
***xmlhttp* = new ActiveXObject("Microsoft.XMLHTTP");**

Tìm hiểu về XMLHttpRequest

```
if (window.XMLHttpRequest) {
    // code for modern browsers
    xmlhttp = new XMLHttpRequest();
 } else {
    // code for old IE browsers
    xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
}
```

Phương thức	Mô tả
new XMLHttpRequest()	Tạo một đối tượng XMLHttpRequest mới
abort()	Hủy yêu cầu hiện tại
getAllResponseHeaders()	Trả về tất cả thông tin header
getResponseHeader()	Trả về thông tin header được chỉ định
open(method,url,async,user,psw)	Các thông số đầu vào method: loại phương thức GET hoặc POST url: địa chỉ tập tin dữ liệu async: true (bất đồng bộ) or false (đồng bộ) user: tùy chỉnh thêm tên người dùng psw: tùy chỉnh thêm mật khẩu
send()	Gửi yêu cầu lên máy chủ Mặc định được gửi với phương thức GET
send(string)	Gửi yêu cầu lên máy chủ Sử dụng phương thức POST
setRequestHeader()	Thêm 1 cặp label/value vào header được gửi đi.

Thuộc tính	Mô tả
onreadystatechange	Xác định một hàm được gọi khi thuộc tính readyState thay đổi.
readyState	Giữ trạng thái của XMLHttpRequest. 0: yêu cầu không được khởi tạo 1: kết nối máy chủ được thiết lập 2: Request được nhận 3: Request đang được xử lý 4: Request hoàn tất dự liệu được nhận
responseText	Trả dữ liệu về dạng chuỗi
responseXML	Trả dữ liệu về dạng XML
status	Status trình duyệt 200: "OK" 403: "Forbidden“ – Bị chặn 404: "Not Found“ – Không tìm thấy For a complete list go to the Http Messages Reference
statusText	Trạng trái trả về ở dạng chữ (e.g. "OK" or "Not Found")

Tìm hiểu về AJAX Request

- **Sử dụng phương thức open() và send() của đối tượng XMLHttpRequest để gửi yêu cầu tới server:**

```
xhttp.open("GET", "ajax_info.txt", true);
xhttp.send();
```

Phương thức	Mô tả
open(*method, url, async*)	Chỉ định loại request được gửi đi *method*: loại request: GET or POST *url*: địa chỉ (tập tin) trên máy chủ *async*: true (bất đồng bộ) or false (đồng bộ)
send()	Gửi dữ liệu lên máy chủ (Sử dụng GET)
send(*string*)	Gửi dữ liệu lên máy chủ (Sử dụng POST)

Tìm hiểu về AJAX Request

- **Vậy khi nào thì sử dụng GET hoặc POST ?**
  - GET đơn giản và nhanh hơn POST và có thể được sử dụng trong hầu hết các trường hợp.
- **Tuy nhiên, luôn sử dụng POST khi:**
  - Tệp được lưu trong bộ nhớ cache không phải là một tùy chọn (cập nhật tệp hoặc cơ sở dữ liệu trên máy chủ).
  - Gửi một lượng lớn dữ liệu đến máy chủ (POST không có giới hạn về kích thước).
  - Gửi thông tin đầu vào của người dùng (có thể chứa các ký tự không xác định), POST mạnh mẽ và an toàn hơn GET.

Tìm hiểu về AJAX Request

- **Ví dụ GET**

  Không tham số

```
xhttp.open("GET", "demo_get.asp", true);
xhttp.send();
```

  Có tham số

```
xhttp.open("GET", "demo_get2.asp?fname=Henry&lname=
Ford", true);
xhttp.send();
```

Tìm hiểu về AJAX Request

❖ Ví dụ POST

Không truyền tham số.

```
xhttp.open("POST", "demo_post.asp", true);
xhttp.send();
```

Để gửi dữ liệu dạng form cần thêm header HTTP với setRequestHeader(). Sau đó chỉ định dữ liệu được gửi trong phương thức send();

```
xhttp.open("POST", "demo_post2.asp", true);
xhttp.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
xhttp.send("fname=Henry&lname=Ford");
```

Tìm hiểu về AJAX Request

❖ Synchronous Request

Tham số thứ 3 này mặc định sẽ là true. Nếu chọn false thì sẽ không cần thuộc tính onreadystatechange vì nó sẽ đợi server thực thi xong thì trả kết quả về. Nên sử dụng false trong trường hợp testing.

```
xhttp.open("GET", "ajax_info.txt", false);
xhttp.send();
document.getElementById("demo").innerHTML = xhttp.responseText;
```

Không nên sử dụng XMLHttpRequest đồng bộ (async = false) vì JavaScript sẽ ngừng thực thi cho đến khi phản hồi của máy chủ sẵn sàng. Nếu máy chủ bận hoặc chậm, ứng dụng sẽ bị treo hoặc dừng.

Hiện nay Synchronous XMLHttpRequest đang trong quá trình bị xóa khỏi tiêu chuẩn web, nhưng quá trình này có thể mất nhiều năm.

Tìm hiểu về AJAX Responce

- Một request gửi lên sẽ nhận được 1 response trả về thông qua thuộc tính **readyState**
- Thuộc tính **onreadystatechange** xác định một hàm sẽ được thực thi khi **readyState** thay đổi

Tìm hiểu về AJAX Responce

- Ví dụ

```
function loadDoc() {
    var xhttp = new XMLHttpRequest();
    xhttp.onreadystatechange = function() {
        if (this.readyState == 4 && this.status == 200) {
            document.getElementById("demo").innerHTML =
            this.responseText;
        }
    };
    xhttp.open("GET", "ajax_info.txt", true);
    xhttp.send();
}
```

Tìm hiểu về AJAX Responce

- Từ thời điểm khởi tạo đối tượng XMLHttpRequest đến lúc nhận dự liệu trả về từ máy chủ thì thuộc tính **readyState** trải qua 5 trạng thái
  - **UNSENT – 0**
  - **OPENED – 1**
  - **HEADERS RECEIVED – 2**
  - **LOADING – 3**
  - **DONE – 4**

Tìm hiểu về AJAX Responce

- Hàm **onreadystatechange** được gọi mỗi khi **readyState** thay đổi.
- Sự kiện **onreadystatechange** được kích hoạt bốn lần (1-4), một lần cho mỗi thay đổi trong readyState.
- Khi **readyState** là 4 và trạng thái là 200, phản hồi đã sẵn sàng:

```
function loadDoc() {
    var xhttp = new XMLHttpRequest();
    xhttp.onreadystatechange = function() {
        if (this.readyState == 4 && this.status == 200) {
            document.getElementById("demo").innerHTML =
            this.responseText;
        }
    };
    xhttp.open("GET", "ajax_info.txt", true);
    xhttp.send();
}
```

Tìm hiểu về AJAX Responce

- Nếu bạn có nhiều hơn một tác vụ AJAX trong một trang web, bạn nên tạo một hàm để thực thi đối tượng **XMLHttpRequest** và một hàm gọi lại cho mỗi tác vụ **AJAX**.
- Sử dụng hàm gọi lại (callback function) được truyền dưới dạng tham số cho một hàm khác .
- Ví dụ

```
loadDoc("url-1", myFunction1);

loadDoc("url-2", myFunction2);

function loadDoc(url, cFunction) {
  var xhttp;
  xhttp=new XMLHttpRequest();
  xhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
      cFunction(this);
    }
 };
  xhttp.open("GET", url, true);
  xhttp.send();
}

function myFunction1(xhttp) {
  // action goes here
}
function myFunction2(xhttp) {
  // action goes here
}
```

Tìm hiểu về AJAX Responce

❖ Dữ liệu máy chủ trả về sẽ có các dạng dưới đây

Thuộc tính	Mô tả
responseText	Dữ liệu trả về hiện thị ở dạng chuỗi
responseXML	Dữ liệu trả về hiện thị dạng XML

Ví dụ : document.getElementById("demo").innerHTML = xhttp.responseText;

```
xmlDoc = xhttp.responseXML;
txt = "";
x = xmlDoc.getElementsByTagName("ARTIST");
for (i = 0; i < x.length; i++) {
  txt += x[i].childNodes[0].nodeValue + "<br>";
  }
document.getElementById("demo").innerHTML = txt;
xhttp.open("GET", "cd_catalog.xml", true);
xhttp.send();
```

Tìm hiểu về AJAX Responce

- Ngoài ra chúng ta còn có thể lấy thông tin response header được trả về thông qua các thuộc tinhs au

Phương thức	Mô tả
getResponse Header()	Trả về thông tin header cụ thể từ máy chủ
getAllRespons eHeaders()	Trả về tất cả thông tin header từ máy chủ

Thank you and
happy learning !!!