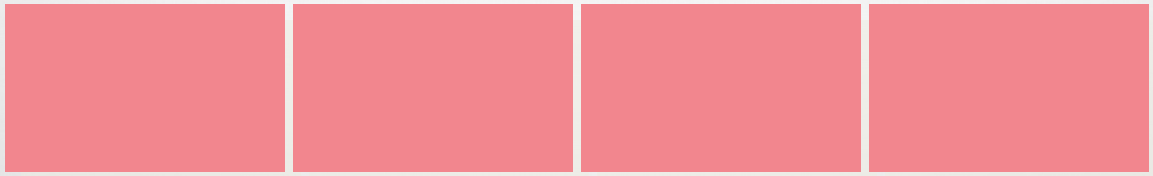
Bài giảng

PHÁT TRIỂN ỨNG DỤNG WEB

Lê Đình Thanh
Khoa Công nghệ Thông tin
Trường Đại học Công nghệ, ĐHQGHN
E-mail: thanhld@vnu.edu.vn Mobile: 0987.257.504

Bài 8

AJAX



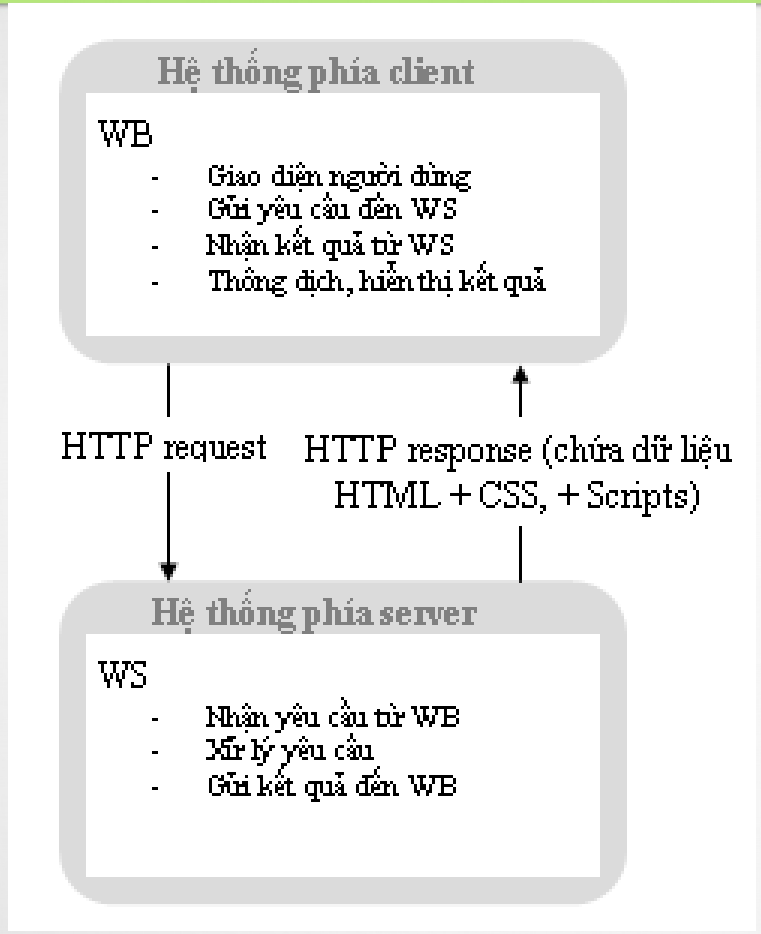
Nội dung

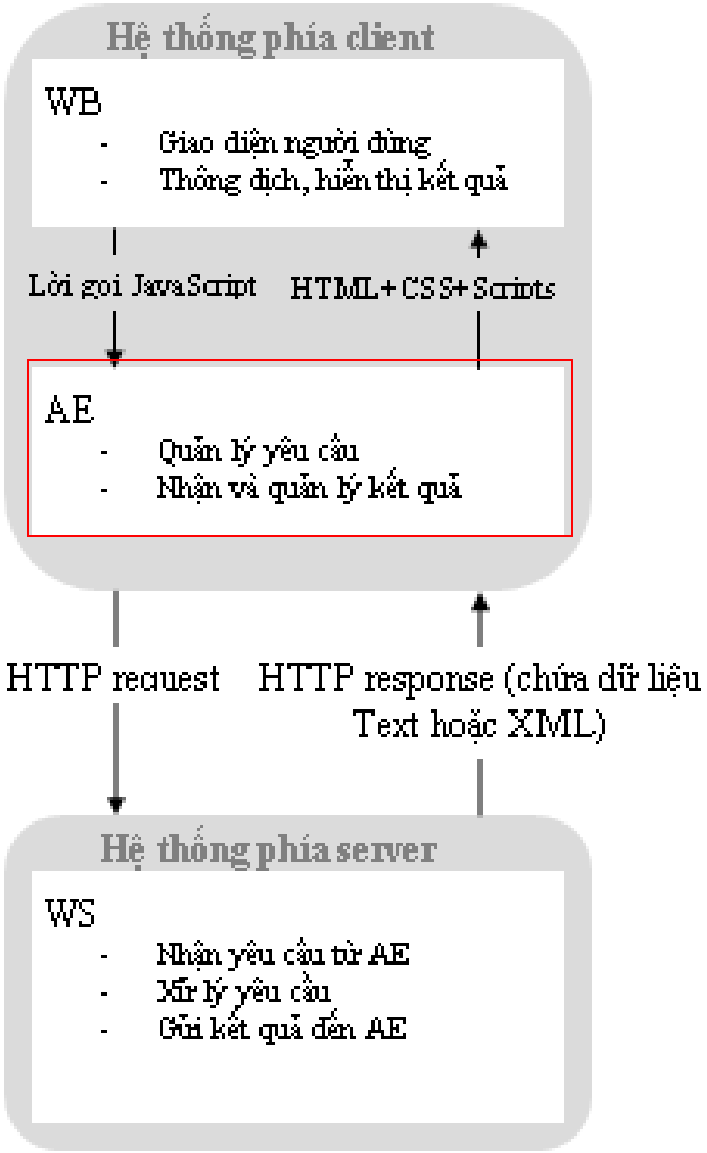
- AJAX là gì?
- Hoạt động của ứng dụng web với Ajax
- So sánh web truyền thống với Ajax web
- Các trình duyệt hỗ trợ Ajax
- Ajax engine
- Sử dụng Ajax gửi/nhận text
- Sử dụng Ajax gửi/nhận xml
- Sử dụng thư viện jQuery để xử lý ajax

AJAX là gì?

- AJAX (Asynchronous Javascripts and XML) là một kỹ thuật kết hợp một số công nghệ web để xây dựng các ứng dụng web mà tương tác giữa người dùng với ứng dụng được thực hiện không đồng bộ. Các công nghệ bao gồm:
  - Hiển thị dựa trên chuẩn sử dụng HTML và CSS
  - Tương tác động sử dụng DOM
  - Trao đổi và xử lý dữ liệu sử dụng XML, text
  - Thu nhận dữ liệu không đồng bộ sử dụng XMLHttpRequest
  - Kết hợp các công nghệ sử dụng JavaScript

Web truyền thống <> Ajax Web

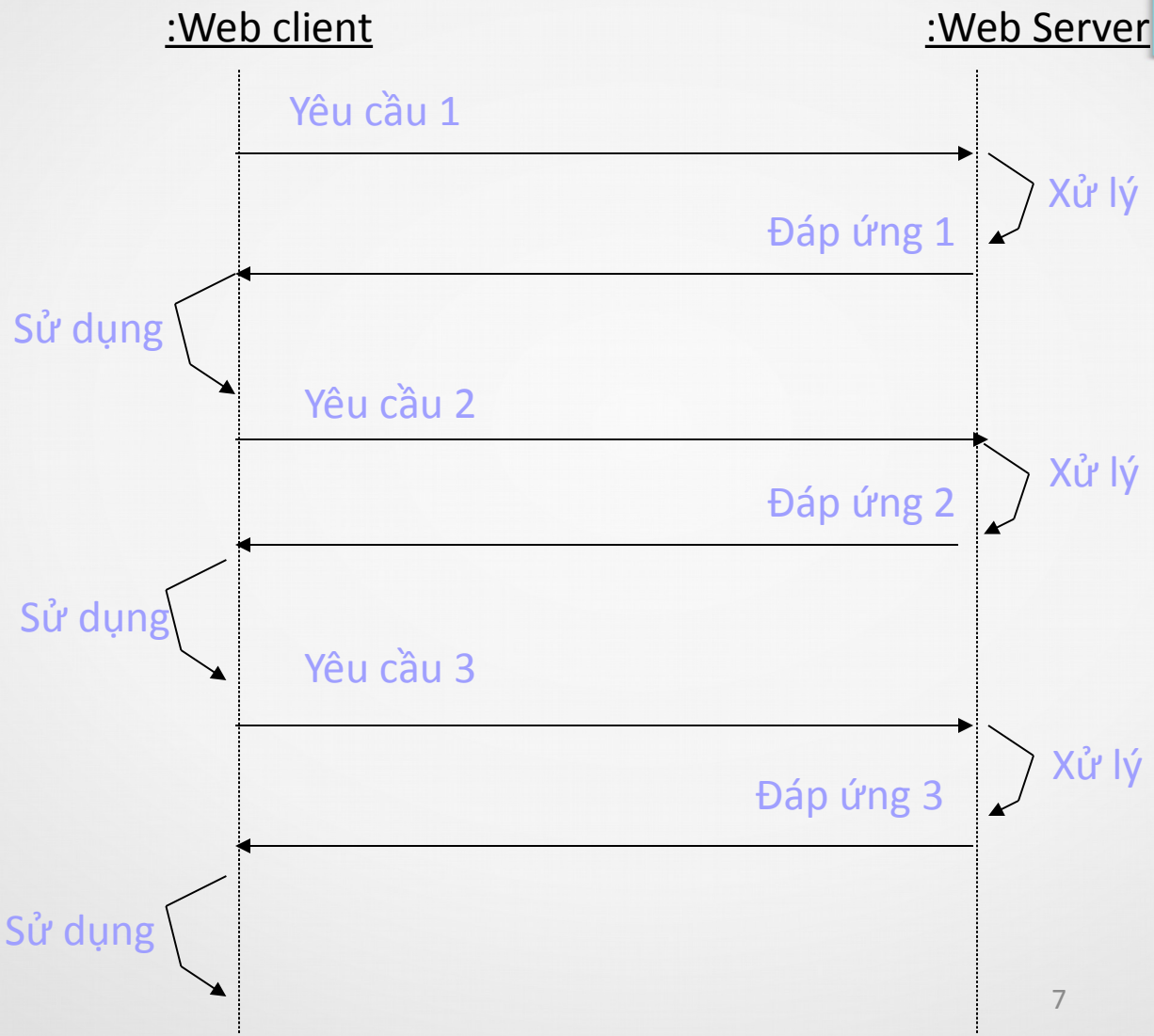




Web truyền thống

- Yêu cầu của người dùng được gửi trực tiếp từ browser đến Web server thông qua HTTP request
- Khi nhận được HTTP request, Web server xử lý yêu cầu, sinh ra trang HTML mới, rồi gửi toàn bộ trang HTML (chứa HTML và CSS) mới đến browser. Browser xóa trang cũ và hiển thị trang mới.
- Từ khi gửi yêu cầu đi, người dùng *không được làm thêm bất kỳ thao tác gì* cho đến khi trang HTML mới được gửi đến client: mỗi yêu cầu phải được giải quyết dứt điểm trước khi có yêu cầu tiếp theo = đồng bộ.

Hoạt động của web truyền thống



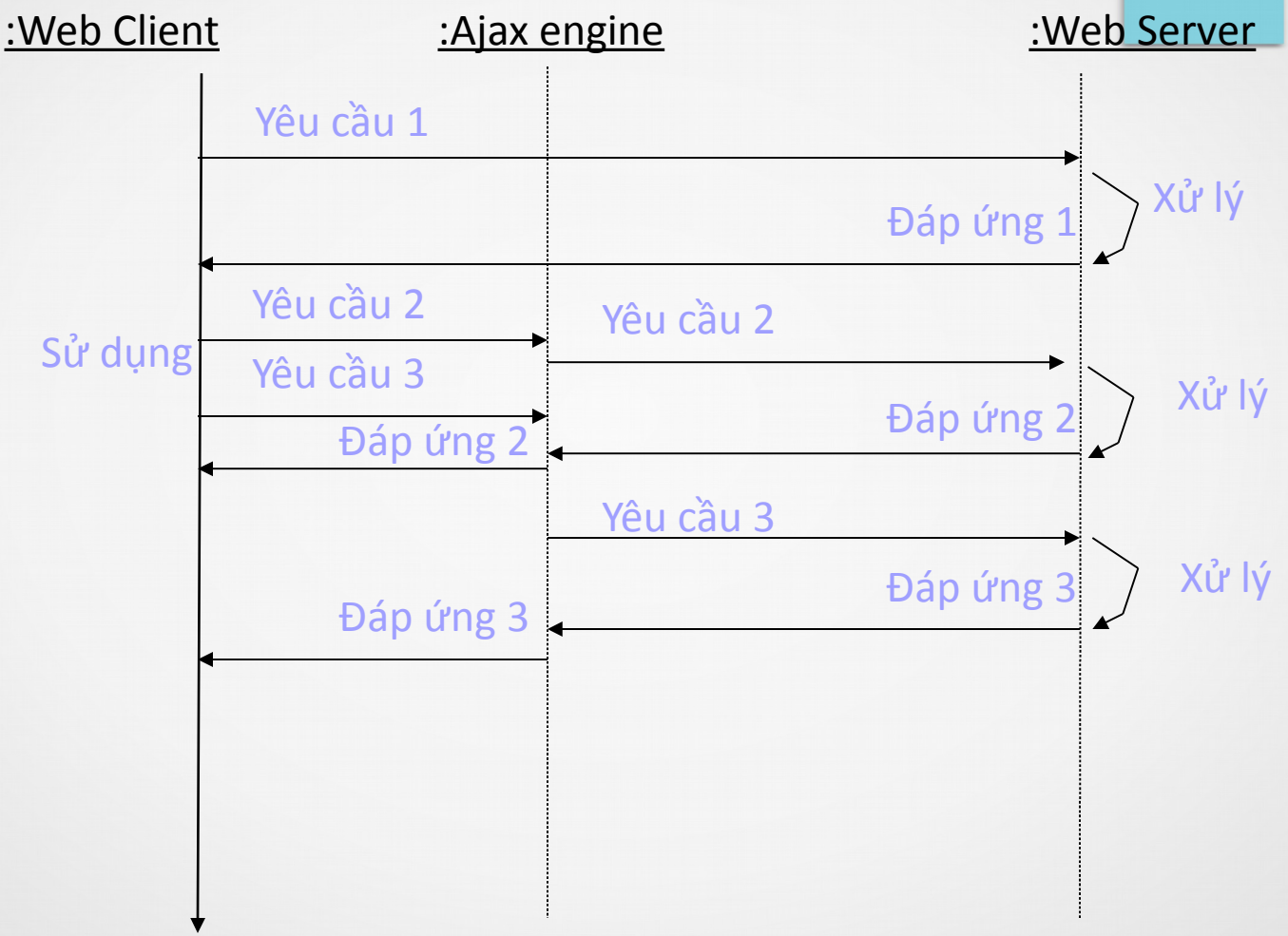
Web truyền thống: Hạn chế

- Khi người dùng thao tác thì server "nghỉ" và ngược lại
  - Lãng phí thời gian, hiệu quả sử dụng thấp
  - Người dùng phải vừa làm vừa đợi: gửi yêu cầu $\rightarrow$ đợi $\rightarrow$ nhận kết quả $\rightarrow$ gửi yêu cầu $\rightarrow$ đợi $\rightarrow$ ... $\Rightarrow$ Người dùng phải đợi lâu nếu yêu cầu xử lý lớn và server mất nhiều thời gian xử lý + Hiển thị không liên tục, "nhấp nháy" gây khó chịu (! HCI).
- Toàn bộ trang HTML mới được gửi từ server đến client
  - Không cần thiết vì có thể nhiều chi tiết trên trang mới vẫn như trang cũ
  - Lượng thông tin trao đổi giữa client-server lớn $\Rightarrow$ chi phí truyền thông (thời gian, băng thông) lớn.

Ajax Web

- Ajax engine được cài trên client, làm nhiệm vụ giao tiếp trung gian giữa browser với web server
  - Browser gửi yêu cầu đến Ajax engine bằng lời gọi Javascript.
  - Ajax engine chuyển yêu cầu của Client thành HTTP request và gửi cho web server
  - Web server xử lý yêu cầu rồi gửi kết quả cho Ajax engine ở dạng XML
  - Ajax engine biên dịch XML thành HTML và gửi HTML cho browser
- Một yêu cầu của người dùng chưa cần được giải quyết xong thì người dùng đã có thể đưa ra yêu cầu khác
  - Trao đổi giữa Browser với Ajax engine và giữa Ajax engine với Server để thực hiện các yêu cầu diễn ra không đồng bộ.

Hoạt động của Ajax web



Ajax Web: Ưu điểm

- Người dùng và server thực hiện một cách song hành
  - Không lãng phí thời gian, hiệu quả sử dụng cao
  - Người dùng không phải vừa làm vừa đợi
  - Hiển thị liên tục, không gây khó chịu (HCI).
- Chỉ phần khác biệt của trang mới so với trang cũ mới được gửi từ server đến client
  - Lượng thông tin trao đổi giữa client-server tối thiểu $\Rightarrow$ tiết kiệm chi phí (thời gian, băng thông) truyền thông.

Vì sao dùng Ajax

- Để tạo ra các ứng dụng web mà giao tiếp của nó với người dùng diễn ra như giao tiếp của ứng dụng Winform với người dùng: liên tục.
- hiệu quả trong sử dụng và trong truyền thông

Sử dụng AJAX

- Sử dụng Ajax Engine (đối tượng Javascript XMLHttpRequest) để gửi yêu cầu đến server và lấy dữ liệu về từ server.
- Sau khi XmlHttpRequest nhận được dữ liệu từ server, sử dụng javascript để sửa đổi trang web trên client với dữ liệu mới nhận được.

Lấy đối tượng XMLHttpRequest

```
function getXmlHttpObject()  {
     var xmlHttp = null;
 try    {
                    // Firefox, Opera 8.0+, Safari
                     xmlHttp = new XMLHttpRequest();
       }  catch (e) {
                   // Internet Explorer
                     try  {
                         xmlHttp = new ActiveXObject("Msxml2.XMLHTTP");
                         }  catch (e) {
                              try {
                                   xmlHttp=new
   ActiveXObject("Microsoft.XMLHTTP");
                                  } catch (e)  {
                                      alert("Trinh duyet khong ho tro
   AJAX!");
                                  }
                          }
       }
   return xmlHttp;
 }
```

XMLHttpRequest::readyState

```
if(xmlHttp.readyState==4)
    {
   // Đã nhận đủ trả lời từ
    server
   if (xmlHttp.status ==
    200) {
   //Đã thực hiện thành
    công trên server
   //Dùng javascript để sửa
    đổi trang
    }
}
```

ready State	Ý nghĩa
0	Chưa thiết lập yêu cầu
1	Đã thiết lập yêu cầu
2	Đã gửi yêu cầu
3	Đang trả lời
4	Đã trả lời xong

XMLHttpRequest.onreadystatechange

Là một con trỏ hàm không đối, được kích hoạt mỗi khi thuộc tính readyState thay đổi.

```
xmlHttp.onreadystatechange = tenHamXuly;
function tenHamXuly() {}
```

Hoặc

```
xmlHttp.onreadystatechange = function() {
    //Noi dung xu ly
}
```

Gửi yêu cầu lên server theo GET

xmlHttp.open(“GET”, path?querystring, asynchronous);
xmlHttp.send(null);

Ví dụ:

```
xmlHttp.open("GET", "time.php?zone=7", true);
xmlHttp.send(null);
```

Gửi yêu cầu lên server theo POST

```
xmlHttp.open(“POST”, url, asynchronous);
xmlHttp.setRequestHeader("Content-Type",
    "application/x-www-form-urlencoded");
xmlHttp.send(params);
```

Ví dụ:

```
xmlHttp.open(“POST", " time.php", true);
xmlHttp.setRequestHeader("Content-Type",
    "application/x-www-form-urlencoded");
xmlHttp.send(“zone=7”);
```

XMLHttpRequest.responseText

Nội dung dạng text/html do server gửi về.

Muốn sử dụng thuộc tính này, server phải thiết lập thuộc tính ContentType của trang là text/html (mặc định)

Server gửi dữ liệu dạng text

```
$time = 100;
echo $time;
```

Trình duyệt nhận và xử lý dữ liệu dạng text

```
xmlHttp.onreadystatechange = function() {
if(xmlHttp.readyState==4 &&
   xmlHttp.status==200)
   {       document.write(xmlHttp.responseText);
 }
}
```

XMLHttpRequest. responseXML

Nội dung dạng XML do server gửi về.

Muốn sử dụng thuộc tính này, server phải thiết lập thuộc tính ContentType của trang là text/xml

Server gửi dữ liệu dạng XML

```
echo "<?xml version='1.0' encoding='UTF-8'?>";
echo "<company>";
echo "<compname>$companyname</compname>";
echo "</company>";
```

Trình duyệt nhận và xử lý XML

```
xmlHttp.onreadystatechange = function() {
   if (xmlHttp.readyState == 4 && xmlHttp.status=200)    {
      var xmlDoc=xmlHttp.responseXML.documentElement;
      document.getElementById("companyname").innerHTML
      =  xmlDoc.getElementsByTagName("compname")
     [0].childNodes[0].nodeValue;
   }
}
```

Sử dụng thư viện jQuery để xử lý ajax

get, post, load, ajax

jQuery AJAX Get

$.get(path?querystring, *[callback]*);

- Gửi yêu cầu lên server theo phương thức GET và nhận kết quả về theo AJAX, sau đó thực hiện hàm callback
  - path: Địa chỉ tệp được yêu cầu
  - callback: Hàm được gọi sau khi tải xong. Hàm có hai tham số là *nội dung, trạng thái* trả về

Ví dụ jQuery AJAX Get

- Yêu cầu nội dung trang "clone.php?v1=100&v2=101“ với các tham số theo phương thức GET, đặt nội dung nhận được cho đối tượng tài liệu có định danh test1.

```
$.get("clone.php?v1=100&v2=101",
    function (data, status) {
        $("#test1").html(data);
    });
```

jQuery AJAX Post

$.post(*URL, [data], [callback]*);

- Gửi yêu cầu lên server theo phương thức POST và nhận kết quả về theo AJAX, sau đó thực hiện hàm callback
  - url: Địa chỉ tệp được yêu cầu
  - data: Các cặp tham số/giá trị được gửi
  - callback: Hàm được gọi sau khi tải xong. Hàm có hai tham số là *nội dung, trạng thái* trả về

Ví dụ jQuery AJAX Post

- Yêu cầu nội dung trang "clone.php” với các tham số v1=10, v2=12 theo phương thức POST, đặt nội dung nhận được cho đối tượng tài liệu có định danh test2.

```
$.post("clone.php", {v1: "10", v2: "12”},
   function (data, status) {
   $("#test2").html(data);
      });
```

jQuery AJAX Load

$(*selector*).load(*URL, [data], [callback]*);

- Tải nội dung từ URL và đặt vào đối tượng được chọn (Tương đương gửi yêu cầu lên server theo phương thức POST và nhận kết quả về theo AJAX, sau đó đặt kết quả trả về vào innerHTML của đối tượng được chọn), sau đó thực hiện hàm callback
  - url: Địa chỉ tệp được yêu cầu
  - data: Các cặp tham số/giá trị được gửi cùng url
  - callback: Hàm được gọi sau khi tải xong. Hàm có ba tham số là *nội dung, trạng thái* trả về và đối tượng *XMLHttpRequest* đã thực hiện các công việc của hàm load

Ví dụ jQuery AJAX Load

- Tải nội dung tệp văn bản “text.txt” và đặt vào innerHTML của đối tượng có định danh test3

```
$("#test3").load("text.txt");
```

  - Tải nội dung tại “clone.php?v1=8&v2=9” và đặt vào innerHTML của đối tượng có định danh test4

```
$("#test4").load("clone.php", {v1: "8", v2: "9” });
```

Ví dụ jQuery AJAX Load

- Tải nội dung tệp văn bản “text.txt” và đặt vào innerHTML của đối tượng có định danh test3

```
$("#test3").load("text.txt");
```

  - Tải nội dung tại “clone.php?v1=8&v2=9” và đặt vào innerHTML của đối tượng có định danh test4

```
$("#test4").load("clone.php", {v1: "8", v2: "9” });
```

jQuery ajax

- $.ajax(url [, settings]);
- Settings:
  - type: POST/GET/HEAD, …
  - data: {Các cặp tham số/giá trị}
  - contentType: Kiểu nội dung và mã hóa được sử dụng
  - dataType: Kiểu dữ liệu muốn nhận về từ server (xml, json, script, html)
  - success: function (data, status, jqXHR) { },
  - error: function (jqXHR, status, errorThrown) {}
  - …

Thực hành kỹ thuật AJAX

Ngày nay, những ứng dụng web cao cấp (như các trang của Google) đều được làm theo kỹ thuật AJAX.

Để sử dụng tốt kỹ thuật AJAX

Nắm vững nội dung một trang web

Hiểu rõ vai trò "trình thông dịch" của web browser

Hiểu mô hình đối tượng tài liệu DOM

Sử dụng javascript để truy cập các đối tượng HTML

Hiểu về cấu trúc tài liệu XML

Hiểu về cơ chế truyền thông giữa web server với ajax engine.

Tiếp theo

Viết lại URL