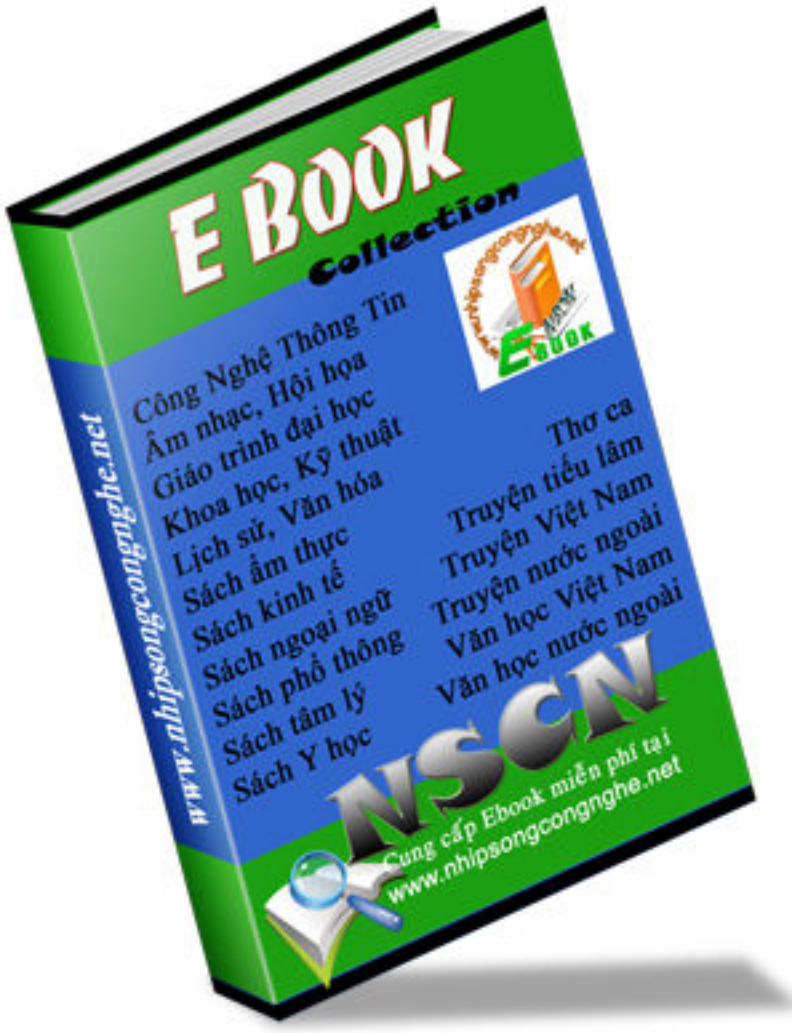

KĨ THUẬT LẬP TRÌNH AJAX

AJAX là một trong những từ ngữ thời thượng bậc nhất hiện nay trong giới công nghệ thông tin và là bí quyết kĩ thuật đứng đằng sau các ứng dụng lớn thuộc Google, Flickr, GMail, Google Suggest, Google Maps. Mặc dù cái tên này mới xuất hiện được vài tháng, InfoWorld số tháng 5/2005 đã đưa ra nhận định: Ajax đang tạo nên cuộc cách mạng trong thế giới web.

AJAX LÀ GÌ?

JavaScript, ngôn ngữ lập trình chạy trên trình duyệt đã quá quen thuộc với thế giới web kể từ khi Netscape phát minh ra nó. Sự phát triển của công nghệ và nhu cầu người sử dụng ngày càng cao buộc các nhà phát triển tạo ra một kĩ thuật khác cho phép xử lý các tác vụ phức tạp hơn. Tháng 2/2005, trên Internet bắt đầu lan truyền thuật ngữ Ajax như là một kỹ thuật mới cho ứng dụng web. Những thành công vang dội và sự hấp dẫn kì lạ của Gmail, Google Suggest và Google Maps đã khiến cho Ajax được chú ý một cách đặc biệt.

Ajax là viết tắt của Asynchronous JavaScript and XML - kỹ thuật kết hợp hai tính năng mạnh của JavaScript được các nhà phát triển đánh giá rất cao:

• Gửi yêu cầu (request) đến máy chủ mà không cần nạp lại trang

• Phân tách và làm việc với XML

Các ứng dụng Ajax xoay quanh một tính năng có tên là XMLHttpRequest. Các kĩ sư của dự án Mozilla bắt đầu hỗ trợ tính năng này ở bản Mozilla 1.0 (và Netscape 7). Apple cũng đã thực hiện một tính năng tương tự kể từ Safari 1.2.

Ajax là sự phối hợp một loạt các công nghệ đang thu hút sự quan tâm của giới công nghiệp trong thời gian gần đây. Đó là:

• Trình bày trang web dựa trên XHTML và CSS, các chuẩn của W3C, được Firefox (Mozilla), Safari (Apple), Opera, Netscape 8.0 (nhân Firefox) hỗ trợ rất tốt;

• Biểu diễn động và tương tác sử dụng Document Object Model, chuẩn của W3C;

• Trao đổi và xử lý dữ liệu dùng XML và XSLT, chuẩn của W3C;

• Thu hồi dữ liệu bất đối xứng dùng XMLHttpRequest;

• Dùng JavaScript để liên kết mọi thứ lại với nhau. JavaScript ở đây là ECMAScript, chuẩn của ECMA, không phải là JScript của Microsoft.

Ở các ứng dụng web truyền thống, client sẽ gửi HTTP Request đến web server và web server sẽ gửi trả response chứa các thông tin dưới dạng HTML và CSS. Ajax cho phép tạo ra một Ajax Engine nằm giữa giao tiếp này. Khi đó, các yêu cầu gửi resquest và nhận response do Ajax Engine thực hiện. Thay vì trả dữ liệu dưới dạng HTML và CSS trực tiếp cho trình duyệt, web server có thể gửi trả dữ liệu dạng XML và Ajax Engine sẽ tiếp nhận, phân tách và chuyển hóa thành XHTML+CSS cho trình duyệt hiển thị. Việc này được thực hiện trên client nên giảm tải rất nhiều cho server, đồng thời người sử dụng cảm thấy kết quả xử lý được hiển thị tức thì mà không cần nạp lại trang. Mặt khác, sự kết hợp của các công nghệ web như CSS và XHTML làm cho việc trình bày giao diện trang web tốt hơn nhiều và giảm đáng kể dung lượng trang phải nạp. Đây là những lợi ích hết sức thiết thực mà Ajax đem lại.

LẬP TRÌNH AJAX

Xử lý HTTP Request

Để gửi một HTTP Request đến server bằng JavaScript, bạn cần tạo một đối tượng của lớp cung cấp tính năng này. Trong IE thì lớp này tồn tại dưới dạng một đối tượng ActiveX có tên là XMLHTTP. Đối tượng này có từ IE 4.0.

```
var httpRequest = new ActiveXObject("Microsoft.XMLHTTP");
```

Nếu MSXML được cài đặt thì bạn cũng có thể gọi:

```
var httpRequest = new ActiveXObject("Msxml2.XMLHTTP");
```

Ở Mozilla, Firefox, Opera 8.0, Safari và các trình duyệt khác thì lớp này có tên là XMLHttpRequest. Đối tượng XMLHttpRequest không phải là một chuẩn của W3C (tương lai có thể được W3C chấp thuận). Đối tượng XMLHttpRequest được hỗ trợ ở IE 5.0+, Safari 1.2+, Mozilla 1.0+/ Firefox, Opera 8.0+ và Netscape 7+.

```
var httpRequest = new XMLHttpRequest();
```

Do sự khác biệt này, nên để ứng dụng của bạn chạy trên nhiều trình duyệt thì bạn có thể làm như sau:

```
if (window.XMLHttpRequest) { // Mozilla, Safari, ...

httpRequest = new XMLHttpRequest();

} else if (window.ActiveXObject) { // IE

httpRequest = new ActiveXObject("Microsoft.XMLHTTP");

}
```

Do ActiveX trên IE rất nguy hiểm cho người dùng nên nhiều trường hợp tính năng này được mặc định cấm, vì vậy bạn cần kiểm tra trình duyệt của người sử dụng trước khi gọi

đối tượng XMLHTTP. Việc kiểm tra này được thực hiện qua giá trị của window.ActiveXObject. Ví dụ:

```
if (window.ActiveXObject) {

xmlhttp=new ActiveXObject("Microsoft.XMLHTTP")

}

else { ...}
```

Một số phiên bản của trình duyệt Mozilla không làm việc đúng khi nhận đáp trả từ server không có header chứa XML mime-type. Để khắc phục vấn đề này, bạn có thể sử dụng phương thức định nghĩa lại phần header do server gửi đến trong trường hợp nó không phải là text/xml.

```
httpRequest = new XMLHttpRequest();

httpRequest.overrideMimeType(text/xml);
```

Tiếp theo, bạn cần xác định muốn làm gì sau khi nhận được đáp trả (response) từ server. Giai đoạn này, bạn chỉ cần báo cho đối tượng HTTP request biết hàm JavaScript sẽ làm công việc xử lý đáp trả. Điều này được thực hiện bằng cách thiết lập thuộc tính onreadystatechange của đối tượng thành tên của hàm javascript:

```
httpRequest.onreadystatechange = nameOfTheFunction;
```

Chú ý không dùng cặp ngoặc đơn () sau tên hàm và không truyền tham số vào hàm đó. Thêm nữa, thay vì trao tên hàm thì bạn có thể sử dụng kĩ thuật định nghĩa hàm động:

```
httpRequest.onreadystatechange = function() {

// do the thing

};
```

Sau khi đã khai báo những gì sẽ diễn ra lúc nhận được response, bạn tiến hành gửi request. Bạn cần gọi các phương thức open() và send() của lớp HTTP request:

```
httpRequest.open(GET, http://www.example.org/some.file, true);

httpRequest.send(null);
```

Tham số đầu tiên của lời gọi đến open() là phương thức HTTP Request – GET, POST, HEAD hay bất cứ phương thức nào mà bạn muốn sử dụng và phương thức đó cần được server hỗ trợ. Chú ý viết hoa theo quy định của chuẩn HTTP, nếu không thì một số trình duyệt như Firefox có thể không xử lý request.

Tham số thứ hai là địa chỉ URL của trang mà bạn gửi request đến. Do thiết lập bảo mật, bạn không thể gọi các trang trên tên miền của bên thứ ba. Chú ý là nếu bạn không gọi đúng tên miền trên tất cả các trang thì bạn sẽ nhận được thông báo permission denied khi gọi open().

Tham số thứ ba xác lập request có bất đối xứng hay không (asynchronous). Nếu TRUE, thì việc thực thi hàm JavaScript sẽ tiếp tục trong khi response của server chưa đến. Đây là chữ A trong AJAX.

Tham số gửi đến phương thức send() có thể là bất cứ dữ liệu nào mà bạn muốn gửi tới server nếu bạn dùng phương thức POST để gửi request. Dữ liệu sẽ có dạng của một chuỗi truy vấn:

name=value&anothername=othervalue&so=on

Đối tượng XMLHttpRequest có một tập các thuộc tính dùng chung trên tất cả các môi trường hỗ trợ. Dưới đây là danh sách các thuộc tính chủ yếu của đối tượng này.

Xử lý Server Response

Chú ý khi gửi request, bạn cung cấp tên của hàm JavaScript được thiết kế để xử lý response.

```
httpRequest.onreadystatechange = nameOfTheFunction;
```

Hãy xem hàm này nên làm gì. Đầu tiên, hàm cần kiểm tra trạng thái của request. Nếu trạng thái có giá trị là 4, nghĩa là ứng dụng của bạn đã nhận được response đầy đủ từ server và đó là dấu hiệu tốt để bạn tiếp tục xử lý nó.

```
if (httpRequest.readyState == 4) {

// không xảy ra vấn đề gì và bạn đã nhận được response

} else {

// chưa sẵn sàng

}
```

Tiếp theo cần kiểm tra mã trạng thái của HTTP server response. Tất cả các mã có thể thàm khảo ở site của W3C. Trong bài viết này, chúng ta quan tâm đến response 200 (OK).

```
if (httpRequest.status == 200) {

// trạng thái response trả lại dấu hiệu tốt!

} else {

// có vấn đề trong việc tiếp nhận và xử lý request,

// ví dụ 404 (Not Found)

// hay 500 (Internal Server Error)

}
```

Sau khi đã kiểm tra trạng thái của request và mã trạng thái của HTTP response, việc xử lý dữ liệu mà server gửi tới tùy ở bạn. Bạn có hai lựa chọn:

• httpRequest.responseText – trả lại dưới dạng chuỗi văn bản

• httpRequest.responseXML – trả lại dưới dạng đối tượng XMLDocument và bạn có thể duyệt bằng cách sử dụng các hàm JavaScript DOM

Ví dụ với Text Response

Chúng ta sẽ xây dựng một ví dụ đơn giản. Các mã JavaScript gửi request đến một trang HTML, test.html. Trang này chứa dòng "Im a test", sau đó chúng ta dùng alert() để gửi ra nội dung của file đó.

```
<script type="text/javascript" language="javascript">

var httpRequest = false; //Ban đầu chưa có request

function makeRequest(url) {

httpRequest = false;

if (window.XMLHttpRequest) { // Kiểm tra hỗ trợ đối tượng XMLHttpRequest trên Mozilla, Safari,...

httpRequest = new XMLHttpRequest();

if (httpRequest.overrideMimeType) {
```

```
httpRequest.overrideMimeType(text/xml);
}
} else if (window.ActiveXObject) { // Trên IE, kiểm tra xem ActiveX có bị disable
try {
httpRequest = new ActiveXObject("Msxml2.XMLHTTP");
} catch (e) {
try {
httpRequest = new ActiveXObject("Microsoft.XMLHTTP");
} catch (e) {
//Xử lý khác của bạn
}
}
}
if (!httpRequest) {
alert(Giving up :( Cannot create an XMLHTTP instance);
return false;
}
httpRequest.onreadystatechange = alertContents;
httpRequest.open(GET, url, true);
httpRequest.send(null);
}
function alertContents() {
if (httpRequest.readyState == 4) {
```