Javascript Frameworks(Libraries)

NGUYỄN THỊ THÙY LIÊN

KHOA CNTT – ĐH PHENIKAA

LIEN.NGUYENTHITHUY@PHENIKAA-UNI.EDU.VN

Giới thiệu

JQuery

Prototype

Mootools

YUI

Ext JS

Giới thiệu - JQuery

Phổ biến nhất hiện nay

Sử dụng CSS selectors để truy cập đến các phần tử HTML, các đối tượng DOM trên trang web

Cung cấp nhiều hàm tiện ích, nền tảng giao diện người dùng, cùng nhiều plug-ins

Nhiều website lớn sử dụng Jquery:

- Google
- Microsoft
- IBM

Giới thiệu - Prototype

Prototype là một thư viện JavaScript cung cấp một API đơn giản để thực hiện nhiệm vụ web phổ biến.

API viết tắt của Application Programming Interface. Đây là một thưu viện các thuộc tính và phương thức cho các đối tượng HTML DOM

Prototype phát triển JavaScript bằng cách cung cấp các lớp và kế thừa

Giới thiệu - Mootools

Mootools cùng là một framework cung cấp API giúp lập trình JavaScript thông thường dễ dàng hơn

Mootools bao gồm nhiều hàm hiệu ứng ảnh động nhẹ, dễ sử dụng.

JQuery

jQuery là gì?

jQuery là thư viện Javascript rất nhẹ “write less, do more”

Được xây dựng với mục đích giúp sử dụng Javascript trên website một cách dễ dàng hơn

Thay thế nhiều dòng lệnh JavaScript bởi một số dòng lệnh đơn giản

Giúp đơn giản hóa việc sử dụng các thành phần, công việc phức tạp như gọi AJAX, hay thao tác với DOM

Đặc điểm

Thao tác với các đối tượng HTML/DOM

Thao tác thay đổi chỉnh sửa CSS

Lập trình xử lý sự kiện

AJAX

Và các tiện ích

Có rất nhiều plugin tiện ích đã được xây dựng

Sử dụng jQuery cho website

Download jQuery

- Download thư viện jQuery từ jQuery.com
- 2 phiên bản
  - Production version : có dung lượng nhỏ.
  - Development version: dùng cho testing và development

Jquery CDN (Content Delivery Network)

- Google

```
<head>
<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.1/jquery.min.js"></script>
</head>
```

- Microsoft

```
<head>
<script src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-1.11.1.min.js"></script>
</head>
```

Cú pháp

$(selector).action()

- $: định nghĩa/truy cập jQuery
- (selector): chọn/tìm các phần tử HTML
- action(): hành động được thực hiện trên selector

Ví dụ:

```
$(this).hide()
$("p").hide()
$(".test").hide()
$("#indentify").hide()
```

Jquery Selector

Được sử dụng để tìm/chọn các phần tử HTML dựa rên id, class, type, attribute, giá trị của attribute

Tất cả các selector bắt đầu bằng dấu $: $()

Selector dựa trên tên phần tử HTML: $(“**p**”)

Selector dựa trên id: $(“**#test**”)

Selector dựa trên tên lớp: $(“**.test**”)

Jquery Selector

Selector	Ví dụ	Mô tả
*	$("*")	Chọn tất cả các phần tử
this	$(this)	Chọn phần tử hiện tại
.class .class, .class	$("p.intro") $(".intro, .demo")	Chọn tất cả các thẻ p có class =“intro” Chọn tất cả các thẻ có class = intro hoặc class = demo
element	$(p)	Chọn tất cả thẻ p
:first :last	$("p:first") $("ul li:first")	Chọn thẻ p đầu tiên Chọn thẻ li đầu tiên trong thẻ ul đầu tiên
:even :odd	$("tr:even") $("tr:odd")	Chọn tất cả các thẻ tr chẵn Chọn tất các thẻ tr lẻ
:first-child	$("ul li:first-child")	Chọn thẻ li đầu tiên trong tất cả các thẻ ul
:first-of-type	$(“p:first-of-type”)	Tất cả các thẻ p mà các thẻ p đó là thẻ p đầu tiên hoặc duy nhất của cha nó
:last-child :last-of-type	$(“p:last-child”) $(“p:last-of-type”)	Tất cả các thẻ p cuối hoặc duy nhất Tả các thẻ p mà thẻ p đó là thẻ p cuối

Jquery Selector

Selector	Ví dụ	Mô tả
:nth-child(n)	$("p:nth-child(2)")	Thẻ p là con thứ 2 của cha nó
:nth-last-child(n)	$("p:nth-last-child(2)")	Thẻ p là con thứ 2 của cha nó tính từ cuối lên
:nth-of-type(n)	$("p:nth-of-type(2)")	Thẻ p là con thứ 2 (tính riêng thẻ p) của cha nó
:nth-last-of-type(n)	$("p:nth-last-of-type(2)")	Thẻ p là con thứ 2 (tính riêng thẻ p) của cha nó
:only-child	$("p:only-child")	Thẻ p là con duy nhất
:only-of-type	$("p:only-of-type")	Thẻ p là con duy nhất (chỉ tính riêng thẻ p)
parent > child	$("div > p")	Thẻ p là con trực tiếp của thẻ div
Parent descendant	$("div p")	Thẻ p là con cháu của thẻ div

Jquery Selector

Selector	Ví dụ	Mô tả
Element + next	$("div + p")	Thẻ p ngay sau thẻ div
Element ~ siblings	$("div ~ p")	thẻ p là anh em của thẻ div
:eq(index)	$("ul li:eq(3)")	Phần tử li thứ 4 trong danh sách (bắt đầu từ 0)
:gt(no)	$("ul li:gt(3)")	Các phần tử li có vị trí lớn hơn 3
:lt(no)	$("ul li:lt(3)")	Các phần tử li có vị trí nhỏ hơn 3
:not(selector)	$("input:not(:empty)")	Các phần tử input không rỗng
:header	$(":header")	Các phần tử tiêu đề h1, h2, h3
:animated	$(":animated")	Các phần tử động
:focus	$(":focus")	Phần tử đang bị chiếm quyền điều khiển

Jquery Selector

Selector	Ví dụ	Mô tả
:contains(text)	$(":contains('Hello')")	Phần tử chứa chữ Hello
:has(selector)	$("div:has(p)")	Các thẻ div mà chứa thẻ p
:empty	$(":empty")	Các thẻ rỗng
:parent	$(":parent")	Các thẻ không rỗng
:hidden	$("p:hidden")	Các thẻ p bị ẩn
:visible	$("table:visible")	Các thẻ table đang hiển thị
:root	$(":root")	Thẻ gốc của tài liệu
[attribute]	$("[href]")	Các thẻ có thuộc tính href
[attribute=value]	$("[href='default.htm']")	Các thẻ có thuộc tính href=default.htm

Jquery Selector

Selector	Ví dụ	Mô tả
[attribute!=value]	$("[href!='default.htm']")	Các thẻ có thuộc tính href có giá trị khác default.htm
[attribute$=value]	$("[href$='.jpg']")	Các thẻ có thuộc tính href kết thúc là .jpg
[attribute|=value]	$("[title|='Tomorrow']")	Các thẻ có thuộc tính title có giá trị bằng hoặc bắt đầu là tommorrow
[attribute^=value]	$("[title^='Tom']")	Các thẻ có thuộc tính title có bắt đầu là tom
[attribute~=value]	$("[title~='hello']")	Các thẻ có thuộc tính title là hello
[attribute*=value]	$("[title*='hello']")	Các thẻ có thuộc tính title chứa từ hello
:input	$(":input")	Tất cả các thẻ input

Jquery Selector

Selector	Ví dụ	Mô tả
:enabled	$(":enabled")	Các thẻ input được ở trạng thái enabled
:disabled	$(":disabled")	Các thẻ input ở trạng thái disabled
:selected	$(":selected")	Các thẻ input ở trạng thái selected
:checked	$(":checked")	Các thẻ input ở trạng thái checked
:text, :password, :radio, :checkbox, :submit, :reset, :button, :image, :file	$(":text")	Các thẻ input có type = text

Sự kiện

Sự kiện sử dụng chuột	Sự kiện sử dụng bàn phím	Form Events	Document/ Window Events
click	keypress	submit	load
dblclick	keydown	change	resize
mouseenter	keyup	focus	scroll
mouseleave		focusin	unload
mousedown		focusout	
mouseup		blur	
hover			
mouseover			
mouseout			

Cú pháp sử dụng sự kiện

```
$(selector).event_name(function(){

        //action goes here

});
```

Ví dụ:

```
$("p").click(function(){
        $(this).hide();
});
```

Sự kiện document ready

Sự kiện xảy ra khi trang web được load thành công (is ready)

Sử dụng

```
$(document).ready(function(){

        // jQuery methods go here...

});
```

```
$(function(){

        // jQuery methods go here...

});
```

jQuery event – bind()

Gắn một hoặc nhiều hàm xử lý sự kiện cho các phần tử

$(*selector*).**bind**(*event,data,function,map*)

- Event: tên sự kiện
- Data: dữ liệu xử lý
- Function: hàm xử lý

```
$("p").bind("click",function(){
        alert("The paragraph was clicked.");
});
```

Jquery Hiệu ứng

jQuery Hiệu ứng Hide/Show

- **hide**(): ẩn
  - $(*selector*).**hide**(*speed,callback*);
- **show**(): hiện
  - $(*selector*).**show**(*speed,callback*);
- **toggle**(): chuyển đổi trạng thái ẩn/hiện
  - $(*selector*).**toggle**(*speed,callback*);

```
$("button").click(function(){
        $("p").hide(1000);
});
```

- Speed: tốc độ thực hiện (miliseconds / slow /fast)
- Callback: hàm được gọi khi thực hiện xong ẩn/hiện

jQuery Hiệu ứng Fading

Làm mờ dần các hiệu ứng ẩn hiện

fadeIn(): hiện dần 1 phần tử

- $(*selector*).**fadeIn**(*speed,callback*);

fadeOut(): ẩn dần 1 phần tử

- $(*selector*).**fadeOut**(*speed,callback*);

fadeToggle(): ẩn/hiện dần 1 phần tử

- $(*selector*).**fadeToggle**(*speed,callback*);

fadeTo(): làm mờ phần tử tới 1 mức độ opacity (0-1)

- $(*selector*).**fadeTo**(*speed,opacity,callback*);

jQuery Hiệu ứng - sliding

slideDown(): trượt xuống

- $(*selector*). **slideDown**(*speed,callback*);

slideUp(): trượt lên

- $(*selector*). **slideUp**(*speed,callback*);

slideToggle(): trượt lên/xuống

- $(*selector*). **slideToggle**(*speed,callback*);

```
$("#flip").click(function(){
        $("#panel").slideToggle("slow");
});
```

jQuery Hiệu ứng - animation

Hiệu ứng ảnh động

animate():

- $(*selector*).**animate**({*params*},*speed*,*callback*);
- Params: định nghĩa css cho hiệu ứng

```
$("button").click(function(){
        $("div").animate({left:'250px‘,
                          height:'+=150px',});
});
```

- Các tên thuộc tính css phải được viết dưới dạng camelcased

jQuery dừng hiệu ứng

stop(): dừng hiệu ứng sliding, fading, animation trước khi hiệu ứng kết thúc.

- $(*selector*).**stop**(*stopAll,goToEnd*);

finish(): dừng hiệu ứng sliding, fading, animation trước khi hiệu ứng kết thúc.

- $(*selector*). **finish**(*stopAll,goToEnd*);
- Ví dụ:

```
$("#complete").click(function(){
            $("div").finish();
});
```

jQuery callback function

Hàm callback sẽ được thực thi ngay sau khi hiệu ứng kết thúc

- **$(*selector*).hide(*speed,callback*);**
- $("button").click(function(){
  $("p").hide("slow",function(){
  alert("The paragraph is now hidden");
  });
  });

jQuery Chaining

Có thể thực hiện một chuỗi các hiệu ứng

```
$("#p1").css("color","red")
        .slideUp(2000)
        .slideDown(2000);
```

jQuery HTML

Truy xuất nội dung phần tử HTML

text(): truy xuất nội dung text của phần tử được chọn

html(): truy xuất nội dung của phần tử được chọn bao gồm cả thẻ HTML

val(): truy xuất giá trị của phần tử trong form

```
<script>
 $(document).ready(function(){
    alert("Text: " + $("#test").text());
   alert("HTML: " + $("#test").html());
});
</script>
```

```
<p id="test">
   This is some
   <b>bold</b> text in a
   paragraph.
</p>
```

Truy xuất đến thuộc tính

attr(“attribute_name”): truy xuất đến thuộc tính của phần tử HTML được chọn

- $(*selector*).**attr**(*attribute*)
- $(*selector*).**attr**(*attribute,value*)
- $(*selector*).**attr**({*attribute*:*value*, *attribute*:*value*,...})

```
$("img").attr("width","500");
```

removeAttr(“attribute_name”): xóa thuộc tính

- $(*selector*).r**emoveAttr**(*attribute*)
-

```
$("p").removeAttr("style");
```

Thêm phần tử

append(): Thêm vào cuối nội dung của phần tử HTML

- $(*selector*).**append**(*content*,function(*index*,*html*))

```
$("p").append("Some appended text.");
```

appendTo("selector"): Thêm vào cuối nội dung của phần tử HTML

- $(*content*).**appendTo**(*selector*)

```
$("<span>Hello World!</span>").appendTo("p");
```

Thêm phần tử

prepend(): Thêm vào đầu nội dung phần tử HTML

- $(*selector*).prepend(*content*,function*(index,html)*)

```
$("p").prepend("Some prepended text.");
```

prependTo(): Thêm vào đầu nội dung phần tử HTML

- $(*content*).prependTo(*selector*)

```
$("<span>Hello World!</span>").prependTo("p");
```

Thêm phần tử

insertAfter(): Thêm phần tử vào sau phần tử được chọn

- $(*content*).**insertAfter**(*selector*)

```
$("<span>Hello world!</span>").insertAfter("p");
```

insertBefore(): Thêm phần tử vào trước phần tử được chọn

- $(*content*). **insertBefore**(*selector*)
-

```
$("<span>Hello world!</span>"). insertBefore("p");
```

Thêm phần tử

after(): Thêm vào phía sau phần tử HTML

- $(*selector*).**after**(*content*,function(*index*))

```
$("p").after("<p>Hello world!</p>");
```

before(): Thêm vào phía trước phần thử HTML

- $(*selector*).**before**(*content*,function*(index)*)

```
$("p").before("<p>Hello world!</p>");
```

- //index vị trí phần tử trong nhóm

Copy phần tử

clone(): copy phần tử bao gồm cả các phần tử con, text, và thuộc tính

- $(*selector*).**clone**(true|false)
  - True: copy cả các sự kiện
  - False: mặc định không copy các sự kiện

```
$("p").clone().appendTo("body");
```

Xóa phần tử

remove(): Xóa phần tử được chọn và các phần tử con

- $(*selector*).**remove**()

remove("conditions") : xóa phần tử thỏa mãn điều kiện conditions (- điều kiện lọc theo cú pháp selector)

- $(*selector*).**remove**("selector")

empty(): xóa các phần tử con của phần tử được chọn

- $(*selector*).**empty**()

```
$("#div1").remove();
$("p").remove(".italic");
$("#div1").empty();
```

Thêm/xóa class name

addClass(): Thêm một hoặc nhiều class

- $("h1,h2,p").**addClass**("blue");
- $("#div1").**addClass**("important blue");

removeClass(): Xóa một hoặc nhiều class

- $("h1,h2,p").**removeClass**("blue");

toggleClass(): Thêm/xóa class

- $("h1,h2,p").**toggleClass**(“hidden");

Truy xuất thuộc tính style

css(): truy xuất đến thuộc tính style của phần tử HTML

- **css**(“propertyname”) : trả về giá trị thuộc tính
- **css**("*propertyname*","*value*"): thiết lập giá trị thuộc tính
- **css**({"*propertyname*":"*value*","*propertyname*":"*value*",...}): Thiết lập nhiều giá trị thuộc tính

```
$("p").css("background-color");
$("p").css("background-color","yellow");
$("p").css({"background-color":"yellow","font-size":"200%"});
```

removeProp(): xóa một thuộc tính trong tập thuộc tính

- $(*selector*).removeProp(*property*)
- $(“p”).prop("color","FF0000");

Jquery Dimensions

width()

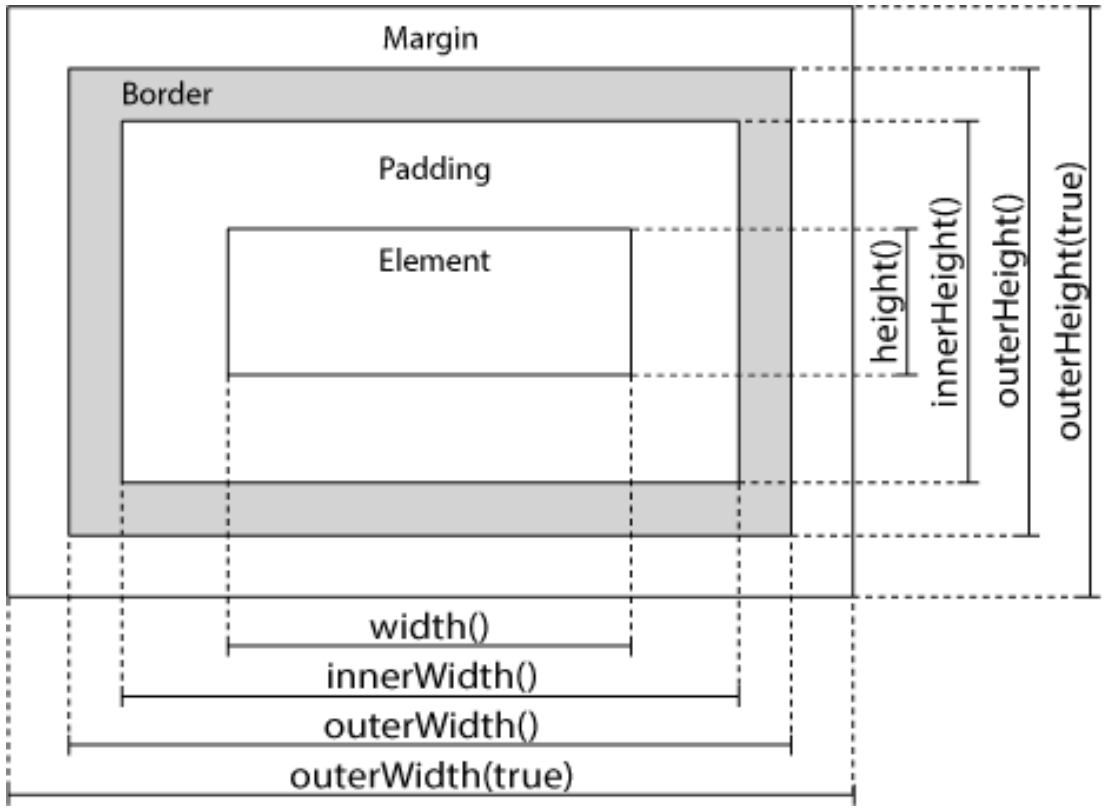
height()

innerWidth()

innerHeight()

outerWidth()

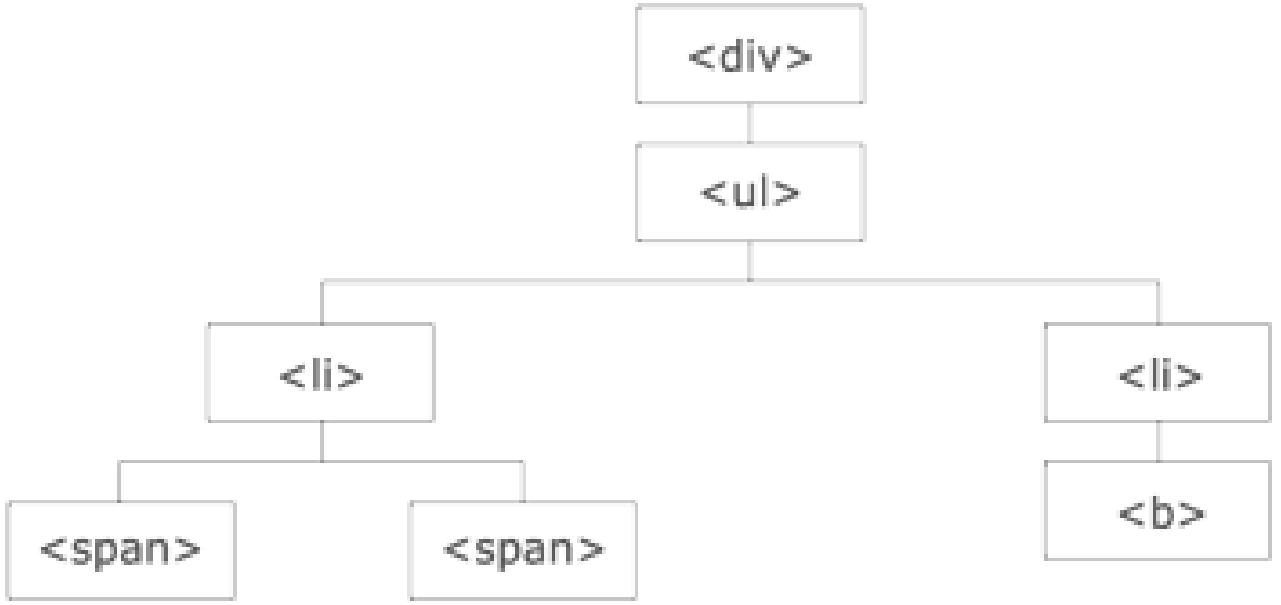
outerHeight()



jQuery Traversing

jQuery Traversing

Sử dụng để truy xuất đến các phần tử HTML dựa trên quan hệ của chúng với các phần tử khác.



jQuery Traversing – tổ tiên

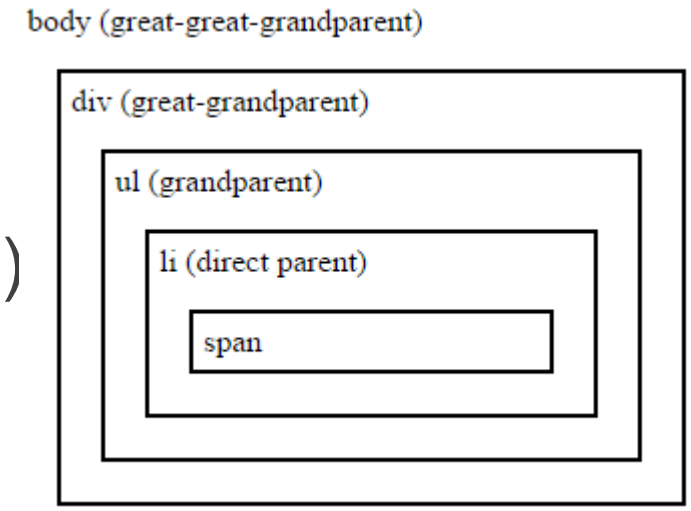
parent(): Trả về phần tử cha trực tiếp

- $("span").parent();

parents(): Trả về tất cả các phần tử cha (tổ tiên)

- $("span").parents();
- parents("option"): trả về tất cả các phần tử cha thỏa mãn option

parentsUntil("phần tử giới hạn"): Trả về tất cả các phần tử cha cho đến khi gặp phần tử giới hạn



jQuery Traversing – con cháu

children(): trả về tất cả con trực tiếp

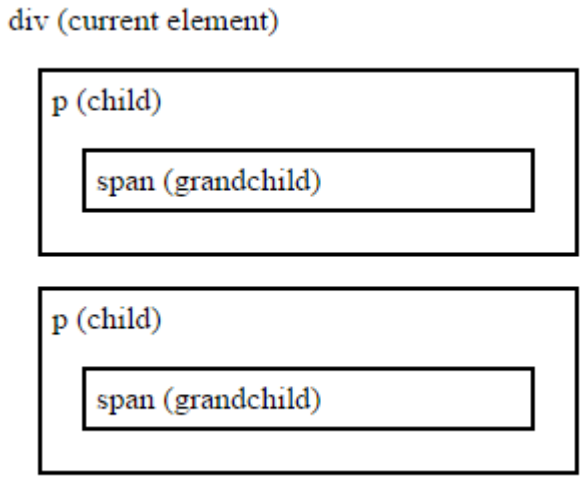
- $("div").children();

children("option"): trả về tất cả con trực tiếp thỏa mãn option

- $("div").children("p.hidden");

find("option"): trả về tất cả con cháu thỏa mãn option

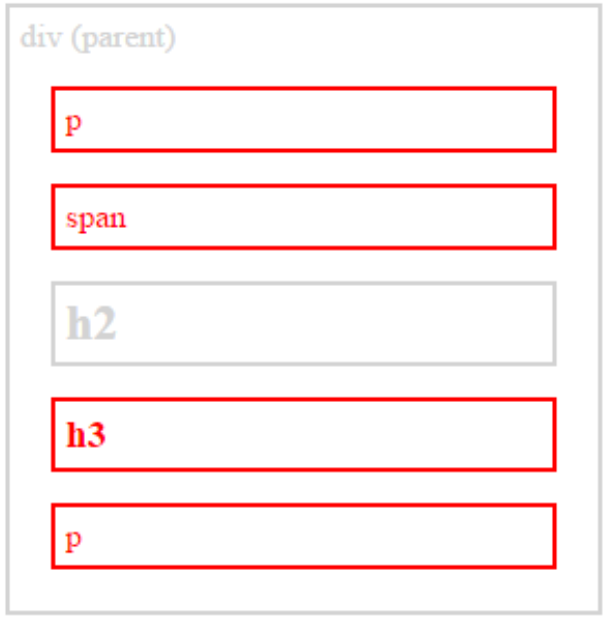
- $("div").find("span");
- $("div").find("*");



jQuery Traversing – anh em

siblings(): Trả về tất cả các phần tử anh em

- $("h2").siblings();



siblings("option"): Trả về tất cả các phần tử anh em thỏa mãn option

- $("h2").siblings("p");

jQuery Traversing – anh em

next(): trả về phần tử anh em ngay phía sau

- $("h2").next();

nextAll(): trả về tất cả các anh em phía sau

- $("h2").nextAll();

nextUntil("option"): trả về tất cả các anh em phía sau cho đến khi gặp phần tử giới hạn.

- $("h2").nextUntil("h6");

jQuery Traversing – anh em

prev(): Trả về phần tử anh em ngay phía trước

- $("h2").prev();

prevAll(): Trả về tất cả phần tử anh em phía trước

- $("h2").prevAll();

prevUntil("option"): Trả về tất cả phần tử anh em phía trước cho tới khi gặp phần tử giới hạn

- $("h6").nextUntil("h2");

jQuery Traversing – lọc

Lọc/chọn phần tử dựa trên vị trí của chúng trong một nhóm các phần tử

first(): Trả về phần tử đầu tiên

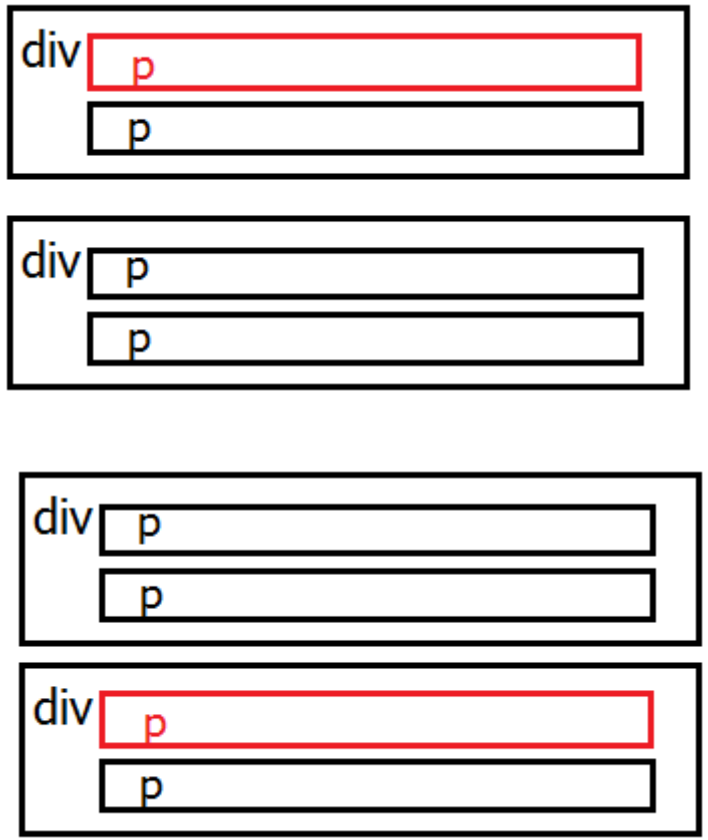
- $("div p").first();

last(): Trả về phần tử cuối cùng

- $("div p").last();

eq("index"): trả về phần tử với vị trí nhất định (bắt đầu từ 0)

- $("p").eq(2);



jQuery Traversing – lọc

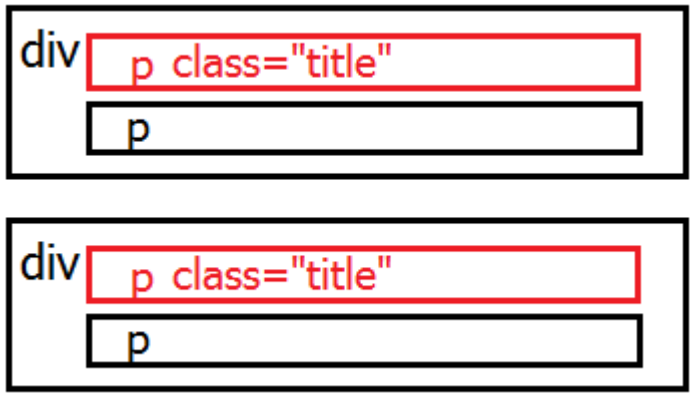
Lọc/chọn các phần tử phù hợp với các tiêu chí nhất định

filter(“option”): Lọc các phần tử phù hợp option

- $("p").filter(".title");

not (“option”): Lọc các phần tử không phù hợp option

- $("p").not(".title");



jQuery Ajax

jQuery - Ajax

Ajax = Asynchronous Javascript And Xml

Cho phép load dữ liệu và hiển thị trên trang web mà không phải tải lại trang.

- Ví dụ: Gmail, Youtube, Facebook tabs

jQuery ajax cung cấp các phương thức sử dụng trên tất cả các trình duyệt khác nhau một cách dễ dàng

jQuery - Load

load(): tải dữ liệu từ server và hiển thị trong phần tử được chọn

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

- URL: địa chỉ URL tải dữ liệu
- Data: tập hợp các chuỗi truy vấn cặp tên/giá trị gửi cùng yêu cầu
- Callback: hàm được thực hiện sau khi tải dữ liệu hoàn thành, chứa 3 tham số mặc định function(responseTxt,statusTxt,xhr)
  - responseTxt – nội dung kết quả nếu gọi lệnh thành công
  - statusTxt – trạng thái của việc gọi lệnh
  - xhr – chứa đối tượng XMLHttpRequest

jQuery - ajax

ajax(): thực hiện một yêu cầu ajax

$.ajax(*{name:value, name:value, ... }*)

Name	Value/Description
beforeSend(xhr)	Hàm thực thi trước khi gửi yêu cầu
cache	True/false: cho phép trình duyệt cache
complete(xhr,status)	Hàm thực thi sau khi yêu cầu hoàn thành (sau hàm success và error)
contentType	Loại nội dung được sử dụng khi gửi dữ liệu
data	Dữ liệu gửi lên server
dataType	Loại dữ liệu trả về
error(xhr,status,error)	Hàm thực thi khi yêu cầu xử lý bị lỗi
success(result,status,xhr)	Hàm thực thi khi yêu cầu xử lý thành công
type	Phương thức gửi dữ liệu(GET or POST)
url	Địa chỉ xử lý yêu cầu

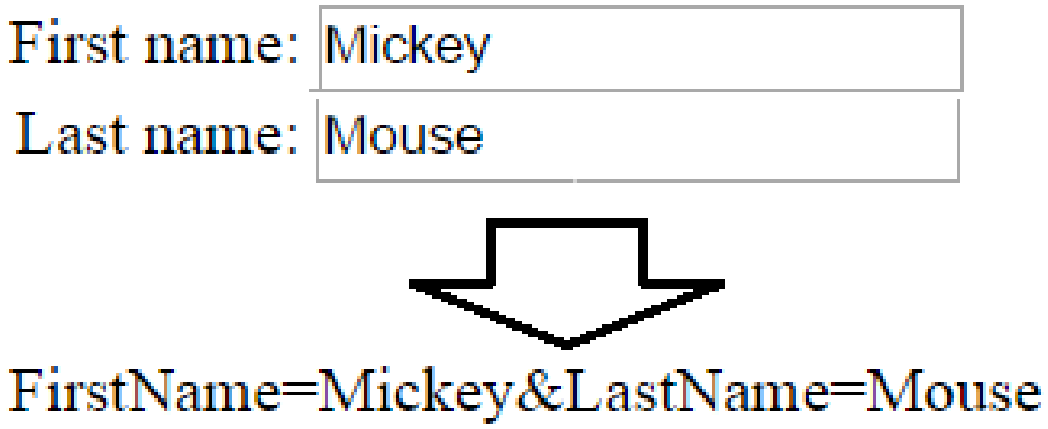
jQuery - ajax

```
$.ajax({

    url:"demo_test.txt",

    success:function(result){
            $("#div1").html(result);
     }

 });
```

jQuery – serialize

serialize(): hàm tạo chuỗi mã URL từ các giá trị dữ liệu trong form.

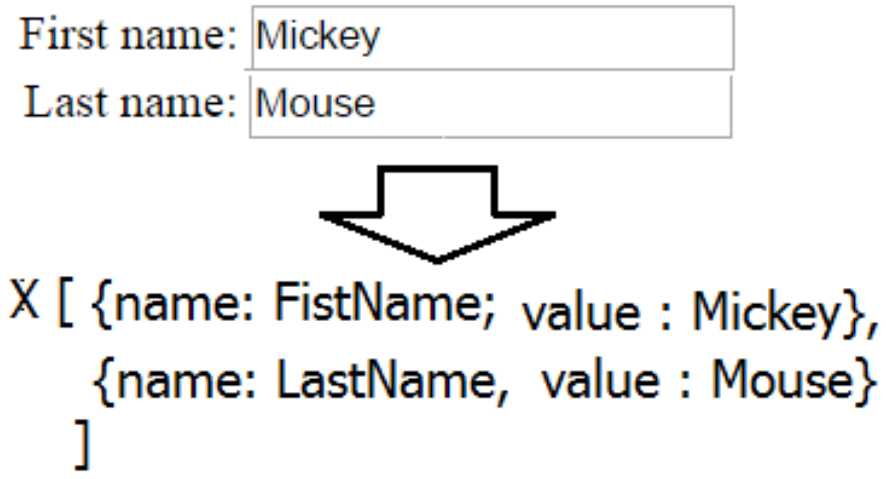
- $(*selector*).serialize()
- $("div").text($("form").serialize());



jQuery – serializeArray

serializeArray(): hàm tạo một mảng các đối tượng (name, value) từ các đối tượng trong form

- $(*selector*).serializeArray()
- x=$("form").serializeArray();



jQuery – Get vs POST

Sử dụng gửi yêu cầu đến server thông qua yêu cầu post/get

get(): Gửi yêu cầu theo phương thức get

- $.**get**(*URL,callback*);

```
$("button").click(function(){
        $.get("demo_test.asp",function(data,status){
          alert("Data: " + data + "\nStatus: " + status);
         });
});
```

jQuery – Get vs POST

post(): Gửi yêu cầu theo phương thức post

- $.post(*URL,data,callback*);

```
$("button").click(function(){
        $.post("demo_test_post.asp", {
                name:"Donald Duck",
                city:"Duckburg"
          },
        function(data,status){
         alert("Data: " + data + "\nStatus: " + status);
          });
});
```