Nền tảng phát triển Web

Bài 7: Javascript - jQuery

NỘI DUNG

1) Giới thiệu:
    1) Khái niệm
    2) Cài đặt
    3) Chương trình đầu tiên
    4) DOM
    5) Quy trình biên dịch Javascript

Nội dung

6) Các module trong jQuery
   - jQuery Selector
   - jQuery Attribute
   - jQuery DOM
   - jQuery Event
   - jQuery Effect
   - jQuery Ajax
7) JQuery với Bootstrap

1. GIỚI THIỆU

- Là thư viện được xây dựng từ Javascript
- jQuery - "viết ít, làm nhiều"
- Là thư viện được tích hợp nhiều nhất trong các website hiện nay
- Tạo các trang web có khả năng tương tác cao và tương thích trên nhiều trình duyệt.
- Thư viện Bootstrap cũng đã được tích hợp sẵn jQuery

1. GIỚI THIỆU

- Truy xuất các thành phần nội dung trang web với cú pháp tương tự ***css*** (thông qua các bộ chọn selector).
- Hỗ trợ nhiều thao tác xử lý trên tập các element chỉ bằng một dòng lệnh (***statement chaining***).

❑ $(“selector”).func1().func2().func3()…;

- Đơn giản hóa cách viết mã nguồn javascript (***write less, do more***). Tách biệt mã xử lý javascript và thành phần thể hiện HTML.

CÀI ĐẶT

- Tải về tệp jquery.min.js từ https://jquery.com/ và sử dụng offline trong Project.
- Sử dụng link bên ngoài như:
  - Google CDN
    - <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
  - Microsoft CDN
  - CDNJS CDN
  - jsDelivr CDN

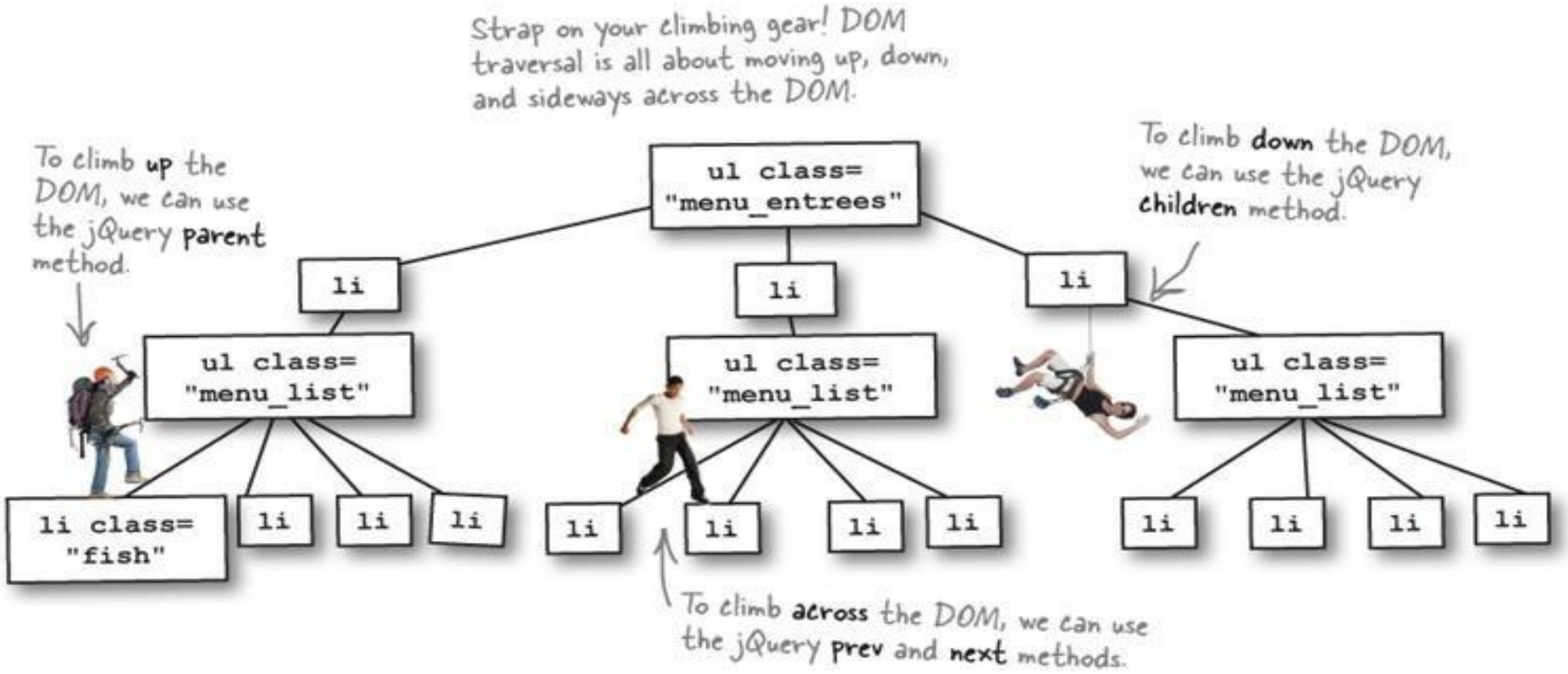
CHƯƠNG TRÌNH ĐẦU TIÊN

- jQuery luôn bắt đầu bằng từ khóa **$** hoặc **jQuery**
- Cơ chế hàm callback – hàm đóng vai trò như 1 tham số - thường xuyên được sử dụng trong lập trình

```
<!DOCTYPE html>
<html lang="en">
        <head>
                <meta charset="UTF-8">
                <script src="js/jquery-3.4.1.js"></script>
                <title>Title</title>
        </head>
        <body>
                <script type="text/javascript">
                        $(document).ready(function () {
                           document.write("Hello, World");
                            })
                </script>
        </body>
</html>
```

DOM

- Document Object Model
- Các thẻ HTML được quản lý bởi DOM



QUY TRÌNH BIÊN DỊCH

- File HTML được biên dịch từ trên xuống, từ trái qua phải
- Nguyên tắc khi code JS: chờ DOM load xong rồi mới thực thi các mã JS
- Sự kiện document.ready trong jQuery tương đương với window.onload trong Javascript
- Luôn đặt code JS trong document.ready (jQuery) hoặc window.onload (Javascript)

Sử dụng jquery (sự kiện onload)

- Xử lý sự kiện *onload* khởi tạo các thành phần trong trang.
- Cách tiếp cận Javascript truyền thống:

```
function onloadHandler()
    {
        alert("run after all page contents have been downloaded,
including image");
    }
window.onload = onloadHandler;
```

- Với JQuery, hàm xử lý sự kiện onload sẽ gọi ngay sau khi DOM của document đã nạp xong

Sử dụng jquery (sự kiện onload)

```
$(“document”).ready( function()
    {
        alert(“hello world”);
    }
                                );
```

- $(“document”).*ready* có thể được gọi nhiều lần, các hàm XL sự kiện sẽ được gọi theo thứ tự nó được đăng ký.

Sử dụng jquery (sự kiện onload)

$("document").*ready*(*function*(){

alert("hello world");

});

- $("document").*ready* có thể được gọi nhiều lần, các hàm XL sự kiện sẽ được gọi theo thứ tự nó được đăng ký.

```
<html xmlns="http://www.w3.org/1999/xhtml" >
<head>
    <title>Untitled Page</title>
    <!-- khai báo sử dụng thư viện jquery -->
    <script type="text/javascript" src="jquery-1.3.2.js" ></script>
    <!-- xử lý sự kiện onload -->
    <script type="text/javascript" >
        $("document").ready( function ()
            {
                alert("hello world");
            });
    </script>
</head>
```

CÁC THÀNH PHẦN TRONG JQUERY

- Core functionality: các phương thức core của JQuery và các hàm tiện ích được sử dụng thường xuyên.
- Selector & Traveral: chọn, tìm kiếm element, duyệt qua các element trong document.
- Manipulation & CSS: thay đổi nội dung các element trong document, làm việc với css.
- Event: đơn giản hóa việc xử lý event. Cung cấp event helper function đăng ký nhanh các event.
- Effect & Animation: cung cấp các hàm hỗ trợ tạo animation & effect.
- Ajax
- User interface: tập widget với các control: accordion, datepicker, dialog, progressbar, slider, tab
- Extensibility: hỗ trợ tạo plugin bổ sung thêm các chức năng mới vào core library.

JQUERY SELECTOR

JQUERY SELECTOR

- Truy xuất nội dung (element) trong document dựa trên biểu thức selector cung cấp. Selector sử dụng cú pháp tương tự CSS.

- Tập kết quả do Selector và Filter trả về: JQuery objects (không phải DOM objects).

Module jQuery – Bộ chọn

- Cú pháp và cách chọn tương tự CSS

SELECTOR	Ý NGHĨA
TAGNAME	Chọn tất cả các element có tên là TAGNAME
#IDENTIFIER	Chọn tất cả các element có ID là IDENTIFIER
.className	Chọn tất cả các element với thuộc tính class có giá trị là className
Tag.className	Chọn tất cả các element thuộc loại Tag, với thuộc tính class có giá trị là className
*	Chọn tất cả các element trên document.

Module jQuery – Bộ chọn

- Xác định phần tử HTML sẽ thao tác, cơ chế giống selector CSS
  - Selector by tag name
    - $(p)
  - Selector by ID
    - $('#some-id')
  - Selector by class name
    - $('.some-class')
  - Selector multiple
    - $('.some-class, #some-id')

Module jQuery – Bộ chọn

- Ví dụ

```
<ul id ="list1">
    <li class="a">item 1</li>
    <li class="a">item 2</li>
    <li class="b">item 3</li>
    <li class="b">item 3</li>
</ul>

<p class="a">this is paragraph 1</p>
<p id="para2">this is paragraph 2</p>
<p class="b">this is paragraph 3</p>
<p>this is paragraph 4</p>
```

```
<script type="text/javascript">
    $("document").ready(function () {
        $("p").css("border","1px solid red");

    });

</script>
```

- item 1
- item 2
- item 3
- item 3

this is paragraph 1

this is paragraph 2

this is paragraph 3

this is paragraph 4

Module jQuery – Bộ chọn

Ví dụ:

```
<ul id ="list1">
    <li class="a">item 1</li>
    <li class="a">item 2</li>
    <li class="b">item 3</li>
    <li class="b">item 3</li>
</ul>

<p class="a">this is paragraph 1</p>
<p id="para2">this is paragraph 2</p>
<p class="b">this is paragraph 3</p>
<p>this is paragraph 4</p>
```

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

    $("#para2").css("border","1px solid red");

});
```

- item 1
- item 2
- item 3
- item 3

this is paragraph 1

this is paragraph 2

this is paragraph 3

this is paragraph 4

Module jQuery – Bộ chọn

Ví dụ:

```
<ul id ="list1">
    <li class="a">item 1</li>
    <li class="a">item 2</li>
    <li class="b">item 3</li>
    <li class="b">item 3</li>
</ul>

<p class="a">this is paragraph 1</p>
<p id="para2">this is paragraph 2</p>
<p class="b">this is paragraph 3</p>
<p>this is paragraph 4</p>
```

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

    $("li.a").css("border","1px solid red");

});
```

- item 1
- item 2
- item 3
- item 3

this is paragraph 1

this is paragraph 2

this is paragraph 3

this is paragraph 4

Module jQuery – Bộ chọn

- Chọn element dựa trên mối quan hệ phân cấp giữa các element

SELECTOR	Ý NGHĨA
Selector1, .., selectorN	Chọn tất cả các element được xác định bởi tất cả các Selector
.class1,.class2	Chọn tất cả các element có khai báo class1 hoặc class2
Parent > Child	Chọn tất cả các Child element là con trực tiếp của Parent
Ancestor Descendant	Chọn tất cả các Descendant element là con cháu của Ancestor (chứa bên trong Ancestor)
Prev + Next	Chọn tất cả các Next element nằm kế tiếp Prev element
Prev ~ Siblings	Chọn tất cả các element anh em khai báo sau Prev và thỏa Sibling selector

Module jQuery – Bộ chọn

Ví dụ:

```
<ul id ="list1">
    <li class="a">item 1</li>
    <li class="a">item 2</li>
    <li class="b">item 3</li>
    <li class="b">item 3</li>
</ul>

<p class="a">this is paragraph 1</p>
<p id="para2">this is paragraph 2</p>
<p class="b">this is paragraph 3</p>
<p>this is paragraph 4</p>

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

    $("li.a,p.a,p#para2").css("border","1px solid red");

});
```

- item 1
- item 2
- item 3
- item 3

this is paragraph 1

this is paragraph 2

this is paragraph 3

this is paragraph 4

Module jQuery – Bộ chọn

Ví dụ:

```
<ul id ="list1">
    <li class="a">item 1</li>
    <li class="a">item 2</li>
    <li class="b">item 3</li>
    <li class="b">item 3</li>
</ul>

<p class="a">this is paragraph 1</p>
<p id="para2">this is paragraph 2</p>
<p class="b">this is paragraph 3</p>
<p>this is paragraph 4</p>
```

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

    $(".a,.b").css("border","1px solid red");

});
```

- item 1
- item 2
- item 3
- item 3

this is paragraph 1

this is paragraph 2

this is paragraph 3

this is paragraph 4

Module jQuery – Bộ chọn

Ví dụ:

```
<ul id ="list1">
    <li class="a">item 1</li>
    <li class="a">item 2</li>
    <li class="b">item 3</li>
    <li class="b">item 3</li>
</ul>
<p>
    Hello
    <a href="#">world</a>
    <span>
        <a href="#">2009</a>
    </span>

</p>
```

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

    $("p > a").css("border","1px solid red");
    $("span > a").css("border","1px solid red");
});
```

- item 1
- item 2
- item 3
- item 3

Hello world 2009

Module jQuery – Bộ chọn

Ví dụ:

```
<ul id ="list1">
    <li class="a">item 1</li>
    <li class="a">item 2</li>
    <li class="b">item 3</li>
    <li class="b">item 3</li>
</ul>
<p>
    Hello
    <a href="#">world</a>
    <span>
        <a href="#">2009</a>
    </span>

</p>
```

```
$("document").ready(function () {
    $("ul li.b, p a").css("border","1px solid red");

});
```

- item 1
- item 2
- item 3
- item 3

Hello world 2009

Module jQuery – Bộ chọn

Ví dụ:

```
<ul id ="list1">
    <li class="a">item 1</li>
    <li class="a">item 2</li>
    <li class="b">item 3</li>
    <li class="b">item 3</li>
</ul>
<p>
    Hello
    <a href="#">world</a>
    <span id="abc">
        <a href="#">2009</a>
    </span>
    <span >
        <a href="#">2010</a>
    </span>
</p>
<div>abc</div>
```

```
$("document").ready(function () {
    $("ul + p").css("border","1px solid red");
    $("ul + div").css("border","1px solid red");
});
```

- item 1
- item 2
- item 3
- item 3

Hello world 2009 2010

abc

Module jQuery – Bộ chọn

Ví dụ:

```
<ul id ="list1">
    <li class="a">item 1</li>
    <li class="a">item 2</li>
    <li class="b">item 3</li>
    <li class="b">item 3</li>
</ul>
<p>
    Hello
    <a id="link1" href="#">world</a>
    <span id="abc">
        <a href="#">2009</a>
    </span>
    <span >
        <a href="#">2010</a>
    </span>
</p>
<div>abc</div>
```

```
$("document").ready(function () {
    $("#link1 ~ span").css("border","1px solid red");

});
```

- item 1
- item 2
- item 3
- item 3

Hello world 2009 2010

abc

Form Selector

BỘ LỌC	Ý NGHĨA
:input	Chọn tất cả thẻ input, textarea trên Form
:text	Chọn tất cả text field trên Form
:password	Chọn tất cả password field
:radio	Chọn tất cả radio button
:checkbox	Chọn tất cả checkbox
:submit	Chọn tất cả button submit
:reset	Chọn tất cả button reset
:image	Chọn tất cả image
:button	Chọn tất cả generalized button
:file	Chọn tất cả control upload file

Form Selector

```
$("form :input").css("border","1px solid red");
```

First Name

Last Name

Disabled Text Field

Gender M F

What products are you interested in?
- Widgets
- Hibbity Jibbities
- SplashBangers
- Whatzits

Comments:

Optional life story file Browse...

submit reset

Form Selector

```
$("form :text").css("border","1px solid red");
```

First Name

Last Name

Disabled Text Field

Gender M F

What products are you interested in?
- Widgets
- Hibbity Jibbities
- SplashBangers
- Whatzits

Comments:

Optional life story file Browse...

submit reset

JQuery Filter

- JQuery Selector thường trả về 1 tập đối tượng. ***JQuery Filter được dùng để lọc trên kết quả chọn của JQuery Selector***.
- Có 6 loại Filter:
  - Basic: lọc phần tử ở vị trí đầu tiên, cuối cùng, chẵn, lẻ, …
  - Content: lọc dựa trên nội dung
  - Visibility: lọc dựa trên trạng thái hiển thị của element
  - Attribute: lọc dựa trên thuộc tính của element
  - Child: lọc dựa trên mối quan hệ với element cha
  - Form: lọc trên các thành phần khai báo trên Form

Basic JQuery Filter

Bộ lọc	Ý nghĩa
:first	Chọn phần tử ***đầu tiên*** trong tập kết quả do Selector trả về
:last	Chọn phần tử ***cuối cùng*** trong tập kết quả do Selector trả về
:even	Chọn phần tử chẵn
:odd	Chọn phần tử lẻ
:eq (index)	Chọn phần tử tại vị trí index
:gt (index)	Chọn phần tử có vị trí > index
:lt (index)	Chọn phần tử có vị trí < index
:header	Chọn tất cả header element (H1, H2, .. H6)
:not (selector)	Chọn phần tử không thỏa selector

JQuery Filter

```
<ul id ="list1">
    <li class="a">item 1</li>
    <li class="a">item 2</li>
    <li class="b">item 3</li>
    <li class="b">item 3</li>
</ul>
<p class="a" >this is paragraph 1</p>
<p id="para2">this is paragraph 2</p>
<p class="b">this is paragraph 3</p>
<p>this is paragraph 4</p>

$("document").ready(function () {
    $("p:odd").css("border","1px solid red");
});
```

- item 1
- item 2
- item 3
- item 3

this is paragraph 1

this is paragraph 2

this is paragraph 3

this is paragraph 4

JQuery Filter

```
<ul id ="list1">
    <li class="a">item 1</li>
    <li class="a">item 2</li>
    <li class="b">item 3</li>
    <li class="b">item 3</li>
</ul>
<p class="a" >this is paragraph 1</p>
<p id="para2">this is paragraph 2</p>
<p class="b">this is paragraph 3</p>
<p class="a">this is paragraph 4</p>
```

- item 1
- item 2
- item 3
- item 3

this is paragraph 1

this is paragraph 2

this is paragraph 3

this is paragraph 4

```
$("document").ready(function () {
    $(".a:odd").css("border","1px solid red");
});
```

JQuery Filter

```
<ul id ="list1">
    <li class="a">item 1</li>
    <li class="a">item 2</li>
    <li class="b">item 3</li>
    <li class="b">item 3</li>
</ul>
<p class="a" >this is paragraph 1</p>
<p id="para2">this is paragraph 2</p>
<p class="b">this is paragraph 3</p>
<p class="a">this is paragraph 4</p>
```

- item 1
- item 2
- item 3
- item 3

this is paragraph 1

this is paragraph 2

this is paragraph 3

this is paragraph 4

```
$("document").ready(function () {
    $("p:eq(1)").css("border","1px solid red");
    $("p:eq(3)").css("border","1px solid red");
});
```

JQuery Filter

```
<ul id ="list1">
    <li class="a">item 1</li>
    <li class="a">item 2</li>
    <li class="b">item 3</li>
    <li class="b">item 3</li>
</ul>
<p class="a" >this is paragraph 1</p>
<p id="para2">this is paragraph 2</p>
<p class="b">this is paragraph 3</p>
<p class="a">this is paragraph 4</p>

$("document").ready(function () {
   $("p:gt(1),p:lt(1)").css("border","1px solid red");

});
$("document").ready(function () {
   $("p:not(p:eq(1))").css("border","1px solid red");

});
```

- item 1
- item 2
- item 3
- item 3

this is paragraph 1

this is paragraph 2

this is paragraph 3

this is paragraph 4

Attribute Filter

BỘ LỌC	Ý NGHĨA
[attribute]	Lọc các phần tử có khai báo attribute
[attribute=value]	Lọc các phần tử có attribute với giá trị = value
[attribute!=value]	Lọc các phần tử có attribute với giá trị != value
[attribute^=value]	Lọc các phần tử có attribute với giá trị ***bắt đầu*** là value
[attribute$=value]	Lọc các phần tử có attribute với giá trị ***kết thúc*** là value
[attribute*=value]	Lọc các phần tử có attribute ***chứa*** giá trị value
[attributeFilter1] [attributeFilter2] …	Lọc các phần tử thỏa tất cả các attribute filter.

Attribute Filter

```
<ul id ="list1">
    <li class="a">item 1</li>
    <li class="a">item 2</li>
    <li class="b">item 3</li>
    <li class="b">item 3</li>
</ul>
<p class="a" >this is paragraph 1</p>
<p id="para2">this is paragraph 2</p>
<p class="b">this is paragraph 3</p>
<p class="a">this is paragraph 4</p>
```

- item 1
- item 2
- item 3
- item 3

this is paragraph 1

this is paragraph 2

this is paragraph 3

this is paragraph 4

```
$("document").ready(function () {
   $("p[class]").css("border","1px solid red");

});
```

Attribute Filter

```
<ul id ="list1">
    <li class="a">item 1</li>
    <li class="a">item 2</li>
    <li class="b">item 3</li>
    <li class="b">item 3</li>
</ul>
<p class="a" >this is paragraph 1</p>
<p id="para2">this is paragraph 2</p>
<p class="b">this is paragraph 3</p>
<p class="a">this is paragraph 4</p>
```

- item 1
- item 2
- item 3
- item 3

this is paragraph 1

this is paragraph 2

this is paragraph 3

this is paragraph 4

```
$("document").ready(function () {
    $("li[class=b]").css("border","1px solid red");

});
```

Attribute Filter

```
<ul id ="list1">
    <li class="a">item 1</li>
    <li class="a">item 2</li>
    <li class="b">item 3</li>
    <li class="b">item 3</li>
</ul>
<p class="a" >this is paragraph 1</p>
<p id="para2">this is paragraph 2</p>
<p class="b">this is paragraph 3</p>
<p class="a">this is paragraph 4</p>
```

- item 1
- item 2
- item 3
- item 3

this is paragraph 1

this is paragraph 2

this is paragraph 3

this is paragraph 4

```
$("document").ready(function () {
   $("p[id^=para]").css("border","1px solid red");

});
```

Attribute Filter

```
<ul id ="list1">
    <li class="a">item 1</li>
    <li class="a">item 2</li>
    <li class="b">item 3</li>
    <li class="b">item 3</li>
</ul>
<p class="a" >this is paragraph 1</p>
<p id="para2">this is paragraph 2</p>
<p class="b">this is paragraph 3</p>
<p class="a" lang="en-us" >this is paragraph 4</p>
```

- item 1
- item 2
- item 3
- item 3

this is paragraph 1

this is paragraph 2

this is paragraph 3

this is paragraph 4

```
$("document").ready(function () {
    $("p[class=a][lang*=us]").css("border","1px solid red");

});
```

Content & Visibility Filter

BỘ LỌC THEO NỘI DUNG	Ý NGHĨA
:contains(text)	Lọc các phần tử có chứa chuỗi text
:empty	Lọc các phần tử rỗng
:has(selector)	Lọc các phần tử có chứa ít nhất 1 element thỏa selector
:parent	Lọc các phần tử là cha (chứa ít nhất 1 element khác hoặc text)

BỘ LỌC THEO TRẠNG THÁI HIỂN THỊ	Ý NGHĨA
:visible	Lọc các phần tử có trạng thái là visible (đang hiển thị)
:hidden	Lọc các phần tử có trạng thái hidden (đang ẩn)

Content Filter

```
<ul id ="list1">
    <li class="a">item 1</li>
    <li class="a">item 2</li>
    <li class="b">item 3</li>
    <li class="b">item 3</li>
</ul>
<p class="a" >this is paragraph 1</p>
<p id="para2">this is paragraph 2</p>
<p class="b">this is paragraph 3</p>
<p class="a" lang="en-us" >this is paragraph 4</p>
```

- item 1
- item 2
- item 3
- item 3

this is paragraph 1

this is paragraph 2

this is paragraph 3

this is paragraph 4

```
$("document").ready(function () {
    $("p:contains(2)").css("border","1px solid red");

});
```

Content Filter

```
<ul id ="list1">
    <li class="a">item 1</li>
    <li class="a">item 2</li>
    <li class="b">item 3</li>
    <li class="b">item 3</li>
</ul>
<p class="a" >this is paragraph 1</p>
<p id="para2">this is paragraph 2</p>
<p class="b">this is paragraph 3</p>
<p class="a" lang="en-us" >this is paragraph 4</p>
<p></p>
<div></div>
```

- item 1
- item 2
- item 3
- item 3

this is paragraph 1

this is paragraph 2

this is paragraph 3

this is paragraph 4

```
$("document").ready(function () {
    $("p:empty").css("border","1px solid red");
    $("div:empty").css("border","1px solid red");
});
```

Content Filter

```
<p>
    Hello
    <a id="link1" href="#">world</a>
    <span id="abc">
        <a href="#">2009</a>
    </span>
    <span >
        <a href="#">2010</a>
    </span>
</p>
```

- item 1
- item 2
- item 3
- item 3

Hello world 2009 2010

abc

```
$("document").ready(function () {
    $("span:has(a:contains(2010))").css("border","1px solid red");

});
```

Content Filter

```
<p>
    Hello
    <a id="link1" href="#">world</a>
    <span id="abc">
        <a href="#">2009</a>
    </span>
    <span >
        <a href="#">2010</a>
    </span>
    <span>
    </span>
</p>
```

Hello world 2009 2010

```
$("document").ready(function () {
    $("span:parent").css("border","1px solid red");

});
```

Child Filter

BỘ LỌC	Ý NGHĨA
:nth-child(index) :nth-child(even) :nth-child(odd)	Lọc các phần tử theo vị trí so với cha của nó
:nth-child(equation)	Lọc phần tử theo vị trí (vị trí thỏa phương trình tham số) so với cha của nó
:first-child	Lấy phần tử đầu tiên so với cha của nó
:last-child	Lấy phần tử cuối cùng so với cha của nó
:only-child	Lấy phần tử nếu phần tử này là con duy nhất so với cha của nó

Child Filter

```
<ul id ="list1">
    <li class="a">item 1</li>
    <li class="a">item 2</li>
    <li class="b">item 3</li>
    <li class="b">item 3</li>
</ul>
```

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

    $("li:nth-child(2)").css("border","1px solid red");

});
```

```
$("document").ready(function () {
    $("li:nth-child(2n+1)").css("border","1px solid red");

});
```

- item 1
- item 2
- item 3
- item 3

- item 1
- item 2
- item 3
- item 3

Module jQuery – Thuộc tính

- Thao tác với các thuộc tính của thẻ HTML như id, class, title, src, … Các phương thức thường sử dụng
  - selector.attr(name): lấy giá trị của thuộc tính name
  - selector.attr(name, value): set giá trị value cho thuộc tính name
  - selector.removeAttr(name): xóa thuộc tính name
  - selector.hasClass(className): trả về true nếu có class className, ngược lại là false
  - selector.removeClass(className): xóa class className
  - selector.html(): lấy nội dung dạng html của selector
  - selector.html(value): set nội dung value cho selector
  - selector.text(): lấy nội dung dạng text của selector
  - selector.text(value): set nội dung text cho selector
  - selector.val(): lấy giá trị của thuộc tính value trong input form
  - selector.val(value): set giá trị value cho thuộc tính value trong input form

JQUERY - DOM

Module jQuery - DOM

- Thao tác với các thành phần DOM
- Các phương thức thường sử dụng
  - selector.replaceWith(element): thay thế nội dung selector bằng nội dung element
  - selector.remove(): xóa selector
  - selector.after(element): thêm phần tử element ở vị trí phía sau và ngang hàng với selector
  - selector.before(element): thêm phần tử element ở vị trí trước và ngang hàng với selector
  - selector.append(element): thêm phần tử element là con của selector, có vị trí đầu tiên
  - selector.prepend(element): thêm phần tử element là con của selector, có vị trí cuối cùng

Duyệt danh sách các element trong document

BỘ LỌC	Ý NGHĨA
size(), length	Lấy số phần tử trong tập kết quả của Selector
get()	Lấy tập DOM elements trong tập kết quả của Selector
get(index)	Lấy DOM element ở vị trí index
find(expression)	Lấy các element con cháu thỏa expression
each()	Gọi thực thi phương thức với từng element trong tập kết quả của Selector

Duyệt danh sách các element trong document

```
<p class="a" >this is paragraph 1</p>
<p id="para2">this is paragraph 2</p>
<p class="b">this is paragraph 3</p>
<p class="a" lang="en-us" >this is paragraph 4</p>
```

```
alert( $("p").size() );                    // 4
for(var i=0 ; i < $("p").size() ; ++i )
{
    var name = $("p").get(i); // DOM element
    var innerText = $("p").get(i).innerText;
}
```

Duyệt danh sách các element trong document

```
<p class="a" >this is paragraph 1</p>
<p id="para2">this is paragraph 2</p>
<p class="b">this is paragraph 3</p>
<p class="a" lang="en-us" >this is paragraph 4</p>
```

```
$("ul").find("li.a").css("border","1px solid red");

var i = 1;
$("p").each(function () {
        $(this).html("Custom paragraph " + i);
        i++;
});
```

Tạo nội dung mới

- Phương thức **$("html content")**, kết quả trả về là 1 JQuery object.

Ví dụ:

```
var h1 = $("<h1>heading 1</h1>"); // tạo thẻ h1 với nội dung

var temp = "<h1>heading 1</h1>";
var newH1 = $(temp); // tạo thẻ h1 từ biến temp

$("p:eq(0)").html(newH1);
```

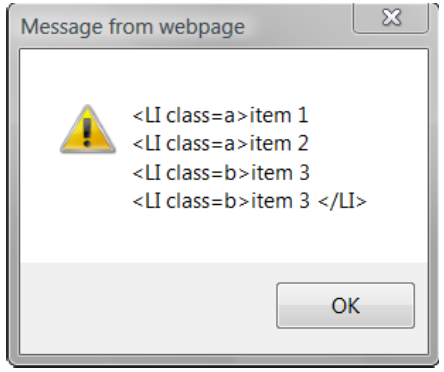
Truy cập, thay đổi nội dung trong element

Phương thức	Ý nghĩa
html()	Lấy nội dung html bên trong element đầu tiên thỏa selector
html(newContent)	Thay đổi nội dung html bên trong mọi element thỏa selector (tương tự innerHTML trong DOM)
text()	Lấy nội dung text bên trong element đầu tiên
text(newTextContent)	Thay đổi nội dung text bên trong mọi element thỏa selector (tương tự innerText)

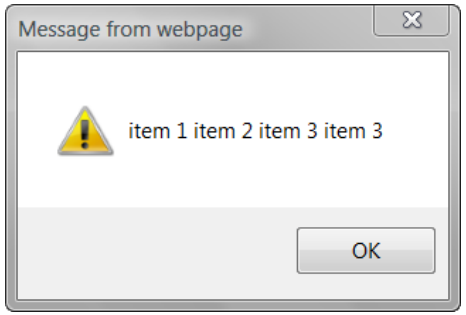
Truy cập, thay đổi nội dung trong element

```
<ul id ="list1">
    <li class="a">item 1</li>
    <li class="a">item 2</li>
    <li class="b">item 3</li>
    <li class="b">item 3</li>
</ul>
```

```
alert($("ul").html());
```




```
alert($("ul").text());
```



Truy cập, thay đổi nội dung trong element

```
var h1 = $("<h1>heading1</h1>");
$("#p1").html(h1);
var img1 = $("<img src='book2.jpg' />");
$("#p1").html(img1);

$("p:last").text("new content");
```

Thay đổi giá trị thuộc tính

Phương thức	Ý nghĩa
attr(name)	Lấy attribute value của element đầu tiên thỏa selector
attr(properties)	Thiết lập tập attribute cho mọi element thỏa selector. Properties có dạng object-notation syntax.
attr(key, value)	Thiết lập attribute cho mọi element thỏa selector
attr(key, function)	Thiết lập giá trị attribute dựa trên 1 function với mọi element thỏa selector.
removeAttr(name)	Xóa attribute với mọi element

Thay đổi giá trị thuộc tính – ví dụ

```
<a href="trang1.html">Trang 1</a>
    $("a").attr("href","trang2.html");
    $("a").text("trang 2");

<a href="book1.jpg">
   <img src="book1.jpg" />
</a>

    $("a").attr("target","_blank");
    $("a img").attr("src","book2.jpg");
    $("a").removeAttr("href");
    $("img").attr(   {src:"book2.jpg",alt:"hello world"}    );
```

Chèn nội dung

Phương thức	Ý nghĩa
append(content)	Chèn content vào sau nội dung có sẵn của các element thỏa selector
appendTo(selector)	Chèn element thỏa selector vào sau nội dung có sẵn của các element thỏa selector tham số
prepend(content)	Chèn content vào trước nội dung có sẵn của các element thỏa selector
prependTo(selector)	Chèn element thỏa selector vào trước nội dung có sẵn của các element thỏa selector tham số
after(content)	Chèn content vào sau các element thỏa selector
before (content)	Chèn content vào trước các element thỏa selector

Chèn nội dung

```
<ul id ="list1">
    <li class="a">item 1</li>
    <li class="a">item 2</li>
    <li class="b">item 3</li>
    <li class="b">item 3</li>
</ul>
<p class="a" >this is paragraph 1</p>
<p id="para2">this is paragraph 2</p>
<p class="b">this is paragraph 3</p>
<p class="a" lang="en-us" >this is paragraph 4</p>
```

```
$("li").append("<b>new content</b>");
```

- item 1 **new content**
- item 2 **new content**
- item 3 **new content**
- item 3 **new content**

Chèn nội dung

```
$("p#para2").appendTo("li.a");
```

- item 1

  this is paragraph 2

- item 2

  this is paragraph 2

- item 3
- item 3

this is paragraph 1

this is paragraph 3

this is paragraph 4

```
$("p.b").prependTo("li.b");
```

- item 2
- this is paragraph 3

  item 3
- this is paragraph 3

  item 3

this is paragraph 1

this is paragraph 2

this is paragraph 4

Làm việc với CSS

Phương thức	Ý nghĩa
css (name)	Lấy giá trị thuộc tính name của element đầu tiên thỏa selector
css (properties)	Thiết lập tập thuộc tính css đối với mọi element thỏa selector
css (property, value)	Thiết lập giá trị 1 thuộc tính đối với mọi element thỏa selector

Làm việc với CSS

```
<p class="a" >this is paragraph 1</p>
<p id="para2">this is paragraph 2</p>
<p class="b">this is paragraph 3</p>
<p class="a" lang="en-us" >this is paragraph 4</p>
```

```
$("p#para2").css({ "color" : "red" , "background-color" : "green" })
```

this is paragraph 2

```
$("p.a").css("color","blue");
```

this is paragraph 1

this is paragraph 2

this is paragraph 3

this is paragraph 4

Làm việc với CSS

Phương thức	Ý nghĩa
addClass (class)	Thêm class vào các element thỏa selector
hasClass (class)	Kiểm tra class có tồn tại trong các element thỏa selector
removeClass (class)	Xóa class khỏi các element thỏa selector
toggleClass (class)	Thêm class vào các element thỏa selector nếu class chưa khai báo, ngược lại nếu đã tồn tại rồi, class sẽ bị xóa

Thay đổi kích thước

Phương thức	Ý nghĩa
height ()	Lấy chiều cao của element đầu tiên thỏa selector
width ()	Lấy chiều rộng của element đầu tiên thỏa selector
height (val)	Thiết lập chiều cao của mọi element thỏa selector
width (val)	Thiết lập chiều rộng của mọi element thỏa selector

JQUERY - EVENTS

Module jQuery – Sự kiện (Event)

- Thao tác với các sự kiện
- jQuery viết lại cách gọi các sự kiện, nhưng về bản chất vẫn hoạt động sử dụng cơ chế của Javascript
- Có thể có 2 cách viết

```
$('#paragraph').click(function () {
  console.log("clicked");
  });
$('#paragraph').bind('click', function () {
  console.log("clicked");
  })
```

Module jQuery – Sự kiện (Event)

```
$("selector").bind(event,[data],[handler]);
```

Tham số	Ý nghĩa
event	Sự kiện selector xử lý, bao gồm: load, blur, click, dbclick, mousedown, mouseup, mousemove, mouseover, mouseout, submit, keydown, keypress, keyup, ..
data	Tùy chọn, dữ liệu truyền vào handler khi event xảy ra
handler	Tên hàm xử lý sự kiện

Module jQuery – Sự kiện (Event)

```
$("div").bind("mouseover",highLight);
$("div").bind("mouseleave",highLight);
$("div").bind("click", function () {
        $("div").unbind("mouseover",highLight);
        $("div").unbind("mouseleave",highLight);
        $("div").html("<p style='color:green;'>turn off</p>");
      })

function highLight(evt)
{
    $("div").toggleClass("highlight");
}
```

Module jQuery – Sự kiện (Event)

Xử lý nhanh một số sự kiện thường gặp

Phương thức	Ý nghĩa
click(func)	Xử lý sự kiện click của 1 selector. Một số hàm khác: blur, mousedown, mouseover, mouseout, submit, ..
hover (func1, func2)	Func1: hàm xử lý được gọi khi mouse di chuyển trên selector Func2: hàm xử lý được gọi khi mouse di chuyển ra khỏi selector

```
$("div").hover( highLight , highLight );
 function highLight(evt)
 {
        $("div").toggleClass("highlight");
 }
```

Đối tượng Event

- Cung cấp các thông tin về event để xử lý.

Thuộc tính / Phương thức	Ý nghĩa
type	Loại event xảy ra, ví dụ: “click”
target	Element mà event xảy ra
data	Dữ liệu được truyền vào handler bởi phương thức bind
pageX, pageY	Tọa độ chuột khi event xảy ra
preventDefault ( )	Ngăn trình duyệt không thực thi xử lý mặc định, ví dụ khi click vào liên kết

```
$("div").click(function (evt)
{
        $(this).html("pageX:" + evt.pageX + ", pageY:" + evt.pageY + ", type:"+ evt.type + ", target:" + evt.target);
});
```

JQUERY- HIỆU ỨNG

Module jQuery – Hiệu ứng (Effect)

- Cung cấp các hàm liên quan đến hiệu ứng cho phần tử
- Một số hàm thông dụng:
  - selector.show(speed): hiển thị selector với tốc độ speed tính bằng milliseconds
  - selector.hide(speed): ẩn selector với tốc độ speed tính bằng milliseconds
  - selector.toggle(speed): hiển thị/ẩn selector dựa vào trạng thái hiện tại, nếu đang hiển thị -> ẩn và ngược lại
  - selector.animate(properties, speed): tạo hiệu ứng chuyển động cho selector dựa vào các properties với tốc độ speed tính bằng milliseconds
  - selector.fadeIn(speed): hiển thị selector theo cơ chế fade với tốc độ speed
  - selector.fadeOut(speed): ẩn selector theo cơ chế fade với tốc độ speed

Ẩn/hiện element

Phương thức	Ý nghĩa
show ( )	Hiển thị các element thỏa selector nếu trước đó bị ẩn
show(speed, callback)	Hiển thị các element thỏa selector nếu trước đó bị ẩn, speed xác định tốc độ hiển thị. Sau khi hiển thị xong, phương thức callback sẽ được thực thi.
hide ( )	Ẩn element nếu trước đó đang hiển thị.
hide (speed, callback)	Ẩn element nếu trước đó đang hiển thị, tham số có ý nghĩa tương tự phương thức show.
toggle ( )	Chuyển qua lại trạng thái ẩn/hiện các element.
toggle (speed, callback)	Chuyển qua lại trạng thái ẩn/hiện các element, tham số có ý nghĩa tương tự phương thức show.

- ***speed***: tốc độ hiệu ứng quy định bởi các giá trị: “slow”, “normal”, “fast” hoặc millisecond

Ẩn/hiện element

- ```
  $("#div1").show("normal");
  $("#div1").hide("slow");

  $("#div1").hide(4000); // ẩn trong 4 giây

  // thay đổi luân phiên trạng thái ẩn/hiện
  $("#div1").toggle("fast");
  ```
  ```

Fade in/fade out

```
$("#button_fadein").bind("click",function () {
    $("#div1").fadeIn("normal");
});

$("#button_fadeout").bind("click",function () {
    $("#div1").fadeOut("slow");
});

$("#button_fadeto3").bind("click",function () {
    $("#div1").fadeTo("slow",0.3,function () { alert("finished");
    });
});

$("#button_fadeup").bind("click",function () {
    $("#div1").fadeTo("slow",1.0);
});
```

Sliding

```
$("#button_slideup").bind("click",function () {
        $("#div1").slideUp("normal");
    });

$("#button_slidedown").bind("click",function () {
        $("#div1").slideDown("slow");
    });

$("#button_toggleslide").bind("click",function ()
        {
        $("#div1").slideToggle(3000);
    });
```

Phương thức	Ý nghĩa
slideDown(speed, callback)	Hiển thị element bằng cách tăng chiều cao.
slideUp(speed, callback)	Ẩn element bằng cách giảm chiều cao.
slideToggle(speed, callback)	Chuyển đổi trạng thái ẩn/hiện element.

Custom Animation

$(“selector").animate(properties,[duration], [easing],[callback]);

Tham số	Ý nghĩa
properties	Các thuộc tính xác trạng thái hiển thị sau khi animate.
duration	Animate kéo dài trong bao lâu (“slow”, “normal”, “fast”, milisecond)
easing	Hiệu ứng xóa : swing, linear
Callback	Hàm được gọi sau khi animate xong

$(“selector").stop();

Custom Animation

```
$("#button_growright").click(function () {
   $("#div1").animate({width:"800"},"normal");
});

$("#button_growleft").click(function () {
   $("#div1").animate({width: "100"},"fast");
});

$("#button_bigtext").click(function () {
   $("#div1").animate({fontSize:"40"},2000);
});
$("#button_movediv").click(function () {
        $("#div1").animate( { left : "500", fontSize: "50"} , 1000,"linear" );
   })
```

Module jQuery - Ajax

- Ajax - Asynchronous JavaScript and XML, là kỹ thuật cho phép gửi nhận dữ liệu tới server mà không cần load lại trang
- Thường được sử dụng trong PHP, theo mô hình client – server
- Các phương thức thường gặp
  - selector.load(url, [data]): load dữ liệu từ url, có thể gửi dữ liệu data hoặc không
  - $.ajax(options): load data theo cơ chế ajax
  - $.get(options): load data theo cơ chế ajax sử dụng phương thức GET
  - $.post(options): sử dụng phương thức POST

jQuery - Bootstrap

CSS

Copy-paste the stylesheet `<link>` into your `<head>` before all other stylesheets to load our CSS.

```
<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css" integrity="sha384-
```

Copy

JS

Many of our components require the use of JavaScript to function. Specifically, they require jQuery, Popper.js, and our own JavaScript plugins. Place the following `<script>`s near the end of your pages, right before the closing `</body>` tag, to enable them. jQuery must come first, then Popper.js, and then our JavaScript plugins.

We use jQuery's slim build, but the full version is also supported.

```
<script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" integrity="sha384-KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6a
<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js" integrity="sha384-ApNbgh9B+Y1Q
<script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js" integrity="sha384-JZR6Spejh4U02d8jOt
```

Copy

Nền tảng phát triển Web

Bài 8: Javascript – jQuery nâng cao

Nội dung

1) Biểu thức chính qui
2) jQuery Validate plugin
3) jQuery UI

1. Biểu thức chính qui

Nội dung

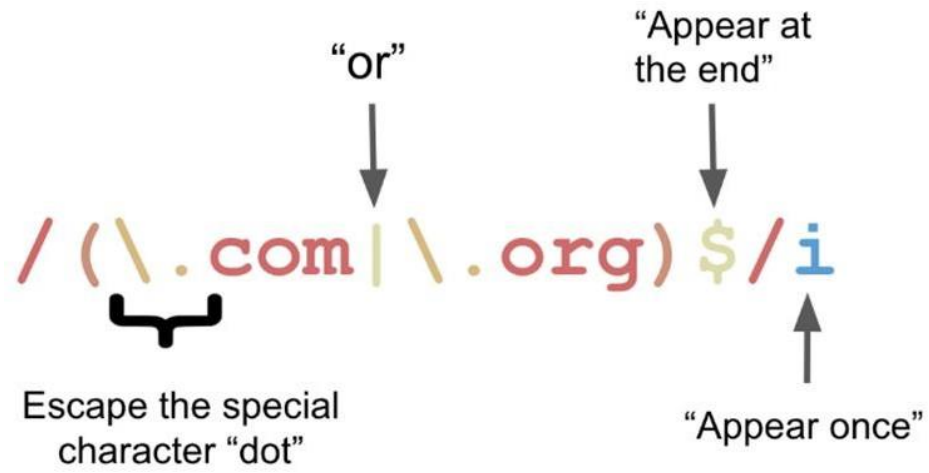
1) Khái niệm
2) Khởi tạo
3) Phương thức test
4) Một số điều kiện sẵn có
5) Các điều kiện hay sử dụng
6) Setting Regex
7) Thực hành

Khái niệm

- Hiểu đơn giản RegEx là 1 pattern (mẫu) cho phép bạn kiểm tra 1 chuỗi ký tự như email, mật khẩu .v.v có khớp với pattern mà bạn quy định từ trước hay không
- Regex rất khó học, khó viết, khó nhớ, khó sửa. tuy nhiên có những việc mà chỉ sử dụng regex thì mới

  có thể thực hiện được
- VD: kiểm tra xem 1 chuỗi bất kỳ có kết thúc bằng .com, .org hay không chẳng hạn
  - abc.com => true
  - abc.org => true
  - abc.comvn => false
  - => chuỗi regex này sẽ có dạng /(\.com|\.org)/

Ví dụ



- Ký tự \ đặt trước các ký tự đặc biệt để hiện thị chính xác ký tự đó
- Ký tự | là điều kiện OR các biểu thức Regex
- Ký tự $ là điều kiện phải xuất hiện ở cuối chuỗi, ngược lại sẽ có ký tự ^ là phải xuất hiện ở đầu chuỗi
- Phần nằm giữa ký tự // là nơi viết các biểu thức Regex của bạn
- Ký tự I ở cuối là setting cho phép không phân biệt hoa thường khi sử dụng Regex

Khởi tạo

- Trong Javascript, Regex là 1 object, được khởi tạo theo 2 cách sau
  - var regex = new RegExp(pattern);
  - var regex =/abc/; //thường sử dụng cách này

Phương thức Test

- Kiểm tra xem chuỗi ban đầu có khớp với biểu thức regex hay không, trả về true nếu khớp, ngược lại trả về false
  - var regex = /abc/;
  - console.log(regex.test("abc abc 123"));//true
  - console.log(regex.test("bca"));//false

Phương thức Exec

- Tìm kiếm trong chuỗi ban đầu mà khớp với regex. Nếu tìm thấy sẽ trả về mảng kết quả, ngược lại trả về null
  - var regex = /abc/;
  - console.log(regex.exec("abc abc 123"));

Một số điều kiện có sẵn

Ký tự	Giải thích
\d	Tương đương với [0-9], chứa ký tự số
\D	Tương đương với [^0-9], chứa ký tự không phải số
\w	Tương đương [A-Za-z0-9], chứa ký tự là chữ hoặc số
\W	Tương đương [^A-Za-z0-9], chứa ký tự không phải chữ và số
\s	chứa ký tự khoảng trống như space, tab(\t), newline (\n)
\S	Ngược lại với \s, không chứa ký tự khoảng trống

Các điều kiện hay sử dụng

- Phải xuất hiện ở đầu chuỗi: ^
  - var regex = /^nvanh/;
  - console.log(regex.test(“nvanh is")); //true
  - console.log(regex.test("is nvanh is")); //false

Các điều kiện hay sử dụng

- Phải xuất hiện ở cuối chuỗi: $
  - var regex = /nvanh$/;
  - console.log(regex.test("is nvanh")); //true
  - console.log(regex.test("nvanh is")); //false

Các điều kiện hay sử dụng

- So sánh tuyệt đối: kết hợp 2 ký tự ^ và $
  - var regex = /^nvanh$/;
  - console.log(regex.test("nvanh")); //true
  - console.log(regex.test("nvanh is")); //false

Các điều kiện hay sử dụng

- Bắt đầu bằng 1 pattern, kết thúc là 1 pattern: .*
  - Ví dụ: bắt đầu bằng Name, kết thúc là Anh
  - var regex = /^Name.*Anh$/;
  - console.log(regex.test("Name is Anh")); //true
  - console.log(regex.test("nvanh is")); //false

Các điều kiện hay sử dụng

- Nằm trong khoảng: [Điểm_bắt_đầu-Điểm_kết_thúc]
  - [a-z]: các ký tự từ a,b,c … , x,y,z
  - [b-e]: các ký tự b,c,d,e
  - [A-Z]: các ký tự A,B,C …, X,Y,Z
  - [0-9]: các ký tự 0,1,2 … 9
  - [a-zA-Z0-9]: các ký tự a-z, A-Z, 0-9
- Ví dụ
  - var regex = /[a-z]/;
  - console.log(regex.test("Name is Anh 123")); //true
  - console.log(regex.test("1212")); //false

Các điều kiện hay sử dụng

- Đảo ngược kết quả: [^]
  - var regex = /[^0-9]/;
  - console.log(regex.test("anh is nvanh")); //true
  - console.log(regex.test("12")); //false

Các điều kiện hay sử dụng

- Điều kiện hoặc: |
  - var regex = /anh|nvanh/;
  - console.log(regex.test("anh is nvanh")); //true
  - console.log(regex.test(“an")); //false

Các điều kiện hay sử dụng

- Xuất hiện đúng x lần: {x}
  - var regex = /^[0-9]{3}$/;
  - console.log(regex.test("12")); //false
  - console.log(regex.test("123")); //true
  - console.log(regex.test("1234")); //false

Các điều kiện hay sử dụng

- Xuất hiện từ x đến y lần: {x,y}
- var regex = /^[0-9]{3,5}$/;
  - console.log(regex.test("12")); //false
  - console.log(regex.test("123")); //true
  - console.log(regex.test("1234")); //true
- Nếu không set y, thì hiểu là >=x lần
  - var regex = /^[0-9]{3,}$/;
  - console.log(regex.test("12")); //false
  - console.log(regex.test("123")); //true
  - console.log(regex.test("123412121")); //true

Các điều kiện hay sử dụng

- Điều kiện không bắt buộc: ?
  - var regex = /^[0-9]{0,}?$/;
  - console.log(regex.test("")); //true
  - console.log(regex.test("123")); //true
  - console.log(regex.test("123412121")); //true

Các điều kiện hay sử dụng

- Nhóm điều kiện: ()
  - var regex = /^(\d{2})+$/;
  - console.log(regex.test("1")); //false
  - console.log(regex.test("3412")); //true
  - console.log(regex.test("232")); //false

Các điều kiện hay sử dụng

- Kiểm tra xem chuỗi có đáp ứng điều kiện ngay sau nó hay không: (?=)
- Ví dụ: kiểm tra xem chuỗi có chứa ký tự ngay sau từ is là abc hoặc def không
  - var pattern = /is(?=abc|def)/;
  - console.log(pattern.test('Hello isabc')); //true
  - console.log(pattern.test(''Hello isdef')); //true
  - console.log(pattern.test(''Hello is abc')); //false
- Ví dụ: Kiểm tra 1 chuỗi có độ dài từ 5 đến 10 ký tự, và phải chứa chuỗi abc hoặc def
  - var pattern = /^(?=.*abc|def)[a-z0-9]{5,9}$/;
  - console.log(pattern.test('1212isabc'));
  - console.log(pattern.test('11isabc'));

Thiết lập RegExp

- /abc/i

Ký tự	Giải thích
g	kiểm tra điều kiện nhiều lần
i	không phân biệt hoa thường
m	cho phép kiểm tra xuống dòng

2. jQuery validate plugin

Nội dung

- Khái niệm
- Khai báo
- Cách sử dụng

Khái niệm

- Là thư viện viết sẵn dựa trên jQuery
- Giúp validate dữ liệu theo kiểu realtime, nghĩa là thông báo lỗi ngay khi user nhập sai mà chưa cần submit

Cài đặt

- Có thể download về hoặc nhúng link online (CDN)
- Phải nhúng thư viện jQuery trước
- Nhúng sử dụng link CDN

```
<script src="//code.jquery.com/jquery-1.11.3.min.js"></script>
<script type="text/javascript"
src="http://ajax.aspnetcdn.com/ajax/jquery.validate/1.13.1/jquery.validate.min.js"></script>
```

Sử dụng

```
<script type="text/javascript">
    $(document).ready(function () {
        var object = {
            errorPlacement: function(error, element) {
                error.insertAfter(element);
                error.wrap("<span class='l-error-text'>");
            },
            rules: {
                username: {
                    required: true,
                },
                age: {
                    required: true
                },

            },
            messages: {
                username: {
                    required: "vui lòng nhập tên"
                },
                age: {
                    required: "vui lòng nhập tuổi"

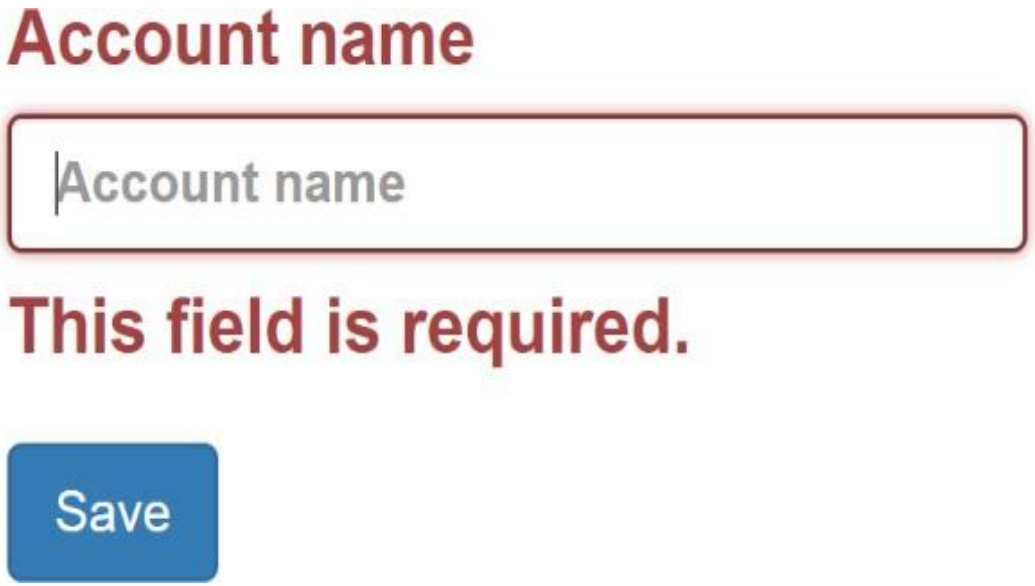
                }
        }
        $('#form').validate(object);
    });
</script>
```

Sử dụng – Các luật

required	Không được bỏ trống
remote	Gửi yêu cầu về Web Server để xác thực
minlength	Độ dài tối thiểu
min	Số tối thiểu
max	Số tối đa
range	Số tối thiểu từ x tới y
email	Xác thực định dạng Email
url	Xác thực định dạng URL
date	Xác thực định dạng ngày tháng
number	Phải là số, bao gồm số thập phân
equalTo	Phải trùng với phần tử nào đó

Thực hành

- Sử dụng jquery validate để validate form như hình sau, nếu trường Account name để trống sẽ báo lỗi như trong hình



2. jQuery UI

Khái niệm

- jQuery UI – jQuery User Interface
- Là 1 thư viện miễn phí được xây dựng dựa trên jQuery, cung cấp những ứng dụng, widget, theme nâng cao hơn.
- Ví dụ về mặt hiệu ứng, trong khi jQuery chỉ có phương thức $(selector).show(time, callback) thì jQuery UI bổ sung 1 đống các extend vào như sau: $(selector).show("EFFECT NAME", time, callback)

Sử dụng

- Truy cập trang chủ jQuery UI: https://jqueryui.com/
- Trải nghiệm các component, click view source để xem code