

1. Mục tiêu

- Giải thích được công dụng của Javascript.
- Định nghĩa các khái niệm cơ bản trong Javascript.
- Tạo và sử dụng hàm trong Javascript.
- Sử dụng Javascript để kiểm tra form.

2. Tóm tắt lý thuyết

2.1 Javascript là gì?

- Javascript là một ngôn ngữ lập trình được xây dựng sẵn trong các trình duyệt web, dùng để bổ sung tính tương tác cho các website như thay đổi màu sắc, kích thước, hình ảnh, kiểm tra giá trị nhập...
- Hạn chế của Javascript là không thể giao tiếp với máy chủ, làm việc khác nhau trên các trình duyệt khác nhau.

2.2 Sử dụng Javascript.

Các đoạn Javascript đặt trong cặp tag `<script> </script>`. Cặp tag này có thể đặt trong `<head></head>` hay trong `<body></body>`. Ngoài ra các đoạn Javascript có thể đặt trong 1 file có phần mở rộng là .js. Để sử dụng các đoạn script trong 1 file riêng (ví dụ như abc.js)

```
<script type="text/javascript" src="abc.js"> </script>
```

Ví dụ:

```
<head>  
  <title>Ví dụ Javascript</title>  
  <script type="text/javascript"> alert("Hello world!");  
  </script>  
</head>
```

2.3 Các khái niệm cơ bản trong Javascript

a. Biến

Khai báo biến: `var tên_biến = giá_trị_khởi_tạo;`

Ví dụ: `var i = 10;`

Từ khóa `var` có thể bỏ qua (có nghĩa là không cần khai báo biến)

Quy tắc đặt tên biến: có thể dùng chữ cái, số và ký tự “_” để đặt tên biến, nhưng tên biến không được bắt đầu bằng số. Tên biến không được trùng với các từ khóa của Javascript. Javascript phân biệt chữ hoa và chữ thường nên biến a và biến A là 2 biến khác nhau.

b. Ghi chú

Sử dụng `//` : ghi chú 1 dòng

Sử dụng `/* */`: ghi chú nhiều dòng

2.4 Hàm trong Javascript

Hàm do người định nghĩa:

```
function ten_ham (các tham số)
{
    //Khởi lệnh
}
```

Một số hàm thông dụng trong Javascript:

- **eval(s)**: thực thi chuỗi s. Ví dụ eval("1+1"), cho kết quả là 2.
- **isNaN(n)**: kiểm tra n có phải là số hay không. Ví dụ: isNaN("abc") cho kết quả là true, isNaN("7") cho kết quả là false.
- **Number(o)**: chuyển đối tượng o thành kiểu số. Ví dụ: Number(true) cho kết quả là 1, Number("123") cho kết quả là 123.
- **parseInt(s [,x])**: chuyển chuỗi s, 1 số có cơ số x, sang kiểu số nguyên. Nếu không chuyển được sẽ trả về NaN. Ví dụ: parseInt("101",2) cho kết quả là 5, parseInt("A",16) cho kết quả là 10, parseInt("abc") cho kết quả là NaN.

2.5 Sự kiện trong Javascript

Các sự kiện cung cấp các tương tác với các cửa sổ trình duyệt và tài liệu hiện hành được load trong trang web và các đối tượng có trong trang web. Một vài sự kiện trong Javascript:

onblur	: Xảy ra khi một đối tượng not active
onchange	: Xảy ra khi giá trị của một đối tượng bị thay đổi
onclick	: Xảy ra khi click vào đối tượng
ondblclick	: Xảy ra khi double click vào đối tượng
onerror	: Xảy ra khi javascript có lỗi
onfocus	: Xảy ra khi đối tượng có quyền điều khiển
onload	: Xảy ra khi khởi tạo một trang
onreset	: Xảy ra khi reset form
onscroll	: Xảy ra khi cuộn (scroll) trang
onselect	: Xảy ra khi chọn text trong textfield hay textarea.
onsubmit	: Xảy ra khi User xác nhận đã nhập xong dữ liệu
onmousedown	: Xảy ra khi nhấn chuột
onmousemove	: Xảy ra khi di chuyển chuột
onmouseout	: Xảy ra khi mouse di chuyển ra khỏi đối tượng
onmouseover	: Xảy ra khi mouse di chuyển qua đối tượng
onmouseup	: Xảy ra khi nhả chuột

- `onkeydown` : Xảy ra khi nhấn 1 phím
- `onkeypress` : Xảy ra khi nhấn và nhả 1 phím
- `onkeyup` : Xảy ra khi nhả 1 phím

Javascript có thể thực hiện một đoạn code sau 1 khoảng thời gian xác định gọi là sự kiện thời gian.

- `var t = setTimeout("s",x)`: thực thi đoạn code s sau x millisecond.
- `clearTimeout(t)`: hủy bỏ thiết lập setTimeout được gán cho biến t.
- `var t=setInterval("s",x)`: thực thi đoạn code s sau mỗi x millisecond.
- `clearInterval(t)`: hủy bỏ thiết lập setInterval được gán cho biến t.

Sử dụng sự kiện trong **Javascript**:

`<html_tag các_thuộc_tính_sự_kiện= "khởi_lệnh">`

Ví dụ khi click vào nút Cộng (sự kiện `onclick` xảy ra trên đối tượng nút Cộng), **Javascript** lấy giá trị trong ô số 1 + giá trị trong ô số 2 và gán giá trị kết quả vào ô kết quả.

Số 1:	<input type="text" value="3"/>	txt1
Số 2:	<input type="text" value="4"/>	txt2
<input type="button" value="Cộng"/>		
KQ:	<input type="text" value="7"/>	

```
<input type="button" name="btnKQ" id="btnKQ" value="Cộng"
onclick="cong(document.getElementById('txt1').
value,document.getElementById('txt2').value);" />
```

2.6 Javascript & Form.

Xác thực JavaScript

Hãy nhập dữ liệu có trong khoảng 1 đến 10:

Đây là một nút nhấn để gọi hàm `myFunction` khi người dùng nhấn vào nút này.

```
<h2> Xác thực JavaScript</h2>
<p>Hãy nhập dữ liệu có trong khoảng 1 đến 10:</p>
<input id="numb">
<button type="button" onclick="myFunction()">Submit</button>
<p id="demo"></p>
```

```
<script>
function myFunction() {
    let x = document.getElementById("numb").value;
    let text;
    if (isNaN(x) || x < 1 || x > 10) {
        text = "Không tìm thấy dữ liệu";
    } else {
        text = "OK";
    }
    document.getElementById("demo").innerHTML = text;
}
</script>
```

- `function myFunction() {...}`: Đây là một hàm JavaScript được gọi khi người dùng nhấn vào nút "Submit".
- `document.getElementById("numb").value`: Đây là cách lấy giá trị nhập vào từ trường có id là "numb".
- `if (isNaN(x) || x < 1 || x > 10) {...}`: Đây là một câu lệnh điều kiện để kiểm tra xem giá trị nhập vào có phải là một số không hoặc có nằm ngoài khoảng từ 1 đến 10 không.
- `document.getElementById("demo").innerHTML = text;`: Đây là cách để thay đổi nội dung của thẻ có id là "demo" thành giá trị của biến text tùy thuộc vào kết quả của câu lệnh điều kiện.

2.7 Getter - setter.

getter và *setter* là các hàm hoặc phương thức được dùng để lấy ra hoặc thiết lập giá trị cho các biến. Khái niệm *getter* - *setter* rất phổ biến trong ngôn ngữ lập trình.

Hầu hết các ngôn ngữ lập trình bậc cao đều có bộ cú pháp để thực hiện get và set, bao gồm cả JavaScript.

```
JS demo.js > ...
1  var obj = {
2      foo: 'this is the value of foo',
3      getFoo: function() {
4          return this.foo;
5      },
6      setFoo: function(val) {
7          this.foo = val;
8      }
9  }
10
11  console.log(obj.getFoo());
12  // "this is the value of foo"
13
14  obj.setFoo('hello');
15
16  console.log(obj.getFoo());
17  // "hello"
18
```

Đây là cách đơn giản nhất để khởi tạo *getter* và *setter*. Có 1 thuộc tính foo và 2 phương thức:

- `setFoo` để gán giá trị cho thuộc tính foo.
- `getFoo` để trả về giá trị của thuộc tính foo

- ⇒ Để khởi tạo một *getter* ta chỉ việc thêm từ khóa *get* vào trước khai báo hàm để biến hàm đó thành *getter*; cũng tương tự với từ khóa *set* trước hàm *setter*. Cú pháp như sau:

```

1  var obj = {
2      fooVal: 'this is the value of foo',
3      get foo() {
4          return this.fooVal;
5      },
6      set foo(val) {
7          this.fooVal = val;
8      }
9  }
10
11  // getter
12  console.log(obj.foo);
13  // "this is the value of foo"
14
15  // setter
16  obj.foo = 'hello';
17
18  console.log(obj.foo);
19  // "hello"
20

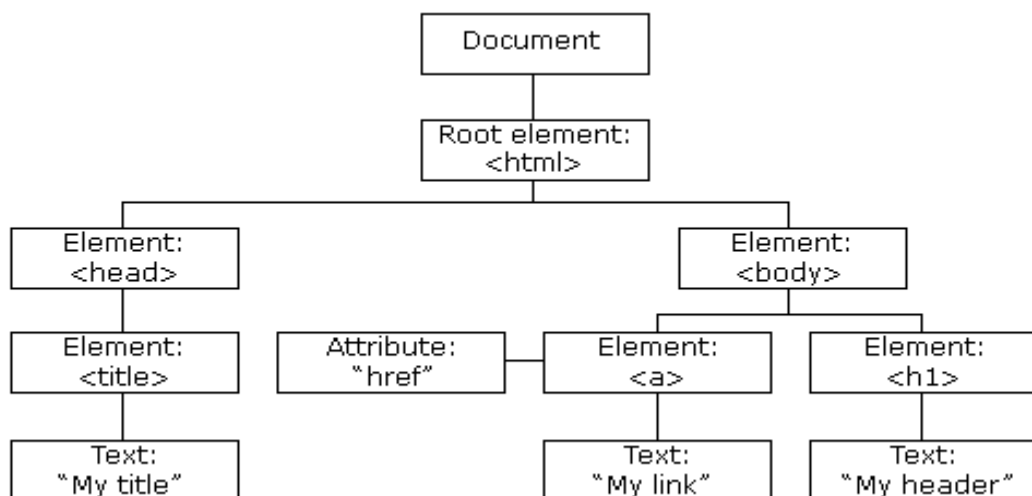
```

3. DOM.

3.1 DOM là gì?

DOM là viết tắt của **D**ocument **O**bject **M**odel, tạm dịch là mô hình các đối tượng trong tài liệu. DOM dùng để truy xuất và thao tác trên các tài liệu có cấu trúc dạng HTML hay XML bằng các ngôn ngữ lập trình thông dịch (*scripting language*) như JavaScript, PHP, Python.

3.2 Cấu trúc của DOM đối với HTML.



- **Nút gốc:** chính là tài liệu HTML, thường được biểu diễn bởi thẻ <html>.
- **Nút phần tử:** biểu diễn cho 1 phần tử HTML.
- **Nút văn bản:** mỗi đoạn kí tự trong tài liệu HTML, bên trong 1 thẻ HTML đều là 1 nút văn bản. Đó có thể là tên trang web trong thẻ <title>, tên đề mục trong thẻ <h1>, hay một đoạn văn trong thẻ
- Ngoài ra, còn có nút thuộc tính (attribute node) và nút chú thích (comment node).
- **Quan hệ giữa các nút:**
 - o **Nút gốc** (*document*) luôn là nút đầu tiên. Tất cả các nút không phải là nút gốc đều chỉ có 1 **nút cha** (*parent*).
 - o Một nút có thể có một hoặc nhiều con, nhưng cũng có thể không có con nào.
 - o Những nút có cùng nút cha được gọi là các **nút anh em** (*siblings*). Trong các nút anh em, nút đầu tiên được gọi là **con cả** (*firstChild*) và nút cuối cùng là **con út** (*lastChild*).

3.3 Các loại DOM trong JS

- o **DOM Document:** có nhiệm vụ lưu trữ toàn bộ các thành phần trong tài liệu của website.
- o **DOM Element:** có nhiệm vụ truy xuất tới thẻ HTML nào đó thông qua các thuộc tính như tên class, id, name của thẻ HTML.

- Truy xuất đến 1 thẻ HTML thông qua ID:

Tên thẻ HTML chính là tên các thẻ như: p, a, div, input ...

```
// Lấy thẻ input
var element = document.getElementById('website');

// Lấy giá trị của thẻ input
document.write(element.value);
```

- Truy xuất đến 1 thẻ HTML thông qua tên của thẻ:

```
// Lấy thẻ input
var element = document.getElementsByTagName('input');

// Lấy giá trị của thẻ input
document.write(element[0].value);
```

- Truy xuất đến 1 thẻ HTML thông qua tên class.

```
var element = document.getElementsByClassName('input');
```

Tương tự như truy xuất thông qua tên của thẻ **HTML**, cách truy xuất thông qua tên **class** sẽ trả về một mảng các object nên chúng ta phải sử dụng cú pháp truy xuất mảng để chọn đúng đối tượng muốn lấy.

```
// Lấy thẻ input
var element = document.getElementsByClassName('website');

// Lấy giá trị của thẻ input
document.write(element[0].value);
```

- Truy xuất đến 1 thẻ HTML thông qua cú pháp của CSS Selector.

```
var element = document.querySelector("selector.css");
```

Cách truy xuất đến thẻ **HTML** thông qua tên class thường sẽ trả về hàng loạt các kết quả nên đôi khi sẽ có những kết quả mà chúng ta không mong muốn. Chính vì vậy, DOM trong **JavaScript** có một phương thức kết hợp với **CSS Selector** để truy vấn có độ chính xác cao hơn.

Làm thế nào chúng ta có thể chọn đúng một thẻ input nằm trong thẻ div và có class="website"?

```
<html>
<body>
  <input type="text" value="thẻ không cần lấy">
  <div>
    <input type="text" class="website" value="Thẻ Cần lấy">
    <input type="text" value="thẻ không cần lấy">
  </div>
</body>
</html>
```

Trong **CSS Selector**, để chọn thẻ input nằm trong thẻ div và có class="website" thì cú pháp là: div input.website.

```
var element = document.querySelector("div input.website");
```

- o **DOM HTML:** có nhiệm vụ thay đổi giá trị nội dung và giá trị thuộc tính của các thẻ HTML

a) Thay đổi và lấy nội dung bên trong thẻ HTML.

- Thay đổi nội dung của 1 thẻ HTML

```
var html = document.getElementById("content").innerHTML = "<h1>...</h1>";
```

- Lấy nội dung bên trong 1 thẻ HTML

```
var html = document.getElementById("content").innerHTML
```

```

<html>
<body>
  <script language="javascript">
    // Hàm thiết lập nội dung cho thẻ div#content
    function set_content()
    {
      document.getElementById("content").innerHTML = "<h1>Nội dung đã được thay đổi</h1>";
    }

    // Hàm lấy nội dung cho thẻ div#content
    function get_content()
    {
      var html = document.getElementById("content").innerHTML;
      alert("Nội dung cần lấy là: " + html);
    }
  </script>
  <div id="content">Nội dung của thẻ DIV</div>
  <input type="button" value="Lấy nội dung" id="get_content" onclick="get_content()" />
  <input type="button" value="Thay đổi nội dung" id="set_content" onclick="set_content()" />
</body>
</html>

```

b) Thay đổi và lấy giá trị thuộc tính của thẻ HTML.

- Thay đổi giá trị của 1 thuộc tính HTML bất kì

```
document.getElementById("element").attributeName = "new value";
```

- Lấy giá trị của 1 thuộc tính HTML

```
var value = document.getElementById("element").attributeName;
```

VD: Viết chương trình khi click vào một button thì chuyển nó thành textbox và tiếp tục click vào textbox thì sẽ đổi thành button.

```

<html>
<body>
  <script language="javascript">
    function change()
    {
      // Lấy đối tượng
      var object = document.getElementById("object");

      // Lấy thuộc tính type
      var type = object.type;

      // Kiểm tra thuộc tính type và thay đổi
      if (type == "button")
      {
        object.type = 'text';
      }
      else
      {
        object.type = "button";
      }
    }
  </script>
  <input type="button" value="Click me" onclick="change()" id="object" />
</body>
</html>

```


- **DOM CSS:** có nhiệm vụ thay đổi các định dạng CSS của thẻ HTML.
- **DOM Event:** có nhiệm vụ gán các sự kiện như onclick(), onload() vào các thẻ HTML
- **DOM EventListener:** có nhiệm vụ lắng nghe các sự kiện tác động lên thẻ HTML đó.
- **DOM Navigation:** dùng để quản lý, thao tác với các thẻ HTML, thể hiện mối quan hệ cha – con của các thẻ HTML.
- **DOM Node, Nodelist:** có nhiệm vụ thao tác với HTML thông qua đối tượng (Object).

4. Thời gian thực hành

120 phút

5. Đánh giá

Thang điểm tối đa: 10 điểm/Lab

6. Ví dụ

Chạy ví dụ form đặt hàng trong **Lab07/Vi_du/index.html**, ta thấy có các input để nhập thông tin. Nếu nhập sai hoặc thiếu một thông tin, sau khi nhấn nút **ĐẶT HÀNG**, form sẽ hiển thị một dòng chữ màu đỏ để người dùng dễ dàng nhận biết và bổ sung thông tin chính xác. Ngược lại, nếu nhập đầy đủ thông tin, form sẽ thông báo đã đặt hàng thành công.

Thông tin thanh toán

Tên*

Họ*

Quốc gia*

Địa chỉ*

Địa chỉ thường trú

Căn hộ, suite, unite ect (tùy chọn)

Quê quán*

Mã bưu / ZIP*

Điện thoại*

Email*

☐ Gửi đến một địa chỉ khác?

Ghi chú hóa đơn*

Ghi chú về đơn đặt hàng của bạn, ví dụ: ghi chú đặc biệt để giao hàng.

Hóa đơn của bạn

Sản phẩm	Giá
Tổng phụ	0 đ
Tổng hóa đơn	0 đ

Độc sách là một điều rất tốt cho cuộc sống của bạn, hãy luôn đặt mình vào hoàn cảnh nào đó, bạn sẽ nhận ra cuộc sống này rất thú vị.

☐ Paypal

ĐẶT HÀNG

hello@gmail

Vui lòng nhập một địa chỉ email hợp lệ.

This page says

Đặt hàng thành công!

OK

7. Yêu cầu thực hành

Bài 1: (5 đ)

Dựa vào ví dụ trên, hãy hoàn thành các hiển thị thông báo màu đỏ cho người dùng biết chỗ điền còn thiếu.

- Nếu người dùng điền thiếu một Input nào đó, sau khi nhấn **ĐẶT HÀNG**, lập tức sẽ thông báo tại vị trí đó còn thiếu hoặc sai sót.
- Nếu đã điền đầy đủ thông tin, sau khi **ĐẶT HÀNG** sẽ hiển thị thông báo Đặt hàng thành công!.

Thông tin thanh toán

Tên*

Vui lòng nhập tên của bạn.

Họ*

Vui lòng nhập họ của bạn.

Quốc gia*

Vui lòng nhập quốc gia của bạn.

Địa chỉ*

Vui lòng nhập địa chỉ của bạn.

Vui lòng nhập nơi của bạn.

Quê quán*

Vui lòng nhập quê quán của bạn.

Mã bưu / ZIP*

Vui lòng nhập mã bưu của bạn.

Điện thoại*

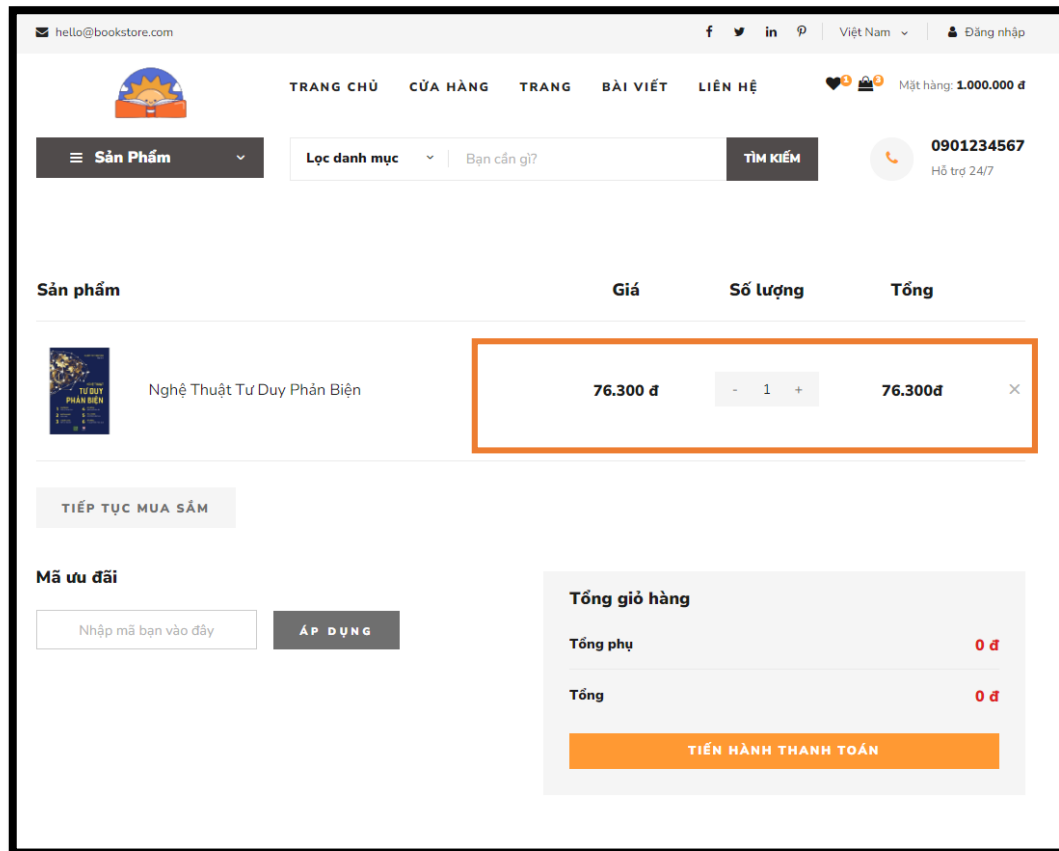
Vui lòng nhập số điện thoại của bạn.

Email*

Vui lòng nhập một địa chỉ email hợp lệ.

Bài 2: (5 đ)

Chạy trang Giỏ hàng trong **Lab07/demo_bai2/index.html**.



Hãy viết một hàm **function updateTotal()** tăng giảm số lượng sản phẩm. Khi ta tăng hoặc giảm số lượng thì tổng sẽ bằng **Giá * Số lượng**.


Hãy bổ sung hàm đó ngay bên dưới đoạn source code sau (đoạn source code nằm trong **Lab07/demo_bai_2/js/main.js**):

```

193  /*-----
194  |   Quantity change
195  |----- */
196  var proQty = $('.pro-qty');
197  proQty.prepend('<span class="dec qtybtn">-</span>');
198  proQty.append('<span class="inc qtybtn">+</span>');
199  proQty.on('click', '.qtybtn', function () {
200      var $button = $(this);
201      var oldValue = $button.parent().find('input').val();
202      if ($button.hasClass('inc')) {
203          var newVal = parseFloat(oldValue) + 1;
204      } else {
205          // Don't allow decrementing below zero
206          if (oldValue > 0) {
207              var newVal = parseFloat(oldValue) - 1;
208          } else {
209              newVal = 0;
210          }
211      }
212      $button.parent().find('input').val(newVal);
213      updateTotal();
214  });

```

Kết quả:

Sản phẩm	Giá	Số lượng	Tổng
 Nghệ Thuật Tư Duy Phản Biện	76.300 đ	- 3 +	228.900đ ×