Trong Javascript có một sự kiện được sử dụng khá nhiều đó là sự kiện onload. Nếu như bạn đã từng xem qua jQuery thì bạn cũng sẽ thấy thường trước khi viết người ta sẽ dùng cú pháp $(document).ready() thì bản chất đó chính là sự kiện onload trong Javascript.

onload Xảy ra khi thẻ HTML bắt đầu chạy, nó giống như [hàm khởi tạo](https://freetuts.net/ham-khoi-tao-va-ham-huy-trong-lap-trinh-huong-doi-tuong-php-36.html) trong [lập trình hướng đối tượng](https://freetuts.net/tag/lap-trinh-huong-doi-tuong) vậy đó.

Local variable and Local Variable

Nếu trong hàm có sử dụng từ khóa var và để tạo một biến và tên của biến đó **đã tồn tại ở bên ngoài** (toàn cục) thì lúc này bên trong hàm nó sẽ hiểu là đang sử dụng biến cục bộ chứ không ảnh hưởng gì biến bên ngoài cả.

**Ví dụ**: [XEM DEMO](https://freetuts.net/editor.html?id=308)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | <html>      <body>          <script language="javascript">              // Biến toàn cục            var comment = "Nội dung comment toàn cục";              // Hàm có sử dụng biến toàn cục            function add\_comment()            {               var comment = "Nội dung comment cục bộ";               alert(comment);            }              // Gọi fuction comment            add\_comment();              // In biến toàn cục            alert(comment);            </script>      </body>  </html> |

Nếu bạn không sử dụng từ khóa var để tạo tên biến trong hàm thì nó sẽ sử dụng biến toàn cục nên mọi thay đổi của biến đó trong hàm sẽ bị ảnh hưởng ra ngoài hàm.

**Ví dụ**: [XEM DEMO](https://freetuts.net/editor.html?id=309)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | <html>      <body>          <script language="javascript">              // Biến toàn cục            var comment = "Nội dung comment trước khi thay đổi";              // Hàm có sử dụng biến toàn cục            function add\_comment()            {               comment = "Nội dung comment đa bị thay đổi";               alert(comment);            }              // Gọi fuction comment            add\_comment();              // In biến toàn cục            alert(comment);            </script>      </body>  </html> |

Nếu liên tưởng qua ngôn ngữ PHP thì không thể thực hiện như hai ví dụ này được mà nó phải thông qua một từ khóa nữa đó là global thì trong hàm mới sử dụng được biến toàn cục ở ngoài hàm.

**Vị trí đặt JS**

Tuy nhiên, mới đây hai thuộc tính async và defer được giới thiệu để sử dụng cho thẻ <script>. Sử dụng hai thuộc tính này sẽ giúp mã lệnh JavaScript có thể được tải bất đồng bộ cùng với mã HTML. Điều này giúp cho trình duyệt có thể đồng thời tải cả mã HTML và JavaScript cùng một lúc.

Khi đó bạn nếu bạn sử dụng thẻ <script> ở bên trong thẻ <head> như sau:

</html>

<head>

<script src="/js/main.js" type="text/javascript" async></script>

</head>

<body>

...

</body>

</html>

Thì tốc độ tải trang sẽ nhanh hơn khi để thẻ <script> ở cuối trang (trước thẻ <body>) vì lúc này chúng ta tận dụng được lợi thế trình duyệt tải về đồng thời HTML và JavaScript cùng một lúc.

Có 3 cách để chèn CSS/Javascript:

* Inline:

|  |  |
| --- | --- |
| Inline CSS | |
| 1 | <div style="color:red;"> |
| Inline JS | |
| 1 | <button class="btn" onclick="alert('Hello World')"> |

* Nhúng (embed)

|  |  |
| --- | --- |
| Embedded CSS | |
| 1 | <style>.red {color: red;}</style> |
| Embedded JS | |
| 1 | <script>alert('Hello World');</script> |

* File bên ngoài

|  |  |
| --- | --- |
| External CSS | |
| 1 | <link rel="stylesheet" href="file.css"> |
| External JS | |
| 1 | <script type="text/javascript" src="script.js"> |

Đối với 2 cách inline/nhúng, số lượng kết nối đến server sẽ giảm, nhưng việc chèn code sẽ làm tăng kích thước HTML. 2 cách này chỉ thực sự hữu ích khi bạn có ít code CSS/JS. Cách thứ 3 nên được sử dụng cho hầu hết các trường hợp còn lại, giúp bạn quản lý code tốt hơn và giúp trình duyệt có thể lưu cache lại cho các file này.

**1. Synchronous là gì?**

**Synchronous có nghĩa là xử lý đồng bộ**, chương trình sẽ chạy theo từng bước và chỉ khi nào bước 1 thực hiện xong thì mới nhảy sang bước 2, khi nào chương trình này chạy xong mới nhảy qua chương trình khác. Đây là nguyên tắc cơ bản trong lập trình mà bạn đã được học đó là khi biên dịch các đoạn mã thì trình biên dịch sẽ biên dịch theo thứ tự từ trên xuống dưới, từ trái qua phải và chỉ khi nào biên dịch xong dòng thứ nhât mới nhảy sang dòng thứ hai, điều này sẽ sinh ra một trạng thái ta hay gọi là **trạng thái chờ**. Ví dụ trong quy trình sản xuất dây chuyền công nghiệp được coi là một hệ thống xử lý đồng bộ.

Synchronous hai mặt là mặt xấu và mặt tốt.

**Mặt tốt của Synchronous**

Chương trình sẽ chạy theo đúng thứ tự và có nguyên tắc nên sẽ không mắc phải các lỗi về tiến trình không cần thiết. Không chỉ trong lập trình mà trong thực tế cũng vậy, một công ty đưa ra quy trình đồng bộ sẽ đảm bảo được chất lượng của sản phẩm, nếu bị lỗi thì sẽ biết ngay là lỗi tại quy trình nào và từ đó sẽ dễ dàng khắc phục.

**Mặt xấu của Synchronous**

Chương trình chạy theo thứ tự đồng bộ nên sẽ sinh ra trạng thái chờ và là không cần thiết trong một số trường hợp, lúc này bộ nhớ sẽ dễ bị tràn vì phải lưu trữ các trạng thái chờ vô duyên đó.

Khi bạn viết một chương trình quản lý và trong đó có thao tác lưu, mỗi khi lưu bạn yêu cầu người dùng có muốn lưu hay không? Nếu muốn lưu thì click Yes, ngược lại click No. Trường hợp nay gây tai họa nếu người dùng vô tình chỉ click Lưu mà không chú ý đến câu hỏi mà hệ thống đưa ra nên ngồi nhâm nhi cafe, đột nhiên cúp điện thế là cứ tưởng đã lưu rồi :) Vậy quy trình xử lý nên đưa ra chức năng lưu tự động, nghĩa là thao tác lưu sẽ bỏ qua bước hỏi đáp kia đi, không nhất thiết phải chờ nó OK mới lưu.

**3. CSS đặt trước JS**

Bạn đã thấy JS làm gián đoạn việc load HTML như thế nào. Do đó, nếu JS của bạn không có xử lý gì quá đặc biệt thì chỉ nên load JS sau cùng. CSS thì ngược lại, nên được đặt ở đầu trang để người dùng khi đợi tải trang không phải thấy một trang web xấu xí khi chưa có CSS.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | <html lang="vi">  <head>      <meta charset="UTF-8" />      <title>Test | KarmiPhuc</title>      <!-- đặt CSS ở thẻ head -->      <link...>  </head>  <body>      <!-- HTML here -->      ...  **<!-- đặt JS trước khi đóng body -->**      <script...>  </body>  </html> |

**4. Đừng sử dụng @import**

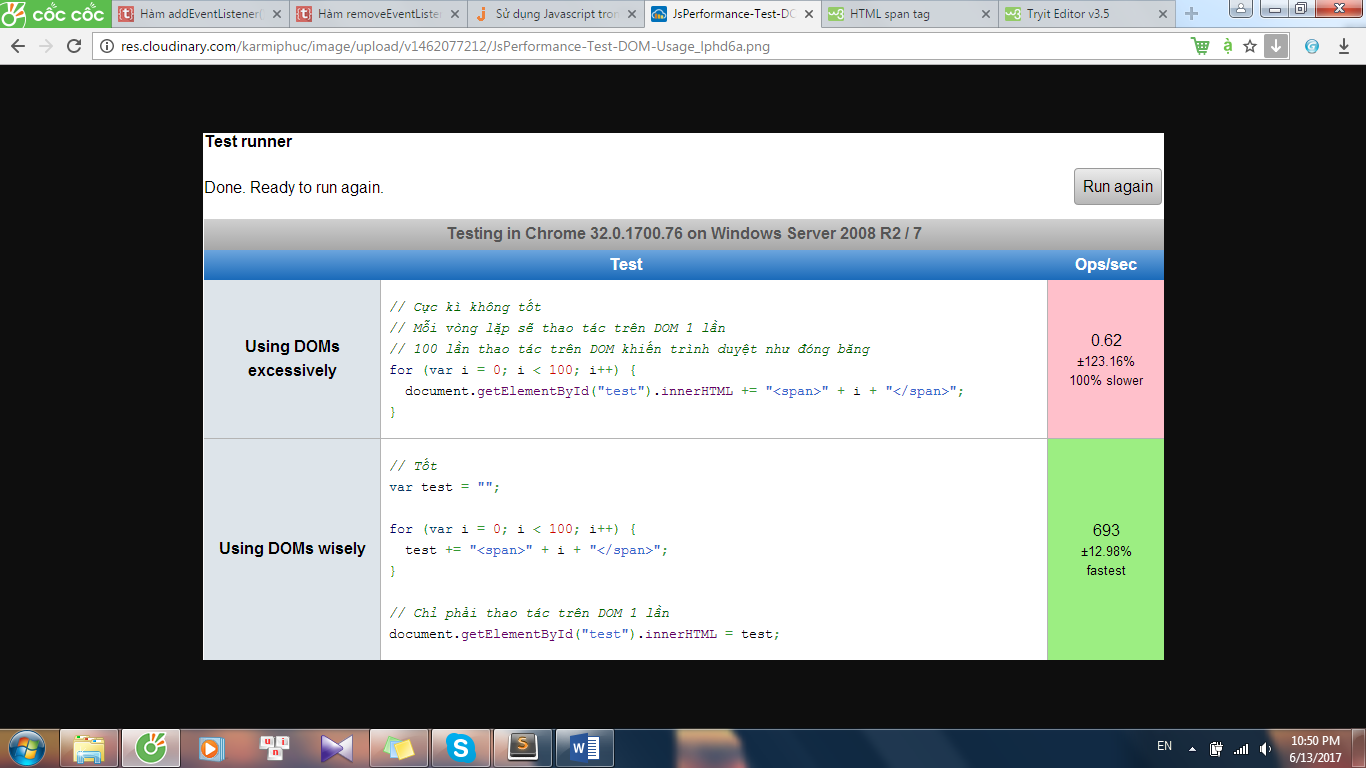
Có 2 cách để chèn 1 file CSS bên ngoài vào trang web:

|  |  |
| --- | --- |
| External CSS Link | |
| 1 | <link rel="stylesheet" href="file.css"> |
| CSS Import via external or inline style | |
| 1  2 | /\* Chỉ gọi bên trong 1 file CSS ngoài, hoặc CSS inline\*/  @import url('style.css'); |

Với cách thứ 1, trình duyệt có thể tải nhiều file CSS đồng thời. Còn với cách thứ 2, file style.css chỉ có thể được tải về sau khi file CSS gốc đã tải xong, làm hạn chế khả năng tải song song của trình duyệt.

<p>My mother has **<span style="color:blue;font-weight:bold">blue</span>** eyes and my father has **<span style="color:darkolivegreen;font-weight:bold">dark green</span>** eyes.</p>

**Hạn chế thao tác các DOM**



## jQuery là gì?

jQuery là một thư viện kiểu mới của JavaScript, được tạo bởi John Resig vào năm 2006 với một phương châm tuyệt vời: **Write less, do more - Viết ít hơn, làm nhiều hơn**.

jQuery làm đơn giản hóa việc truyền tải HTML, xử lý sự kiện, tạo hiệu ứng động và tương tác Ajax. Với jQuery, khái niệm Rapid Web Development đã không còn quá xa lạ.

jQuery là một bộ công cụ tiện ích JavaScript làm đơn giản hóa các tác vụ đa dạng với việc viết ít code hơn. Dưới đây liệt kê một số tính năng tối quan trọng được hỗ trợ bởi jQuery:

* **Thao tác DOM −** jQuery giúp dễ dàng lựa chọn các phần tử DOM để *traverse (duyệt)* một cách dễ dàng như sử dụng CSS, và chỉnh sửa nội dung của chúng bởi sử dụng phương tiện Selector mã nguồn mở, mà được gọi là **Sizzle**.
* **Xử lý sự kiện −** jQuery giúp tương tác với người dùng tốt hơn bằng việc xử lý các sự kiện đa dạng mà không làm cho HTML code rối tung lên với các Event Handler.
* **Hỗ trợ AJAX −** jQuery giúp bạn rất nhiều để phát triển một site giàu tính năng và phản hồi tốt bởi sử dụng công nghệ AJAX.
* **Hiệu ứng −** jQuery đi kèm với rất nhiều các hiệu ứng đa dạng và đẹp mắt mà bạn có thể sử dụng trong các Website của mình.
* **Gọn nhẹ −** jQuery là thư viện gọn nhẹ - nó chỉ có kích cỡ khoảng 19KB (gzipped).
* **Được hỗ trợ hầu hết bởi các trình duyệt hiện đại −** jQuery được hỗ trợ hầu hết bởi các trình duyệt hiện đại, và làm việc tốt trên IE 6.0+, FF 2.0+, Safari 3.0+, Chrome và Opera 9.0+
* **Cập nhật và hỗ trợ các công nghệ mới nhất −** jQuery hỗ trợ CSS3 Selector và cú pháp XPath cơ bản

.

**DOM**

* DOM document: có nhiệm vụ lưu trữ toàn bộ các thành phần trong tài liệu của website
* 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
* DOM HTML: có nhiệm vụ thay đổi giá trị nội dung và giá trị thuộc tính của các thẻ 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 Listener: 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)

addEventListener or use directly onclick, onkeyup…

Cách 1: input.addEventListener('keyup', function(){

document.getElementById("showContent").innerHTML = input.value;

if (input.value.length > 5){

alert("It's over 5 letters");

}

});

Cách 2: dùng trực tiếp

<input type="text" id="message" value="" onkeyup="show\_result()"/>

|  |  |  |
| --- | --- | --- |
| 1 | onclick | Xảy ra khi click vào thẻ HTML |
| 2 | ondbclick | Xảy ra khi double click vào thẻ HTML |
| 3 | onchange | Xảy ra khi giá trị (value) của thẻ HTML đổi. Thường dùng trong các đối thẻ form input |
| 4 | onmouseover | Xảy ra khi con trỏ chuột bắt đầu đi vào thẻ HTML |
| 5 | onmouseout | Xảy ra khi con trỏ chuột bắt đầu rời khỏi thẻ HTML |
| 6 | onmouseenter | Tương tự như onmouseover |
| 7 | onmouseleave | Tương tự như onmouseout |
| 8 | onmousemove | Xảy ra khi con chuột di chuyển bên trong thẻ HTML |
| 9 | onkeydown | Xảy ra khi gõ một phím bất kì vào ô input |
| 10 | onload | Sảy ra khi thẻ HTML bắt đầu chạy, nó giống như [hàm khởi tạo](https://freetuts.net/ham-khoi-tao-va-ham-huy-trong-lap-trinh-huong-doi-tuong-php-36.html) trong [lập trình hướng đối tượng](https://freetuts.net/tag/lap-trinh-huong-doi-tuong) vậy đó. |
| 11 | onkeyup | Xảy ra khi bạn gõ phím nhưng **lúc bạn nhã phím ra sẽ được kích hoạt** |
| 12 | onkeypress | Xảy ra khi bạn nhấn môt phím vào ô input |
| 14 | onblur | Xảy ra khi con trỏ chuột rời khỏi ô input |
| 15 | oncopy | Xảy ra khi bạn copy nội dung của thẻ |
| 16 | oncut | Xảy ra khi bạn cắt nội dung của thẻ |
| 17 | onpaste | Xảy ra khi bạn dán nội dung vào thẻ |