Javascript Nền Tảng

Mục lục

[ẩn]

[sửa] Giới Thiệu

Bài này lập ra 1 lịch trình để học nền tảng Javascript, lấy ví dụ là product MyWorkplace.
Yêu cầu: hiểu và biết cách dùng prototype framework

[sửa] Lịch sử phát triển của JavaScript

-JavaScript theo phiên bản hiện nay mà chúng ta đang sử dụng,là một Ngôn ngữ lập trình kịch bản dựa trên đối tượng.JavaScript đươc sử dụng rộng rãi cho các trang web hiện nay,nhưng nó cũng được dùng để tạo khả năng viết script bằng việc sử dụng các đối tượng có sẵn trong các ứng dụng.
-Vậy JavaScript xuất hiện từ khi nao?Vâng,Nó vốn được phát triển bởi Brendan Eich tại hãng truyền thông NetScape với tên đầu tiên là MoCha, qua một thời gian tiếp theo thì Javascript đổi tên là LiveScript.Và cái tên Javascript là cái tên hiện tại ngày nay.JavaScript có cấu trúc tương tự như C. Các tập tin của Javascript được lưu với định dạng là .js(vd: demo.js).
-Phiên bản mới nhất của JavaScript là phiên bản 1.5, tương ứng với ECMA-262 bản 3. ECMAScript là phiên bản chuẩn hóa của JavaScript. Trình duyệt Mozilla phiên bản 1.8 beta 1 có hỗ trợ không đầy đủ cho E4X - phần mở rộng cho JavaScript hỗ trợ làm việc với XML, được chuẩn hóa trong ECMA-357.

[sửa] Vậy JavaScript khác với Java và JScript như thế nào?

-Cùng với sự ra đời của Java như một hiện tượng thì LiveScript cũng đã đổi thành JavaScript để thu hút những Người Lập Trình hơn.Suy cho cùng thì Java và JavaScript hoàn toàn khác nhau(Bạn đừng nhầm lẫn Chúng giống nhau là java mà cho rằng chúng có họ hàng),ngoại trừ cú pháp của chúng giống với C
-Sau thành công của JavaScript thì Microsoft bắt đầu phát triển JScript, một ngôn ngữ có cùng ứng dụng và tương thích được với JavaScript.JScript được bổ sung vào IE 3.0

[sửa] Vậy JavaScript nó có thể làm được gì?Ứng dụng của JavaScript như thế nào?

- Javascript có thể làm được rất nhiều thứ chẳng hạn :
  - Nó có thể tạo ra một HTML động
  - Nó có thể thiết kế một giao diện

    - Nó có thể tạo một sự kiện (Event) cho các button.
    - Nó cũng có thể tạo ra một Cookies.
    - Và còn nhiều thứ khác nữa mà bạn khó có thể tưởng tượng được lợi ích của nó.Ta có thể tìm hiểu kỹ hơn.
- Ứng dụng của Javascript.

-Hiện nay,có rất nhiều trang web trên mạng sử dụng JavaScript để thiết kế trang web động và một số hiệu ứng hình ảnh thông qua DOM để tạo sức thu hút ngững con mạng.
-JavaScript dùng để xử lý một số thao tác không thể thực hiện được với một trang HTML bình thường như kiểm tra thông tin nhập vào,username,pass,hay tự động thay đổi hình ảnh...vv -Một số công nghệ nổi bậc kết hợp JavaScript tương tác với DOM như:DHTML, Ajax và SPA.

[sửa] Các cú pháp của JavaScript

[sửa] Chú Thích

-Trong khi lập trình chúng ta cũng có thể vô hiêu hóa một đoạn nào đó(chú thích) để dễ tìm kiếm những đoạn code cần tìm.
-Chú Thích dòng: //chú thích ở đây
-Chú Thích Khối: /* chú thích nhiều dòng*/

[sửa] Khai báo Biến

- Cách Đặt Tên biến.

-Khi đặc tên cho một biến ta chú ý không được bắt đầu bằng một số (vd:1abc), hay chúng ta không được dùng các ký tự đặt biệt (vd:*,+,-...)

Cú Pháp: var x; (đây là cách khai báo biến không nắm giữ giá trị)
Chú ý:Các cách khai báo sau:

```
var X;       ->[hợp lệ]
var x;       ->[hợp lệ]
var _x;      ->[hợp lệ]
var 1x;      ->[không hợp lệ]
```

- Cách khai báo biến

Chúng ta sử dụng từ khóa var để khai báo một biến,các biến có thể nắm giữ giá trị hoặc là không

```
var x;    ->không nắm giữ giá trị
var x=5;  ->nắm giữ giá trị là 5
```

-Code:

```
<script language="JavaScript">
     var tên_biến;
 </script>
```

- Phạm vi của biến (Scope variable).

Khi một biến khai báo có thể là Local hoặc là Global
-Một biến gọi là Local khi chúng được khai báo trong một function(hàm) để phục vụ cho hàm đó.
-Một biến gọi là Glocal khi chúng được khai báo nằm bên ngoài các function(hàm) để phục vụ cho các hàm
-Một ví dụ để ta dễ thấy.(Demo nhỏ)

```
<html>
 <head>
 <title>Hien Thi De Mo</title>

 <script language="JavaScript">
 var a;                        //bien Glocal
 var b=2;                      //bien Glocal
 var result=0;                 //bien Glocal
        function myFunction1(){
        var b=10;              // bien local
        result=a+b;
        document.write("Ket Qua cua ham myFunction1 la :
"+result+"<br>");
        }
        //////
        function myFunction2(){
        result=a+b;
        document.write("Kat Qua cua ham myFunction2 la : "+result);
        }

        ///
        function calculate(){
        var inputText_a=document.getElementById("so_a");
        a=parseInt(inputText_a.value);
        myFunction1();
        myFunction2();
        }
        function init(){
                var btSum_a_b=document.getElementById("btSum_a_b");
                btSum_a_b.onclick=calculate;
        }

 window.onload=init;
 </script>
 </head>

 <body>
 <h1>Hay Nhap Gia Tri cua a:</h1>

 <table border=1>
 <tr>
```

```
 <td><b><u>So a:</u></b> <input type="text" id="so_a" size=20/></td>
 </tr>

 <tr>
 <td align="center"><b><input type="button" id="btSum_a_b"
value="Submit"/></td>
 </tr>
 </table>
 </body>
 </html>
```

- Giới Thiệu Về Hàm (Functions)

Trong JavaScript ta dễ dàng nhận thấy có 2 loại Hàm
-Các Hàm JavaScript đã hỗ trợ sẵn cho chúng ta chỉ việc sử dụng nếu chúng ta hiểu được chúng.Các hàm đó được gọi là :Built-in Functions
-Ngoài ra người dùng có thể định nghĩa ra các hàm để phục vụ cho minh.Các hàm đó được gọi là : User-defined Functions

-
  - Một số hàm JavaScript (Built-in Functions)

a)isNaN(var)
-kiểm tra một biến có phải là chuỗi hay không?.Nếu như không là số thì sẽ trả về giá trị NaN Vd:

```
var bien_1="hello";
var bien_2=2;
if(isNaN(bien_1)){
return true;
}else{
return false;
}
```

b)paserInt(var)
-Chuyển một chuỗi sang số Int vd:

```
function tong(){
var a="12";
var b="13";
var result=parseInt(a)+parseInt(b);
return result;
}
```

c)paserFloat(var)
- Chuyển một chuỗi sang số Float(tương tự như paserInt())
d)eval("")
-Định giá trị cho các statement hoặc expression được lưu trữ như một chuỗi Vd:

```
var X=5;
```

```
Eval(“2*X + 5”);        ->giá trị cuối cùng là 15
```

e)alert()
-Đây là phương thức của Window Object,dùng để gởi một thông báo cho User Vd:

```
alert(“message to the user”);
```

f)prompt("string_a","string_b");
-Dùng để tạo ra một dialog box tương tác với User với 2 button OK ,CANCEL + string_a: ghi một nhãn lên dialog box +string_b:giá trị mặc định trong text box Vd:

```
document.write(prompt("Enter your id:","Emp_id"));
```

-
  - Các hàm do User tự định nghĩa (User-defined Functions)

-Cú pháp:

```
function tenHam(bien_1,bien_2,...){
return value;
}
```

-Hàm có thể chứa hoặc không chứa tham số. vd:Hàm không chứa tham số.

```
function tenHam(){
return value;
}
```

[sửa] Đối tượng

-Trong javaScript có 2 kiểu dữ liệu:Kiểu Cơ bản và Kiểu Đối Tượng

- Kiểu Cơ Bản:là kiểu mà chỉ có một giá trị duy nhất.Sau đây là Bảng Kiểu Cơ bản(Data types)

Data Types	Examples
Number	kiểu dữ liệu thuộc về số (4, 5.3, hoặc là 789)
String	kiểu dữ liệu thuộc về chuỗi ("Hello to you!","554-212-023","KJH566XHJD")
Boolean	Có 2 giá trị True hoặc False

-vd:Bạn muốn gán giá trị cho một biến nào đó có giá trị là kiểu cơ bản.

```
var str="Hello!" ->biến str là kiểu String của javascript
```

- Kiểu Đối Tượng là một thực thể có tên xác định và có thuộc tính(attribute) trỏ đến giá trị,hàm hoặc là một đối tượng khác.(Đối tượng có thể là do javascript cung cấp hoặc là chính do chúng ta tạo ra)

-vd:

```
var checkbox=document.getElementById("ch1"); //lấy một đối tượng
Checkbox
var result=checkbox.checked;                 //lúc này result=true hoặc
false
```

-javaScript cung cấp cho ta một số đối tượng sau:

Anchor, Applet ,Area , Array, Boolean, Button, Checkbox, Date, Document, Event, FileUpload, Form, Frame, Function, Hidden, History, Image, Layer, Math, Object, Reset, Screen, String, Submit, Text, Textarea, Window, Link, Location, Navigator, Number, Option, Password, Radio, RegExp, Select

-Chúng ta có thể tạo ra một đối tượng (OBJECT) và thêm hoặc xóa thuộc tính hoặc hàm trong đối tượng sau khi đối tượng đã được tạo. Để làm việc này cho tất cả các đối tượng được tạo từ cùng một hàm khởi tạo, Chúng ta có thể sử dụng thuộc tính prototype của hàm khởi tạo để truy cập đối tượng nguyên mẫu. Chúng ta không nhất thiết phải tự xóa các đối tượng đã tạo, JavaScript tự động gom rác tất cả những biến không còn được dùng nữa.

-Sau đây là một vd để tạo ra đối tượng **Example** thông qua từ khóa **function**(còn nhiều cách để tạo ra đối tượng,chúng ta xem phần OOP(Object Oriented Programming) của javascript)

```
     <script language="JavaScript">

          function Example() {

              this.attribute1 = "someValue"; // thêm một thuộc tính cho
đối tượng

              this.attribute2 = 234; // thêm thuộc tính nữa cho đối
tượng

              this.function1 = testFunction; // thêm một hàm vào đối
tượng

          }


          function testFunction() {

              alert(this.attribute2); //hiển thị 234
```

```
        }


        var example= new Example; // khởi tạo một đối tượng

        example.function1(); // gọi hàm function1 của đối tượng
sampleObject


        example.attribute3 = 123; // thêm một thuộc tính nữa cho đối
tượng sampleObject


       delete example.attribute1; // xóa bỏ 1 thuộc tính
        delete example; // xóa bỏ đối tượng
    </script>
```

[sửa] Cấu trúc điều khiển

- Câu lệnh IF ..ELSE (rẽ nhánh).

-Cấu trúc if..else được sử dụng trong trường hợp bạn muốn rẽ nhánh.Vd như nếu a thì b, còn không a thì c.Các if..else có thể lồng nhau
-Cú pháp :if không else(nếu biểu thức không đúng thì không làm gì cả)

```
if(biểu_thức){

 khối lệnh được thực hiện nếu biểu thức đúng;

}
```

-VD:

```
<script type="text/javascript">
//Write a "Good morning" greeting if
//the time is less than 10

var d=new Date();
var time=d.getHours();

if (time<10) {
 document.write("<b>Good morning</b>");
}
</script>
```

-Cú pháp : if..else đơn giản

```
if (biểu_thức)
    {
        khối lệnh được thực hiện nếu biểu thức đúng;
    }
    else {
        khối lệnh được thực hiện nếu biểu thức không đúng;
    }
```

-VD:

```
<script type="text/javascript">
//Write "Lunch-time!" if the time is 11

var d=new Date();
var time=d.getHours();

if (time==11)
{
document.write("<b>Lunch-time!</b>");
}else{
document.write("<b>Not Lunch-time!</b>");
}
</script>
```

-Cú pháp : if..else lồng nhau

```
if (biểu_thức_1)
    {
        khối lệnh được thực hiện nếu biểu thức 1 đúng;
    }
    else if (biểu_thức_2)
    {
        khối lệnh được thực hiện nếu biểu thức 2 đúng;
    }
    else
    {
        khối lệnh được thực hiện nếu cả hai biểu thức trên đều không
đúng;
    }
```

-VD:

```
<script type="text/javascript">
var d = new Date()
var time = d.getHours()
if (time<10)
{
document.write("<b>Good morning</b>");
}
else if (time>10 && time<16)
{
document.write("<b>Good day</b>");
```

```
}
else
{
document.write("<b>Hello World!</b>");
}
</script>
```

[sửa] Toán tử điều kiện

-Nếu như điều kiện đúng thì trả về giá trị này, còn nếu sai thì trả về giá trị kia.
-Cú Pháp:

```
var x=(đk) ? giá_trị_x_nhận_nếu_đk_đúng:giá_trị_x_nhận_nếu_đk_sai;
```

-VD:

```
var x=(ham>=1)?1:0; //nếu giá trị ham lớn hơn hoặc bằng 1->x=1;ngược
lại x=0
```

[sửa] Vòng lặp while

-Vòng lặp while có mục đích lặp đi lặp lại một khối lệnh nhất định cho đến khi biểu thức điều kiện trả về false. Khi dùng vòng lặp while phải chú ý tạo lối thoát cho vòng lặp (làm cho biểu thức điều kiện có giá trị false), nếu không đoạn mã nguồn sẽ rơi vào vòng lặp vô hạn, là một lỗi lập trình. Vòng lặp while thường được dùng khi chúng ta không biết chính xác cần lặp bao nhiêu lần. Cú pháp của vòng lặp while như sau:

```
while (biểu_thức_điều_kiện) {
   khối lệnh cần thực hiện nếu biểu_thức_điều_kiện trả về true;
}
```

-VD:

```
     <script language="JavaScript">
        var array=new array[1,"asd","fdg",3];
             while () {
                khối lệnh cần thực hiện nếu biểu_thức_điều_kiện trả về
true;
             }

      </script>
```

[sửa] Vòng lặp do.. while

-Tương tự như vòng lặp while.Nhưng có điểm khác nhau là :Đối với while nếu gặp giá trị biểu thức là false lần đầu tiên chạy vòng lặp thì khối lệnh sẽ không bao giời được thực thi.Còn đối với do..while thì ít nhất thực hiện một lần cho dù điều kiện sai.

-Cú pháp của vòng lặp do ... while như sau:

```
<script language="JavaScript">
     do{
         khối lệnh;
     } while (biểu_thức_điều_kiện);
 </script>
```

-VD:

```
<script language="JavaScript">
     while (0 > 1)
     {
         document.write(" 0 thì bé hơn 1"); // Câu lệnh này sẽ
không bao giờ được thực hiện
     }

     do
     {
         document.write("Hello"); // ta thấy được dòng Hello một
lần duy nhất cho dù điều kiện là sai
     } while (0 > 1);
 </script>
```

[sửa] Vòng lặp for

-Chúng ta thường dùng khi muốn thực hiện một khối lệnh nào đó mà chúng ta biết được số lần đó.
-Cú pháp của vòng lặp for:

```
for(khởi_tạo_biến; biểu_thức_đk; biểu_thức_thay_đổi_theo_biến){
    //khối lệnh cần thực hiện
 }
```

-Đầu tiên vòng lặp chạy thì một biến sẽ được khởi tạo(vd biến đó là x), sau đó x sẽ được đưa vào biểu_thức_đk(vd là exp) để kiểm tra nếu exp=true thì thực hiện khối lệnh.Tiếp theo biểu_thức_thay_đổi_theo_biến được gọi rồi đưa vào exp kiểm tra cho đến khi exp

=false thì vòng lặp dừng.
-VD:

-VD:

```
 <script language="javascript">
  for(var i=1; i<10; i++){
  document.write("Hello "+i); //thực hiện cho đến khi i=9
  }
  </script>
```

[sửa] Vòng lặp for .. in

-Dùng để duyệt qua tất cả các phần tử trong mảng hay Chúng ta cũng có thể duyệt qua tất cả các thuộc tính của một Object.
-Cú pháp của for..in

```
for(bien in Container){
//khối lệnh được thực hiện
}
```

-Vd:Duyệt qua tất cả các thuộc tính của đối tượng SinhVien

```
<html>
<head>

</head>
<body>
<script language="javascript">
 function Sinhvien(){
        this.ten="A";
        this.tuoi=20;
 }

        var sv=new Sinhvien;
 for(var attr in sv){
        document.write(attr+"<br>" );
}
</script>
</body>
</html
```

-Tương tự cho mảng

[sửa] switch

-Cấu trúc switch cũng tương tự như if..else nhưng khác ở chổ nó sẽ kiểm tra bằng với kết quả được đưa ra.Nếu trường hợp mà biến không trùng với những kết quả cần kiểm tra thì nó sẽ thực thi lệnh default
-Cú pháp Switch:

```
switch(biến){
 case value1:code1 thực thi;break;
 case value2:code2 thực thi;break;
 .
 .
 default:code_default thực thi;
}
```

-VD về Swicth:

```
<html>
<head>

</head>
<body>
<script language="javascript">
 var bien="Cat";
 switch(bien){
 case "Cat":document.write("Day la Cat");break;
 case "Dog":document.write("Day la khong Cat");break;
 default:document.write("Khong la gi ca");
 }
</script>
</body>
</html>
```

[sửa] Lập trình hướng đối tượng trong Javascript

[sửa] Lập trình theo hướng đối tượng là gì?

-Lập trình hướng đối tượng (gọi tắt là OOP, từ chữ Anh ngữ object-oriented programming), hay còn gọi là lập trình định hướng đối tượng, là kĩ thuật lập trình hỗ trợ công nghệ đối tượng. OOP được xem là giúp tăng năng suất, đơn giản hóa độ phức tạp khi bảo trì cũng như mở rộng phần mềm bằng cách cho phép lập trình viên tập trung vào các đối tượng phần mềm ở bậc cao hơn. Ngoài ra, nhiều người còn cho rằng OOP dễ tiếp thu hơn cho những người mới học về lập trình hơn là các phương pháp trước đó.

-Một cách giản lược, đây là khái niệm và là một nỗ lực nhằm giảm nhẹ các thao tác viết mã cho người lập trình, cho phép họ tạo ra các ứng dụng mà các yếu tố bên ngoài có thể tương tác với các chương trình đó giống như là tương tác với các đối tượng vật lý.

-Những đối tượng trong một ngôn ngữ OOP là các kết hợp giữa mã và dữ liệu mà chúng được nhìn nhận như là một đơn vị duy nhất. Mỗi đối tượng có một tên riêng biệt và tất cả các tham chiếu đến đối tượng đó được tiến hành qua tên của nó. Như vậy, mỗi đối tượng có khả năng nhận vào các thông báo, xử lý dữ liệu (bên trong của nó), và gửi ra hay trả lời đến các đối tượng khác hay đến môi trường.

Khi lập trình hướng đối tượng ta cần chú ý 3 điểm chính sau:

- Tính đóng gói (encapsulation) :cung cấp những phương thức gọi đến các đối tượng trong javaScript như là một Class
- Tính đa hình (polymorphism): Thể hiện thông qua việc gửi các thông điệp (message). Việc gửi các thông điệp này có thể so sánh như việc gọi các hàm bên trong của một đối tượng. Các phương thức dùng trả lời cho một thông điệp sẽ tùy theo đối tượng mà thông điệp đó được gửi tới sẽ có phản ứng khác nhau. Người lập trình có thể định nghĩa một đặc tính (chẳng hạn thông qua tên của các phương thức) cho một loạt các đối tượng gần nhau nhưng khi thi hành thì dùng cùng một tên gọi mà sự thi hành của mỗi đối tượng sẽ tự động xảy ra tương ứng theo đặc tính của từng đối tượng mà không bị nhầm lẫn.

Thí dụ khi định nghĩa hai đối tượng "hinh_vuong" và "hinh_tron" thì có một phương thức chung là "chu_vi". Khi gọi phương thức này thì nếu đối tượng là "hinh_vuong" nó sẽ tính theo công thức khác với khi đối tượng là "hinh_tron".

- Tính kế thừa (inheritance): Đặc tính này cho phép một đối tượng có thể có sẵn các đặc tính mà đối tượng khác đã có thông qua kế thừa. Điều này cho phép các đối tượng chia sẻ hay mở rộng các đặc tính sẵn có mà không phải tiến hành định nghĩa lại. Tuy nhiên, không phải ngôn ngữ định hướng đối tượng nào cũng có tính chất này.

[sửa] Lập trình OOP trong Javascript.

- Tạo một Object đơn giản thông qua đối tượng Object đã có sẵn trong Javascript:

```
obj = new Object;
obj.x = 1;
obj.y = 2;
<html>
<head>
     <script language="JavaScript">
        obj =new Object;                          //tạo ra đối tượng
Object
        obj.name="ABC";                           //thêm thuộc tính
name cho object
        obj.address="q10 tphcm";                  //thêm một thuộc tính
nữa
        document.write("kqua : "+obj.name);       //hiển thị name của
object
    </script>
</head>
<body>

</body>
</html>
```

-Chúng ta có thể tạo ra vô số các thuộc tính cho đối tượng Object tại mọi lúc ta cần.Như vậy lúc này obj của chúng ta có 2 thuộc tính name và address ngoài ra nó còn có thêm

thuộc tính là constructor có giá trị là Object do môt hàm function trong javascript thông qua thuộc tính prototype (Object.prototype).Chúng ta sẽ giải thích prototype sau.

- Tạo ra một Class và một constructor cho đối tượng.

-Để tạo ra một class bằng cách dùng một hàm đơn giản.Khi mà hàm được gọi với phép toán new thì Server sẽ tạo ra một class.sau đó javascript sẽ tạo ra một đối tượng và khi đó đối tượng sẽ gọi hàm constructor.Trong constructor biến this trỏ đến object vừa được tạo ra. -VD:

```
function Foo(){
   this.x = 1;
   this.y = 2;
}
obj = new Foo;
```

- Giải thích Prototype

Trong javascript có thể kế thừa các trường từ những đối tượng khác, được gọi là prototype.Khi giá trị được trả về cho một biến thì đầu tiên javascript kiểm tra xem biến đó đã được tạo trong object chưa,nếu chưa thì nó tiếp tục tìm kiếm trong prototype của object vừa được tạo ,sau đó thay đổi giá trị của prototype.
-VD:Nếu ta muốn tạo ra đối tượng x từ hàm constructor B.

```
<html>
<head>
     <script language="JavaScript">
       Object.prototype.inObj = 1;

          function A(){
          this.inA = 2;
          }
           A.prototype.inAProto = 3;  //them mot truong cho A thong qua
prototype

                B.prototype = new A;            // B kế thừa A
                B.prototype.constructor = B;

           function B(){
           this.inB = 4;}
           B.prototype.inBProto = 5;
           x = new B;
           document.write(x.inObj + ', ' + x.inA + ', ' + x.inAProto +
', ' + x.inB + ', ' + x.inBProto);

    </script>
</head>
<body>

</body>
</html>
```

-Kết quả là in ra man hinh la: 1, 2, 3, 4, 5

- Định nghĩa một phương thức và gọi phương thức của một Class.

-Trong javascript cũng cho phép ta tạo ra một số phương thức cho Class.Khi ta gọi obj.Function() thì nó sẽ thực thi hàm của đối tượng đó.
-VD:

```
<html>
<head>
     <script language="JavaScript">
       Object.prototype.inObj = 1;

          function A(){
          this.inA = 2;

          this.addA=function (){//định nghĩa hàm trong constructor(this
tham chiếu đến A)
           var x="Hello";
           return x;
          }

          }
      a=new A;
      document.write("Kqua : "+a.addA());//gọi phương thức addA của A
</script>
<body>

</body>
</html>
```

-Ngoài ra chúng ta còn có thể định nghĩa một hàm trong prototype của constructor. VD:

```
<html>
<head>
     <script language="JavaScript">

          function Foo(){
             this.x = 1;
          }
          Foo.prototype.AddX = function(y){  //định nghĩa ra một phương
thức
          return this.x+=y;
          }

          foo=new Foo;
          var re=foo.AddX(5);

          document.write(re);

    </script>
</head>
<body>
```

```
</body>
</html>
```

- Tính đa hình (Polymorphism):

-Các đối tượng khác nhau cũng có thể có các phương thức cùng tên.Vì thế khi gọi hàm ta nên gọi chính xát tên
-VD:

```
<html>
<head>
     <script language="JavaScript">
      function A(){
          this.x = 1;
          }

          A.prototype.DoIt = function() {
          this.x += 1;
          }

         function B(){
          this.x = 1;
          }
          B.prototype.DoIt = function(){
          this.x += 2;
          }
          a = new A; //ta thấy A và B cùng có biến tên a.Và cùng có
phương thức có tên là DoIt
          b = new B;

          var bien_x_A= a.DoIt();
          var bien_x_B=b.DoIt();
     document.write(bien_x_A + ', ' + bien_x_B);
    </script>
</head>
<body>

</body>
</html>
```

-Tính đa hình thể hiện ở chổ: A và B cùng có biến tên a.Và cùng có phương thức có tên là DoIt

- Định nghĩa một Class con(Sub-Class).

-Sử dụng từ khóa prototype để kế thừa một Super-Class(Lớp cha).Lớp con có thể kế thừa các phương thức cũng như các trường của lớp cha mà chúng không cần khai báo.ngoài ra lớp con còn có thể có thêm các trương khác nữa.
-VD:

```
<html>
```

```
<head>
     <script language="JavaScript">
function A() {                          // Định nghĩa lớp Cha
  this.x = 1;
}

A.prototype.DoIt = function()           // Viết Method cho lớp cha
{    this.x += 1;}

 B.prototype = new A;                   // Định nghĩa sub-class

 B.prototype.constructor = B;function B(){

 A.call(this);                      // gọi constructor của lớp cha
  this.y = 2;}

 B.prototype.DoIt = function() {     // Định nghĩa  Method cho lớp con
B
 A.prototype.DoIt.call(this);    // gọi một phương thức của lớp cha
 this.y += 1;}

 b = new B;
 document.write((b instanceof A) + ', ' + (b instanceof B) + '<BR/>');

 b.DoIt();
 document.write(b.x + ', ' + b.y);
  <body>

</body>
</html>
```

[sửa] Mô mình MVC (Model - View - Controler)

[sửa] Chức năng của Model

-Trong mô hình MVC thì Model gần với Server nhất,Khi View cần thông tin View gởi Controler thì Controler yêu cầu đến Model ,nhiệm vụ của Model là giao dịch với Server để lấy thông tin rồi trả về cho controler.Như vậy Model chịu sự điều khiển của controler

[sửa] Chức năng của View

-View là tầng trên nhất so với controler và Model.View dùng để hiển thị hay tương tác với người dùng.Cũng giống như Model thì View cũng chịu điều khiển của Controler

[sửa] Chức năng của Control

-Control là tầng giữa trong MVC ,Control có chức năng là điều khiển Model và View hay là cầu nối giữa View - Model.

[sửa] Ứng dụng của mô hình MVC trong MyWorkplace

-Mô tả MVC trong workplace.Workplace quản lý 4 Đối tượng
+Đối tượng Workplace quản lý nhiều đối tượng Tabber.
+Một đối tượng Tabber quản lý nhiều đối tượng Tab.
+Một đối tượng Tab có một đối tượng TabGrid.
+Một đối tượng TabGrid có nhiều đối tượng Gadget.
-Trong Workplace thì View và Control là một.Workplace có Model là Workplace_Model dùng để tương tác với Server để lưu trữ hay lấy dữ liệu.
-Một đối tượng Tabber có Tabber_Model
-Một đối tượng Tab có Tab_Model
-Một đối tượng TabGrid có TabGrid_Model
-Một đối tượng Gadget có Gadget_Model

[sửa] Lập trình sự kiện (Observerble)

-Lập trình sự kiện bao gồm Đăng ký sự kiện và thông báo sự kiện.

[sửa] Đăng ký sự kiện

```
objectModel.registerObserver(observer,"even_name")->observer đăng ký sự kiện even_name với objectModel
```

objectModel : Model của đối tượng.
observer :thằng quan sát(tức là sẽ nhận được thông báo khi sự kiện even_name của objectModel xảy ra).

[sửa] Thông báo sự kiện

```
objectModel.notifyoObserver("even_name",msg)
->objectModel thông báo cho đối tượng đăng ký sự kiện even_name với một msg(thông báo)
```

[sửa] Sự thay đổi của con trỏ 'this' trong phương thức của 1 class

-Có nghĩa là 'this' lúc này không còn trỏ đến Class nữa.

[sửa] Lý do con trỏ 'this' thay đổi

-Để hiểu được vấn đề ta xem một vd nhỏ.
-Vd: -Ta có Class A:

```
Class A=function(){
    this.name="Nam";

    this.b=function a(){
     return this.name;//this trỏ đến Class A
    }
  }
```

-Và một button

```
var AInts=new A;
var btElement=document.getElementById("idok");
btElement.onclick=AInts.b;
```

-Lúc này : btElement.onclick=AInts.b=this.name. Nhưng 'this' ở đây không còn trỏ đến A nữa. Mà 'this' đã trỏ sang btElement do:

- Khi btElement.onclick=registerBt được gọi thì btElement copy phương thức this.b=function a() của Class A.Nhưng lúc này phương thức này lại là của chính nó.Nên 'this' đã thay đổi.

[sửa] Cách khắc phục

- Ta tạo ra một biến và gán giá trị là 'this'.Khi ấy 'this' sẽ không thay đổi.

-VD:

```
var seft=this;
```

- Hoặc Chúng ta sử dụng thuộc tính prototype

-VD:

```
this.a=this.a.bind(this); //khi này thì giá trị của 'this' cũng không thay đổi
```

[sửa] Thiết kế giao diện 1 ứng dụng web

[sửa] Tạo nên các tags html bằng javascript (Elements) và định dạng chúng bằng CSS

-Ta biết các Tags trong html cũng có tên,thuộc tính..Hay nói khác đi chúng cũng là đối tượng Element trong javascript.Vì vậy chúng ta cũng có thể tạo các Tags bằng javascript.Vd trong javascript bạn muốn tạo ra Table :

```
1. var tble = document.createElement("table");
2. table.style["width"] = "100%";
3. table.id="tb1";

4. var row_1 = document.createElement("tr");
5. row_1.style["height"] = "30px";

6. var cell_1 = document.createElement("td");
7. cell_1.style["width"] = "25%";

8. var cell_2 = document.createElement("td");
9. cell_2.style["width"] = "75%";

10.row_1.appendChild(cell_1);
11.row_1.appendChild(cell_2);
12.tble.appendChild(row_1);
```

-Dòng (1):document sẽ tạo ra một table thông qua hàm **createElement**("table")->nếu muốn tạo tag gì thì ta khai báo trong hàm đó tên của tag đó.Vd :**document.createElement**("a");->tạo ra tag a.Hàm createElement chỉ tạo ra một đối tượng rỗng.
-Dòng (2)(3):ta thêm thuộc tính cho đối tượng table cùng với giá trị của các thuộc tính.
-Dòng (4)->(9) Ta tạo ra các đối tượng hàng và cột ,các thuộc tính cho chúng.
-Dòng (10),(11):Ta gọi hàm appendChile để nối các cột vào hàng.Dòng (12) ta nối hàng vào table.Vậy là ta có một table.

-Khi một Element đã tồn tại rồi ta có thể lấy được Element thông qua hàm getElementById("id").Với id đã được khai báo.
Vd:

```
var table1=document.getElementById("idtable");
```

[sửa] Đăng ký sự kiện cho các Elements

-Đang ký sự kiện cho các Element là khi có một sự kiện tạo ra trên Element đó thì Element phải thực hiện một việc.Khi các Element được tạo ra thì nó nguyên bản chưa có sự kiện.Vd như khi tạo ra Element là Button "OK" thì khi một sự kiên Click xảy ra nó thực hiện hành động là đóng cửa sổ Window chẳn hạn. vd:

```
var bton= document.getElementById("id_bton");
btnAddNhanVien.onclick = handleAddButton; //đăng ký sự kiện

function handleAddButton () {
    document.write("Hello");
};
```

[sửa] VD

-Có một CÔng ty muốn thêm Nhân viên vào Danh sách Nhân viên của cong ty đó.
+Trước hết ta thấy có 2 Đối tượng là Công ty, Nhân viên.Công ty chứa nhiều Nhân Viên.
-Giao Diện hiển thị ra danh sách nhân viên ,khi thêm mới thì nhân viên đó phải nằm trong Ds Nhân viên của Công ty.

1)Tạo ra file nhanvien.js như sau:file này định nghĩa ra đối tượng Nhân Viên gồm có tên ,địa chỉ và có một id.

```
function NhanVien (id, ten, diaChi) {
    this.id = id;
    this.ten = ten;
    this.diaChi = diaChi;

    this.getId = function () {
        return this.id;
    };

    this.getTen = function () {
        return this.ten;
    };

    this.setTen = function (value) {
        this.ten = value;
    };

    this.getDiaChi = function () {
        return this.diaChi;
    };

    this.setDiaChi = function (value) {
        this.diaChi = value;
    };
}
```

2)Tạo một index.html là giao diện hiển thị danh sách nhân viên của công ty.

```
<html>
<head>
    <LINK href="default.css" type="text/css" rel="stylesheet">
    <script language="JavaScript" src="nhanvien.js"></script> //chỉ đến
file javascript
    <script language="JavaScript" src="congty.js"></script> //chỉ đến
file javascript
    <script language="JavaScript" src="addform.js"></script> //chỉ đến
file javascript
    <script language="JavaScript">
   function init () {
    var divContainer = document.getElementById("dsNhanVienContainer");
    CTyInstance = new CongTy(divContainer);
```

```
    var btnAddNhanVien =
document.getElementById("btnAddNhanVien");//lấy Element Button Add nhân
viên
    btnAddNhanVien.onclick = handleAddButton;//đang ký sự kiện cho
Element Button Add nhân viên
     };

        function handleAddButton () {
            var AddFormInstance = new AddForm(CTyInstance,
CTyInstance.addNhanVien);
            AddFormInstance.show();
        };


        window.onload = init;
    </script>
</head>
<body>
<center>
    <h3>--- oOo ---</h3>
    <div id="dsNhanVienContainer">
    <table border='1' cellpadding="0" cellspacing="0"
style="width:60%">
    <thead>
        <th style="width:10%">   </th>
        <th style="width:10%"> STT </th>
        <th style="width:40%"> Ho & Ten </th>
        <th style="width:40%"> Dia Chi </th>
    </thead>
    </table>
    </div>
    <br>
    <br>
    <input type="button" id="btnAddNhanVien" value="Add" />

</center>
</body>
</html
```

3)Tạo file congty.js

```
function CongTy (divContainer) {
    this.dsNhanVien = new Array();
    this.divContainer = divContainer;

    this.getDSNhanVien = function () {
        return this.dsNhanVien;
    };

    this.setDSNhanVien = function (dsNhanVien) {
    };

    this.addNhanVien = function (ten, diaChi) {
        var max = 0;
        for (var i=0; i<this.dsNhanVien.length; i++)
```

```
            if (this.dsNhanVien[i].getId()>max) max =
this.dsNhanVien[i].getId();
        var nhanVienObj = new NhanVien(max + 1, ten, diaChi);
        this.dsNhanVien.push(nhanVienObj);
        console.log(this.dsNhanVien);
        this.showDSNhanVien();
    };



    this.showDSNhanVien = function () {
        console.log(this.divContainer);
        var htmlStr  =  '<table border="1" cellpadding="0"
cellspacing="0" style="width:60%">\
                            <thead>\
                                <th style="width:10%">   </th>\
                                <th style="width:10%"> STT </th>\
                                <th style="width:40%"> Ho & Ten </th>\
                                <th style="width:40%"> Dia Chi </th>\
                            </thead>\
                            <tbody>'
        for (var i=0; i<this.dsNhanVien.length; i++ ) {
            htmlStr += '<tr>\
                            <td><input type="checkbox"
id="'+this.dsNhanVien[i].getId()+'"></td>\
                            <td>'+i+'</td>\
                            <td> '+this.dsNhanVien[i].getTen()+'</td>\
                            <td>
'+this.dsNhanVien[i].getDiaChi()+'</td>\
                        </tr>'
        }
        htmlStr += '</tbody>'
        htmlStr += '</table>'
        this.divContainer.innerHTML = htmlStr;
    };
}
```

4)Ta tạo file addform.js

```
function AddForm (observer, callBackFunction) {
    var self = this;
    this.observer = observer;
    this.observer.callBackFunction = callBackFunction;

    this.show = function () {
        var formBody =  document.createElement("div");
        formBody.id = "divAddForm";
        formBody.setAttribute("class", "addFrom");
        formBody.style.top = (document.body.clientHeight - 100)/2 ;
        formBody.style.left = (document.body.clientWidth - 300)/2 ;

        var labelInputTen = document.createElement("label");
        labelInputTen.innerHTML = "Ho & Ten: "
        var inputTen = document.createElement("input");
        inputTen.id = "inputTen";
```

```
inputTen.type='text';
inputTen.style["width"] = "100%";

var labelInputDiaChi = document.createElement("label");
labelInputDiaChi.innerHTML = "Dia Chi: "
var inputDiaChi = document.createElement("input");
inputDiaChi.id = "inputDiaChi";
inputDiaChi.type='text';
inputDiaChi.style["width"] = "100%";

var table = document.createElement("table");
table.style["width"] = "100%";
// create row 1
var row_1 = document.createElement("tr");
row_1.style["height"] = "30px";
var cell_1 = document.createElement("td");
cell_1.style["width"] = "25%";
var cell_2 = document.createElement("td");
cell_2.style["width"] = "75%";
cell_1.appendChild(labelInputTen);
cell_2.appendChild(inputTen);
row_1.appendChild(cell_1);
row_1.appendChild(cell_2);
table.appendChild(row_1);

// create row 2
var row_2 = document.createElement("tr");
row_2.style["height"] = "30px";
var cell_1 = document.createElement("td");
cell_1.style["width"] = "25%";
var cell_2 = document.createElement("td");
cell_2.style["width"] = "75%";
cell_1.appendChild(labelInputDiaChi);
cell_2.appendChild(inputDiaChi);
row_2.appendChild(cell_1);
row_2.appendChild(cell_2);
table.appendChild(row_2);

var btnAddFormOk = document.createElement("input");
btnAddFormOk.id = "btnAddFormOk";
btnAddFormOk.type='button';
btnAddFormOk.value='OK';

var btnAddFormCancel = document.createElement("input");
btnAddFormCancel.id = "btnAddFormCancel";
btnAddFormCancel.type='button';
btnAddFormCancel.value='Cancel';

formBody.appendChild(table);
formBody.appendChild(btnAddFormOk);
formBody.appendChild(btnAddFormCancel);

var dlgBackGround=document.createElement('div');
dlgBackGround.setAttribute('id','dlgBackGround');
dlgBackGround.setAttribute('class','dlgBackGround');
dlgBackGround.style.height = document.body.clientHeight + "px";
dlgBackGround.appendChild(formBody);
```

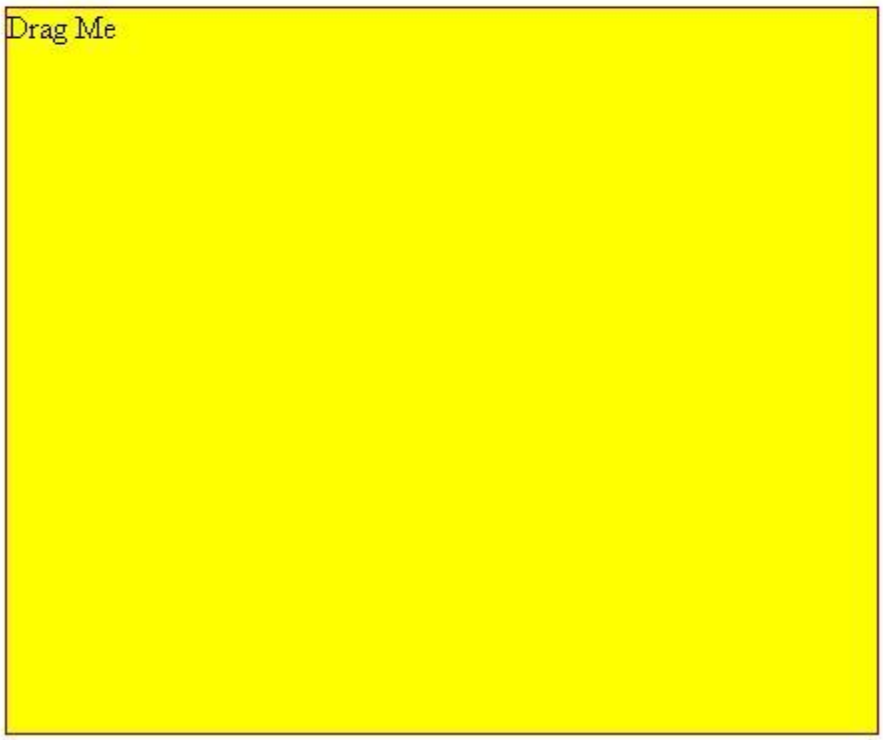
```
        var htmlBody=document.getElementsByTagName('body')[0];
        htmlBody.appendChild(dlgBackGround);
        self.registerEvents();
    };

    this.registerEvents = function () {
        var btnAddFormOk = document.getElementById("btnAddFormOk");
        btnAddFormOk.onclick = self.handleOkButton;
        var btnAddFormCancel =
document.getElementById("btnAddFormCancel");
        btnAddFormCancel.onclick = self.close;
    };

    this.handleOkButton = function () {
        var tenElement = document.getElementById("inputTen");
        var diaChiElement = document.getElementById("inputDiaChi");
        var ten = tenElement.value;
        var diaChi = diaChiElement.value;
        self.observer.callBackFunction(ten, diaChi);
        self.close();
    };

    this.close = function () {
        var addForm = document.getElementById("dlgBackGround");
        var htmlBody=document.getElementsByTagName('body')[0];
        htmlBody.removeChild(addForm);
    };

}
```

[sửa] Kéo và thả (drag and drop) các Elements

[sửa] Drag cơ bản

-Drag là một hành động di chuyển một Element trên màn hình khi chúng ta click chuột vào Element đó và kéo chúng đến một vị trí khác.Để làm việc này Chúng ta cần phải tạo ra một Element,việc tiếp theo là chúng ta phải xây dựng cho Element đó các sự kiện.Có 3 sự kiện cơ bản là onmousedown (bắt sự kiện Drag),onmousemove (đang Drag) và onmouseup(kết thúc Drag).Và để cho Element đó di chuyển thì thuộc tính position phải có giá trị là:absolute, fixed, relative.

-VD:Xét một ví dụ.Ta có một tag trên màn hình



-Ta sẽ di chuyển nó.
+B1)Đầu tiên ta tạo ra một CSS như sau:

```
<style type="text/css">
<!--
.vidFrame {
            position: absolute ;
            background-color: yellow;
            border: 1px solid #800000;
            width: 435px;
            height: 362px;
            cursor: move;
          }
-->
```

```
</style>
```

+B2)Tiếp theo ta tạo ra một Element có Class là CSS vừa tạo.

```
<div id="vidPane" class='vidFrame'>Drag Me</div>
```

-Ta thấy tag vừa tạo ra có class='vidFrame' là CSS ở B1.
+B3)Ta set giá trị ban đầu cho tag

```
<script language="JavaScript" type="text/javascript">
<!--
   var savedTarget=null;
   var orgCursor=null;
   var dragOK=false;     // true nếu ta move nó
   var dragXoffset=0;    // Chúng ta sẽ di chuyển theo tọa độ X là bao
nhiêu
   var dragYoffset=0;    // Chúng ta sẽ di chuyển theo tọa độ Y là bao
nhiêu
   vidPaneID = document.getElementById('vidPane'); // Our movable layer
  vidPaneID.style.top='75px';                      // xuất hiên trên màn
hình với tọa độ
   vidPaneID.style.left='75px';
//-->
</script>
```

+B4)Ta sẽ gọi sự kiện bắt đầu Drag.

```
document.onmousedown=dragHandler;
```

+B5)Ta viết hàm sự kiện dragHandler

```
function dragHandler(e){
      var htype='-moz-grabbing';
      if (e == null) {
                e = window.event;
                htype='move';
                }
      var target = e.target != null ? e.target : e.srcElement;
      orgCursor=target.style.cursor;
      if (target.className=="vidFrame") {
         savedTarget=target;
         target.style.cursor=htype;
         dragOK=true;
         dragXoffset=e.clientX-parseInt(vidPaneID.style.left);
         dragYoffset=e.clientY-parseInt(vidPaneID.style.top);
         document.onmousemove=moveHandler;
         document.onmouseup=cleanup;
         return false;
      }
   }
```

-Ta thấy trong dragHandler ta nhận vào biến ta sự kiện và trong dragHandler lại có:

```
document.onmousemove=moveHandler; //dùng để gọi di chuyển nó.
function moveHandler(e){

      if (e == null) {
                e = window.event ;
                }
      if (e.button<=1&&dragOK){
         savedTarget.style.left=e.clientX-dragXoffset+'px';
         savedTarget.style.top=e.clientY-dragYoffset+'px';
         return false;
      }
   }
```

-Ta lại có:

```
document.onmouseup=cleanup;
function cleanup(e) {
      document.onmousemove=null;
      document.onmouseup=null;
      savedTarget.style.cursor=orgCursor;
      dragOK=false;
   }
```

B6)Cuối cùng ta có file demo.html như sau

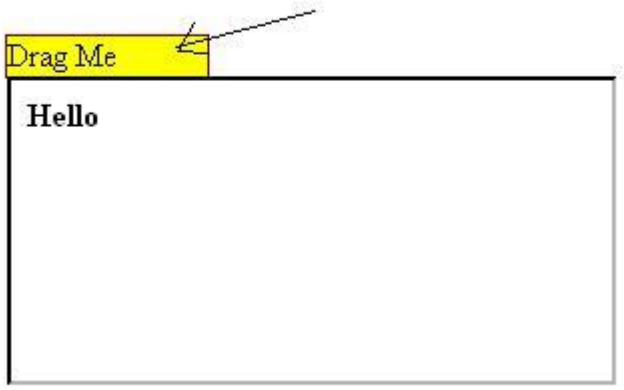
```
<html>
<head>
<title>target example</title>
<style type="text/css">
<!--
.vidFrame {
            position: absolute ;
            background-color: yellow;
            border: 1px solid #800000;
        width: 435px;
            height: 362px;
            cursor: move;
          }
-->
</style>
<script language="JavaScript" type="text/javascript">
<!--

   function moveHandler(e){

      if (e == null) {
                e = window.event ;
                }
      if (e.button<=1&&dragOK){
         savedTarget.style.left=e.clientX-dragXoffset+'px';
         savedTarget.style.top=e.clientY-dragYoffset+'px';
         return false;
      }
```

```
   }

   function cleanup(e) {
      document.onmousemove=null;
      document.onmouseup=null;
      savedTarget.style.cursor=orgCursor;
      dragOK=false;
   }

   function dragHandler(e){
      var htype='-moz-grabbing';
      if (e == null) {
                e = window.event;
                htype='move';
                }
      var target = e.target != null ? e.target : e.srcElement;
      orgCursor=target.style.cursor;
      if (target.className=="vidFrame") {
         savedTarget=target;
         target.style.cursor=htype;
         dragOK=true;
         dragXoffset=e.clientX-parseInt(vidPaneID.style.left);
                 console.log("xxxxx"+e.clientX);
         dragYoffset=e.clientY-parseInt(vidPaneID.style.top);
                 console.log("yyyyyy"+e.clientY);
         document.onmousemove=moveHandler;
         document.onmouseup=cleanup;
         return false;
      }
   }

         document.onmousedown=dragHandler;
//-->
</script>
</head>

<body>
<div id="vidPane" class='vidFrame'>Drag Me</div>

<script language="JavaScript" type="text/javascript">
<!--
   var savedTarget=null;
   var orgCursor=null;
   var dragOK=false;
   var dragXoffset=0;
  var dragYoffset=0;
   vidPaneID = document.getElementById('vidPane');
  vidPaneID.style.top='75px';
   vidPaneID.style.left='75px';
//-->
</script>

</body>
</html>
```

[sửa] Drag khi mouse đi qua iframe

[sửa] Vấn đề

-Khi mouse đi qua iframe nằm trong document thì Drag không còn hiệu lực nữa ,nguyên nhân là do sự kiện mà ta đăng ký trên document không có hiệu lực trong iframe



khi mà ta click chuột vào vùng màu vàng kéo đi thì làm tốt nhưng khi chuột rơi vào phần dưới thì không còn kéo đi được nữa.

[sửa] Cách khắc phục

-Cách khắc phục ta tạo ra một thẻ lớn, trong suốt che hết màn hình khi đó ta vẫn thấy được iframe mà vẫn bắt được sự kiện đó.
-VD:

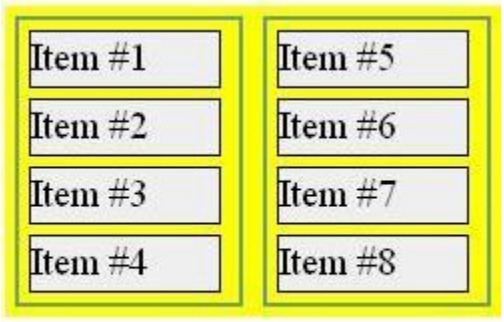
```
<html>
<head>
<title>target example</title>
<style type="text/css">
<!--
.vidFrame {
            position: absolute ;
            background-color: yellow;
            border: 1px solid #800000;
            width: 100px;
            height: 20px;
            cursor: move;
         }
 -->


</style>
<script language="JavaScript" type="text/javascript">

function init(){
        var savedTarget=null;
```

```
        var orgCursor=null;
        var dragOK=false;
        var dragXoffset=0;
        var dragYoffset=0;
        vidPaneID = document.getElementById('vidPane');
        vidPaneID.style.top='100px';
        vidPaneID.style.left='100px';
}


   function moveHandler(e){

      if (e == null) {
                e = window.event ;
                }
      if (e.button<=1&&dragOK){
         savedTarget.style.left=e.clientX-dragXoffset+'px';
         savedTarget.style.top=e.clientY-dragYoffset+'px';
         return false;
      }
   }

   function cleanup(e) {

      document.onmousemove=null;
      document.onmouseup=null;
      savedTarget.style.cursor=orgCursor;
      dragOK=false;
          //var divDocument=document.getElementById("idDiv");
                //document.removeChild(divDocument);
   }

   function dragHandler(e){

        var htype='-moz-grabbing';
      if (e == null) {
                e = window.event;
                htype='move';
                }
      var target = e.target != null ? e.target : e.srcElement;
      orgCursor=target.style.cursor;
      if (target.className=="vidFrame") {
         savedTarget=target;
         target.style.cursor=htype;
         dragOK=true;
         dragXoffset=e.clientX-parseInt(vidPaneID.style.left);
         dragYoffset=e.clientY-parseInt(vidPaneID.style.top);
                //Khac phuc:----------tao ra <div>  de bat su kien.Tag
<div> nay an ben duoi
                var divDocument=document.createElement("div");
                divDocument.id="idDiv";
                divDocument.style["width"]="1000px";
                divDocument.style["height"]="1000px";
                divDocument.style["zIndex"]=100000;
                divDocument.style["position"]="absolute";
                var htmlBody=document.getElementsByTagName('body')[0];
                htmlBody.appendChild(divDocument);
```

```
                //--------------------
         document.onmousemove=moveHandler;
         document.onmouseup=cleanup;

         return false;
      }
   }

         document.onmousedown=dragHandler;


</script>
</head>

<body onload="init()">
        <div id="vidPane" class='vidFrame'>
        Drag Me
        <iframe id="Idiframe"src="new5.html"></iframe>
</div>


</body>
</html>
```

[sửa] Drag and drop

-Nếu phần trên ta vừa nghiên cứu Drag theo những vị trị thoải mái thì Drag and Drog là Chúng ta Drag trong những vị trí nhất định trong một Grid(lưới) tức là khi ta thả chuột ra thì Element sẽ nằm trong Grid.
Vd:



-Từ hình ta thấy Drag Drog chỉ xảy ra trong phạm vi màu vàng.Hay chính xác là Các item chỉ có thể Drag được trong phạm vi là 2 cột(item có thể lên-xuống,qua-lại)
-B1)Ta tạora CSS

```
<style>
<!--
.DragContainer {
        border-right: #669999 2px solid;
        padding-right: 5px;
        border-top: #669999 2px solid;
```

```
        padding-left: 5px;
        float: left;
        padding-bottom: 0px;
        margin: 5px;
        border-left: #669999 2px solid;
        width: 100px;
        padding-top: 5px;
        border-bottom: #669999 2px solid
}

.DragBox {
        background-color: #ffff99;
        border-right: #000 1px solid;
        border-top: #000 1px solid;
        font-size: 20px;
        margin-bottom: 5px;
        padding-bottom: 2px;
        border-left: #000 1px solid;
        width: 94px;
        cursor: pointer;
        padding-top: 2px;
        border-bottom: #000 1px solid;
        background-color: #eee
}
-->
</style>
```

-B2)Ta tạo ra các thẻ div

```
<div id="did" style="background-color:yellow;z-index=10000;position:
absolute;">
                <div class="DragContainer" id="DragContainer1">
                <div class="DragBox" id="Item1">Item #1</div>
                <div class="DragBox" id="Item2">Item #2</div>
                <div class="DragBox" id="Item3">Item #3</div>
                <div class="DragBox" id="Item4">Item #4</div>
                </div>

                <div class="DragContainer" id="DragContainer2">
                <div class="DragBox" id="Item5">Item #5</div>
                <div class="DragBox" id="Item6">Item #6</div>
                <div class="DragBox" id="Item7">Item #7</div>
                <div class="DragBox" id="Item8">Item #8</div>
                </div>
</div>
```

-Tiếp theo là đoạn code sau,trước tiên ta làm cho các thẻ div có thể chuyển động tự do trong màn hình đã:file (index.html)

```
<html>
<head>
<title>DEMO34</title>

<style>
```

```
<!--
.DragContainer {
        border-right: #669999 2px solid;
        padding-right: 5px;
        border-top: #669999 2px solid;
        padding-left: 5px;
        float: left;
        padding-bottom: 0px;
        margin: 5px;
        border-left: #669999 2px solid;
        width: 100px;
        padding-top: 5px;
        border-bottom: #669999 2px solid
}

.DragBox {
        background-color: #ffff99;
        border-right: #000 1px solid;
        border-top: #000 1px solid;
        font-size: 20px;
        margin-bottom: 5px;
        padding-bottom: 2px;
        border-left: #000 1px solid;
        width: 94px;
        cursor: pointer;
        padding-top: 2px;
        border-bottom: #000 1px solid;
        background-color: #eee
}
-->
</style>


<script language="JavaScript" src="dragDrop2.js"></script>

<script language="JavaScript" type="text/javascript">
        function init(){
                        var
divContainer=document.getElementById("divContainer");
                        divContainer.style.top="100px";
                        divContainer.style.left="100px";
                        var dragDrop=new DragDrop(divContainer);
                        divContainer.onmousedown=dragDrop.dragHandler;

        }

        window.onload=init;
</script>
</head>

<body>
        <div id="divContainer" style="background-color:yellow;z-
index=0;position: absolute;">
                <div class="DragContainer" id="DragContainer1">
                <div class="DragBox" id="Item1">Item #1</div>
                <div class="DragBox" id="Item2">Item #2</div>
                <div class="DragBox" id="Item3">Item #3</div>
```

```
                <div class="DragBox" id="Item4">Item #4</div>
                </div>

                <div class="DragContainer" id="DragContainer2">
                <div class="DragBox" id="Item5">Item #5</div>
                <div class="DragBox" id="Item6">Item #6</div>
                <div class="DragBox" id="Item7">Item #7</div>
                <div class="DragBox" id="Item8">Item #8</div>
                </div>
        </div>
</body>
</html>
```

-Tiếp theo ta tạo ra đối tượng DragDrop và lưu với tên file là: (drapDrop2.js)

```
function DragDrop (divContainer){
                this.divContainer=divContainer;
                //
                this.savedTarget=null;
                this.orgCursor=null;
                this.dragOK=false;
                this.dragXoffset=0;
                this.dragYoffset=0;
                var self=this;
                //

        this.moveHandler=function(e){
                try{
                        console.log("---------");
                        if (e == null){
                        e = window.event ;
                        }
                        console.log("-------222");
                        if (e.button<=1&& self.dragOK){
                 self.savedTarget.style.left=e.clientX-
self.dragXoffset+'px';
                 self.savedTarget.style.top=e.clientY-
self.dragYoffset+'px';
                         console.log("-------333");
                 return false;
                        }
          }catch(e){console.log(e.toString())};
   }
////////cleanup
   this.cleanup=function(e) {
                        document.onmousemove=null;
                   document.onmouseup=null;
                   self.savedTarget.style.cursor=self.orgCursor;
                   self.dragOK=false;
                        var
htmlBody=document.getElementsByTagName('body')[0];
                        var div2=document.getElementById("idDiv");
                        htmlBody.removeChild(div2);
         }
        //dragHandler
this.dragHandler=function(e){
```

```
                console.log("+++++++++");
                var htype='-moz-grabbing';
                        if (e == null) {
                        e = window.event;
                        htype='move';
                }
                var target = e.target != null ? e.target :
e.srcElement;
                self.orgCursor=target.style.cursor;

         if (target.className=="DragBox"){
            self.savedTarget=target;
        target.style.cursor=htype;
                target.style["position"]="absolute";
        self.dragOK=true;

                console.log("xxxxxxxx"+e.clientX);

console.log("yyyyyyyyy"+parseInt(divContainer.style.left));
        self.dragXoffset=e.clientX-parseInt(divContainer.style.left);
                self.dragYoffset=e.clientY-
parseInt(divContainer.style.top);
                 //----------tao ra <div>  de bat su kien.Tag <div> nay
an ben duoi
                var divDocument=document.createElement("div");
                divDocument.id="idDiv";
                divDocument.style["width"]="1000px";
                divDocument.style["height"]="1000px";
                divDocument.style["zIndex"]=10000;
                divDocument.style["position"]="absolute";
                //divDocument.style["backgroundColor"]="yellow";
                var htmlBody=document.getElementsByTagName('body')[0];
                htmlBody.appendChild(divDocument);
           //--------
        document.onmousemove=self.moveHandler;
        document.onmouseup=self.cleanup;

        return false;
      }
   }

}
```

-Tiếp theo ta sẽ đi xác định lại tọa độ để các div nhỏ chỉ có thể di chuyển trong hai thẻ div lớn hơn trong vùng màu vàng.

[sửa] Ví dụ:

[sửa] Cách vẽ và đăng ký sự kiện cho tabs trong MyWorkplace

[sửa] Cách vẽ và đăng ký sự kiện cho gadgets trong MyWorkplace

[sửa] Drag gadgets

[sửa] Grid của MyWorkplace

[sửa] Ráp 3 thành phần tabs, grid, gadgets lại với nhau

[sửa] Drag and Drop gadgets

- [ ] VIQR*
- [ ] Tắt [F12]

- [x] Bỏ dấu kiểu cũ [F7]
- [x] Đúng chính tả [F8]

Thanh công cụ

- Các liên kết đến đây
- Thay đổi liên quan
- Các trang đặc biệt
- Bản để in
- Liên kết thường trực