

Xử lý sự kiện trong jQuery

Chúng ta có khả năng tạo các trang web động bởi sử dụng các Sự kiện (Event). Các sự kiện là các hành động mà có thể được phát hiện bởi ứng dụng web của bạn.

Sau đây ví dụ một số sự kiện:

- Nhấp chuột
- Tải trang web
- Di chuyển chuột qua một phần tử
- Đệ trình một HTML Form
- Thao tác nhấn phím trên bàn phím
- etc.

Khi các sự kiện này được kích hoạt, bạn có thể sử dụng các hàm custom để phản hồi bất kỳ những gì bạn muốn với sự kiện đó. Những hàm custom này gọi là Event Handler.

Bind các Event Handler trong jQuery

Sử dụng Event Model trong jQuery, chúng ta có thể thiết lập các Event Handler trên các phần tử DOM với phương thức **bind()** như sau:

```
<html>    <head>       <title>The jQuery Example</title>       <script
type="text/javascript"
src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js"></script>
                        <script type="text/javascript" language="javascript">
$(document).ready(function() {             $('div').bind('click', function(
event ){                alert('Hi there!');             });          });
</script>                        <style>          .div{
margin:10px;padding:12px; border:2px solid #666; width:60px;}       </style>
style="background-color:green;">TWO</div>       <div class="div"
style="background-color:red;">THREE</div>             </body>       </html>
```

Code trên sẽ làm cho phần tử div phản hồi lại sự kiện click; khi người sử dụng nhấp chuột bên trong phần tử div này, ngay sau đó, thông báo sẽ được hiển thị.

Nó sẽ cho kết quả sau:

Click on any square below to see the result:

ONE

TWO
THREE

Cú pháp đầy đủ của lệnh **bind()** trong jQuery như sau:

```
selector.bind( eventType[, eventData], handler)
```

Tiếp theo, chúng tôi diễn tả chi tiết các tham số:

- **eventType** − Một chuỗi chứa một loại JavaScript event, như click hoặc đệ trình. Bạn theo dõi phần dưới để thấy danh sách đầy đủ các loại sự kiện.
- **eventData** − tham số tùy ý là một **map** của dữ liệu mà sẽ được truyền tới Event Handler.
- **handler** − Một hàm để thực thi mỗi khi sự kiện được kích hoạt.

Gỡ bỏ Event Handler trong jQuery

Một nét đặc trưng là, mỗi khi một Event Handler được thành lập, nó vẫn còn hiệu quả trong phần sống còn lại của trang web. Có một sự cần thiết khi bạn muốn gỡ bỏ Event Handler này.

jQuery cung cấp lệnh **unbind()** để gỡ bỏ một Event Handler đang tồn tại. Cú pháp của unbind() trong jQuery như sau:

```
selector.unbind(eventType, handler)  or   selector.unbind(eventType)
```

Chi tiết về tham số:

- **eventType** − Một chuỗi chứa một loại JavaScript Event, như click hoặc đệ trình. Bạn theo dõi phần dưới để thấy danh sách đầy đủ các loại sự kiện.
- **handler** − Nếu được cung cấp, nó nhận diện Event Handler cụ thể cần gỡ bỏ.

Các loại sự kiện trong jQuery

Bạn có thể kết nối (bind) các sự kiện sau bởi sử dụng jQuery:

STT	Loại Event & Miêu tả

1	**blur** Xuất hiện khi phần tử mất trọng tâm
2	**change** Xuất hiện khi phần tử thay đổi
3	**click** Xuất hiện khi click chuột
4	**dblclick** Kích hoạt khi nhấp đúp chuột
5	**error** Xuất hiện khi có một lỗi trong quá trình tải
6	**focus** Xuất hiện khi phần tử nhận trọng tâm
7	**keydown** Xuất hiện khi phím được nhấn
8	**keypress** Kích hoạt khi phím được nhấn và thả ra
9	**keyup** Kích hoạt khi phím được thả ra
10	**load** Xuất hiện khi tài liệu được tải
11	**mousedown** Xuất hiện khi nút chuột được nhấn
12	**mouseenter**

	Xuất hiện khi chuột di chuyển vào trong khu vực một phần tử
13	**mouseleave** Xuất hiện khi chuột di chuyển ra khỏi khu vực một phần tử
14	**mousemove** Kích hoạt khi con trỏ chuột di chuyển
15	**mouseout** Kích hoạt khi con trỏ chuột di chuyển ra khỏi một phần tử
16	**mouseover** Kích hoạt khi con trỏ chuột di chuyển qua một phần tử
17	**mouseup** Xuất hiện khi một nút chuột được nhả ra
18	**resize** Kích hoạt khi kích cỡ cửa sổ thay đổi
19	**scroll** Kích hoạt khi cửa sổ được cuốn
20	**select** Kích hoạt khi một text được chọn
21	**submit** Kích hoạt khi một form được đệ trình
22	**unload** Kích hoạt khi tài liệu không được tải

Đối tượng Event trong jQuery

Hàm callback nhận một tham số đơn; khi một Handler được gọi, đối tượng JavaScript Event sẽ được truyền qua nó.

Đối tượng Event thường không cần thiết và tham số được bỏ qua, khi context là thường có sẵn khi Handler được kết nối để biết chính xác những gì cần được thực hiện khi Handler được kích hoạt; tuy nhiên có một số thuộc tính nào đó mà bạn cần truy xuất.

Các thuộc tính của đối tượng Event trong jQuery

Các thuộc tính của Event là có sẵn và an toàn để truy cập theo một phương thức độc lập:

STT	Thuộc tính & Miêu tả
1	**altKey** Thiết lập là true nếu phím Alt được nhấn khi sự kiện được kích hoạt, nếu không là false. Phím Alt được gán nhãn là Option trên hầu hết bàn phím Mac
2	**ctrlKey** Thiết lập là true nếu phím Ctrl được nhấn khi sự kiện được kích hoạt, nếu không là false
3	**data** Giá trị, nếu bất kỳ, được truyền như là tham số thứ hai tới lệnh bind() khi Handler được thành lập.
4	**keyCode** Cho sự kiện các phím di chuyển lên, xuống, điều này trả về phím mà được nhấn
5	**metaKey** Thiết lập là true nếu phím Meta được nhấn khi sự kiện được kích hoạt, nếu không là false. Phím Meta là phím Ctrl trên các PC và phím Command trên Macs
6	**pageX** Cho các sự kiện liên quan tới chuột, xác định tọa độ ngang của sự kiện trong mỗi

	quan hệ với trang ban đầu.
7	**pageY** Cho các sự kiện liên quan tới chuột, xác định tọa độ dọc của sự kiện trong mỗi quan hệ với trang ban đầu
8	**relatedTarget** Với một số sự kiện liên quan đến chuột, nhận diện phần tử mà con trỏ chuột rời khỏi hoặc đi vào khi sự kiện được kích hoạt.
9	**screenX** Với một số sự kiện liên quan đến chuột, xác định tọa độ ngang của sự kiện trong mối quan hệ với màn hình ban đầu
10	**screenY** Với một số sự kiện liên quan đến chuột, xác định tọa độ dọc của sự kiện trong mối quan hệ với màn hình ban đầu
11	**shiftKey** Thiết lập là true nếu phím Shift được nhấn khi sự kiện được kích hoạt, nếu không là false
12	**target** Nhận diện phần tử mà với nó sự kiện được kích hoạt
13	**timeStamp** timestamp (giá trị mili giây) khi sự kiện được tạo ra
14	**type** Với tất cả sự kiện, xác định loại sự kiện mà được kích hoạt
15	**which** Với các sự kiện liên quan tới bàn phím, xác định code giá trị số cho phím mà gây ra sự kiện, và với các sự kiện liên quan tới chuột, xác định nút nào được nhấn (1 cho nút

	trái, 2 cho ở giữa và 3 cho nút phải)

Ví dụ

```
<html>    <head>       <title>The jQuery Example</title>       <script
type="text/javascript"
src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js"></script>
                        <script type="text/javascript" language="javascript">
$(document).ready(function() {             $('div').bind('click', function(
event ){                alert('Event type is ' + event.type);
alert('pageX : ' + event.pageX);                alert('pageY : ' +
event.pageY);                alert('Target : ' + event.target.innerHTML);
});          });       </script>                  <style>          .div{
margin:10px;padding:12px; border:2px solid #666; width:60px;}       </style>
style="background-color:green;">TWO</div>       <div class="div"
style="background-color:red;">THREE</div>              </body>       </html>
```

Nó sẽ cho kết quả sau:

Các phương thức của đối tượng Event trong jQuery

Dưới đây liệt kê các phương thức mà có thể được gọi trên một đối tượng Event trong jQuery:

STT	Phương thức & Miêu tả
1	**preventDefault()** Ngăn cản trình duyệt thực thi hành động mặc định
2	**isDefaultPrevented()** Trả về có hay không phương thức event.preventDefault() đã từng được gọi trên đối tượng Event này
3	**stopPropagation()** Dừng bubble một sự kiện tới các phần tử cha, ngăn cản bất cứ phần tử cha nào được thông báo về sự kiện này
4	**isPropagationStopped()** Trả về có hay không event.stopPropagation() đã từng gọi trên đối tượng Event này

5	**stopImmediatePropagation()** Dừng phần còn lại của các Handler từ việc được thực thi
6	**isImmediatePropagationStopped()** Trả về có hay không event.stopImmediatePropagation() đã từng được gọi trên đối tượng Event này

Các phương thức thao tác đối tượng Event trong jQuery

Bảng dưới liệt kê các phương thức quan trọng liên quan tới Event trong jQuery:

STT	Phương thức & Miêu tả
1	**bind(type, [data], fn)** Bind một Handler tới một hoặc nhiều sự kiện cho mỗi phần tử đã so khớp. Có thể cũng bind các sự kiện Custom
2	**off(events [, selector] [, handler(eventObject)])** Nó gỡ bỏ một sự kiện sống được bind
3	**hover(over, out)** Bắt chước việc hover cho sự di chuyển ví dụ của chuột trên và rời khỏi một đối tượng
4	**on(events [, selector] [, data], handler)** Bind một Handler tới một sự kiện cho tất cả phần tử hiện tại, tương lai, và đã kết nối. Có thể cũng bind các sự kiện custom.
5	**one(type, [data], fn)** Bind một Handler tới một hoặc nhiều sự kiện để được thực thi một lần cho mỗi phần tử đã so khớp
6	**ready(fn)** Bind một hàm để được thực thi bất cứ khi nào DOM sẵn sàng để được thao tác

7	**trigger(event, [data])** Kích hoạt một sự kiện trên mỗi phần tử đã so khớp
8	**triggerHandler(event, [data])** Kích hoạt tất cả Event Handler được bind trên một phần tử
9	**unbind([type], [fn])** Thực hiện ngược lại với bind, nó gỡ bỏ các đối tượng được bind từ mỗi phần tử đã so khớp

Các phương thức Event Helper trong jQuery

jQuery cũng cung cấp một tập hợp các hàm Event Helper mà có thể được sử dụng hoặc để kích hoạt một sự kiện hoặc để bind bất kỳ loại sự kiện nào được đề cập ở phần trên.

Các phương thức Trigger trong jQuery

Dưới đây là ví dụ sẽ kích hoạt sự kiện blur trên tất cả đoạn văn:

```
$("p").blur();
```

Phương thức Binding trong jQuery

Ví dụ sau sẽ bind một sự kiện **click** trên tất cả phần tử trong Thư viện C:

```
$("div").click( function () {     // do something here });
```

Bảng dưới liệt kê đầy đủ tất cả phương thức được hỗ trợ bởi jQuery:

STT	Phương thức & Miêu tả
1	**blur( )** Kích hoạt sự kiện blur của mỗi phần tử đã so khớp
2	**blur(fn)** Bind một hàm tới sự kiện blur của mỗi phần tử đã so khớp
3	**change( )**

	Kích hoạt sự kiện change của mỗi phần tử đã so khớp
4	**change(fn)** Bind một hàm tới sự kiện change của mỗi phần tử đã so khớp
5	**click( )** Kích hoạt sự kiện click của mỗi phần tử đã so khớp
6	**click(fn)** Bind một hàm tới sự kiện click của mỗi phần tử đã so khớp
7	**dblclick( )** Kích hoạt sự kiện dblclick của mỗi phần tử đã so khớp
8	**dblclick(fn)** Bind một hàm tới sự kiện dblclick của mỗi phần tử đã so khớp
9	**error( )** Kích hoạt sự kiện error của mỗi phần tử đã so khớp
10	**error(fn)** Bind một hàm tới sự kiện error của mỗi phần tử đã so khớp
11	**focus( )** Kích hoạt sự kiện focus của mỗi phần tử đã so khớp
12	**focus(fn)** Bind một hàm tới sự kiện focus của mỗi phần tử đã so khớp
13	**keydown( )** Kích hoạt sự kiện keydown của mỗi phần tử đã so khớp
14	**keydown(fn)**

	Bind một hàm tới sự kiện keydown của mỗi phần tử đã so khớp
15	**keypress( )** Kích hoạt sự kiện keypress của mỗi phần tử đã so khớp
16	**keypress(fn)** Bind một hàm tới sự kiện keypress của mỗi phần tử đã so khớp
17	**keyup( )** Kích hoạt sự kiện keyup của mỗi phần tử đã so khớp
18	**keyup(fn)** Bind một hàm tới sự kiện keyup của mỗi phần tử đã so khớp
20	**load(fn)** Bind một hàm tới sự kiện load của mỗi phần tử đã so khớp
21	**mousedown(fn)** Bind một hàm tới sự kiện mousedown của mỗi phần tử đã so khớp
22	**mouseenter(fn)** Bind một hàm tới sự kiện mouseenter của mỗi phần tử đã so khớp
23	**mouseleave(fn)** Bind một hàm tới sự kiện mouseleave của mỗi phần tử đã so khớp
24	**mousemove(fn)** Bind một hàm tới sự kiện mouseover của mỗi phần tử đã so khớp
25	**mouseout(fn)** Bind một hàm tới sự kiện mouseout của mỗi phần tử đã so khớp
26	**mouseover(fn)**

	Bind một hàm tới sự kiện mouseover của mỗi phần tử đã so khớp
27	**mouseup(fn)** Bind một hàm tới sự kiện mouseup của mỗi phần tử đã so khớp
28	**resize(fn)** Bind một hàm tới sự kiện resize của mỗi phần tử đã so khớp
29	**scroll(fn)** Bind một hàm tới sự kiện scroll của mỗi phần tử đã so khớp
30	**select( )** Kích hoạt sự kiện select của mỗi phần tử đã so khớp
31	**select(fn)** Bind một hàm tới sự kiện select của mỗi phần tử đã so khớp
32	**submit( )** Kích hoạt sự kiện submit của mỗi phần tử đã so khớp
33	**submit(fn)** Bind một hàm tới sự kiện submit của mỗi phần tử đã so khớp
34	**unload(fn)** Bind một hàm tới sự kiện unload của mỗi phần tử đã so khớp