Closure là một khái niệm không phải ai cũng biết và thực sự hiểu về nó, đây có thể coi là một cách định nghĩa hàm nâng cao giúp code nhìn trong sáng, cách sử dụng linh hoạt hơn, vì vậy nếu bạn muốn học Javascript nâng cao hoặc học các Javascript Framework khác thì bắt buộc phải hiểu về Closure. Vậy **Closure là gì** thì chúng ta cùng tìm hiểu nhé.

**Nội dung chính**

* [1. Closure là gì?](http://freetuts.net/closure-la-gi-closure-trong-javascript-758.html#goto-h2-0)
* [2. Các ví dụ hàm Closure trong Javascript](http://freetuts.net/closure-la-gi-closure-trong-javascript-758.html#goto-h2-1)
  + [Closure có tham số](http://freetuts.net/closure-la-gi-closure-trong-javascript-758.html#goto-h3-0)
  + [Closure thay đổi giá trị biến toàn cục lẫn cục bộ](http://freetuts.net/closure-la-gi-closure-trong-javascript-758.html#goto-h3-1)
  + [Return nhiều Closure Function](http://freetuts.net/closure-la-gi-closure-trong-javascript-758.html#goto-h3-2)
  + [Độ ưu tiên các biến](http://freetuts.net/closure-la-gi-closure-trong-javascript-758.html#goto-h3-3)
* [3. Lời kết](http://freetuts.net/closure-la-gi-closure-trong-javascript-758.html#goto-h2-2)

**1. Closure là gì?**

Closure là một **hàm được tạo ra từ bên trong một hàm khác** (*hàm cha*), nó có thể sử dụng các biến toàn cục, biến cục bộ của hàm cha và biến cục bộ của chính nó. Việc viết hàm theo kiểu closure trong một số trường hợp sẽ giúp code nhìn sáng và dễ quản lý hơn, linh hoạt hơn trong việc xử lý các chức năng.

**Ví dụ**: Xây dựng hàm closure hiển thị thông báo với câu thông báo được nhận từ một tham số, sau đó nối thêm chữ freetuts.net đằng sau chuỗi đó.

|  |  |
| --- | --- |
| 1  2  3  4  5 | function showMessage(message){      alert(message + ' - Freetuts.net');  }    showMessage('Xin chào các bạn'); |

Cách viết này rất đơn giản và dễ hiểu. Nhưng bây giờ mình muốn viết một dạng khác nhìn rối hơn chút xíu đó là cách sử dụng closure function.

[XEM DEMO](http://freetuts.net/editor.html?id=680)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | // Bước 1: Tạo hàm closure  function showMessage(message)  {      return function(){          alert(message + ' - Freetuts.net');      };  }    // Bước 2: khởi tạo hàm closure  var messageFunc = showMessage('Xin chào các bạn');    // Bước 3: Chạy hàm closure  messageFunc(); |

Rõ ràng nhìn cách này rất rối, khó hiểu nhất là tại bước 2, vì vậy mình sẽ giải thích bước này nhé. Bạn để ý trong hàm showMessage mình đã return về một function, vì vậy lúc khởi tạo và gán hàm nó vào biến messageFunc thì biến messageFunc sẽ là một [#function](http://freetuts.net/hashtag/function) chưa được khởi động, vì vậy câu thông báo sẽ chưa xuất hiện. Đoạn code ở bước 3 sẽ khởi động hàm trả về đó và câu thông báo sẽ xuất hiện.

**2. Các ví dụ hàm Closure trong Javascript**

Cách viết hàm Closure rất linh động, nó phụ thuộc vào bài toán cụ thể mà đưa ra giải pháp khác nhau. Và sau đây mình sẽ đưa ra những ví dụ về một số cách viết thông thường.

**Closure có tham số**

Trong ví dụ ở phần một mình tạo một closure không có tham số, vậy nếu trường hợp có tham số thì cách viết như thế nào? Bạn xem ví dụ dưới đây sẽ hiểu.

[XEM DEMO](http://freetuts.net/editor.html?id=682)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | // Bước 1: Tạo hàm closure  function showMessage(message)  {      return function(time){          for (var i = 1; i <= time; i++){              alert(message + ' - Freetuts.net');          }      };  }    // Bước 2: khởi tạo hàm closure  var messageFunc = showMessage('Xin chào các bạn');    // Bước 3: Chạy hàm closure  messageFunc(2); |

Trong ví dụ này hàm closure trả về có một tham số time, lúc này tại bước 3 ta phải truyền nó vào thì chương trình mới hoạt động bình thường.

**Closure thay đổi giá trị biến toàn cục lẫn cục bộ**

Các closure function có thể sử dụng biến ở 3 phạm vi, thứ nhất là biến toàn cục, thứ hai là biến cục bộ của hàm cha và thứ ba là biến cục bộ của chính nó. Không chỉ sử dụng được mà còn có khả năng thay đổi giá trị của các biến đó.

[XEM DEMO](http://freetuts.net/editor.html?id=681)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27 | // Bước 1: Tạo hàm closure  function Student()  {      var name = '';      var age = '';        return {          set : function(in\_name, in\_age){              name = in\_name;              age = in\_age;          },          getName : function(){              return name;          },          getAge : function(){              return age;          }      };  }    // Bước 2: khởi tạo hàm closure  var studentObj = Student();    // Bước 3: Chạy hàm closure  studentObj.set('Nguyễn Văn Cường', '27');  alert(studentObj.getName());  alert(studentObj.getAge()); |

**Return nhiều Closure Function**

Nếu bạn muốn return nhiều hàm closure thì bạn phải sử dụng một object, trong đó mỗi phần tử sẽ là một closure function.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16 | function multiClosure()  {      return {          func1 : function(){              console.log('Closure1');          },          func2 : function(){              console.log('Closure2');          }      };  }    // Cách sử dụng  var object = multiClosure();  object.func1();  object.func2(); |

**Độ ưu tiên các biến**

Như ta biết closure function có thể sử dụng biến tại ba vị trí đó là biến toàn cục, biến cục bộ hàm cha và biến cục bộ của chính nó. Bây giờ có một trường hợp là tên các biến ở ba vị trí đó bị trùng nhau thì sẽ có chuyện gì xảy ra? Trường hợp này nó sẽ ưu tiên từ trong ra ngoài, nghĩa là nó sẽ thực hiện các bước như sau:

* Bước 1: Xem biến cục bộ trong hàm closure có trùng không? Nếu không trùng thì nó sẽ qua bước 2, còn nếu có trùng thì nó sẽ coi biến đó là biến cục bộ của chính nó.
* Bước 2: Xem biến cục bộ của hàm cha có trùng không? Nếu không trùng thì qua bước 3, nếu trùng thì nó sẽ coi biến đó là biến cục bộ của hàm cha.
* Bước 3: Xem biến toàn cục có trùng không? Nếu không trùng thì nó sẽ khởi tạo mới và đó sẽ là biến cục bộ của hàm closure, nếu trùng thì nó sẽ coi biến đó là biến toàn cục.

Xem ví dụ sau:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | // Bước 1: Tạo hàm closure  var message = 'Biên toàn cục';  function showMessage()  {      var message = 'Biến cục bộ của hàm cha';      return function(){          alert(message);      };  }    // Bước 2: khởi tạo hàm closure  var messageFunc = showMessage();    // Bước 3: Chạy hàm closure  messageFunc(); |

Trong ví dụ này biến message trong hàm closure sẽ lấy giá trị của biến cục bộ hàm cha.

**3. Lời kết**

Qua bài này mình đã giải thích sơ lược về khái niệm **Closure là gì**, đồng thời cũng đưa ra một số ví dụ rất căn bản về cách sử dụng hàm **Closure trong javascript**. Bài này mình chỉ tập trung giải thích về lý thuyết nên các ví dụ đưa ra vẫn chưa thực sự chuyên sâu, hy vọng sẽ giúp ích cho các bạn.