# Lodash

## Loadash là gì

Là một thư viện javascript mạnh mẽ để xử lý array, object, function,…

Lodash giúp việc xử lý các cấu trúc phức tạp một cách dễ dàng, gắn gọn code, hiệu suất cao,…

*Cách cài đặt lodash với npm*

|  |
| --- |
| $ npm I –g npm  $ npm I –save lodash |

*Cách cài đặt loadash với node.js*

|  |
| --- |
| var \_ = require (‘lodash’); |

## Một vài hàm của lodash

**\_.forEach(collection, [iterate = \_.idetity])**

*Lặp lại các phần tử của mảng và thực hiện lặp cho mỗi phần tử.*

|  |  |  |
| --- | --- | --- |
| **VD:**   |  | | --- | | \_.forEach([1, 2], function (value) {  Console.log(value);  } |   **Kết quả:**   |  | | --- | | `1`  `2` | |

**.map(collection, [iterate=\_.identity])**

*Tạo một mảng giá trị bằng cách chạy từng phần tử trong mảng cũ*

|  |  |  |
| --- | --- | --- |
| **VD:**   |  | | --- | | \_.map([1, 2], function (value) {  Return value +100;  } |   **Kết quả:**   |  | | --- | | [101, 102] | |

**\_.find(collection, [predicate=\_.identity], [fromIndex=0])**

*Trả về giá trị đầu tiên thỏa mãn điều kiện*

|  |  |  |
| --- | --- | --- |
| **VD:**   |  | | --- | | var user = [  {‘name’ : ‘Huyen’, ‘age’ : 22},  {‘name’ : ‘Tuan Anh’, ‘age’ : 21},  {‘name’ : ‘Huong’, ‘age’ : 12}  ]  \_.find(user, (element) => element.age === 22; |   **Kết quả:**   |  | | --- | | {‘name’ : ‘Huyen’, ‘age’ : 22} | |

**\_.without(array, [values])**

*Tạo một mảng loại trừ tất cả các giá tị đã cho bằng SameValueZero*

|  |  |  |
| --- | --- | --- |
| **VD:**   |  | | --- | | \_.without([2, 1, 2, 3], 1, 2); |   **Kết quả:**   |  | | --- | | 3 | |

**\_.groupBy(collection, [iterate=\_.identity])**

Tạo một đối tượng bao gồm các khóa được tạo ra từ kết quả của việc chạy từng phần tử của bộ sưu tập qua lần lặp. Thứ tự của các giá trị được nhóm được xác định theo thứ tự chúng xuất hiện trong bộ sưu tập. Giá trị tương ứng của mỗi khóa là một mảng các phần tử chịu trách nhiệm tạo khóa. Vòng lặp được gọi với một đối số: (giá trị).

|  |  |  |
| --- | --- | --- |
| **VD:**   |  | | --- | | \_.without([‘one’, ‘two’, ‘three’, ‘four’], ‘length’); |   **Kết quả:**   |  | | --- | | {‘3’: [‘one’, ‘two’], ‘4’: ‘four’, ‘5’ : ‘three’] | |

**\_.includes(collection, value, [fromIndex=0])**

*Kiểm tra xem giá trị có được có trong chuỗi hay không*

|  |  |  |
| --- | --- | --- |
| **VD:**   |  | | --- | | \_.includes([2, 1, 2, 3], 1); |   **Kết quả:**   |  | | --- | | true | |

**\_.reduce(collection, [iterate=\_.identity],[accumulator])**

*Với mỗi một bước đầu vào sẽ là hai tham số, kết quả sâu khi xử lý sẽ là tham số đầu của vòng lặp tiếp theo*

|  |  |  |
| --- | --- | --- |
| **VD:**   |  | | --- | | \_.reduce([1, 2, 3], (sum, n) => sum+n; |   **Kết quả:**   |  | | --- | | 6 | |

**\_.clone(value)**

*Tạo ra một bản sao mới*

|  |  |  |
| --- | --- | --- |
| **VD:**   |  | | --- | | var objects = [{ 'a': 1 }, { 'b': 2 }];    var shallow = \_.clone(objects);  console.log(shallow[0] === objects[0]); |   **Kết quả:**   |  | | --- | | true | |

**\_.debounce**

*sẽ gọi lại một function sau một khoảng thời gian nhất định kể từ lần cuối cùng function đó được gọi.*

|  |  |
| --- | --- |
| **VD:**   |  | | --- | | function validateEmail() {  // Validate email here and show error message if not valid  }  var emailInput = document.getElementById("email-field");  emailInput.addEventListener("keyup", \_.debounce(validateEmail, 500)); | |

**\_.assign(object, [sources])**

*Đây là hàm hợp nhất các object*

|  |  |  |
| --- | --- | --- |
| |  | | --- | | var firstObject = { 'name': 'A', 'age': 10};  var secondObject = { 'address': 'Ha Noi' }  var object = \_.assign(firstObject, secondObject); |   **Kết quả:**   |  | | --- | | { 'address': 'Ha Noi', 'name': 'A', 'age': 10 } | |