

Lodash Library

What is that and what does it consist of?

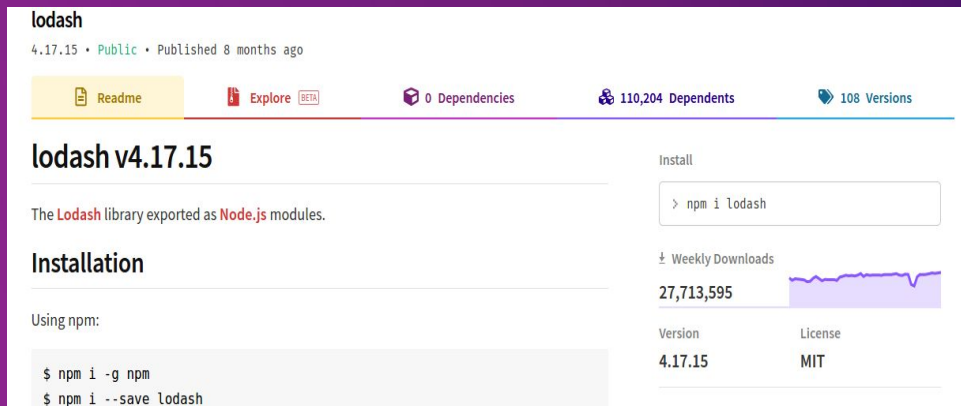
Tunisian **JavaScript** Community
Fourth Meetup

FARES KARBIA

Tunisian **JS** Community

What is a **Lodash** Library?

Lodash is a superset of Underscore. What is Underscore? Underscore is “[...] a whole mess of useful functional programming helpers without extending any built-in objects.” In short, Lodash is a JS helper library for arrays, strings and objects. Documentation is here: <https://lodash.com/docs>



The screenshot shows the Lodash package page on npm. At the top, it displays 'lodash' with version '4.17.15', 'Public' status, and 'Published 8 months ago'. Below this are links for 'Readme', 'Explore' (with a 'BETA' tag), '0 Dependencies', '110,204 Dependents', and '108 Versions'. The main heading is 'lodash v4.17.15'. A description states 'The Lodash library exported as Node.js modules.' The 'Installation' section shows 'Using npm:' with two commands: '\$ npm i -g npm' and '\$ npm i --save lodash'. On the right, there's an 'Install' box with the command '> npm i lodash', a 'Weekly Downloads' chart showing 27,713,595 downloads, and a table with 'Version' 4.17.15 and 'License' MIT.

lodash
4.17.15 • Public • Published 8 months ago

[Readme](#) [Explore](#) [BETA](#) [0 Dependencies](#) [110,204 Dependents](#) [108 Versions](#)

lodash v4.17.15

The **Lodash** library exported as **Node.js** modules.

Installation

Using npm:

```
$ npm i -g npm  
$ npm i --save lodash
```

Install

```
> npm i lodash
```

± Weekly Downloads
27,713,595

Version	License
4.17.15	MIT



_.map()



_.filter()



_.times()



var _ = require('lodash')



_.chain()

_.find()

_.debounce()





Traditional way VS **Lodash**



_.map()

To extract some property from an array of objects:

```
1 let arr = [{ n: 1 }, { n: 2 }]
2 // ES6
3 arr.map((obj) => obj.n)
4 // Lodash
5 _.map(arr, 'n')
```

This can be more helpful when it comes to complex objects:

```
1 let arr = [
2   { a: [ { n: 1 } ] },
3   { b: [ { n: 1 } ] }
4 ]
5 // ES6
6 arr.map((obj) => obj.a[0].n) // TypeError: property 'a' is not defined in arr[1]
7 // Lodash
8 _.map(arr, 'a[0].n') // => [1, undefined]
```

`_.filter()`

For `_.filter`, there's also predicate shorthand.

```
1 let users = [  
2   { 'user': 'barney', 'age': 36, 'active': true },  
3   { 'user': 'fred',   'age': 40, 'active': false }  
4 ];  
5 // ES6  
6 users.filter((o) => o.active)  
7 // Lodash  
8 _.filter(users, 'active')  
9 _.filter(users, ['active', true])  
10 _.filter(users, {'active': true, 'age': 36})
```

`_.chain()`

```
1 let lines = `
2 an apple orange the grape
3 banana an apple melon
4 an orange banana apple
5 `.split('\n')
6
7 _.chain(lines)
8   .flatMap(line => line.split(/\s+/))
9   .filter(word => word.length > 3)
10  .groupBy(_.identity)
11  .mapValues(_.size)
12  .forEach((count, word) => { console.log(word, count) })
13
14 // apple 3
15 // orange 2
16 // grape 1
17 // banana 2
18 // melon 1
```

Processing collections with chaining, lazy evaluation, along with short, easy-to-test functions, is quite popular these days. Most **Lodash** functions regarding collections can be chained easily.

_.debounce will invoke a function after a certain amount of time since the last time it was invoked.

_.debounce()

```
1 function validateEmail() {  
2   // Validate email here and show error message if not valid  
3 }  
4  
5 var emailInput = document.getElementById("email-field");  
6 emailInput.addEventListener("keyup", _.debounce(validateEmail, 500));  
7
```


Instead iterating through an array with a loop to find a specific object, we can simply use `_.find`. That's nice, but this is not the only thing you can do with `_.find`. You can also find an object using multiple properties with a single line of code. Take a look!

`_.find()`

```
1  var users = [  
2    { firstName: "John", lastName: "Doe", age: 28, gender: "male" },  
3    { firstName: "Jane", lastName: "Doe", age: 5, gender: "female" },  
4    { firstName: "Jim", lastName: "Carrey", age: 54, gender: "male" },  
5    { firstName: "Kate", lastName: "Winslet", age: 40, gender: "female" }  
6  ];  
7  
8  var user = _.find(users, { lastName: "Doe", gender: "male" });  
9  // user -> { firstName: "John", lastName: "Doe", age: 28, gender: "male" }  
10  
11  var underAgeUser = _.find(users, function(user) {  
12    | return user.age < 18;  
13  | });  
14  // underAgeUser -> { firstName: "Jane", lastName: "Doe", age: 5, gender: "female" }  
15
```

`_.times` receives as arguments the number of iterations and a function to execute n times and returns an array of the results. Very useful when creating dynamic test data.

`_.times()`

```
1 function getRandomInteger() {  
2   return Math.round(Math.random() * 100);  
3 }  
4  
5 var result = _.times(5, getRandomNumber);  
6 // result => [64, 70, 29, 10, 23]
```



ES6 introduces some useful syntaxes like destructuring, spread and arrow function, which can be used to replace a lot of Lodash functions.

```
1 // Lodash
2 _.head([1, 2, 3]) // => 1
3 _.tail([1, 2, 3]) // => [2, 3]
4 // ES6 destructuring syntax
5 const [head, ...tail] = [1, 2, 3]
6
7 // Lodash
8 let say = _.rest((who, fruits) => who + ' likes ' + fruits.join(','))
9 say('Jerry', 'apple', 'grape')
10 // ES6 spread syntax
11 say = (who, ...fruits) => who + ' likes ' + fruits.join(',')
12 say('Mary', 'banana', 'orange')
13
14 // Lodash
15 _.constant(1)() // => 1
16 _.identity(2) // => 2
17 // ES6
18 (x => (() => x))(1)() // => 1
19 (x => x)(2) // => 2
20
21 // Partial application
22 let add = (a, b) => a + b
23 // Lodash
24 let add1 = _.partial(add, 1)
25 // ES6
26 add1 = b => add(1, b)
27
28 // Curry
29 // Lodash
30 let curriedAdd = _.curry(add)
31 let add1 = curriedAdd(1)
32 // ES6
33 curriedAdd = a => b => a + b
34 add1 = curriedAdd(1)
```

Conclusion

Lodash adds great power to JavaScript language. One can write concise and efficient codes with minor efforts. Besides, Lodash is fully modularized. Though some of its functions will eventually deprecate, but I believe it'll still bring many benefits to developers, while pushing the development of **JS** language as well.



THANKS

Does anyone have any
questions?

fareswardeni@gmail.com

mobelite.fr

CREDITS: This presentation template was created by **Slidesgo**, including icons by **Flaticon**, and infographics & images by **Freepik**.

Please keep this slide for attribution.