Atitit lambda表达式

目录

[1. Lambda 表达式。它们只是创建函数的表达式 1](#_Toc17067)

[1.1. 句法 1](#_Toc17192)

[1.1.1. 基本语法 1](#_Toc14783)

[1.2. 一个参数。不需要简单的表达式返回： 1](#_Toc320)

[1.3. 多个参数需要括号。不需要简单的表达式返回： 2](#_Toc11932)

[1.4. 多行语句需要正文括号并返回： 2](#_Toc18908)

[1.5. 多个参数需要括号。多行语句需要正文括号并返回： 2](#_Toc16723)

[2. 高级语法 2](#_Toc9656)

[2.1. 要返回一个对象字面量表达式，需要在表达式周围加上括号： 2](#_Toc25815)

# Lambda 表达式。它们只是创建函数的表达式

大多数现代编程语言（Python、Ruby、Java...）中都存在 Lambda 表达式。它们只是创建函数的表达式。这对于支持一流函数的编程语言来说非常重要，这基本上意味着将函数作为参数传递给其他函数或将它们分配给变量。

在 ES6 之前的 JavaScript 中，我们有函数表达式，它给我们一个匿名函数（一个没有名字的函数）。

var anon = function (a, b) { return a + b };

## [句法](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Functions/Arrow_functions" \l "syntax" \o "语法固定链接)

### [基本语法](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Functions/Arrow_functions" \l "basic_syntax" \o "基本语法的永久链接)

## 一个参数。不需要简单的表达式返回：

param => expression

复制到剪贴板

## 多个参数需要括号。不需要简单的表达式返回：

(param1, paramN) => expression

复制到剪贴板

## 多行语句需要正文括号并返回：

param => {

let a = 1;

return a + param;}

复制到剪贴板

## 多个参数需要括号。多行语句需要正文括号并返回：

(param1, paramN) => {

let a = 1;

return a + param1 + paramN;}

复制到剪贴板

# [高级语法](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Functions/Arrow_functions" \l "advanced_syntax" \o "高级语法的永久链接)

## 要返回一个对象字面量表达式，需要在表达式周围加上括号：

params => ({foo: "a"}) // returning the object {foo: "a"}

复制到剪贴板

支持[其余参数](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Functions/rest_parameters)：

(a, b, ...r) => expression

复制到剪贴板

支持[默认参数](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Functions/Default_parameters)：

(a=400, b=20, c) => expression

复制到剪贴板

[解构](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Destructuring_assignment) PARAMS中支持：

([a, b] = [10, 20]) => a + b; // result is 30({ a, b } = { a: 10, b: 20 }) => a + b; // result is 30