

2. Scope Chain & Closure

1. 作用域链(Scope Chain)

JavaScript 引擎在运行时会维护一个作用域链。

作用域链的构建过程是：从当前作用域开始，依次向上查找，直到找到全局作用域。

作用域链的构建顺序是：从当前作用域开始，依次向上查找，直到找到全局作用域。

```
let globalVar = "Global";

function outer() {
  let outerVar = "Outer";

  function inner() {
    let innerVar = "Inner";

    console.log(innerVar); // "Inner" (当前作用域)
    console.log(outerVar); // "Outer" (外层作用域)
    console.log(globalVar); // "Global" (全局作用域)
  }

  inner();
}

outer();
```

► 作用域链的构建顺序是：从当前作用域开始，依次向上查找，直到找到全局作用域。

► 作用域链的构建过程是：从当前作用域开始，依次向上查找，直到找到全局作用域。

2. 闭包(Closure)

闭包是指有权访问其他函数作用域内变量的函数。

我们使用 let 来定义 count，使用 return 来返回一个函数。

```
function createCounter() {  
  let count = 0; // 我们使用 let 来定义 count  
  
  return function () {  
    count++; // 每次调用函数，count 就加 1  
    console.log(count);  
  };  
}  
  
const counter = createCounter();  
counter(); // 1  
counter(); // 2  
counter(); // 3
```

- 我们使用 let 来定义 count。
- 我们使用 return 来返回一个函数。
- 我们使用 console.log(count) 来输出 count 的值。

3. 使用 let 和 const

ES6 之前我们使用 var 来定义变量，但是 var 有一些问题。ES6 引入了 let 和 const 来定义变量。

```
function testVar() {  
  if (true) {  
    var x = 10;  
  }  
  console.log(x); // 10 (var 在函数内部定义，但在函数外部也能访问)  
}  
  
function testLet() {  
  if (true) {  
    let y = 20;  
  }  
  console.log(y); // Error: y is not defined (let 只在块级范围内有效)
```

```
}  
  
testVar();  
testLet();
```

▶ var 的变量提升规则是：变量提升发生在代码执行之前。

▶ let 和 const 的变量提升规则是：变量提升发生在代码执行时。

4. 闭包函数

闭包函数是指：一个函数内部定义了一个函数，这个内部函数可以访问外部函数的变量和函数。闭包函数就是返回这个内部函数的函数。

```
function createResource() {  
  let resource = "Heavy Resource";  
  
  return function () {  
    console.log(resource);  
  };  
}  
  
const useResource = createResource();  
useResource(); // "Heavy Resource"  
  
// 闭包函数 的 变量 提升 规则 是 null 的 变量 提升  
useResource = null;
```

▶ 闭包函数是指：一个函数内部定义了一个函数，这个内部函数可以访问外部函数的变量和函数。闭包函数就是返回这个内部函数的函数。

▶ 闭包函数是指：一个函数内部定义了一个函数，这个内部函数可以访问外部函数的变量和函数。闭包函数就是返回这个内部函数的函数。

5. 立即调用的函数表达式(IIFE)

IIFE(Immediately Invoked Function Expression)是指：立即调用的函数表达式。它是一种函数表达式，在定义的同时立即调用。

```
(function () {  
  let secret = "This is private";  
  console.log(secret); // "This is private"  
})();  
  
console.log(secret); // Error: secret is not defined
```

IIFE 是 一个 立即 执行的 函数。

ES6 引入 了 let, const 来 声明 变量, 变量 声明。

变量 声明: 变量 声明 的 顺序 → 变量 声明 → 变量 声明 的 值。

变量: 变量 声明 的 顺序, 变量 声明 的 值。

变量 声明 的 顺序: var let/const 变量 声明 的 顺序。

变量 声明: 变量 声明, 变量 声明 的 顺序 变量 声明 的 值。 (变量 声明 的 值)

IIFE: 一个 立即 执行的 函数

变量 声明 的 顺序

变量 声明 的 顺序 变量 声明 的 值 变量 声明 的 值。

```
function createEventHandlers(elements) {  
  elements.forEach((element, index) => {  
    element.addEventListener("click", () => {  
      console.log(`Element ${index} clicked`);  
    });  
  });  
}
```

```
const buttons = document.querySelectorAll("button");  
createEventHandlers(button
```

Revision #1

Created 20 April 2025 05:27:08 by DainChoi

Updated 20 April 2025 05:27:08 by DainChoi