简书





## JS预编译精解



j CodeLife

2020.08.04 20:46:40 字数 759 阅读 27

编辑文章

JS运行分三个步骤:

- 语法分析
- 预编译
- 解释执行

通常**js**在执行代码前,系统会先执行语法分析,通篇扫描一遍看是否有语法错误,有错误,程序终止,没有错误就会走到预编译环节,预编译又称预处理,主要做一些代码变量的提升工作,它发生在函数执行的前一刻。预编译执行完才会解释执行,读一行执行一行,读一行执行一行。这就是**JS**运行的整个过程。

语法分析和解释执行都比较好理解,预编译到底做了什么事情呢,对代码执行有什么影响,我 们先来看看一个简单的代码

```
1  | fn();
2  | function fn(){
3          console.log("fn");
4  | }
```

结果在控制台会打印出fn,我们奇怪的发现fn的执行语句在fn出生前都能执行,按道理说不应该,程序读一行执行一行,这还没有fn函数呢。

事实上着都是预编译的结果,在全局预编译的时候,它会自动将函数整体提升到最上面。 预编译发生在函数执行前的前一刻,具体过程分四步:

- 1. 创建AO对象; (Active Object, 执行期上下文)
- 2. 找形参和变量声明,将变量名和形参名作为AO对象的属性名,值为undefined;
- 3. 将实参和形参相统一;
- 4. 在函数体里面找函数声明(不包括函数表达式),将函数声明的名作为AO对象的属性名挂起,值为该函数。

预编译导致两个规律:

- 1. 函数声明整体提升
- 2. 变量声明提升

所以预编译的目的是,把AO对象创建好,方便一会执行的时候去用。

在全局开始执行的前一刻,会发生全局预编译,生成的对象是GO(Global Object),但是全局没有参数,也就少了一步(形参和实参相统一)

- **1.** 创建GO对象; (Global Object, 全局上下文)
- 2. 找变量声明,将变量名作为AO对象的属性名,值为undefined;
- 3. 在全局找函数声明(不包括函数表达式),将函数声明的名作为GO对象的属性名挂起,值为该函数。

全局预编译发生在全局开始执行前,所以先生成GO,再生成AO

## GO就是window就是全局

全局上有两个规则,也称预编译前奏:

## 推荐阅读

Web 前端面试

阅读 149

前端模块化的四种规范

阅读 87

vue.config.js打包优化

阅读 263

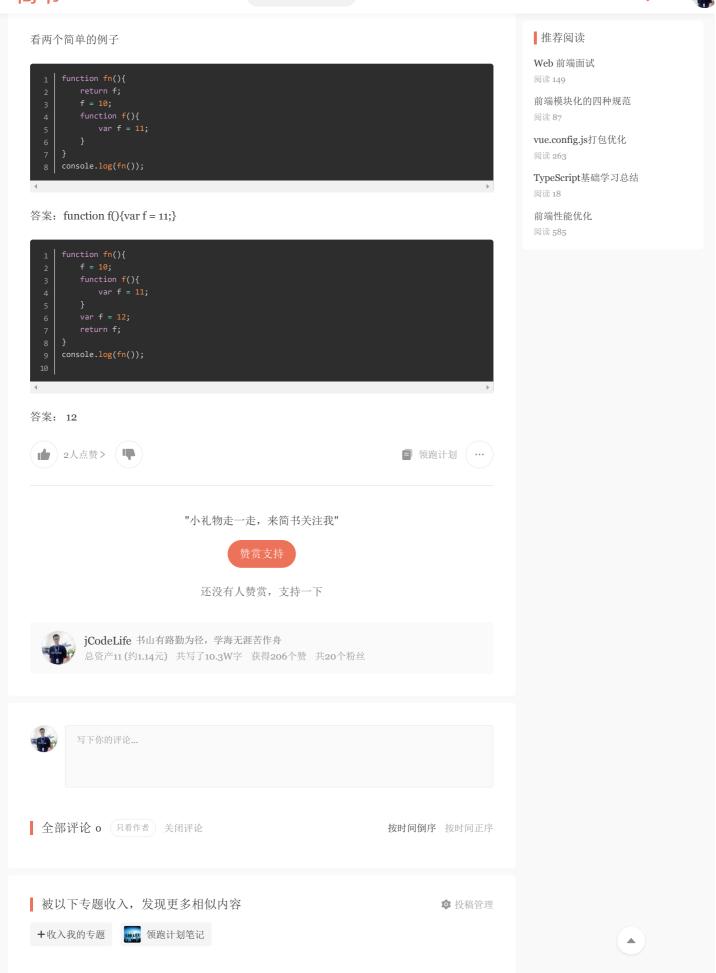
TypeScript基础学习总结

阅读 18

前端性能优化

阅读 585





发现 关注 消息

搜索



## 推荐阅读

Web 前端面试

阅读 149

前端模块化的四种规范

阅读 87

vue.config.js打包优化

阅读 263

TypeScript基础学习总结

阅读 18

前端性能优化

阅读 585

