JavaScript进阶

---JS预解析

河北师范大学软件学院
Software College of Hebei Normal University

内容提纲

- JS解析及执行简介
- JS预解析（声明提升）
- 预解析与作用域

JS解析及执行简介

- JS脚本语言（非提前编译，由解析器边解析边执行）
  - 区别于C/C++编译成二进制和Java/C#编译成字节码（运行在跨平台虚拟机上）的解析执行
- JS代码案例（思考：是否会报错，区别于其他语言）

```
console.log(a);//undefined
var a = 2;
console.log(a);//2
```

```
//从解析器角度看到的代码
var a;
console.log(a);
a = 2;
console.log(a);
```

- JS的解析和执行过程
  - 全局预解析阶段（全局变量和函数声明前置）
  - 全局顺序执行阶段（变量赋值、函数调用等操作）
  - 当遇到函数调用时，在执行函数内代码前，进行函数范围内的预解析
  - 当存在函数嵌套时，以此类推，会进行多次函数预解析

注：解析和执行是一个不断交替的过程

内容提纲

- JS解析及执行简介
- JS预解析（声明提升）
- 预解析与作用域

JS预解析-声明提升

- 预解析主要工作（变量声明和函数声明提升）
  - 解析器在执行代码前的进行代码扫描（var、function）
  - 将变量和函数声明在当前作用域（全局、函数）内进行提升
- 变量声明提升案例

```
console.log(a);
var a = 1;
console.log(a);
```

等价于=>

```
var a;
console.log(a);//undefined
a = 1;
console.log(a);//1
```

参见实例demo08_Part1

JS预解析-声明提升

- 函数声明提升案例

```
foo();//f_2
function foo(){
    console.log("f_1");
}
function foo(){
    console.log("f_2");
}
```

等价于=>

```
function foo(){
    console.log("f_1");
}
function foo(){
    console.log("f_2");
}
foo();//f_2
```

注：ES5中函数及变量声明重复的话，相当于覆盖　参见实例demo08_Part2

JS预解析-声明提升

- 同时有var和function关键字时（情形1：函数表达式）

```
foo();//报错
var foo = function(){
    console.log("foo");
}
```

```
▶Uncaught TypeError: foo is not a function
```

- 当function前有运算符的话，认定为表达式，不提升

```
//上述代码等效如下
var foo;
foo();//报错
foo = function(){
    console.log("foo");
};
```

```
//思考以下代码是否会报错
console.log(foo);//输出什么
var foo = function(){
    console.log("foo");
};
foo();//是否会报错
```


参见实例demo08_Part3

JS预解析-声明提升

- 同时有var和function关键字时（情形2：变量名同函数名）

```
AA();
function AA(){
    console.log("AA_1");
}

var AA = function AA(){
    console.log("AA_2");
};
AA();
```

等价于=>

```
function AA(){
    console.log("AA_1");
}
var AA;//在最顶端和在这等效

AA();
AA = function AA(){
    console.log("AA_2");
};
AA();
```


参见实例demo08_Part4

内容提纲

- JS解析及执行简介
- 变量及函数声明前置
- **预解析与作用域**

JS变量作用域简介

- 变量的作用域是指变量在何处可以被访问到
  - JS采用的是静态词法作用域，代码完成后作用域链就已形成，与代码的执行顺序无关
- 全局变量与局部变量
  - 全局变量：拥有全局作用域的变量（JS代码中任何地方都可以访问）
  - 全局变量是跨域了所有函数自身作用域的自由变量，可以在函数内和函数外直接访问
  - 局部变量：函数内声明的变量，只在函数体内有定义，作用域是局部性的
  - 在函数外不能直接访问函数的局部变量，但可以通过闭包来访问
  - 函数内访问同名变量时，局部变量会覆盖全局变量
- ES5中无块作用域（ES5作用域缺陷及解决办法参见IIFE）
  - 全局作用域、函数作用域、ES5中可以使用函数立即执行表达式来模拟块作用域

参见实例demo09 作用域的案例（回顾）

JS预解析与作用域

- 声明前置与作用域的关系（全局作用域、函数作用域）

```
if(true){
    var i = 0;
}
```

等价于=>

```
var i;
if(true){
    i = 0;
}
```

```
function foo(){
    console.log("j:",j);//undefined
    var j = 10;
    console.log("j:",j);//10
}
foo();

console.log("i:",i);
console.log("j:",j);
```

等价于=>

```
function foo(){
    var j;
    console.log("j:",j);//undefined
    j = 10;
    console.log("j:",j);//10
}
foo();

console.log("i:",i);//0
console.log("j:",j);//报错
```


参见实例demo10 预解析与作用域

总结

- JS解析及执行简介
- 变量及函数声明前置
- 预解析与作用域

Thank You!

作业：

- 复习本章节课件及练习