JavaScript进阶

---JS逻辑运算符进阶

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

内容提纲

- &&与||的基本理解及应用
- &&与||的深层次理解（非布尔类型）
- &&与||在实际中的应用

&&与||基本理解和应用

- 最常见情况（运算符两边的操作数都是布尔类型）
  - 对于&&来说， 除了两侧都为真时为真，其他情况都为假
  - 对于||来说，除了两侧都为假时为假，其他情况都为真

```
console.log(2>1&&4<5);
console.log(true&&(!2));
console.log(false&&("2" == 2));
console.log(false&&false);
```

```
true
false
false
false
```

```
console.log(2>1||4<5);
console.log(true||(!2));
console.log(false||("2" == 2));
console.log(false||false);
```

```
true
true
true
false
```

参见实例demo14

内容提纲

- &&与||的基本理解及应用
- &&与||的深层次理解（非布尔类型）
- &&与||在实际中的应用

&&与||深层次理解

- 当逻辑运算符&&和||两侧的操作数不是布尔类型时
  - 首先将左操作数转换成布尔类型
  - 对转换后的左操作数进行逻辑判断（true or false）
  - 根据短路原则返回原始左操作数或原始右操作数
- 短路原则（忽略对右操作数的判断）
  - 对于&&，转换后的左操作数若为true，则直接返回原始右操作数，若为false则直接返回原始左操作数
  - 对于| |，转换后的左操作数若为true，则直接返回原始左操作数，若为false则直接返回原始右操作数
  - 通过短路原则，可以用&&和| |来实现复杂的条件语句来代替if-else

&&与||深层次理解

- 实例：&&和||两侧的操作数不是布尔类型时

```
console.log(2&&4);
console.log(0&&4);
console.log({x:2}&&{name:"Jame"});
console.log(null&&"hello");
console.log({}&&"world");
```

```
4
0
▶ Object {name: "Jame"}
null
world
```

```
console.log(2||4);
console.log(0||4);
console.log({x:2}||{name:"Jame"});
console.log(null||"hello");
console.log({}||"world");
```

```
2
4
▶ Object {x: 2}
hello
▶ Object {}
```


参见实例demo15 布尔对象实例

内容提纲

- &&与||的基本理解及应用
- &&与||的深层次理解（非布尔类型）
- &&与||在实际中的应用

&&与||在实际中的应用

- 遵循短路特性，使用&&和||可用来实现条件语句

```
var score = 76;
if(score>90){
    console.log("优");
}else if(score>75){
    console.log("良");
}else if(score>60){
    console.log("及格");
}else{
    console.log("不及格");
}

//通过&&和||的组合实现如上功能，注：小括号优先级最高
console.log((score>90&&"优")||(score>75&&"良")||(score>60&&"及格")||"不及格");
```


参见实例demo16 查看通过逻辑运算符来实现条件语句

&&与||在实际中的应用

- 使用||来设置函数参数的默认值
  - 函数定义时可以给参数指定默认值，调用时若未传参数则该参数的值取它定义时的默认值
  - JS（ES6之前）不能直接为函数的参数指定默认值，可以通过 | | 来实现

```
var sum = function(a,b,c){
    b = b||4;
    c = c||5;
    return a+b+c;
}
console.log(sum(1,2,3));//1+2+3
console.log(sum(1,2));  //1+2+5
console.log(sum(1));    //1+4+5
```

案例中：未传实参的话，形参初始为undefined，undefined转为布尔类型为false，根据||短路原则直接返回右操作数，等效代码如下

```
var sum = function(a,b=4,c=5){
    return a+b+c;
}
```

参见实例demo17 前半部分

问题：思考sum(1,0,0)返回多少？1还是10

&&与||在实际中的应用

- sum函数的问题及完善
  - 若实参转换为布尔类型为false，返回值则可能不是预期结果，如sum(1,0,0)为10
  - 增加判断，确保实参转换为布尔类型时为true

```
var sum = function(a,b,c){
    if(b!=false){b = b||4;} //(b!=false)&&(b=b||4);
    if(c!=false){c = c||5;} //(c!=false)&&(c=c||5);
    return a+b+c;
};
console.log(sum(1,2,3));//1+2+3
console.log(sum(1,2));  //1+2+5
console.log(sum(1));    //1+4+5
console.log(sum(1,0,0));//1+0+0
```

参见实例demo17 后半部分

总结

- &&与||的基本理解及应用
- &&与||的深层次理解（非布尔类型）
- &&与||在实际中的应用

补充

- ！运算符与逻辑运算符

  ！（A&&B） === ！A ||！B

  ！（A||B） === ！A &&！B

  用!!实现布尔类型转换

$$\overline{A \cap B} = \overline{A} \cup \overline{B}$$

$$\overline{A \cup B} = \overline{A} \cap \overline{B}$$

- 使用||实现返回值的限制

```
function foo(a,b){
    return (a+b)||"和不能为0";
}
foo(-2,2);//和不能为0
```

Thank You!

作业

- 复习本章课件及练习
- 将自己的本章练习提交到个人仓库中
- 慕课网（选看，第1-4章，JS基础好的不要求）
  https://www.imooc.com/learn/10
- 慕课网（必看，第1-3章）
  https://www.imooc.com/learn/277