JavaScript 进阶 - 第3天

了解构造函数原型对象的语法特征，掌握 JavaScript 中面向对象编程的实现方式，基于面向对象编程思想实现 DOM 操作的封装。

- 了解面向对象编程的一般特征
- 掌握基于构造函数原型对象的逻辑封装
- 掌握基于原型对象实现的继承
- 理解什么原型链及其作用
- 能够处理程序异常提升程序执行的健壮性

面向对象

学习 JavaScript 中基于原型的面向对象编程程序的语法实现，理解面向对象编程的特征。

面向对象编程是一种程序设计思想，它具有 3 个显著的特征：封装、继承、多态。

封装

继承

多态

封装

封装数组解构是将数组的单元值快速批量赋值给一系列变量的简洁语法

封装的本质是将具有关联的代码组合在一起，其优势是能够保证代码复用且易于维护，函数是最典型也是最基础的代码封装形式，面向对象思想中的封装仍以函数为基础，但提供了更高级的封装形式。

命名空间

先来回顾一下以往代码封装的形式：

```
<script>
  // 普通对象（命名空间）形式的封装
  let beats = {
    name: '狼',
    setName: function (name) {
      this.name = this.name;
    },getName() {
      console.log(this.name);
    }
  }
  beats.setName('熊');beats.getName();
</script>
```

构造函数

构造函数相当于一个"模子"，能够像字面量那样创建出对象来，所不同的是借助构造函数创建出来的实例对象之间是**彼此不影响**的。

- 总结：

- 构造函数体现了面向对象的封装特性
- 构造函数实例创建的对象彼此独立、互不影响
- 命名空间式的封装无法保证数据的独立性
- 注：可以举一些例子，如女娲造人等例子，加深对构造函数的理解。

原型对象

实际上每一个构造函数都有一个名为 prototype 的属性，译成中文是原型的意思，prototype 的是对象类据类型，称为构造函数的原型对象，每个原型对象都具有 constructor 属性代表了该原型对象对应的构造函数。

当访问对象的属性或方法时，先在当前实例对象是查找，然后再去原型对象查找，并且原型对象被所有实例共享。

- 什么是原型对象
- 答：是构造函数的一个属性，它的数据类型是对象
- 原型对象有啥用？？
- 答：原型对象对应的构造函数的实例方法或属性不存在时会去查找原型对象
- 总结：**结合构造函数原型的特征，实际开发重往往会将封装的功能函数添加到原型对象中。**

继承

继承是面向对象编程的另一个特征，通过继承进一步提升代码封装的程度，JavaScript 中大多是借助原型对象实现继承的特性。

原型继承：基于构造函数原型对象实现面向对象的继承特性。

创建对象将公共的的属性和方法独立出来，然后赋值给构造函数的 prototype 这样无论有多少个子集都可以共享公共的属性和方法了：

继承：把实例对象赋值给原型对象，指回构造函数本身

原型链

基于原型对象的继承使得不同构造函数的原型对象关联在一起，并且这种关联的关系是一种链状的结构，我们将原型对象的链状结构关系称为原型链

在 JavaScript 对象中包括了一个非标准备的属性 __proto__ 它指向了构造函数的原型对象，通过它可以清楚的查看原型对象的链状结构。

最后整理

面向对象（OOP）是编程时的一种指导思想，需要通过不断的实践才能体会面向对象编程的优势，在 JavaScript 中面向对象编程的实现是以构造函数和原型对象为核心的，因此掌握构造函数和原型对象的语法是灵活运用面向对象的基础。

面向对象多态的特性在 JavaScript 中应用场景相对较少，本次课中暂不讲解。

异常处理

了解 JavaScript 中程序异常处理的方法，提升代码运行的健壮性。

throw

异常处理是指预估代码执行过程中可能发生的错误，然后最大程度的避免错误的发生导致整个程序无法继续运行。

- 总结：
- throw 抛出异常信息，程序也会终止执行
- throw 后面跟的是错误提示信息
- Error 对象配合 throw 使用，能够设置更详细的错误信息

try ... catch

- 总结：
- try...catch 用于捕获错误信息
- 将预估可能发生错误的代码写在 try 代码段中
- 如果 try 代码段中出现错误后，会执行 catch 代码段，并截获到错误信息

传智教育旗下高端IT教育品牌