jQuery 中的设计模式

从入门到工作：JS 编程接口

版权声明

联系方式

如果你想要购买本课程
请微信联系 xiedaimala02 或 xiedaimala03

如果你发现有人盗用本课程
请微信联系 xiedaimala02 或 xiedaimala03

用 jQuery 风格
重新封装 DOM

这节课你可能经常对自己说：我怎么没想到？！

不用慌张
因为你肯定想不到

这是许多程序员多年摸索出来的经典代码

你只需要站在巨人的肩膀上，继续向上探索

链式风格

- 也叫 jQuery 风格
  - window.jQuery() 是我们提供的全局函数
- 特殊函数 jQuery
  - jQuery(选择器) 用于获取对应的元素
  - 但它却不返回这些元素
  - 相反，它返回一个对象，称为 jQuery 构造出来的对象
  - 这个对象可以操作对应的元素
  - 听不懂？直接写代码！

jQuery 是构造函数吗?

- 是
  - ✓ 因为 jQuery 函数确实构造出了一个对象
- 不是
  - ✓ 因为不需要写 new jQuery() 就能构造一个对象
  - ✓ 以前讲的构造函数都要结合 new 才行
- 结论
  - ✓ jQuery 是一个不需要加 new 的构造函数
  - ✓ jQuery 不是常规意义上的构造函数
  - ✓ 这是因为 jQuery 用了一些技巧（目前没必要讲）

jQuery 对象代指 jQuery 函数构造出来的对象

口头约定

jQuery 对象

不是说「jQuery 这个对象」

一定要记清楚

术语

- 举例

- Object 是个函数
- Object 对象表示 Object 构造出的对象
- Array 是个函数
- Array 对象/数组对象表示 Array 构造出来的对象
- Function 是个函数
- Function 对象/函数对象表示 Function 构造出来的对象

链式风格

- 查
  - jQuery('#xxx') 返回值并不是元素，而是一个 api 对象
  - jQuery('#xxx').find('.red') 查找 #xxx 里的 .red 元素
  - jQuery('#xxx').parent() 获取爸爸
  - jQuery('#xxx').children() 获取儿子
  - jQuery('#xxx').siblings() 获取兄弟
  - jQuery('#xxx').index() 获取排行老几（从0开始）
  - jQuery('#xxx').next() 获取弟弟
  - jQuery('#xxx').prev() 获取哥哥
  - jQuery('.red').each(fn) 遍历并对每个元素执行 fn

什么？你嫌 jQuery 太长

你是对的

window.$ = window.jQuery

还记得 bash alias 吗，添加一个别名即可

命名风格

- 下面的代码令人误解
  - const div = $('div#test')
  - 我们会误以为 div 是一个 DOM
  - 实际上 div 是 jQuery 构造的 api 对象
  - 怎么避免这种误解呢?
- 改成这样
  - const $div = $('div#test')
  - $div.appendChild 不存在，因为它不是 DOM 对象
  - $div.find 存在，因为它是 jQuery 对象

我代码中所有 $ 开头的变量都是 jQuery 对象

这是约定，除非特殊说明

链式风格

- 查
  - $('#xxx') 返回值并不是元素，而是一个 api 对象
  - $('#xxx').find('.red') 查找 #xxx 里的 .red 元素
  - $('#xxx').parent() 获取爸爸
  - $('#xxx').children() 获取儿子
  - $('#xxx').siblings() 获取兄弟
  - $('#xxx').index() 获取排行老几（从0开始）
  - $('#xxx').next() 获取弟弟
  - $('#xxx').prev() 获取哥哥
  - $('.red').each(fn) 遍历并对每个元素执行 fn

链式风格

- 增
  - $(' 1 ')
  - 等下，你刚刚不是说 $(选择器) 吗?
  - 怎么又可以传标签内容?
- 所以说 jQuery 牛 X 呀
  - 这是 JS 的函数「重载」
  - 一个函数可以接受不同的参数
  - 怎么做到的？写代码！
  - http://js.jirengu.com/hasok/1/edit?html,js,output

链式风格

- 增
  - $(' 1 ')
  - 返回值并不是新增的元素，而是 api 对象
  - $(' 1 ').appendTo(...)
  - appendTo 可以把新增的元素放到另一个元素里
- 这是一种什么感觉
  - 就感觉 DOM 是不可见的
  - 你不需要知道 DOM 的任何细节
  - 只需要使用简洁的 API 即可
  - 一个好的封装，能让使用者完全不知道内部细节
  - 这是通过闭包实现的

我就是想知道细节咋办

- 举例1
  - const $div = $('div#test')
  - $div 并不是 DOM 对象，而是 jQuery 构造的 api 对象
  - 我现在就是想从 $div 得到 div 元素，行不行?
- 满足你
  - $div.get(0) 获取第 0 个元素 // div
  - $div.get(1) 获取第 1 个元素 // undefined
  - $div.get(2) 获取第 2 个元素 // undefined
  - $div.size() 得到元素的个数
  - $div.length 也可以得到元素的个数
  - $div.get() 获取所有元素组成的数组 // [div]

我就是想知道细节咋办2

- 举例2
  - const $div = $('.red') // 假设有 3 个 div.red
  - $div 不是 DOM 对象们，而是 jQuery 构造的 api 对象
  - 我现在就是想从 $div 得到 3 个 div 元素，行不行?
- 满足你
  - $div.get(0) 获取第 0 个元素 // div
  - $div.get(1) 获取第 1 个元素 // div
  - $div.get(2) 获取第 2 个元素 // div
  - $div.size 得到元素的个数
  - $div.length 也可以得到元素的个数
  - $div.get() 获取所有元素组成的数组 // [div,div,div]

觉得 $div.get(0) 太麻烦了

再改简单点！

$div[0] 获取第一个 div

满足吗？那我开始写代码了

链式风格

- 增
  - $('body') 获取 document.body
  - $('body').append($(' 1 ')) 添加小儿子
  - $('body').append(' 1 ') 更方便
  - $('body').prepend(div或$div) 添加大儿子
  - $('#test').after(div 或 $div) 添个弟弟
  - $('#test').before(div 或 $div) 添个哥哥

链式风格

- 删
  - $div.remove()
  - $div.empty()

链式风格

- 改
  - $div.text(?) 读写文本内容
  - $div.html(?) 读写 HTML 内容
  - $div.attr('title', ?) 读写属性
  - $div.css({color: 'red'}) 读写 style // $div.style 更好
  - $div.addClass('blue') / removeClass / hasClass
  - $div.on('click', fn)
  - $div.off('click', fn)
- 注意
  - $div 可能对应了多个 div 元素

后续

- 使用原型
  - 把共用属性（函数）全都放到 $.prototype
  - $.fn = $.prototype // 名字太长不爽
  - 然后让 api.__proto__ 指向 $.fn
- 把代码公开
  - 发布到 GitHub
  - 添加文档，告诉别人怎么样
  - 获得称赞
  - 这就是程序员的社区，人人为我，我为人人

jQuery 有多牛 X

它是目前前端最长寿的库，2006年发布

它是世界上使用最广泛的库，全球 80% 的网站在用

设计模式?

- jQuery 用到了哪些设计模式
  - ✓ 不用 new 的构造函数，这个模式没有专门的名字
  - ✓ $(支持多种参数)，这个模式叫做重载
  - ✓ 用闭包隐藏细节，这个模式没有专门的名字
  - ✓ $div.text() 即可读也可写，getter / setter
  - ✓ $.fn 是 $.prototype 的别名，这叫别名
  - ✓ jQuery 针对不同浏览器使用不同代码，这叫适配器
- 设计模式是啥
  - ✓ 老子这个代码写得太漂亮了，别人肯定也用得到
  - ✓ 那就给这种写法取个名字吧，比如适配器模式
  - ✓ 设计模式就是对通用代码取个名字而已

我应该学习设计模式吗?

- 设计模式不是用来学的
  - 你看了这些代码
  - 但你并不知道这代码用来解决什么问题
  - 看了白看
- 设计模式是用来总结的
  - 你直管去写代码
  - 把你的代码尽量写好，不断重写
  - 总结你的代码，把写得好的地方抽象出来
  - 看看符合哪个设计模式
  - 你就可以告诉别人你用到了这几个设计模式
  - 显得你特别高端

有人说不用学 jQuery

- 真相
  - ✓ jQuery 这么简单、经典的库为什么不学?
  - ✓ 通过 jQuery 可以学会很多封装技巧，为什么不学?
  - ✓ 连 jQuery 都理解不了，Vue / React 肯定学不好
- 推荐文章
  - ✓ 《jQuery 都过时了，那我还学它干嘛？》
- 学习路线
  - ✓ 理解 jQuery 原理
  - ✓ 使用 jQuery 做一两个项目
  - ✓ 总结一篇博客，然后再也不碰 jQuery 了
  - ✓ 滚去学 Vue / React，找工作要紧 ☺

再见

下节课实际使用一下 jQuery 库