那些年我们追过的jquery ----前言

不知什么时候jQuery闯进了我们的世界，不知过了多久离开了我们的视野。

辣条A：我们为什么要追jquery？

辣条A：因为她好看吗？



辣条B：瞎说什么大实话。

不是说笑，jQuery确实是不仅优而且雅，让众多程序员一见倾心，爱不释手



先来总结一下，她的优点：

1. 强大的浏览器间的兼容性
2. 强大的选择器
3. 隐式迭代
4. 出色的DOM操作封装.
5. 链式编程
6. 支持丰富的扩展

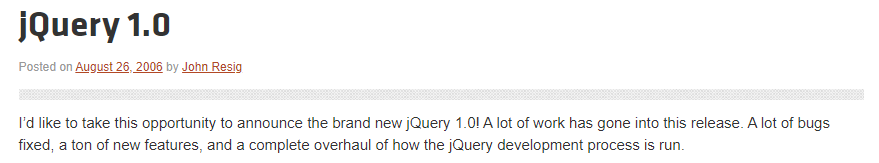
本文重点就是介绍上面这几点。都清楚的就不用往下看了。stand up下课，记得点波关注。

jQuery的作者。John Resig，<http://ejohn.org/>

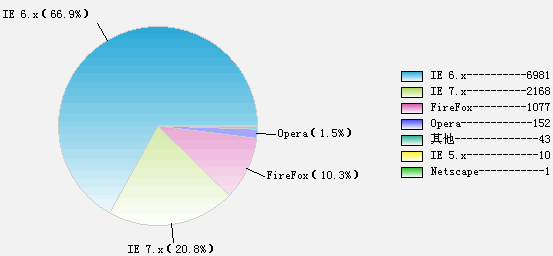


嗯~一个\*\*的胖子，果然很程序员。

1. 强大的兼容性



Jquer可以说是PC时代的英雄，官网上可以看到第一版正式版jQuery发布于2006年8月26日，相当有年代感了。当时是PC的天下，就是微软的天下，自然主流浏览器就是IE，当时的IE的版本是6.0。好可怕（2007年各个浏览器市场占比）



对ie有了解的同学都知道这玩意的兼容性有多恶心，更不用说是6.0的古老版本。仗着平台优势强推VBScript，而其他浏览器则使用JavaScript，导致前端开发的脚本需要至少写两套（虽然大多其他浏览器使用JavaScript但是对标准的实现都有些许差异。

比如获取span元素中的文本内容（现在都是innerText））。

<span id="text">文本内容</span>

    <script>

        // Chrome

        document.getElementById('text').innerText

        // Firefox

        document.getElementById('text').textContent

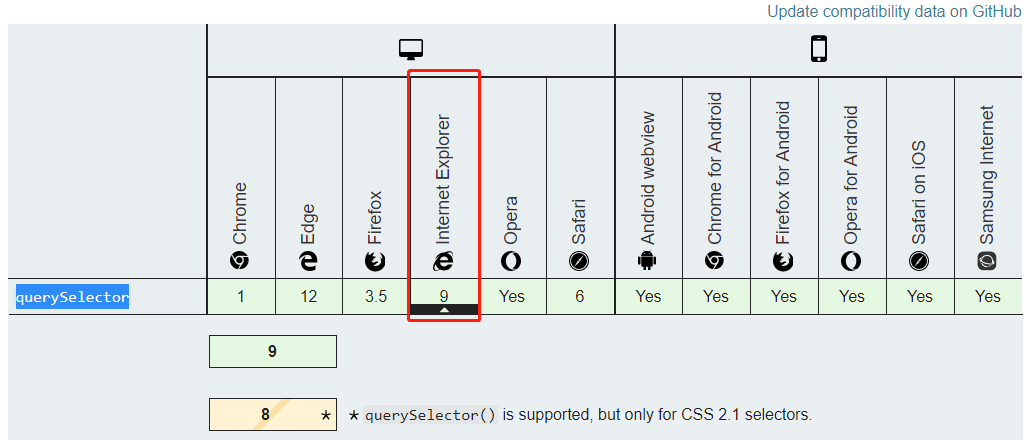
// 但是 jQuery统一了API 做到不同浏览器的兼容

        $('#text).text()

    </script>

每当处理ie的兼容性和其他主流浏览器的兼容性都是让切图仔或设计或后端痛苦不已，当时是真的没有前端这个称呼（那个设计师下周把页面生成好给后端，那个后端找个框架下下周我们这个后台系统要上线用SSH+jqueryUI（又一神器）就行）。

1. 强大的选择器



在document.querySelector，document.querySelectorAll出来前document.getElementById，document.getElementsByClassName，document.getElementsByTagName是我们只能常用的几个选择元素的方式，当然document.getElementById是最快也最节省性能的选择方式，但是如果需要操作一个元素就给元素命名一个id，如果需要二三十个控制元素不起点有语义化的名字，不容易记住，都起语义化的名字又比较费事，而往往这个元素是banner，class会叫banner ，id也叫banner又明显重复，这时候可以用document.getElementByClassName(),但这个方法选取的是HTMLCollection []元素集合（类数组）。因为类名用来控制一类元素的样式，所以在页面中又可以重复，这样我们还需要查看是哪个元素才是需要被操控的，元素增减了还需要变化选取序列。

如果需要选取div.article1下的.title还需要先获取div.article1再选取.title

 <div class="article1">

        <h2 class="title">文章一标题</h2>

    </div>

    <div class="article2">

        <h2 class="title">文章二标题</h2>

    </div>

    <script>

        var article\_1 = document.getElementsByClassName('article1')[0]

        var title\_1 = article\_1.getElementsByClassName('title')[0]

        // title\_1 文章一的标题

        console.log(title\_1.innerHTML)

// 但是 jQuery 挖槽节省了很多代码量

$('.article1 .title').text()

    </script>

1. 隐式迭代

如果要给一类元素绑定事件，JavaScript需要先get到一类元素集合再遍历对每一个元素进行单个事件绑定，增加许多业务无关的冗余代码。

<ul>

        <li class="click-hide">

            <img src="//www.coolz.com.cn/public/img1x/5a13058c32fb3.jpg" alt="">

        </li>

        <li class="click-hide">

            <img src="//www.coolz.com.cn/public/img1x/5a13058c32fb3.jpg" alt="">

        </li>

        <li class="click-hide">

            <img src="//www.coolz.com.cn/public/img1x/5a13058c32fb3.jpg" alt="">

        </li>

    </ul>

    <script>

        var li\_img = document.getElementsByClassName('click-hide')

        for(var i =0,length=li\_img.length;i<length;i++){

            li\_img[i].onclick=function(){

                alert('遍历绑定事件')

            }

        }

        // 但jQuery

        $('.click-hide').click(function(){

            alert('隐式迭代不需要手写遍历代码')

        })

    </script>

1. DOM操作封装

如果要隐藏元素

Javascript 应该写

Element.style.display = ‘none’

jQuery应该写

$(element).hide()

是不是少很多字，哈哈

   <div class="box" style="width: 100px;height: 100px;background: green;">

    </div>

    <button id="change1">change-js</button>

    <button id="change2">change-jQuery</button>

    <button id="reset">reset</button>

    <script>

        var btn = document.getElementById('change1')

        btn.onclick = function () {

            var box = document.getElementsByClassName('box')[0]

            box.style.display = 'none'

        }

        // jQuery

        $('#change2').click(function(){

            $('.box').hide()

        })

        $('#reset').click(function(){

            $('.box').show()

        })

    </script>

1. 链式编程

jQuery的灵魂操作

如果想给article元素添加黑色背景，白色字，然后改title元素字体为red

<div class="article">

        <h2 class="title">文章标题</h2>

        文章内容

    </div>

    <button id="change1">change-js</button>

    <button id="change2">change-jquery</button>

    <button id="reset">reset</button>

    <script>

        var btn = document.getElementById('change1')

        btn.onclick = function () {

            var article = document.getElementsByClassName('article')[0]

            article.style.background = 'black'

            article.style.color = 'white'

            var title = article.getElementsByClassName('title')[0]

            title.style.color="red"

        }

        // jQuery

        $('#change2').click(function(){

            $('.article').css({

                background:'black',

                color:'white'

            }).find('.title').css('color','red')

        })

        $('#reset').click(function(){

            $('.article').css({

                background:'white',

                color:'black'

            }).find('.title').css('color','black')

        })

    </script>

jQuery不需要每次都从元素开始操作，可以接着上次元素的操作结果继续操作。直接省去了相当多的代码量（厉害，怎么想到的）



1. 支持丰富的扩展

最后也是相当重要的，一个工具不论多完美都有其还没有的功能，这就需要自身有便捷的扩展性。

jquery.fn.extend() 函数可以为jQuery扩展一个或多个实例属性和方法(主要用于扩展方法)。

比如我要获取一个元素内所有img标签的src集合，这肯定是jQuery中不会直接提供的方法。只能我们自己加入

<div class="box" style="width: 200px;">

        <img width="100%" alt="" data-v-b24caf1c="" src="//www.coolz.com.cn/public/img1x/5caf329a43dad086505641.png">

        <img width="100%" alt="" data-v-b24caf1c="" src="//www.coolz.com.cn/public/img1x/5b8aa435f1c00.jpg">

        <img width="100%" alt="" data-v-b24caf1c="" src="//www.coolz.com.cn/public/img1x/5c5d4cd3a7adc445704358.jpg">

    </div>

    <script>

        $.fn.extend({

            getImgSrc: function () {

                var src\_arr = []

                this.each(function () {

                    src\_arr.push($(this).attr('src'))

                });

                return src\_arr

            },

        });

        $('.box img').getImgSrc()

    </script>

非常简单，我们就能扩展一个功能在里面。

嗯，一个神器



OK，怀念就到这里。

我为什么要写这样的一系列文章？

俗话说合抱之木，起于累土；九层之台，始于足下；千里之行，生于毫末。不积跬步，何以成江海！好像是这样。

现在框架横行。个个都在倡导敏捷开发，致使我这个小学搬运工，渐渐忘却了js是什么，所以想借着分析jQuery源码的过程（当然已经是老生常谈，老马识途，老当益壮，老无所依，老子天下第一，(⊙o⊙)…），重新熟悉下JavaScript。

下篇开始从头到脚打量一下jquery。