jQuery与Bootstrap

王远亮

2018-6-2

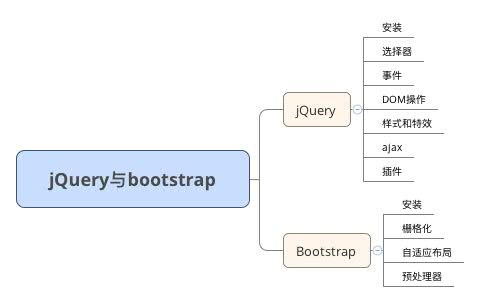
# QQ群：

DJANGO /

强哥



# 课程大纲



# What

**What is jQuery?**

jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript.

jQuery是一个快速，小巧，功能丰富的JavaScript库。 它使诸如

HTML文档遍历和操作，事件处理，动画和Ajax等事情变得更加简单，使用易于使用的可在多种浏览器中工作的API。 结合多功能性和可扩展性，jQuery改变了数百万人编写JavaScript的方式。

# Why

* 原生的API，操作复杂
* 兼容性问题
* 一些常用功能的抽象

# 与原生的比较

原生事件处理操作：

**var** hiddenBox = document.getElementById( “banner-message” ); hiddenBox.addEventListener(“click”, function(event) { // do something.

})

jQuery事件处理：

**var** hiddenBox = $( "#banner-message" ); hiddenBox.on( "click", **function**( event ) { //do something

});

# 原生ajax

|  |
| --- |
| var xhr = new XMLHttpRequest(); xhr.onreadystatechange = function(){  if (xhr.readyState === 4){  if (xhr.status === 200){  console.log(xhr.responseText);  } else { console.error(xhr.statusText);  }  } };  xhr.onerror = function (e) {  console.error(xhr.statusText);  };  xhr.open('GET', '/endpoint', true); xhr.send(null); |

Ajax通信

|  |
| --- |
| $.ajax({ url: "/api/getWeather", data: { zipcode: 97201  },  success: **function**( result ) {  $( "#weather-temp" ).html( "<strong>" + result +  "</strong> degrees" );  } }); |

# How

* 3.1 安装使用
* 3.2 常用api
* 选择器
* 事件
* DOM操作
* 样式和特效
* ajax
* 插件

jQuery 的三大特性

◦ 轻量

◦ CSS3 兼容

◦ 跨浏览器（兼容性）



# 3.1 jQuery 安装使用

jQuery 发布过多个重大版本，并且为了适应不同需求每个主流版本也分别发布了：

◦ 压缩：生产环境，体积小；

◦ 非压缩版：开发测试环境，方便调试和二次开发；

◦ CDN版本：生产环境，保持 新；

jQuery，作为一个 资深的Js框架库，被无数个其他的框架和类库引用和扩展，并不是所有的API都会被用到，所以为了近一步减小自身体积， 新版本还有 slim版本（瘦身）；

# 安装使用

CDN 方式引用；

CDN的全称是Content Delivery Network，即内容分发网络。其目的是通过在现有的Internet中增加一层新的网络架构，将网站的内容发布到 接近用户的网络“边缘”，使用户可以就近取得所需的内容，提高用户访问网站的响应速度。

如果需要时刻保持 新版本或者因为一些限制条件，无法把文件放在本地，则可以使用CDN方式引用；

jQuery 的官方CDN： https://code.jquery.com/ 引用方式：

<script src="https://code.jquery.com/jquery-

3.3.1.slim.js" integrity="sha256-

fNXJFIlca05BIO2Y5zh1xrShK3ME+/lYZ0j+ChxX2DA=" crossorigin=" anonymous"></script>

# 安装使用

在使用任何一个框架（类库）之前，需要注意几点：

1，兼容性，例如：10.0 （含）之前的IE版本，只能使用1.x；

2，依赖关系；例如jQuery-UI 必须要先引用jQuery； 3，确保被引用的资源有效；如果是CDN，注意协议！ jQuery的使用：$()，或者jQuery()

# 3.2 jQuery 选择器

jQuery 选择器兼容CSS1～3所有版本，且还外带自由选择器。用好jQuery的选择器，是**掌握jQuery的必要技能。**



思考：

为什么要加入两个

“

\

”

？



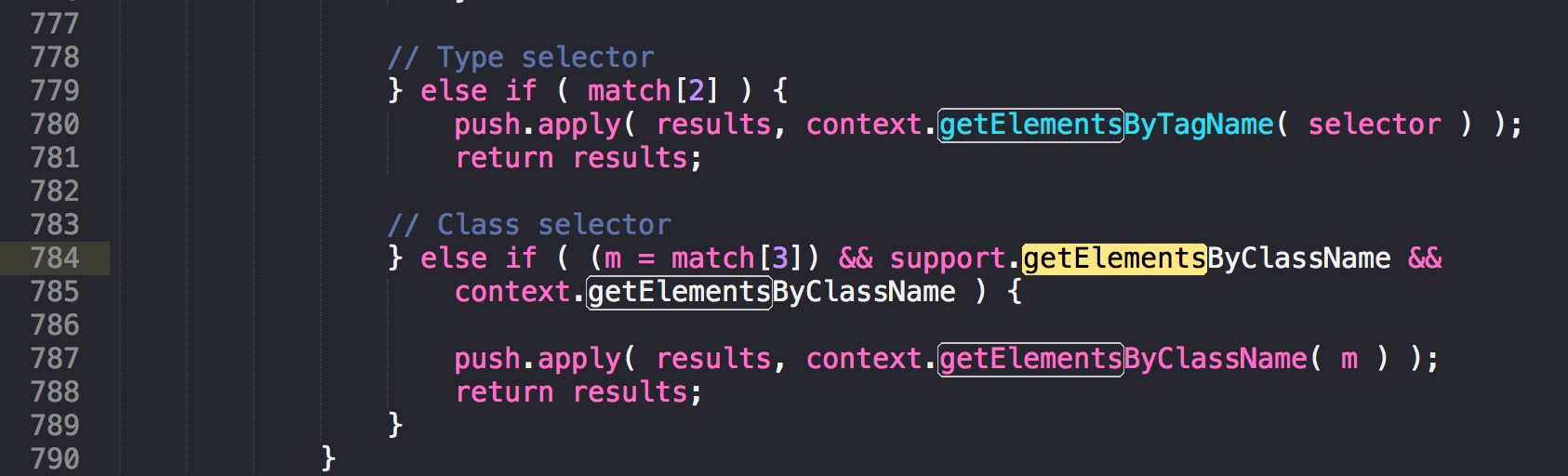
注意：在使用选择器但是遇到特殊字符，例如 !”#$等，需要加入转义符 “\\”。例如：

$(“#foo\\.bar”)

jQuery 3.3.1 未压缩源代码：

◦ 2911行初始化；

◦ 780行～790行附近的Sizzle函数（对象）；



样式类选择：

$(“.classname”)

标签元素选择：

$(“h7”)

Id选择：

|  |
| --- |
|  |
| $(“$(“#id#id”)”) |
|  |

多重选择：

$(“#id , h7 , .classname”)

# 过滤器

在选择器后追加“：”（冒号），是基于选择选中元素再次过滤；例如：

$(“td:eq(2)”)

过滤器主要用于对选中的元素进行过滤，返回一组加入“限定条件的”子元素；主要过滤类型有：

◦ 基于类型的过滤，语法是 [attrName = value]

◦ 基于伪类的过滤:first-child :last-child() 等；

◦ 基于内容的过滤，有： :contains() :empty :has() :parent

◦ 基于是否被显示的过滤，有： :hidden :visible

◦ jQuery开发的过滤器，例如，选中所有文件输入 :file 所有按钮 :button 等

**留意性能！**

选择其中，尽量选择id和元素标签选择器，效率 高，类选择器其次， 差的是伪类和属性选择器；



提示：

一篇关于正则表达式神作：

https://

deerchao.net

/tutorials/regex/

regex.htm



# Core

jQuery 调用方式：

jQuery(selector | element | elementArray | object | selection | html )

jQuery()函数对象有一个别名：$

调用后，返回一个匹配到的带有这个DOM元素的jQuery扩展对象；

# 事件绑定

当某一些特殊过程被用户或者程序触发时，根据需求需要执行一些逻辑过程，这可以称之为“事件”。

例如当用户按下鼠标，则是click事件，在纯JS里可以用

addEventListener方法，针对某一个DOM对象进行绑定操作。在jQuery中，有些事件已经被封装为方法，有些可以通过一些事件绑定函数进行操作。

并且可以设定不同的执行次数、延迟、追加和解除等操作。

# 事件对象

jQuery的事件系统根据W3C标准规范事件对象。 事件对象保证传递给事件处理程序。 来自原始事件的大多数属性都被复制并标准化为新的事件对象。

事件对象在事件被触发时被构造，并且传递给绑定的事件处理函数。

公共事件属性主要有：

◦ currentTarget：当前触发事件的对象

◦ target ：绑定事件的DOM对象

◦ relatedTarget：关联DOM对象

◦ result ：（如果有）返回 后一个事件函数的返回值；

◦ pageX ：鼠标坐标（对于document）

◦ pageY ：鼠标坐标（对于document）

◦ which ：鼠标或者键盘按键

◦ metaKey ：按键的键码

# 事件绑定

bind()：绑定一个或者多个事件，例如：

$(“.classname”).bind(“click”,function(){ … });

delegate()：绑定子节点中，一个或者多个事件；

on()：绑定一个或者多个事件，在子节点中； off()：移除绑定的事件； one()：每一个选择器选中的元素，只相应一次绑定的事件； unbind()：移走对象绑定的特定的事件函数； off()：移走对象绑定的事件；

# 浏览器事件绑定

error() ： 当发生异常时，绑定一个错误处理程序予以执行；

<img alt="Book" id="book">

<script>

$( "#book" ).error(**function**() { alert( "Handler for .error() called." )

}).attr( "src", "missing.png" );

</script>

resize()：当浏览器窗口被调整尺寸时执行，可以通过

window.width或者$(window).width() 获取宽度； scroll()：当发生滚动时；

# 文档加载事件

ready() 和 load()

load是html、图片等文件全部加载完毕执行的方法； ready是加载完成后，界面也已经绘制完成执行的方法；

一般：在网页加载完成后执行一系列的诸如：按钮绑定、异步调用，均可用：

$(document).ready(function(){ … });

# 鼠标事件

click()：单击 contextmenu(): 右键 dblclick()：双击

mousedown()：鼠标按下（一瞬） mouseup()：鼠标松开

mouseenter() 移入 mouseleave() 移出 mousemove() 移动 mouseout() 移出 mouseover()经过其中：mouseleave 阻止冒泡，而 mouseout 则不阻止冒泡

# 键盘事件

keydown()：按下 keypress()：按下松开 keyup()：松开



观察：

这些事件同时被送来的参数，都有一些什么内容？对象类型和

结构又是怎么样的？



# 表单事件

blur()：焦点移走 change()：内容改变 focus()：获得焦点 focusin()：获得焦点（冒泡） focusout()：失去焦点（冒泡） select()：选中（文本） submit()：表单提交

# 表单事件

blur()：焦点移走 change()：内容改变 focus()：获得焦点 focusin()：获得焦点（冒泡） focusout()：失去焦点（冒泡） select()：选中（文本） submit()：表单提交

可以通过jQuery提供的API，对DOM对象的属性、样式和文本进行插入、复制、删除等操作，是诸如动画、UI等第三方插件的基础功能；

本节内容分别有以下几个方面

◦ DOM 操作基础

◦ CSS 属性操作

◦ 对象属性操作

◦ 尺寸和偏移量计算

针对文本节点的包裹操作：

wrap() wrapAll() wrapInner()

以上，jQuery选择器选中一个或者多个DOM对象，在对象之内和在对象之内的文本节点之前，插入元素标签，例如：

针对文本节点的包裹操作：

wrap() wrapAll() wrapInner() 以上，jQuery选择器选中一个或者多个DOM对象，在对象之内和在对象之内的文本节点之前，插入元素标签，例如：

<span>Span Text</span>

<strong>What about me?</strong>

<span>Another One</span>

<script>

$( "span" ).wrap( "<div><div><p><em><b></b></em></p></div></div>" ); </script>

内部插入操作（常用！） append() ：给每一个选中的元素内部追加html或对象； appendTo() ：将选中的对象追加到一个目标对象内部； html() ：读取、修改对象的HTML内容；相当于 innerHTML属性；

prepend() ：给每一个选中的元素内部 前加入html或对象；

prependTo() ：将选中的对象追加到一个目标对象内部 前； text() ：读取、或者修改一个对象内到文本，自动过滤 html标签；外部插入操作，针对同级别的兄弟之间的元素操作； after() ：在选中元素之后，插入对象或者HTML； before() ：在选中元素之前，插入对象或者HTML；

insertAfter() ：将jQuery对象插入到一个或多个对象之后； insertBefore() ：将jQuery对象插入到一个或多个对象之前；移除对象、元素操作 unwap() ：移除被选中的对象下层节点和文本节点上的节点； empty() ：清空被选中节点中所有内容； remove() ：移除被选中的节点；

detach() ：分离被选中的对象，即：从document中移除，放入（返回）内存变量；

replaceAll() ：将jQuery对象替换为选中的元素 replaceWith() ：将选中的元素替换为HTML或对象；

<div class="container">

<div class="inner first">Hello</div>

<div class="inner second">And</div>

<div class="inner third">Goodbye</div>

</div>

<script>

$( "<h2>New heading</h2>" ).replaceAll( ".inner" );

$(".inner" ).replaceWith("<h2>New heading</h2>" ); </script>

<div class="container">

<h2>New heading</h2>

<h2>New heading</h2>

<h2>New heading</h2>

</div>

复制元素：clone()，可以将选中的jQuery对象（DOM）进行一个深度复制（所有子节点以及孙子节点）；并将此返回；

CSS样式的类操作：

addClass() ：添加一个或者多个类给jQuery对象； hasClass() ：验证jQuery对象是否含有某个或者多个样式类； removeClass() ：从jQuery对象中移走某一个样式类； toggleClass() ：为jQuery对象开启或者关闭某一个样式类；

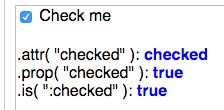
# DOM方法

.get(n) 返回第n个jQuery选中的DOM，注意，这是DOM对象，而非jQuery对象，如果要以jQuery的API针对其操作，需要$(dom)

.index(n) 返回n是当前jQuery中第几个子节点（兄弟排行），两者交互亦可。从0开始计算第一个节点；

# 属性操作

属性操作：有两个参数，第一个参数是属性名，第二个如果有参数则代表是设定，否则代表读取；

attr()：获取选中元素（如果有多个，则只返回第一个）的属性值； prop() ：功能基本同attr，差别是prop方法是jQuery针对HTML的一致性问题而开发的功能，例如：

removeAttr()：移除匹配到每一个元素的的一个或者多个属性； removeProp()：基本同上，常用于移除一些自定义属性；

# 样式操作

jQuery 提供了一系列的API，用于针对CSS的样式（style）进行操作；

css()：设定和读取jQuery对象的style值；例如$(a).css(“color”) height() width()：读取/设定对象的高和宽（css）； outerHeight() outerWidth()：读取、设定对象的全高和宽； offset()：读取对象在文档中的坐标位置，返回的对象有top和left 书属性；

position()：返回被选中第一个的对象的相对父级坐标；top，left scrollLeft() scrollTop()：读取和设置向右、上滚动的量（像素）

# 基本特效

hide() 隐藏匹配的节点元素 show() 显示匹配的节点元素

toggle() 开或关匹配的节点元素

jQuery的特效方法，基本都会支持几个参数，分别是

duration：字符串或者数字，代表特效的持续事件，如果是字符串，可以用slow或者fast这两个单词，如果是数字，则代表毫秒数； callback：当特效完成后执行的回调函数；

# 特效

fadeIn() ：渐显 fadeout() ：渐隐 fadeTo() ：显示到特定的透明度 fadeToggle() ：开关方式的渐隐渐显 slideDown() ：下翻 slideUp() ：上翻 slideToggle() ：开关方式上下翻

# 自定义动画

animate() ：以CSS的animate方式定义特效； delay() ：延时，参数是毫秒；

finish() ：结束所有动画，直接跳到动画效果完成的状态；

clearQueue() ：结束后续动画，移除序列，停止在当前状态； stop() ：停止动画；

其中 clearQueue 和 stop 用途略不一样；

# 自定义动画

$().animate(properties,options)

properties ： css 属性和值的键值对对象； options：动画附加方法对象参数

|  |
| --- |
| $( "#clickme" ).click(**function**() {  $( "#book" ).animate({ opacity: 0.25, left: "+=50", height: "toggle"  }, 5000, **function**() {  *// Animation complete.*  });  }); |

http://api.jquery.com/animate/

jQuery 将 xmlhttpRequest对象进行了扩展，通过不同的简写和丰富的API实现了Ajax的全部功能且又能够被方便的调用；

Ajax的工具类（方法）有：

jQuery.param() ：jQuery的全局函数，将JSON对象序列化为GET 方式调用的字符串（自动encode）；

serialize()：将选中的对象中的表单类元素(input select等）序列化为GET方式的字符串；

serializeArray()：将选中的对象中的表单类元素，返回分别以 name和value组成键值对对象的数组；

全局Ajax事件处理具柄，需要使用document作为jQuery的AJax 的全局处理对象

$(document).ajaxComplete()：完成请求处理

$(document).ajaxError()：请求错误处理

$(document).ajaxSend()：发送请求之前的处理

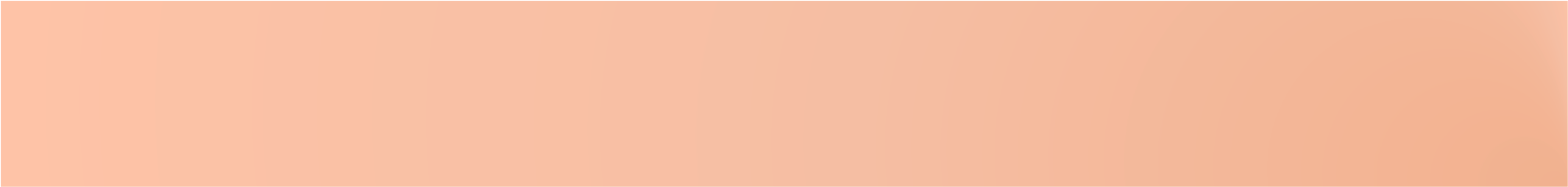
$(document).ajaxStart()：调用发送

$(document).ajaxStop()：结束调用

$(document).ajaxSuccess()：成功完成调用

观察：

以上这些事件被触发的先后顺序是怎样的



|  |
| --- |
| <!DOCTYPE html>  <html>  <body>  <script src="https://code.jquery.com/jquery-1.10.2.js"></script>  <button id="a">a</button>  <ul id="msg"></ul>  <script>  $(document).ajaxComplete(function(event, request, settings) {  $("#msg").append("<li>Request Complete.</li>");  });  $(document).ajaxSend(function(event, request, settings) {  $("#msg").append("<li>Starting request at " + settings.url + "</li>");  });  $(document).ajaxError(function(event, request, settings) { console.log(event);  $("#msg").append("<li>Error requesting page " + settings.url + "</li>");  });  $(document).ajaxStart(function(event, request, settings) {  $("#msg").append("<li>start</li>");  });  $(document).ajaxStop(function(event, request, settings) {  $("#msg").append("<li>stop</li>");  });  $(document).ajaxSuccess(function(event, request, settings) {  $("#msg").append("<li>Successful Request!</li>");  });  $("#a").click(function() {  $.getJSON("http://118.25.3.51/test.json",function(data){console.log(data);})  });  </script> </body>  </html> |

jQuery Ajax调用，底层接口

jQuery. ajaxPrefilter()：处理、修改一个每次请求之后收到返回数据之前的执行过程；

jQuery.ajaxSetup()：设定一些默认参数，使得真正调用ajax时可以减少代码量；

jQuery.ajaxTransport()：为每次传输定制特定的处理事务，与 jqXHR的接口对接实现完成回掉；

jQuery Ajax调用，底层接口

jQuery.ajax()，可以定制相当丰富的参数，此处指做简单介绍；例子：

|  |
| --- |
| **var** menuId = $( "ul.nav" ).first().attr( "id" ); **var** request = $.ajax({ url: "script.php",  method: "POST", data: { id : menuId }, dataType: "html"  });  request.done(**function**( msg ) { $( "#log" ).html( msg );  });  request.fail(**function**( jqXHR, textStatus ) { alert( "Request failed: " + textStatus ); }); |

jQuery Ajax 封装的短方法；在ajax调用中，大多数情况只需要通过简单的调用即可实现需求，并不需要进行复杂的配置，jQuery将很多调用进行了封装，方便开发者使用的API；

jQuery.get(url, params,success,dataType) get方式调用，参数

如果是JSON将会被自动转义；

jQuery.getJSON() 以 HTTP GET方式加载一个JSON资源； jQuery.getScript() 以HTTP GET方式加载一个Js文件； jQuery.load() 以HTTP GET方式加载资源； jQuery.post() 以HTTP POST方法发送请求；

为了提高开发效率，jQuery提供了一系列的全局函数，非常小巧但功能强大；

jQuery.each()：遍历数组或者对象； jQuery.extend()：深度合并多个对象为一个新的对象；

jQuery.globalEval()：全局范围内执行js代码（字符串），与eval不同的是，作用域是在全局中，那怕执行是在内部函数里；

jQuery.grep()：类似linux的管道命令，按照一定逻辑过滤数组的元素； jQuery.inArray()：查找数组中是否存在某个元素，如果存在返回index； jQuery.isArray()：判断参数是否为数组类型 jQuery.isEmptyObject()：判断参数是否为空对象； jQuery.isFunction()：判断参数是否为一个方法（函数） jQuery.isNumberic()：判断是否为数字，可以是字符串的16进制 jQuery.isPlainObject()：判断是否为（jQuery特有）简单对象 jQuery.makeArray()：将jQuery对象数组转换为js数组； jQuery.map()：同each，但会接受一个返回，变成一个新数组 jQuery.merge()：连接2个数组； jQuery.noop()：验证是否为空函数

jQuery.now()：等同 (new Date).getTime()

jQuery.parseHTML()：将字符串转换为DOM对象数组（含节点） jQuery.parseJSON()：JSON字符串转换为JSON对象； jQuery.trim()：去字符串掉前后的空格； jQuery.proxy()：在执行事件时修改对象的上下文关系，绑定到DOM对象上；

jQuery.type()：返回参数的类型（class），function，date等（比 typeof 功能更强）

# 插件

jQuery之所以能够如此普及，除了其自身的优秀之外，它丰富的插件和生态系统，也为它的流行和推广起到了很大的作用。几乎你想到的所有应用场景，都会找到相对应的jquery插件。

* 轮播图 swiper
* 省市区选择
* 日期选择
* 上传图片
* …

# 实践

动手开发：猜牌游戏；特别要求：纯JS；

1，HTML5 界面设计

2，CSS 样式设计

3，程序逻辑

1. 定义事务；
2. 事务拆解为过程，定义全局标志；
3. 编写过程代码；
4. 运行、测试与微调；

What is Bootstrap?

Bootstrap是Twitter推出的一个用于前端开发的开源工具包。它由 Twitter的设计师Mark Otto和Jacob Thornton合作开发，是一个 CSS/HTML框架。目前，Bootstrap新版本为4.0 。本课程围绕

新版讲解；

关键词：

◦ 栅格化（ 1.0）

◦ 自适应布局（又被称为响应式布局）（2.0）

◦ 移动版优化（优先）（3.0） ◦ 定制主题（4.0）

主流版本：

◦ 2.3.2

◦ 3.3.7

◦ 4.0

# why

* 统一样式、风格，代码复用
* 自适应布局
* 简单样式不依赖设计

# how

* 3.1 bootstrap的安装

* 3.2 样式
* 栅格布局
* 自适应

* 3.3 了解SASS、LESS、Stylus

# 3.1 Bootstrap 安装

bootstrap 框架有JS和CSS文件；可以根据需要下载对应的版本，也可以直接应用CDN；如果项目用了bootstrap的“组件”和“实用工具”，则还需要引用jQuery作为底层的JS支持类库。

如果下载源码进行二次开发，则需要一些其他工具进行编译（转码）；

Bootstrap也支持主流的包管理工具，一条命令即可完成下载、项目引用和依赖库下载，如npm(node) gem(ruby) composer和 nuget(VS)

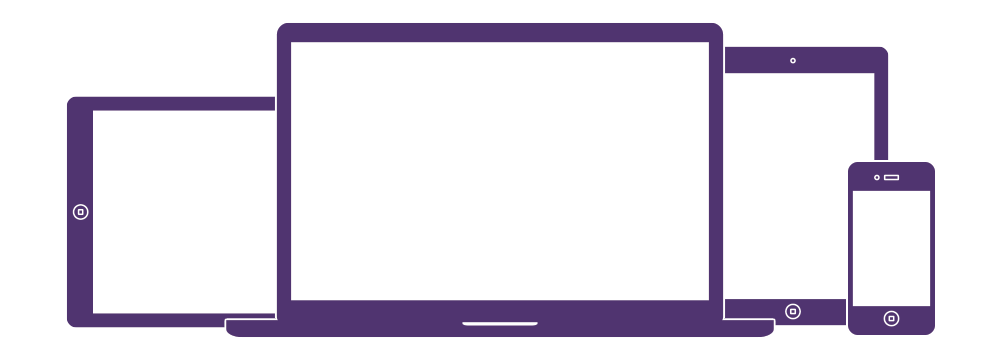
# Bootstrap 4.0

bootstrap 框架有JS和CSS文件；可以根据需要下载对应的版本，也可以直接应用CDN；如果项目用了bootstrap的“组件”和“实用工具”，则还需要引用jQuery和popper.js（弹出提示）作为底层的JS支持类库。

如果下载源码进行二次开发，则需要一些其他工具进行编译（转码）；

Bootstrap也支持主流的包管理工具，一条命令即可完成下载、项目引用和依赖库下载，如npm(node) gem(ruby) composer和 nuget(VS)

# 3.2 栅格布局和自适应



# 模拟多尺寸

使用chrome 开发者工具模拟多尺寸

# 栅格布局

使用bs框架，在布局方面有两个关键概念：**容器**和**栅格**

**容器**是基础的布局元素；有响应式、固定宽度和流式宽度（fluidwidth)；

其中响应式的宽度临界点（断点）有：576px，768px，992px和

1200px，分别对应个尺寸的界面；分别对应的变量名为： xs,sm,md,lg,xl

**栅格系统**是bs框架 重要的一个功能，栅格系统同样也具有一定的 “弹性”，当到达一个宽度临界点事，可以根据程序来转换页面的样式；在bs框架中，一行被分为12列，

# 布局

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Extra small <576px | Small ≥576px | Medium ≥768px | Large ≥992px |  |
| Max  container width | None (auto) | 540px | 720px | 960px | 1140px |
| Class prefix | .col- | .col-sm- | .col-md- | .col-lg- | .col-xl- |
| # of  columns | 12 | | |  |  |
| Gutter width | 30px (15px on each side of a column) | | |  |  |
| Nestable | Yes | | |  |  |
| Column ordering | Yes | | |  |  |

# 布局

栅格系统的主要布局方式（功能和原则）有：

◦ 自动列宽，一行可以定义任意数量列，宽度均分

◦ （一行容器中的）多行多列等宽，分行用 w-100 类

◦ 相应类样式 col-\*-n

◦ flex方式的纵横对齐；

◦ 当列超过12时，自动换行

◦ 列内可以嵌套

◦ 图像支持响应式

# 内容

如果说jQuery解决了浏览器之间javascript的兼容问题，那么 bootstrap解决了浏览器之间显示的差异性问题。

# Bootstrap 内容

bs重新定义了一些HTML元素的默认样式，例如 table 、 rem,font-size 列表 h1～h7等等，根据系统环境还定义字体，源码：

$font-family-sans-serif:

// Safari for OS X and iOS (San Francisco)

-apple-system,

// Chrome < 56 for OS X (San Francisco)

BlinkMacSystemFont,

// Windows

"Segoe UI",

// Android

"Roboto",

// Basic web fallback

"Helvetica Neue", Arial, sans-serif,

// Emoji fonts

"Apple Color Emoji", "Segoe UI Emoji", "Segoe UI Symbol" !default;

# 3.3 Bootstrap 简介 SASS

SASS 与 CSS

由于CSS没有变量也没有条件判断，所以处理一些复杂状态比较麻烦，有人就为CSS加入了变成元素，这类程序统称为“CSS预处理器”。SASS是CSS的预处理器之一，其他还有LESS，SCSS等；早先，SASS需要经过便已处理称为普通的CSS后，才能被使用，处理这个语法的是Ruby语言 ;

# Bootstrap 简介SASS

SASS的变量，以$开头，如：

$blue:#1234e6; $side:left;

可以将变量嵌入到css属性或者变量中，如：

color:$blue; border-#{side}-radius: 5px;

SASS 可以在代码中使用公式，如：

margin: (14px/2); right:$var \* 10%;

也可以使用前套，如将 div h1 {…} 可以写为：

div { h1{ …} }

甚至是属性也可以前套（CSS简写属性）

# Bootstrap 简介 SASS

SASS 支持继承，如

|  |
| --- |
| .class1 { border: 1px solid #ddd;  }  .class2 {  @extend .class1;  font-size:12px;  } |

代码块重用（Bootstrap 4的重要特性之一），甚至是可以带有入口参数和缺省参数

|  |
| --- |
| @mixin myDefine { border: 1px solid #ddd；  }  .class2 {  @include myDefine;  } |

# 下午练习

bootstrap 案例解读

轮播：https://getbootstrap.com/docs/4.0/examples/carousel/

自己动手做一个轮播图和弹出框

按钮使用bootstrap提供的样式

下课

