1. 基础
   1. Jcrop

**1.参考**

官网：http://jcrop.org/。

中文使用文档和实例：http://code.ciaoca.com/jquery/jcrop/。

**2.简介**

图片裁剪工具，支持ie6。

* 1. jCanvas

**1.参考**

中文使用参考：http://www.2cto.com/kf/201208/145285.html。

**2.简介**

装了一些绘制图形的方法，只需编写几行代码即可生成图形。

* 1. jquery.lazyload.js

延迟加载长页面中的图片. 在浏览器可视区域外的图片不会被载入, 直到用户将页面滚动到它们所在的位置. 这与图片预加载的处理方式正好是相反的.

http://www.cnblogs.com/hooray/archive/2011/10/15/2212922.html

* 1. floatingAd.js

漂浮广告插件

* 1. Google Code Prettify

（1）下载地址：https://github.com/google/code-prettify。

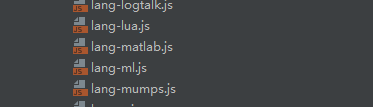
（2）使用pre/code包裹代码，并使用prettyprint和linenums=0（行号从什么数字开始）样式类。



（3）需要引入css和js文件。



（4）会针对不同语言对高亮代码语法，默认html/css/js等，如果其他语法，需要引入相应的js文件。



（5）需要使用body的onload事件。

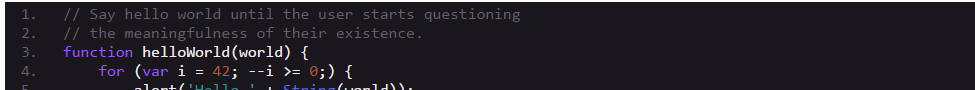


（6）默认样式比较难看，可以从https://jmblog.github.io/color-themes-for-google-code-prettify/下载好看的样式。

默认样式



下载的样式



* 1. jsRender
     1. 参考
* 官网：http://www.jsviews.com/。
* 中文参考：http://www.jianshu.com/p/3151d2256410。
* 中文参考：https://msdn.microsoft.com/zh-cn/magazine/hh975379.aspx。
  + 1. 使用
* 引入jsrender.js。
* 定义模板。



* 准备好要显示的数据json对象。
* 编译成html：document.getElementById(模板id).render(data)。
* 将html显示。
  + 1. 概念
* jsRender3要素：模板、容器和数据。
* 基本的jsRender标签.

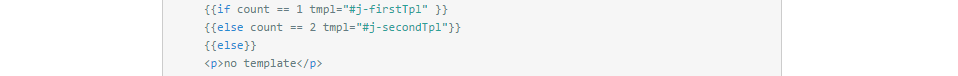
|  |  |
| --- | --- |
| {{: pathOrExpr}} | 值类型（未被html编码），里面可以有运算符，如：{{:movie.releaseYear < 2000}} |
| {{> pathOrExpr}}或{{html: pathOrExpr}} | html编码后的值。 |
| {{\* mycode}} | 代码。 |

* 注释<!-- -->是不起作用，要用{{!-- --}}。
* 可以使用的运算符有：

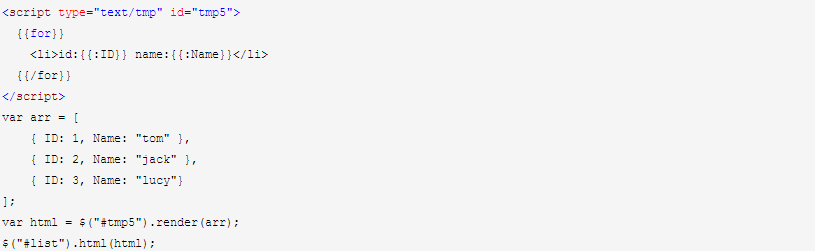
|  |  |  |
| --- | --- | --- |
| **表达式** | **举例** | **注释** |
| || | {{ :a || b }} | 或 |
| && | {{ :a && b }} | 且 |
| ! | {{ :!a }} | 非 |
| <=和>=和<和> | {{ :a <= b }} | 比较 |
| ===和!== | {{ :a === b }} | 等于和不等于 |

* + 1. 逻辑判断和循环
* if-else

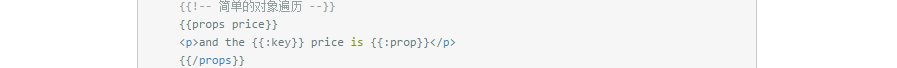
jsrender并没有elseif这样的写法。



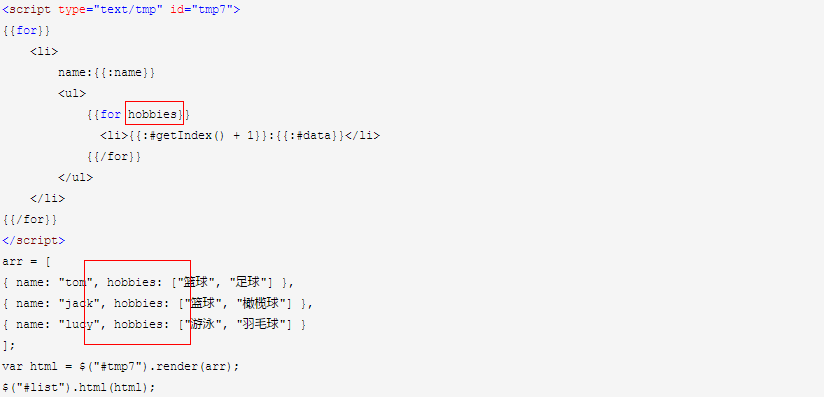
* for



#index，#data为内置的jsrender关键字。#index获取当前的索引。#data获取整个数据。#parent获取父模板。可以使用props可以获取到key/prop值。

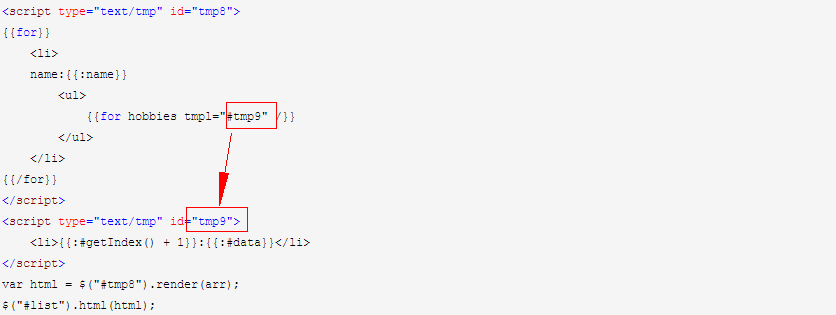


* 嵌套for

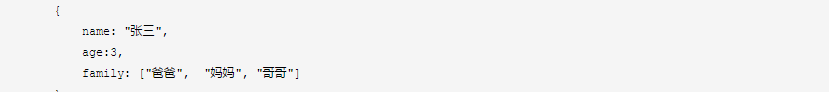


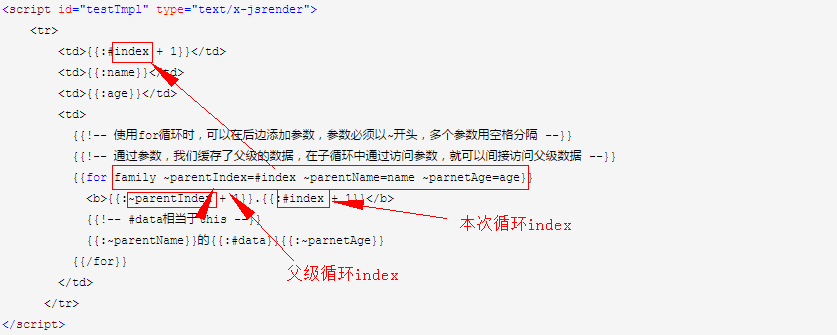
* 分离for

如果for的逻辑比较复杂，嵌套的for就会让我们的模板变得复杂，维护难度加大；我们可以将for分离，以上面的例子，可以将for放到一个新的模板，然后通过 tmpl属性指定。



嵌套循环使用参数访问父级数据。



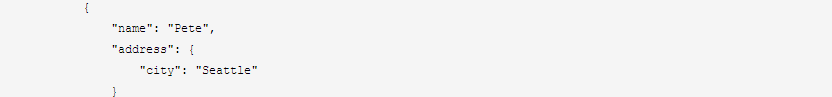


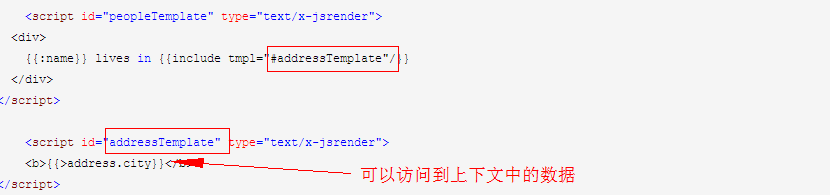
* + 1. 分离视图和逻辑

视图最好就是简单的标签，而逻辑都写在js里。

* + - 1. 组合模板

语法：include tmpl="模板id"。在一个模板中引入另一个模板，并用上下文的数据来渲染template。





* + - 1. 自定义标签
         1. views.tags

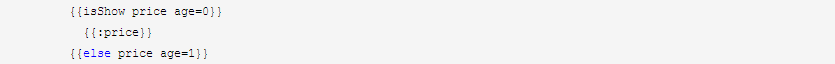
$.views.tags中，可以使用render处理参数或者属性、使用template设置模板。在render中可以使用this.tagCtx.args/this.tagCtx.props/this.tagCtx.view.data获取当前视图层数据。



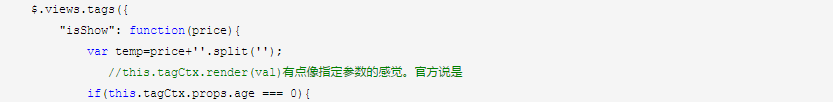
如果不需要模板，只定义一个方法即可。



* 视图：{{"标签名称" 标签参数 附加参数}}。

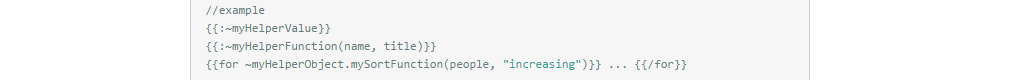


* 逻辑：$.views.tags({"标签名称":function(参数){this.tagCtx.props.prefix附加参数}})

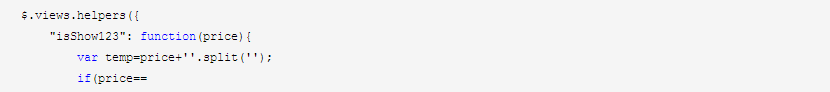


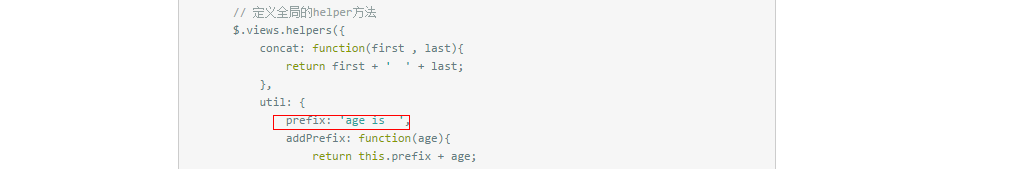
* + - * 1. views.helpers
* 视图：{{~“标签名称”(附加参数)}}。使用helper方法是以~为前缀作为方法的标示，加上方法名，然后把值当参数传进去。





* 全局逻辑：$.views.helpers（{“标签名称”：function（参数）{}}）





* 局部逻辑。



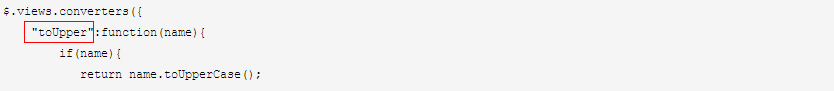
* + - 1. 转换器

转换器可以对输出结果进行处理或格式化。语法：

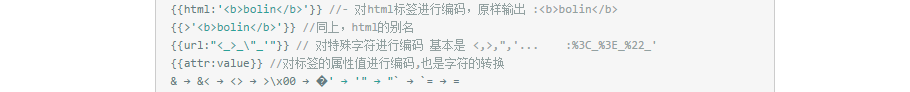
* 视图：{{"转化器名称":参数}}



* 逻辑：$.views.converters({"转换器名称":function(参数){...}})



jsrender本身自带了三个转换器。



* + 1. API
       1. $.templates()

用来注册或者编译模板的。

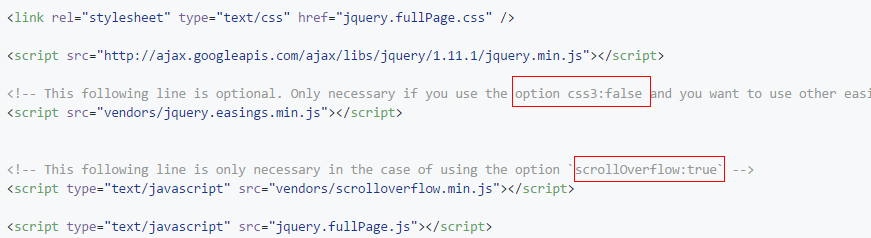
* 可以将把html字符串编译成模板。



* 可以给模板定义名称。



* 也可以把html字符串单独写在script标签中，然后根据id来获取。
  + - 1. render()
      2. $.views.helpers
      3. $.views.converters
  1. fullPage
     1. 参考
* github地址：https://github.com/alvarotrigo/fullPage.js。
* http://www.cnblogs.com/EnSnail/p/5880385.html。
* http://www.uedsc.com/fullpage.html。
  + 1. 简介
* 支持IE8+及其他现代浏览器。
  + 1. 使用
* 引入文件。在不使用css3可引入jquery.easings.js实现动画。在需要使用scrollOverflow（内容超过满屏后是否显示滚动条）引入scrolloverflow.min.js。



* html结构。每个页面默认均应有section类，页面由容器包裹，自定义。



* 默认第1个section作为homePage。可以给其他section添加active作为第1页。



* 每个section内部可以使用.slide类作为section内部的幻灯片。

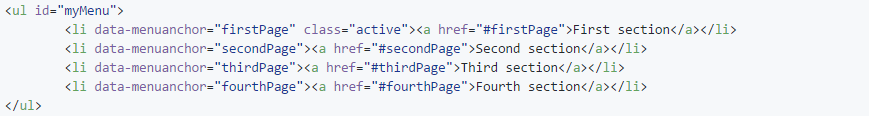


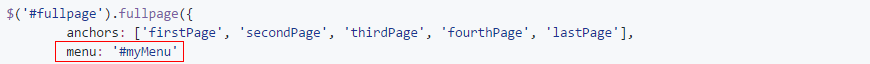
* 初始化，选择容器作为插件元素。

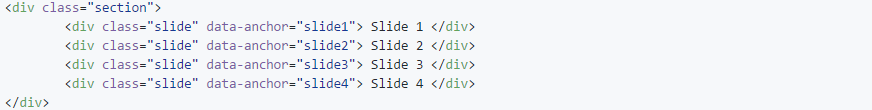


* + 1. 选项
       1. Navigation
* menu/anchors/lockAnchors

关联menu和section或slide。使用配置项anchors或直接使用data-anchor给每个section设定标识。再使用data-menuanchor对menu设置同样标识即可关联。甚至，还可以对slide设置data-anchor关联slide和menu。







当滚动setions时，会设置menu下的li为active。menu默认值false。lockAnchors控制a标签是否继续可用。如果可用，则不能再能滚动页面，通常用于和其他插件结合使用。

* navigation/navigationPosition/navigationColor/navigationTooltips

navigation控制是否显示小圆点项目导航，默认为false。navigationPosition项目导航的位置，可选left或right。navigationTooltips项目导航的tip，如['firstSlide', 'secondSlide']，默认为[]。

* slidesNavigation/slidesNavPosition

slidesNavigation控制是否显示每个slide的小圆点项目导航，默认为false。slidesNavPosition左右滑块的项目导航的位置，可选top或bottom。

* + - 1. Scrolling
* css3/scrollingSpeed/autoScrolling

是否使用CSS3-transforms控制滚动，css3默认值true。scrollingSpeed默认值700。autoScrolling默认值true。

* fitToSection/fitToSectionDelay

是否让section填满整个视窗。fitToSection默认值true。fitToSectionDelay默认值1000。

* scrollBar

是否显示页面右侧的滚动条，默认值false。

* easing/easingcss3

前者为jquery.easings动画方式，后者为css3动画方式。

* loopBottom/loopTop/loopHorizontal

滚动到最后/第1页是否反向滚动，默认值都是false。loopHorizontal为slides是否反向滚动，默认值true。

* normalScrollElements/normalScrollElementTouchThreshold

当鼠标在元素上方滚动时，只会滚动该元素内部的内容，不会滚动屏幕，默认值null，可设置为jquery选择器，如'#element1,.element2'，不能是section或slide。normalScrollElementTouchThreshold控制手机滑动节点层次。

* scrollOverflow/scrollOverflowOptions/bigSectionsDestination

内容超过满屏后是否显示滚动条，需要引入scrolloverflow.min.js，不兼容ie8，默认值false。scrollOverflowOptions为对应的配置项。

bigSectionsDestination控制当section尺寸大于视窗，开始显示位置，可以为top/bottom/null，默认top。

* touchSensitivity

手机端滑动多少像素才翻屏。

* + - 1. Accessibility
* keyboardScrolling

键盘控制滚动。

* animateAnchor

设置了anchor时滚动到目标页面是否动画滚动或者直接切换页面，默认true。

* recordHistory

将section/slide浏览保存到历史记录中。

* + - 1. Design
* controlArrows

是否现在slide左右滑动箭头，默认true。

* verticalCentered

section内容是否垂直居中。

* sectionsColor

数组，section背景颜色。

* paddingTop/paddingBottom

section的padding。

* fixedElements
* responsiveWidth/responsiveHeight
  + - 1. Custom selectors
* sectionSelector/slideSelector

自定义section/slide的css类。

* + - 1. 其他
* lazyLoading

要使用需要将data-src作为数据源，默认true。

* + 1. 回调函数

|  |  |
| --- | --- |
| **名称** | **说明** |
| afterRender | 页面结构生成后的回调函数，或者说页面初始化完成后的回调函数。 |
| afterLoad | 滚动到某一屏后的回调函数，接收anchorLink和index两个参数，anchorLink是锚链接的名称，index是序号，从1开始计算。 |
| onLeave | 滚动前的回调函数，接收index、nextIndex和direction3个参数：index是离开的“页面”的序号，从1开始计算。nextIndex是滚动到的“页面”的序号，从1开始计算。direction判断往上滚动还是往下滚动，值是up或down。 |
| afterSlideLoad | 滚动到某一水平滑块后的回调函数，与afterLoad类似，接收anchorLink、index、slideIndex、direction4个参数。 |
| onSlideLeave | 某一水平滑块滚动前的回调函数，与onLeave类似，接收anchorLink、index、slideIndex、direction4个参数。 |

* + 1. 方法

|  |  |
| --- | --- |
| **名称** | **说明** |
| moveSectionUp | 向上滚动 |
| moveSectionDown | 向下滚动 |
| moveTo(section,slide) | 滚动到 |
| moveSlideRight | slide向右滚动 |
| moveSlideLeft | slide向左滚动 |
| setAutoScrolling | 设置页面滚动方式，设置为true时自动滚动 |
| setAllowScrolling | 添加或删除鼠标滚轮/触控板控制 |

使用方法如：



* 1. waypoints
     1. 参考
* 官网：http://imakewebthings.com/waypoints/api/previous/。
  + 1. 简介

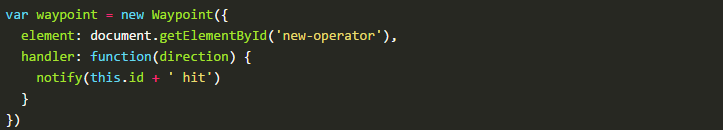
实现在滚动、调整元素大小时，触发事件等。可用于jquery、zepto，也可以不依赖任何框架使用（版本也分为3种，下载是否需要下载对应的版本？）。

* + 1. 使用
       1. 初始化

2中初始化方法，均返回waypoints对象数组。

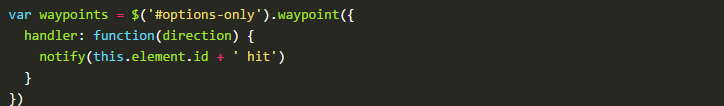


* + - * 1. 构造函数方式

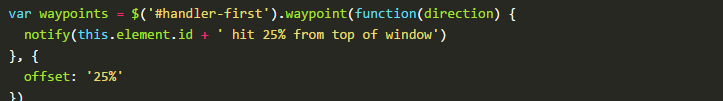


element和handler为必须配置项。

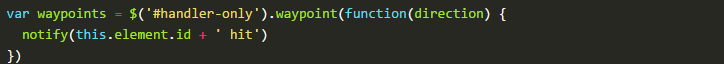
* + - * 1. 插件方式
* 入参为1个配置项对象。



* handler函数和配置项对象。



* 也可以只是handler函数。



* + - 1. 配置项参数
* enabled

是否启用，默认true。

* context

waypoints对象的上层元素，默认值为window。

* handler

操作句柄。

* offset

触发事件的位置，可以为数字（px）、百分比、函数（返回px）、bottom-in-view、right-in-view。

* horizontal

可以为true和false，默认为false，即默认事件为垂直事件。

* continuous

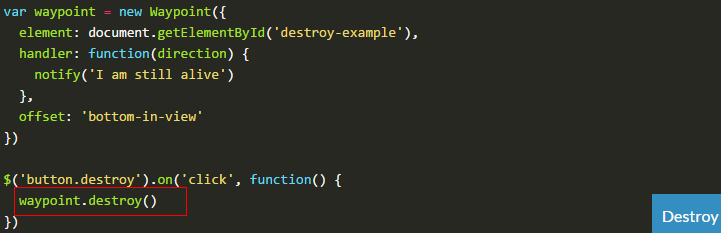
当1个滚动或resize事件触发有多个waypoints对象时，是否按顺序连续显示，默认值为true，连续显示。如果设置为false时，则会根据滚动快慢显示部分。

* group

将多个waypoints对象分组。当滚动到设置值时，不会立即触发事件函数，而是将该waypoints对象注册到其对应的分组中，最后按设定的waypoints对象顺序触发事件函数。可以使用previous、next等顺序方法。

* + - 1. 方法
* destroy

销毁waypoints对象事件，之后便不会再被使用。



注意，此时的waypoint是1个waypoint对象，而非数组，数组没法直接用destory等方法，可以用静态方法。

* disable/enable

启用、关闭。

* next/next

获取1个分组内当前触发事件的waypoints的下一个/上一个分组中的对象。

* 构造函数方法

destroyALL/disableAll/enableAll/refreshAll/viewportHeight/viewportWidth。最后2个为获取视窗高度/宽度。

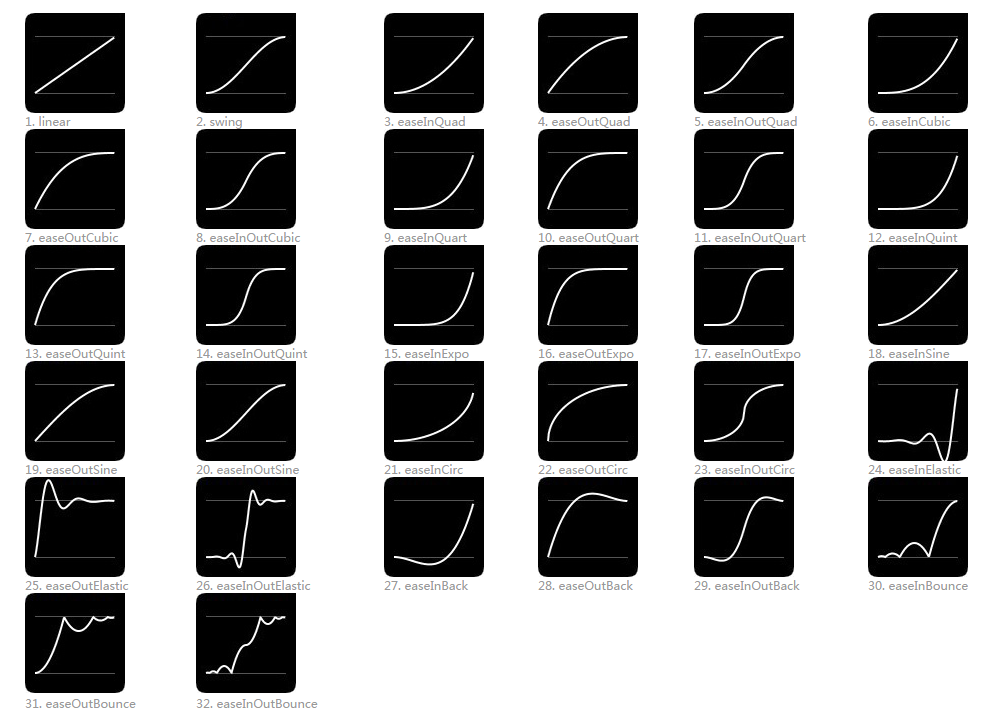


* group对象方法

first/last。

* 1. jQuery.easing

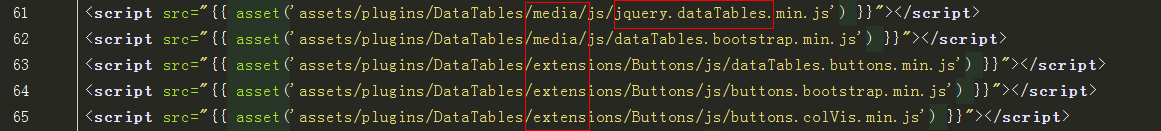
用于jquery动画函数.animate(properties[,duration][,easing][,complete])。有32种效果：



1. 表格和图标
   1. DataTables
      1. 使用
         1. 简单使用

**1.引入文件**

（1）直接引入：



（2）es6中（注意大小写）：





不过仍然依赖jquery，源码中有：



**2.设置数据源**

（1）有两种处理模式：客户端处理和服务器端处理。前者不与服务端交互，而后者无论是排序、分页、筛选等所有操作都要和服务端交互。

（2）有3种指定数据源的方式：

1）在js中指定能访问到的数据源，即把数据指定到data配置项中：



2）DOM，即渲染已经存在的table。DataTables初始化后，会（优先）自动检查表格中的数据。此时，使用data和ajax传递的数据将不会被显示（表头优先使用dom中的thead）。

3）ajax：通过制定url作为数据源（服务端方式）。

**3.数据源格式**

（1）数据源必须是数组格式。数据源中的每1项就是数据项，对应表中1行。

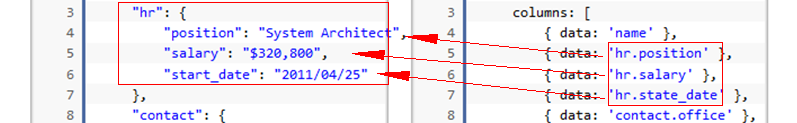
（2）数据项则可以为数组、对象、实例3种格式。

1）数组：此时数组中的元素数量必须等于表中的列数，在columns.data中使用数组索引对应。



2）对象：在columns.data中使用属性名对应数据。



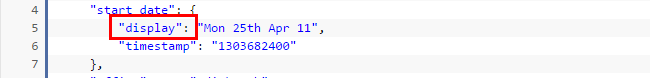


3）实例：即实例化1个js构造函数。

**4.正交数据**

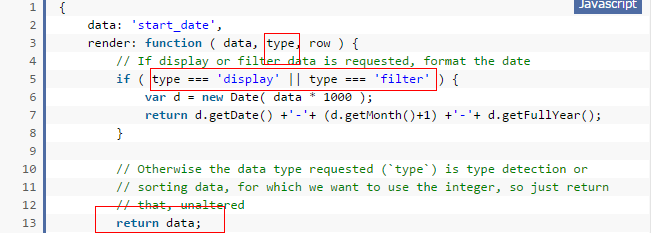
DataTables提供展示（display）、排序（sort）、过滤（filter）、类型检测（type）4种角度用于不同情形数据处理。默认情况，DataTables使用同一种数据源提供给4种操作。可以在columns.render选项中修改。

（1）如果使用1个字面量对象作为render配置项，必须指定\_作为4类正交操作的数据源（至少作为展示）。





（2）也可以使用匿名函数作为render，根据函数的type参数返回数据（匿名函数必须有返回数据，否则对应操作没有数据）。



* + - 1. 客户端模式

就是通过js和dom设置数据源的方式。

* + - 1. 服务端模式

即由服务端处理排序、分页、过滤等操作。每次绘制表格的时候，DataTables会给服务器发送一个请求，然后由服务器返回数据。如果要使用服务端模式，需要设置配置项[serverSide](http://datatables.club/reference/option/serverSide.html)为true，然后设置ajax配置项。



**1.ajax配置项参数类型**

可以为3种类型：

（1）string，指定获取数据的url。



（2）object，用法同jQuery.ajax，见后“附加数据”。



（3）function，见后。

**2.附加数据ajax.data**

当ajax配置项为object时，基本等同于jQuery.ajax。不同的地方：1）data还可以是function。2）有额外属性ajax.dataSrc，使用见ajax.dataSrc。

[ajax.data](http://datatables.club/reference/option/ajax.data.html)如果为object，将作为额外的参数发送到服务器。



还可以是function，function接收的参数为Datatables内部构造的数据，结构如下（和ajax使用function时，传入function中的data参数一样）。



在这个function中，可以对Datatables内部构造的数据修改，然后发送到服务器。

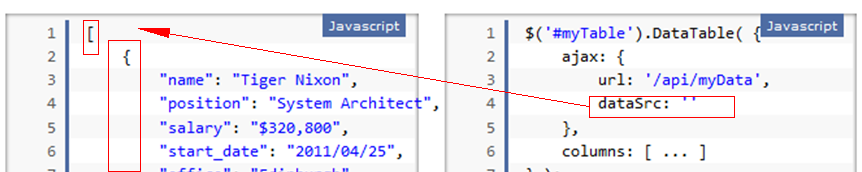


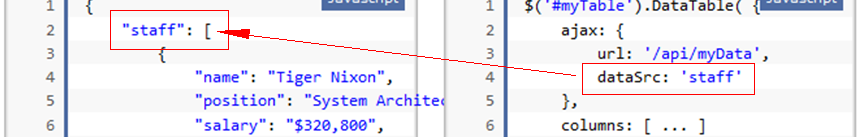
**3.获取服务端返回的数据**

（1）返回的数据最好是json格式，否则需要额外处理。

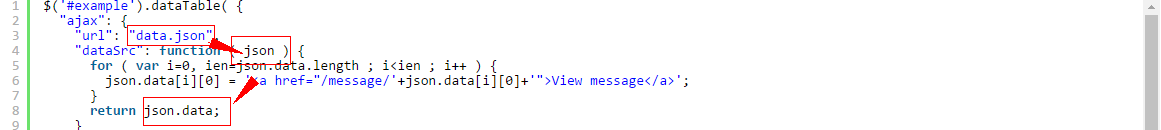
（2）服务器端返回的数据，如果是简单数据项（就是json格式，且每1个数据项都和columns对应），什么都不用处理，datatables会自动将数据渲染处理。

（3）如果返回的数据不是1个简单的数据项，那么就需要用ajax.dataSrc选项指定数据源在返回数据中的位置。此时[ajax.dataSrc](http://datatables.club/reference/option/ajax.dataSrc.html)为string，如下：





[ajax.dataSrc](http://datatables.club/reference/option/ajax.dataSrc.html)还可以为function，function的参数就是服务器返回的数据，可以在里面处理，返回的数据作为数据源。



**4.ajax为function情况**

（1）不使用datatables自带的ajax，而是需要自己在function中实现。

（2）基本上等同于上面ajax.data和[ajax.dataSrc](http://datatables.club/reference/option/ajax.dataSrc.html)功能之和。可以在其中自定义一些事情（分页不是，那个是datatables自己实现的），如spinner等。



接受3个参数：

1）data：发送给服务器的参数，和[ajax.data](http://datatables.club/reference/option/ajax.data.html)一样。



2）callback：必须执行，且要将处理后的数据作为入参传递给callback，这样做Datatables才能获取到数据。

3）setting：配置。

**5.**[**ajax.data**](http://datatables.club/reference/option/ajax.data.html)**和ajax为function时获取到的数据**

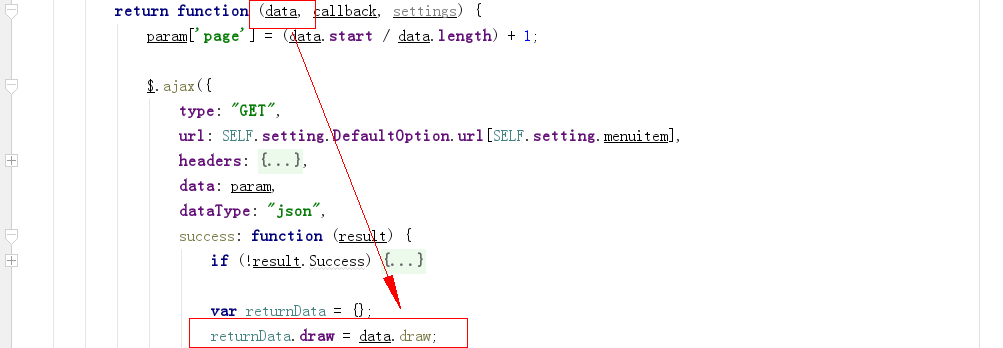
这些数据为DataTables内部构造的，用于发送到服务器的数据。如，可以利用这些数据，计算出排序、分页、过滤服务器端需要的数据。



这些数据有：

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| draw | integer | 绘制计数器。用来确保Ajax从服务器返回的是对应的。 |
| start | integer | 当前页第1条数据的起始位置，从0开始。 |
| length | integer | 每页显示项目数。  值为-1时，代表需要返回全部数据。 |
| search[value] | string | 全局的搜索条件（但列searchable需要设置为true）。 |
| search[regex] | boolean | 为true代表全局搜索的值是作为正则表达式处理（但需要服务器支持）。 |
| order[i][column] | integer | 通知后台那些列是需要排序的。i为数组索引。  Columns为数组，从0开始。 |
| order[i][dir] | string | 通知对应列的排序的方式，desc/asc。 |
| columns[i][data] | string | columns数据，由columns.data定义。 |
| columns[i][name] | string | columns名字，由columns.name定义。 |
| columns[i][searchable] | boolean | 标记列是否能被搜索。 |
| columns[i][orderable] | boolean | 标记列是否能排序。 |
| columns[i][search][value] | string | 标记具体列的搜索条件。 |
| columns[i][search][regex] | boolean | 特定列的搜索条件是否视为正则表达式。 |

其中，draw的作用是用来确保Ajax从服务器返回的是对应的。因Ajax是异步的，所以返回的顺序是不确定的。可以通过两种方式实现每个表发送接收的请求一致。一是将draw发送给服务器，然后要求在服务器接收到此参数后再返回。而是在js中控制，



每个表调用ajax方法时传参会形成执行环境，从而实现draw一致。

**6.**[**ajax.dataSrc**](http://datatables.club/reference/option/ajax.dataSrc.html)**和ajax为function时需要传递给DataTables的数据**

|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| draw | [integer](http://datatables.net/reference/type/integer) | 必要。 |
| recordsTotal | [integer](http://datatables.net/reference/type/integer) | 必要。即没有过滤的记录数（数据库里总共记录数）。 |
| recordsFiltered | [integer](http://datatables.net/reference/type/integer) | 必要。过滤后的记录数（如果DataTables设置了过滤条件）。 |
| data | [array](https://datatables.net/reference/type/array) | 必要。表中中需要显示的数据。 |

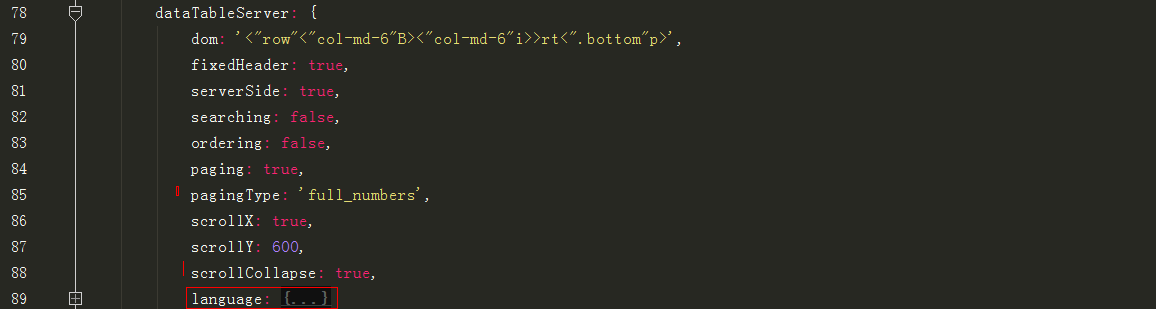
除上面的返回参数外，data中还可以加入下面的参数，来实现对表格数据的自动绑定。

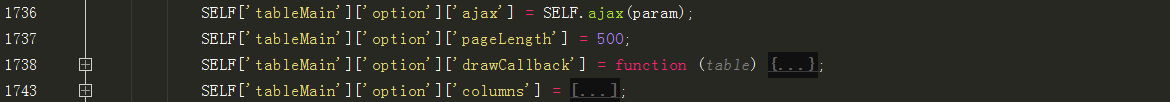
|  |  |  |
| --- | --- | --- |
| **名称** | **类型** | **描述** |
| DT\_RowId | [string](http://datatables.net/reference/type/string) | 自动绑定到tr节点上。 |
| DT\_RowClass | [string](http://datatables.net/reference/type/string) | 自动把这个类名添加到tr。 |
| DT\_RowData | [object](http://datatables.net/reference/type/object) | 使用[jQuery.data()](http://api.jquery.com/data/)方法把数据绑定到row中，方便之后用来检索（比如加入一个点击事件）。 |
| DT\_RowAttr | [object](http://datatables.net/reference/type/object) | 自动绑定数据到tr上，使用[jQuery.attr()](http://api.jquery.com/attr/)方法，对象的键用作属性，值用作属性的值。注意这个需要Datatables1.10.5+的版本才支持。 |

* + - 1. 配置项使用示例

（1）初始化传入配置选项。







（2）设置默认选项$.fn.dataTable.defaults。



* + - 1. API使用示例

**1.概念**

（1）DataTables提供6类可以操作表格数据的API：表、列、行、单元格、核心方法、工具类。

（2）DataTables初始化后的表格就是1个DataTables实例。可以通过3种方式获取1个DataTables实例：$(selector).DataTable()、$(selector).dataTable().api()、new $.fn.dataTable.Api(selector)。

（3）$(selector).DataTable()与$(selector).dataTable()两者的区别为：前者返回1个datatable中API实例，而后者返回1个添加了DataTables的api()方法的jQuery对象，该jQuery对象既可以操纵表节点，也可以访问API。

（4）DataTables初始化环境即其上下文（跟dom都没关系，就是初始化时返回的对象），如果只实例化1个表，该环境中就只有这1个实例。如果实例化多个表，在这多个表的上下文实例中调用API，则对每个表进行同样的操作。



（5）调用api接口获取到的结果就是结果集，本质与jQuery数组相同，拥有和数组相同的方法。

（6）除部分API（如cell().node()得到一个td/th对象）不会返回API实例，大部分API返回结果提供多级链式调用方法和属性。且，无论在何种嵌套层级，顶级方法将永远可用。

* + - 1. 事件使用示例

（1）可以直接在DataTables使用on接口，或通过jquery获取table再使用on方法监听。

（2）DataTables自定义事件与标准DOM事件使用机制一致。所有的DataTables自定义事件均带.dt命名空间，当使用API接口on时，会自动加上命名空间，如果使用jquery的on方法，则需手动添加。





（3）使用off接口，或jquery的off方法可以移除事件。

（4）DataTables自定义事件会在DOM树冒泡传递，因此可以在上级节点监听事件。

* + 1. 配置选项
       1. 基础配置

**1.布局**

|  |  |  |
| --- | --- | --- |
| **名称** | **描述** | **示例** |
| [dom](http://datatables.club/reference/option/dom.html) | * 定义DataTables组件元素显示和顺序。 * dom中如果出现成对<>，会在对应位置生成1个div元素，可以使用"#id"、"class"、"#id.class"设置div元素的id和class。 |  |
| [autoWidth](http://datatables.club/reference/option/autoWidth.html) | * 控制Datatables是否自适应宽度。 * 设置为true会默认计算并设置每列及table的width。 * 设置为false，table的width将为100%。 * 和scrollX没有关系，[scrollX](http://datatables.club/reference/option/scrollX.html)是设置在容器wrapper上的。 |  |
| [scrollX](http://datatables.club/reference/option/scrollX.html" \t "_blank) | * 设置水平滚动。 | "scrollX":true/px |
| [scrollY](http://datatables.club/reference/option/scrollY.html) | * 设置垂直滚动，不会应用于header或者footer。 |  |
| [scrollCollapse](https://datatables.net/reference/option/scrollCollapse" \t "_blank) | * 在使用scrollY时，当显示较少记录时，是否允许表格减少高度。 |  |
| [processing](http://datatables.club/reference/option/processing.html) | * 是否显示处理状态。 * 设置为true会显示“处理中...”，但较小，可以更改样式覆盖整个表。 | "processing":true |
| [stripeClasses](https://datatables.net/reference/option/stripeClasses) | * 设置斑马条（奇偶行）的css类 | "stripeClasses":['strip1','strip2','strip3'] |
| [renderer](https://datatables.net/reference/option/renderer) | * 显示组件的渲染方式，支持对header和pageButton的设置 | renderer:"bootstrap"  renderer:{  "header":"jqueryui",  "pageButton":"bootstrap"  } |

**2.分页**

|  |  |  |
| --- | --- | --- |
| **名称** | **描述** | **示例** |
| [paging](http://datatables.club/reference/option/paging.html) | * 是否开启本地分页。 |  |
| [pageLength](https://datatables.net/reference/option/pageLength" \t "_blank) | * 改变初始的页面长度（每页显示的记录数）。 |  |
| [displayStart](https://datatables.net/reference/option/displayStart) | * 初始化显示的时候从第几条数据开始显示（从而自动设置开始显示是第几页）。 | "displayStart":20 |
| [pagingType](https://datatables.net/reference/option/pagingType) | * 分页按钮的显示方式，有6个值：numbers，只显示数字，simple：只显示previous和next，simple\_numbers：前面两个，full；full\_numbers；first\_last\_numbers。 |  |
| [lengthChange](http://datatables.club/reference/option/lengthChange.html) | * 是否允许用户改变表格每页显示的记录数。设为true会显示一个可以让用户选择每页显示多少条记录的下拉框。 * 下拉框中的选项值利用lengthMenu来配置。 * 如果pagination使用paging配置成禁用，这个选项就会自动被设为false。 | "lengthChange":false |
| [lengthMenu](https://datatables.net/reference/option/lengthMenu) | * 定义在每页显示记录数的select中显示的选项。 | "lengthMenu":[[10,25,50,-1],[10,25,50,"All"]] |

**3.排序**

|  |  |  |
| --- | --- | --- |
| **名称** | **描述** | **示例** |
| [ordering](http://datatables.club/reference/option/ordering.html) | * 是否允许Datatables开启排序，开启后，默认所有列都可以排序。 |  |
| [order](https://datatables.net/reference/option/order) | * 表格在初始化的时，按那些列排序。 | "order":[[0,'asc'],[1,'asc']] |
| [orderFixed](https://datatables.net/reference/option/orderFixed) | * 在表格中总是起排序作用的列，即使用户没有点击表头某列。 | "orderFixed":{  "pre":[0,'asc']  } |
| [orderMulti](https://datatables.net/reference/option/orderMulti) | * 是否允许多列排序，bool值。 |  |
| [orderClasses](https://datatables.net/reference/option/orderClasses) | * 排序列是否高亮显示，bool值。 |  |
| [orderCellsTop](https://datatables.net/reference/option/orderCellsTop) | * 复杂表头排序响应，bool值。 |  |

想要控制某些列可以排序/不能排序，需要到colomns.orderable中设置，前提是[ordering](http://datatables.club/reference/option/ordering.html)为true。这里一旦开启，则默认所有列都可以排序。

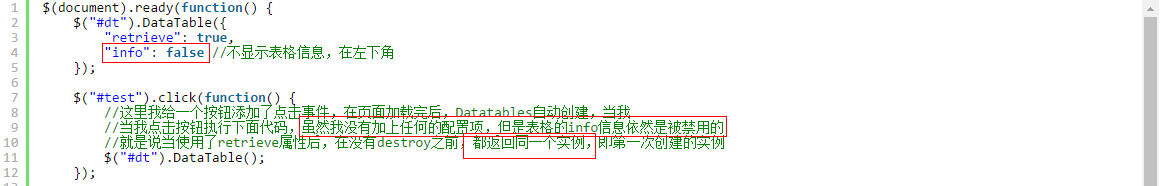
**4.搜索/过滤**

|  |  |  |
| --- | --- | --- |
| **名称** | **描述** | **示例** |
| [searching](http://datatables.club/reference/option/searching.html) | * 是否允许并显示搜索/过滤。 |  |
| [search.caseInsensitive](https://datatables.net/reference/option/search.caseInsensitive) | * 搜索/过滤时，是否大小写敏感。 |  |
| [search.regex](https://datatables.net/reference/option/search.regex) | * 是否允许搜索字符串中出现的正则表达式，bool值。 |  |
| [search.smart](https://datatables.net/reference/option/search.smart) | * 是否允许搜索/过滤时的分词功能。 | "smart":false |
| [search.search](https://datatables.net/reference/option/search.search) | * 初始化时的表格的过滤（搜索）条件（则初始化后表格中只显示满足搜索条件的行）。 | "search":"Fred" |
| [searchCols](https://datatables.net/reference/option/searchCols) | * 与[search.search](https://datatables.net/reference/option/search.search)类似，给每一列定义一个初始的搜索条件。 | "searchCols":[  null,  {"search":"Myfilter"},  null,  {"search":"^[0-9]","escapeRegex":false}  ] |
| [searchDelay](https://datatables.net/reference/option/searchDelay) | * 设定搜索的间隔时间。 | searchDelay:350 |
| [search](https://datatables.net/reference/option/search) | * 1个对象，可以在里面定义上面的这些选项。 |  |

**5.其他**

|  |  |  |
| --- | --- | --- |
| **名称** | **描述** | **示例** |
| [info](http://datatables.club/reference/option/info.html) | * 是否显示表信息。 |  |
| [destroy](http://datatables.club/reference/option/destroy.html) | * 在初始化一个Datatables，如果实例已经存在，是否销毁（配置和数据），成为一个全新的Datatables实例。 * 如果你只想改变数据而不是配置选项，使用ajax.reload()方法或者rows.add()。 |  |
| [retrieve](http://datatables.club/reference/option/retrieve.html) | * 是否检索已经存在的Datatables实例，并继续使用之前的Datatables实例。 |  |
| [stateSave](http://datatables.club/reference/option/stateSave.html) | * 是否保存状态。当开启了状态储存，Datatables会存储一个状态到浏览器上，包含分页位置，每页显示的长度，过滤后的结果和排序。当用户重新刷新页面，表格的状态将会被设置为之前的设置。 * 使用当前页面的路径和table的domid作为每个表状态数据、信息储存的唯一标示。 |  |
| [stateDuration](https://datatables.net/reference/option/stateDuration) | * 间隔多长时间保存一次状态 | "stateDuration":60\*60\*24 |

[destroy](http://datatables.club/reference/option/destroy.html)/[retrieve](http://datatables.club/reference/option/retrieve.html)示例：





* + - 1. 数据

|  |  |  |
| --- | --- | --- |
| **名称** | **描述** | **示例** |
| [serverSide](http://datatables.club/reference/option/serverSide.html) | * 是否开启服务器模式。 |  |
| [data](http://datatables.club/manual/daily/2016/04/21/option-ajax-data.html) | * 客户端模式下，指定table显示的数据源。 |  |
| [ajax](http://datatables.club/manual/daily/2016/04/18/option-ajax.html) | | |
| [ajax.data](http://datatables.club/reference/option/ajax.data.html) | * 将额外的参数添加到请求，类似于jQuery的ajax.data属性。 * 值是函数的时候用于每次Datatables请求服务器时都可以动态计算提交的参数，如用于文本框 |  |
| [ajax.dataSrc](http://datatables.club/reference/option/ajax.dataSrc.html) | * 改变从服务器返回的数据给Datatables，或操作数据从一种形式转换成另一种形式（比如xml、json、yaml等）。 * String：定义为从数据源 * 函数：操作从服务器返回的数据转化成另一种。 |  |
| [deferRender](http://datatables.club/reference/option/deferRender.html) | * 控制Datatables的延迟渲染，可以提高初始化的速度。 * 当DataTables加载Ajax数据或者Javascript数据时，它会提前创建所需的所有HTML元素。 | "deferRender":true |
| [deferLoading](https://datatables.net/reference/option/deferLoading) | * 延迟请求加载服务端的数据，直到表格第二次绘制。 | "deferLoading":57（57项数据记录可用）  "deferLoading":[57,100]（57项被过滤，100项未被过滤） |
| [rowId](https://datatables.net/reference/option/rowId) | * tr绑定ID，可用于行选择或事件数据获取。 |  |

* + - 1. 回调函数

|  |  |
| --- | --- |
| **名称** | **描述** |
| [drawCallback](http://datatables.club/reference/option/drawCallback.html" \t "_blank) | 表格每次重绘回调函数。 |
| [preDrawCallback](https://datatables.net/reference/option/preDrawCallback) | 表格绘制前的回调函数。 |
| [initComplete](http://datatables.club/reference/option/initComplete.html) | 初始化结束后的回调函数。 |
| [headerCallback](https://datatables.net/reference/option/headerCallback) | 表格Header显示时的回调函数。 |
| [footerCallback](http://datatables.club/reference/option/footerCallback.html) | tfoot的回调函数。 |
| [createdRow](http://datatables.club/reference/option/createdRow.html) | 当tr元素被创建（所有的td元素已经插入进去）。 |
| [rowCallback](https://datatables.net/reference/option/rowCallback) | 表格行(Row)绘制的回调函数，在每一行绘制完成后、渲染到页面前调用。 |
| [infoCallback](https://datatables.net/reference/option/infoCallback) | 改变表格状态信息的回调函数。 |
| [formatNumber](https://datatables.net/reference/option/formatNumber) | 数字格式化时的回调函数？ |
| [stateLoadCallback](https://datatables.net/reference/option/stateLoadCallback) | 该回调函数定义了从哪里和如何读取保存的状态。默认情况下DT从localStorage读取，但是也许你也会从服务器或者cookies中获取这些状态。这个回调函数需要和stateSaveCallback配合使用。一个是如何读取，一个是如何存储。 |
| [stateLoaded](https://datatables.net/reference/option/stateLoaded) | 状态加载完成之后的回调函数。 |
| [stateLoadParams](https://datatables.net/reference/option/stateLoadParams) | 状态加载完成之后，对数据处理的回调函数。 |
| [stateSaveCallback](https://datatables.net/reference/option/stateSaveCallback) | 该回调函数定义了状态该存储在什么地方及如何存储。 |
| [stateSaveParams](https://datatables.net/reference/option/stateSaveParams) | 对状态进行存储时，对数据处理的回调函数。 |

* + - 1. 列

|  |  |  |
| --- | --- | --- |
| **名称** | **描述** | **示例** |
| [columnDefs](http://datatables.club/reference/option/columnDefs.html) | 定义多个列的初始属性。 | "columnDefs":[{  "targets":0,  "searchable":false  }] |
| [columns.title](https://datatables.net/reference/option/columns.title) | 该列显示在thead中的标题。 |  |
| [columns.name](https://datatables.net/reference/option/columns.name) | 为列设定一个别名（描述性名字），可用于从row()中取数据。 |  |
| [columns.data](https://datatables.net/reference/option/columns.data) | 设置列的数据源，即如何从配置项data中获取数据。 |  |
| [columns.defaultContent](https://datatables.net/reference/option/columns.defaultContent) | 设定该列的默认值。 |  |
| [columns.render](http://datatables.club/reference/option/columns.render.html) |  |  |
| [columns.type](https://datatables.net/reference/option/columns.type) | 设定该列类型，可以为date、num、num-fmt、html-num、html-num-fmt、html、string等，可用于列排序或者搜索。 | {"type":"html"} |
| [columns.cellType](https://datatables.net/reference/option/columns.cellType) | 为列指定一个单元格类型（td/th）。 | "cellType":"th" |
| [columns.className](https://datatables.net/reference/option/columns.className) | 为指定的列的每个单元格都指定一个cssclass。 |  |
| [columns.contentPadding](https://datatables.net/reference/option/columns.contentPadding) |  |  |
| [columns.visible](https://datatables.net/reference/option/columns.visible) | 显示或隐藏该列。 |  |
| [columns.width](https://datatables.net/reference/option/columns.width) | 设定列宽。 | {"width":"20%"} |
| [columns.createdCell](https://datatables.net/reference/option/columns.createdCell) | 每个单元格生成以后的回调函数，这样你可以在这里改变DOM。 |  |
| [columns.searchable](https://datatables.net/reference/option/columns.searchable) | 在该列上允许或者禁止过滤搜索记录。 |  |
| [columns.orderable](https://datatables.net/reference/option/columns.orderable) | 在该列上允许或者禁止排序功能。 |  |
| [columns.orderData](https://datatables.net/reference/option/columns.orderData) | 定义多个列的默认排序条件（有些列数据是关联的，如firstname和lastname，需要1起排序）。 | {"orderData":[0,1]}, |
| [columns.orderDataType](https://datatables.net/reference/option/columns.orderDataType) | 列的表格的DOM类型（如果列内容是form，指出的话会加快时间），可以用来方便排序。 | "orderDataType":"dom-select", |
| [columns.orderSequence](https://datatables.net/reference/option/columns.orderSequence) | 排序的顺序，正序或倒序。 |  |
| [columns](http://datatables.club/reference/option/columns.html) | 设定列的所有初始属性。 |  |

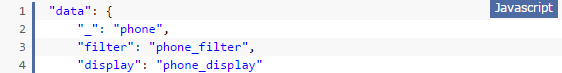
* [columns.data](https://datatables.net/reference/option/columns.data)：值可以为：

（1）undefined（未定义），将从columns.defaultContent中获取值，如果没有则报错。

（2）null，但仍然会从columns.render/columns.defaultContent中获取值，如果都没有则显示为空。

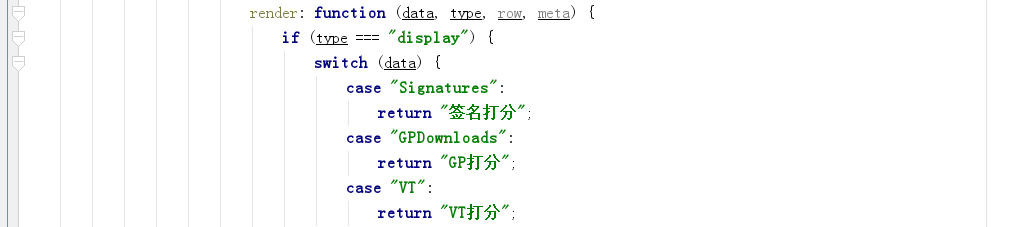
（3）string，对应配置项中的key，注意转义./[]/()等。

（4）object，如：



（5）function。

* [columns.render](http://datatables.club/reference/option/columns.render.html)：主要为使用function类型配置对从[columns.data](https://datatables.net/reference/option/columns.data)获取的数据进一步处理（其他功能和[columns.data](https://datatables.net/reference/option/columns.data)重复），如：



data为源数据，type为数据类型，如display/filter/sort等，row为整个row的数据，meta为请求单元格的行/列索引即配置项等其他数据。

* + 1. API
       1. 核心

|  |  |
| --- | --- |
| **名称** | **描述** |
| [draw()](http://datatables.club/reference/api/draw().html) | 重绘表格 |
| [$()](http://datatables.club/reference/api/$().html) | 在整个表格里执行（完成）一个jQuery选择器操作 |
| [destroy()](http://datatables.net/reference/api/destroy()) | 销毁当前上下文中的datatables实例 |
| [clear()](http://datatables.net/reference/api/clear()) | 清除表格里所有数据 |
| [ajax.reload()](http://datatables.club/reference/api/ajax.reload().html) | 重新加载数据源获取数据（不能指定新的数据源）。 |
| [ajax.url().load()](http://datatables.club/reference/api/ajax.url().load().html) | 设置新的url数据源，并重新加载数据。 |
| [ajax.url()](http://datatables.club/reference/api/ajax.url().html) | 设置新的url数据源。 |
| [ajax.json()](http://datatables.net/reference/api/ajax.json()) | 获得最终的json数据 |
| [ajax.params()](http://datatables.net/reference/api/ajax.params()) | 获得最终请求到服务器的参数 |
| [on()](http://datatables.net/reference/api/on()) | 打开表格的监听事件 |
| [one()](http://datatables.net/reference/api/one()) | 第一次监听表格事件，然后全部移除监听 |
| [off()](http://datatables.net/reference/api/off()) | 移除表格的监听事件 |
| [data()](http://datatables.net/reference/api/data()) | 获得表格中所有数据 |
| [settings()](http://datatables.net/reference/api/settings()) | 获得表格的设置对象 |
| [info](http://datatables.club/reference/option/info.html) | 控制是否显示表格的信息 |
| page.info() | 获取分页相关信息 |
| [page()](http://datatables.club/reference/api/page().html) | 获得或者设置表格当前页 |
| [order()](http://datatables.net/reference/api/order()) | 获得表格当前的排序列或者是设置表格那些列排序 |
| [order.listener()](http://datatables.net/reference/api/order.listener()) | 给任意一个元素（按下这个按钮就排序）添加一个排序的监听 |
| [search()](http://datatables.net/reference/api/search()) | 搜索表格里的数据 |
| [state()](http://datatables.net/reference/api/state()) | 得到表格最后存储的状态 |
| [state.clear()](http://datatables.net/reference/api/state.clear()) | 清楚表格储存的状态 |
| [state.loaded()](http://datatables.net/reference/api/state.loaded()) | 表格初始化的时候读取之前的状态 |
| [state.save()](http://datatables.net/reference/api/state.save()) | 保存表格的当前状态 |

* + - * 1. draw



当分页、排序、过滤后，需要此接口触发表格重绘。参数reset表示是否重置当前的分页位置。

* + - * 1. reload

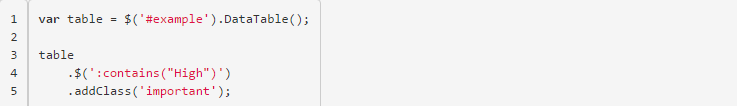


传入callback参数的数据就是服务器返回的数据，可以在这里对一些dom进行操作。

* + - * 1. $()



选择器功能等同于对table执行.find，查找子元素



* + - * 1. page.info()

可以获取所有分页信息。



* + - 1. 单元格/列/行
* 单元格

|  |  |
| --- | --- |
| **名称** | **描述** |
| [cell().cache()](http://datatables.club/reference/api/cell().cache().html) | 获取被选择的单元格的缓存数据 |
| [cell().data()](http://datatables.club/reference/api/cell().data().html) | 获取/设置被选择的单元格的数据 |
| [cell().index()](http://datatables.club/reference/api/cell().index().html) | 获取被选择的单元格的索引信息 |
| [cell().invalidate()](http://datatables.club/reference/api/cell().invalidate().html) |  |
| [cell().node()](http://datatables.net/reference/api/cell().node()) | 获得选中单元格的dom |
| [cell().render()](http://datatables.net/reference/api/cell().render()) | 获得渲染过的单元格数据（正交数据） |
| [cell()](http://datatables.net/reference/api/cell()) | 从表格中选择一个单元格 |
| [cells().cache()](http://datatables.net/reference/api/cells().cache()) | 从缓存里获取选中多个单元格的数据 |
| [cells().data()](http://datatables.net/reference/api/cells().data()) | 获取选中的多个单元格值 |
| [cells().indexes()](http://datatables.net/reference/api/cells().indexes()) | 获得选中的多个单元格的索引信息 |
| [cells().invalidate()](http://datatables.net/reference/api/cells().invalidate()) |  |
| [cells().nodes()](http://datatables.net/reference/api/cells().nodes()) | 获得选中的多个单元格的dom |
| [cells().render()](http://datatables.net/reference/api/cells().render()) | 获得渲染过的多个单元格数据 |
| [cells()](http://datatables.net/reference/api/cells()) | 从表格中选择多个单元格 |

* 列

|  |  |
| --- | --- |
| **名称** | **描述** |
| [column().cache()](http://datatables.net/reference/api/column().cache()) | 从缓存的数据里获取选中的列 |
| [column().data()](http://datatables.net/reference/api/column().data()) | 获取选中列单元格的值 |
| [column().dataSrc()](http://datatables.net/reference/api/column().dataSrc()) | 获取选中列数据源的属性名 |
| [column().footer()](http://datatables.net/reference/api/column().footer()) | 获得选中列footer的node |
| [column().header()](http://datatables.net/reference/api/column().header()) | 获得选中列header的node |
| [column().index()](http://datatables.net/reference/api/column().index()) | 获得选中列的索引 |
| [column().nodes()](http://datatables.net/reference/api/column().nodes()) | 获得选中列所有单元格node |
| [column().order()](http://datatables.net/reference/api/column().order()) | 给指定列排序 |
| [column().search()](http://datatables.net/reference/api/column().search()) | 在指定列搜索 |
| [column().visible()](http://datatables.net/reference/api/column().visible()) | 获得那些列隐藏或者设置指定列隐藏 |
| [column()](http://datatables.net/reference/api/column()) | 在表格上选择一列 |
| [columns().cache()](http://datatables.net/reference/api/columns().cache()) | 获取表格缓存里被选中的列 |
| [columns().data()](http://datatables.net/reference/api/columns().data()) | 获取被选中列的单元格数据 |
| [columns().dataSrc()](http://datatables.net/reference/api/columns().dataSrc()) | 获取选中列的数据源属性名称 |
| [columns().footer()](http://datatables.net/reference/api/columns().footer()) | 获取选中列footer的nodes |
| [columns().header()](http://datatables.net/reference/api/columns().header()) | 获取选中列header的node |
| [columns().indexes()](http://datatables.net/reference/api/columns().indexes()) | 获取选中列的索引 |
| [columns().nodes()](http://datatables.net/reference/api/columns().nodes()) | 获取选中列单元格nodes |
| [columns().order()](http://datatables.net/reference/api/columns().order()) | 给选中列排序 |
| [columns().search()](http://datatables.net/reference/api/columns().search()) | 在指定列搜索 |
| [columns().visible()](http://datatables.net/reference/api/columns().visible()) | 得到隐藏列或者设置隐藏列 |
| [columns()](http://datatables.net/reference/api/columns()) | 从表格选择多列 |
| [columns.adjust()](http://datatables.net/reference/api/columns.adjust()) | 重新计算列宽 |

* 行

|  |  |
| --- | --- |
| **名称** | **描述** |
| [row().cache()](http://datatables.net/reference/api/row().cache()) | 获取缓存里行的数据 |
| [row().index()](http://datatables.net/reference/api/row().index()) | 获取行的索引 |
| [row().node()](http://datatables.net/reference/api/row().node()) | 获得tr节点 |
| [row().remove()](http://datatables.net/reference/api/row().remove()) | 删除行 |
| [row()](http://datatables.net/reference/api/row()) | 获取一行 |
| [row.add()](http://datatables.net/reference/api/row.add()) | 添加一行 |
| [row().child().hide()](http://datatables.net/reference/api/row().child().hide()) | 隐藏子行然后创建一个新的子行 |
| [row().child().remove()](http://datatables.net/reference/api/row().child().remove()) | 删除子行 |
| [row().child().show()](http://datatables.net/reference/api/row().child().show()) | 显示子行 |
| [row().child()](http://datatables.net/reference/api/row().child()) | 获取子行或者设置子行 |
| [row().child.hide()](http://datatables.net/reference/api/row().child.hide()) | 隐藏子行 |
| [row().child.isShown()](http://datatables.net/reference/api/row().child.isShown()) | 检测子行是否显示 |
| [row().child.remove()](http://datatables.net/reference/api/row().child.remove()) | 移除子行 |
| [row().child.show()](http://datatables.net/reference/api/row().child.show()) | 显示子行 |
| [row().child](http://datatables.net/reference/api/row().child) | 子行方法命名空间 |
| [row().data()](http://datatables.net/reference/api/row().data()) | 获取行数据或者设置行数据 |
| [rows().cache()](http://datatables.net/reference/api/rows().cache()) | 获取缓存里的行 |
| [rows().data()](http://datatables.net/reference/api/rows().data()) | 获取多行数据 |
| [rows().indexes()](http://datatables.net/reference/api/rows().indexes()) | 获取多行的索引 |
| [rows().invalidate()](http://datatables.net/reference/api/rows().invalidate()) | Invalidate the data held in DataTables for the selected rows（不理解） |
| [rows().nodes()](http://datatables.net/reference/api/rows().nodes()) | 获取多个 tr节点 |
| [rows().remove()](http://datatables.net/reference/api/rows().remove()) | 删除多行 |
| [rows()](http://datatables.net/reference/api/rows()) | 多行对象 |

* + - * 1. 单数/复数

单数均只能获取1个（貌似是第1个），即使里面的选择器能得到多个。

* + - * 1. 选择器

单元格/列/行可通过各自选择器和selector-modifier选择器选择。









单元格/列/行均有如下几种：

* No selector

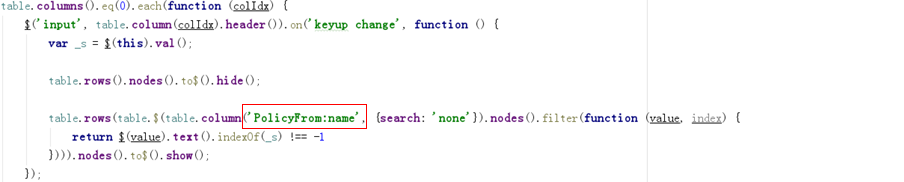
所有单元格/列/行。

* integer

即单元格/列/行的index，从0开始，只涉及tbody。如table.cell(1, 2)表示第2行第3列。table.row(2)表示第3行。

对于列，还可以有以下3种形式：

* {integer}:visIdx：可见列，如3:visIdx。
* {integer}:visible：同上。
* {string}:name：列名，如salary:name。



* string

jQuery选择器名。



* node：可以为：

对于cell，只能是该td的dom节点。

对于column，可以是该列中的td或对应的th（没有列，因为就没有列）。



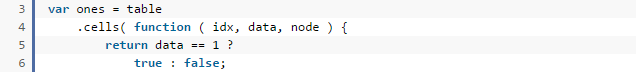
对于row，可以是改行中的td，也可以是tr。

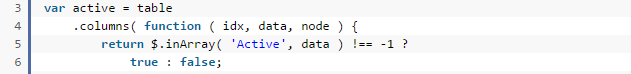


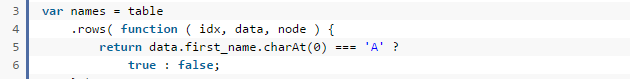


* function

传入function的参数均为index、data、node。







* jQuery对象

node的jQuery对象。

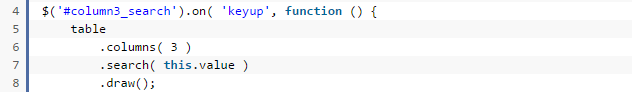
* array

对于cells/columns/rows，可以是上述值组成的数组。



* + - * 1. 列排序/过滤/搜索









* + - * 1. 子行

见实例。

* + - 1. 表格

|  |  |
| --- | --- |
| **名称** | **描述** |
| [table().body()](http://datatables.net/reference/api/table().body()) | 得到表格tbody节点（node） |
| [table().container()](http://datatables.net/reference/api/table().container()) | 得到表格的容器div，包括所有的控件（node） |
| [table().footer()](http://datatables.net/reference/api/table().footer()) | 得到tfoot节点（node） |
| [table().header()](http://datatables.net/reference/api/table().header()) | 得到thead节点（node） |
| [table().node()](http://datatables.net/reference/api/table().node()) | 得到table节点（node） |
| [table()](http://datatables.net/reference/api/table()) | 基于选择器获得表格的API对象 |



* + - 1. 工具方法

|  |  |
| --- | --- |
| **名称** | **描述** |
| [to$()](http://datatables.net/reference/api/to%24()) | 转为jquery实例（与api方法对应？） |
| [toJQuery()](http://datatables.net/reference/api/toJQuery()) | 转为jQuery实例 |
| [length](http://datatables.net/reference/api/length) | 返回结果集的长度 |
| [toArray()](http://datatables.net/reference/api/toArray()) | 把结果集转换为javascript数组 |
| [unique()](http://datatables.net/reference/api/unique()) | 去重 |
| [sort()](http://datatables.net/reference/api/sort()) | 对结果集进行排序 |
| [splice()](http://datatables.net/reference/api/splice()) | 对结果集进行分割 |
| [flatten()](http://datatables.net/reference/api/flatten()) | 把一列或者几列数据从二维数组变成一维数组 |
| [reverse()](http://datatables.net/reference/api/reverse()) | 反转结果集 |
| [join()](http://datatables.net/reference/api/join()) | 给结果集数据以字符连接然后返回一个字符串 |
| [indexOf()](http://datatables.net/reference/api/indexOf()) | 从结果集中找匹配的值，返回找到个数 |
| [lastIndexOf()](http://datatables.net/reference/api/lastIndexOf()) | 返回与字符相匹配第一次出现的位置（从后往前） |
| [iterator()](http://datatables.net/reference/api/iterator()) | 遍历表格、列，行，单元格结果集 |
| [each()](http://datatables.net/reference/api/each()) | 遍历结果集 |
| [filter()](http://datatables.net/reference/api/filter()) | 从结果集中过滤 |
| [map()](http://datatables.net/reference/api/map()) | 通过回调函数，创建一个新的结果集 |
| [pluck()](http://datatables.net/reference/api/pluck()) | 返回指定属性结果集 |
| [reduce()](http://datatables.net/reference/api/reduce()) | 遍历结果集，通过回调函数返回从左到右的数据 |
| [reduceRight()](http://datatables.net/reference/api/reduceRight()) | 遍历结果集，通过回调函数返回从右到左的数据 |
| [pop()](http://datatables.net/reference/api/pop()) | 从结果集中移除最后一个项目 |
| [push()](http://datatables.net/reference/api/push()) | 添加一个多个项目到结果集 |
| [shift()](http://datatables.net/reference/api/shift()) | 移除并返回结果集中的第一个 |
| [unshift()](http://datatables.net/reference/api/unshift()) | 在结果集里从头添加一个或多个项目，返回长度 |

* + - 1. 静态方法

|  |  |
| --- | --- |
| **名称** | **描述** |
| [$.fn.dataTable.isDataTable()](http://datatables.club/reference/api/%24.fn.dataTable.isDataTable().html) | 检查一个table节点是不是DataTables实例 |
| [$.fn.dataTable.tables()](http://datatables.club/reference/api/%24.fn.dataTable.tables().html) | 获取该页面上所有的DataTables实例 |
| [$.fn.dataTable.util.escapeRegex()](http://datatables.club/reference/api/%24.fn.dataTable.util.escapeRegex().html) | 使用正则表达式转义特殊字符 |
| [$.fn.dataTable.util.throttle()](http://datatables.club/reference/api/%24.fn.dataTable.util.throttle().html) | 减少方法调用的频率 |
| [$.fn.dataTable.versionCheck()](http://datatables.club/reference/api/%24.fn.dataTable.versionCheck().html) | 版本号兼容性检查 |

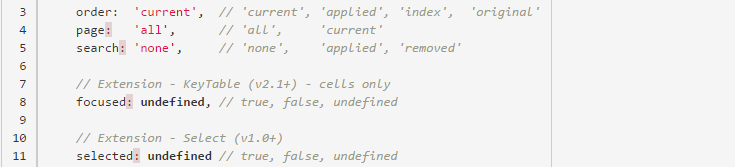
* + 1. 事件

|  |  |
| --- | --- |
| **名称** | **描述** |
| [init](http://datatables.net/reference/event/init) | 加载完成事件-当表格完全加载完毕后 |
| [draw](http://datatables.net/reference/event/draw) | 重绘事件-当表格重绘完成后 |
| [destroy](http://datatables.net/reference/event/destroy) | 表格销毁事件-当表格被销毁时 |
| [error](http://datatables.net/reference/event/error) | 错误事件-当在加载数据发生错误时 |
| [xhr](http://datatables.net/reference/event/xhr) | ajax事件-当datatable发送ajax请求完成时 |
| [preXhr](http://datatables.net/reference/event/preXhr) | ajax事件-当datatable发出ajax请求前 |
| [order](http://datatables.net/reference/event/order) | 排序事件-当表格里数据发生排序 |
| [search](http://datatables.net/reference/event/search) | 搜索事件-当过滤数据时 |
| [page](http://datatables.net/reference/event/page) | 页面改变事件-当页面发生改变 |
| [length](http://datatables.net/reference/event/length) | 分页长度改变事件-当分页长度发生改变 |
| [processing](http://datatables.net/reference/event/processing) | 处理过程事件-当表格正在处理（排序，或者加载）时 |
| [column-sizing](http://datatables.net/reference/event/column-sizing) | 列大小事件-当表格的列宽重新计算 |
| [column-visibility](http://datatables.net/reference/event/column-visibility) | 列显示事件-当表格列显示或者隐藏 |
| [stateLoaded](http://datatables.net/reference/event/stateLoaded) | 状态被加载事件-一旦状态已经被加载应用 |
| [stateLoadParams](http://datatables.net/reference/event/stateLoadParams) | 状态加载事件-当状态正在加载时 |
| [stateSaveParams](http://datatables.net/reference/event/stateSaveParams) | 状态存储事件-当正在保存表格状态信息时 |

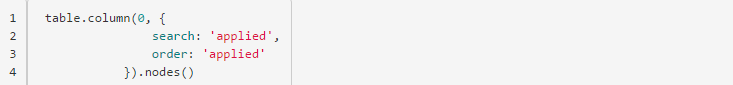
* + 1. 类型

|  |  |
| --- | --- |
| **名称** | **描述** |
| [cell-selector](https://datatables.net/reference/type/cell-selector) | cells选择器 |
| [column-selector](https://datatables.net/reference/type/column-selector) | column选择器 |
| [row-selector](https://datatables.net/reference/type/row-selector) | row选择器 |
| [selector-modifier](https://datatables.net/reference/type/selector-modifier) |  |

[selector-modifier](https://datatables.net/reference/type/selector-modifier)有如下值可供使用：



current为表中当前看到的数据。index为原始数据。applied是current的别名。original为index的别名。使用[selector-modifier](https://datatables.net/reference/type/selector-modifier)可以限定范围获取数据。



* + 1. 错误处理

默认是显示错误。可以设置为：

* string

将$.fn.dataTable.ext.errMode设置为alert/throw（console）/none。



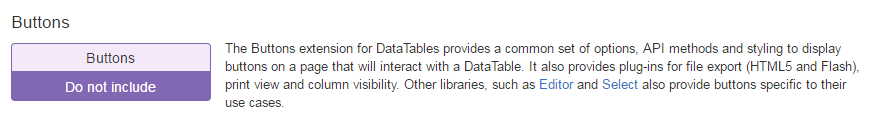
* $.fn.dataTable.ext.errMode(settings,techNote,message)

settings为配置项，techNote,message为错误代码和信息。

* + 1. 按钮
       1. 使用

Buttons提供了一些内置按钮，如打印，导出等，也提供了API让用户自己扩展按钮，触发自定义的操作。

* 下载



* 引入





* 初始化

作为DataTables初始化参数的1部分。



使用new $.fn.dataTable.Buttons来创建buttons实例。

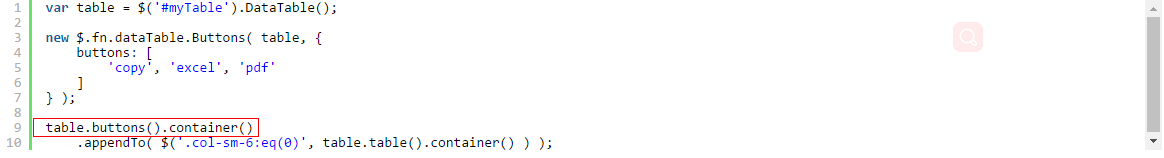


* 显示位置

使用B作为DOM参数的内容。



使用buttons().container()获得buttons的jquery对象，然后插入到自定义位置。



* + - 1. 配置

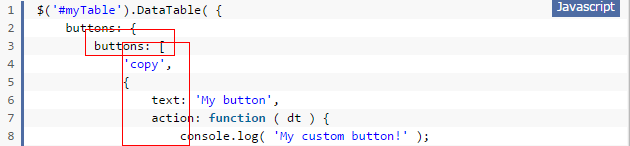
|  |  |  |
| --- | --- | --- |
| **名称** | **描述** | **示例** |
| [buttons](http://datatables.net/reference/option/buttons) | 按钮配置对象。 |  |
| [buttons.buttons](http://datatables.net/reference/option/buttons.buttons) | 创建一组按钮。 |  |
| [buttons.name](http://datatables.net/reference/option/buttons.name) | 设置组选择器的实例名称 |  |
| [buttons.buttons.name](http://datatables.net/reference/option/buttons.buttons.name) | 给每个选项定义一个名称，可用于buttons()接口，作用等同于columns中的name。 |  |
| [buttons.buttons.namespace](http://datatables.net/reference/option/buttons.buttons.namespace) | 给每个按钮不同的命名空间。 |  |
| [buttons.buttons.text](http://datatables.net/reference/option/buttons.buttons.text) | 按钮上显示的文字。 |  |
| [buttons.buttons.titleAttr](http://datatables.net/reference/option/buttons.buttons.titleAttr) | 按钮上title属性上的文字提示（如，当按钮是1个图标时）。 |  |
| [buttons.buttons.key](http://datatables.net/reference/option/buttons.buttons.key) | 定义一个键盘键来激活按钮 |  |
| [buttons.buttons.className](http://datatables.net/reference/option/buttons.buttons.className) | 给按钮加上class。 |  |
| [buttons.buttons.enabled](http://datatables.net/reference/option/buttons.buttons.enabled) | 设置按钮的初始启用状态。 | enabled:false |
| [buttons.buttons.available](http://datatables.net/reference/option/buttons.buttons.available) | Function，返回值为true才会启用该按钮。 | available(dt,config) |
| [buttons.buttons.init](http://datatables.net/reference/option/buttons.buttons.init) | 初始化函数，给按钮添加具体的事件。 | init(dt,node,config) |
| [buttons.buttons.action](http://datatables.net/reference/option/buttons.buttons.action) | 按钮触发后的动作。 | action(e,dt,node,config) |
| [buttons.buttons.destroy](http://datatables.net/reference/option/buttons.buttons.destroy) | 当按钮被摧毁时调用。 | destroy(dt,node,config) |
| [buttons.buttons.extend](http://datatables.net/reference/option/buttons.buttons.extend) | 定义一个按钮是基于什么样的功能 | extend:'excel' |
| [buttons.dom](http://datatables.net/reference/option/buttons.dom) | 控制按钮Dom的创建的结构 |  |
| [buttons.dom.container](http://datatables.net/reference/option/buttons.dom.container) | 所有按钮的容器Dom配置 |  |
| [buttons.dom.collection](http://datatables.net/reference/option/buttons.dom.collection) | 一组按钮显示的Dom配置 |  |
| [buttons.dom.buttonContainer](http://datatables.net/reference/option/buttons.dom.buttonContainer) | 一个按钮容器Dom配置 |  |
| [buttons.dom.button](http://datatables.net/reference/option/buttons.dom.button) | 单个按钮元素的Dom配置 |  |
| [buttons.dom.buttonLiner](http://datatables.net/reference/option/buttons.dom.buttonLiner) | 一个按钮边框Dom配置 |  |

* + - * 1. buttons配置形式
* 数组，只是一组按钮时使用。





* 对象，包含buttons和dom。



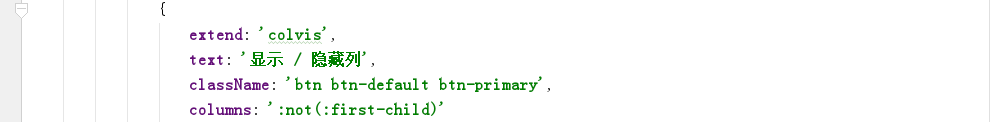
* + - * 1. dom设置

可以使用dom给默认的dom结构添加classname等。

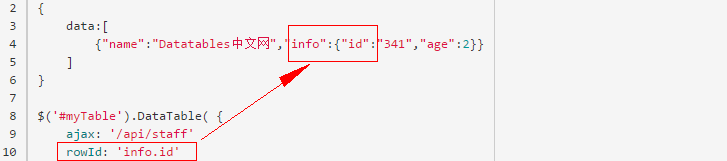
* + - * 1. 内置按钮

有文件导出、转为excel、打印、列隐藏等。需要额外引入对应的js文件，如：





* + 1. 实例
       1. 给每行自动绑定唯一id
* 有2中方法，1是通过配置项rowId绑定，使用如下：



* 在服务端模式下，可以使用DT\_RowId绑定。



* + - 1. 定时更新数据



* + - 1. 跳转到指定页
* 外观。

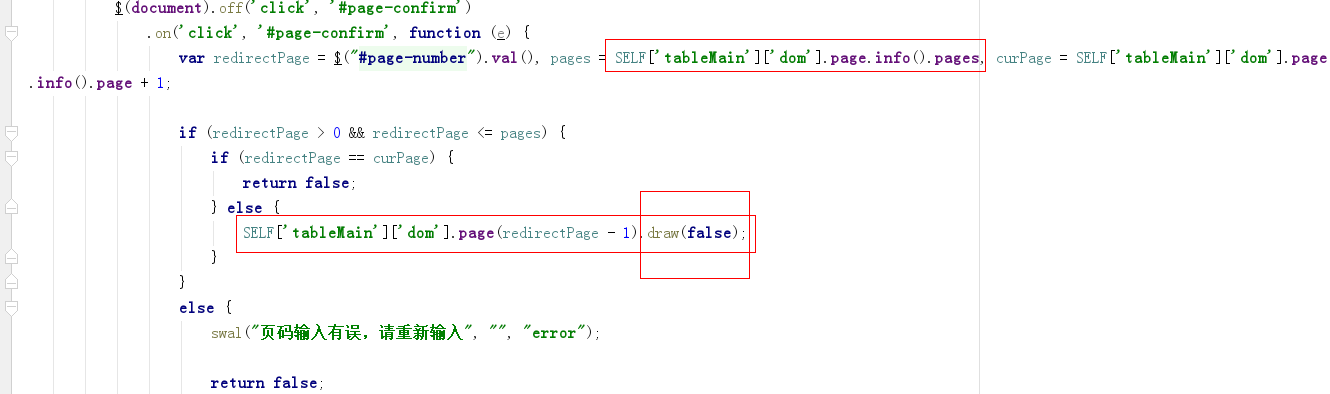


* 设置样式，在每次页面重绘时添加代码。



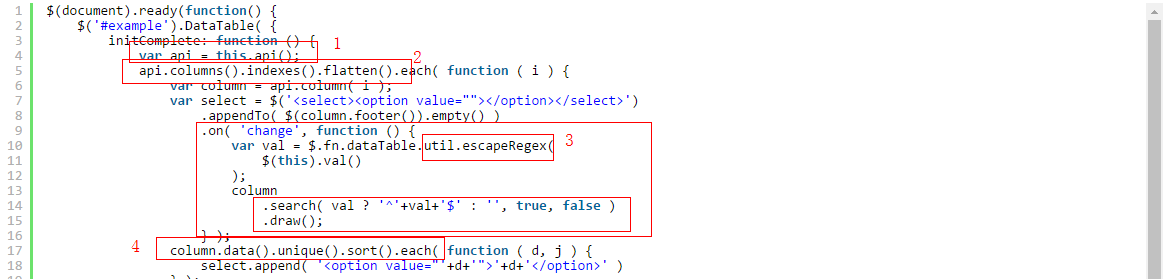
fnDrawCallback配置项已被[drawCallback](http://datatables.club/reference/option/drawCallback.html)取代。即每次表重绘完成时，添加这段代码到指定位置。

* 给按钮绑定事件。



设置page()之后，需要重新draw。

* + - 1. 在tfooter给每列增加1个下拉列表式搜索框



1.在配置项回调函数中获取的this都是添加了api方法的table的jquery对象。

2.结果集都是添加了工具方法的类数组。

3.认为用户输入的不是正则，因此转义，然后加上正则元字符。

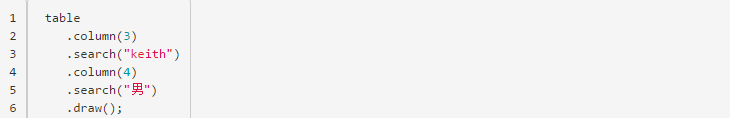
4.非常棒的方法，将结果集去重添加到select中。

* + - 1. 去掉全局搜索

不能使用"searching":false配置设置，因为这样也会禁用search方法，导致无法搜索。在dom配置中去掉f项，不让显示全局搜索即可。

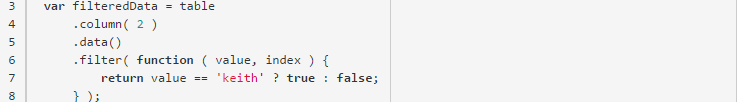
* + - 1. 对多列搜索

去掉全局搜索后，自定义一些输入框，获取值对特定列搜索。



上述搜索应该是and。

如果想要获取结果集，用filter()工具方法（search是对表操作，这个只是结果集操作）。



* + - 1. 添加索引列

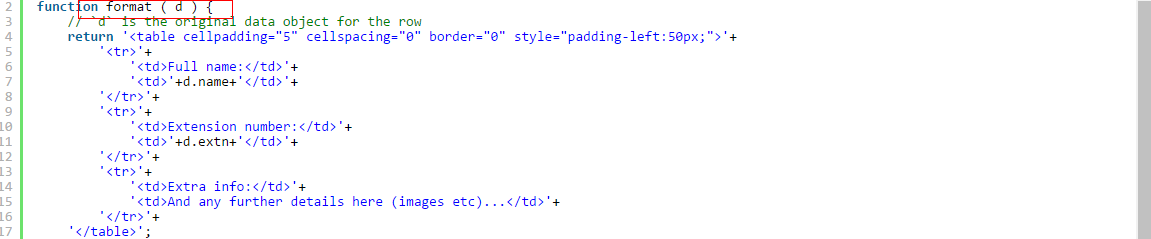


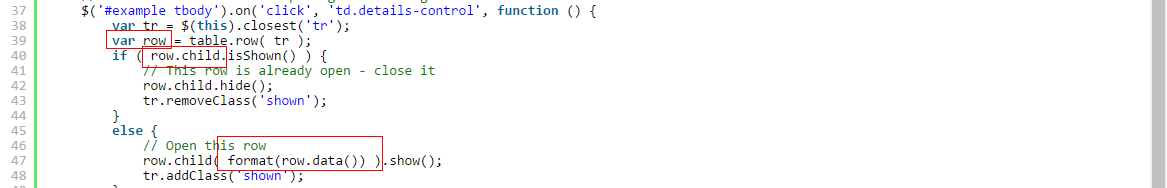
* 对索引列禁用排序、搜索。即，全局搜索不会搜索到本列。该列也没有排序按钮。
* 当表格在搜索、排序时，对该列的dom添加索引。

【获取dom然后操作】

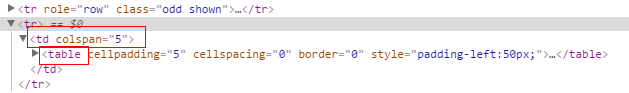
* + - 1. 子行（行的附加信息）





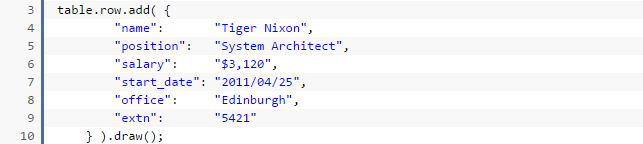


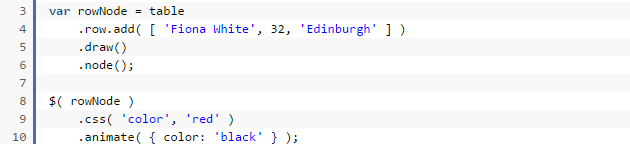
* 点击按钮，找到该按钮所在行，将dom传入child即可。
* child不一定必须是table，任何dom都可以。



* + - 1. 添加行









* + - 1. 其他

（1）表头名称

也是在columns中配置。



* 1. ECharts
     1. title

标题组件，包含主标题和副标题。均可以设置样式、链接、排版、边框、阴影等。

如果只有1个标题，title为1个对象：



ECharts3中可以存在任意多个标题组件（多个grid），需要标题进行排版，此时title为1个数组，对应grid，其中的left等配置项均相对整个坐标系。



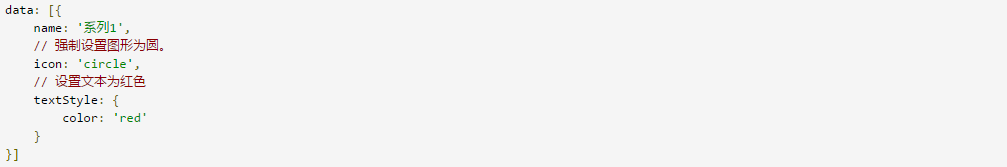
由上图可见grid是32项，title是33项，所以，它们之间没有对应关系，完全是自己设置title的坐标位置来对应（上例最后1个title为总的title）。

* + 1. legend

图例组件，展示不同系列的标记(symbol)，颜色和名字。可以通过点击图例控制哪些系列不显示。

可以设置样式、排版、边框、阴影，以及方向（orient：horizontal/horizontal）、图例项gap/大小等、格式化图例文本（formatter：可以为字符串，如'Legend {name}'，或函数），以及：

* selectedMode：图例选择的模式，控制是否可以通过点击图例改变系列的显示状态，默认开启图例选择，可以设成false关闭。除此之外也可以设成single或者multiple使用单选或者多选模式。
* selected：图例选中状态表，那些图例默认选中。
* tooltip：当图例文本过长时剪切，用tooltip方式显示。
* data：图例的数据数组，可以在其中设置某个单独图例的样式（即，单独设置上面的一些配置项，覆盖上面的通用项）。



* + 1. grid

绘图网格，ECharts3中可以存在任意个grid组件。单个grid内最多可以放置上下2个X轴，左右2个Y轴。可以在网格上绘制折线图，柱状图，散点图（气泡图）。

特殊配置项：containLabel，grid区域是否包含坐标轴的刻度标签，在无法确定坐标轴标签的宽度，容器又比较小无法预留较多空间的时候，可以设为true防止标签溢出容器。

当存在多个grid时：



x轴、y轴都需要设置为数组，并对应上grid：



series再将数据对应上坐标系：



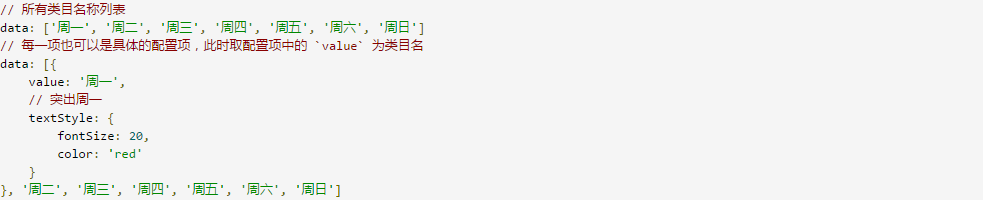
* + 1. xAxis/yAxis

单个grid为对象，多个为数组。



有如下特殊配置项：

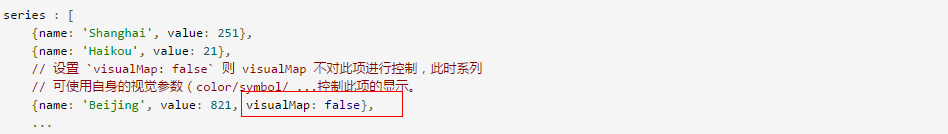
* gridIndex：x轴所在的grid的索引，默认位于第一个grid。
* position：top/bottom，x轴的第1个默认为bottom，第2为默认为top。
* nameLocation：坐标轴名称显示位置。
* boundaryGap：坐标轴两边留白策略，类目轴和非类目轴的设置和表现不一样。可以设置为为true和false，也可以设置为数组百分比。
* splitNumber：坐标轴的分割段数。
* minInterval：自动计算的坐标轴最小间隔大小。
* interval：强制设置坐标轴分割间隔。
* splitLine：坐标轴分隔线设置。
* data：数组，在类目轴（category）中有效（连续型的话没有必要，自动生成）。



* + 1. visualMap

visualMap是视觉映射组件，用于视觉编码，即，将数据映射到视觉元素（视觉通道）上。

如果option中设置有该组件，显示项颜色将根据series中对应的数据计算作色。如果series中某个数据项需要避开visualMap映射，可以如下配置：



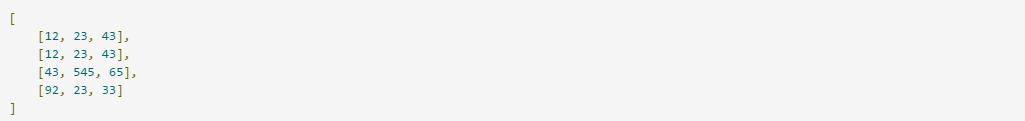
* + - 1. 连续型视觉映射组件（visualMapContinuous）

主要配置项：

* min/max：visualMapContinuous 组件的允许的最小值/最大值，必需设置。
* range：指定手柄初始值的位置，值应在min与max范围内（不设置默认为min/max）。
* calculable：是否显示拖拽用的手柄（手柄能拖拽调整选中范围）。
* realtime：拖拽时，是否实时更新。
* seriesIndex：指定取哪个系列的数据，即哪个系列的series.data，默认取所有系列（即所有的系列颜色都受控制）。series中可以有多个系列，如下：



* dimension：指定用数据中的『哪个维度』映射到视觉元素上。『数据』即series.data。可以把series.data理解成一个二维数组，例如：



其中每个列是一个维度，即dimension。例如dimension为1时，取第二列（即23，23，545，23），映射到视觉元素上。默认取data中最后一个维度。

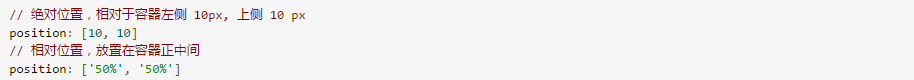
* inRange：定义目标系列（seriesIndex）视觉形式，也同时定义了visualMap-continuous本身的视觉样式。如visualMap-continuous控制的是散点图，那么inRange同时定义了散点图颜色、尺等，也定义visualMap-continuou本身颜色、尺寸等（这二者能对应上）。视觉映射的方式：支持两种方式：线性映射、查表映射。可选的视觉元素（以线性映射为例）：
* symbol：图元的图形类别，ECharts提供的标记类型包括circle、rect、roundRect、triangle、diamond、pin、arrow，也可以通过url设置为图片。线性映射下，如设置为['circle','rect','diamond']，则与visualMap.min相等的值会映射到circle，与visualMap.max相等的值会映射到diamond，对于中间的其他点，会根据他们和visualMap.min和visualMap.max的数值距离，映射到circle、rect、diamond中某个值上。
* symbolSize：图元的大小。
* color：图元的颜色，使用数组表示。如['#333','#78ab23','blue']，即以这3个点作为基准，形成一种渐变的色带，数据映射到这个色带上。
* colorAlpha：图元的颜色的透明度。
* opacity：图元以及其附属物（如文字标签）的透明度，建议使用透明度（opacity），而非颜色透明度（colorAlpha），前者能也同时控制图元中的附属物的透明度，而后者只能控制图元本身颜色的透明度。
* colorLightness：颜色的明暗度。
* colorSaturation：颜色的饱和度。
* colorHue：颜色的色调。

查表映射表示series.data中的所有值（dataValue）是可枚举的，会根据给定的映射表查表得到映射结果。

* 另有left、top等选项，用于控制图例的位置（不是显示元素的位置）。
  + - 1. 分段型视觉映射组件（visualMapPiecewise）
    1. tooltip

提示框组件，有如下选项：

* alwaysShowContent：是否一直显示。
* enterable：鼠标是否可进入提示框浮层中，默认为false，如需详情内交互，如添加链接，按钮，可设置为true。
* position：提示框浮层的位置，默认不设置时位置会跟随鼠标的位置。值可以为inside、top、left、right、bottom，以及数组，function。

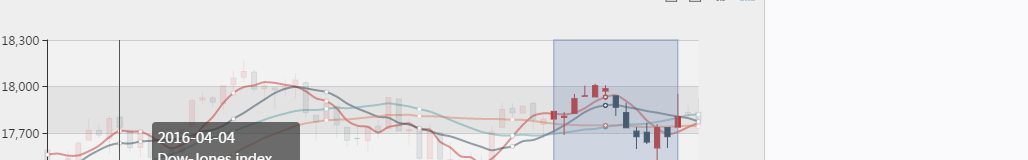


* + 1. toolbox

工具栏。内置有导出图片，数据视图，动态类型切换，数据区域缩放，重置五个工具。

* + 1. brush

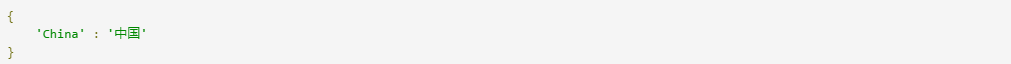
brush是区域选择组件，用户可以选择图中一部分数据，从而便于向用户展示被选中数据，或者他们的一些统计计算结果。目前brush组件支持的图表类型scatter、bar、candlestick等。



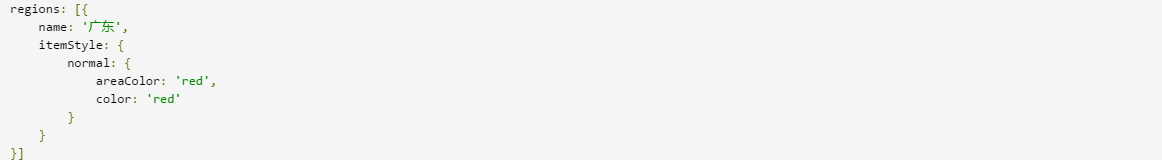
* + 1. geo

地理坐标系组件，用于地图的绘制，支持在地理坐标系上绘制散点图、线集。geo区域的颜色也可以被mapseries所控制。有如下选项：

* map：地图类型，如china、world。
* roam：是否开启鼠标缩放和平移漫游。默认不开启。
* center：当前视角的中心点，用经纬度表示，如：center:[115.97,29.71]。
* zoom：当前视角的缩放比例。
* nameMap：自定义地区的名称映射，如：



* layoutCenter：提供了除了left/right/top/bottom/width/height之外的布局手段。
* regions[i]：在地图中对特定的区域配置样式，如：



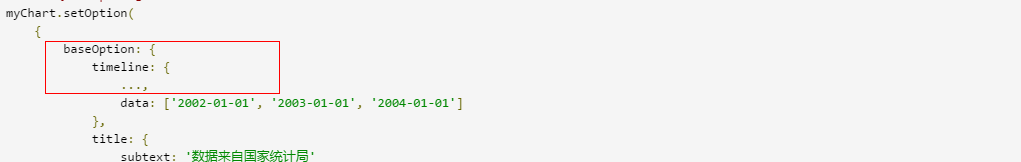
* silent：图形是否不响应和触发鼠标事件，默认为false，即响应和触发鼠标事件。
  + 1. timeline

timeline组件，实现多个ECharts option间进行切换、播放等操作的功能。因为需要操作多个option，传入ECharts的option就成为了一个集合多个原子option的复合option，并可以有1个独特的baseOption放置公共配置项。

timeline播放切换时，会把options数组中的对应的option，与baseOption进行merge形成最终的option。

options数组中，如果某一数组项中配置了某个属性，那么其他数组项中也必须配置某个属性，而不能缺省。否则这个属性的执行效果会遗留。

需要注意的是，timeline在options和baseOption中。



主要配置项有：

* axisType：轴的类型。可选值为：value数值轴，适用于连续数据。category类目轴，适用于离散的类目数据。time时间轴，适用于连续的时序数据，与数值轴相比时间轴带有时间的格式化，在刻度计算上也有所不同，例如会根据跨度的范围来决定使用月，星期，日还是小时范围的刻度。
* autoPlay：表示是否自动播放。
* loop：表示是否循环播放。
* playInterval：表示播放的速度（跳动的间隔），单位毫秒（ms）。
* realtime：拖动圆点的时候，是否实时更新视图，默认true。
* symbol：timeline标记的图形。
* data：timeline数据，格式如下：



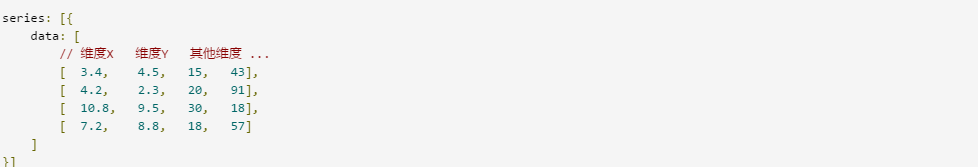
* + 1. series

系列列表。每个系列通过type决定自己的图表类型。

* + - 1. 通用选项
* name：系列名称，用于关联tooltip的显示，legend的图例筛选（应该与这些选项中的值相对应）等。
* label：图形上的文本标签，可用于说明图形的一些数据信息，比如值、名称、位置等。
* markPoint/ markLine/markArea：图表标注/图表标线/图表标域，即网格线等。
* coordinateSystem：该系列使用的坐标系，可选：cartesian2d二维的直角坐标系（也称笛卡尔坐标系），为默认值，以及polar极坐标系。
* xAxisIndex：使用的x轴的index，在单个图表实例中存在多个x轴的时候有用。
  + - 1. line

折线/面积图，主要有如下配置项：

* stack：数据堆叠，同个类目轴上系列配置相同的stack值后，后一个系列的值会在前一个系列的值上相加。
* clipOverflow：是否对超出部分裁剪，默认裁剪。
* step：是否是阶梯线图，可以设置为true显示成阶梯线图，也支持设置为start、middle、end，分别配置在当前点，当前点与下个点的中间点，下个点拐弯。
* data：系列中的数据内容数组。数组项通常为具体的数据项。通常来说，数据用一个二维数组表示。如下，每一列被称为一个『维度』。



在直角坐标系中『维度X』和『维度Y』会默认对应于xAxis和yAxis。后面的其他维度是可选的，可以在别处被使用，例如：在visualMap中可以将一个或多个维度映射到颜色，大小等多个图形属性上。在series.symbolSize中可以使用回调函数，基于某个维度得到symbolSize值。使用tooltip.formatter或series.label.normal.formatter可以把其他维度的值展示出来。

当只有一个轴为类目轴（axis.type为category）的时候，数据可以简化用一个一维数组表示。

* + - 1. bar

柱状/条形图。

* barWidth/barMaxWidth/barGap等。
  + - 1. pie

饼图。

* clockwise：饼图的扇区是否是顺时针排布。
* roseType：是否展示成南丁格尔图，通过半径区分数据大小。
* avoidLabelOverlap：是否启用防止标签重叠策略，默认开启，在标签拥挤重叠的情况下会挪动各个标签的位置，防止标签间的重叠。
* radius：饼图的半径，数组的第一项是内半径，第二项是外半径，默认值为default: [0, '75%']。
  + - 1. scatter

散点（气泡）图。

* coordinateSystem：该系列使用的坐标系，可选：cartesian2d、polar、geo（地理坐标系，通过geoIndex指定相应的地理坐标系组件）。
* geoIndex：使用的地理坐标系的index，在单个图表实例中存在多个地理坐标系的时候有用。
* large：是否开启大规模散点图的优化，在数据图形特别多的时候（>=5k）可以开启。开启后配合largeThreshold在数据量大于指定阈值的时候对绘制进行优化。缺点：优化后不能自定义设置单个数据项的样式。
  + - 1. effectScatter

带有涟漪特效动画的散点（气泡）图。

* rippleEffect：涟漪特效相关配置，有动画的时间、动画中波纹的最大缩放比例、波纹的绘制方式等设置。
  + - 1. map

地图。主要用于地理区域数据的可视化，配合visualMap组件用于展示不同区域的人口分布密度等数据。多个地图类型相同的系列会在同一地图上显示，这时候使用第1个系列的配置项作为地图绘制的配置。

* roam：是否开启鼠标缩放和平移漫游。默认不开启。如果只想要开启缩放或者平移，可以设置成scale、move。设置成true为都开启。
* center：当前视角的中心点，用经纬度表示。
* zoom：当前视角的缩放比例。
* scaleLimit：滚轮缩放的极限控制，通过min,max最小和最大的缩放值，默认的缩放为1。
* nameMap：自定义地区的名称映射，如：{'China':'中国'}。
  + - 1. gauge

仪表盘。

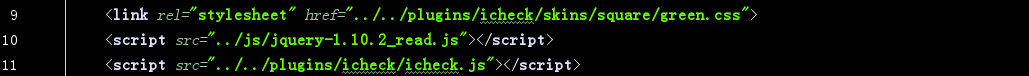
* radius：仪表盘半径，可以是相对于容器高宽中较小的一项的一半的百分比，也可以是绝对的数值。
* startAngle/endAngle：仪表盘起始角度。圆心正右手侧为0度，正上方为90度，正左手侧为180度。
* clockwise：仪表盘刻度是否是顺时针增长。
* min/max：最小/大的数据值。
* splitNumber：仪表盘刻度的分割段数。
* splitLine：分隔线样式。
* axisTick：刻度样式。
* axisLabel：刻度标签。
* pointer：仪表盘指针。
* itemStyle：仪表盘指针样式。
* detail：仪表盘详情，用于显示数据。
  1. D3

1. 表单
   1. jquery.fileupload.js
      1. 参考

* github地址：https://github.com/blueimp/jQuery-File-Upload。
* 中文参考文档：http://www.jianshu.com/p/a01a09a4c56e。
* 中文参考文档：http://www.jb51.net/article/99179.htm
  + 1. 简介
* 支持多文件上传，拖拽文件上传，进度条，预览图片、视频和音频。
* 支持跨域上传，分块上传和可恢复的文件上传和客户端图像调整。
* 支持ie6。
* 可以在支持标准HTML表单文件上传的所有服务端平台工作，包括PHP/Python等。
  + 1. 使用方法
  1. icheck

（1）下载地址：http://icheck.fronteed.com/#skins。兼容ie6。

（2）首先引入使用的皮肤css文件和js文件。



（3）使用js初始化checkbox和radio（不使用css样式，而是使用js初始化）。

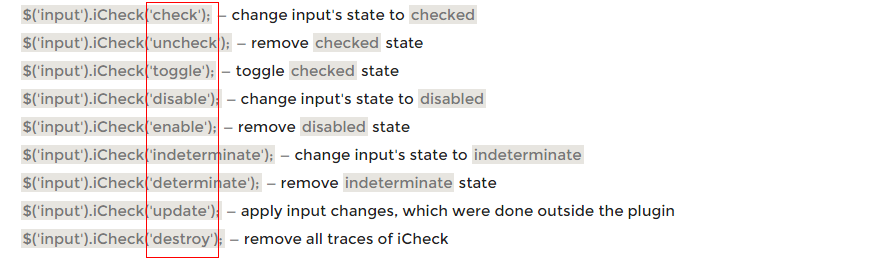


（4）初始化可以使用如下选项：

* checkboxClass/radioClass：使用的skin样式，需要引入对应的css文件。
* uncheckedClass/disabledClass/enabledClass/hoverClass/activeClass等：对应的样式类，通常不需要设置，使用默认值。
* increaseArea：扩大点击区域，即通过扩大包裹元素padding实现。

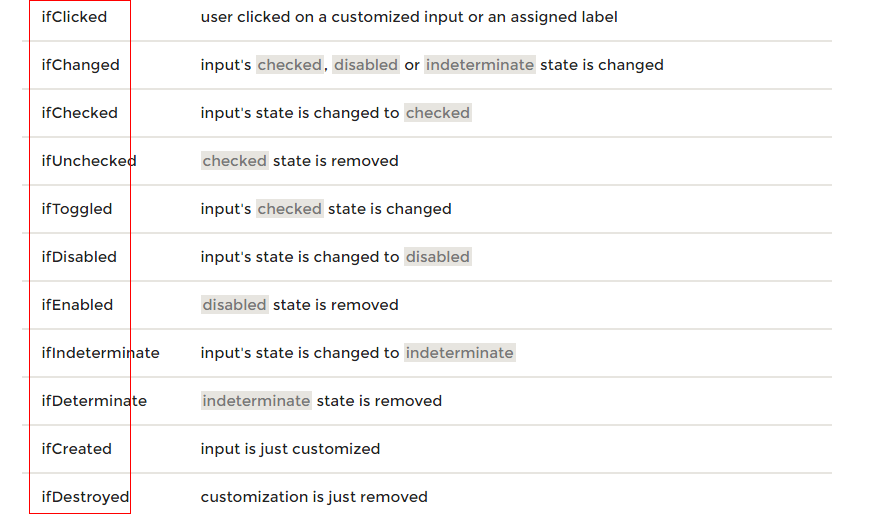
（5）方法

可以通过js操作选项。



（6）回调函数

可以在回调函数中绑定事件。



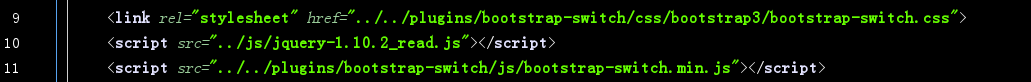
* 1. placeholder.js

在浏览器端生成placeholder image，提供完善的配置和简单的方法。

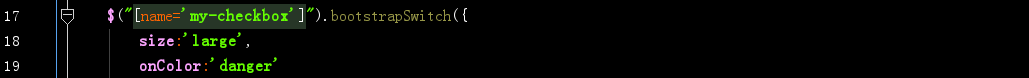
* 1. bootstrap-switch

（1）下载地址：https://github.com/Bttstrp/bootstrap-switch。兼容ie9以上。

（2）其实并不依赖bootstrap。只需要引入css文件和js文件即可，依赖jquery。



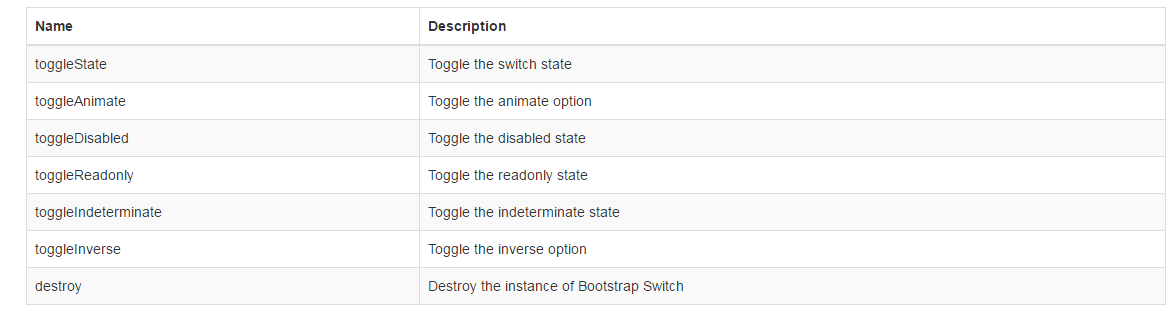
（3）仍然使用js初始化。



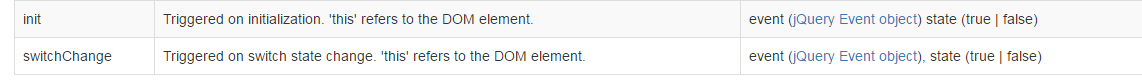
（4）配置选项

* size：值有null、mini、small、normal、large等。
* state/disabled/readonly：初始值相关。
* onColor/offColor：选中和非选中颜色。
* onText/offText/labelText：相关显示文本。
* handleWidth/labelWidth：尺寸相关。
* onInit/onSwitchChange：回调函数。

（5）方法



（6）事件



* 1. jQuery-File-Upload

兼容ie6。如果浏览器支持，还可以批量上传、超大文件上传、图片预览、拖拽上传、上传进度显示、跨域上传等功能。

**1.文件结构**

（1）依赖jquery。

（2）核心文件

* js/vendor/jquery.ui.widget.js：用于扩展文件。
* js/jquery.iframe-transport.js：使用iframe实现文件上传。
* js/jquery.fileupload.js：核心逻辑。
* js/cors/jquery.xdr-transport.js：在IE下应载入此文件解决跨域问题。
* css/jquery.fileupload.css：基础css文件。

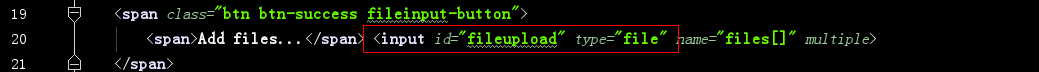
（3）其他文件

* js/jquery.fileupload-process.js：需要进度条时需要加载的文件。
* js/jquery.fileupload-validate.js：文件上传有效性验证文件，依赖于js/jquery.fileupload-process.js。
* js/jquery.fileupload-image.js/jquery.fileupload-audio.js/jquery.fileupload-video.js：图片/视频预览、调整大小需要加载的文件，该扩展又依赖于JavaScript-Load-Image和JavaScript-Canvas-to-Blob两个插件。
* js/jquery.fileupload-ui.js：完整功能js文件。
* css/jquery.fileupload-ui.css：完整功能css文件。

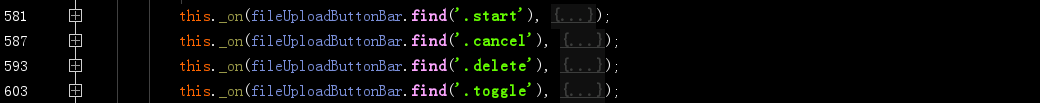
**2.样式**



（1）上传按钮使用relative+absolute，因input=file的opacity设为0，且font-size设置特别大，所以能点到。



（2）根据start、cancel、delete、toggle样式类添加事件。因此，上述按钮应该使用这些默认样式类名。



（3）样式类.files是预览文件区域。fileupload-progress是进度条区域。

**3.使用**

插件包含2个版本：基础版本（jquery.fileupload.js）和完整版本（jquery.fileupload-ui.js）。插件默认使用xhr与后台交换数据，如果不支持，使用iframe降级处理。

（1）初始化和选项

可以不用form，直接用input=file初始化；也可以对form初始化，会自动查找其中的input=file元素。



1）jquery.ajax的配置选项（应该是继承了ajax方法，所以可以直接设置这些配置项）

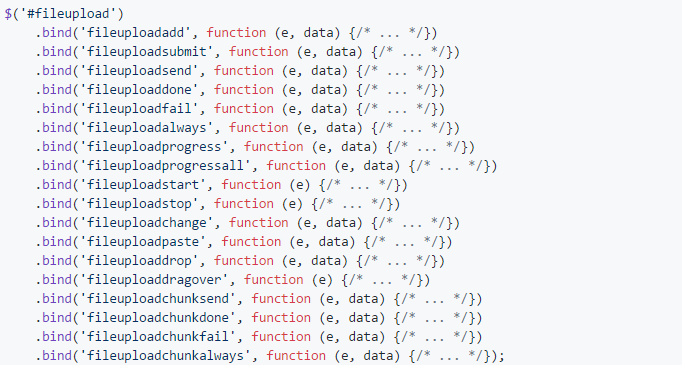
jquery.ajax的配置选项中，processData、contentType、cache应该设置为false，且timeout应该设置为0，其他选项如url、type、dataType等可以由用户自定义。

2）插件通用选项

* dropZone：拖拽上传区域，默认为document，如果设置为空或null则关闭拖拽功能（默认开启）。
* pasteZone：粘贴上传区域，设置同上，只有chrome支持。
* fileInput：监听change的上传按钮，没有设置取初始化元素。
* replaceFileInput：
* paramName：如果input=file没有设置name，取这个，可以是string或array。
* formAcceptCharset
* multipart：xhr和iframe默认都使用multipart/form-data方式。
* singleFileUploads：貌似是1个input只能上传1个文件。如果要上传多个文件，需要multipart选项设置为true。
* limitMultiFileUploads：上传文件数量限制。
* limitMultiFileUploadSize：上传文件大小限制。
* sequentialUploads：上传多个文件时是否按顺序上传。
* forceIframeTransport：强制使用iframe方式上传文件。
* redirect：貌似是上传完成后跳转页面地址。
* maxChunkSize：最大上传块大小。
* progressInterval：计算进度条时间和触发进度条事件间隔，单位为毫秒。
* autoUpload：是否自动上传。
* formData：额外附加的数据。

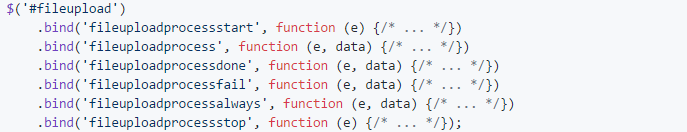
3）回调选项

也可以作为事件，如下：



4）进度条选项

也可以作为事件，如下：



5）图片/视频预览和resize选项

6）有效性选项

acceptFileTypes：接受的文件类型。

maxFileSize/minFileSize：最大/最小文件大小。

maxNumberOfFiles：最多文件数量。

disableValidation：关闭验证。

7）完整版本选项

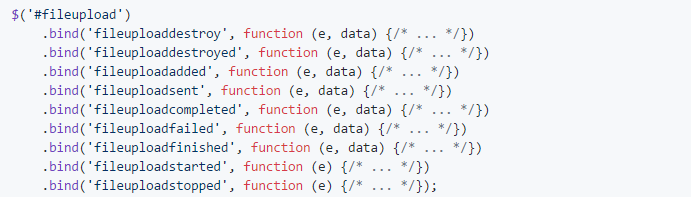
getFilesFromResponse：从服务端获取文件数量。

getNumberOfFiles：从服务端获取上传文件数量。

uploadTemplate/uploadTemplateId：预览上传文件模板文件。

downloadTemplate/downloadTemplateId：下载文件模板文件。

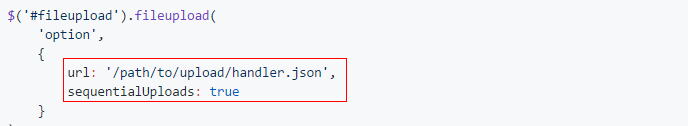
8）完整版本回调选项



（2）重置初始化选项

可以有2种方式。





（3）销毁/disabled







（4）获取进度条数据



* 1. jquery.validate

**1.参考**

http://www.runoob.com/jquery/jquery-plugin-validate.html

https://github.com/jquery-validation/jquery-validation/releases/tag/1.16.0

不支持jQuery-1.10，支持jQuery-1.11。

**2.插件方法**

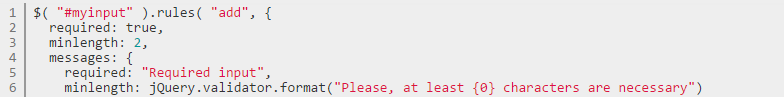
一共提供了3个方法：

（1）validate：验证所选form。



（2）valid：验证所选form是否通过验证，不接受参数。

（3）rules：获取、添加、移除某个表单元素的rules。







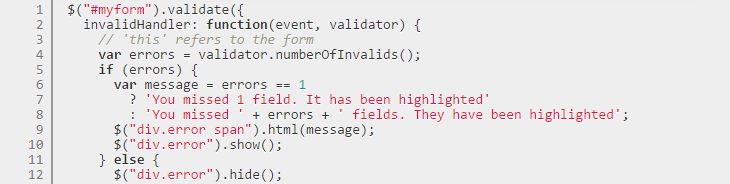
**3.选项**

debug：如果值为true，不会提交。

submitHandler：当全部验证通过时触发。入参form为绑定的元素（原生）。

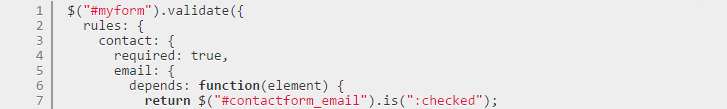


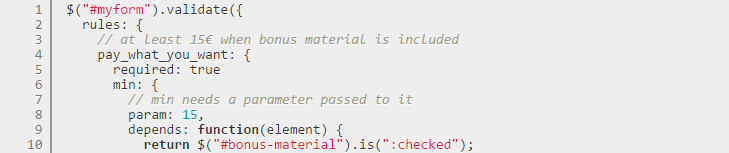
invalidHandler：当验证没有通过时触发。入参event为提交事件，validator为当前验证对象。



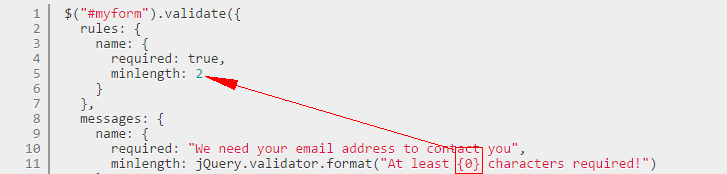
ignore：不用验证的元素。使用jquery的not过滤方法。因此，只要满足此方法的选择表达式都行。

rules：验证规则，key-value形式。其中key为验证表单的name，value为字符串或对象形式，为验证规则，还可以为依赖某个操作。

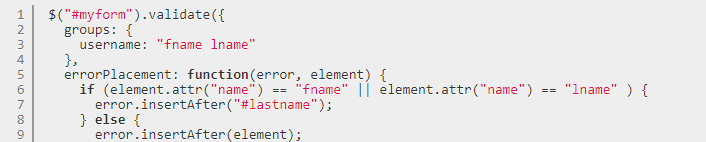




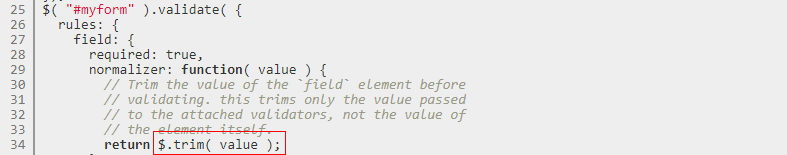
messages：错误提示。也是key-value形式。key为验证表单的name。value应该和rules中的值对应。



groups：将多个元素组合，可以给组合起1个名称。使用errorPlacement配置项判断。



normalizer：格式化元素值，或预处理元素的value。



onsubmit：是否在submit时验证表单。默认true。设置为false时需要在其他事件中验证表单。

onfocusout/onkeyup/onclick：是否在blur/keyup/click时验证表单。

focusInvalid：是否提交验证无效后，focus到第1个无效的表单元素上。

focusCleanup：是否当focus到任何元素时，去掉所有错误信息。

errorClass：错误class名，默认值为error。



validClass：正确class名，默认valid。

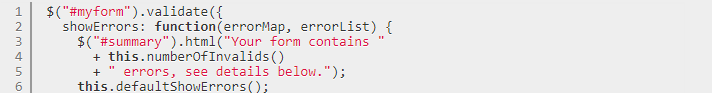
errorElement：显示错误信息的标签。

wrapper：包裹错误标签的标签，默认值window。

errorLabelContainer：所有验证信息都显示到这里面。如下，所有错误信息由li包裹，都显示到messageBox中。



showErrors：显示特定的错误信息。入参errorMap为message对象，入参errorList为元素和其错误信息。



errorPlacement：控制将错误信息放在那里。入参error为显示错误信息的标签。入参element为验证元素。



success：要验证的元素通过验证后的动作，如果跟一个字符串，会当作一个 css 类，也可跟一个函数。





highlight：可以给未通过验证的元素加效果、闪烁等。入参element为验证元素，入参errorClass为选项中的错误类，validClass为选项中的正确类。



**4.插件扩展的选择器**

（1）:blank：可以选择没有值或全是空格的表单元素。



（2）:filled：选择有值的表单元素。



（3）:unchecked：选择没有被checked的表单元素。



**5.Validator对象**

validate()方法返回1个Validator对象。可以通过该对象的方法，操作表单元素。

（1）form：验证表单是否通过验证，无入参。

（2）element：检查某个表单元素是否通过验证。



（3）resetForm：恢复表单到验证前原来的状态。

（4）showErrors：显示特定的错误信息。



（5）numberOfInvalids：返回验证通过的表单元素数量。

（6）destroy：销毁验证对象。

（7）addMethod：添加一个新的验证方法。必须包括一个独一无二的名字，一个 JAVASCRIPT 的方法和一个默认的信息。



（8）format：用参数代替模板中的{n}。



（9）setDefaults：改变默认的设置。

（10）addClassRules：增加组合验证类型，在一个类里面用多种验证方法时比较有用

**6.其他**

additional-methods.js包含附加的一些验证规则程序，如上传文件类型验证等。

1. 兼容
   1. Modernizr.js
      1. 参考

* 官网：https://modernizr.com/download?setclasses。
* 中文介绍：http://www.cnblogs.com/TomXu/archive/2011/11/18/detecting-html5-css3-features- using-modernizr.html。
* 中文介绍：http://blog.csdn.net/bingkingboy/article/details/51155254。
* 中文介绍：http://blog.chinaunix.net/uid-21633169-id-4286857.html。
  + 1. 简介

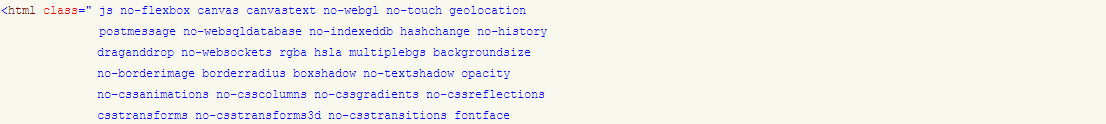
Modernizr帮助探测浏览器是否支持某些HTML5/CSS3新特性，还可以加载额外的script脚本，但不会在旧版本浏览器上添加HTML5/CSS3的新特性。

* + 1. 使用
       1. 引入js文件



* + - 1. html/css

引入之后，Modernizr会自动运行，检测浏览器支持的特性，并在html以class方式显示出来。



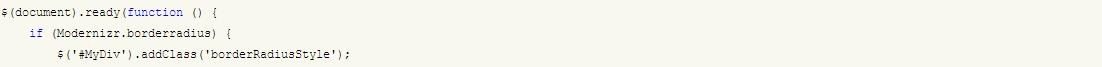
如上，支持就没有no-前缀，不支持就有。因此，可以根据这个来为不同特性支持情况添加样式。如，在支持shadow阴影的浏览器显示shadow，不支持的浏览器显示标准的边框。



* + - 1. js
* Modernizr除了添加相应的class到<html>元素以外，还提供一个全局的Modernizr JavaScript对象，提供了不同的属性来表示某种新特性在当前浏览器下是否支持。如，用于判断浏览器是否支持canvas。



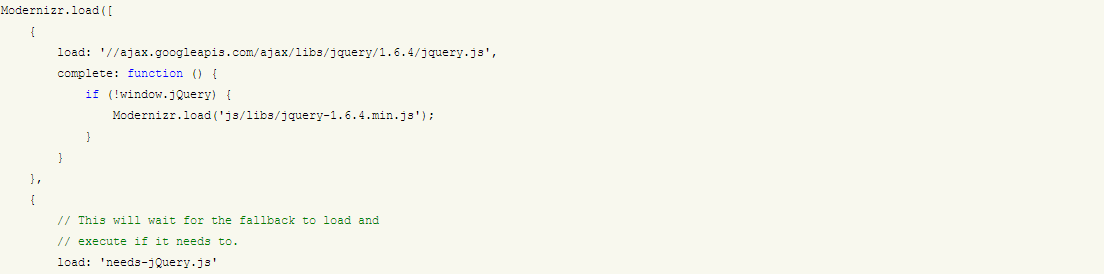
* 也可以用来探测是否支持CSS3特性。



* 使用Modernizr加载Script脚本。如果测试成功支持的话，就加载yep属性设置的脚本，如果不支持就加载nope属性设置的脚本，不管是否支持，both属性里设置的脚本都会加载的。



* 还引用的第三方脚本。



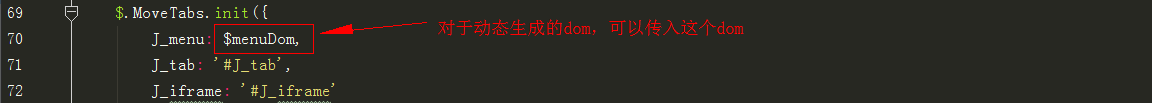
* 1. jquery.jplaceholder
  2. jquery.pngFix.js

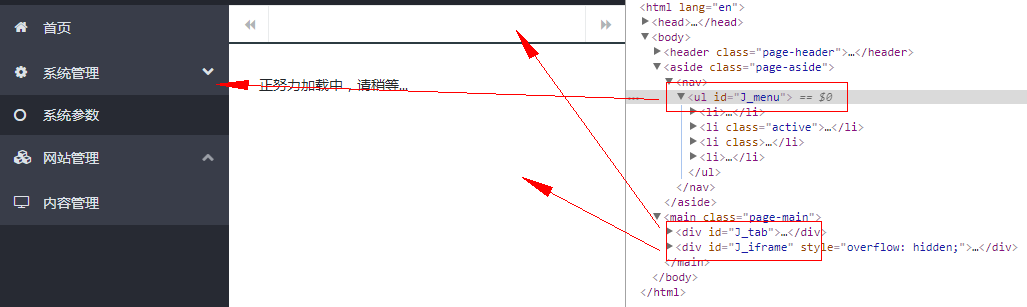
PNG透明插件

1. 自写
   1. $.MoveTabs

**1.使用**

（1）初始化，需要传入3个标签：menu、tab、content的容器。





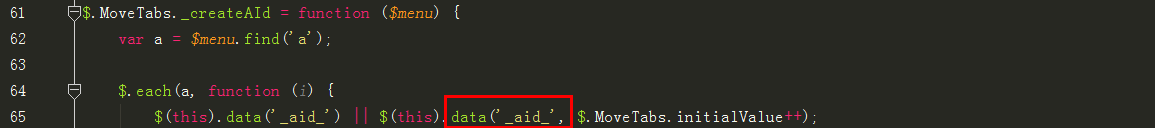
（2）插件内部直接就对menu中的a标签绑定了事件，就可以直接使用了，不需要再单独绑定事件。



（3）会自动生成tab中content中的内容，也不需要再操作。

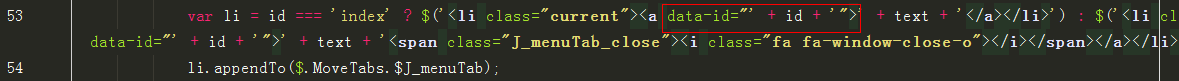
**2.代码**

（1）核心在于如何判断menu和tab之间的对应关系。内部使用\_createAId在初始化的时候，给menu下的每个a标签设置1个\_aid\_。



（2）menu下a的点击事件函数是\_J\_menuSide，但内部还是调用的\_iframeBuild方法，这里面是主要的逻辑。

首先检查a标签是否已有对应的tab，没有，则新生成1个，新生成的tab会绑定1个data-id，等于a的\_aid\_。



此时还会生成iframe。



然后调用\_showTab方法移动tabs，保证当前的tab在合适的位置显示。

（3）\_showTab方法会保证移动tabs的ul到合适位置，然后再调用\_current方法显示当前tab。



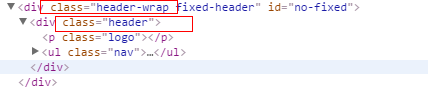


* 1. 页面
     1. 固定顶部导航条
        1. 效果
* 当滚动条超过顶部80px时，导航条减为60px，固定在顶部，且字体变化和下面的横线有一定的视差。





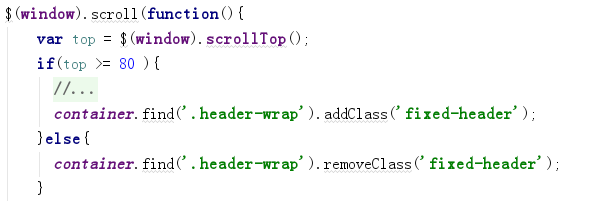
* + - 1. 实现
         1. html
* 为了实现tab与header不同步移动效果，用了header-wrap和header两层标签，但都是动画0.5秒。



* + - * 1. css

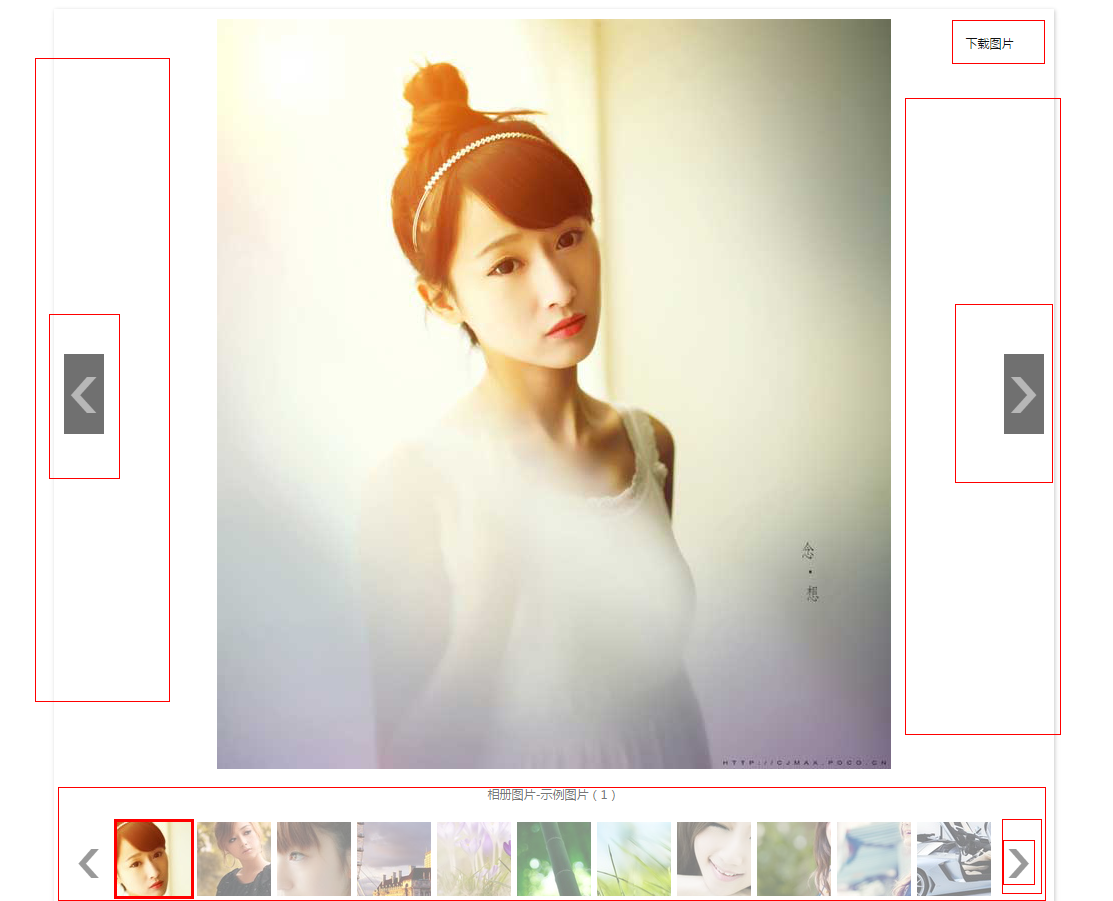


* + - * 1. js
* 顶部静态高度为80px，当滚动条据顶部大于80px时，给header-wrap加fixed-header样式。

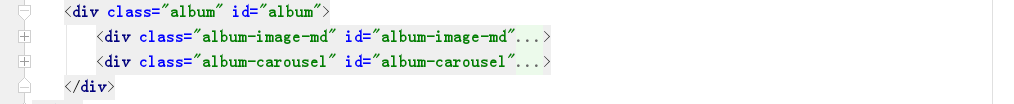


* 1. pic-ralate
     1. 微缩图相册
        1. 外观

分为2部分，上面为大图片，下面为微缩图。



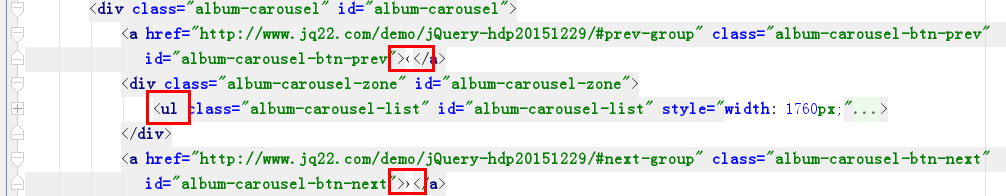
* + - 1. html结构
* 大图和微缩图并列。



* 大图部分，每次只显示1张图片，在p.album-image-bd（背景）中。ul.album-image-nav等都是绝对定位。ul.album-image-nav平时并不现实，只有在鼠标hover时才显示。



* 微缩图部分，pre/ul/next都是浮动定位，div.album-carousel-zone是微缩图的可视区域，ul. album-carousel-list包含所有图片长度大于div.album-carousel-zone，是绝对定位。



* + - 1. js

有2个js文件，album.js和Carousel.js。

* + - * 1. Album.js

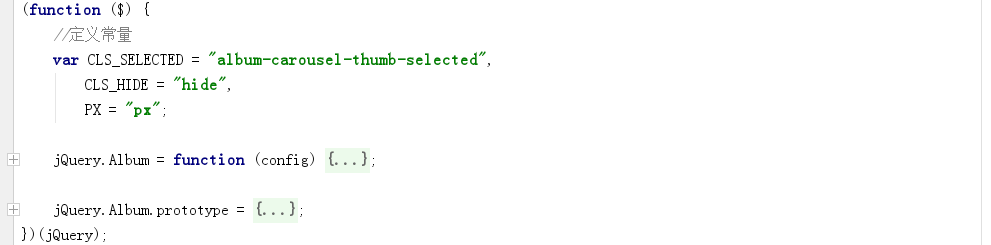
整体思路

* 主要实现的功能有：（1）点击微缩图，更换对应的大图。（2）向前/后换页。（3）保证图片原来的长宽比。
* 实现核心在于：（1）将大图片地址放在微缩图上的a标签中，将index保存为变量，管理大图和微缩图高亮。（2）通过预加载图片获取原来的长宽高和容器的长宽高，求比例来保证不失真。

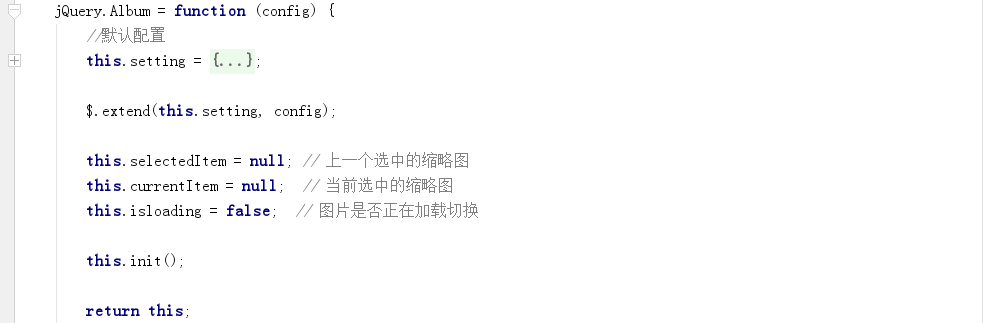
代码结构

* 定义jQuery全局扩展方法，使用时直接实例化即可。





* 定义3个常量，如上图。
* 全局扩展方法jQuery.Album中只定义了默认配置、变量，并调用init()方法。



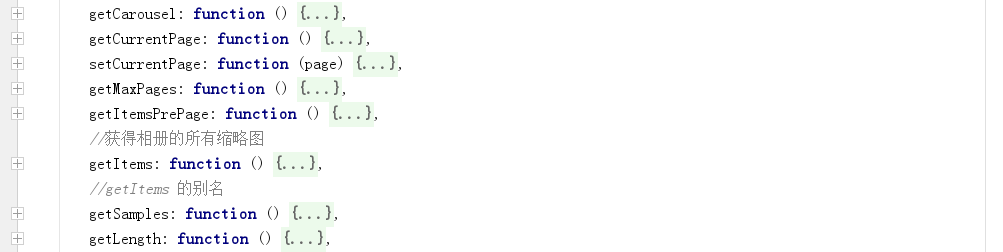
* 所有方法定义在jQuery.Album.prototype中，包括以下几类：
* 核心方法



* 获取dom的辅助方法



* 跟微缩图相关的辅助方法

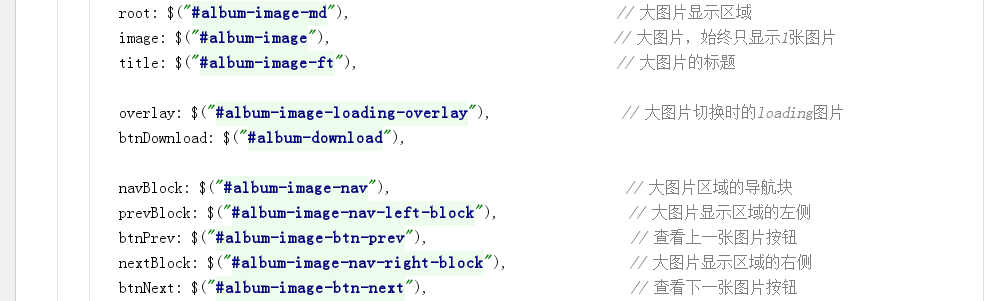


* 其他辅助方法

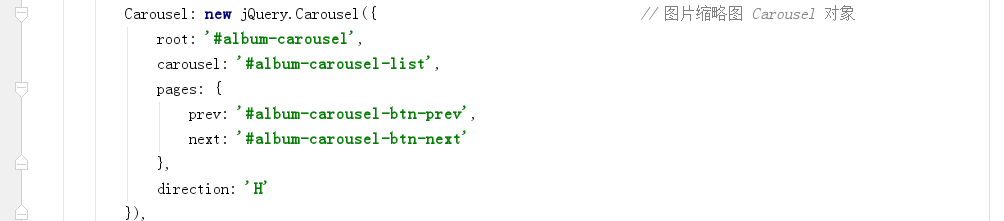


默认配置

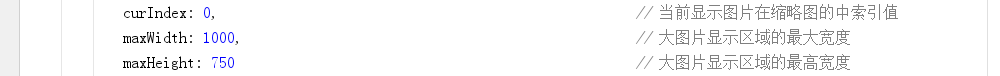
* 获取大图片显示区域中图片、按钮等dom对象。



* 实例化微缩图对象，将微缩图需要的dom对象传入。



* 设置默认index等。



init

实现3个功能：选中对应的微缩图，修改大图，绑定一些dom节点事件方法。

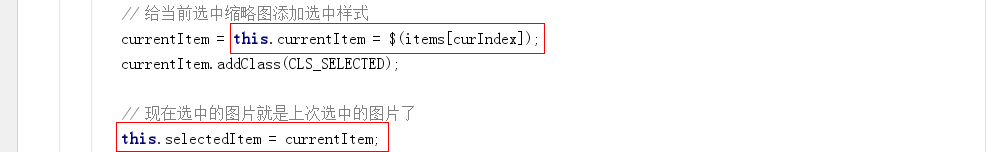


select

* 初始化时，上一个和当前选中的微缩图都是null，curIndex是0。

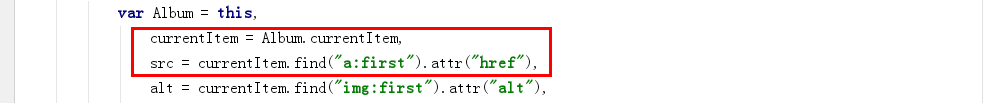


* 方法的核心是设置selectedItem和currentItem变量。



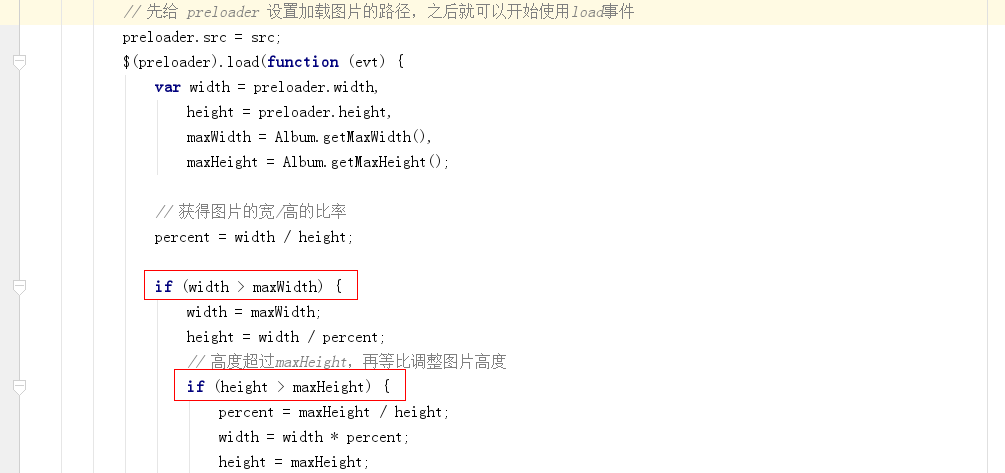
change

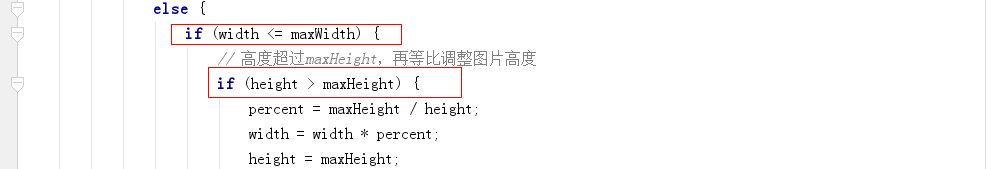
* 从微缩图的a标签中获取大图的src。



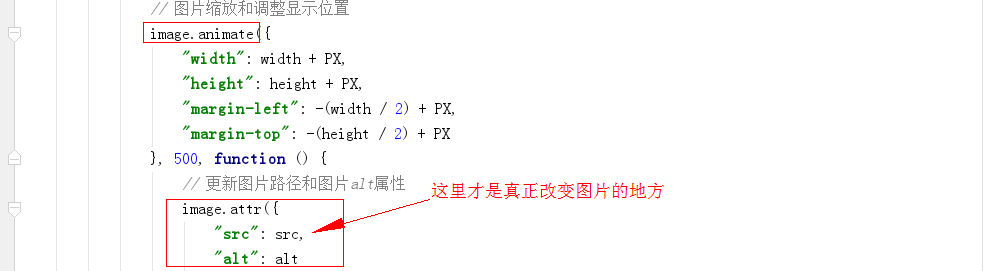
* 通过new Image()预加载图片，获取图片大小，计算宽高比，让图片不失真。





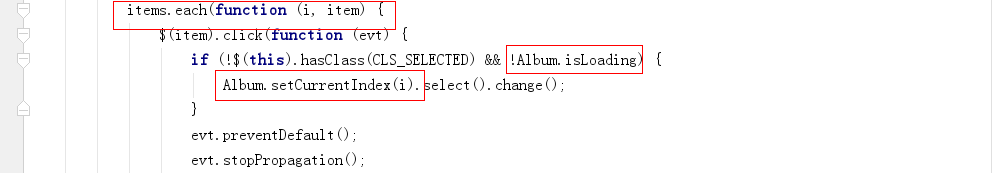


* 使用动画效果缩放图片



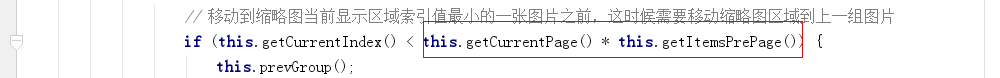
bindEvent

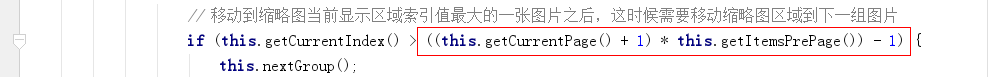
* 鼠标放到大图图片区域显示上下翻页按钮。
* 给每个微缩图绑定事件（关键在于i）。另外，图片预加载中不能点击微缩图再次更新图片。



prev/next

* 关键是获取当前的index，然后+/-1，再次调用this.select().change()。
* 如果当前index不在微缩图当前页时，需要对微缩图换页。





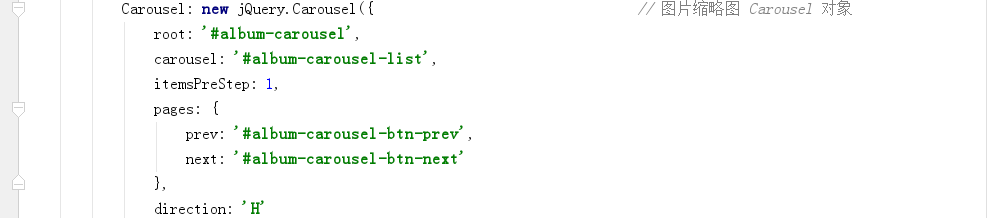
* + - * 1. Carousel.js

整体思路

* 核心方法是move，通过设置微缩图中的ul（绝对定位）的left/top值实现移动。
* 2种移动方法：分页移动或者是按item（可以多个）移动。
* 通过init初始化方法获取、计算配置（一开始就计算好最大移动数量等作为配置），并绑定事件方法。
* 可以通过设置，在最后一页时是否转到下一组图片组。

代码结构

* 定义jQuery全局扩展方法，使用时直接实例化即可。





* 只有setting这个配置属性变量，没有其他属性变量。

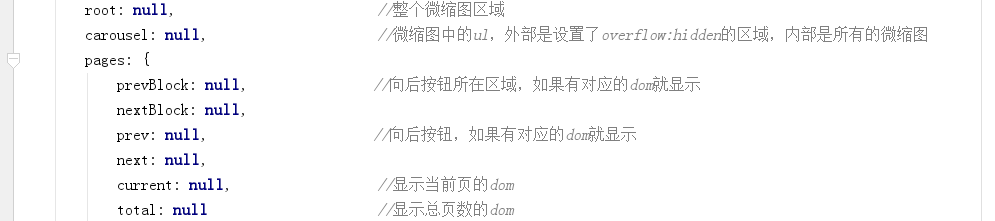


* 方法都在jQuery.Carousel.prototype中定义。



默认配置

* dom相关配置。

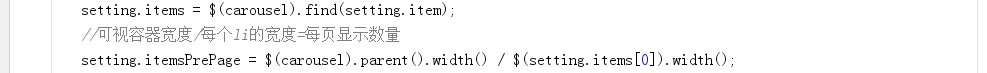


* 页码数量/方法。

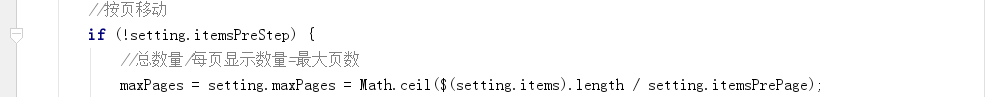


init

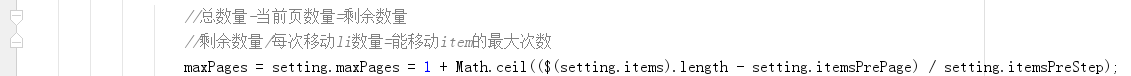
* 计算每页显示数量。



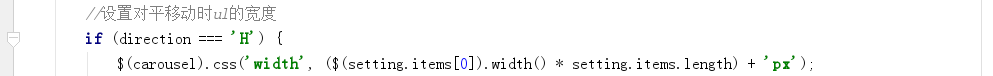
* 按页移动，计算总页数。



* 按item移动，

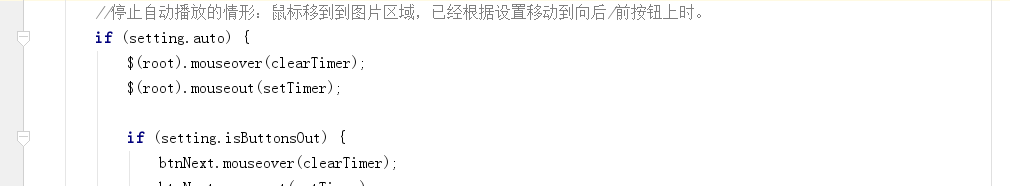


* 设置对平移动时ul的宽度



bind

* 停止自动播放的情形



next/prev

* 设置当前页。



* 调用current实现移动等。



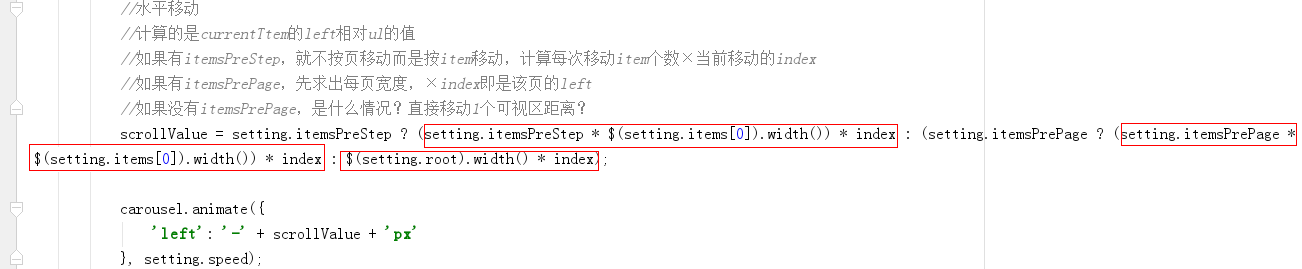
current

* 调用move实现页码移动。



move

* 核心方法。
* 水平移动，计算ul的left值。



* 垂直移动，计算ul的top值。



deleteItems

* 删除内部<img>src链接不是jpg图片的a标签。

auto

* 自动轮播，每次都要清除之前的定时器。



* 只有1页，不轮播。



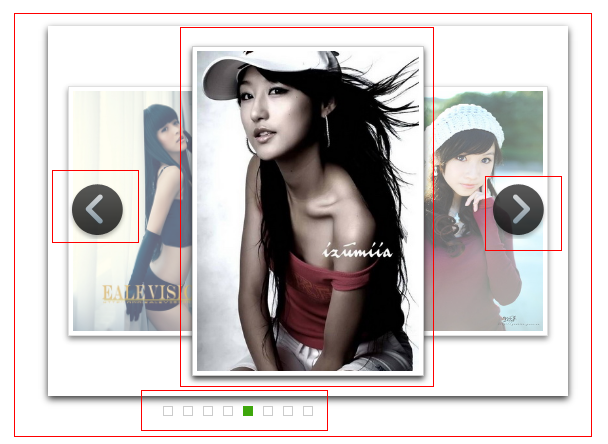
* 轮播要么按页，要么按item，和前面统一。



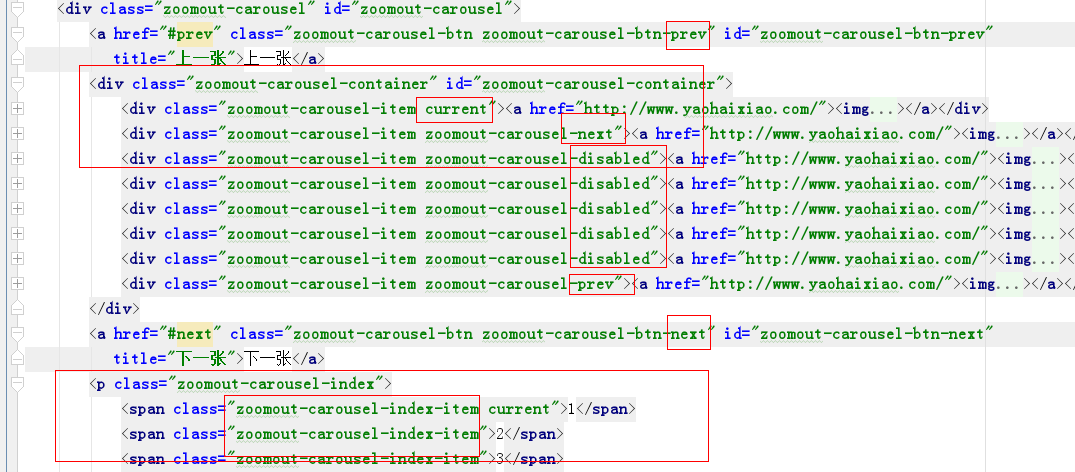
* 循环调用。



* + 1. 带伸缩效果图册
       1. 外观



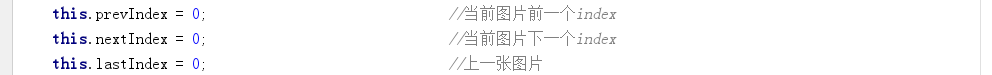
* + - 1. html结构



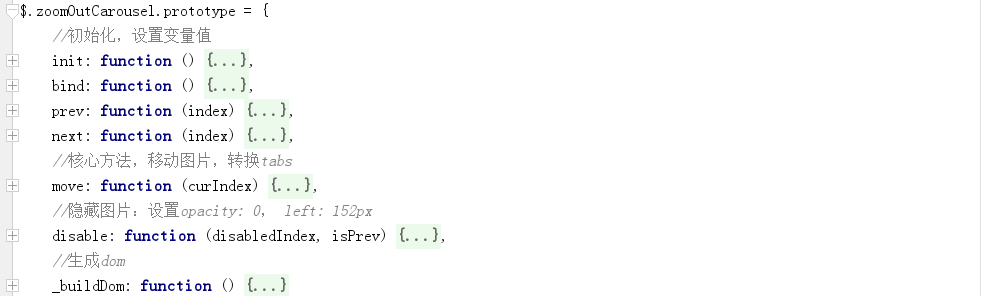
* + - 1. js
         1. 思路
* 核心在于确定当前显示图片index，计算出preindex和nextindex，其他图片隐藏。隐藏图片在当前图片后面，加上动画，就会出现想要的效果。
  + - * 1. 结构



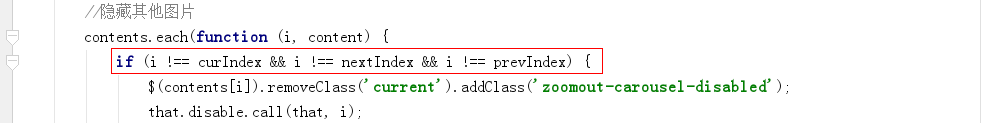
* 同时定义静态方法扩展和对象扩展。
* 静态方法中为定义变量。



* 方法定义在原型方法中。



* + - * 1. 代码分析
* 核心代码在move方法中，首先隐藏除当前/prev/next外的图片。



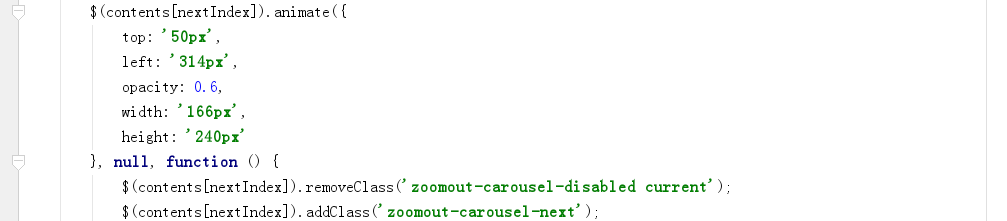
* 调整图片大小。

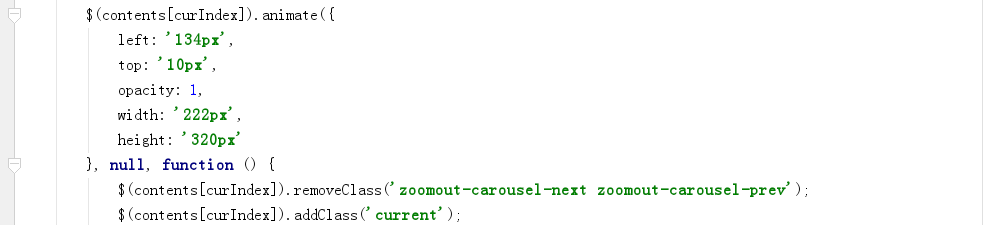






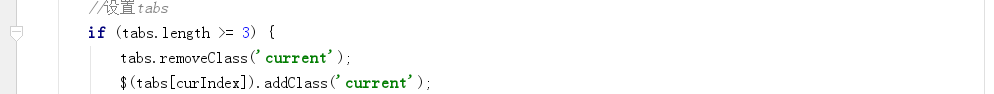
* 调整图片容器的大小和位置。







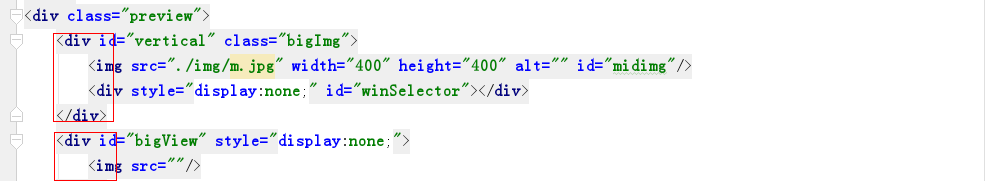
* 设置tabs的样式。



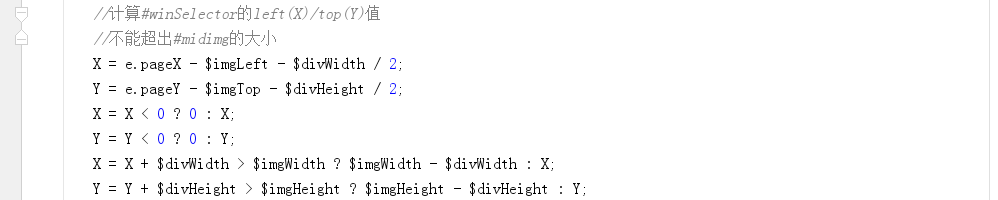
* 无论是init/prev/next，确定index原理一样，就是判断最大值和最小值，超出就是要循环。



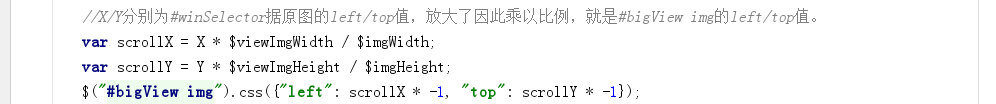
* + 1. 图片放大效果
       1. html结构
* 最外层相对定位，winSelector为放大镜，绝对定位。bigView绝对定位，位于preview外侧。



* + - 1. js分析
* 计算winSelector的left/top值，不能超出#midimg的大小。



* 放大效果直接扩展原图width/height，不用新图，bigView中的img同比例扩展，关键在于确定bigView中的img相对bigView的left/top，就是根据winSelector的left/top同比例计算得出。



* + 1. 背景反差图片浏览效果
  1. 时间轴效果
     1. 实现1
        1. html结构
* 内容部分：.container>section，每个section就是1个时间轴节点对应的段落。
* 时间轴：ul.timeline>li.bar+li.node\*n，每个li.node就是时间轴1个节点，li.bar则为标记当前内容段落对应节点的位置。
  + - 1. css
* .timeline为fixed，width为1px，height由js根据内容与window的高度比计算。
* .timeline .bar为absolute，width也为1px，height由js根据内容与window的高度比计算。
* .timeline .node为absolute，什么都没有，left偏移到.timeline上，圆点由:before生成。
  + - 1. js
* 首先根据内容，生成node。
* 给node绑定点击事件，能够指引到对应的内容上。
* 设置.bar的高度，能够和内容段落成比例。
* 给window绑定scroll和resize事件，实现如上效果。
  1. 流星雨效果

+(function () {

var starListBox = $('.mod-star-list');

//var sectionBox = $('.mod-bg-layer');

starListBox.height(4000);

var starListInner = starListBox.find('.inner');

var meteorSeed = 100;

var meteorHTML = '<div class="meteor"><div class="box"></div></div>';

var starListHeight = starListBox.height();

var starListWidth = starListBox.width();

for (var i = 0; i < meteorSeed; i++) {

var meteorEl = $(meteorHTML);

var scaleRandom = Math.random() \* 1.2 + 0.3;

var rotateRandom = Math.random() \* 10 - 2.5 + 'deg';

meteorEl.css({

'-webkit-transform': 'scale(' + scaleRandom + ') rotate(' + rotateRandom + ')',

'-moz-transform': 'scale(' + scaleRandom + ') rotate(' + rotateRandom + ')',

'transform': 'scale(' + scaleRandom + ') rotate(' + rotateRandom + ')',

'top': (Math.random() + Math.floor(i / 20)) \* 800 + 'px',

'right': Math.random() \* starListWidth + 'px'

});

meteorEl.find('.box').css({

'-webkit-animation-delay': Math.random() \* 10 + 's',

'-moz-animation-delay': Math.random() \* 10 + 's',

'animation-delay': Math.random() \* 10 + 's',

'-webkit-animation-duration': Math.random() \* 2 + 3 + 's',

'-moz-animation-duration': Math.random() \* 2 + 3 + 's',

'animation-duration': Math.random() \* 2 + 3 + 's'

});

meteorEl.appendTo(starListInner).waypoint(function (direction) {

$(this).find('.box').addClass('action');

}, {

offset: '100%',

triggerOnce: true

});

}

})();