一、概述

1 什么是jQuery

- jQuery 是一个高效、精简并且功能丰富的 JavaScript 工具库
- jQuery极大的简化了JavaScript 编程

2 什么是JS类库

- 它就是一些函数的集合，就是把特定效果的代码写好，你只需要在用的时候要用很少的代码去调用。
- 起主导作用的是你的代码，由你来决定何时使用类库。

3 常见JS类库

- jQuery
- ExtJS
- prototype.js
- zepto.js

4 jQuery的优势

- 轻量、开源、免费、易于学习
- 兼容性处理
- 强大的选择器
- 链式操作
- 便捷的DOM操作
- 可靠的事件机制
- 封装了简单易用的Ajax操作
- 丰富的插件

5 jQuery的版本

```
1.x.x的版本是可以 支持 IE6~IE8的
2.x.x到3.x.x 的版本，不在兼容IE8以及以下浏览器
```

6 jQuery 相关网站

- 官网 https://jquery.com/
- 文档API： http://jquery.cuishifeng.cn/index.html

二、jQuery基础

1 安装

下载到本地，再引入

下载地址：https://jquery.com/download/

```
<script src="jquery-3.3.1.min.js"></script>
<script>
    //注意，一定在引入jQuery之后，再使用jQuery提供的各种操作
</script>
```

或者 直接使用CDN

```
<script src="https://cdn.bootcss.com/jquery/3.3.1/jquery.js"></script>
<script>
    code...
</script>
```

BootCDN jQuery各个版本地址： https://www.bootcdn.cn/jquery/

根据浏览器版本加载不同版本的jQuery：

```
<!--IE9及以上以及其他浏览器-->
<!--[if gt IE 8]><!-->
<script src="../../dist/jquery-3.1.1.min.js"></script>
<!--<![endif]-->
<!--IE8以及以下浏览器-->
<!--[if lte IE 8]>
<script src="../../dist/jquery-1.12.4.min.js"></script>
<![endif]-->
```

2 jQuery对象（核心函数）

`$`是`jQuery`的别名

```
jQuery('.item').show()
//等同于
$('.item').show()
```

`$`对象可以用作选择器获取元素，还可以创建DOM对象

3 文档就绪事件

```
$(document).ready(function(){

   // 开始写 jQuery 代码...

});
```

上述写法可以简写

```
$(function(){

   // 开始写 jQuery 代码...

});
```

4 连贯操作

```
//对象可以连贯调用
$(dom).find('img').css('border','1px solid #ccc').css('color',
'red').prop('src', '1.jpg').toggle()
```

5 jQueryDOM和jsDOM

- 通过原生js获取的dom对象，我们称之为jsDOM或者原生DOM
- 通过jQuery选择器获取的dom对象，我们称之为 jQuery DOM
- jQuery DOM本质上 是由 jsDOM组成的集合，是个类数组对象。可以通过 [索引] 获取其中的 jsDOM
- `$(jsDOM)` 可以转为 jQuery DOM

三、选择器

通过选择器，可以获取到页面元素。jQuery具有强大的选择器，跟CSS3选择器类似

1 基本选择器

```
#id              根据给定的ID匹配一个元素
element          根据给定的元素标签名匹配所有元素
.class          根据给定的css类名匹配元素。
*              匹配所有元素
selector1,selector2,selectorN     将每一个选择器匹配到的元素合并后一起返回
```

2 层级选择器

```
ancestor descendant     在给定的祖先元素下匹配所有的后代元素
parent>child          在给定的父元素下匹配所有的子元素
prev+next              匹配紧接在 prev 元素后的 next 元素
prev~siblings          匹配 prev 元素之后的所有 siblings 元素
```

3 过滤选择器

```
:first              获取第一个元素
:not(selector)     去除所有与给定选择器匹配的元素
:even             匹配所有索引值为偶数的元素，从 0 开始计数
:odd             匹配所有索引值为奇数的元素，从 0 开始计数
:eq(index)          匹配一个给定索引值的元素
:gt(index)          匹配所有大于给定索引值的元素
:lang             选择指定语言的所有元素。1.9+
:last              获取最后个元素
```

```
:lt(index)          匹配所有小于给定索引值的元素
:header              匹配如 h1, h2, h3之类的标题元素
:animated          匹配所有正在执行动画效果的元素
:focus               匹配当前获取焦点的元素
:root              选择该文档的根元素    1.9+
:target           选择由文档URI的格式化识别码表示的目标元素     1.9
```

4 内容选择器

```
:contains(text)     匹配包含给定文本的元素
:empty             匹配所有不包含子元素或者文本的空元素
:has(selector)     匹配含有选择器所匹配的元素的元素
:parent             匹配含有子元素或者文本的元素
```

5 可见性选择器

```
:hidden          匹配所有不可见元素，或者type为hidden的元素
:visible     匹配所有的可见元素
```

6 属性选择器

```
[attribute]              匹配包含给定属性的元素
[attribute=value]     匹配给定的属性是某个特定值的元素
[attribute!=value]     匹配所有不含有指定的属性，或者属性不等于特定值的元素
[attribute^=value]     匹配给定的属性是以某些值开始的元素
[attribute$=value]     匹配给定的属性是以某些值结尾的元素
[attribute*=value]     匹配给定的属性是以包含某些值的元素
[attrSel1][attrSel2][attrSelN]      复合属性选择器，需要同时满足多个条件时使用
```

7 子元素选择器

```
:first-child         匹配所给选择器( :之前的选择器)的第一个子元素
:first-of-type       结构化伪类，匹配E的父元素的第一个E类型的孩子          1.9+
:last-child              匹配最后一个子元素
:last-of-type        结构化伪类，匹配E的父元素的最后一个E类型的孩子      1.9+
:nth-child()         匹配其父元素下的第N个子或奇偶元素
:nth-last-child()     选择所有他们父元素的第n个子元素。计数从最后一个元素开始到第一个
1.9+
:nth-last-of-type()      选择的所有他们的父级元素的第n个子元素，计数从最后一个元素到第一个
1.9+
:nth-of-type()       选择同属于一个父元素之下，并且标签名相同的子元素中的第n个      1.9+
:only-child              如果某个元素是父元素中唯一的子元素，那将会被匹配
:only-of-type            选择所有没有兄弟元素，且具有相同的元素名称的元素      1.9+
```

8 表单选择器

```
:input              匹配所有 input, textarea, select 和 button 元素
:text              匹配所有的单行文本框
:password          匹配所有密码框
:radio              匹配所有单选按钮
:checkbox          匹配所有复选框
:submit               匹配所有提交按钮，匹配 type="submit" 的input或者button
:image              匹配所有图像域
:reset              匹配所有重置按钮
:button               匹配所有按钮
:file              匹配所有文件域
```

9 表单对象选择器

```
:enabled          匹配所有可用元素
:disabled          匹配所有不可用元素
:checked          匹配所有选中的被选中元素(复选框、单选框等，select中的option)
:selected          匹配所有选中的option元素
```

10 混淆选择器

```
$.escapeSelector(selector)  className或IDName是有特殊符号
```

四、使用筛选器获取元素

1 过滤

```
eq(index|-index)          获取当前链式操作中第N个jQuery对象，返回jQuery对象
first()                       获取第一个元素
last()                      获取最后个元素
filter(expr|obj|ele|fn)       筛选出与指定表达式匹配的元素集合。
not(expr|ele|fn)          从匹配元素的集合中删除与指定表达式匹配的元素
has(expr|ele)              保留包含特定后代的元素，去掉那些不含有指定后代的元素。
slice(start,[end])          选取一个匹配的子集
```

2 查找

```
children([expr])          取得一个包含匹配的元素集合中每一个元素的所有子元素的元素集合。
find(e|o|e)                   搜索所有与指定表达式匹配的元素。这个函数是找出正在处理的元素的后
代元素的好方法

parent([expr])              取得一个包含着所有匹配元素的唯一父元素的元素集合
parents([expr])               取得一个包含着所有匹配元素的祖先元素的元素集合（不包含根元素）
parentsUntil([e|e][,f]) 查找当前元素的所有的父辈元素，直到遇到匹配的那个元素为止
offsetParent()              返回第一个匹配元素用于定位的父节点。

next([expr])              取得一个包含匹配的元素集合中每一个元素紧邻的后面同辈元素的元素集合
nextAll([expr])               查找当前元素之后所有的同辈元素
nextUntil([e|e][,f])      查找当前元素之后所有的同辈元素，直到遇到匹配的那个元素为止
prev([expr])              取得一个包含匹配的元素集合中每一个元素紧邻的前一个同辈元素的元素集
合
```

```
prevall([expr])              查找当前元素之前所有的同辈元素
prevUntil([e|e][,f])     查找当前元素之前所有的同辈元素，直到遇到匹配的那个元素为止
siblings([expr])          取得一个包含匹配的元素集合中每一个元素的所有唯一同辈元素的元素集合

closest(e|o|e)             1.7* 从元素本身开始，逐级向上级元素匹配，并返回最先匹配的元素
```

3 串联

```
add(e|e|h|o[,c])           1.9* 把与表达式匹配的元素添加到jQuery对象中
andSelf()                 1.8- 加入先前所选的加入当前元素中
addBack()                  1.9+ 添加堆栈中元素集合到当前集合，一个选择性的过滤选择器。
contents()                 查找匹配元素内部所有的子节点（包括文本节点）
end()                     回到最近的一个"破坏性"操作之前
```

4 其他元素处理

```
is(expr|obj|ele|fn)          根据选择器、DOM元素或 jQuery 对象来检测匹配元素集合，如果其中至少有一个元素符合这个给定的表达式就返回true
map(callback)              将一组元素转换成其他数组（不论是否是元素数组）
```

五、DOM操作

1 内部插入

```
append(content|fn)          向每个匹配的元素内部追加内容
appendTo(content)         把所有匹配的元素追加到另一个指定的元素元素集合中
prepend(content|fn)          向每个匹配的元素内部前置内容
prependTo(content)         把所有匹配的元素前置到另一个、指定的元素元素集合中
```

2 外部插入

```
after(content|fn)          在每个匹配的元素之后插入内容
before(content|fn)          在每个匹配的元素之前插入内容
insertAfter(content)     把所有匹配的元素插入到另一个、指定的元素元素集合的后面
insertBefore(content)     把所有匹配的元素插入到另一个、指定的元素元素集合的前面
```

3 包裹

```
wrap(html|ele|fn)          把所有匹配的元素用其他元素的结构化标记包裹起来
unwrap()                  这个方法将移出元素的父元素
wrapAll(html|ele)          移出元素的父元素
wrapInner(html|ele|fn)     将每一个匹配的元素的子内容(包括文本节点)用一个HTML结构包裹起来
```

4 替换

```
replaceWith(content|fn)          将所有匹配的元素替换成指定的HTML或DOM元素
replaceAll(selector)          用匹配的元素替换掉所有 selector匹配到的元素
```

5 删除

```
empty()              删除匹配的元素集合中所有的子节点
remove([expr])     从DOM中删除所有匹配的元素
detach([expr])     从DOM中删除所有匹配的元素  这个方法不会把匹配的元素从jQuery对象中删除，因而可以在将来再使用这些匹配的元素。与remove()不同的是，所有绑定的事件、附加的数据等都会保留下来
```

6 复制

```
clone([Even[,deepEven]])     克隆匹配的DOM元素并且选中这些克隆的副本
```

六、属性和样式操作

1 元素属性操作

1.1 属性

```
attr(name|pro|key,val|fn)     设置或返回被选元素的属性值
removeAttr(name)              从每一个匹配的元素中删除一个属性
prop(n|p|k,v|f)                 获取在匹配的元素集中的第一个元素的属性值
removeProp(name)              用来删除由.prop()方法设置的属性集
```

1.2 class

```
addClass(class|fn)              为每个匹配的元素添加指定的类名
removeClass([class|fn])          从所有匹配的元素中删除全部或者指定的类
toggleClass(class|fn[,sw])     如果存在（不存在）就删除（添加）一个类
hasClass(class)                  检查当前的元素是否含有某个特定的类，如果有，则返回true
```

1.3 代码、文本、值

```
html([val|fn])          取得第一个匹配元素的html内容
text([val|fn])          取得所有匹配元素的内容
val([val|fn|arr])     获得匹配元素的当前值
```

2 元素样式操作

2.1 设置CSS

```
css(name|pro|[,val|fn])          访问匹配元素的样式属性
```

2.2 元素位置

```
offset()          获取匹配元素在当前视口的相对偏移
position()          获取匹配元素相对父元素的偏移
scrollLeft()     获取匹配元素相对滚动条顶部的偏移
scrollTop()          获取匹配元素相对滚动条左侧的偏移
```

2.3 元素尺寸

```
widht()              取得第一个匹配元素当前计算的宽度值（px）
height()          取得匹配元素当前计算的高度值（px）
innerWidth()     匹配元素内部区域宽度（包括补白、不包括边框）
innerHeight()     匹配元素内部区域高度（包括补白、不包括边框）
outerWidth()     匹配元素外部宽度（默认包括补白和边框）
outerHeight()     匹配元素外部高度（默认包括补白和边框）
```

七、jQuery事件机制

1 事件操作

1.1 页面载入事件

```
$(document).ready(function(){
  // 在这里写你的代码...
});
或者
$(function($) {
  // 你可以在这里继续使用$作为别名...
});
```

1.2 事件绑定

```
on(eve,[sel],[data],fn)     1.7+ 在选择元素上绑定一个或多个事件的事件处理函数
bind(type,[data],fn)     3.0- 请使用on()
one(type,[data],fn)     为每一个匹配元素的特定事件（像click）绑定一个一次性的事件处理函数
```

1.3 解除事件绑定

```
off(eve,[sel],[fn])     1.7+ 在选择元素上移除一个或多个事件的事件处理函数
unbind(t,[d|f])     3.0- 请使用off()
```

1.4 触发事件

```
trigger(type,[data])
在每一个匹配的元素上触发某类事件

triggerHandler(type, [data])
这个特别的方法将会触发指定的事件类型上所有绑定的处理函数。但不会执行浏览器默认动作，也不会产生事件冒泡
这个方法的行为表现与trigger类似，但有以下三个主要区别：
* 第一，他不会触发浏览器默认事件。
* 第二，只触发jQuery对象集合中第一个元素的事件处理函数。
* 第三，这个方法的返回的是事件处理函数的返回值，而不是据有可链性的jQuery对象。此外，如果最开始的jQuery对象集合为空，则这个方法返回 undefined 。
```

1.5 事件委派

```
live(type,[data],fn)    1.7-
die(type,[fn])    1.7-
delegate(s,[t],[d],fn)    3.0-
undelegate([s,[t],fn])    3.0-
全部移除了，请使用 on()
$(document).on('click', 'button', fn)
```

1.6 事件切换

```
hover([over,]out)    一个模仿悬停事件（鼠标移动到一个对象上面及移出这个对象）的方法
toggle([spe],[eas],[fn])    1.9- 用于绑定两个或多个事件处理器函数，以响应被选元素的轮流的 click 事件
```

2 事件列表

```
鼠标事件：
click([[data],fn])    单击
dblclick([[data],fn])    双击
mousedown([[data],fn])    鼠标按键按下
mouseup([[data],fn])    鼠标按键抬起
mousemove([[data],fn])    鼠标移动
mouseover([[data],fn])    鼠标悬停在元素
mouseout([[data],fn])    鼠标离开元素
mouseenter([[data],fn])    当鼠标指针穿过元素时，会发生 mouseenter 事件。该事件大多数时候会与mouseleave 事件一起使用。与 mouseover 事件不同，只有在鼠标指针穿过被选元素时，才会触发 mouseenter 事件。如果鼠标指针穿过任何子元素，同样会触发 mouseover 事件。
mouseleave([[data],fn])    当鼠标指针离开元素时，会发生 mouseleave 事件。该事件大多数时候会与mouseenter 事件一起使用。与 mouseout 事件不同，只有在鼠标指针离开被选元素时，才会触发 mouseleave 事件。如果鼠标指针离开任何子元素，同样会触发 mouseout 事件。

表单事件：
blur([[data],fn])    失去焦点
focus([[data],fn])    获取焦点
change([[data],fn])    变化
select([[data],fn])    输入框文本被选中
submit([[data],fn])    表单提示时
focusin([data],fn)    当元素获得焦点时，触发 focusin 事件。 focusin事件跟focus事件区别在于，他可以在父元素上检测子元素获取焦点的情况
focusout([data],fn)    当元素失去焦点时触发 focusout 事件。focusout事件跟blur事件区别在于，他可以在父元素上检测子元素失去焦点的情况。

键盘事件：
keydown([[data],fn])    键盘按键按下
keypress([[data],fn])    键盘按键按下，只有可输入字符按键才能触发
keyup([[data],fn])    键盘按键抬起

文档和窗口事件：
resize([[data],fn])    窗口尺寸变化
scroll([[data],fn])    窗口滚动
unload([[data],fn])    离开文档 1.8-

图片事件：
error([[data],fn])    加载错误 1.8-
```

3 事件对象

属性

```
event.currentTarget     在事件冒泡阶段中的当前DOM元素
event.data     当前执行的处理器被绑定的时候，包含可选的数据传递给jQuery.fn.bind
event.delegateTarget     1.7+ 当currently-called的jQuery事件处理程序附加元素
event.namespace     当事件被触发时此属性包含指定的命名空间
event.pageX     鼠标相对于文档的左边缘的位置
event.pageY     鼠标相对于文档的顶部边缘的位置
event.relatedTarget     在事件中涉及的其它任何DOM元素
event.result     这个属性包含了当前事件事件最后触发的那个处理函数的返回值，除非值是
undefined
event.target     最初触发事件的DOM元素
event.timeStamp     返回事件触发时距离1970年1月1日的毫秒数
event.type     事件类型
event.which     针对键盘和鼠标事件，这个属性能确定你到底按的是哪个键或按钮
```

方法

```
event.preventDefault()     阻止默认事件行为的触发
event.isDefaultPrevented()     根据事件对象中是否调用过 event.preventDefault() 方法来
返回一个布尔值
event.stopPropagation()     防止事件冒泡到DOM树上，也就是不触发的任何前辈元素上的事件处理函
数
event.isPropagationStopped()     检测 event.stopPropagation() 是否被调用过
event.stopImmediatePropagation()     阻止剩余的事件处理函数执行并且防止事件冒泡到DOM树上
event.isImmediatePropagation()     检测 event.stopImmediatePropagation() 是否被调用
过
```

八、动画效果

1. 基本效果

```
show([s,[e],[fn]])          显示隐藏的匹配元素
hide([s,[e],[fn]])          隐藏显示的元素
toggle([s],[e],[fn])     如果元素是可见的，切换为隐藏的；如果元素是隐藏的，切换为可见的
```

参数详解

```
speed     : 三种预定速度之一的字符串("slow","normal", or "fast")或表示动画时长的毫秒数值
(如：1000)
easing     : (Optional) 用来指定切换效果，默认是"swing"，可用参数"linear"
fn          : 在动画完成时执行的函数，每个元素执行一次。
```

2. 滑动效果

```
slideDown([s],[e],[fn])          通过高度变化（向下增大）来动态地显示所有匹配的元素
slideUp([s,[e],[fn]])          通过高度变化（向上减小）来动态地隐藏所有匹配的元素
slideToggle([s],[e],[fn])     通过高度变化来切换所有匹配元素的可见性
```

参数详解

```
speed    : 三种预定速度之一的字符串("slow","normal", or "fast")或表示动画时长的毫秒数值(如：1000)
easing   : (Optional) 用来指定切换效果，默认是"swing"，可用参数"linear"
fn       : 在动画完成时执行的函数，每个元素执行一次。
```

3 淡入淡出效果

```
fadeIn([s],[e],[fn])          通过不透明度的变化来实现所有匹配元素的淡入效果
fadeOut([s],[e],[fn])          通过不透明度的变化来实现所有匹配元素的淡出效果
fadeToggle([s,[e],[fn]])      通过不透明度的变化来开关所有匹配元素的淡入和淡出效果
fadeTo([[s],o,[e],[fn]])      把所有匹配元素的不透明度以渐进方式调整到指定的不透明度
```

参数详解

```
speed    : 三种预定速度之一的字符串("slow","normal", or "fast")或表示动画时长的毫秒数值(如：1000)
easing   : (Optional) 用来指定切换效果，默认是"swing"，可用参数"linear"
fn       : 在动画完成时执行的函数，每个元素执行一次。

opacity    : (用户fadeTo)一个0至1之间表示透明度的数字。
```

4 自定义动画

```
animate(p,[s],[e],[fn])
```

参数详解

```
params   : 一组包含作为动画属性和终值的样式属性和及其值的集合
speed    : 三种预定速度之一的字符串("slow","normal", or "fast")或表示动画时长的毫秒数值(如：1000)
easing   : 要使用的擦除效果的名称(需要插件支持).默认jQuery提供"linear" 和 "swing".
fn       : 在动画完成时执行的函数，每个元素执行一次。
```

5 动画控制

```
stop([c],[j])          停止所有在指定元素上正在运行的动画
delay(d,[q])          设置一个延时来推迟执行队列中之后的项目
finish([queue])          停止当前正在运行的动画，删除所有排队的动画，并完成匹配元素所有的动画
```

6 设置

```
//关闭页面上所有的动画
jQuery.fx.off = true;
```

九、工具和其他操作

1 jQuery 对象访问

```
each(callback)                  遍历jquery dom
size()                          1.8- 返回dom集合中的个数 同length
length                          返回dom集合中的个数
selector                        返回选择器
context                          返回原生js dom
get([index])                    获取dom集合中一个
index([selector|element])       索引值
toArray()                       转为纯数组
```

2 数组和对象操作

```
$.each(object,[callback])          遍历数组 对象
$.extend([d],tgt,obj1,[objN])      合并对象 合并到第一个参数
$.grep(array,fn,[invert])          过滤数组
$.makeArray(obj)                   把类数组对象变为数组
$.map(arr|obj,callback)              操作数组的每一单元
$.inArray(val,arr,[from])          判断值是否在数组中
$.merge(first,second)              合并数组
$.unique(array)                      去重
$.parseJSON(json)                  解析json
$.parseXML(data)                   解析xml
```

3 类型检测

```
$.contains(c,c)              判断一个元素是否是另一个元素的后代吗元素
$.type(obj)                  判断类型
$.isarray(obj)               是否为数组
$.isFunction(obj)            是否是function
$.isEmptyObject(obj)         是否为空对象
$.isPlainObject(obj)         是否为纯粹的对象
$.isWindow(obj)                是否是window对象
$.isNumeric(value)           1.7+是否是number
```

4 其他操作

```
$.trim(str)                       去除左右两边的空格
$.param(obj,[traditional])        把对象或数组 转为 特殊格式的字符串 序列化
```

十、jQuery 插件

1 jQuery 插件的网站

- http://plugins.jquery.com/ 官网
- http://www.jq22.com/ jQuery插件库
- http://www.htmleaf.com/ jQuery 之家
- http://www.jq-school.com/ jQuery-school

2 经典jQuery插件

2.1 ### select2 下拉框搜索插件

- 官网 https://select2.org/
- github https://github.com/select2/select2
- 用法

```
$(dom).select2()
$(dom).select({
  width:,
  data:,
  ajax:,
  ....
})
```

2.2 datetimepicker 时间日期插件

- github https://github.com/xdan/datetimepicker
- 文档 https://xdsoft.net/jqplugins/datetimepicker/
- 用法

```
//设置语言
$.datetimepicker.setLocale('zh');
//调用插件
$(dom).datetimepicker({
  datepicker:,
  timepicker:,
  format:"Y-m-d H:i",
  value:,
  ....
})
```

2.3 fullpage 全屏滚动插件

- 官网 https://alvarotrigo.com/fullPage/zh/
- github https://github.com/alvarotrigo/fullpage.js/
- 用法

```
<!--HTML部分-->
<div id="fullpage>
    <div class="section"></div>
    <div class="section">
        <div class="slide"></div>
        <div class="slide"></div>
        <div class="slide"></div>
    </div>
    <div class="section"></div>
    <div class="section"></div>
```

```
</div>
自定义的导航  卸载包裹元素的外面

<!--JS部分-->
<script>
    $("#fullpage").fullpage({
      navigation: true,
      sectionsColor: []
      .....
    })
</script>
```

2.4 lazyload 图片懒加载

- 官网 https://appelsiini.net/projects/lazyload/
- github https://github.com/tuupola/jquery_lazyload/tree/2.x
- 用法

```
$("#lazyWrapper img").lazyload()
```

2.5 layer 弹窗插件

- 官网 http://layer.layui.com/?alone
- github https://github.com/sentsin/layer/
- 用法

```
layer.alert()
layer.confirm()
layer.msg()
layer.load()
layer.tips()
layer.colse()
layer.open({
  type: ,
  title: ,
  content:
  ....
})
...
```

2.6 nice validator 表单验证

- 官网 https://validator.niceue.com/
- github https://github.com/niceue/nice-validator
- 用法

```
$("form").validator({

})
```

2.7 jQuery-easing

- 官网 http://gsgd.co.uk/sandbox/jquery/easing/
- github https://github.com/gdsmith/jquery.easing
- 用法

```
$(dom).hide(speed, easing, fn)
```

3 自定义jQuery 插件

jQuery.fn.extend() 给jQueryDom扩展方法

```
$.fn.extend({
  方法名:function(){}
})
//或者
$.fn.方法名 = function(){

}
```

jQuery.extend() 给jQuery 对象本身扩展方法

```
$.extend({
   方法名: function(){
   }
})
```