**选择元素**

**本节内容**

* DOM对象和jQuery对象的定义
* $()函数
* CSS选择符
* 自定义选择符
* DOM遍历方法
* 访问DOM元素

**DOM对象**

* jQuery最强大的特性之一就是它能够简化在DOM中选择元素的任务。
* DOM(Document Object Model，文档对象模型)是W3C组织推荐的处理可扩展置标语言的标准编程接口。可被 JavaScript 用来读取、改变 HTML、XHTML 以及 XML 文档。
* 它以对象树而非纯文本的形式来表现HTML的源代码。

**jQuery 对象**

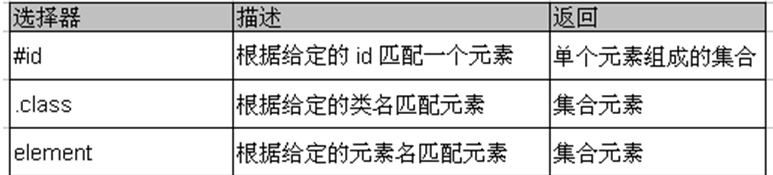
* **jQuery 对象就是通过 jQuery 包装 DOM 对象后产生的对象**
* jQuery 对象是 jQuery 独有的. 如果一个对象是 jQuery 对象, 那么它就可以使用 jQuery 里的方法:   
  例如$(“#tab”).html();
* jQuery 对象无法使用 DOM 对象的任何方法, 同样 DOM 对象也不能使用 jQuery 里的任何方法
* 建议约定：如果获取的是 jQuery 对象, 那么要在变量前面加上 $.
* **var $variable = jQuery 对象**
* **var variable = DOM 对象**

**$()函数**

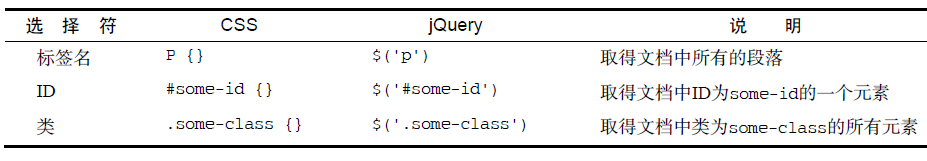
* 我们通过jQuery的各种选择符和方法取得的结果集合会被包装在jQuery对象中。
* 为了创建jQuery对象，就要使用$()函数。这个函数接受CSS选择符作为参数，充当一个工厂，返回包含页面中对应元素的jQuery对象。所有能在样式表中使用的选择符都可以传给个这个函数。
* $()函数也叫选择器函数。
* **选择器是 jQuery 的根基,** 在 jQuery 中, 对事件处理, 遍历 DOM 和 Ajax 操作都依赖于选择器。jQuery支持CSS规范1到规范3中的几乎所有的选择符。

**基本选择器**

* 基本选择器是 jQuery 中最常用的选择器, 也是最简单的选择器, 它通过元素 id, class 和标记名来查找 DOM 元素



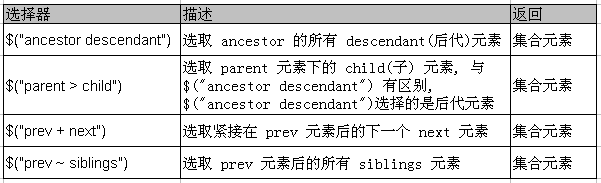
* 例



* 例
* 改变 id 为 one 的元素的背景色为 红色
* $("#one").css("backgroundColor","red");
* 改变元素名为 <p> 的所有元素的背景色为 # bbffaa，字体颜色为红色
* $("p").css({color:"red",backgroundColor:"#bbffaa"});
* 改变第一个<p>元素的背景色为红色
* $("p").eq(0).css("backgroundColor","red");
* 改变所有<h1>元素和 id 为 one 的元素的背景色为 # bbffaa
* $("h1,#one").css("backgroundColor","#bbffaa");

**层次选择器**

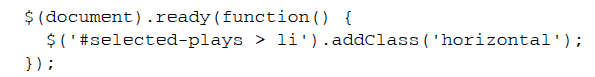
* 如果想通过 DOM 元素之间的层次关系来获取特定元素, 例如后代元素, 子元素, 相邻元素, 兄弟元素等, 则需要使用层次选择器



注意: (“#prev ~ div”) 选择器只能选择 “# prev ” 元素后面的同辈元素; 而 jQuery 中的方法 siblings() 与前后位置无关, 只要是同辈节点就可以选取

**层次选择器示例**

* <body> 内所有 <div>
* $(“body div")
* <body> 内的<div>
* $(“body>div")
* id 为 one 的下一个 <div>
* $("#one+div")
* id 为 two 的元素后面的所有兄弟<div>的元素$("#two~div")
* id 为 two 的元素所有 <p> 兄弟元素
* $("#two"). siblings("p")

**例题2-1**

**属性过滤选择器**

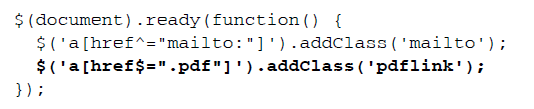
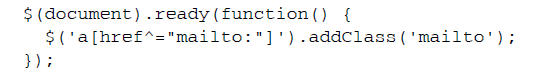
* 属性过滤选择器的过滤规则是通过元素的属性来获取相应的元素
* [attribute] 选取使用该属性的元素
* [attribute=value] 选取属性值为value的元素
* [attribute!=value] 选取属性值不等于value的元素
* [attribute^=value] 选取属性值以value开始的元素  
  [attribute$=value] 选取属性值以value结束的元素  
  [attribute\*=value] 选取属性值含有value的元素  
  [attribute|=value] 选取属性等于给定字符串或以该字符串为前缀(该字符串后跟一个连接字符“-”)的元素
* [attribute~=value] 属性值是空格分隔的多个字符串，其中一个字符串的值为value的元素[attribute1][attribute2][attributeN] 用属性选择器合并成一个复合属性选择器，满足多个条件。每选择一次，缩小一次范围。

**实例**

* $("div[id]") 选取拥有此属性id的元素
* $("div[title=test]") 选取属性title为"test"的<div>元素
* $("div[title!=test]") 选取属性title不等于"test"的<div>的元素（注意没有属性title的<div>元素也会被选取）
* $("div[title^=test]") 选取属性title以"test"开始的<div>元素
* $("div[title$=test]") 选取属性以"test"结束的<div>元素
* $("div[title\*=test]") 选取属性title含有"test"的<div>元素
* $("div[title|="en"]") 选取属性title等于en或者以en为前缀(该字符串后跟一个连字符‘-’)的元素
* $("div[title~="uk"]") 选取属性title用空格分隔的值中包含字符串uk的元素
* $(“div[class~=btn-info]”)选取类选择器中有btn-info样式的元素
* $("div[id][title$='test']") 选取拥有属性id，并且属性title以"test"结束的<div>元素

**例题**

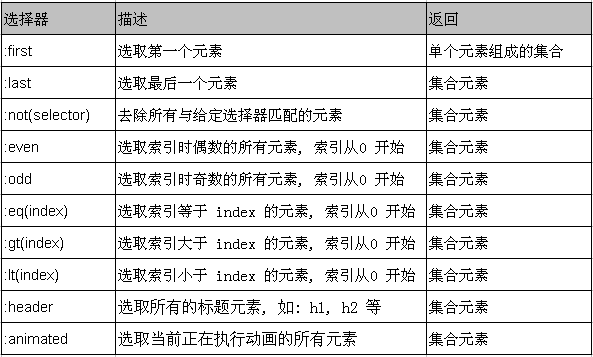
例题2.-3.2-4



**自定义选择符**

* 过滤选择器
* 过滤选择器主要是通过特定的过滤规则来筛选出所需的 DOM 元素, 该选择器都以 “:” 开头
* 按照不同的过滤规则, 过滤选择器又可分为基本过滤, 内容过滤, 可见性过滤, 属性过滤, 子元素过滤和表单对象属性过滤选择器.
* 注：尽量不要频繁的使用自定义选择符，以确保性能。

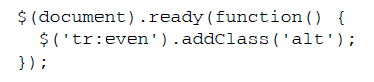
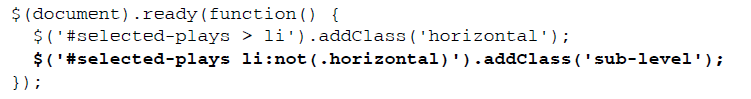
**基本过滤选择器**



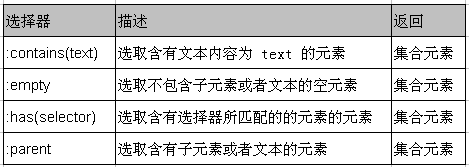
* 第一个 div 元素
* $("div:first")
* id不为 one 的所有p元素
* $("p:not('#one')")
* 索引值为偶数的 tr元素
* $(“tr:even")
* 索引值为大于 3 且为奇数的 p元素
* $(“p:gt(3):odd")
* 所有的标题元素
* $(":header")

**例题**

* 例题2.-2，2-6



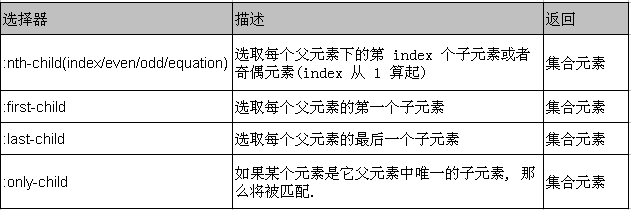
**内容过滤选择器**

* 内容过滤选择器的过滤规则主要体现在它所包含的子元素和文本内容上

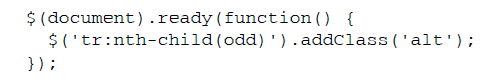
**内容过滤选择器示例**

* 含有文本 ‘di’ 的 p元素
* $("p:cotains(di)")
* 不包含子元素(或者文本元素) 的 div 空元素
* $(“div:empty")
* 含有 class 为 mini 元素的 p元素
* $("p:has(.mini)")

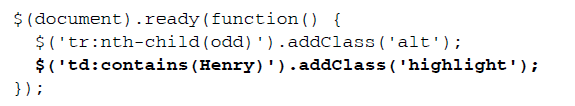
**子元素过滤选择器**



* nth-child() 选择器详解如下：
* (1) :nth-child(even/odd): 作为每个父元素下的索引值为偶(奇)数的子元素的元素
* (2):nth-child(2):作为每个每个父元素下的索引值为 2 的子元素的元素
* (3):nth-child(3n):作为每个父元素下的索引值是 3 的倍数 的子元素的元素
* (3):nth-child(3n + 1):作为每个父元素下的索引值是 3n + 1的子元素的元素
* 例题2-7

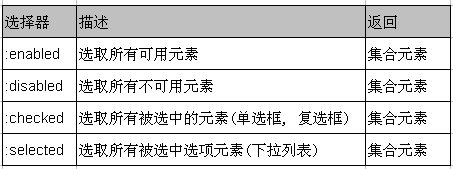


* 例题2-8

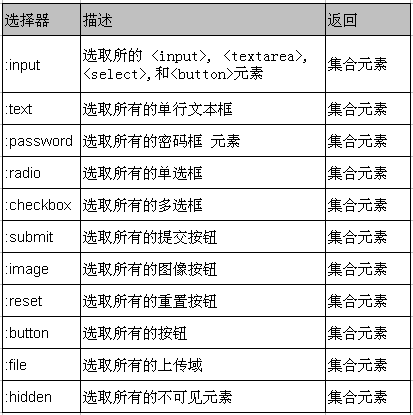


**表单对象属性过滤选择器**

* 此选择器主要对所选择的表单元素进行过滤

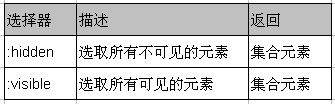


**表单选择器**

****

**可见性过滤选择器**

* 可见性过滤选择器是根据元素的可见和不可见状态来选择相应的元素



* 可见选择器 :hidden 不仅包含样式属性 display 为 none 的元素, 也包含文本隐藏域 (<input type=“hidden”>)和 visible:hidden 之类的元素

**DOM遍历方法**

**常用的遍历节点方法**

* jQuery在可使用选择符取得元素，也可以使用**遍历方法**取得元素。

例如：

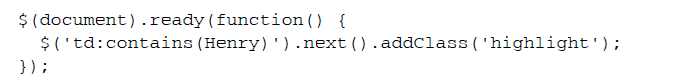
* $(‘tr:even’).addClass(‘alt’);
* 可以使用.filter方法重写成下面这样：
* $(‘tr’).filter(‘:even’).addClass(‘alt’);
* 这两种取得元素的方式在很大程度上可以互补。
* filter()的功能也十分强大，因为它可以接受函数参数。通过传入的函数，可以执行复杂的校验，以决定相应元素是否应该保留在匹配的集合中。
* 例：
* 为所有外部链接添加一个类。
* $('a').filter(function() {
* return this.hostname && this.hostname != location.hostname;
* }).addClass('external');

**常用DOM遍历方法**

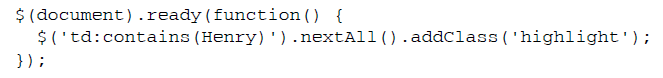
* 见表C2

例

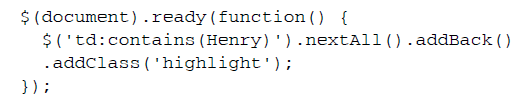
* 2-10



* 2-11



* 2-12



**连缀**

* 。在jQuery中，可以通过一行代码取得多个元素集合并对这些元素集合执行多次操作。jQuery的这种连缀能力不仅有助于保持代码简洁，而且在替代组合重新指定选择符时，还有助于提升脚本性能。
* 例2.14
* 2-15

方法连缀的原理

* 几乎所有jQuery方法都会返回一个jQuery对象，因而可连缀调用多个jQuery方法。

**访问DOM元素**

* 所有选择符表达式和多数jQuery方法都返回一个jQuery对象，但是有时我们仍然有可能需要在代码中直接访问DOM元素。
* 对于这些情形，jQuery提供了get()方法
* 例
* 如果想知道带有id=“my-element”属性的元素的标签名，应使用如下代码：
* var myTag=$(‘# my-element’).get(0).tagName;
* jQuery还为get方法提供了一种简写方式：
* var myTag=$(‘# my-element’)[0].tagName;