Seleccionando elementos para a súa manipulación

Nos apartados anteriores vimos moitos xeitos de utilizar a función $() de jQuery. As súas capacidades van dende a selección dos elementos DOM á definición de funcións que se executará cando o DOM está cargado. Agora imos a ver en detalle a forma en que os elementos DOM están asociados e de que xeito poderemos acceder a eles a través da función $(): mediante a selección a través dos selectores e a creación de novos obxectos no DOM.

O primeiro que necesitamos facer cando utilizando practicamente calquera método jQuery (tamén chamado comando jQuery) é o de seleccionar elementos da páxina para poder operar con eles.

Ás veces, o conxunto de elementos que desexamos seleccionar é fácil de describir, como "todos os elementos parágrafo na páxina". Noutros casos, requírese unha descrición máis complexa como "todos os elementos da lista que teñen a clase listElement e conteñen un vínculo". Afortunadamente, jQuery proporciona unha sólida selección da sintaxe: poderemos facilmente especificar calquera conxunto de elementos de xeito elegante e conciso. jQuery utiliza a sintaxe CSS que xa coñeces para seleccionar obxectos no DOM, e ademais ten algúns métodos personalizados que lle axudan a realizar tarefas comúns e moi complexas sobre ditos obxectos.

SUMARIO

Selectores básicos

Para a aplicación de estilos aos elementos da páxina Web, os desarrolladores estan familiarizados cun pequeno pero potente grupos de métodos de selección que funcionan en tódolos navegadores. Eses métodos inclúen a selección dun elemento polo ID, nome da clase CSS, marca html, etc. Aqui temos unha pequena lista de exemplos:

---

a — Este selector seleccionará todos os hiperenlaces (<a>).

#identificador - Este selector selecciona elementos que teñan como ID o identificador especificado.

.unhaClase - Este selector incluirá todos os elementos que empreguen a clase "unhaClase".

a#identificador.unhaClase - Este selector incluirá todos os hiperenlaces que teñan como ID identificador e como clase "unhaClase".

p a.unhaClase - Este selector incluirá a todos os hiperenlaces que empreguen a clase "unhaclase" e que estén dentro dun párrafo.

Podemos mesturar e combinar o selector de base para seleccionar obxectos chegando a un nivel moi detallado dentro da páxina. Polo tanto podemos empregar en jQuery o mesmo sistema que usamos en CSS.

Lembrar

Para seleccionar elementos utilizando jQuery, o que facemos é encapsular entre $(" ") o selector CSS.

Por exemplo:

$("p a.unhaClase")

Cunhas poucas excepcións, jQuery é plenamente compatible con CSS3, polo que empregaremos os mesmos métodos que usamos en CSS. Teña en conta que jQuery non depende da CSS da aplicación ou do navegador que está executando a páxina. jQuery correxirá as incompatibilidades que haxa entre navegadores para que nos deixe facer a selección de xeito correcto sempre cumprindo o estándar W3C.

Selectores fillo, contenedores e atributos

Para o uso de selectores avanzados, jQuery tamén da soporte á próxima xeración de CSS empregada en navegadores Firefox, IE Explorer 7, Safari ou outros navegadores máis modernos. Estes selectores avanzados inclúen a selección de fillos directos dalgúns elementos, elementos que están a continuación doutros no DOM, ou elementos que teñen atributos cunhas certas condicións.

Ás veces queremos soamente seleccionar os fillos directos dun elemento. Por exemplo poderíanos interesar seleccionar os elementos directos dunha lista, pero que non estén contidos dentro doutra (os que perteñecen á sublista).

Vexamos o código HTML seguinte:

```
<ul class="Lista">
  <li><a href="http://jquery.com">jQuery soporta</a>
    <ul>
      <li><a href="css1">CSS1</a></li>
      <li><a href="css2">CSS2</a></li>
      <li><a href="css3">CSS3</a></li>
    </ul>
  </li>
  <li>jQuery tamen soporta
    <ul>Seleccionando elementos para manipulacion
      <li>Selectores detallados</li>
      <li>Selectores de formulario</li>
    </ul>
  </li>
</ul>
```

Si por exemplo queremos seleccionar o hiperenlace á páxina jQuery, pero non o resto dos links, empregando CSS poderíamos facer algo semellante a:

```
ul.Lista li a
```

Desafortunadamente este selector seleccionará todos os hiperenlaces que están dentro dos elementos da lista que usa a clase Lista.

Unha aproximación máis avanzada consiste en usar os selectores fillo. Para elo empregaremos o símbolo > para separar o pai do fillo

```
p > a
```

Este selector seleccionará soamente os enlaces que son fillos directos (é dicir que non teñan entre eles outras marcas intermedias) dunha marca p.

Regresando ó noso exemplo, se pomos como selector:

```
ul.Lista > li > a
```

Este selector selecciona soamente enlaces que son fillos directos dun elemento li, que a súa vez é fillo directo dunha lista desordenada que emprega a clase Lista.

Os enlaces contidos na sublista son excluídos xa que esta sublista non emprega a clase Lista.

Os selectores de atributos son tamén extremadamente potentes. Por exemplo poderíamos asignar un comportamento especial soamente ós hiperenlaces que apunten a direccións externas ó noso sitio web.

Vexamos o seguinte exemplo:

```
<li><a href="http://jquery.com">jQuery supports</a>
    <ul>
      <li><a href="css1">CSS1</a></li>
      <li><a href="css2">CSS2</a></li>
      <li><a href="css3">CSS3</a></li>
      <li>Basic XPath</li>
    </ul>
</li>
```

O que fai que o hiperenlace sexa externo á nosa web según o exemplo é a presencia da cadea **http://** no atributo href.

Selectores de atributo:

```
$("[atributo]")
```

Isto é un selector básico por atributo. Seleccionará todos aqueles elementos que teñan como atributo o indicado. Por exemplo **$('[href]')** seleccionaría todos aqueles elementos que conteñan como atributo href.

```
$("[atributo=valor]")
```

Fará o mismo que o exemplo anterior, pero ademais que o atributo conteña o valor indicado como valor.

Poderíamos seleccionar enlaces que no seu atributo href comencen por: http:// co seguinte selector:

```
a[href^='http://']
```

Isto seleccionará tódolos enlaces cun href que comence exactamente por http://

O carácter ^ emprégase para indicar que a coincidencia ocurra ó comenzo. Este carácter é empregado por moitas expresións regulares para especifica-la coincidencia ó principio da cadea candidata.

Temos outras formas de empregar selectores de atributos. Por exemplo para seleccionar nun formulario un elemento que teña un atributo específico podemos empregar:

```
form[method]
```

Para indicar un valor específico nun atributo poderíamos usar algo como:

```
input[type=text]
```

Este selector seleccionará todos os elementos input que teñan como atributo type=text.

Ata agora vimos como seleccionar un atributo con valores ó comezo:

```
 div[title^='gali']
```

Isto selecciona todos os div que teñan un atributo title que comence por "gali". ¿E que pasa se queremos seleccionar un "atributo que finalice" por algo en concreto?

```
a[href$='.pdf']
```

Isto seleccionará todos aqueles hiperenlaces que fagan referencia a un fichero PDF.

E aquí temos un selector que permite localizar elementos que teñan en calquera parte dos seus atributos un valor:

```
a[href*=jquery.com]
```

Este selector seleccionará todos os hiperenlaces que teñan no atributo href o texto "jquery.com".

Por riba dos atributos, a veces queremos seleccionar un elemento soamente si contén outro elemento. Por exemplo, na lista anterior, si queremos aplicar algún comportamento ós elementos da lista que conteñan enlaces; jQuery soporta esta clase de seleccion con selector container:

```
li:has(a)
```

Este selector selecciona todos os elementos dunha lista que conteñan un hiperenlace.

Hai que diferenciar que isto non é igual que li a, que selecciona todos os enlaces que estan contidos dentro de elementos li

```
li a
```

Táboa de selectores básicos CSS soportados en jQuery:

Selector	Descripción
*****	Atopa calqueira elemento.
E	Atopa elementos coa etiqueta E.
E F	Atopa elementos coa etiqueta F que son fillos descendentes de E.
E > F	Atopa todos os elementos F que son fillos directos de E.
E+F	Atopa calqueira elemento F que é inmediatamente precedido polo seu irmán E.
E~F	Atopa calqueira elemento F precedido por calqueira irmán E.
E:has(F)	Atopa todos os elementos con marca E que teñen polo menos un descendente con marca F.
E.C	Atopa todos os elementos E que están empregando a clase C. Si se omite E poderíase escribir *.C
E#I	Atopa o elemento E con id I. Omitindo E poderíase escribir *#I ou ben #I supoñendo claro está que non pode haber elementos con id's repetidas no documento.
E[A]	Atopa todos os elementos E que conteñen un atributo A e calqueira valor nese atributo.
E[A=V]	Atopa todos os elementos E con atributo A de valor igual a V.
E[A^=V]	Atopa todos os elementos E con atributo A de xeito que o valor comence por V.
E[A$=V]	Atopa todos os elementos E con atributo A de xeito que o valor remate por V.
E[A*=V]	Atopa todos os elementos E con atributo A de xeito que o valor conteña V.

Selectores por posición

As veces necesitamos seleccionar elementos pola súa posición na páxina ou en relación con outros elementos. Por exemplo poderíamos necesitar seleccionar o primeiro enlace na páxina, ou calqueira outro párrafo, ou os derradeiros elemento de tódalas listas. jQuery soporta mecanismos para conseguir todas estas seleccións específicas. Por exemplo:

```
a:first
```

Seleccionará o primeiro hiperenlace da páxina.

E si queremos seleccionar calquer outro?

```
p:odd
```

Este selector selecciona todos os párrafos impares do documento.

Si queremos seleccionar todos os párrafos pares:

```
p:even
```

Outra forma:

```
li:last-child
```

Escollerá o último fillo li referente a un elemento pai. No noso exemplo das listas escollerá o derradeiro elemento li das lista ul.

Ver a seguinte táboa de selectores posicionais avanzados:

Selector	Descripcion
:first	O primeiro elemento atopado no documento. *li a:first* devolve o primeiro enlace que está contido nun elemento li.
:last	O derradeiro elemento atopado no documento. *li a:last* devolve o derradeiro enlace atopado dentro dun elemento li.
:first-child	O primeiro elemento fillo. Cando falamos de fillos estamos falando de grupos de irmáns (comparten o mesmo pai). Por exemplo *li:first-child* devolve o primeiro elemento de cada lista.
:last-child	O derradeiro elemento fillo. *li:last-child* devolverán o derradeiro elemento da lista.
:only-child	Devolve todos os elementos que non teñen irmáns, é dicir que son fillos únicos.
:nth-child(n)	O fillo número nth. Por exemplo *li:nth-child(2)* devolverá o segundo elemento de cada lista. Con nth-child(n) todos os fillos son contados independientemente do que sexan, e o elemento específico é seleccionado sempre e cando coincida có selector enlazado coa pseudo-clase.
:nth-child(even |odd)	Fillos pares ou impares. *li:nth-child(even)* devolve os elementos pares de cada lista.
:nth-child(Xn+Y)	O fillo número nth calculado a partir da fórmula proporcionada.Si Y é igual a 0, pode omitirse. *li:nth-child(3n)* devolve os elementos múltiplos de 3, mentras que *li:nth-child(5n+1)* devolve o elemento que ven cada grupos de 5 elementos.
:even e :odd	Elementos pares ou impares no documento. *li:even* devolverá cada elemento par das listas.
:eq(n)	Devolve o elemento con posición n. n ten como inicio de posición o 0. (Equivale a "igual a"). Con :eq(n) soamente o selector enlazado á pseudo-clase será tido en conta á hora de contar os elementos.
:gt(n)	Devolve os elementos con posición maior que n. (n ten como inicio de posición o 0, equivale a "maior que")
:lt(n)	Devolve os elementos con posición inferior a n. (Equivale a "menor que")

Vexamos o seguinte exemplo:

```
<table id="languages">
  <thead>
    <tr>
      <th>Language</th>
      <th>Type</th>
      <th>Invented</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Java</td>
      <td>Static</td>
      <td>1995</td>
    </tr>
    <tr>
      <td>Ruby</td>
      <td>Dynamic</td>
      <td>1993</td>
    </tr>
    <tr>
      <td>Smalltalk</td>
      <td>Dynamic</td>
      <td>1972</td>
    </tr>
    <tr>
      <td>C++</td>
      <td>Static</td>
      <td>1983</td>
    </tr>
  </tbody>
</table>
```

Si queremos obter todo o que conteñan as celdas que teñen os nomes de linguaxes de programación, e como vemos que son as primeiras celdas nas filas podemos por como selector:

```
table#languages tbody td:first-child  (é mais elegante esta sintaxe)
```

Ou tamén poderíamos por:

```
table#languages tbody td:nth-child(1)
// Escollerá as primeiras celdas fillas (dun mesmo pai, neste caso o
pai de cada grupo de fillas
// será o TR correspondente) contidas na marca tbody na táboa
languages.
```

Para obter as celdas que conteñen o tipo de linguaxe, cambiaremos o selector para:

```
:nth-child(2)
```

e para o ano no que se inventou, usaremos:

```
:nth-child(3) ou :last-child
```

Para obter a derradeira celda da táboa (a que conten o texto 1983), empregaremos:

```
td:last
```

Ainda que td:eq(2) devolve a celda que contén o texto 1995, td:nth-child(2) devolverá tódalas celdas que teñen o tipo de linguaxe de programación.

Pensade que cando falamos de fillos xxx-child estamos facendo subconxuntos (elementos que teñen un pai común). É dicir que si por exemplo escribimos o selector "li:first-child", o que jQuery fai é buscar todos os li que teñamos no documento; fará subconxuntos entre eles tomando como referencia que teñan o mesmo pai, e deses subconxuntos escollerá o primeiro fillo, que será o primeiro elemento de cada un dos subconxuntos.

Nota

Recordar tamén que :eq ten como orixe 0, mentras que :nth-child ten como orixe 1.

Selectores avanzados en jQuery

Os selectores CSS dannos unha gran flexibilidade e potencia para buscar os elementos desexados no documento, pero ás veces queremos seleccionar elementos en base a características que a especificación CSS non soporta.

Por exemplo si quixéramos seleccionar todos os checkbox que foron clicados polo usuario. Si queremos face-la selección por atributo non nos servirá xa que ten en conta o atributo que ten na definición do documento, non ten en conta a acción do usuario. Polo tanto jQuery ofrécenos un selector específico, **:checked** que filtra os elementos ós que se lles fixo click. Por exemplo:

```
input:checked
```

Seleccionará de tódolos elementos input soamente aqueles que foron chequeados polo usuario. Isto poderíase combinar co resto, por exemplo:

```
   :radio:checked
e
   :checkbox:checked
```

A continuación, unha táboa cos selectores avanzados ou selectores de filtro en jQuery:

Selector	Descripcion
:animated	Selecciona elementos que están baixo o control dunha animación.
:button	Selecciona calqueira botón. Equivaldría ó selector: "input[type=submit],input[type=reset],input[type=button]".
:checkbox	Selecciona elementos de tipo checkbox (input[type=checkbox]).
:checked	Selecciona elementos de tipo checkbox ou radio que estén chequeados (soportado por CSS).
:contains(texto)	Selecciona elementos que conteñan o texto pasado como parámetro. Non sirve para buscar contido HTML, soamente buscará texto que sexa visible no documento.
:disabled	Selecciona elementos que estén deshabilitados (soportado por CSS).
:enabled	Selecciona elementos que estén habilitados (soportado por CSS).
:file	Selecciona todos os elementos de tipo file (input[type=file]).
:header	Selecciona soamente elementos que son cabeceiras; por exemplo <h1> ... <h6>
:hidden	Selecciona os elementos que están ocultos.
:image	Selecciona elementos input de tipo imaxen. (input[type=image])
:input	Selecciona elementos do formulario (input, select, textarea, button).
:not(filtro)	Nega o filtro especificado.
:parent	Selecciona soamente elementos que teñen fillos (incluindo texto), pero non elementos vacíos.
:password	Selecciona elementos de tipo password (input[type=password]).
:radio	Selecciona elementos de tipo radio (input[type=radio]).
:reset	Selecciona botóns de tipo reset (input[type=reset] ou button[type=reset]).
:selected	Selecciona elementos do desplegable que estén seleccionados.
:submit	Selecciona botóns de tipo submit (button[type=submit] ou input[type=submit]).
:text	Selecciona elementos de tipo texto (input[type=text]).
:visible	Selecciona os elementos que están visibles.

Moitos dos selectores específicos en jQuery están relacionados cos formularios, permitíndonos especificar dun xeito moito máis elegante un tipo de elemento ou estado. Podemos combinar filtros de selección tamén. Por exemplo, si queremos seleccionar soamente os checkbox habilitados que nos que se fixo click para marcalos poderíamos usar:

```
:checkbox:checked:enabled
```

Uso do filtro :not

Si queremos negar un filtro, por exemplo para seleccionar os elementos input que non sexan checkbox:

```
input:not(:checkbox)
```

É importante recoñecer a diferencia entre **selectores de filtro**, os cais nos permiten detallar un conxunto de elementos aplicando criterios de selección (como os amosados anteriormente) e os selectores de búsqueda. Os **selectores de búsqueda** como o selector descendente (espacio en blanco), selector fillo (>), irmán (+) permítenos buscar elementos que teñen relación cós que xa están seleccionados.

Podemos aplicar o filtro :not ós selectores de filtro pero non ós de búsqueda. Por exemplo:

```
div p:not(:hidden)  é un selector válido, pero
div:not(p:hidden)  non o é.
```

No primeiro caso, todos os elementos p descendentes dun div e que non estén ocultos, serán seleccionados.

O segundo selector está incorrecto xa que intenta aplicar un :not a un selector que non é un filtro (o p en p:hidden non é un filtro).

Sobre os Selectores de Filtro

Para face-las cousas máis sinxelas, os selectores de filtro son identificados fácilmente por que todos comezan co caracter : ou un corchete [. Calquera outro selector non poderá ser usado con :not