



COMP 4021

Internet Computing

More on CSS Selectors

Dr. Kenneth LEUNG

Slides created by Prof. Dik Lun LEE

More Selectors: Class

```
<p class="important"></p>
```

```
<p></p>
```

```
<p class="warning"></p>
```

```
<p class="important warning"></p>
```

<p> belongs to two classes: «important» and «warning»

\$(**'p.important, p.warning'**) returns which <p> tags?

\$("ancestor descendant")

<form>

<label>Child:**</label>****<input** name="name" **/>**

<fieldset>

<label>Address:**</label>****<input** name="address"

**/>
**

<label>Age:**</label>****<input** name="age" **/></fieldset></form>**

Sibling to form: **<input** name="none" **/>**

<script>\$("form input").css("border", "2px dotted red");</script>

<input> tag with ancestor **<form>**

Child:

Address:

Age:

Sibling to form:

Child Selector: >

```
<ul class="topnav">
```

```
<li>Item 1</li>
```

```
<li>Item 2
```

```
<ul>
```

```
<li>Nested item 1</li>
```

```
<li>Nested item 2</li>
```

```
<li>Nested item 3</li>
```

```
</ul></li>
```

```
<li>Item 3</li>
```

```
</ul>
```

- Item 1
- Item 2
 - Nested item 1
 - Nested item 2
 - Nested item 3
- Item 3

** tag with parent which has class "topnav"**

```
<script>$("ul.topnav > li").css("border", "1px double red");</script>
```

Immediate Sibling Selector: +

<input> immediately preceded by <label>

`<script>$("label + input").css({"border":"2px dotted red",
"color":"blue"}).val("Enter value"); </script>`

```
<form>  
  <label>Name:</label><input name="name" />  
  <fieldset>  
    <label>Address:</label><input name="address" /><br/>  
    <label>Age:</label><input name="age" /></fieldset></form>  
<input name="none" />
```

Name:

Address:

Age:

What about:
`<script>$("form input").css("color",
"blue").val("Enter value")</script>`

Next Sibling Selector: ~

`$("label ~ input")`

- `<label>` does not have to be immediately preceding `<input>`
- But sibling means `<label>` and `<input>` belong to the same parent

Combining Several Selectors

- **\$("div > ul a")** reads:

All `<a>` elements which are **descendants** of `` elements which are **direct children** of `<div>` elements

```
<div>
  <ul><li> This is a sublist:
    <ul>
      <li> <a href=...>Click here</a></li>
    </ul>
  </li>
</ul>
<a href=...>Click here</a>
</div>
```

Which `<a>` would be selected?

Select and Action Example

- Select all elements within element with ID=orderedlist, and add the class "blue"

```
$(document).ready(function() {  
    $("#orderedlist > li").addClass("blue");  
});
```

```
<ol id="orderedlist">  
    <li class="blue">... </li>
```

```
$("#orderedlist > li").removeClass("blue");
```

```
<ol id="orderedlist">  
    <li >... </li>
```

For another example see: <https://jsfiddle.net/qb2n7h5L/2/>

Some Useful Selectors For Reference Only

<code>\$('#id')</code>	element with the specified ID
<code>\$('p')</code>	all elements with the specified name
<code>\$('.class')</code>	all elements with the specified class
<code>\$('*')</code>	all elements
<code>\$('p.class')</code>	<code><p></code> elements having the CSS class
<code>\$('p:first')</code>	<code>\$('p:last')</code> / <code>\$('p:odd')</code> / <code>\$('p:even')</code>
<code>\$('p')[1]</code>	gets the 2 nd <code><p></code> element (0 is first)
<code>\$('p a')</code>	<code><a></code> elements, descended from a <code><p></code>
<code>\$('p>a')</code>	<code><a></code> elements, direct child of a <code><p></code>
<code>\$('p+a')</code>	<code><a></code> elements, preceded by a <code><p></code> (next sibling)
<code>\$('div, p, a')</code>	<code><div></code> , <code><p></code> and <code><a></code> elements
<code>\$('li:has(ul)')</code>	<code></code> elements that have at least one <code></code> descendent
<code>\$(':not(p)')</code>	all elements but <code><p></code> elements
<code>\$('p:hidden')</code>	only <code><p></code> elements that are hidden
<code>\$('p:empty')</code>	<code><p></code> elements that have no child elements

Useful Selectors For Reference Only

E[a]	all E elements with attribute 'a' of any value
E[a=v]	all E elements with attribute 'a' exactly 'v'
E[a^=v]	all E elements with attribute 'a' starting with 'v'
E[a\$=v]	all E elements with attribute 'a' ending with 'v'
E[a*=v]	all E elements with attribute 'a' containing 'v'

\$(‘img’[alt])	 elements having an alt attribute
\$(‘a’[href^=http://])	<a> elements with href starting with ‘http://’
\$(‘a’[href\$=.pdf])	<a> elements with href ending with ‘.pdf’
\$(‘a’[href*=hkust])	<a> elements with href containing ‘hkust’

jQuery Functions for Formatting

<code>\$("p").css(property, value)</code>	Set property to value
<code>\$("p").html()</code> <code>\$("p").html(htmlString)</code>	Get the content of the first <code><p>...</p></code> Set the html contents of each <code><p>...</p></code>
<code>\$("input").val()</code>	For form <code><input></code>
<code>\$("p").addClass('class')</code>	Add <code>class="class"</code> property to all <code><p></code>
<code>\$("p").removeClass('class')</code>	Remove <code>class="class"</code> property from all <code><p></code>

Add Page Elements

- `$('#target').before('<p>Inserted before #target</p>');`
- `$('#target').after('<p>This is added after #target</p>');`
- `$('#target').append('<p>Goes inside #target, at end</p>');`
- `$('#target').wrap('<div></div>');`

Take Home Message

- CSS-like selectors are like a query language to the DOM to retrieve elements matching some fairly sophisticated conditions
- There are still many jQuery and autocomplete functions not covered
- Will cover jQuery ajax and server side programming (PHP) later in this course