
COMP 4021
Internet Computing

jQuery and Autocomplete Introduction

Dik Lun Lee

What does jQuery Look Like? How Easy is it?

```
$ (function() {  
    $("#greenbox").click(function() {  
        $("#greenbox").hide();  
        $("#redbox").show();  
    });  
    $("#redbox").click(function() {  
        $("#redbox").hide();  
        $("#greenbox").show();  
    });  
});
```

Try out the demo on course homepage, read the code, and imagine do the same using JavaScript.

Website Administration

Instructor: Harry Plantinga



Course objectives: With modern content management systems, complex, good-looking, and functional web sites can be constructed with little programming. This course presents an introduction to many of the topics needed for setting up and administering a Web site with a content management system.

Putting all this knowledge to good use, we will attempt to work with local non-profit organizations, setting up a website for them according to their specifications.

Website Administration

Instructor: Harry Plantinga



Course objectives: With modern content management systems, complex, good-looking, and functional web sites can be constructed with little programming. This course presents an introduction to many of the topics needed for setting up and administering a Web site with a content management system.

Putting all this knowledge to good use, we will attempt to work with local non-profit organizations, setting up a website for them according to their specifications.

Structure of a jQuery Function Call

```
$ (function() {  
  $("#greenbox").click(function() {  
    $("#greenbox").hide();  
    $("#redbox").show();  
  });  
  $("#redbox").click(function() {  
    $("#redbox").hide();  
    $("#greenbox").show();  
  });  
});
```

\$ (Selector) **.** action();

\$ Sign denotes jQuery function

Select the HTML element

. separator

Perform action on selected element

jQuery JavaScript Library

- ❑ You know a bit about JavaScript and DOM now and know it is not easy to write JavaScript to manipulate DOM
- ❑ JQuery is a JavaScript library (<http://jquery.com>)
 - And Like any other libraries, jQuery aims to implement popular operations in easy-to-use jQuery API, e.g., DOM manipulation, event handling, client-server interaction
- ❑ Free and open source; most popular JavaScript library in use today; means lots of support
- ❑ Reasonably small footprint
- ❑ Cross browser support
- ❑ Other toolkits: Yahoo UI Library (YUI), Google Web Toolkit, etc.

What jQuery Does?

- ❑ A "easy-to-use" API for DOM manipulation:
 - Select DOM elements using CSS-like selectors
 - Set properties of selected DOM elements
 - Create, delete, show, hide DOM elements
 - Defines event behavior (click, mouse movement, dynamic styles, animations, dynamic content)
 - While CSS separate content from style, jQuery separate behavior from structure (i.e., no need to worry about traversing a tree)
- ❑ You can do all of these using plain JavaScript, but jQuery is a lot easier to use. Why?
- ❑ Use CSS selector to **select a group of objects**, AND
- ❑ Apply an **operation to all objects**, OR
- ❑ **Iterate through each of them** to perform specialized action

Let's Begin: jQuery Ready Function

- Execute a function as soon as a page is fully loaded

```
<html> <head>
```

```
<script src="jquery-2.1.4.js"></script>
```

```
<script>
```

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

Your jQuery code here

```
});
```

```
</script>
```

```
</head><body>...</body></html>
```

Or abbreviated as:

```
$(function() {  
...  
});
```

What is the problem of executing a function before a page is fully loaded?

Select Objects and Perform Action (1)

`$(selector).methodName();`

`$("#div").addClass("blue");`

`$("#div").removeClass("blue");`

`.blue { color: blue; }`

`.red { color: red; }`

- `$("#div")` returns a jQuery object containing all DIV elements in DOM
- `addClass()` makes each DIV :
`<div class="blue"> I have a new color! </div>`

Select Objects and Perform Action (2)

`$(selector).methodName();`

<code>\$("p").click()</code>	Add click event (and event handler) to every <p> object
<code>\$("p").hide()</code>	Hide all <p> objects
<code>\$("p").show()</code>	Show all <p> objects
<code>\$("p").toggle()</code>	Toggle all <p> objects between "hide" and "show"

Example: jQuery in Action

```
<script type="text/javascript">
```

```
$(function(){
```

```
  $("p").html("Hello World !! ");
```

```
});
```

```
</script>
```

```
<body>
```

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

```
</body>
```

- What would be displayed?
- What if there are many <p>...</p> tags?
- What happens to <p>Existing text</p>
- How to achieve same result with pure JavaScript?

jQuery vs JavaScript

- ▣ Hide all DIVs

```
var divs = document.getElementsByTagName("div");  
  for (i=0 ; i<divs.length; i++) {  
    divs[ i ].style.display = "none";  
  }
```

```
$("div").hide();
```

jQuery vs JavaScript

- Click a button to display or hide a DIV

```
function toggle_visibility(id) {  
    var e = document.getElementById(id);  
    if (e.style.display != "none")  
        e.style.display = "none";  
    else  
        e.style.display = "block";  
    ... ..  
}
```

```
<button id="mybutton"  
    onclick="toggle_visibility('foo');">  
    Click to toggle visibility</button>  
<div id="foo">Hi</div>
```

```
<button id="mybutton">Click</button>  
$(function(){  
    $("#mybutton").click(function(){  
        $("#foo").toggle();  
        return false;  
    });  
});
```

Power of jQuery

By now, you can see that jQuery's power comes from:

- ❑ Powerful CSS selectors
- ❑ Perform action on a whole group of tags
- ❑ High-level, commonly used functions: `hide()`, `show()`, `toggle()`, etc.

Basic CSS Selectors (more to come next)

- ❑ `$("#id")` element with a specified ID
- ❑ `$("p")` all elements with the specified name
- ❑ `$(".class")` all elements with the specified class
- ❑ `$("*")` all elements

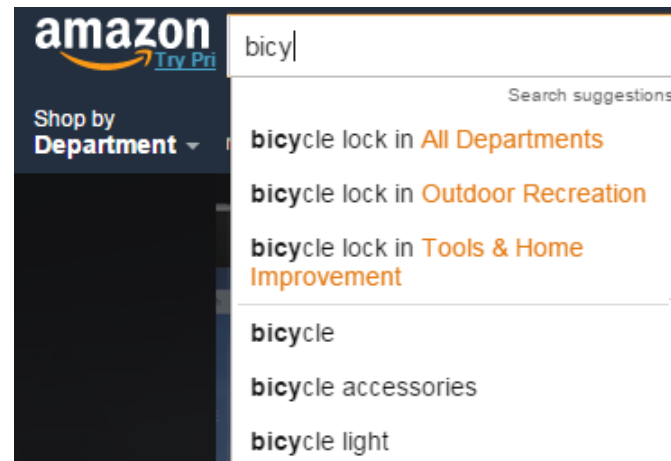
COMP 4021
Internet Computing

jQuery Autocomplete

Dik Lun Lee

What is Autocomplete?

- ❑ Autocomplete displays a list of "suggested" values as user types and allows user to a value to trigger action
 - E.g., Type in a query in Bing's search box and see how it works



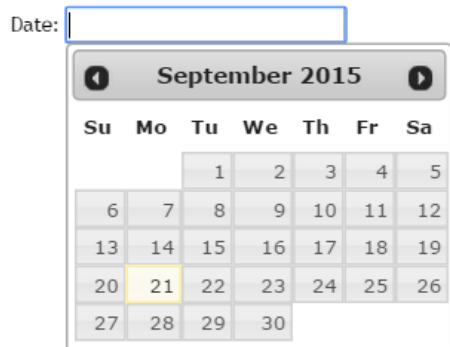
Doing Autocomplete in JavaScript

- ❑ Imagine how much work is required to implement autocomplete with JavaScript only:
 - Display an input box
 - Create event trigger for each keystroke
 - Display a text box (div) under the search box
 - Fill the box with suggestions, which most likely is obtained dynamically from the server (e.g., server returns the most popular queries matching the partial query user has typed in)
 - Handle "small things": Do not display suggestions when user is typing very fast; backspace also updates the list

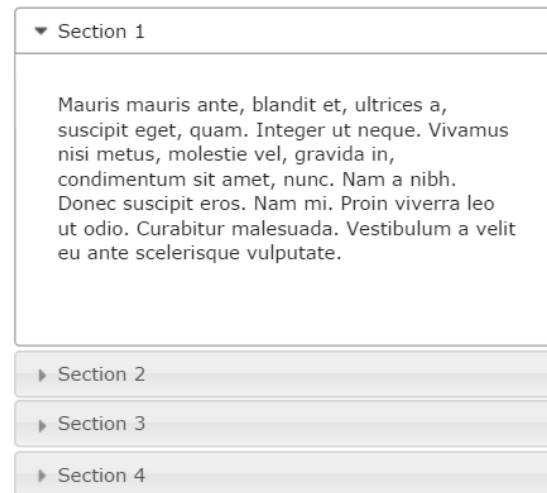
jQuery UI and Autocomplete Widget

- ❑ JQuery autocomplete is a **widget** under **JQuery UI**, a widget library built on jQuery
 - A text field widget tied to a text field
 - Needs a data source containing the values to be displayed
 - Download Autocomplete from: <http://jqueryui.com/download/>
- ❑ Other jQuery UI widgets:

Datepicker



Accordion / Collapsible Content

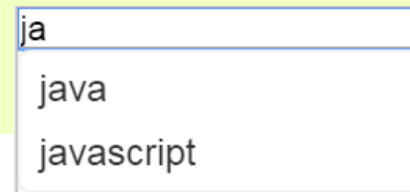


Data Source

□ Source of data could be:

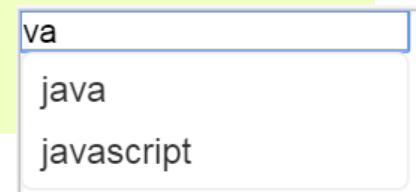
- **Static list** on client side, like a hardcoded array (see below)
- Dynamic list coming from server (discussed in future lecture)

```
var languages = ["java", "javascript", "perl", "python", "php"];  
$( "#searchBox" ).autocomplete( {  
    minLength: 2, // min # of chars input before suggestions are triggered  
    delay: 200, // wait 0.2 sec before trigger  
    source: languages  
});
```



ja

- java
- javascript



va

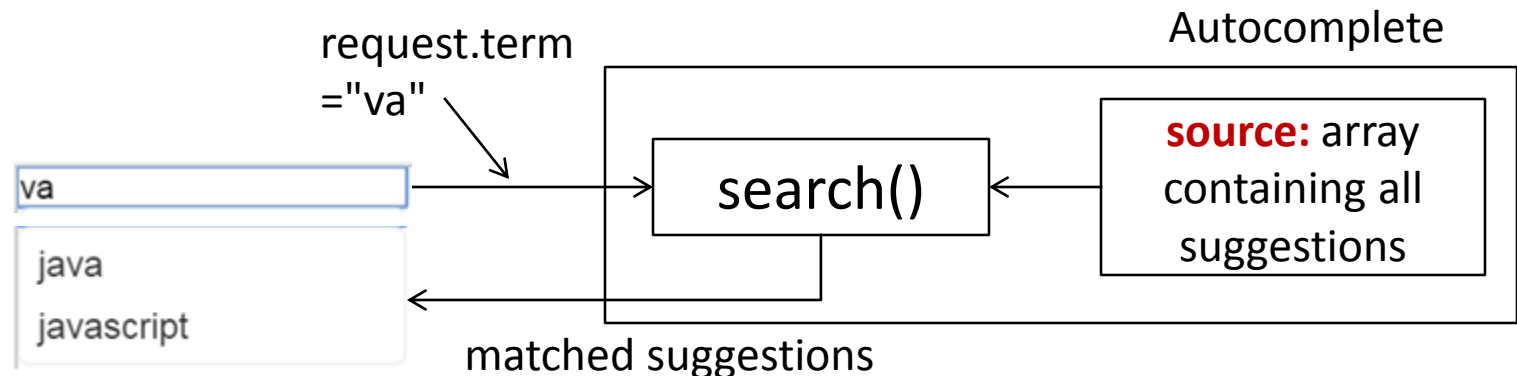
- java
- javascript

```
<input type="text" id="searchBox"  
    name="q" size="20" value="">
```

- The candidate list of suggestions is FIXED !!!
- By default, it performs substring matching

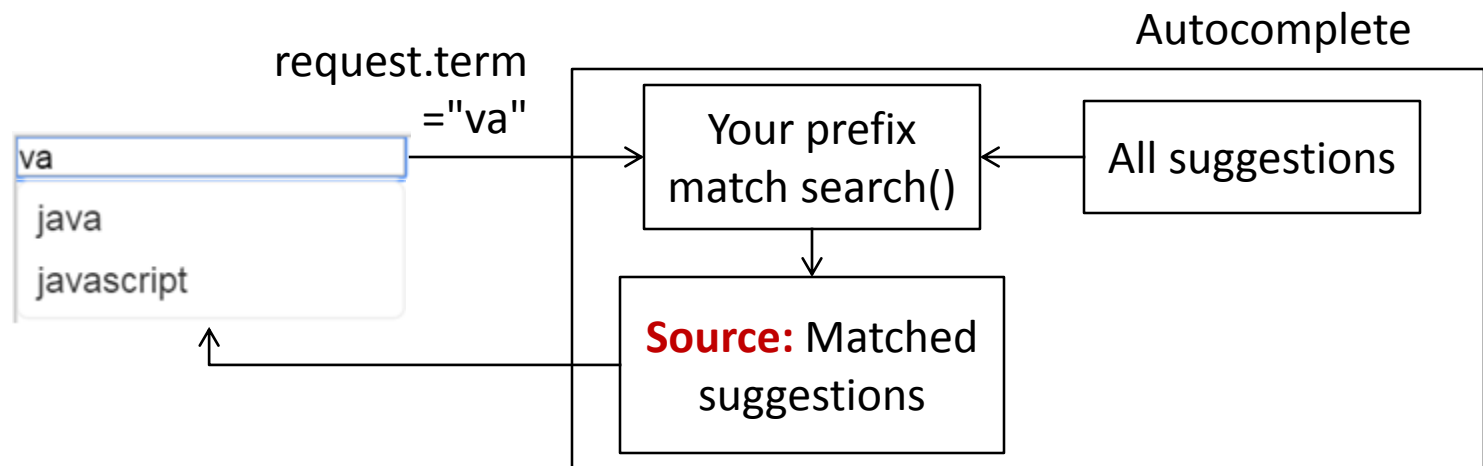
Using an Array as Data Source

- ❑ Autocomplete() will handle everything
- ❑ If you find autocomplete default behavior acceptable, this is the simplest way to use autocomplete()
 - Default display style of menu window
 - Matching request.term anywhere in suggestion



How to Customize Autocomplete

- ❑ There are many ways to customize autocomplete
- ❑ If you want request.term to match the prefix of a suggestion
 - 1) Disable builtin search() and implement your own
 - 2) Use one array to contain all suggestions and then implement your own search and put matching suggestions into another array
 - 3) Use the suggestion array as source



Example: Using Function as Source

```
$(function() {  
  var languages = ["java", "javascript", "perl", "python", "php" ];  
  $( "#searchBox" ).autocomplete( {  
    source: function ( request, response ) {  
      var languages1 = [];  
      for (var i = 0; i < languages.length; i++) {  
        if (languages[i].startsWith(request.term)) {  
          languages1.push(languages[i]); } }  
      response(languages1); },  
  }); };
```

```
<div class="ui-widget">  
  <input type="text" id="searchBox" name="q" size="20" value="">  
</div>
```

❑ Note: This code is NOT the best for implementing the desired behavior and not meant for your homework to follow

Example: Using label and value

```
$(function() {  
  $( "#searchBox" ).autocomplete( {  
    minLength: 2, // min # of chars input before suggestions are triggered  
    delay: 200, // wait 0.2 sec before trigger  
    source: [  
      { label: "java", value: "JAVA" },  
      { label: "javascript", value: "JAVASCRIPT" },  
      { label: "perl", value: "PERL" },  
      { label: "python", value: "Python" },  
      { label: "php", value: "PHP" } ] );  
});
```

- ❑ Display value when you click on label
- ❑ You do not need the language array

Other Customization

- ❑ How to bold the matched (or unmatched) substring in a suggestion?
- ❑ How to trigger an action when a suggestion is selected?
- ❑ How to display the selected suggestion in the text box?
- ❑ Other questions:
 - How to obtain suggestions from a server
 - How to change the style of the menu window?

Take Home Message

- ❑ JQuery is a popular JavaScript library because it makes JavaScript/DOM programming simpler
- ❑ Provide a CSS-like selectors to specific elements to which actions are applied (compared to navigating DOM using JavaScript only)
- ❑ jQuery provides convenient API for popular functions
- ❑ Still need many practices but better than pure JavaScript
- ❑ Autocomplete is very helpful for user interaction
- ❑ jQuery UI makes autocomplete manageable despite its complexity