COMP4021 Internet Computing
Tutorial 3
jQuery Autocomplete Widget

Oct. 4th, 2021

Content

- Introduction to jQuery, jQuery autocomplete
- Basic example: default autocomplete widget
- Customization 1: Change the search keyword programmatically when a suggestion is selected
- Customization 2: Customize the style of autocomplete suggestions

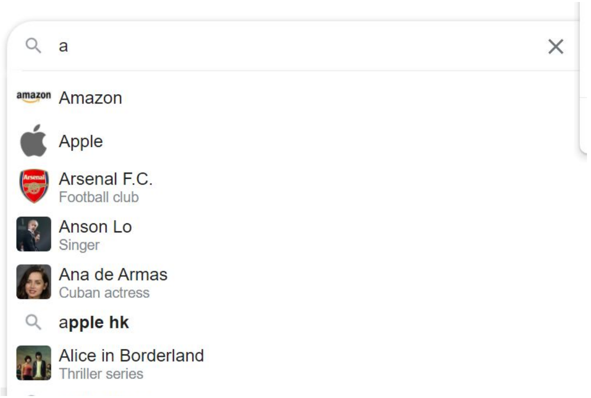
Note: You do not need to submit this lab, but it provides you with a starting point for the assignment project.

jQuery

- **What?** A lightweight, "write less, do more", JavaScript library.
- **Why?** To greatly simplify JavaScript programming.
- **How?** By calling methods that wrap up common tasks, a single line of code can accomplish what needs to be done with many lines of code in plain JavaScript.
- **Features** (https://www.w3schools.com/jquery/jquery_intro.asp):
  - HTML/DOM manipulation
  - CSS manipulation
  - HTML event methods
  - Effects and animations
  - AJAX
  - Utilities

jQuery autocomplete widget

- Autocompletion:
  - A mechanism frequently used in modern websites.
  - To provide the user with a list of suggestions for which he/she has typed in a text box.
  - The user can then select an item from the list, which will be displayed in the input field.
  - This feature prevents the user from having to enter an entire word or a set of words.



jQuery autocomplete widget

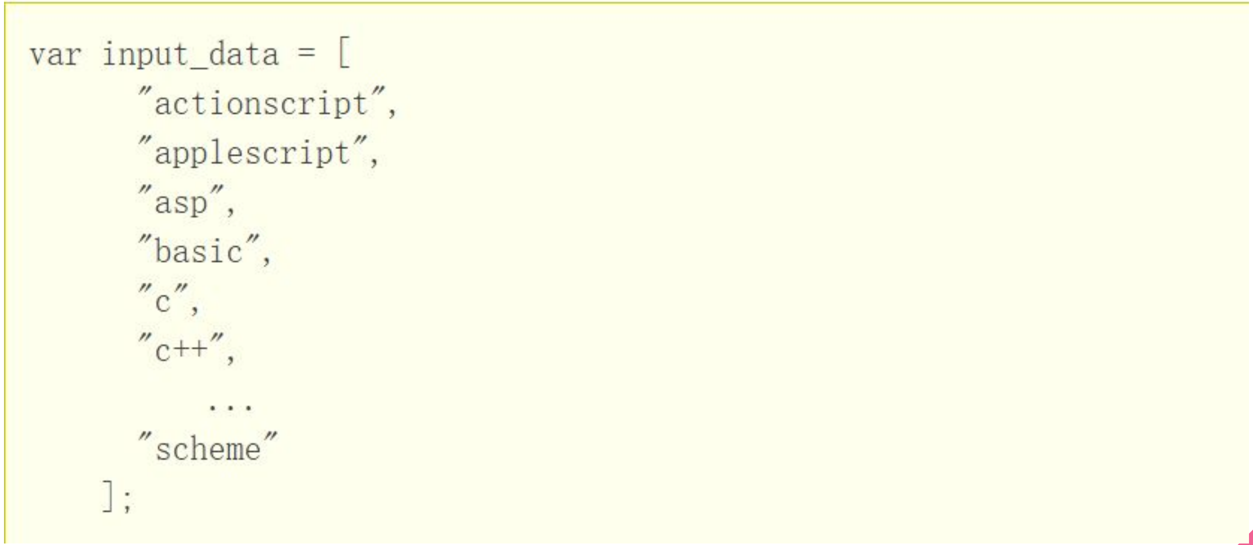
- JQueryUI provides an autocomplete widget:
  - A control that acts a lot like a <select> dropdown, but filters the choices to present only those that match what the user is typing into a control.
  - To create a list of suggestions below the input field and adds new CSS classes to the elements concerned to give them the appropriate style.
- Any field that can receive input can be converted into an Autocomplete.
  - <input> elements, <textarea> elements, and elements with the contenteditable attribute.
- Tutorials by jQuery UI: https://www.tutorialspoint.com/jqueryui/jqueryui_autocomplete.htm

Basic example: default autocomplete widget

- Lab page: https://course.cse.ust.hk/comp4021/hkust/labs/jquery-client-side/index.html
- Starting code in the provided .zip file:
  - task1.html
  - task2.html
- Note on file names:
  - In the lab page it says “Open default_autocomplete_example.html in any text editor.” It refers to task1.html.

Basic example: default autocomplete widget

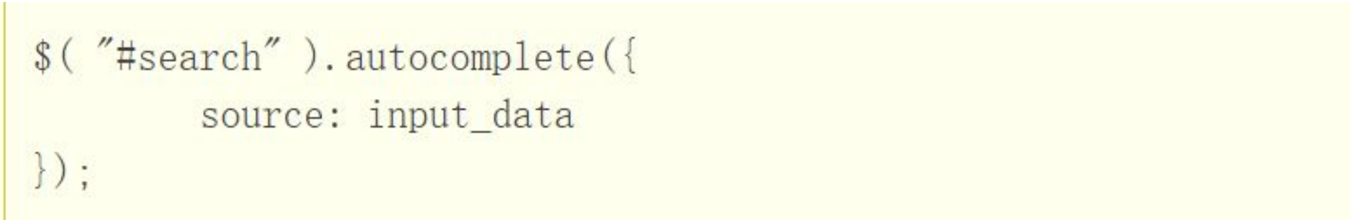
- You can see an array created through javascript by the following code:



```
var input_data = [
      ″actionscript″,
      ″applescript″,
      ″asp″,
      ″basic″,
      ″c″,
      ″c++″,
         ...
      ″scheme″
    ];
```

Basic example: default autocomplete widget

- Right below the array *input_data*, you can see the following code:



```
$( ″#search″ ).autocomplete({
        source: input_data
});
```

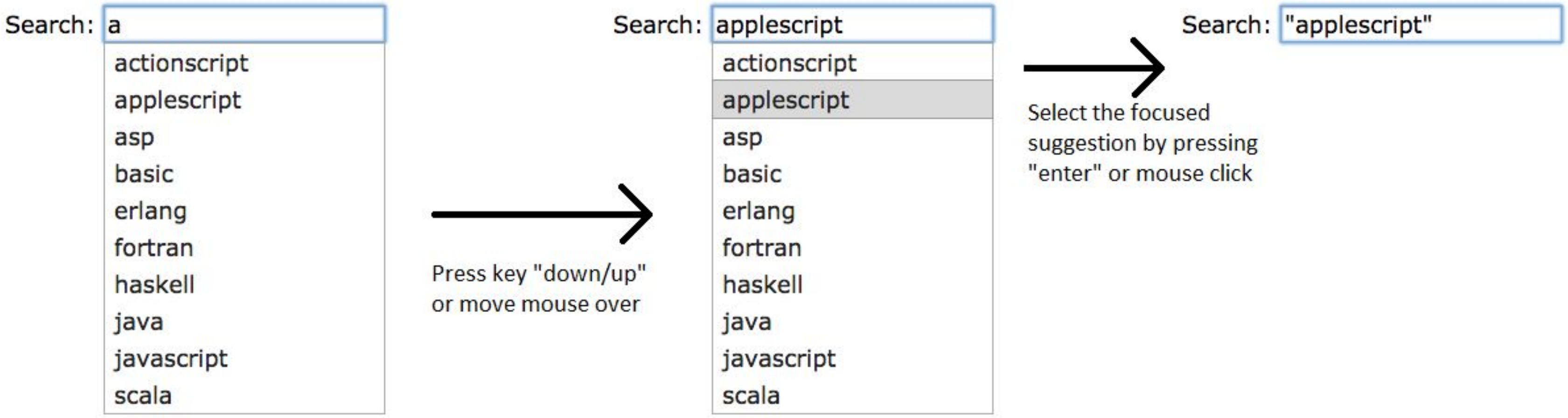
$("#search"): a jquery selector which is to select the object with id search.
$("#search").autocomplete(...): to create the autocomplete widget for the object with id search.
source: input_data: to indicate that input_data is the source data which will be filtered by autocomplete widget and then shows the suggestions.

Basic example: default autocomplete widget

- What if you want to do it with plain js?
    - https://www.w3schools.com/howto/howto_js_autocomplete.asp
    - NOT required, just for those who are interested.

Change the search keyword programmatically when a suggestion is selected

- **Goal**: When a suggestion is selected, show the selected suggestion in the input text box with adding double quote (") at the beginning and the end.



Search: a

Press key "down/up" or move mouse over

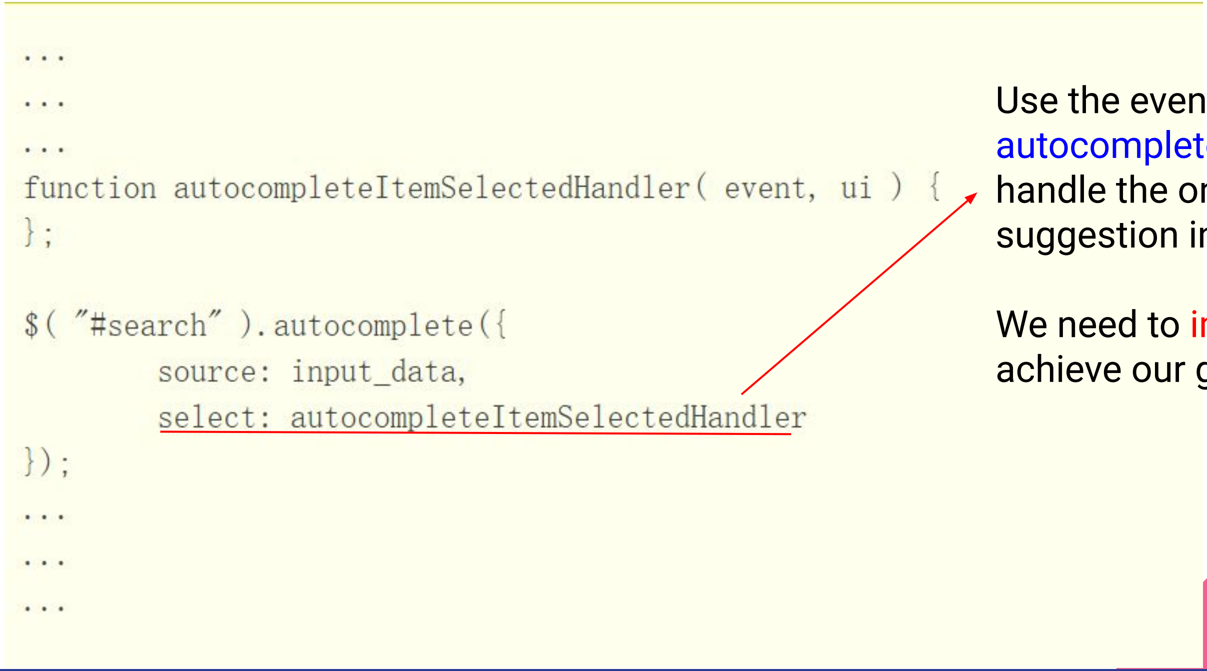
Search: applescript

Select the focused suggestion by pressing "enter" or mouse click

Search: "applescript"

Change the search keyword programmatically when a suggestion is selected

- **To-do**: Modify the content of task_1.html as follows:



```
...
...
...
function autocompleteItemSelectedHandler( event, ui ) {
};

$( ”#search” ).autocomplete({
        source: input_data,
        select: autocompleteItemSelectedHandler
});
...
...
...
```

Use the event handler autocompleteItemSelectedHandler to handle the onselect event of the suggestion in the autocomplete widget.

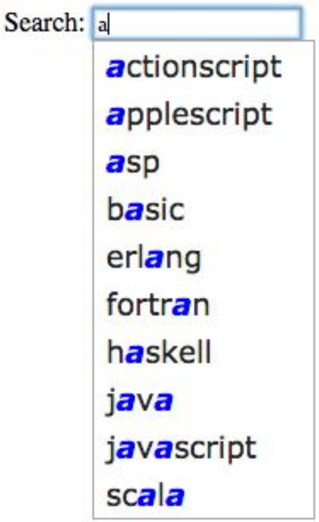
We need to implement this function to achieve our goal.

Change the search keyword programmatically when a suggestion is selected

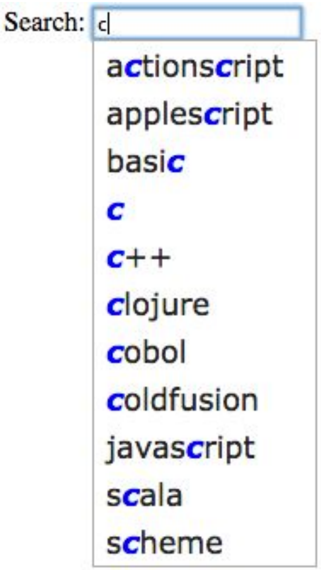
```
function autocompleteItemSelectedHandler(event, ui){
    //retrieve the selected content
    //modify the content
    //write the modified content back to the inputbox object
}
```

Customize the style of autocomplete suggestions

- **Goal**: customize the style of the suggestions such that all substrings in each suggestion that matches the search keyword will be shown in blue, bold and italic.



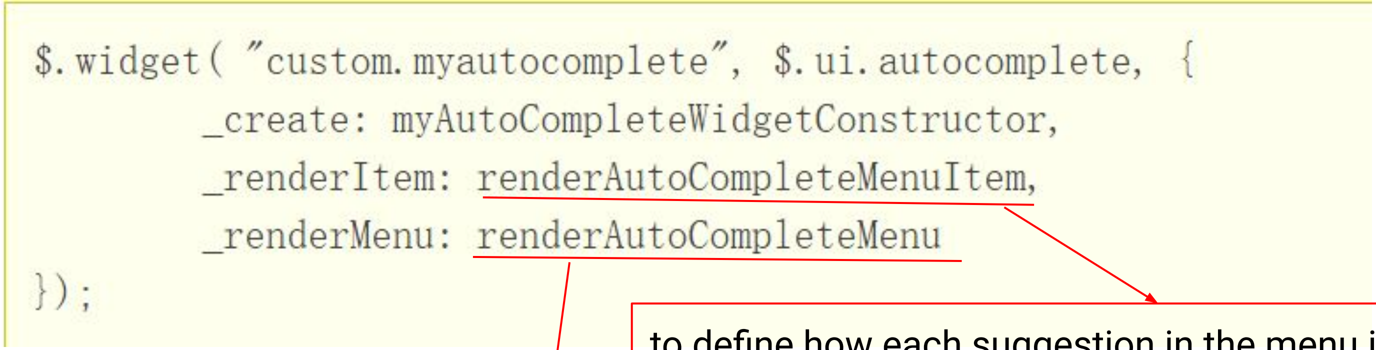
Search: a



Search: c

Customize the style of autocomplete suggestions

- **To-do**: define our customed widget here:



```
$.widget( ″custom.myautocomplete″, $.ui.autocomplete, {
        _create: myAutoCompleteWidgetConstructor,
        _renderItem: renderAutoCompleteMenuItem,
        _renderMenu: renderAutoCompleteMenu
});
```

to define how each suggestion in the menu is displayed

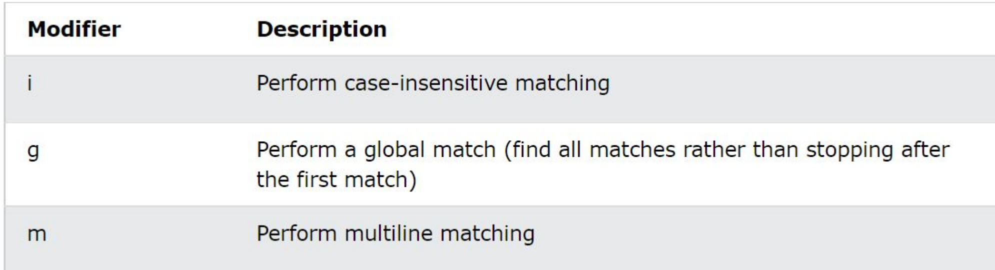
to define how the autocomplete suggestion menu is displayed

Customize the style of autocomplete suggestions

```
function renderAutoCompleteMenuItem(ul, item){
        //retrieve the substring
        //match the substring with items via a regular expression
        //change the format of the matched substrings in items
        //display the content
}
```

Customize the style of autocomplete suggestions

- Regular expression:
  - A sequence of characters that forms a search pattern, which can be used for text search and text replace operations.
- JavaScript regular expressions:
  - Syntax: `/pattern/modifiers;`



Modifier	Description
i	Perform case-insensitive matching
g	Perform a global match (find all matches rather than stopping after the first match)
m	Perform multiline matching

- JavaScript RegExp object:
  - a regular expression object with predefined properties and methods.

Customize the style of autocomplete suggestions

```
function renderAutoCompleteMenu(ul, item){
    //iterate through each item
    //display each item by calling renderAutoCompleteMenuItem()
}
```

```
$.each(items, function( index, item){
            that._renderItemData(ul, item)
      })
```

The End