Useful Tools

* DevTools Autosave – if we anything styled in chrome console it will automatically save in editor
* Another name of it is chrome auto save
* For use autosave open cmd & hit autosave

Snippet for Self-Invoking Function

<snippet>

<content><![CDATA[

(function(){

${1:this}

})();

]]></content>

<!-- Optional: Set a tabTrigger to define how to trigger the snippet -->

<tabTrigger>siaf</tabTrigger>

<!-- Optional: Set a scope to limit where the snippet will trigger -->

<!-- <scope>source.python</scope> -->

</snippet>

jQuery

* The jQuery library makes it easy to manipulate a page of HTML after it's displayed by the browser.
* It also provides tools that help you listen for a user to interact with your page, tools that help you create animations in your page, and tools that let you communicate with a server without reloading the page.
* jQuery uses css selectors to select an element from the dom

Bad Practices of jQuery

* $( something } asks jQuery to rescan for the matching element, wrap it in a jQuery object, and create a new instance of something you already have in memory.
* If your code crashes due to memory cascades, overuse of redundant selectors might be related.
* Instead we should use Caching Selectors

Caching Selectors

* Just declare all of your objects at the top of your closure and use the new variables throughout your script.
* Example –

var $window = $( window ),

$document = $( document ),

$footer = $( '#footer' ),

$sidebar = $( '#sidebar' ),

$images = $( 'img' );

jQuery Best Practices

## jQuery Object Caching

## Caching your jQuery objects may possibly be the best thing you can do to cut your code down to run leaner and meaner.

## Instead of using this // Loop

## for (var i=0; i<100; i++) {

## $('ul.special').append('<li>'+i+'</li>');

## }

## We should use this

var $ul = $('ul.special');  
for (var i=0; i<100; i++) {  
 $ul.append('<li>'+i+'</li>');  
}

* // Just re-querying  
  $('p').width(150);  
  $('p').css('color', 'red');  
  $('p').addClass('awesome');
* We should use this $('p').width(150).css('color', 'red').addClass('awesome');

## Selector Optimization

// Instead of this:  
$('#id p');

// Try one of these:

$('p', '#id');

$('#id').find('p');

$('#id').children('p')

jQuery Method

* Children() will return only the direct children of the element never consider grandchild
* Find() will return every child of the element
* This method is shorten for document.ready method

$(function() {

console.log( 'ready!' );

});

Get some elements

$( '#header' ); // select the element with an ID of 'header'

$( 'li' ); // select all list items on the page

$( 'ul li' ); // select list items that are in unordered lists

$( '.person' ); // select all elements with a class of 'person'

* Text() will add text with the desire selection
* Nth-child - nth-child(n) selector matches every element that is the nth child, regardless of type, of its parent.
* The :eq() selector selects an element with a specific index number.
* The index numbers start at 0, so the first element will have the index number 0 (not 1).
* Example of chaining method : $('ul.emphasis').children('li').eq(3).next().text('added with jQuery');
* We don’t do that in this way because we do that manually like
* $('ul.emphasis li:nth-child(3)').text('added with jQuery');
* For more Readability we write:

$('ul.emphasis')

.children('li')

.eq(3)

.prev().text('added with jQuery');

* $('li').parent().removeClass('emphasis'); Here parent means select the parent of li & apply removeClass.
* Get the parent of each element in the current set of matched elements, optionally filtered by a selector.
* Parents() - Get the ancestors of each element in the current set of matched elements, optionally filtered by a selector.
* Example - $('li').parents('div.container').removeClass('container');
* Closest() - For each element in the set, get the first element that matches the selector by testing the element itself and traversing up through its ancestors in the DOM tree.
* For specific element we should use closest() rather than parents()
* Attr() - Get the value of an attribute for the first element in the set of matched elements or set one or more attributes for every matched element.
* In attr if we use one property it means we want to get the attribute value
* If we use two property i.e. we want to set the attribute value.
* Data() - The data() method attaches data to, or gets data from, selected elements.
* Data(key) - Return the value at the named data store for the first element in the jQuery collection
* End() - End the most recent filtering operation in the current chain and return the set of matched elements to its previous state.
* If we need an older element set, we can use end() to pop the sets back off of the stack.
* If we need an older element set, we can use end() to pop the sets back off of the stack.
* Example

$( "ul.first" )

.find( ".foo" )

.css( "background-color", "red" )

.end()

.find( ".bar" )

.css( "background-color", "green" );

Events

* Event methods trigger or attach a function to an event handler for the selected elements.
* Respond to user interaction with a web page.
* Code that runs when a user clicks on a certain part of the page, or when she moves her mouse over a form element
* For example, this code listens for a user to click on any li element in the page:

$( 'li' ).click(function( event ) {

console.log( 'clicked', $( this ).text() );

});

* console.log(this); refers to the element that was clicked
* <button data-file="day">Day</button> if we clicked on the day button then this will return the day button
* Here this refers to the regular dom method
* If we want to access jQuery method then we should wrap the this with $(this)