### Attribute Ends With Selector \[ \text{name} *=\text{value} \]
Selects elements that have the specified attribute with a value ending exactly with a given string. The comparison is case sensitive.

### Attribute Contains Prefix Selector \[ \text{name} =\text{value} \]
Selects elements that have the specified attribute with a value beginning exactly with a given string.

### Attribute Does Not Equal Selector \[ \text{name} \neq \text{value} \]
Selects elements that either don't have the specified attribute, or do have the specified attribute but not with a certain value.

### Attribute Starts With Selector \[ \text{name} ^=\text{value} \]
Selects elements that have the specified attribute with a value beginning exactly with a given string.

### :empty Selector
Selects all elements that have no children (including text nodes).

### :enabled Selector
Selects all elements that are enabled.

### :eq(\text{n}) Selector
Selects the element at index \text{n} within the matched set.

### :even Selector
Selects elements with an even index.

### :file Selector
Selects all elements of type file.

### :first-child Selector
Selects all elements that are the first child of their parent.

### :first-selector
Selects the first matched element.

### :focus Selector
Selects element if it is currently focused.

### :lt(\text{n}) Selector
Selects all elements at an index less than \text{n} within the matched set.

### :last-child Selector
Selects all elements that are the last child of their parent.

### :last-selector
Selects the last matched element.

### :lt(\text{n}) Selector
Selects all elements at an index less than \text{n} within the matched set.

### Multiple Attribute Selector \[ \text{name}=\text{value} \] \[ \text{name2}=\text{value2} \]
Matches elements that match all of the specified attribute filters.

### Multiple Selector \[ \text{selector1} \text{selector2}, \text{selectorN} \]
Selects the combined results of all the specified selectors.

### Next Adjacent Selector \[ \text{prev} + \text{next} \]
Selects all next elements matching “next” that are immediately preceded by a sibling “prev”.

### Next Siblings Selector \[ \text{prev} \text{siblings} \]
Selects all sibling elements that follow after the “prev” element, have the same parent, and match the filtering “siblings” selector.

### :not(\text{selector}) Selector
Selects all elements that do not match the given selector.

### :nth-child(\text{n}) Selector
Selects all elements that are the \text{n}-th-child of their parent.

### :odd Selector
Selects odd elements, zero-indexed.

### :only-child Selector
Selects all elements that are the only child of their parent.

### :parent Selector
Selects all elements that are the parent of another element, including text nodes.

### :password Selector
Selects all elements of type password.

### :radio Selector
Selects all elements of type radio.

### :reset Selector
Selects all elements of type reset.
### attributes

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>.addClass( className )</td>
<td>Adds the specified class(es) to each element of the set of matched elements.</td>
<td><code>div.addClass('active')</code></td>
</tr>
<tr>
<td>.attr( attributeName, value )</td>
<td>Gets or sets the value of an attribute for the first element of the set of matched elements.</td>
<td><code>div.attr('id', 'example')</code></td>
</tr>
<tr>
<td>.hasClass( className )</td>
<td>Determines whether any of the matched elements are assigned the given class.</td>
<td><code>div.hasClass('active')</code></td>
</tr>
</tbody>
</table>

### traversing

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>.add( selector )</td>
<td>Adds elements to the set of matched elements.</td>
<td><code>div.add('p')</code></td>
</tr>
<tr>
<td>.addSelf()</td>
<td>Adds the previous set of elements on the stack to the current set.</td>
<td><code>div.addSelf()</code></td>
</tr>
<tr>
<td>.children( selector )</td>
<td>Get the children of each element in the set of matched elements, optionally filtered by a selector.</td>
<td><code>div.children('.nested')</code></td>
</tr>
<tr>
<td>.closest( selector )</td>
<td>Get the first element that matches the selector, beginning at the current element and progressing up through the DOM tree.</td>
<td><code>div.closest('.parent')</code></td>
</tr>
<tr>
<td>.contains( selector )</td>
<td>Get the children of each element in the set of matched elements, including text and comment nodes.</td>
<td><code>div.contains('p')</code></td>
</tr>
<tr>
<td>.each( function(index, Element) )</td>
<td>Iterate over a jQuery object, executing a function for each matched element.</td>
<td><code>div.each(function(index, elem) { elem.style.color = 'red' })</code></td>
</tr>
</tbody>
</table>

### manipluation

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>.addProp( propertyName )</td>
<td>Remove a property for the set of matched elements.</td>
<td><code>div.addProp('is-visible')</code></td>
</tr>
<tr>
<td>.filter( selector )</td>
<td>Reduce the set of matched elements to those that match the selector or pass the function’s test.</td>
<td><code>div.filter('.active')</code></td>
</tr>
<tr>
<td>.toggleClass( className )</td>
<td>Add or remove one or more classes from each element in the set of matched elements, depending on either the class's presence or the value of the switch argument.</td>
<td><code>div.toggleClass('active')</code></td>
</tr>
<tr>
<td>.val( value )</td>
<td>Get the current value of the first element in the set of matched elements.</td>
<td><code>div.val('Hello')</code></td>
</tr>
<tr>
<td>.has( selector )</td>
<td>Reduce the set of matched elements to those that have a descendant that matches the selector or DOM element.</td>
<td><code>div.has('.child')</code></td>
</tr>
<tr>
<td>.last( selector )</td>
<td>Reduce the set of matched elements to the final one in the set.</td>
<td><code>div.last('.child')</code></td>
</tr>
<tr>
<td>.map( callback(index, domElement) )</td>
<td>Pass each element in the current matched set through a function, producing a new jQuery object containing the return values.</td>
<td><code>div.map(function(index, elem) { return elem.innerHTML })</code></td>
</tr>
</tbody>
</table>

### effects

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>.removeAttr( attributeName )</td>
<td>Remove an attribute from each element in the set of matched elements.</td>
<td><code>div.removeAttr('id')</code></td>
</tr>
<tr>
<td>.removeClass()</td>
<td>Remove one or more classes from each element in the set of matched elements, optionally filtered by a selector.</td>
<td><code>div.removeClass('active')</code></td>
</tr>
<tr>
<td>.removeProp( propertyName )</td>
<td>Remove a property for the set of matched elements.</td>
<td><code>div.removeProp('is-visible')</code></td>
</tr>
<tr>
<td>.slice( start, end )</td>
<td>Reduce the set of matched elements to a subset specified by a range of indices.</td>
<td><code>div.slice(0, 3)</code></td>
</tr>
</tbody>
</table>

### selectors

- `.eq(index)` | Selects the element at the specified index. | `div.eq(1)` |
- `.has( selector )` | Selects elements that have a descendant that matches the selector or DOM element. | `div.has('.child')` |
- `.is(selector)` | Checks the current matched set of elements against a selector, element, or jQuery object and return true if at least one of these matches the given argument. | `div.is('.active')` |
- `.next( selector )` | Selects all elements that are selected. | `div.next('.child')` |
- `.nextUntil( selector )` | Selects all elements that are selected. | `div.nextUntil('.child')` |
- `.parents( selector )` | Selects all elements that are parents of each element in the current set of matched elements. | `div.parents('.parent')` |
- `.parentsUntil( selector )` | Selects all elements that are parents of each element in the current set of matched elements, up to but not including the element matched by the selector, DOM node, or jQuery object. | `div.parentsUntil('.parent')` |
- `.prev( selector )` | Selects all elements that are previous siblings of each element in the set of matched elements. | `div.prev('.child')` |
- `.prevUntil( selector )` | Selects all elements that are previous siblings of each element in the set of matched elements, up to but not including the element matched by the selector, DOM node, or jQuery object. | `div.prevUntil('.child')` |
- `.siblings( selector )` | Selects all elements that are siblings of each element in the set of matched elements. | `div.siblings('.child')` |
- `.closest( selector )` | Selects all elements that are closest ancestor elements that are positioned. | `div.closest('.parent')` |
- `.end()` | Ends the most recent filtering operation in the current chain and returns the set of matched elements to its previous state. | `div.end()` |
- `.first()` | Selects all elements that are selected. | `div.first()` |
- `.last()` | Selects all elements that are selected. | `div.last()` |
- `.offsetParent()` | Gets the closest ancestor element that is positioned. | `div.offsetParent()` |
- `.parent( selector )` | Gets the parent of each element in the current set of matched elements, optionally filtered by a selector. | `div.parent('.parent')` |
- `.parents( selector )` | Gets the ancestors of each element in the current set of matched elements, optionally filtered by a selector. | `div.parents('.parent')` |
- `.parentsUntil( selector )` | Gets the ancestors of each element in the current set of matched elements, up to but not including the element matched by the selector, DOM node, or jQuery object. | `div.parentsUntil('.parent')` |
- `.slice( start, end )` | Reduce the set of matched elements to a subset specified by a range of indices. | `div.slice(0, 3)` |

### core

- `.slice( start, end )` | Reduce the set of matched elements to a subset specified by a range of indices. | `div.slice(0, 3)` |
- `.contains( selector )` | Get the children of each element in the set of matched elements, including text and comment nodes. | `div.contains('p')` |
- `.each( function(index, Element) )` | Iterate over a jQuery object, executing a function for each matched element. | `div.each(function(index, elem) { elem.style.color = 'red' })` |
- `.closest( selector )` | Get the first element that matches the selector, beginning at the current element and progressing up through the DOM tree. | `div.closest('.parent')` |
- `.hasClass( className )` | Determines whether any of the matched elements are assigned the given class. | `div.hasClass('active')` |
- `.add() | Adds elements to the set of matched elements. | `div.add('p')` |
- `.addSelf()` | Adds the previous set of elements on the stack to the current set. | `div.addSelf()` |
- `.attr( attributeName, value )` | Gets or sets the value of an attribute for the first element of the set of matched elements. | `div.attr('id', 'example')` |
- `.val( value )` | Get the current value of the first element in the set of matched elements. | `div.val('Hello')` |
- `.filter( selector )` | Reduce the set of matched elements to those that match the selector or pass the function’s test. | `div.filter('.active')` |
- `.toggleClass( className )` | Add or remove one or more classes from each element in the set of matched elements, depending on either the class's presence or the value of the switch argument. | `div.toggleClass('active')` |
- `.val( value )` | Get the current value of the first element in the set of matched elements. | `div.val('Hello')` |
**.height()** - Get the current computed height for the first element in the set of matched elements.

**.innerHeight()** - Get the current computed height for the first element in the set of matched elements, excluding padding but not border.

**.innerWidth()** - Get the current computed width for the first element in the set of matched elements, excluding padding but not border.

**.offset()** - Get the current coordinates of the first element in the set of matched elements, relative to the document.

**.offsetHeight( [includeMargin] )** - Get the current computed height for the first element in the set of matched elements, including padding, border, and optionally margin. Returns an integer (without "px") representation of the value or null if called on an empty set of elements.

**.offsetWidth( [includeMargin] )** - Get the current computed width for the first element in the set of matched elements, including padding and border.

**.position()** - Get the current coordinates of the first element in the set of matched elements, relative to the offset parent.

**.removeClass()** - Remove a single class, multiple classes, or all classes from each element in the set of matched elements.

**.scrollLeft()** - Get the current horizontal position of the scroll bar for the first element in the set of matched elements.

**.scrollTop()** - Get the current vertical position of the scroll bar for the first element in the set of matched elements.

**EVENTS**

**.bind()** - Attach a handler to an event for the elements.

**.dblclick()** - Bind an event handler to the "dblclick" JavaScript event, or trigger that event on an element.

**.click()** - Bind an event handler to the "click" JavaScript event, or trigger that event on an element.

**.hover()** - Bind two handlers to the matched elements, to be executed when the mouse pointer enters and leaves the elements.

**.change()** - Bind an event handler to the "change" JavaScript event, or trigger that event on an element.

**.blur()** - Bind an event handler to the "blur" JavaScript event, or trigger that event on an element.

**.focus()** - Bind an event handler to the "focus" JavaScript event, or trigger that event on an element.

**.focusout()** - Bind an event handler to the "focusout" JavaScript event.

**.error()** - Bind an event handler to the "error" JavaScript event.

**.error()** - Bind an event handler to the "evento" event of the given type.

**.delegate()** - Attach a handler to one or more events for all elements that match the selector, now or in the future, based on a specific set of root elements.

**.toggleClass()** - Add or remove one or more classes from each element in the set of matched elements, depending on either the class's presence or the value of the switch argument.
### jQuery 1.7

**VISUAL CHEAT SHEET**

#### EFFECTS
- **animate()**
  - Perform a custom animation of a set of CSS properties.
  ```javascript
  animate(properties [duration] [easing] [complete])
  ```
- **clearQueue()**
  - Remove from the queue all items that have not yet been run.
  ```javascript
  clearQueue([queueName])
  ```
- **delay()**
  - Set a timer to delay execution of subsequent items in the queue.
  ```javascript
  delay([duration] [queueName])
  ```
- **enqueue()**
  - Execute the next function on the queue for the matched elements.
  ```javascript
  enqueue(function) `function` is executed immediately.
  ```
- **dequeue()**
  - Execute the next function on the queue for the matched elements.
  ```javascript
  dequeue([queueName])
  ```
- **fadeIn()**
  - Display the matched elements by fading them to opaque.
  ```javascript
  fadeIn(duration) [callback]
  ```
- **fadeOut()**
  - Hide the matched elements by fading them to transparent.
  ```javascript
  fadeOut([duration] [callback])
  ```
- **fadeToggle()**
  - Display or hide the matched elements by animating their opacity.
  ```javascript
  fadeToggle([duration] [easing] [callback])
  ```
- **undelegate()**
  - Remove a handler from the event for all elements which match the current selector, based upon a specific set of root elements.
  ```javascript
  undelegate(selector [eventType] [callback])
  ```
- **unload()**
  - Bind an event handler to the "unload" JavaScript event.
  ```javascript
  unload([handler(eventObject)])
  ```

#### CORE
- **jQuery**
  ```javascript
  jQuery
  ```
- **$.ajax()**
  ```javascript
  $.ajax([options])
  ```
- **$.ajaxSetup()**
  ```javascript
  $.ajaxSetup([options])
  ```
- **$.ajaxSettings()**
  ```javascript
  $.ajaxSettings(options)
  ```

#### SELECTORS
- **$.array()**
  ```javascript
  $.array(array)
  ```
- **$.attr()**
  ```javascript
  $.attr(element, name, value)
  ```

#### TRAVERSING
- **$.each()**
  ```javascript
  $.each(array, callback)
  ```
- **$.inArray()**
  ```javascript
  $.inArray(element, array)
  ```

#### MANIPULATION
- **$.add()**
  ```javascript
  $.add(array1, array2)
  ```
- **$.append()**
  ```javascript
  $.append(element)
  ```
- **$.clone()**
  ```javascript
  $.clone(element)
  ```

#### CSS
- **$.css()**
  ```javascript
  $.css(element, name, value)
  ```
- **$.css()**
  ```javascript
  $.css(element, name, [value])
  ```

#### EVENTS
- **$.delegate()**
  ```javascript
  $.delegate(selector, events, handler)
  ```
- **$.event()**
  ```javascript
  $.event
  ```

#### AJAX
- **$.ajax()**
  ```javascript
  $.ajax([options])
  ```
- **$.ajaxSetup()**
  ```javascript
  $.ajaxSetup([options])
  ```

#### UTILITIES
- **$.noop()**
  ```javascript
  $.noop
  ```
- **$.proxy()**
  ```javascript
  $.proxy(function, context)
  ```

#### CALLBACKS
- **$.callback()**
  ```javascript
  $.callback(callback)
  ```
- **$.callback()**
  ```javascript
  $.callback(callback)
  ```

#### DATA & MISC
- **$.fn.extend()**
  ```javascript
  $.fn.extend(object)
  ```
- **$.fn.init()**
  ```javascript
  $.fn.init([jQuery], [selector], [array])
  ```

#### DEFERRED OBJECT
- **$.Deferred()**
  ```javascript
  $.Deferred([reject], [resolve], [state])
  ```
**jQuery.queue( function )**
Takes a function and returns a new one that will always have a particular context.

**jQuery.proxy( function, context )**
Converts a function to one that stores the context and allows the context to be changed.

**jQuery.map( array, function )**
Translates all items in an array or object to new arrays of DOM elements, not strings or numbers.

**jQuery.merge( elementOfArray, ...otherArrays )**
Iterate over a jQuery object, executing a function for each matched element.

**jQuery.trim( str )**
Remove the whitespace from the beginning and end of a string.

**jQuery.type( obj )**
Determine the internal JavaScript [[Class]] of an object.

**jQuery.unique( array )**
Sorts an array of DOM elements, in place, with the duplicates removed. Note that this only works on arrays of DOM elements, not strings or numbers.

**jQuery.isXMLDoc( element )**
Check to see if a DOM node is within an XML document.

**jQuery.isWindow( object )**
Check to see if an object is a plain object (created using "{}" or "new Object").

**jQuery.isPlainObject( object )**
Check to see if an object is a plain object (created using "{}" or "new Object").

**jQuery.isNumeric( object )**
Determines whether its argument is a number.

**jQuery.now()**
Return a number representing the current time.

**jQuery.queue( callback )**
A multi-purpose callbacks list object that provides a powerful way to manage callback lists.

**jQuery.callbacks( flag )**
A multi-purpose callbacks list object that provides a powerful way to manage callback lists.

**jQuery.dequeue()**
Execute the next function on the queue for the matched element.

**jQuery.dequeue( name )**
Remove the previously-stored piece of data.

**jQuery.data()**
Store arbitrary data associated with the matched elements.

**jQuery.removeData( element, name )**
Remove a previously-stored piece of data.

**jQuery.proxy( function, context )**
Converts a function to one that stores the context and allows the context to be changed.

**jQuery.each( list, callback )**
Add a callback or a collection of callbacks to a callback list.

**jQuery.add( elementOfArray, ...otherArrays )**
Add a callback or a collection of callbacks to a callback list.

**jQuery.unique( array )**
Sorts an array of DOM elements, in place, with the duplicates removed. Note that this only works on arrays of DOM elements, not strings or numbers.

**jQuery.unique( array )**
Sorts an array of DOM elements, in place, with the duplicates removed. Note that this only works on arrays of DOM elements, not strings or numbers.

**jQuery.proxy( function, context )**
Converts a function to one that stores the context and allows the context to be changed.

**jQuery.each( object, callback )**
Iterate over a jQuery object, executing a function for each matched element.

**jQuery.proxy( function, context )**
Converts a function to one that stores the context and allows the context to be changed.

**jQuery.each( object, callback )**
Iterate over a jQuery object, executing a function for each matched element.

**jQuery.proxy( function, context )**
Converts a function to one that stores the context and allows the context to be changed.

**jQuery.each( object, callback )**
Iterate over a jQuery object, executing a function for each matched element.

**jQuery.proxy( function, context )**
Converts a function to one that stores the context and allows the context to be changed.

**jQuery.each( object, callback )**
Iterate over a jQuery object, executing a function for each matched element.
**deferred.notifyWith()**

Call the progressCallbacks on a Deferred object with the given context and args.
```
deferred.notifyWith( context, args )
```

**deferred.pipe()**

Utility method to filter and/or chain Deferreds.
```
deferred.pipe( doneFilter, failFilter )
deferred.pipe( doneFilter, failFilter, progressFilter )
```

**deferred.progress()**

Add handlers to be called when the Deferred object generates progress notifications.
```
deferred.progress( progressCallbacks )
```

**deferred.promise()**

Return a Deferred’s Promise object.
```
deferred.promise( target )
```

**deferred.reject( args )**

Reject a Deferred object and call any failCallbacks with the given args.
```
deferred.reject( args )
```

**deferred.rejectWith( context, args )**

Reject a Deferred object and call any failCallbacks with the given context and args.
```
deferred.rejectWith( context, args )
```

**deferred.resolve( args )**

Resolve a Deferred object and call any doneCallbacks with the given args.
```
deferred.resolve( args )
```

**deferred.resolveWith( context, args )**

Resolve a Deferred object and call any doneCallbacks with the given context and args.
```
deferred.resolveWith( context, args )
```

**deferred.state()**

Determine the current state of a Deferred object.
```
deferred.state( )
```

**deferred.then()**

Add handlers to be called when the Deferred object is resolved or rejected.
```
deferred.then( doneCallbacks, failCallbacks )
deferred.then( doneCallbacks, failCallbacks, progressCallbacks )
```

**jQuery.when( deferreds )**

Provides a way to execute callback functions based on one or more objects, usually Deferred objects that represent asynchronous events.
```
jQuery.when( deferreds )
```

**jQuery 1.7 New or Changed in jQuery 1.7.1**

**WHAT’S NEW?**

Aspects of the API that were changed in the corresponding version of jQuery. API changes in jQuery 1.7.0 dealt primarily with the new Event APIs: .on() and .off()

**Better Support for HTML5 in IE6/7/8**

jQuery.Callbacks()

**callbacks.add()**

**callbacks.disable()**

**callbacks.empty()**

**callbacks.fire()**

**callbacks.fireWith()**

**callbacks.has()**

**callbacks.lock()**

**callbacks.locked()**

**callbacks.remove()**

**deferred.notify()**

**deferred.notifyWith()**

**deferred.pipe()**

**deferred.progress()**

**deferred.promise()**

**deferred.reject()**

**deferred.rejectWith()**

**deferred.resolve()**

**deferred.resolveWith()**

**deferred.state()**

**deferred.then()**

**jQuery is a fast and concise JavaScript Library that simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development. jQuery is designed to change the way that you write JavaScript.**

**jQuery Official Web Site**

http://jquery.com

**Download**

http://docs.jquery.com/Downloading_jQuery

**Documentation**

http://docs.jquery.com/Main_Page

**Tutorials**

http://docs.jquery.com/Tutorials

**Bug Tracker**

http://bugs.jquery.com/newticket

**Discussion**

http://docs.jquery.com/Discussion

**Credits**

JQUERY 1.7 VISUAL CHEAT SHEET

This jQuery visual cheat sheet is designed by Antonio Lupetti and is distributed for free.

Update as of december 2011, jQuery current release v 1.7.1

**My Blog**

http://woorkup.com

**Twitter**

@woork - http://twitter.com/woork

**Facebook**

http://facebook.com/antoniolupetti

**Google+**

https://plus.google.com/u/0/112930363515083491757