Supported Selectors
* * any element
* E an element of type E
* E:root an E element, root of the document
* E:nth-child(n) an E element, the n-th child of its parent
* E:nth-last-child(n) an E element, the n-th child of its parent, counting from the last one
* E:nth-of-type(n) an E element, the n-th sibling of its type
* E:nth-last-of-type(n) an E element, the n-th sibling of its type, counting from the last one
* E:first-child an E element, first child of its parent
* E:last-child an E element, last child of its parent
* E:first-of-type an E element, first sibling of its type
* E:last-of-type an E element, last sibling of its type
* E:only-child an E element, only child of its parent
* E:only-of-type an E element, only sibling of its type
* E:empty an E element that has no children (including text nodes)
* E:lang(fr) an element of type E in language "fr"
* E:enabled
* E:disabled a user interface element E which is enabled or disabled
* E:checked a user interface element E which is checked (for instance a radio-button or checkbox)
* E.warning an E element whose class is "warning"
* E#myid an E element with ID equal to "myid".
* E:not(s) an E element that does not match simple selector s
* E F an F element descendant of an E element
* E > F an F element child of an E element
* E + F an F element immediately preceded by an E element
* E ~ F an F element preceded by an E element

Supported, but different
All attribute selectors are written like their XPath counter-parts
(in that all attributes should begin with an @ symbol).
* E[@foo] an E element with a "foo" attribute
* E[@foo="bar"] an E element whose "foo" attribute value is exactly equal to "bar"
* E[@foo~="bar"] an E element whose "foo" attribute value is a list of space-separated values, - one of which is exactly equal to "bar"
* E[@foo^="bar"] an E element whose "foo" attribute value begins exactly with the string "bar"
* E[@foo$="bar"] an E element whose "foo" attribute value ends exactly with the string "bar"
* E[@foo*="bar"] an E element whose "foo" attribute value contains the substring "bar"
* E[@hreflang|="en"] an E element whose "hreflang" attribute has a hyphen-
separated list of values beginning (from the left) with "en"

Plugins/Authoring

Plugin writing comes in two steps.
The first is writing any of your public methods, for example:
```
$.fn.debug = function() { return this.each(function(){ alert(this);    });  };
```
Coders will now be able to call your new plugin, like so:
```
$("div p").debug();
```
* All new functions are attached to the $.fn object.
```
$.test = function() {
 // Do some internal stuff
};
```
You can then access it in the same manner:
```
$.test("some stuff");
```

Not supported : jQuery only supports selectors that actually select DOM elements - everything else is ignored.
* E:link
* E:visited an E element being the source anchor of a hyperlink of which the target is not yet visited (:link) or already visited (:visited)
* E:active
* E:hover
* E:focus an E element during certain user actions
* E:target an E element being the target of the referring URI
* E::first-line the first formatted line of an E element
* E::first-letter the first formatted letter of an E element
* E::selection the portion of an E element that is currently selected/highlighted by the user
* E::before generated content before an E element
* E::after generated content after an E element

ChainableMethods:

$("p").addClass("test").show().html("foo");

Each of those individual methods (addClass, show, and html) each return the query object, allowing you to continue applying methods to the current set of elements.

Base/Expression/XPath/Custom

```
$("/html/body//p")
 $("//p")
 $("//p/a")
 $("//a[@src]")
 $("//a[@src='google.com']")
```

Location Paths :
* Absolute Paths
  $("/html/body//p")
  $("/*/body//p")
  $("//p/../div")
* Relative Paths
  $("a",this)
  $("p/a",this)

Supported Predicates
* [@*] Has an attribute
  $("//div[@*]")
* [@foo] Has an attribute of foo
  $("//input[@checked]")
* [@foo='test'] Attribute foo is equal to test
  $("//a[@ref='nofollow']")
* [Nodelist] Element contains a node list,
for example:
  $("//div[p]")
  $("//div[p/a]")

Custom Selectors
* :even Selects every other (even) element from the matched element set.
* :odd Selects every other (odd) element from the matched element set.
* :eq(0) and :nth(0) Selects the Nth element from the matched element set
* :gt(4) Selects all matched elements whose index is greater than N.
* :lt(4) Selects all matched elements whose index is less than N.
* :first Equivalent to :eq(0)
* :last Selects the last matched element.
* :parent Selects all elements which have child elements (including text).
* :contains('test') Selects all elements which contain the specified text.
* :visible Selects all visible elements (this includes items that have a display of block or inline, a visibility of visible, and aren't form elements of type hidden)
* :hidden Selects all hidden elements (this includes items that have a display of none, or a visibility of hidden, or are form elements of type hidden)

jQuery supports basic XPath expressions, in addition to CSS 1-3.

Supported Axis Selectors
* Descendant Element has a descendant element
  $("//div//p")
* Child Element has a child element
  $("//div/p")
* Preceding Sibling Element has an element before it, on the same axes
  $("//div ~ form")
* Parent Selects the parent element of the element
  $("//div/../p")

Supported Predicates, but differently
* [last()] or [position()=last()] becomes :last
  $("p:last")
* [0] or [position()=0] becomes :eq(0) or :first
  $("p:first")
  $("p:eq(0)")
* [position() < 5] becomes :lt(5)
  $("p:lt(5)")
* [position() > 2] becomes :gt(2)
  $("p:gt(2)")

jQuery
New.Wave.Javascript