***http://plugins.jquery.com/project/timers***

**jQuery Timers**

August 22, 2008 - 4:24pm — [blair](http://plugins.jquery.com/user/191)

Average rating

(18 votes)

jQuery Timers is a high level abstraction of setTimeout and setInterval. It links into the jQuery chain so you can apply timers to elements directly in your code, leading to much more intuitive timer-based code. It also adds numerous advanced features such as labelling of timers and abstracted timer ids.

**Features**

**Abstracted Timing**

Saving the timer id when calling setTimeout and setInterval is annoying a troublesome if you need to cancel the timer at some other point in the code. You either have to pass it around as a parameter everywhere, or use a global variable. Timers creates a global cache of timer ids much like jQuery's events system creates a global cache of events.

**Example**

$("#close-button").click(function() {  
  $(this).oneTime(1000, function() {  
    $(this).parent(".main-window").hide();  
  });  
});  
$("#cancel-button").click(function() {  
  $("#close-button").stopTime();  
});

**Timing Labels**

Because jQuery timers connects timers to elements, it is advantageous to allow for a filtration of timers prior to cancelling, in case some other section of code is using jQuery timers for a different feature. Labels serve this purpose by allowing the programmer to assign a specific event a specific label. Labels are also automatically generated based on the timeout value of the timer if no specific label is generated. This means, you can cancel all timers originally set to activate 1000 ms later. This is merely a shortcut for programmers who prefer convention and implicit rules.

**Example**

$("#close-button").click(function() {  
  $(this).oneTime(1000, "hide", function() {  
    $(this).parent(".main-window").hide();  
  });  
});  
$("#cancel-button").click(function() {  
  $("#close-button").stopTime("hide");  
});

**Limited Timers**

Limited Timers are a natural extension of the basic setTimeout/setInterval pair. Rather than limiting the programmer to one event or an unending chain of events, jQuery timers allows you to specify a specific number of times for the event to occur. This may be useful for pseudo-asynchronous looping and other time-related events.

**Example**

var times = chunks.length;  
$(document).everyTime(1000, function(i) {  
  processChunk(i);  
}, times);

**String Time Parsing**

By default, setTimeout and setInterval take a millisecond count, but as human we tend to think in seconds. So if a string is provided as the interval/timeout in jQuery timers it will be parsed for standard time prefixes. So "2s" means 2 seconds, "250ms" means 250 milliseconds and so forth. The list of supported prefixes is larger than should ever be used.

**Methods added to the jQuery chain**

**everyTime(interval : Integer | String, [label = interval : String], fn : Function, [times = 0 : Integer])**

everyTime will add the defined function (fn) as a timed event to run at a given time interval (interval) for a given number of times (times). If times is set to 0, the number of times the method is called is unbounded. A label is also set for the given timed event either to the provided string (label) or to the string representation of the interval provided. Additionally, the interval can be defined by using a string such as "3s" for 3 seconds.

**oneTime(interval : Integer | String, [label = interval : String], fn : Function)**

oneTime will call the defined function (fn) a certain amount of time (interval) after being added to the elements in the jQuery object. A label (label) is also set for the timed event either to the provided string (label) or to the string representation of the interval provided.

**stopTime([label : Integer | String], [fn : Function])**

stopTime will stop any timed events with the provided label (label) and function (fn). If neither is specified, it will stop all timed events acting on the elements in the jQuery object. If only the function is provided, then it will stop all timed events calling that function regardless of label. Finally if only the label is provided, it will stop all timed events given that label at initialization.

**Releases**

| **Official releases** | **Date** | **Size** | **Links** | **Status** | |
| --- | --- | --- | --- | --- | --- |
| [1.2.0](http://plugins.jquery.com/node/10837) | 2009-Oct-16 | 3.26 KB | * [Download](http://plugins.jquery.com/files/jquery.timers-1.2.js.txt) * [Release notes](http://plugins.jquery.com/node/10837) | Recommended for *1.0.x* | This is currently the recommended release for 1.0.x. |

* [View all releases](http://plugins.jquery.com/node/3656/release)

**Resources**

* [Home page](http://jquery.offput.ca/timers)
* [Read documentation](http://jquery.offput.ca/timers)
* [Read license](http://sam.zoy.org/wtfpl)
* [Try out a demonstration](http://jquery.offput.ca/timers)

**Support**

* [View all support requests](http://plugins.jquery.com/project/issues/timers?categories=support&states=all)
* [View pending support requests](http://plugins.jquery.com/project/issues/timers?categories=support)
* [View pending bug reports](http://plugins.jquery.com/project/issues/timers?categories=bug)
* [View pending feature requests](http://plugins.jquery.com/project/issues/timers?categories=feature)
* [Login](http://plugins.jquery.com/user/login?destination=node/add/project-issue/timers) or [register](http://plugins.jquery.com/user/register?destination=node/add/project-issue/timers) to create an issue

**Development**

* [View pending patches](http://plugins.jquery.com/project/issues/timers?states=8,13,14)
* [View available tasks](http://plugins.jquery.com/project/issues/timers?categories=task)
* [View all pending issues](http://plugins.jquery.com/project/issues/timers)