* Overview
* Options
* Events
* Methods
* Theming

jQuery UI Slider

## Overview

The jQuery UI Slider plugin makes selected elements into sliders. There are various options such as multiple handles, and ranges. The handle can be moved with the mouse or the arrow keys.

All callbacks receive two arguments: The original browser event and a prepared ui object, view below for a documentation of this object (if you name your second argument 'ui'):

The slider widget will create handle elements with the class 'ui-slider-handle' on initialization. You can specify custom handle elements by creating and appending the elements and adding the 'ui-slider-handle' class before init. It will only create the number of handles needed to match the length of value/values. For example, if you specify 'values: [1, 5, 18]' and create one custom handle, the plugin will create the other two.

* **ui.handle**: DOMElement - the current focused handle
* **ui.value**: Integer - the current handle's value

### Dependencies

* UI Core
* UI Widget
* UI Mouse

### Example

* Demo
* View Source

A simple jQuery UI Slider.

$("#slider").slider();

<!DOCTYPE html>  
<html>  
<head>  
 <link href="http://ajax.googleapis.com/ajax/libs/jqueryui/1.8/themes/base/jquery-ui.css" rel="stylesheet" type="text/css"/>  
 <script src="http://ajax.googleapis.com/ajax/libs/jquery/1.5/jquery.min.js"></script>  
 <script src="http://ajax.googleapis.com/ajax/libs/jqueryui/1.8/jquery-ui.min.js"></script>  
 <style type="text/css">  
 #slider { margin: 10px; }  
 </style>  
 <script>  
 $(document).ready(function() {  
 $("#slider").slider();  
 });  
 </script>  
</head>  
<body style="font-size:62.5%;">  
   
<div id="slider"></div>  
  
</body>  
</html>

## Options

#### **disabled**Type: Boolean Default: false Disables (true) or enables (false) the slider. Can be set when initialising (first creating) the slider. Code examplesInitialize a slider with the disabled option specified. $( ".selector" ).slider({ disabled: true });Get or set the disabled option, after init. //getter var disabled = $( ".selector" ).slider( "option", "disabled" ); //setter $( ".selector" ).slider( "option", "disabled", true );

#### **animate**Type: Boolean, String, Number Default: false Whether to slide handle smoothly when user click outside handle on the bar. Will also accept a string representing one of the three predefined speeds ("slow", "normal", or "fast") or the number of milliseconds to run the animation (e.g. 1000). Code examplesInitialize a slider with the animate option specified. $( ".selector" ).slider({ animate: true });Get or set the animate option, after init. //getter var animate = $( ".selector" ).slider( "option", "animate" ); //setter $( ".selector" ).slider( "option", "animate", true );

#### **max**Type: Number Default: 100 The maximum value of the slider. Code examplesInitialize a slider with the max option specified. $( ".selector" ).slider({ max: 7 });Get or set the max option, after init. //getter var max = $( ".selector" ).slider( "option", "max" ); //setter $( ".selector" ).slider( "option", "max", 7 );

#### **min**Type: Number Default: 0 The minimum value of the slider. Code examplesInitialize a slider with the min option specified. $( ".selector" ).slider({ min: -7 });Get or set the min option, after init. //getter var min = $( ".selector" ).slider( "option", "min" ); //setter $( ".selector" ).slider( "option", "min", -7 );

#### **orientation**Type: String Default: 'horizontal' This option determines whether the slider has the min at the left, the max at the right or the min at the bottom, the max at the top. Possible values: 'horizontal', 'vertical'. Code examplesInitialize a slider with the orientation option specified. $( ".selector" ).slider({ orientation: 'vertical' });Get or set the orientation option, after init. //getter var orientation = $( ".selector" ).slider( "option", "orientation" ); //setter $( ".selector" ).slider( "option", "orientation", 'vertical' );

#### **range**Type: Boolean, String Default: false If set to true, the slider will detect if you have two handles and create a stylable range element between these two. Two other possible values are 'min' and 'max'. A min range goes from the slider min to one handle. A max range goes from one handle to the slider max. Code examplesInitialize a slider with the range option specified. $( ".selector" ).slider({ range: 'min' });Get or set the range option, after init. //getter var range = $( ".selector" ).slider( "option", "range" ); //setter $( ".selector" ).slider( "option", "range", 'min' );

#### **step**Type: Number Default: 1 Determines the size or amount of each interval or step the slider takes between min and max. The full specified value range of the slider (max - min) needs to be evenly divisible by the step. Code examplesInitialize a slider with the step option specified. $( ".selector" ).slider({ step: 5 });Get or set the step option, after init. //getter var step = $( ".selector" ).slider( "option", "step" ); //setter $( ".selector" ).slider( "option", "step", 5 );

#### **value**Type: Number Default: 0 Determines the value of the slider, if there's only one handle. If there is more than one handle, determines the value of the first handle. Code examplesInitialize a slider with the value option specified. $( ".selector" ).slider({ value: 37 });Get or set the value option, after init. //getter var value = $( ".selector" ).slider( "option", "value" ); //setter $( ".selector" ).slider( "option", "value", 37 );

#### **values**Type: Array Default: null This option can be used to specify multiple handles. If range is set to true, the length of 'values' should be 2. Code examplesInitialize a slider with the values option specified. $( ".selector" ).slider({ values: [1,5,9] });Get or set the values option, after init. //getter var values = $( ".selector" ).slider( "option", "values" ); //setter $( ".selector" ).slider( "option", "values", [1,5,9] );

## Events

#### **create**Type: slidecreate This event is triggered when slider is created. Code examplesSupply a callback function to handle the create event as an init option. $( ".selector" ).slider({ create: function(event, ui) { ... } });Bind to the create event by type: slidecreate. $( ".selector" ).bind( "slidecreate", function(event, ui) { ... });

#### **start**Type: slidestart This event is triggered when the user starts sliding. Code examplesSupply a callback function to handle the start event as an init option. $( ".selector" ).slider({ start: function(event, ui) { ... } });Bind to the start event by type: slidestart. $( ".selector" ).bind( "slidestart", function(event, ui) { ... });

#### **slide**Type: slide This event is triggered on every mouse move during slide. Use ui.value (single-handled sliders) to obtain the value of the current handle, $(..).slider('value', index) to get another handles' value. Return false in order to prevent a slide, based on ui.value. Code examplesSupply a callback function to handle the slide event as an init option. $( ".selector" ).slider({ slide: function(event, ui) { ... } });Bind to the slide event by type: slide. $( ".selector" ).bind( "slide", function(event, ui) { ... });

#### **change**Type: slidechange This event is triggered on slide stop, or if the value is changed programmatically (by the value method). Takes arguments event and ui. Use event.orginalEvent to detect whether the value changed by mouse, keyboard, or programmatically. Use ui.value (single-handled sliders) to obtain the value of the current handle, $(this).slider('values', index) to get another handle's value. Code examplesSupply a callback function to handle the change event as an init option. $( ".selector" ).slider({ change: function(event, ui) { ... } });Bind to the change event by type: slidechange. $( ".selector" ).bind( "slidechange", function(event, ui) { ... });

#### **stop**Type: slidestop This event is triggered when the user stops sliding. Code examplesSupply a callback function to handle the stop event as an init option. $( ".selector" ).slider({ stop: function(event, ui) { ... } });Bind to the stop event by type: slidestop. $( ".selector" ).bind( "slidestop", function(event, ui) { ... });

## Methods

### **destroy**Signature: .slider( "destroy" ) Remove the slider functionality completely. This will return the element back to its pre-init state.

### **disable**Signature: .slider( "disable" ) Disable the slider.

### **enable**Signature: .slider( "enable" ) Enable the slider.

### **option**Signature: .slider( "option" , optionName , [value] ) Get or set any slider option. If no value is specified, will act as a getter.

### **option**Signature: .slider( "option" , options ) Set multiple slider options at once by providing an options object.

### **widget**Signature: .slider( "widget" ) Returns the .ui-slider element.

### **value**Signature: .slider( "value" , [value] ) Gets or sets the value of the slider. For single handle sliders.

### **values**Signature: .slider( "values" , index , [value] ) Gets or sets the values of the slider. For multiple handle or range sliders.

## Theming

The jQuery UI Slider plugin uses the jQuery UI CSS Framework to style its look and feel, including colors and background textures. We recommend using the ThemeRoller tool to create and download custom themes that are easy to build and maintain.

If a deeper level of customization is needed, there are widget-specific classes referenced within the jquery.ui.slider.css stylesheet that can be modified. These classes are highlighed in bold below.

### Sample markup with jQuery UI CSS Framework classes

<div class="ui-slider **ui-slider-horizontal** ui-widget ui-widget-content ui-corner-all">

   <a style="left: 0%;" class="**ui-slider-handle** ui-state-default ui-corner-all" href="#"></a>

</div>

**Note: This is a sample of markup generated by the slider plugin, not markup you should use to create a slider. The only markup needed for that is <div><div>.**