* Overview
* Options
* Events
* Methods
* Theming

jQuery UI Droppable

## Overview

The jQuery UI Droppable plugin makes selected elements droppable (meaning they accept being dropped on by draggables). You can specify which (individually) or which kind of draggables each will accept.

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'):

* **ui.draggable** - current draggable element, a jQuery object.
* **ui.helper** - current draggable helper, a jQuery object
* **ui.position** - current position of the draggable helper { top: , left: }
* **ui.offset** - current absolute position of the draggable helper { top: , left: }

### Dependencies

* UI Core
* UI Widget
* UI Mouse
* [UI Draggable](http://docs.jquery.com/UI/Draggable)

### Example

* Demo
* View Source

Makes the div droppable (a drop target for a draggable).

$("#draggable").draggable();  
 $("#droppable").droppable({  
 drop: function() { alert('dropped'); }  
 });

<!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">  
 #draggable { width: 75px; height: 25px; background: silver; padding: 10px; }  
 #droppable { position: absolute; left: 250px; top: 0; width: 125px; height: 75px; background: gray; color: white; padding: 10px; }  
 </style>  
 <script>  
 $(document).ready(function() {  
 $("#draggable").draggable();  
 $("#droppable").droppable({  
 drop: function() { alert('dropped'); }  
 });  
 });  
 </script>  
</head>  
<body style="font-size:62.5%;">  
   
<div id="droppable">Drop here</div>  
<div id="draggable">Drag me</div>  
  
</body>  
</html>

## Options

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

#### **accept**Type: Selector, Function Default: '\*' All draggables that match the selector will be accepted. If a function is specified, the function will be called for each draggable on the page (passed as the first argument to the function), to provide a custom filter. The function should return true if the draggable should be accepted. Code examplesInitialize a droppable with the accept option specified. $( ".selector" ).droppable({ accept: '.special' });Get or set the accept option, after init. //getter var accept = $( ".selector" ).droppable( "option", "accept" ); //setter $( ".selector" ).droppable( "option", "accept", '.special' );

#### **activeClass**Type: String Default: false If specified, the class will be added to the droppable while an acceptable draggable is being dragged. Code examplesInitialize a droppable with the activeClass option specified. $( ".selector" ).droppable({ activeClass: 'ui-state-highlight' });Get or set the activeClass option, after init. //getter var activeClass = $( ".selector" ).droppable( "option", "activeClass" ); //setter $( ".selector" ).droppable( "option", "activeClass", 'ui-state-highlight' );

#### **addClasses**Type: Boolean Default: true If set to false, will prevent the ui-droppable class from being added. This may be desired as a performance optimization when calling .droppable() init on many hundreds of elements. Code examplesInitialize a droppable with the addClasses option specified. $( ".selector" ).droppable({ addClasses: false });Get or set the addClasses option, after init. //getter var addClasses = $( ".selector" ).droppable( "option", "addClasses" ); //setter $( ".selector" ).droppable( "option", "addClasses", false );

#### **greedy**Type: Boolean Default: false If true, will prevent event propagation on nested droppables. Code examplesInitialize a droppable with the greedy option specified. $( ".selector" ).droppable({ greedy: true });Get or set the greedy option, after init. //getter var greedy = $( ".selector" ).droppable( "option", "greedy" ); //setter $( ".selector" ).droppable( "option", "greedy", true );

#### **hoverClass**Type: String Default: false If specified, the class will be added to the droppable while an acceptable draggable is being hovered. Code examplesInitialize a droppable with the hoverClass option specified. $( ".selector" ).droppable({ hoverClass: 'drophover' });Get or set the hoverClass option, after init. //getter var hoverClass = $( ".selector" ).droppable( "option", "hoverClass" ); //setter $( ".selector" ).droppable( "option", "hoverClass", 'drophover' );

#### **scope**Type: String Default: 'default' Used to group sets of draggable and droppable items, in addition to droppable's accept option. A draggable with the same scope value as a droppable will be accepted. Code examplesInitialize a droppable with the scope option specified. $( ".selector" ).droppable({ scope: 'tasks' });Get or set the scope option, after init. //getter var scope = $( ".selector" ).droppable( "option", "scope" ); //setter $( ".selector" ).droppable( "option", "scope", 'tasks' );

### **tolerance**Type: String Default: 'intersect'

* Specifies which mode to use for testing whether a draggable is 'over' a droppable. Possible values: 'fit', 'intersect', 'pointer', 'touch'.
  + **fit**: draggable overlaps the droppable entirely
  + **intersect**: draggable overlaps the droppable at least 50%
  + **pointer**: mouse pointer overlaps the droppable
  + **touch**: draggable overlaps the droppable any amount

## Code examplesInitialize a droppable with the tolerance option specified. $( ".selector" ).droppable({ tolerance: 'fit' });Get or set the tolerance option, after init. //getter var tolerance = $( ".selector" ).droppable( "option", "tolerance" ); //setter $( ".selector" ).droppable( "option", "tolerance", 'fit' );

Events

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

#### **activate**Type: dropactivate This event is triggered any time an accepted draggable starts dragging. This can be useful if you want to make the droppable 'light up' when it can be dropped on. Code examplesSupply a callback function to handle the activate event as an init option. $( ".selector" ).droppable({ activate: function(event, ui) { ... } });Bind to the activate event by type: dropactivate. $( ".selector" ).bind( "dropactivate", function(event, ui) { ... });

#### **deactivate**Type: dropdeactivate This event is triggered any time an accepted draggable stops dragging. Code examplesSupply a callback function to handle the deactivate event as an init option. $( ".selector" ).droppable({ deactivate: function(event, ui) { ... } });Bind to the deactivate event by type: dropdeactivate. $( ".selector" ).bind( "dropdeactivate", function(event, ui) { ... });

#### **over**Type: dropover This event is triggered as an accepted draggable is dragged 'over' (within the tolerance of) this droppable. Code examplesSupply a callback function to handle the over event as an init option. $( ".selector" ).droppable({ over: function(event, ui) { ... } });Bind to the over event by type: dropover. $( ".selector" ).bind( "dropover", function(event, ui) { ... });

#### **out**Type: dropout This event is triggered when an accepted draggable is dragged out (within the tolerance of) this droppable. Code examplesSupply a callback function to handle the out event as an init option. $( ".selector" ).droppable({ out: function(event, ui) { ... } });Bind to the out event by type: dropout. $( ".selector" ).bind( "dropout", function(event, ui) { ... });

#### **drop**Type: drop This event is triggered when an accepted draggable is dropped 'over' (within the tolerance of) this droppable. In the callback, $(this) represents the droppable the draggable is dropped on. ui.draggable represents the draggable. Code examplesSupply a callback function to handle the drop event as an init option. $( ".selector" ).droppable({ drop: function(event, ui) { ... } });Bind to the drop event by type: drop. $( ".selector" ).bind( "drop", function(event, ui) { ... });

## Methods

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

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

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

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

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

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

## Theming

The jQuery UI Droppable 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.droppable.css stylesheet that can be modified. These classes are highlighed in bold below.

### Sample markup with jQuery UI CSS Framework classes

<div class="**ui-droppable**"></div>

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