JQuery Drag and Drop to multiple div elements

Posted At : 24 March 2011 21:58 | Posted By : Shaun McCran
Related Categories: JQuery, Javascript, RIA

My only previous experience with Drag and Drop web development was with Flex 2, which I have to say was a bit of a nightmare. There was an astronomical amount of setup to do in terms of event listeners and mouse watching services, so I was reluctant to get back into 'Drag and Drop' with any other technology.

This article explains how straight forward the JQuery Drag and Drop interfaces are to use, and how I constructed a Drag and Drop example using multiple divs.

There is an example of a JQuery Drag and Drop demo here. http://www.mccran.co.uk/examples/jquery-drag-drop

What I needed to do was create a draggable element that could be dropped into one of several different divs. Think of the scenario where you are dropping a ballet paper into one of many possible selections. Based on this I needed to create a draggable element, and several 'target' divs that can be uniquely identified when the drag element is dropped within their bounds.

My first recommendation is to go and check out the JQuery UI demos here: http://jqueryui.com/demos/draggable/ . It is a great resource for JQuery UI examples, a lot of this article is based on several of the demo's from the site.

I've started off by including all the JQuery UI js files, the JQuery library and some CSS.

```
<link rel="stylesheet" href="jquery.ui.all.css">
="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.4.2/jquery.
<s/cript src="jquery.ui.core.js"></script>
```

```
<s/cript src="jquery.ui.widget.js"></script>

<s/cript src="jquery.ui.mouse.js"></script>
<s/cript src="jquery.ui.draggable.js"></script>
<s/cript src="jquery.ui.droppable.js"></script>

<link rel="stylesheet" href="demos.css">

<style>
.droppable { width: 150px; height: 150px;
        padding: 0.5em; float: left; margin: 10px; }

.draggable { width: 100px; height: 100px;
        padding: 0.5em; float: left; margin: 10px;
            cursor: hand; cursor: pointer;
        border: 1px solid #000;}
                </style>
```

Next create the div elements that make up the draggable element and the target elements.

```
<div class="draggable" class="ui-widget-content">
    <p>I return home when I'm not dropped in the right place</p>
                </div>

<div class="droppable ui-widget-header" id="element 1">
                <p>Drop me here</p>
                    </div>

<div class="droppable ui-widget-header" id="element 2">
                <p>Drop me here</p>
                    </div>

<div class="droppable ui-widget-header" id="element 3">
                <p>Drop me here</p>
                    </div>
```

I've given each of them unique ID's to show that they are being uniquely identified later.

Next create a function to handle the draggable element. I've set the value of 'revert:invalid' here as I want the div to snap back into its original position if it does not match a destination element.

```
                            <s/cript>
                         $(function() {
        $( ".draggable" ).draggable({ revert: "invalid" });

                $( ".droppable" ).droppable({
                    activeClass: "ui-state-hover",
                    hoverClass: "ui-state-active",
                    drop: function( event, ui ) {

                     var targetElem = $(this).attr("id");

                                    $( this )
                           .addClass( "ui-state-highlight" )
                                     .find( "p" )
                         .html( "Dropped! inside " + targetElem );

                                 alert(targetElem);

                            }
                      });
                  });
                </script>
```

Next inside the droppable function I've specified a few design classes, and a handler for a drop event. This is the event that will 'catch' when a user releases a div onto the target. You can interact with the target div in the same way as other JQuery elements like $(this). In this way you can identify exactly which element a user dropped the dragged object on.

There is an example of a JQuery Drag and Drop demo here. http://www.mccran.co.uk/examples/jquery-drag-drop