There were several goals to this code snippet.

1. To stay with just the jQuery-UI Library and not have to add an additional library for the effect.
2. To have code to the page that would add the tooltip behavior to links of a specific class regardless of what page in the CMS the link was on.
3. To have the tooltips show by link rather than hover (just what I needed for this project).
4. To load the tooltip by ajax so that each link, assigned an ajax loading url specific to that link, would create a tooltip with its own content (They don't all have to be the same even though generated from one snippet).
5. And lastly, to be able to target a specific portion of the ajax loading page so the target could be a stand-alone page. In other words,only the relevant code would load in the dialog, but if javascript was disabled, a standalone page matching the site template would be loaded. By using the same page, the content did not necessarily have to either come from a specific ajax file which would require keeping the content of two files synced, not come from the database (though it could) requiring building a backend interface.

First, this requires jQuery and jQuery-ui to be loaded. I'm using a jQuery-ui library wih all the widgets included, so if that isn't what you're using, you'll need to add the tooltip widget and any other libraries you would need for any effects you might add to the dialog.

Unlike the post on doing something similar with the ui dialog widget, this code creates, on load, a hidden div for each tip that stays on the page. There were a number of issues with trying to create and destroy the tooltip content on demand, so I bypassed it. Thisshould make for a faster response anyway.

I found a lot of ideas for making the tooltip work on click rather than hover, and some for loading the tip with ajax, but none for doing both, and doing both created some challenges.
The posts I found helful were Sanmai's post on stackoverflow here , Wizzud's post in the jQuery forum here , and Elliot's post on stackoverfow
here
.

The links for this code are formatted as My

Link

Here the rel property is structured as the target content in mypage.php where the target can point to a page or a portion of the page.

Here is the code.

```
$().ready(function() {
  /*
  // Ajax Tooltip intitialization functions
  // Initialize counter to create unique id's
  */
  var i = 1;

  // Go through each item
  $("a.clickTip").each(function() {

     // Add a unique id
     $(this).prop("id","clickTip_"+i);

     // Assign to a variable that can reach into sub-functions where $(this) cannot
     // NOTE: it is important to do this after assigning the ID or
     // multiple instances will fail by opening all instead of one instance
     // in short, $(this) while specific within this code block, creates event bahaviors
    // that are general to the class, so the last one would override the others
     var elem = $(this);

     // Get the ajax content url from the rel attribute.
     // NOTE: This is done with load, so target a specific element on the target page
     var url = $(this).attr("rel");

     // Create a hidden on page container for the tip
  var ajaxTip = $('<div id="ajax-tip_'+i+'" style="display:none;"></div>').appendTo('body');

     // Load needs to refer to the ID rather than just ajaxTip.load(...
     $('#'+ajaxTip.attr('id')).load(url);

     // Initialize the tooltip to the id (NOT TO $(this))
      elem.tooltip({
        position: { my: "left-30 bottom-20", at: "left center" },
        /*
        // Make the items attach to the element
```

```
        // (elem is specific $(this) is not since it is attached to the class
        */
        items: elem,
        content: function(callback) {
            var data = ajaxTip.html();
            callback(data)
        }
    });
    // Control event responses
    // Do not string these events as in elem.tooltip(...).off("mouseover"...).on(...
    // It caused me "Cannot access method before initialization" errors

    // Disable the mouseover event
    elem.off( "mouseover" );

    // Set the open method to the click event
    elem.on( "click", function(e){
            e.preventDefault();
            elem.tooltip( "open" );
    });

    // Optional: Set the close method to a timeout on the mouseout event
    elem.on("mouseout", function() {
        setTimeout(function(){
                elem.tooltip( "close" );
            }, 5000);
    });
    // Increment the counter
    i++;
 });

});
```

With this sort of method, I can apply one set of code and make any link a unique ui tooltip on the page. It is also allows me to have a complete page on the target in case javascript is disabled and only get the content I want for the dialog off that page. That makes it easier to update the information in only one place without having to use a database. I plan to just use basic tooltip using the title tag for hover tooltips, and this code for click activated tooltips.