# CENTRIQ TRAINING HIGHLIGHT SELECTED NAVIGATION GUIDE

For simple sites (no data added) that are typically not data driven (where you do not have to worry about CRUD functionality) the process to highlight the selected navigation is simple.

1. Check your template's navigation for any class that changes the look feel of the <li> or the <a>. Class names such as "selected" or "active" are common.
2. If you are able to locate the information above, open the proper style sheet and find the location for the definition of the style being applied.
3. After you have converted your template (you should have basic jquery link(s) at the bottom as well as an @RenderSection(scripts) at the bottom of the _Layout page) go to the bottom of the page under all other jquery references and use the following code as template: See the notes for changes in the code.

## Class Attached to the <li>:

```
//loop through all of the navigation and locate an HREF value of an ANCHOR tag //that matches the
browser URL.      If they match, apply our selected class to the LI.
 <script>
    //use the selector values found (or declared) in your style sheet for the active class – for
    the code below your stylesheet would look like: .mainmenu ul li.selected a {//code}
     $('.mainmenu ul li').each(function () {
    //in this example the class is attached to the li, so you must go one level further to
    //check the href value of the anchor tag and ensure it matches what is found in the
    //browser URL
         if ($(this).find('a').attr('href') == window.location.pathname) {
    //if they match add the correct class name. $(this) refers to the li (the last item in the
    DOM specified before the each function
              $(this).addClass('selected');
         }
     });
 </script>
```

## Class Attached to the <a>:

```
//loop through all of the navigation and locate an HREF value of an ANCHOR tag //that matches the
browser URL.      If they match, apply our selected class to the A tag.
 <script>
    //use the selector values found (or declared) in your style sheet for the active class – for
    the code below your stylesheet would look like: .mainmenu ul li a.selected {//code}
     $('.mainmenu ul li a').each(function () {
    //in this example the class is attached to the <a>, so you do hot have to go one level
    //further to check the href value of the anchor tag and ensure it matches what is found
    //in the browser URL
         if ($(this).attr('href') == window.location.pathname) {
    //if they match add the correct class name. $(this) refers to the a (the last item in the DOM
    specified before the each function
              $(this).addClass('selected');
         }
     });`
 </script>
```

## No Definition Provided for Your Template:

You will need to define the class yourself. *TIP:* Look for style that is used when you hover over navigation. You can use that styling for your selected/active class. Declare the appropriate selector values, then code your jQuery to match!

For more complex sites that **are** typically data driven (where you **do** have to worry about CRUD functionality) the process to highlight the selected navigation is a little more involved.

1. Check your template's navigation for any class that changes the look feel of the <li> or the <a>. Class names such as "selected" or "active" are common.
2. If you are able to locate the information above, open the proper style sheet and find the location for the definition of the style being applied.
3. After you have converted your template (you should have basic jquery link(s) at the bottom as well as an @RenderSection(scripts) at the bottom of the _Layout page) go to the bottom of the page under all other jquery references and use the following code as template: See the notes for changes in the code.

## Code Example for Simple Highlight:

*The commented code at the top of this is what you would use for simple highlight.*

```
//Loop thru all of the elements in our navigation and find the anchor tag
 //that matches the href in navigation.  Then add our selected class to that anchor
 //tag's li element.
 //$('.mainmenu ul li').each(function () {
 //    if ($(this).find('a').attr('href') == window.location.pathname) {
 //        $(this).addClass('selected');
 //    }
 //});

 //With the code above, when you navigate to any sub view (anything OTHER THAN index),
 //you losethe highlighted navigation.

 //Desired outcome is for the highlight to remain on the main nav NO MATTER WHICH view is
 //beingdisplayed in that controller EXCEPT for the HOME controller.
 //loop through the li elements in the nav
 $('.mainmenu ul li').each(function () {
       //if you want to match the stylesheet AND it has the assigned class on the anchor
       //modify code as follows:
       //$('.mainmenu ul li a').each(function () { //add the anchor to the DOM Selection
       //var fullAnchor = $(this).attr('href'); //remove the find('a') from the
       //fullAnchor (we are already there)
       //everything else stays the same.

       //get the full anchor tag and print to the console (in the inspector) var
     fullAnchor = $(this).find('a').attr('href');
     //console.log(fullAnchor);

     //split() the values by '/' (give us a collection) - of the controller Name(s)
     //log in the console
     var aController = fullAnchor.split('/');
     //console.log('aController[1]: ' + aController[1])
```

```
        //get the full context of the url (including any querystring values)
        //substring to remove EVERYTHING before the path
        //(mysite.com/home/index - would return home/index)
        var fullPath = (window.location.pathname + location.search).substr(1);

        //break the string by the '/'
        var pController = fullPath.split('/');
        //check the values in the console
        //console.log('pController[0]: ' + pController[0]);
        //console.log(' ----------------- ');
        //console.log(window.location.pathname);
        //console.log(fullAnchor);
        if (aController[1] == pController[0])
        {
            //apply the selected class to the li where the
            //a.href.controllerName == p.fullName.controllerName

            //locate the anchor tag inside the li and compare the href
            //to the current broswer location - add class if they match
            $(this).addClass('selected');
        }
        //resolve the logic issue.  Without this logic all links that have
        //Home as the controller (except Home/Index) get the class added.

        //make sure that all links show Home as capitalized, this is a case sensitve
        //comparison (if you use Url.Action() or Html.ActionLink(), you should not have any
        //issue as those are methods that require case sensitivity calling the action
        //(method).  If you use pathing ~/home/index, those will function in the browser but
        //will escape this logic).

        //if the controller is Home, but the path does NOT match the ENTIRE anchor - remove
        //the class (prevents 2 nav elements from showing selected simultaneously) if
        (aController[1] == 'Home' && window.location.pathname != fullAnchor)
        {
            $(this).removeClass();
        }
       });
</script>
```