Customizing the My Stuff Section

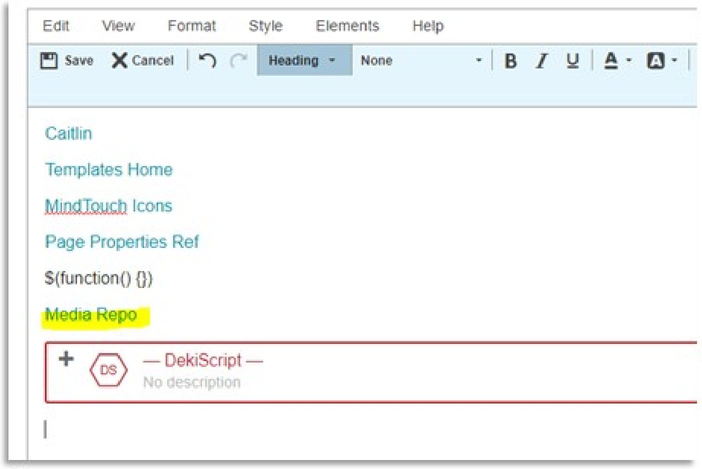
Last updated: Sep 25, 2018

Placing Text in DekiScript

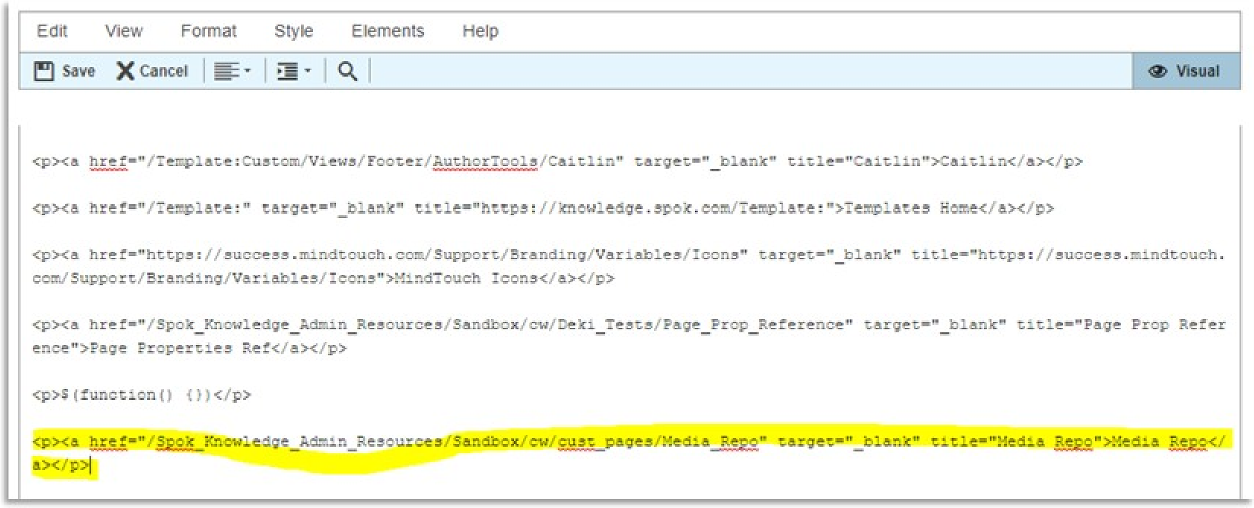
The main thing to remember is that any text between HTML tags needs to be in quotation marks.

The easiest way to get links and text into a DekiScript block is to create them first and then cut/paste into the script.

1. Create the link as you normally would using the **Link** dialog window:

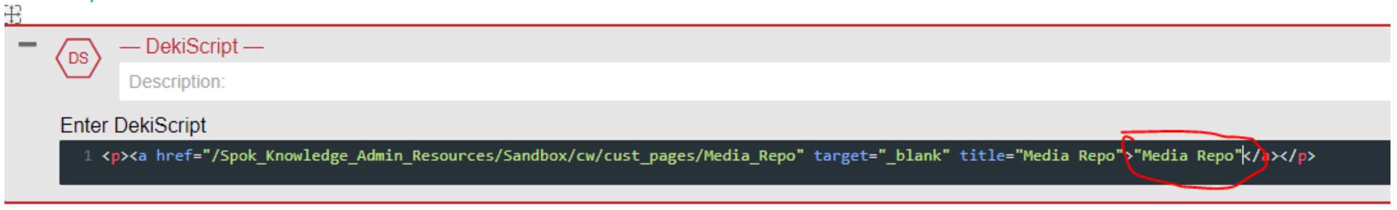


2. Switch the editor to HTML view and grab the raw HTML (include the <p> tags):



```
<p><a href="/Template:Custom/Views/Footer/AuthorTools/Caitlin" target="_blank" title="Caitlin">Caitlin</a></p>

<p><a href="/Template:" target="_blank" title="https://knowledge.spok.com/Template:">Templates Home</a></p>

<p><a href="https://success.mindtouch.com/Support/Branding/Variables/Icons" target="_blank" title="https://success.mindtouch.
com/Support/Branding/Variables/Icons">MindTouch Icons</a></p>

<p><a href="/Spok_Knowledge_Admin_Resources/Sandbox/cw/Deki_Tests/Page_Prop_Reference" target="_blank" title="Page Prop Refer
ence">Page Properties Ref</a></p>

<p>$(function() {})</p>

<p><a href="/Spok_Knowledge_Admin_Resources/Sandbox/cw/cust_pages/Media_Repo" target="_blank" title="Media Repo">Media Repo</
a></p>
```

3. Go back to the normal view and paste the HTML into the DekiScript block.
4. **Add quotation marks** around the link text.



```
<p><a href="/Spok_Knowledge_Admin_Resources/Sandbox/cw/cust_pages/Media_Repo" target="_blank" title="Media Repo">"Media Repo"</a></p>
```

Conditional Links By Path/Tag

You can customize the template so that some links or notes only appear in certain contexts.

For example, let's say Stacy only wants to see a certain link when she's working in the latest Messenger category. To do that, she can use the `string.split` function:

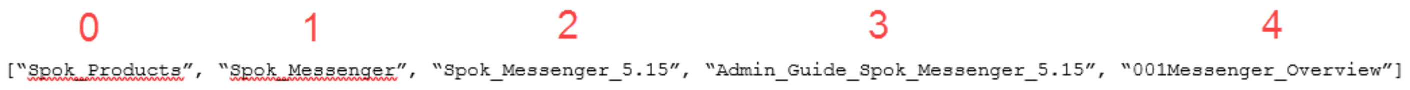
```
var stacy_path = string.split(page.path,"/");
```

- **page.path** - This returns the current page path. The path is considered one single string, e.g.:

  `["Spok_Products/Spok_Messenger/Spok_Messenger_5.15/Admin_Guide_Spok_Messenger_5.15/001Messenger_Overview"]`

- **"/"** -- Where to split the string (be careful on "Care Connect Web/Device Preferences" paths)

The result is a list with 5 strings. Each string has an index number, starting at 0:



["Spok_Products", "Spok_Messenger", "Spok_Messenger_5.15", "Admin_Guide_Spok_Messenger_5.15", "001Messenger_Overview"]

Next she needs to tell MindTouch which string to use. To do that, she creates another variable:

```
var stacy_path = string.split(page.path,"/");
var stacy_page = stacy_path[2]; //Use index 2 - Spok_Messenger_5.15
```

Now she can put them all together and create an 'if' statement:

```
var stacy_path = string.split(page.path,"/");
var stacy_page = stacy_path[2];
if(stacy_page == "Spok_Messenger_5.15"){
    //put links and/or notes here
}
```

Whatever Stacy puts within the brackets will only appear when she's editing pages in the Spok Messenger 5.15 category.

Other useful functions:

- **page.name** -- Only show me something on a certain page.
  I'm using the following script to display a link to my sandbox Media Repo when I'm editing the hardware guide:

  ```
  if(page.name == "Care_Connect_1.9_Hardware") {
      <p><a>href="/Spok_Knowledge_Admin_Resources/Sandbox/cw/cust_pages/Media_Repo" target="_blank" title="Media Repo">"Media Repo"</a></p>
  }
  ```

- **page.name** + **string.contains** -- Only show me something on pages whose names have a certain word.

  The **string.contains** function works like this:

  `string.contains(first, second, ignorecase)`

  This asks, *"Is the second string in the first string? And should we ignore the case?"* (default is to ***not*** ignore the case)

  ```
  string.contains("Caitlin is hungry", "Hungry")        //false. Hungry is capitalized

  string.contains("Caitlin is hungry", "angry")        //false. I'm never angry. That's absurd. Also it wouldn't match anyway.

  string.contains("Caitlin is hungry", "Hungry", true)   //true, because now we can ignore case.

  string.contains("Caitlin is hungry", "ungr", true)    //true. Doesn't have to be a full string.
  ```

  A practical use would be:

  ```
  if (string.contains(page.name, "Avaya", true) {
      <p>"Special link or note about Avaya"</p>
  }
  ```

- **page.tags** -- Only show me something on pages with a certain tag.

```
if (page.tags["Avaya"]) {
    <p>"Special link or note about Avaya"</p>
}
```

- **page.parent.tags** -- Only show me something on pages whose parent has a certain tag. We use this for the release notes heading link:

```
if (page.parent.tags['Release Notes']) {
        <div class="auth-tool release-note-link"><p><a class="F1"
href="/Spok_Knowledge_Admin_Resources/styleguide/manage_content/release_note_headers" title="Open in pop-
up window">'Release Note Header Format'</a></p></div>
   }
```

You can also mix and match:

- <one> && <two> -- Both one and two must be true
- <one> || <two> -- Either one, two, or both must be true.
- <one> && !<two> -- One is true AND two is **not** true

```
if (string.contains(page.name, "Avaya", true) && page.tags["Administration"]) {
    <p>"Special link or note about Avaya admin things or whatever"</p>
}

if (string.contains(page.name, "Avaya", true) && !page.tags["Avaya"]) {
    <p>"HEY PUT AN AVAYA TAG ON THIS!!!"</p>
}
```

CSS

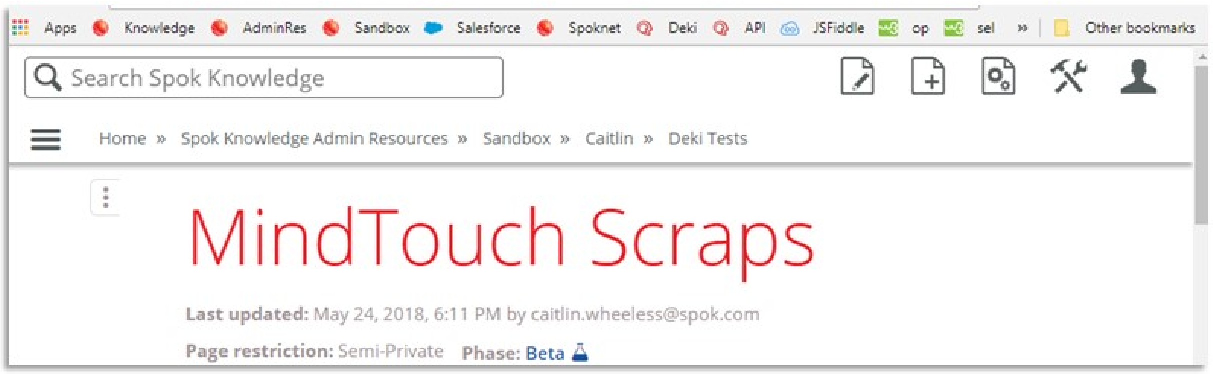
This template is loaded on every page that loads the table of contents, even when you can't see it. This means that you can create your own CSS to control things without disrupting anyone else.

Example: Hide Elements

You're working on a small screen and want to hide the header for the day:

```
.elm-header-logo-container, .elm-header-custom, div #mt-summary.mt-toggle-container {
    display: none !important
}
```

You'll probably want to use **!important** for a lot of things. This CSS is considered the lowest priority style, so it tends to get overwritten a lot.



MindTouch Scraps

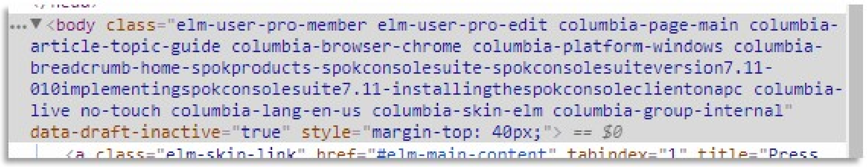
Last updated: May 24, 2018, 6:11 PM by caitlin.wheeless@spok.com

Page restriction: Semi-Private   Phase: Beta

Example: Highlight Outdated Links

You can use CSS to highlight links that still point to previous versions. We'll use Spok Console Suite 7.11 as an example.

1. Use your browser's inspector to look at the body classes. MindTouch adds a breadcrumbs body class that includes the page path:

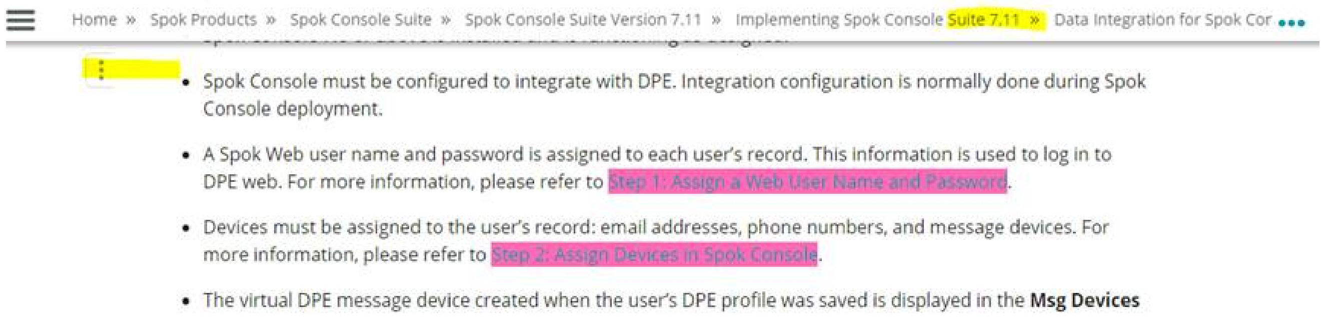

```
...▼<body class="elm-user-pro-member elm-user-pro-edit columbia-page-main columbia-
    article-topic-guide columbia-browser-chrome columbia-platform-windows columbia-
    breadcrumb-home-spokproducts-spokconsolesuite-spokconsolesuiteversion7.11-
    010implementingspokconsolesuite7.11-installingthespokconsoleclientonapc columbia-
    live no-touch columbia-lang-en-us columbia-skin-elm columbia-group-internal"
    data-draft-inactive="true" style="margin-top: 40px;"> == $0
```

2. You can leverage this to create a CSS rule selecting a substring of the class name (using the format [<*attribute name*>*="<*substring*>"):

```
/* [class*="spokconsolesuiteversion7.11"] -- this tells it to only apply the CSS to pages under Spok
Console Suite Version 7.11.
The class name removes spaces, so it becomes "spokconsolesuiteversion7.11" (see in the screenshot above) */

body[class*="spokconsolesuiteversion7.11"] a[href*="7.10"],
body[class*="spokconsolesuiteversion7.11"] a[href*="7.9"],
body[class*="spokconsolesuiteversion7.11"] a[href*="7.8"] {
    background: hotpink;
}
```

CSS selector reference: https://www.w3schools.com/cssref/css_selectors.asp

Here is the result:


Home » Spok Products » Spok Console Suite » Spok Console Suite Version 7.11 » Implementing Spok Console Suite 7.11 » Data Integration for Spok Cor •••

- Spok Console must be configured to integrate with DPE. Integration configuration is normally done during Spok Console deployment.
- A Spok Web user name and password is assigned to each user's record. This information is used to log in to DPE web. For more information, please refer to Step 1: Assign a Web User Name and Password.
- Devices must be assigned to the user's record: email addresses, phone numbers, and message devices. For more information, please refer to Step 2: Assign Devices in Spok Console.
- The virtual DPE message device created when the user's DPE profile was saved is displayed in the **Msg Devices**

That highlighting will disappear once the page is opened in the editor. To display it in the editor, you'll need to use JavaScript (see the use case example below).

JavaScript/jQuery

Similar to CSS, you can use JavaScript to affect all topic pages.

The most common uses cases probably involve interacting with the editor to make quick, mass changed to a topic. The AuthorTools template has a couple examples of how you could do this (the fixListNote and the selectPhase functions): https://knowledge.spok.com/Template:Custom/Views/Footer/AuthorTools

The editor is in an iframe, so it's a little trickier than normal. Every function will need to have something like this:

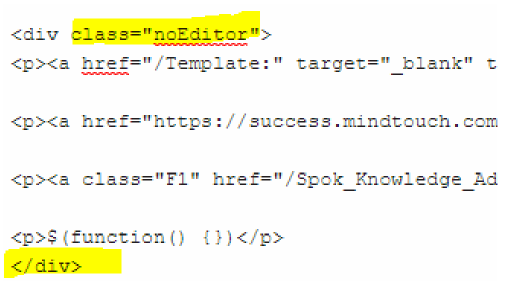
```
var iframe_content = $("iframe.cke_wysiwyg_frame.cke_reset").contents();
iframe_content.find //element you want to work on
```

The good (or bad!) thing about using Javascript on page content is that when you click 'Save' everything you did to the page text is permanent.

› Example: Show "My Stuff" Links Outside the Editor

This will allow you to access your "My Stuff" links without editing the page.

1. From your template, switch to the HTML view and create a div wrapping the links you'll want to use. Give this div a unique class name:

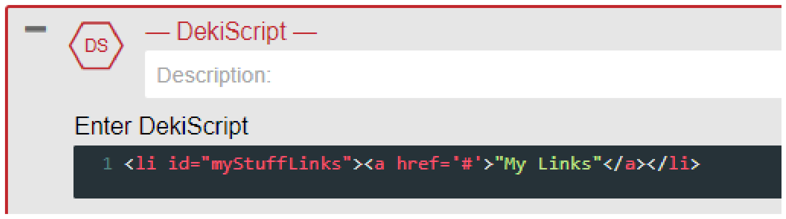


```
<div class="noEditor">
<p><a href="/Template:" target="_blank" t

<p><a href="https://success.mindtouch.com

<p><a class="F1" href="/Spok_Knowledge_Ad

<p>$(function() {})</p>
</div>
```

2. Create a button/link/div in your template.

   In this example, you'll be adding a link to the user menu, which is actually a list. Add a DekiScript block with the following:

```
<li id="myStuffLinks"><a href='#'>"My Links"</a></li>
```



```
<li id="myStuffLinks"><a href='#'>"My Links"</a></li>
```

3. If you don't have one already, add a JavaScript block.
   1. Create a variable holding a copy of the links you want to access.
   2. Use the `.hide()` method to make sure they stay hidden until you're ready to view them.

```
$(function() {
    var noEditor = $('.noEditor').clone();
    noEditor.hide();
})
```

4. Figure out where you want the links to appear. In this example, we'll put them below the table of contents:

```
$(function() {
    var noEditor = $('.noEditor').clone();
    noEditor.hide();
    noEditor.appendTo( $('div.spok-sticky-wrap') );
})
```

5. Move your clickable link to the user menu:

```
$(function() {
    var noEditor = $('.noEditor').clone();
    noEditor.hide();
    noEditor.appendTo( $('div.spok-sticky-wrap') );

    $('#myStuffLinks').prependTo( $('ol.mt-user-menu') ); //will appear before the Edit action

})
```

6. Assign an event handler to the link to show/hide your cloned block:

```
$(function() {
    var noEditor = $('.noEditor').clone();
    noEditor.hide();
    noEditor.appendTo( $('div.spok-sticky-wrap') );

    $('#myStuffLinks').prependTo( $('ol.mt-user-menu') );

    $('#myStuffLinks').on("click", function(e) {
        e.preventDefault();
        //'e.preventDefault' tells it not to try to follow the link
        noEditor.toggle();
    });
})
```

7. Finally, add a bit of CSS to position things a bit. Create a CSS block and add the following:

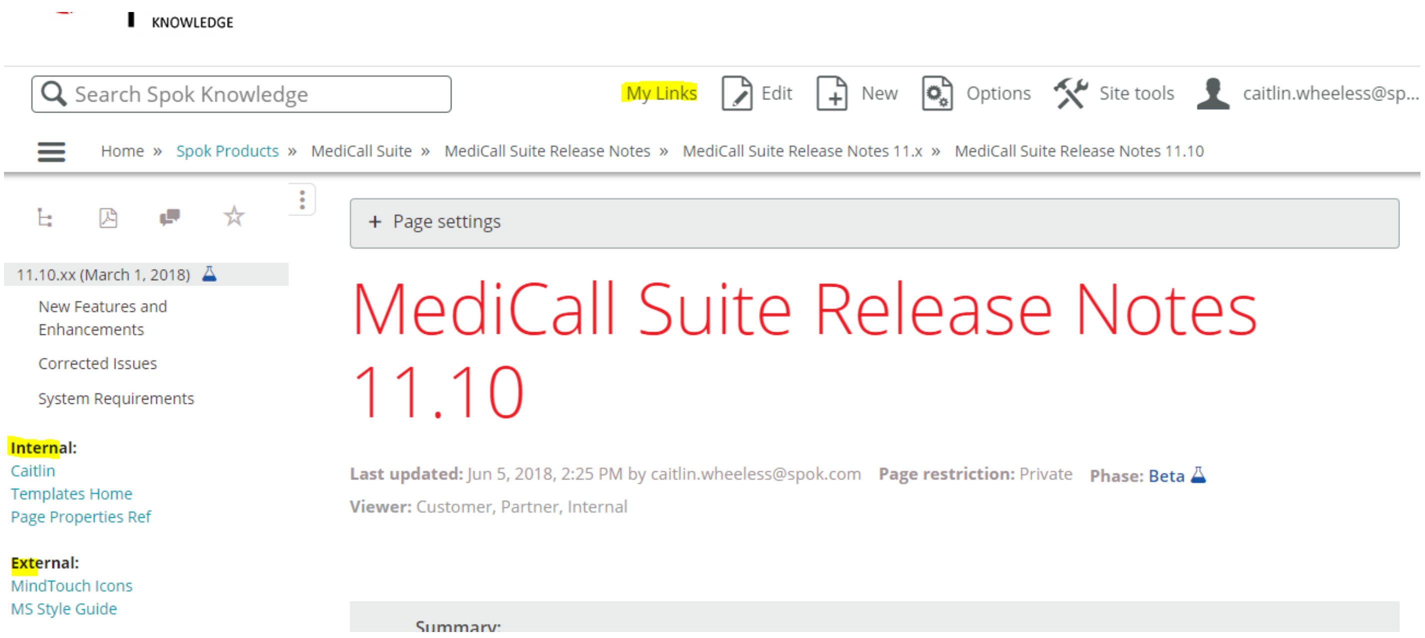
```
.spok-sticky-wrap > .noEditor {
    padding-left: 1em;
    font-size: 87.5%
}
.spok-sticky-wrap > .noEditor p {
    margin-bottom: 0
}

.page-mode-editor .spok-sticky-wrap > .noEditor {
    display: none;
}

#myStuffLinks {
    position: relative;
}

#myStuffLinks a {
    position: relative;
    top: .5em;
}
```

This doesn't work if you have your links sorted into show/hide blocks. Clicking the show/hide header won't do anything. You'll need to add JavaScript to reassign the show/hide function to the cloned links.

Alternatively, you can manually create a copy in your template front and then adjust the JavaScript as follows:

```
var noEditor = $('.noEditor'); // removed this bit: .clone();
    noEditor.hide();
    noEditor.appendTo( $('div.spok-sticky-wrap') );
```

Here is the result:



MediCall Suite Release Notes 11.10

Last updated: Jun 5, 2018, 2:25 PM by caitlin.wheeless@spok.com **Page restriction:** Private **Phase:** Beta

Viewer: Customer, Partner, Internal

Summary:

Combining DekiScript and JavaScript/CSS

In some cases you might want to restrict JavaScript or CSS to certain contexts and need to use the DekiScript properties described above.

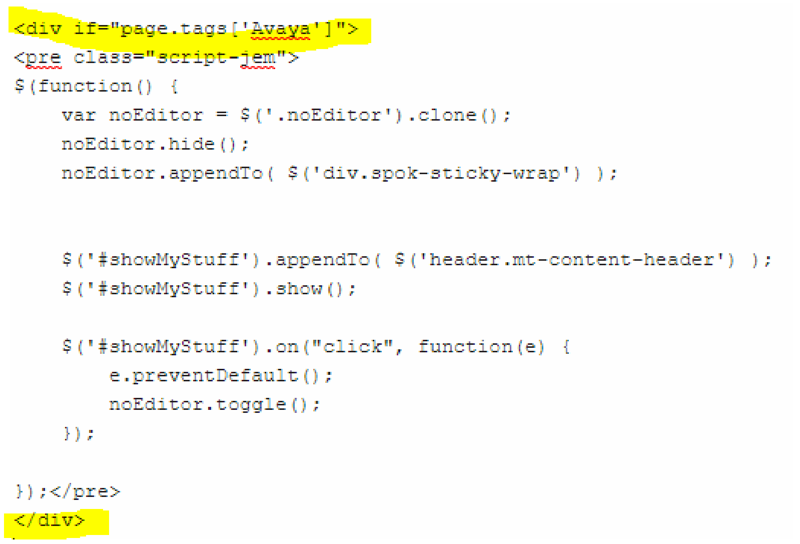
There are several ways you can accomplish this, depending on how complex your variables are.

If it's a simple one-liner, you can nest the entire block within DekiScript, e.g.:

```
var page_path = string.split(page.path,"/");
var page_2 = page_path[2];
if(page_2 == "Spok_Messenger_5.15"){
    <style type="text/css">".show-hide-content {display: block !important}"</style>
}
```

The script above would display show/hide content by default when viewing pages under the Spok Messenger 5.15 category.

If the DekiScript is simple but the JavaScript or CSS is complex, switch to the HTML view and wrap the script in a div element. Give that div an "if" attribute (similar to what we have for internal-only content):



```
<div if="page.tags['Avaya']">
<pre class="script-jem">
$(function() {
    var noEditor = $('.noEditor').clone();
    noEditor.hide();
    noEditor.appendTo( $('div.spok-sticky-wrap') );


    $('#showMyStuff').appendTo( $('header.mt-content-header') );
    $('#showMyStuff').show();

    $('#showMyStuff').on("click", function(e) {
        e.preventDefault();
        noEditor.toggle();
    });

});</pre>
</div>
```

However, if it's a complex JS/CSS block AND a complex DekiScript filter, you'll need to create a custom variable to use first.

For example, you only want to access your "My Stuff" links when working in the Admin Resources reuse section or in your sandbox (using the JavaScript sample above). My sandbox is named "cw."

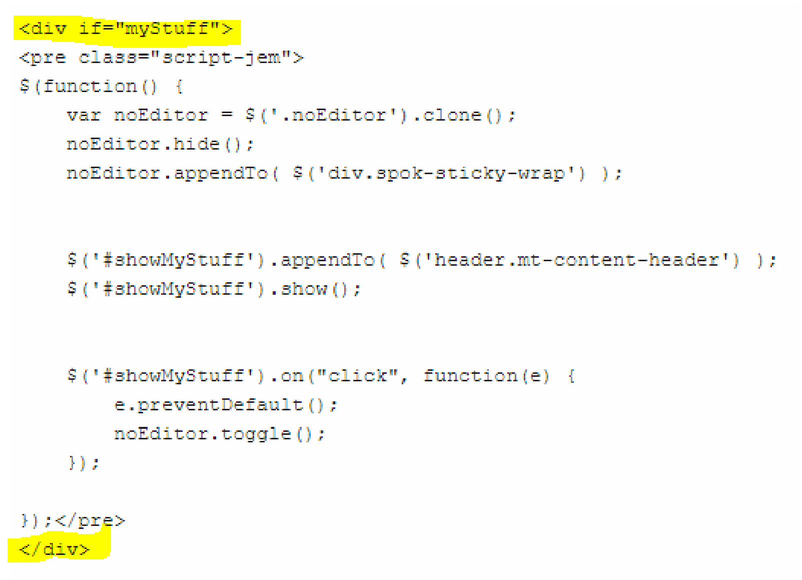
1. Create a DekiScript block to handle the paths:

```
var page_path = string.split(page.path,"/");
var page_1 = page_path[1];
var page_2 = page_path[2];

if(page_2 == "cw" || page_1 == "Reuse") {

}
```

2. Add a custom variable that is 'false' by default, but changes to true when your conditions are met:

```
var page_path = string.split(page.path,"/");
var page_1 = page_path[1];
var page_2 = page_path[2];
var myStuff = false;

if(page_2 == "cw" || page_1 == "Reuse") {
    var myStuff = true;
}
```

3. Now use that variable in your "if" wrapper:



```
<div if="myStuff">
<pre class="script-jem">
$(function() {
    var noEditor = $('.noEditor').clone();
    noEditor.hide();
    noEditor.appendTo( $('div.spok-sticky-wrap') );


    $('#showMyStuff').appendTo( $('header.mt-content-header') );
    $('#showMyStuff').show();


    $('#showMyStuff').on("click", function(e) {
        e.preventDefault();
        noEditor.toggle();
    });

});</pre>
</div>
```

Use Case Example

In this example, you'll use your template to:

1. Create a custom CSS to highlight outdated links.
2. Use jQuery to highlight those links in the editor.
3. Use jQuery/JavaScript/regex to update those links.

We'll use the Spok Console Suite 7.11 scenario from above.

Create Custom CSS Highlighting Outdated Links

The CSS section above has an example of a quick and simple way to create a CSS rule to do this, but you might want more control over where the highlighting occurs. To do that, you can use a combination of CSS and DekiScript.

You want to see links to Spok Console 7.8, 7.9, and 7.10 everywhere under the Spok Console Suite product category ***except*** for under the following paths:

```
Spok_Products/Spok_Console_Suite/Spok_Console_Suite_Version_7.10
Spok_Products/Spok_Console_Suite/Spok_Console_Suite_Version_7.9
Spok_Products/Spok_Console_Suite/Spok_Console_Suite_Version_7.8
```

Write a nested DekiScript "if" statement targeting those paths.

- The first verifies that you're working in the Spok Console Suite category.
- The second excludes Spok Console 7.8, 7.9, and 7.10.

```
var page_path = string.split(page.path,"/");
var page_1 = page_path[1];
var page_2 = page_path[2];
if(page_1 == "Spok_Console_Suite"){
    if (page_2 == "Spok_Console_Suite_Version_7.10" || page_2 == "Spok_Console_Suite_Version_7.9" || page_2 == "Spok_Console_Suite_Version_7.8") {
        return; //do nothing
    }
    else {
        <style type="text/css">"a[href*=\'7.10\'], a[href*=\'7.9\'], a[href*=\'7.8\'] {background: hotpink}"</style> //create style block
    }
}
```

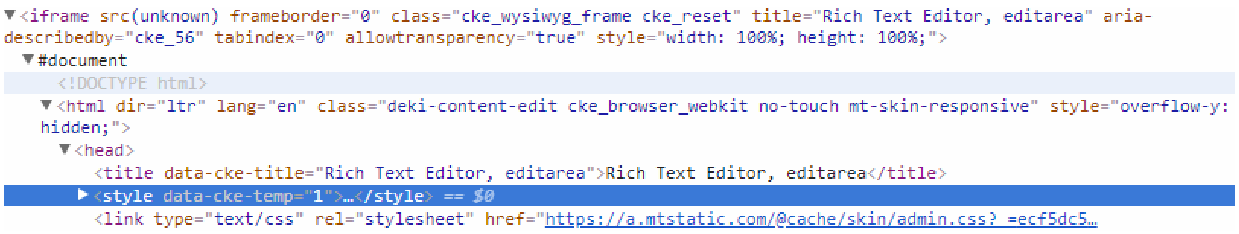
Note the formatting in the style tags. This follows MindTouch's requirements for including HTML in a DekiScript block: https://success.mindtouch.com/Suppor...lines/Add_HTML

Highlight Outdated Links in the Editor

Write the JavaScript Function

Start with writing the JavaScript function to apply a hot pink background to outdated links in the editor iframe.

If you look at the HTML source of the iframe, it's essentially an HTML page within the page -- including its own <head> and <style> tags:



```
▼<iframe src(unknown) frameborder="0" class="cke_wysiwyg_frame cke_reset" title="Rich Text Editor, editarea" aria-
describedby="cke_56" tabindex="0" allowtransparency="true" style="width: 100%; height: 100%;">
  ▼#document
      <!DOCTYPE html>
    ▼<html dir="ltr" lang="en" class="deki-content-edit cke_browser_webkit no-touch mt-skin-responsive" style="overflow-y:
    hidden;">
      ▼<head>
          <title data-cke-title="Rich Text Editor, editarea">Rich Text Editor, editarea</title>
        ▶<style data-cke-temp="1">…</style> == $0
          <link type="text/css" rel="stylesheet" href="https://a.mtstatic.com/@cache/skin/admin.css?_=ecf5dc5…
```

You can use this to apply a temporary style by adding a <style> block under the iframe <head>:

```
function hotPinkLinks() {
    var iframe_content = $("iframe.cke_wysiwyg_frame.cke_reset").contents(),     //targets the editor content
        hot_pink = "<style type='text/css'>a[href*='7.10'], a[href*='7.9'], a[href*='7.8'] {background: hotpink}</style>";

    iframe_content.find("head").each(function(){       //find the head element
        $(this).append(hot_pink );         //append our style block to it.
     });
};
```

Add an Event Handler

Anything you want to do in the editor with jQuery will need to be initiated with a button/link/something you click. This is because the editor doesn't "exist" when the page loads, so JavaScript targeting the editor has nowhere to look when the page loads.

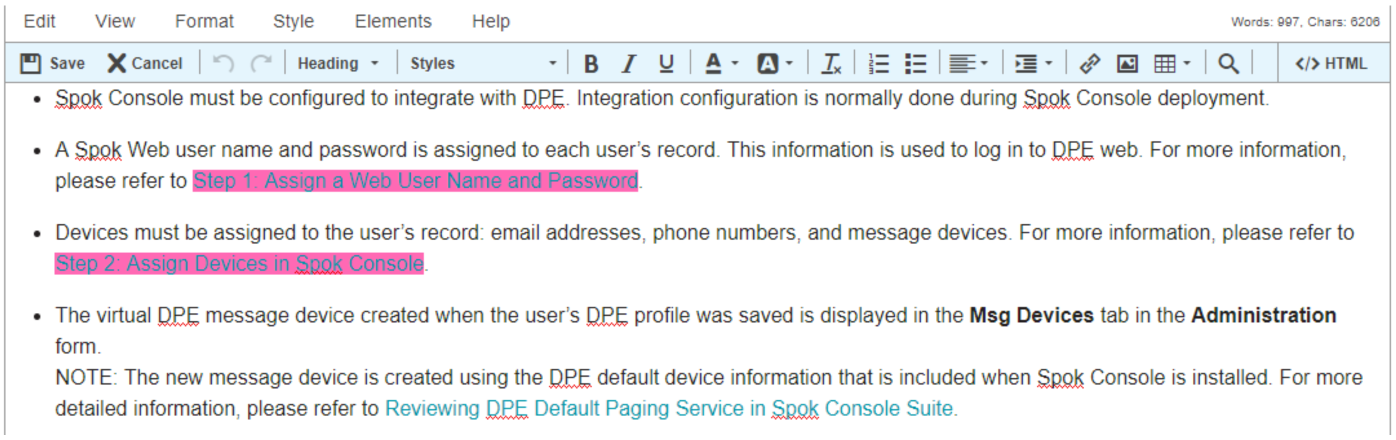
1. Switch to HTML view and add the following:

```
<p><button id="pinkLinks">Show outdated links</button></p>
```

2. Then add an event handler to run the `hotPinkLinks()` function when the button is clicked:

```
function hotPinkLinks() {
    var iframe_content = $("iframe.cke_wysiwyg_frame.cke_reset").contents(),
        hot_pink = "<style type='text/css'>a[href*='7.10'], a[href*='7.9'], a[href*='7.8'] {background: hotpink}</style>";

    iframe_content.find("head").each(function(){
        $(this).append(hot_pink );
     });
};

$(function() {        //this ensures that the following doesn't happen until the page is finished loading
    $('#pinkLinks').on("click", hotPinkLinks);    //adds the event handler
});
```

Now when you open the editor and click the "Show outdated links" button, any outdated links will be highlighted:



- Spok Console must be configured to integrate with DPE. Integration configuration is normally done during Spok Console deployment.
- A Spok Web user name and password is assigned to each user's record. This information is used to log in to DPE web. For more information, please refer to Step 1: Assign a Web User Name and Password.
- Devices must be assigned to the user's record: email addresses, phone numbers, and message devices. For more information, please refer to Step 2: Assign Devices in Spok Console.
- The virtual DPE message device created when the user's DPE profile was saved is displayed in the **Msg Devices** tab in the **Administration** form.
  NOTE: The new message device is created using the DPE default device information that is included when Spok Console is installed. For more detailed information, please refer to Reviewing DPE Default Paging Service in Spok Console Suite.

Conditionalize the Script and Button

The easiest way to ensure the button only appears when viewing Spok Console Suite documentation is to move it to the DekiScript you created earlier (note the quotation marks around the button text):

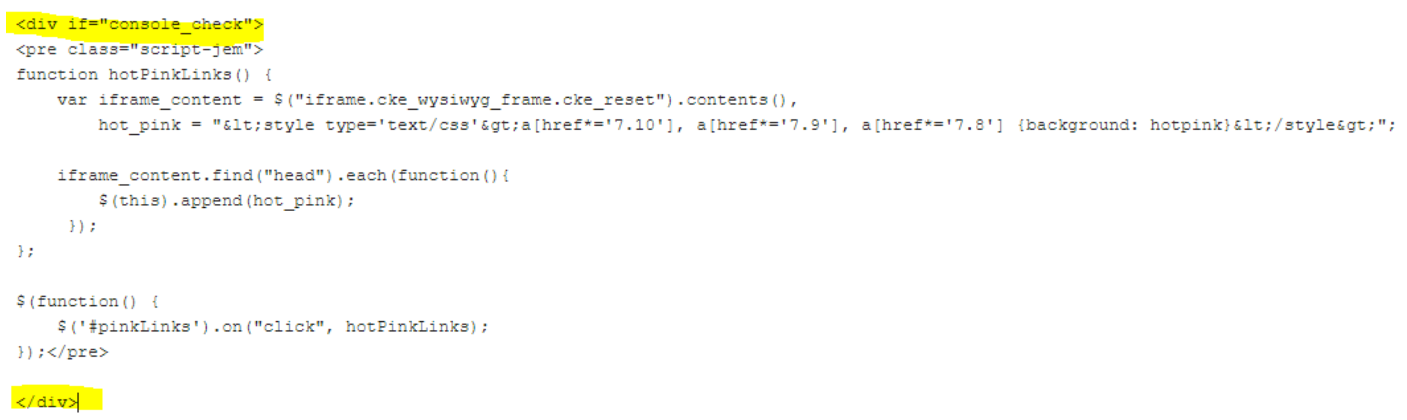
```
var page_path = string.split(page.path,"/");
var page_1 = page_path[1];
var page_2 = page_path[2];

if(page_1 == "Spok_Console_Suite"){
    if (page_2 == "Spok_Console_Suite_Version_7.10" || page_2 == "Spok_Console_Suite_Version_7.9" || page_2
== "Spok_Console_Suite_Version_7.8") {
        return;
     }
     else {
         <style type="text/css">"a[href*=\'7.10\'], a[href*=\'7.9\'], a[href*=\'7.8\'] {background:
hotpink}"</style>
         <p><button id="pinkLinks">"Show outdated links"</button></p> //move button here
     }
}
```

But for the purposes of this use case example, you'll also prevent the entire Javascript block from loading unless you're using it.

Using the method discussed above, add a custom variable to your DekiScript:

```
var page_path = string.split(page.path,"/");
var page_1 = page_path[1];
var page_2 = page_path[2];
var console_check = false;     //new variable

if(page_1 == "Spok_Console_Suite"){
    if (page_2 == "Spok_Console_Suite_Version_7.10" || page_2 == "Spok_Console_Suite_Version_7.9" || page_2
== "Spok_Console_Suite_Version_7.8") {
        return;
     }
     else {
         <style type="text/css">"a[href*=\'7.10\'], a[href*=\'7.9\'], a[href*=\'7.8\'] {background:
hotpink}"</style>
         <p><button id="pinkLinks">"Show outdated links"</button></p>
         var console_check = true;    //reset the variable value
     }
}
```

Next, switch your template to the HTML view and wrap the JavaScript in a div calling the custom "if" tag:



```
<div if="console_check">
<pre class="script-jem">
function hotPinkLinks() {
    var iframe_content = $("iframe.cke_wysiwyg_frame.cke_reset").contents(),
        hot_pink = "&lt;style type='text/css'&gt;a[href*='7.10'], a[href*='7.9'], a[href*='7.8'] {background: hotpink}&lt;/style&gt;";

    iframe_content.find("head").each(function(){
        $(this).append(hot_pink);
     });
};

$(function() {
    $('#pinkLinks').on("click", hotPinkLinks);
});</pre>

</div>
```

Use JavaScript and jQuery to Fix Outdated Links

You can use a combination of jQuery and JavaScript string functions to target outdated links and fix the version number (this, of course, assumes the same page with the same name exists in Spok 7.11 and that only the number identifier needs to be updated. So use this with caution or you'll be turning outdated links into broken links.)

You can use JavaScript string methods to locate and update links. Here is a good reference for the most common functions: https://www.w3schools.com/jsref/jsref_obj_string.asp

A few guidelines:

- The string methods never change the original string. You can only use them to create a new string. Therefore you'll always need a separate variable to "hold" the new string.
- Many string methods allow you to search with regex (https://regex101.com/ can help you create complex searches). This is important if you're using the `.replace()` method. By default, that method only replaces the first instance of a string. If you want to replace every instance, you'll need to use the regex global modifier: `/thing_I_want_to_replace/g` .

1. Create a new function and add the iframe_content variable.

```
function fixPinkLinks() {
    var iframe_content = $("iframe.cke_wysiwyg_frame.cke_reset").contents();

    iframe_content.find("a").each(function(){
        //code here
     });
};
```

In a typical jQuery function, you could target all the outdated links pretty easily:

```
$('a[href*="7.8"], a[href*="7.9"], a[href*="7.8"]').each(function() {
    //do stuff
})
```

But the `.find()` method isn't as flexible, so you need to use simple selectors like the element type ("a") or id ("#pinkLinks") or class name (".inline-phase")

2. Create a variable to read the `href` attribute of each link, and then copy it into a second variable. The first variable is the string we'll read, and the second variable is the string we'll manipulate:

```
function fixPinkLinks() {
    var iframe_content = $("iframe.cke_wysiwyg_frame.cke_reset").contents();

    iframe_content.find("a").each(function(){
        var current_href = $(this).attr("href"), //get the href of the link
            new_href = current_href; //copy of the current href that we can modify
     });
};
```

3. Add `if` statements to check for outdated links:

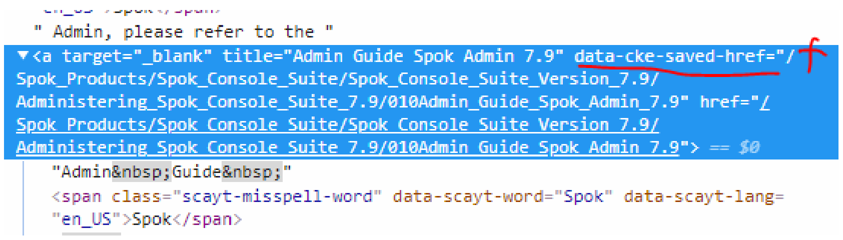
```
function fixPinkLinks() {
    var iframe_content = $("iframe.cke_wysiwyg_frame.cke_reset").contents();

    iframe_content.find("a").each(function(){
        var current_href = $(this).attr("href"),
            new_href = current_href;

        if (current_href.includes("7.8") == true) { //search using the string .includes() method

        }
        if (current_href.includes("7.9") == true) {

        }
        if (current_href.includes("7.10") == true) {

        }
     });
};
```

4. Use the `.replace()` method to search for the old version and replace it with the new one:

```
function fixPinkLinks() {
    var iframe_content = $("iframe.cke_wysiwyg_frame.cke_reset").contents();

    iframe_content.find("a").each(function(){
        var current_href = $(this).attr("href"),
            new_href = current_href;

        if (current_href.includes("7.8") == true) {
            new_href = new_href.replace(/7.8/g, "7.11"); //use the string .replace() method to update
version
        }
        if (current_href.includes("7.9") == true) {
            new_href = new_href.replace(/7.9/g, "7.11");
        }
        if (current_href.includes("7.10") == true) {
            new_href = new_href.replace(/7.10/g, "7.11");
        }
     });
};
```

5. Now we need to ensure that the current link's href is updated to reflect these changes.

MindTouch likes to add a lot of custom elements/tags, and they do this with links in the editor view:



```
  " Admin, please refer to the "
▼<a target="_blank" title="Admin Guide Spok Admin 7.9" data-cke-saved-href="/
Spok_Products/Spok_Console_Suite/Spok_Console_Suite_Version_7.9/
Administering_Spok_Console_Suite_7.9/010Admin_Guide_Spok_Admin_7.9" href="/
Spok_Products/Spok_Console_Suite/Spok_Console_Suite_Version_7.9/
Administering_Spok_Console_Suite_7.9/010Admin_Guide_Spok_Admin_7.9"> == $0
    "Admin&nbsp;Guide&nbsp;"
    <span class="scayt-misspell-word" data-scayt-word="Spok" data-scayt-lang=
    "en_US">Spok</span>
```

So we can't just update the `href` attribute like we normally would. Instead, we need to update `data-cke-saved-href` :

```
function fixPinkLinks() {
    var iframe_content = $("iframe.cke_wysiwyg_frame.cke_reset").contents();

    iframe_content.find("a").each(function(){
        var current_href = $(this).attr("href"),
            new_href = current_href;

        if (current_href.includes("7.8") == true) {
            new_href = new_href.replace(/7.8/g, "7.11");
            $(this).attr("data-cke-saved-href", new_href); //replace the old href with the new one
        }
        if (current_href.includes("7.9") == true) {
            new_href = new_href.replace(/7.9/g, "7.11");
            $(this).attr("data-cke-saved-href", new_href);
        }
        if (current_href.includes("7.10") == true) {
            new_href = new_href.replace(/7.10/g, "7.11");
            $(this).attr("data-cke-saved-href", new_href);
        }
    });
};
```

6. Just like earlier, we'll need to add a button or link:

```
var page_path = string.split(page.path,"/");
var page_1 = page_path[1];
var page_2 = page_path[2];
var console_check = false;

if(page_1 == "Spok_Console_Suite"){
    if (page_2 == "Spok_Console_Suite_Version_7.10" || page_2 == "Spok_Console_Suite_Version_7.9" ||
page_2 == "Spok_Console_Suite_Version_7.8") {
        return;
    }
    else {
        <style type="text/css">"a[href*=\'7.10\'], a[href*=\'7.9\'], a[href*=\'7.8\'] {background:
hotpink}"</style>
        <p><button id="pinkLinks">"Show outdated links"</button></p>
        <p><button id="fixPinkLinks">"Fix links"</button></p> //new button
        var console_check = true;
    }
}
```

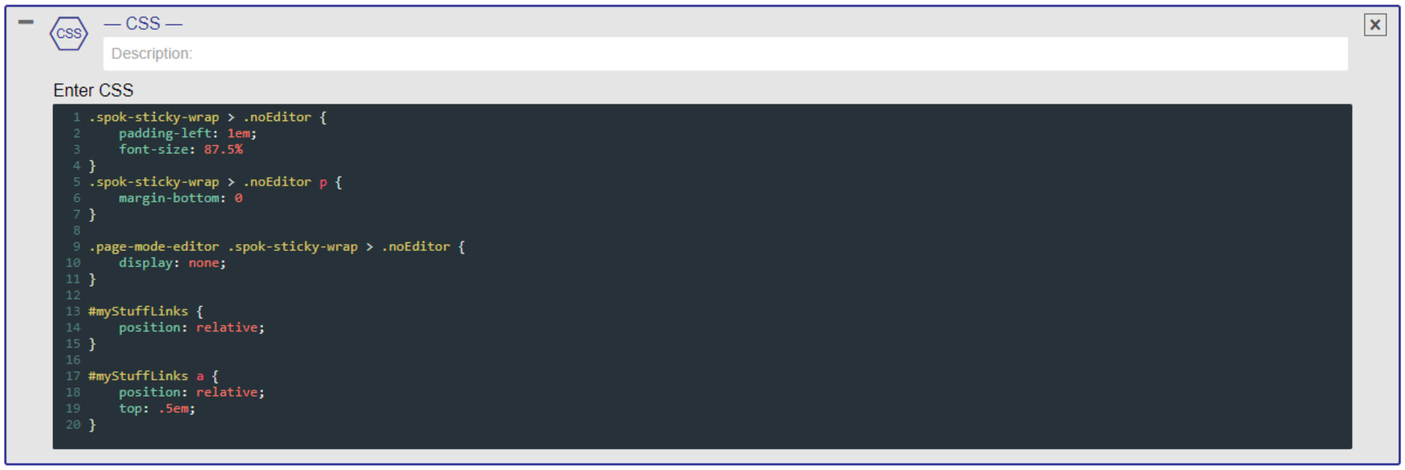
7. Finally, add an event handler for that button:

```
$(function() {
    $('#pinkLinks').on("click", hotPinkLinks);
    $('#fixPinkLinks').on("click", fixPinkLinks);  //new event handler
});
```

A shorter (but not as transparent) way to write the above function:

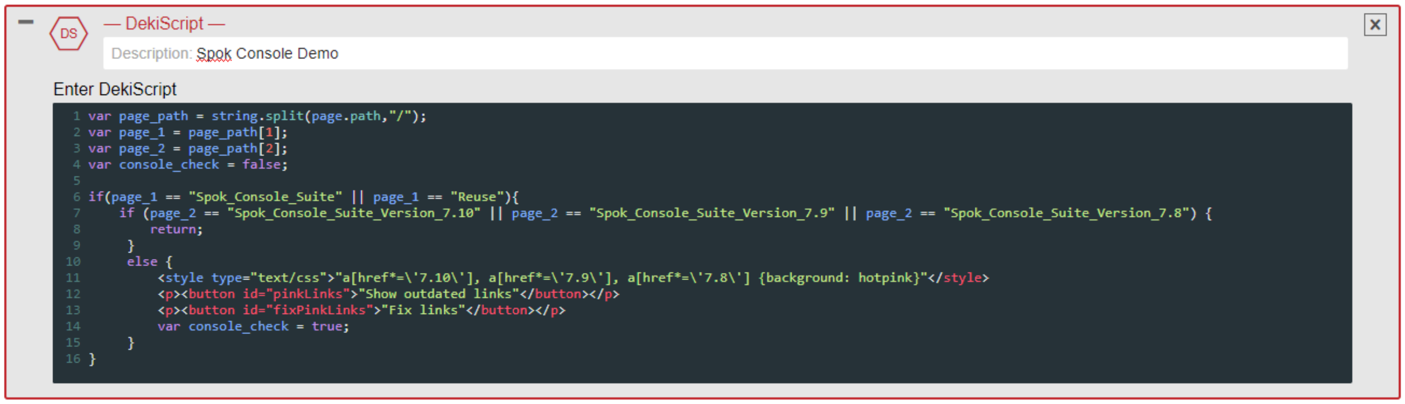
```
function fixPinkLinks() {
    var iframe_content = $("iframe.cke_wysiwyg_frame.cke_reset").contents();

    iframe_content.find("a").each(function(){
        var current_href = $(this).attr("href"),
            new_href = current_href.replace(/7.8|7.9|7.10/g, "7.11");

        $(this).attr("data-cke-saved-href", new_href);
     });
};
```

All together, it will look like this:



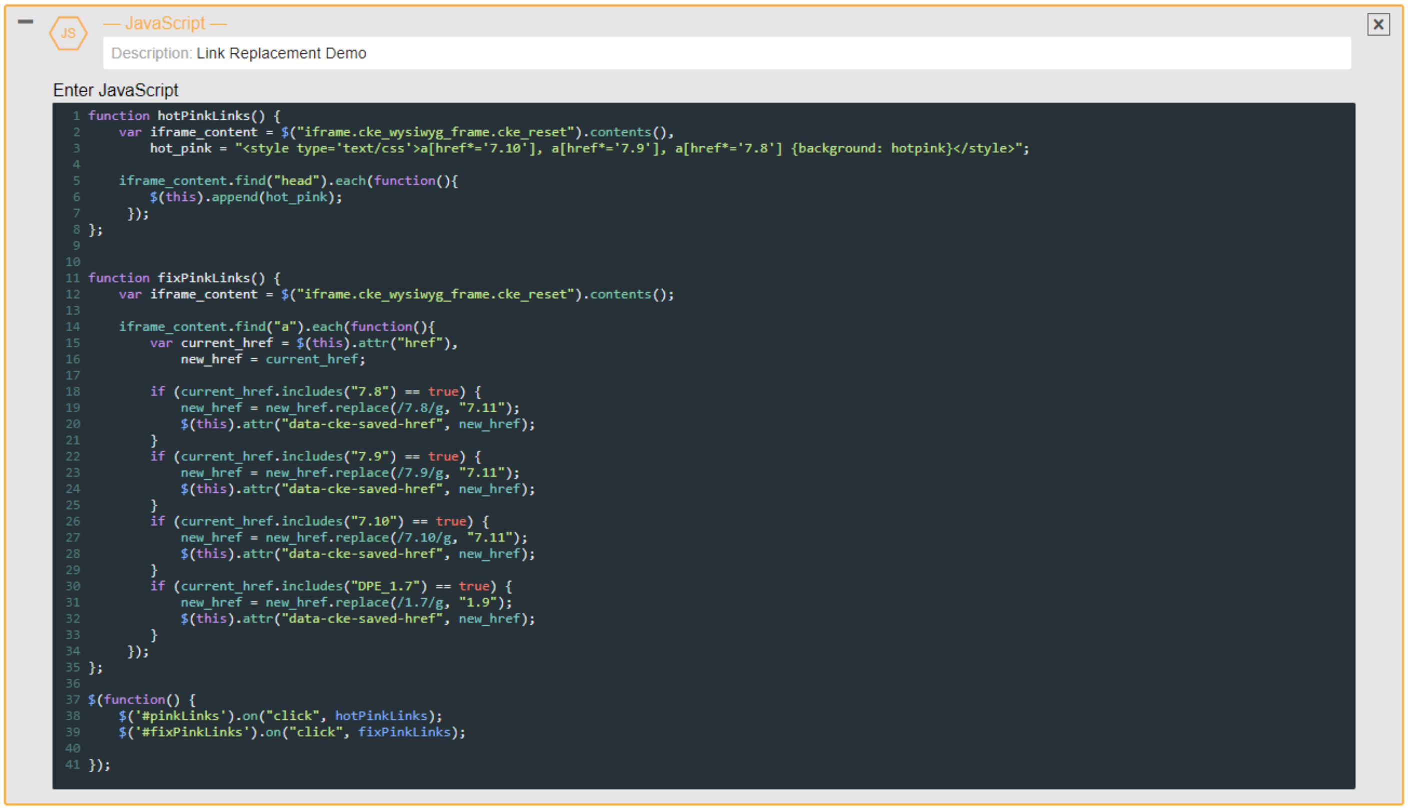
Enter CSS

```css
.spok-sticky-wrap > .noEditor {
    padding-left: 1em;
    font-size: 87.5%
}
.spok-sticky-wrap > .noEditor p {
    margin-bottom: 0
}

.page-mode-editor .spok-sticky-wrap > .noEditor {
    display: none;
}

#myStuffLinks {
    position: relative;
}

#myStuffLinks a {
    position: relative;
    top: .5em;
}
```



Enter DekiScript

```
var page_path = string.split(page.path,"/");
var page_1 = page_path[1];
var page_2 = page_path[2];
var console_check = false;

if(page_1 == "Spok_Console_Suite" || page_1 == "Reuse"){
    if (page_2 == "Spok_Console_Suite_Version_7.10" || page_2 == "Spok_Console_Suite_Version_7.9" || page_2 == "Spok_Console_Suite_Version_7.8") {
        return;
     }
     else {
        <style type="text/css">"a[href*=\'7.10\'], a[href*=\'7.9\'], a[href*=\'7.8\'] {background: hotpink}"</style>
        <p><button id="pinkLinks">"Show outdated links"</button></p>
        <p><button id="fixPinkLinks">"Fix links"</button></p>
        var console_check = true;
     }
}
```



Enter JavaScript

```javascript
function hotPinkLinks() {
    var iframe_content = $("iframe.cke_wysiwyg_frame.cke_reset").contents(),
        hot_pink = "<style type='text/css'>a[href*='7.10'], a[href*='7.9'], a[href*='7.8'] {background: hotpink}</style>";

    iframe_content.find("head").each(function(){
        $(this).append(hot_pink);
     });
};


function fixPinkLinks() {
    var iframe_content = $("iframe.cke_wysiwyg_frame.cke_reset").contents();

    iframe_content.find("a").each(function(){
        var current_href = $(this).attr("href"),
            new_href = current_href;

        if (current_href.includes("7.8") == true) {
            new_href = new_href.replace(/7.8/g, "7.11");
            $(this).attr("data-cke-saved-href", new_href);
        }
        if (current_href.includes("7.9") == true) {
            new_href = new_href.replace(/7.9/g, "7.11");
            $(this).attr("data-cke-saved-href", new_href);
        }
        if (current_href.includes("7.10") == true) {
            new_href = new_href.replace(/7.10/g, "7.11");
            $(this).attr("data-cke-saved-href", new_href);
        }
        if (current_href.includes("DPE_1.7") == true) {
            new_href = new_href.replace(/1.7/g, "1.9");
            $(this).attr("data-cke-saved-href", new_href);
        }
     });
};

$(function() {
    $('#pinkLinks').on("click", hotPinkLinks);
    $('#fixPinkLinks').on("click", fixPinkLinks);

});
```

Article type: Topic **Tags:** This page has no tags.