Use Widget 2.0 on Single Page Applications

Last Modified on 04/03/2024 12:47 pm EDT

The Problem

If you've set up "Recommend on Pages" for contextual help, by default, when a reader navigates to a new page, the widget grabs that URL and uses it to suggest content in the Recommended tab. This method works fine for applications that use full page reloads during navigation.

However, for **single page applications** this is not the case. In **single page applications**, the reader might have navigated to an entirely new area of your application, and while the URL may reflect that, the page never fully reloads. Thus, neither does the widget. This can result in confusion when the reader opens the widget, since the suggested articles may relate to the first page they saw instead of the page they're currently looking at.

The Solution

To solve this problem, you can call a widget method to proactively tell the widget that the current URL has changed:

```
//method to update the current URL used for contextual help
_ko19.updateRecommended('currentUrl');
```

> If you're using this method repeatedly in a Single Page Application, we recommend calling _ko19.reset(); before updateRecommended to clear out whatever's been previously loaded in the widget.

Example Usage

```
var _ko19 = _ko19 || {};
_ko19.__pc = 'xxxxxxxxxxx-xxxxxxxxxxxxx';
_ko19.base_url = '//mykb.knowledgeowl.com';

!function() {
 var ko = document.createElement('script');
 ko.type = 'text/javascript';
 ko.async = true;

 ko.src = _ko19.base_url + '/widget/load';
 document.head.appendChild(ko);

 // Called when the widget is toggled open
 _ko19.onOpen = function () {
  var currentPage = 'some-page';
  _ko19.updateRecommended(currentPage);
 }
}();
```

Other Reasons for Using This Method

Even if your application is not a single page application, your app's URLs may be constructed in a way that make it very difficult to utilize the "Recommend on Pages" feature.

Another benefit of the _ko19.updateRecommended function is that it can accept any string value as the "currentUrl" parameter. This allows you to use a single word or any other format you wish in place of the actual URL. Take a look at the below examples:

Complex URLs to Simple One Word Location

Let's say that you have an application URL that looks like this:

```
https://myapp.com/app/users/edit-user/abcd123354/id/123151ajsdfa?p=12sdfsd12&q=12154aba
```

By default, the following is what gets sent as the current location to the widget:

```
/app/users/edit-user/abcd123354/id/123151ajsdfa
```

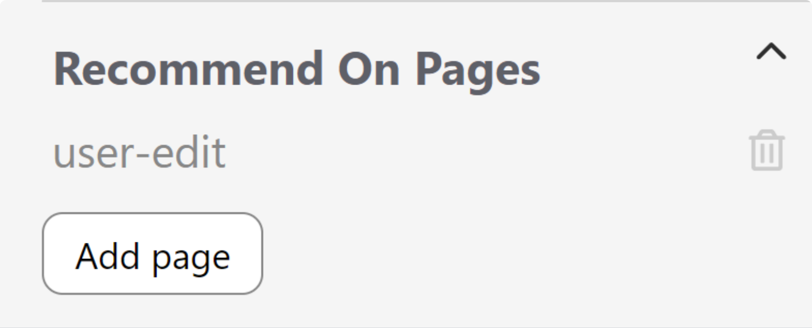
If we leave this as is, we would need to use regex to make sure that the ID portions of the URL get genericized so that we wouldn't need to add a "Recommend On Pages" URL for every single ID possible. So what we'd need to add to articles in the Recommend On Pages section would look some thing like this:

```
/app/users/edit-user/{custom}/id/{custom}
```

While that will work, it is still a fairly complex way to essentially say that we want certain articles to appear when people are looking at the "edit-user" screen. By using the "**updateRecommended**" method, we can dramatically simplify this like so:

```
//define current page and send to widget
var currentUrl = 'user-edit';
_ko19.updateRecommended(currentUrl);
```

Once we've done that, we can tell our KnowledgeOwl authors that if they want a certain article to appear as recommended on the "user-edit" pages in the widget, they can add the string "user-edit" to the "Recommend On Pages" text box like so:



Sample Recommend On Page path

Much simpler than adding `/app/users/edit-user/{custom}/id/{custom}` !

Dealing with More Complex Page Paths

You might find that you need to be more specific with which page a reader is looking at. Let's look at the following URLs for example:

`https://myapp.com/app/users/user/id/123123abasd/reports/data-usage`
`https://myapp.com/app/account/id/asdfksa1233123/reports/data-usage`

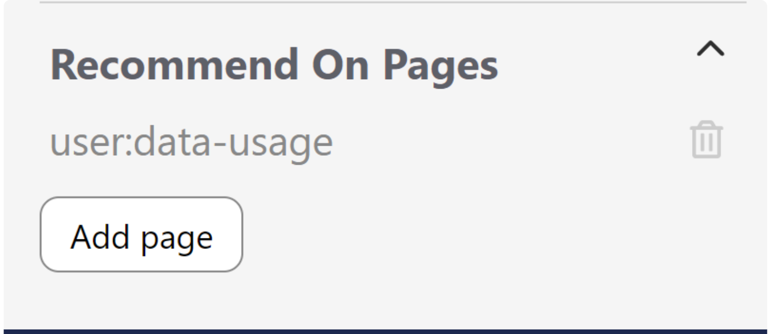
Both of these URLs are referencing a page that shows a report of "data-usage". However, one of the URLs is referencing the data usage for a single user while the other is referencing the overall data usage for the entire account. If we were to only pass over "data-usage" as the current page, this could cause confusion with the content that gets recommended within the widget.

Because the "**updatePageLoc**" function is flexible and accepts any string value, we can handle this in whatever way will make the most sense to our KO authors. For example, we could send the current page location like so:

```
//Page: https://myapp.com/app/users/user/id/123123abasd/reports/data-usage
var currentUrl = 'user:data-usage';
_ko19.updateRecommended(currentUrl);

//Page: https://myapp.com/app/account/id/asdfksa1233123/reports/data-usage
var currentUrl = 'account:data-usage';
_ko19.updateRecommended(currentUrl);
```

We can then tell our KnowledgeOwl authors to make sure they specify the **section** of the app followed by a ":" followed by the **page** of the app when using the "Recommend On Pages" text area:



Example page path using user:data-usage

This flexibility allows you to come up with a naming convention that will make sense to both the developers that install the widget and the people who are actually writing the documentation.

Example Usage

Extremely simple example HTML page for `https://myapp.com/app/users/user/id/123123abasd/reports/data-usage` where the "**currentUrl**" value is passed to the view via the controller or it is populated in some other way dynamically:

```
<!DOCTYPE html>
<html>
 <head>
  <title>Data Usage Report for John Smith</title>
 </head>
 <body>
  <!-- is passed to view via the controller or otherwise populated dynamically -->
  <input type="hidden" id="currentUrl" value="">
 </body>
</html>
```

The below code uses the value of the HTML element "#currentUrl" that we populated above:

```
!function() {
 var ko = document.createElement('script');
 ko.type = 'text/javascript';
 ko.async = true;

 ko.src = _ko19.base_url + '/widget/load';
 document.head.appendChild(ko);

 // Called when the widget is toggled open
 _ko19.onOpen = function () {
  var currentPage = document.getElementById('currentUrl').value;
  _ko19.updateRecommended(currentPage);
 }
}();
```