Cowhite.js Documentation

Cowhite Software Pvt Ltd

May 10, 2018

Contents:

1 Overview **1**
- 1.1 Requirements . . . 1
- 1.2 Features . . . 1

2 Installation **3**

3 Usage **5**
- 3.1 1. Single page application: Basic usage . . . 5
- 3.2 2. Load and insert content automatically from remote url . . . 6
- 3.3 3. Load content automatically from remote url and then manually render it . . . 6
- 3.4 4. Load data manually . . . 6
- 3.5 5. Form submissions . . . 7
- 3.6 6. Button actions or button clicks (delete for example) . . . 8

4 Events **11**
- 4.1 tabname-opened . . . 11

5 Indices and tables **13**

CHAPTER 1

Overview

Cowhite.js allows to build single page applications easily using jquery as dependency.

1.1 Requirements

Jquery

1.2 Features

- Build single page applications easily
- Submit forms easily
- Button actions (delete easily without writing much code)

CHAPTER 2

Installation

- Download jquery and add it above all javascript files embedded in the html page
- Download cowhite.js(https://github.com/cowhite/Cowhite.js/blob/master/cowhite.js) and add it below jquery.

CHAPTER 3

Usage

We can use cowhite.js to build single page applications easily. Here are the various scenarios:

Scenarios:

3.1 1. Single page application: Basic usage

Add below html for each tab(or url in single page app) in the main page

Html:

```
<div class='items-section' class='each-section hide' data-url="https://marketing.
↪cowhite.com/api/v1/items/">
    <div id='items-content'></div>
</div>
```

Url will be like:

```
https://cowhite.com/#items
```

If you want #items to be #learn for example, then code will be like:

```
<div class='learn-section' class='each-section hide' data-url="https://marketing.
↪cowhite.com/api/v1/items/">
    <div id='learn-content'></div>
</div>
```

Url will be like:

```
https://cowhite.com/#learn
```

After opening above url, the content present inside “learn-section” will be shown.

3.2 2. Load and insert content automatically from remote url

Html:

```
<div class='items-section' class='each-section hide' data-url="https://marketing.
↪cowhite.com/api/v1/items/">
    <div id='items-content'></div>
</div>
```

You can load content automatically from remote url by providing data attribute for url, that is "data-url" for the element with class "each-section".

This response will be inserted into the element with id "items-content" where items is the tab.

3.3 3. Load content automatically from remote url and then manually render it

Html:

```
<div class='items-section' class='each-section hide' data-callback="loadItemsCallback
↪" data-url="https://marketing.cowhite.com/api/v1/items/">
    <div id='items-content'></div>
</div>
```

Since you gave both data-url and data-callback, you need to implement the callback like below:

```
function loadItemsCallback(res){
  var html = "", $itemHtml = $("#item-template"), itemHtml;
  for(var i=0; i<res.results.length; i++){
    itemHtml = $itemHtml.html().replace(/ITEM_ID/g, res.results[i].id).replace(
      /ITEM/g, res.results[i].name);
    html += itemHtml;
  }
  if(!res.results.length){
    html += "<div class='big-msg'>No items yet.</div>"
  }
  $("#items-content").html(html);

}
```

In my case, I implemented this because, the server response is JSON but not direct html for this api. So, I need to implement so I can generate html from JSON response. But in most other cases, I generally return html from the server, so I can load the html directly.

3.4 4. Load data manually

Html:

```
<div class='items-section' class='each-section hide'>
    <div id='items-content'></div>
</div>
```

In this scenario, we didnt provide any data-url, so it wont try to fetch automatically from the server.

You need to manually catch the event "items-opened" and write necessary code inside that function:

```
$(document).on("items-opened", function(){
    // Implement code to make ajax calls and then to insert that content
});
```

3.5 5. Form submissions

Add a button like this:

```
    <a href="javascript:;" class='btn-show-form btn-show-submit-form'
data-url="assets/data/new_item_form.html" data-submit-url="http://marketing.cowhite.
↪com/api/v1/items/"
data-form-id="add-item-form" data-aftersuccess='refreshItems'>Add Item</a>
```

Note that this link has two classes:

```
btn-show-form
btn-show-submit-form
```

The class "btn-show-submit-form" is common for two buttons:

```
1. Button that shows the form.
2. Button that submits the form
```

For button that shows the form, it has the additional class:

```
btn-show-form
```

For button that submits the form or submit button, it has the additional class:

```
btn-submit-form
```

And add a bootstrap modal like this for showing the form:

```
<div class="form-modal modal fade" tabindex="-1" role="dialog">
  <div class="modal-dialog" role="document">
    <div class="modal-content">
      <div class="modal-header">
        <button type="button" class="close" data-dismiss="modal" aria-label="Close">
↪<span aria-hidden="true">&times;</span></button>
        <h4 class="modal-title"></h4>
      </div>
      <div class="modal-body">
        <!-- form will be added here automatically -->
      </div>
      <div class="modal-footer">
        <button type="button" class="btn btn-default" data-dismiss="modal">Close</
↪button>
        <a href="javascript:;" class="btn btn-primary btn-show-submit-form btn-submit-
↪form" data-type="POST">Submit</a>
        <!-- Note that this button has the classes btn-show-submit-form and btn-
↪submit-form -->
      </div>
```

(continues on next page)

(continued from previous page)

```
    </div><!-- /.modal-content -->
  </div><!-- /.modal-dialog -->
</div><!-- /.modal -->
```

This modal needs the class:

```
form-modal
```

This modal/popup has class "form-modal" and the submit button needs the classes "btn-show-submit-form btn-submit-form" and needs data attribute "type" as "POST".

After clicking the link, it will open above modal with the form html fetched from the server using the data attribute "url", that is, "data-url".

When you click "submit" shown in the modal, it will send request(POST or GET depending on data attribute 'type', i.e data-type in the submit button) to the url mentioned in data-submit-url. If both data-url and data-submit-url are same, then you can simply ignore giving data-submit-url so it will directly use data-url for both getting the form html as well as submitting the form.

You can use this form to either add a new database record or edit an existing database record, but you need to set proper url in data-url so respective forms will be rendered in the modal.

This is explained using "Add item" link at the top of this page - http://cowhitejs.cowhite.com/

Data attributes for add form button:

url:

```
The url for fetching the form to be rendered in the modal
```

submit-url:

```
The url for submitting the form
```

afterSuccess:

```
Callback that will be executed after ajax success.
```

form-id:

```
This is optional and it will be added as html id to the form being added.
```

Data attributes for the submit button:

type:

```
GET or POST. GET is default type.
```

3.6 6. Button actions or button clicks (delete for example)

Add this anchor in html:

```
  <a href="javascript:;"  class='btn btn-danger btn-action btn-action-delete'
data-url="http://marketing.cowhite.com/api/v1/items/ITEM_ID/" data-request-type=
↪'DELETE'
data-aftersuccess='refreshItems'>Delete</a>
```

This anchor needs the classes “btn-action btn-action-delete”, data attribute url, i.e data-url, data-request-type(DELETE if you are sending delete request but it might change depending on the server).

If your server expects a POST request then data-request-type should be “POST”. But it needs the class “btn-action-delete” so it will ask for confirmation before making the ajax request.

CHAPTER 4

Events

4.1 tabname-opened

Whenever a tab/url is switched, an event will be fired. If the tab name is “learn”, then fired event is “learn-opened”

You can catch the event and then write necessary code:

```
$(document).on("tabname-opened", function(){
    // Implement code here
})
```

CHAPTER 5

Indices and tables

- genindex
- modindex
- search