Version: 4.2.2

Static Resources

Article 08/28/2023

Some JavaScript and CSS files must be included for enabling the dashboard component visualization. Exists an HTML extension method from the namespace `InfoArch.Web.Mvc.UI` to import the static files mentioned above. It is recommended to invoke these methods within the **<head>** tag in the view where it will be used.

Scripts

Attaching Scripts

dReveal uses the `GetScriptsDashboard` method to attach the scripts required for the Dashboard component. It receives an instance of the `ScriptDashbard` class as a parameter, where its property **ExtensionSuiteDash** requires an **ExtensionSuite.Dashboard** value from the namespace `InfoArch.Web.Mvc.Enums`.

The following example demonstrates how to attach the dReveal dashboard component scripts to the view:

```
<head>
    @using InfoArch.Web.Mvc.UI
    @using InfoArch.Web.Mvc.Enums

    @Html.InfoArch().GetScriptsDashboard(
        new ScriptDashbard { ExtensionSuiteDash = ExtensionSuite.Dashboard
}
    )
</head>
```

Dashboard: Grid Chart - Label Characters Length

As a dReveal user, I want to **focus on the charts** themselves. When the labels on the axis of the charts are too long, the content is truncated.

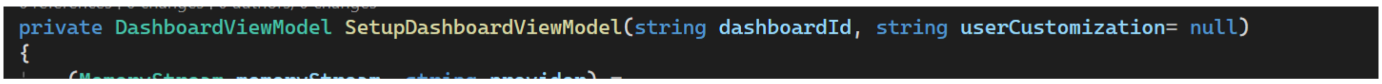
The maximum number of characters supported by default is 50.

To modify the number of displayed characters, use the **SetMaxCharsLength()** function from the **DashboardViewModel** class.

To disable this behavior, use the value "0".

Below is an example (highlighted line) showing how a Client Controller function uses this functionality to limit the display of labels to a maximum of 50 characters.

Note: This property applies to both the Argument Axis and Legends.



```
private DashboardViewModel SetupDashboardViewModel(string dashboardId, string userCustomization= null)
{
```

StyleSheet

The `GetStyleSheetsDashboard` method is used to attach the necessary stylesheet files of the desired dReveal Dashboard component theme. It receives an instance of the **StyleSheetDashboard** class as a parameter, where its property **Theme** can receive a string object with the value **"light"** or **"dark"**, and another property **ExtensionTypeDash** that requires an **ExtensionType.Dashboard** value from the namespace `InfoArch.Web.Mvc.Enums`.

The following code attaches dReveal dashboard stylesheets to provide the default appearance.

```
<head>
    @using InfoArch.Web.Mvc.UI
    @using InfoArch.Web.Mvc.Enums

    @Html.InfoArch().GetStyleSheetsDashboard(
            new StyleSheetDashbard { Theme = "light", ExtensionTypeDash =
ExtensionType.Dashboard }
    )
</head>
```

Third-Party Libraries

dReveal components has third party libraries embedded within needed for its correct work and provide a method to exclude them if the hosting application decide to handle them. The method mentioned above is called `RegisterRequiredThirdPartyLibraries`. It has two parameters, in the first one, for dashboard component purpose is required to use **RegisterThirdPartyType.Dashboard**, that it is obtained from the namespace `InfoArch.Web.Mvc.Enums`, the next one is used to specify the third-party libraries to exclude. Some third-party libraries enable to exclude are **JQuery** and **JQueryUI**.

```
<head>
    @using InfoArch.Web.Mvc.UI
    @using InfoArch.Web.Mvc.Enums

    @Html.InfoArch().RegisterRequiredThirdPartyLibraries(
        RegisterThirdPartyType.Dashboard,
        RegisterThirdPartyElements.JQuery |
RegisterThirdPartyElements.ThemeDreamweaver
    )
</head>
```

> **NOTE**
>
> If the hosting application decide to handle **JQuery**, this should be referenced before the HTML extension methods provided by dReveal.