Version: 4.2.2

Customization

Article 08/28/2023

The dReveal grid component provides a feature to get or load a specific state of it

The grid state is a string object that could be stored in a binary file, database, or any storage mechanism.

> The scope of the grid elements that can be stored in the state are filters applied and modifications with the edit mode feature.

Grid State Methods

dReveal grid component provides throughout the `GridViewModel` class, methods to get and load the grid state.

Get Customization

The `GridViewModel` class provides a method to get the grid customization (grid state), called `Serialize`. For further information about serialize method, see the Grid View Model documentation.

The following code demonstrate how implement this method:

```
1  [HttpPost]
2  public ActionResult
   GridReport([ModelBinder(typeof(DRevealDataStructureBinder))]
   DRevealDataStructure gridState)
3  {
4      GridViewModel viewModel = new GridViewModel(
5          ...
6           // Set controller/action to use in javascript function
7           saveGridControllerAction: $"
   {RouteData.Values["Controller"]}/SaveState",
8          ...
```

```
    );

    viewModel.SetState(gridState);
    viewModel.LoadInitialConfiguration();

    return View("GridReport", viewModel);
}

public ActionResult
SaveState([ModelBinder(typeof(DRevealDataStructureBinder))]
DRevealDataStructure gridState)
{
    // create GridViewModel from DRevealDataStructure
    GridViewModel viewModel =
GridViewModel.GetCustomizationViewModel(gridState);

    // Serialize
    string state = viewModel.Serialize();
    // save using the storage mechanism choosen.
    ....

    return new EmptyResult();
}
```

In the client side is required to add the following method for state saving asynchronously:

```
saveStateButton.addEventListener("click", () => {
    dRevealGrid.SaveState()
        .done(function() {
            alert("Grid saved");
        })
        .fail(function () {
            alert("Error");
        });
})
```

> **NOTE**
>
> This function performs an Ajax request to the controller/Action set in `GridViewModel.saveGridControllerAction`.

Load Customization

The `GridViewModel` class provides a method to load the grid customization (grid state), called `AssignUserCustomizationDefinitionModel()` and after applying the `LoadInitialConfiguration()`, the `UpdateCustomization()` method is required. For further information about these methods, see the Grid View Model documentation.

The load customization process can load a previous version of a grid state once the GridViewModel will be configured. The following code demonstrates how implement this method:

```
[HttpPost]
public ActionResult GridReport()
{
    GridViewModel viewModel = new GridViewModel(
        memoryStream: fileStream,
        connectionString: "Data Source=MyServer; Initial Catalog=myDatabase; User ID=myUser; Password=myPassword",
        gridId: "dasboard-id",
        htmlFormId: "dasboard-from",
        fullGridControllerActionPath: HttpContext.Request.Url.AbsolutePath,
        useNewExportToImagePro: true,
        filterComponentParentId: "filter-component",
        drillThroughControllerAction: string.Empty,
        // Set controller/action to use in javascript function
        saveGridControllerAction: $"{RouteData.Values["Controller"]}/SaveState",
        excelXlsxExportControllerAction: string.Empty
    );

    // Set customization
    viewModel.AssignUserCustomizationDefinitionModel(gridState);
    viewModel.LoadInitialConfiguration();

    // Apply customization
    viewModel.UpdateCustomization();

    return View("GridReport", viewModel);
}
```