The rdg expects its data in json format that follows:

{ Items : [{}, ...{}], NOfItems = 0,NOfPages = 0, CurrentPage = 0}

When one deals with the reactives in ASP.Net MVC she deals with 2 of them. The first one is primary

Controller. Your primary ASP.NET MVC controller should contain the Index action which respond with the View on which Rdg is located. It is its single goal.

After that rdg loads it data from the data controller. F.e. it can be something like ItemsConroller.



So, data controller action should return the viewModel of ReactDataGridViewModel type.

public class ReactDataGridViewModel<TDataItemModel>

{

public IEnumerable<TDataItemModel> Items { get; set; }

public int NOfItems { get; set; }

public int NOfPages { get; set; }

public int CurrentPage { get; set; }

}

For example, you can declare SalesGridViewModel: ReactDataGridViewModel<SalesModel>,

Where SalesModel is

Public class SalesModel {

Public int Id;

Public DateTime SalesDate;

…

}

Design

The spinner container has min-height = height of spinner image to show the spinner on initial loading (when no data displayed)

Rendering

The rendering is based on the principle of “minimalism”. It means that grid render function provides with the base (minimum) data grid DOM, i.e. table markup which consists of <thead> and <tbody> parts only. The detailed content is rendered by means of template functions – header and row templates. These template function are out of the grid component itself. They are developed by the data grid user.

Thus flexibility is provided.

Testing:

1. Karma starts. It launches the webpack as a preprocessor for the tests.webpack.js
2. Webpack processes each –test.js file from /src directory