Sage 300 Web Screens SDK

Generating JavaScript for Grids

August 2022

The MIT License (MIT)

Copyright © 2022 The Sage Group plc or its licensors. All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Contents

[1. Grid JS Generator Solution 4](#_Toc440896561)

[2. Select Model 5](#_Toc440896562)

[3. Select Columns 6](#_Toc440896563)

[4. Select Order 7](#_Toc440896564)

[5. Set Variables 9](#_Toc440896565)

[6. Results 11](#_Toc440896566)

[6.1 JavaScript 11](#_Toc440896567)

[6.2 CSHTML 11](#_Toc440896568)

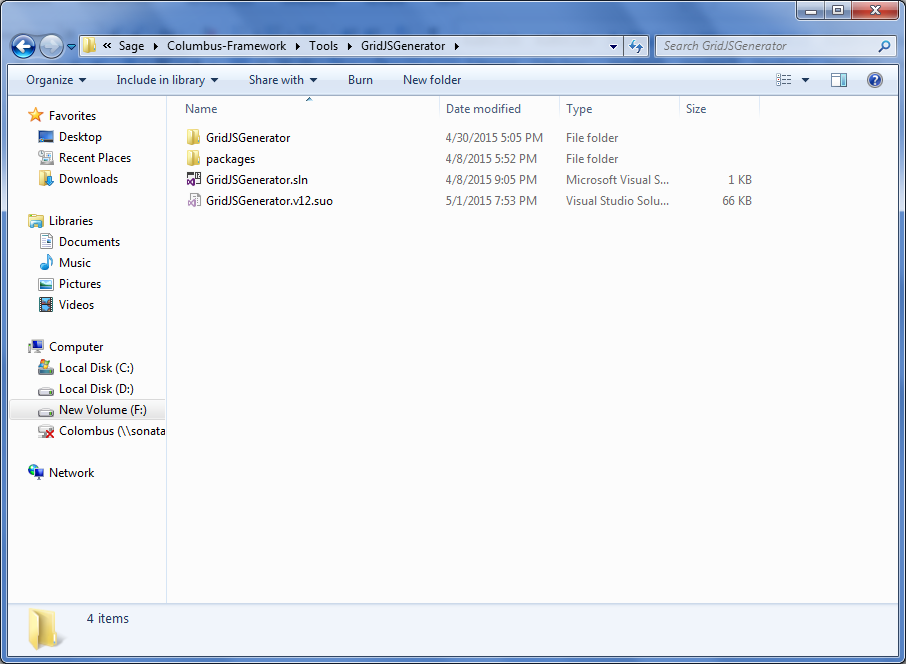
1. Grid JS Generator Solution

Writing JavaScript for grids is a time-consuming process.

The purpose of this utility is to reduce this time and generate “boilerplate” code for grids that will work as-is for non-editable grids and will require only logic changes for editable grids.

To open the utility, open and Run the Grid JS Generator solution (GirdJSGenerator.sln).

The path is ~\Columbus-Framework\Tools\GridJSGenerator



1. Select Model

* Model

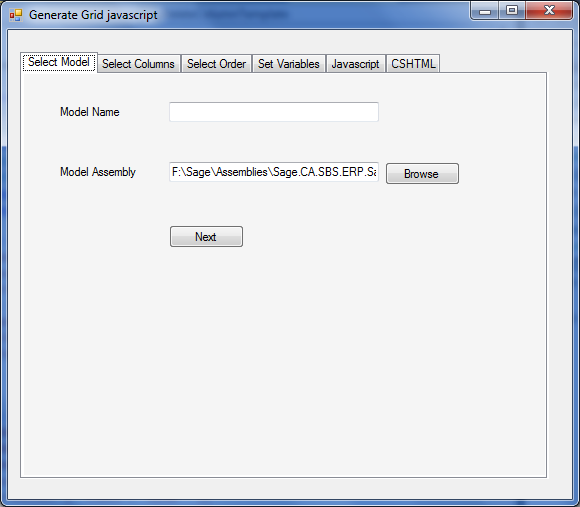
Every grid is associated with a model. Without generating the model for the grid, you will not be able to use this utility. Enter the name (exact) of the model that will be displayed as a grid.

* Path

The utility uses reflection to get all the properties of the model. To do this, the utility needs to know the path in which the DLL of the model is available. This will be the path to the Assembly folder in your local machine.

**Note:** After you use the Class Generation utility to generate the models, make sure the project is rebuilt. If it is not, the DLL will not be available in the Assembly folder.

Click Next to continue.



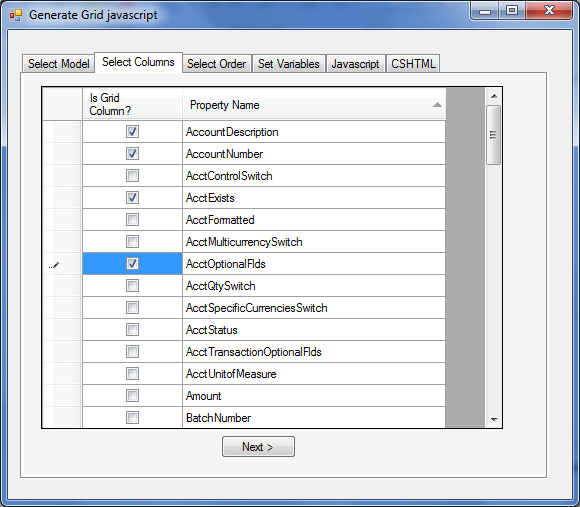
1. Select Columns

Every property in the model will correspond to a column in the grid, but not all properties may need to be displayed in the grid. So the next step is to select these columns.

With the help of reflection, the utility lists out all the properties of the model with a text box alongside each. Find and select the properties that need to be displayed in the grid.

**Note:** The property name column is sortable for easier search

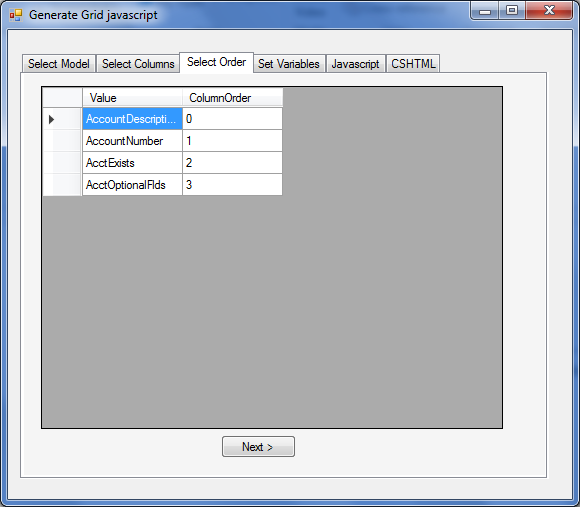
Click Next to continue.



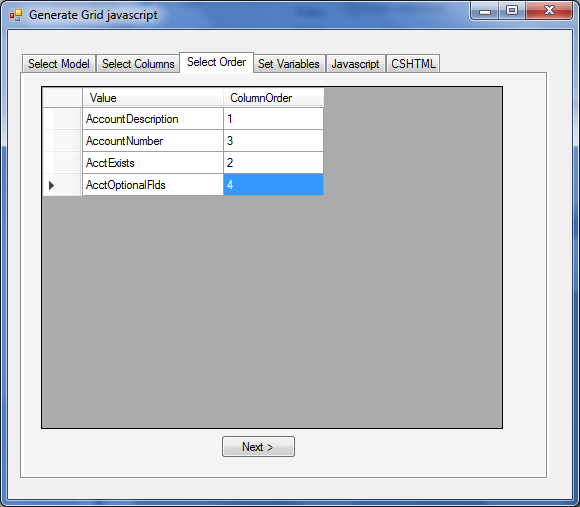
1. Select Order

The order in which the columns are to be displayed in the grid may be different from the order selected on the Select Columns screen.

A default column order is set by the utility and can be changed.



In the following example, AccountDescription should be the first column, followed by AcctExists, then AccountNumber and finally AcctOptionalFields.



Click Next to continue.

1. Set Variables

* Grid ID

The ID of the grid as set in CSHTML.

Example: @Html.KoKendoGrid("JournalBatchDetailsGrid", "journalBatchUI.ModelData.Data.JournalEntry.JournalDetail.Items", "TransactionNumber", "journalBatchUIGrid.entryDetailsMultiConfig", true)

* Namespace

The namespace is auto generated by the utility and is open to be changed. The generated namespace is a suggestion and is not final.

* JS View Model

The JSON View model to which the grid will be bound to on the client side.

Example: journalBatchUI.ModelData.Data

* Module

From the list, select AP, AR, AS, CS, GL, IC, OE, PO, Core or Shared.

* Controller

The name of the Controller to which the grid will connect to get the data.

* Action

The name of the Action method to which the grid will connect to, to get the data

* Has User Preference

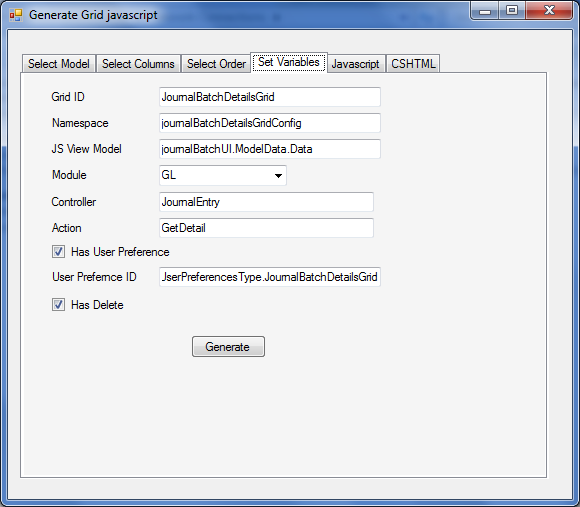
Select or clear this check box as needed. As a general rule, for grids with fewer than 4 columns, user preference is not needed. If the column count is fewer than 4, this option is cleared by default, but you can select it.

* User Preference ID

Every user preference needs an ID. This is generated by the utility by default. Make sure you add this at the common JavaScript file.

* Has Delete

If the grid has delete functionality, select this option.



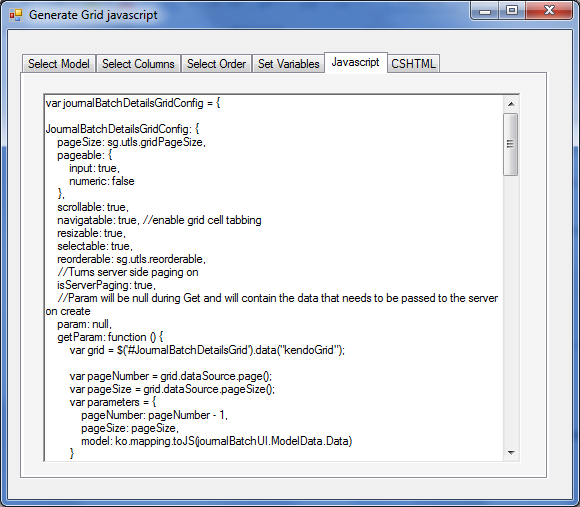
1. Results

The results are displayed in two different tabs

* 1. JavaScript

This contains the JavaScript code that is needed to display the records on the screen, provided the controller has been wired up to bring the data.

Copy and paste the text into the Behaviour.js file of the respective page.



* 1. CSHTML

This contains the CSHTML code that is needed for localization and customization of the grid.

Copy and paste the text into the corresponding .cshtml file of the grid.

