0 - 官网

<https://github.com/ag-grid/ag-grid>

1 - 学习

<https://www.ag-grid.com/angular-grid/>

yarn add ag-grid-community ag-grid-angular

import { AgGridModule } from 'ag-grid-angular';

@NgModule({

imports: [

AgGridModule.withComponents([])

],

})

export class AppModule {}

下一步是添加ag-Grid样式-

C:\yutao\software\frontend-work\angular.json：

"node\_modules/ag-grid-community/dist/styles/ag-grid.css",

"node\_modules/ag-grid-community/dist/styles/ag-theme-alpine.css",

网格有几个不同的主题。选择一个与您的项目设计相匹配的项目。您可以使用Sass变量进一步自定义它，我们将在以后的文章中进一步介绍该技术。

export class AppComponent {

title = 'my-app';

columnDefs = [

{ field: 'make' },

{ field: 'model' },

{ field: 'price'}

];

rowData = [

{ make: 'Toyota', model: 'Celica', price: 35000 },

{ make: 'Ford', model: 'Mondeo', price: 32000 },

{ make: 'Porsche', model: 'Boxter', price: 72000 }

];

}

网格的两个基本配置属性-[列定义](https://www.ag-grid.com/javascript-grid-column-definitions/)（columnDefs）和数据（rowData）

<ag-grid-angular

style="width: 500px; height: 500px;"

class="ag-theme-alpine"

[rowData]="rowData"

[columnDefs]="columnDefs"

>

</ag-grid-angular>

这是ag-grid组件定义，具有两个属性绑定-rowData和columnDefs。该组件还接受标准DOMstyle和class。我们将类设置为ag-theme-alpine，它定义了网格主题。您可能已经注意到，CSS类与我们之前导入的CSS文件的名称匹配。

启用排序和过滤

columnDefs = [

{ field: 'make', sortable: true, filter: true },

{ field: 'model', sortable: true, filter: true },

{ field: 'price', sortable: true, filter: true }

];

获取远程数据

export class AppComponent implements OnInit {

columnDefs = [

{ field: 'make', sortable: true, filter: true },

{ field: 'model', sortable: true, filter: true },

{ field: 'price', sortable: true, filter: true }

];

}

上面的代码将srowData从硬编码数组转换为Observable。为了使网格能够使用它，我们需要向该属性添加一个异步管道：

<ag-grid-angular

style="width: 500px; height: 500px;"

class="ag-theme-alpine"

[rowData]="rowData | async"

[columnDefs]="columnDefs"

>

</ag-grid-angular>

启用选择

，我们需要允许用户从网格中选择某些行，并将其标记为系统中已标记的行。

我们将标志切换状态和持久性留给后端团队。在我们这边，我们应该启用选择，然后获取选定的记录，并将它们与API调用一起传递给远程服务端点。

export class AppComponent implements OnInit {

columnDefs = [

{ field: 'make', sortable: true, filter: true, checkboxSelection: true },

{ field: 'model', sortable: true, filter: true },

{ field: 'price', sortable: true, filter: true }

];

}

接下来，启用多行选择，以便用户可以选择多行：

<ag-grid-angular

style="width: 500px; height: 500px;"

class="ag-theme-alpine"

[rowData]="rowData | async"

[columnDefs]="columnDefs"

rowSelection="multiple"

>

</ag-grid-angular>

通过不绑定属性值，我们在这里采取了一些捷径。如果不使用[]，则赋值会将属性值作为字符串传递，这对于我们的目的是很好的。

大！现在，第一列包含一个复选框，单击该复选框可以选择该行。我们唯一需要添加的就是一个按钮，它可以获取选定的数据并将其发送到服务器。为此，我们将使用 ag-Grid API-我们将通过组件实例对其进行访问。要访问组件实例，我们需要将#agGrid ID添加到我们的组件中。

<button (click)="getSelectedRows()">Get Selected Rows</button>

<ag-grid-angular

#agGrid

style="width: 500px; height: 500px;"

class="ag-theme-alpine"

[rowData]="rowData | async"

[columnDefs]="columnDefs"

rowSelection="multiple"

>

</ag-grid-angular>

现在，让实例在我们的组件中可访问：

export class AppComponent implements OnInit {

@ViewChild('agGrid') agGrid: AgGridAngular;

columnDefs = [

{ field: 'make', sortable: true, filter: true, checkboxSelection: true },

{ field: 'model', sortable: true, filter: true },

{ field: 'price', sortable: true, filter: true }

];}

我们唯一需要添加的就是getSelectedRows组件的方法。

export class AppComponent implements OnInit {

@ViewChild('agGrid') agGrid: AgGridAngular;

columnDefs = [

{ field: 'make', sortable: true, filter: true, checkboxSelection: true },

{ field: 'model', sortable: true, filter: true },

{ field: 'price', sortable: true, filter: true }

];

getSelectedRows() {

const selectedNodes = this.agGrid.api.getSelectedNodes();

const selectedData = selectedNodes.map(node => node.data );

const selectedDataStringPresentation = selectedData.map(node => node.make + ' ' + node.model).join(', ');

alert(`Selected nodes: ${selectedDataStringPresentation}`);

}

}

自定义主题外观

我们要做的最后一件事是通过修改主题的一些参数来更改网格的外观。

打开style.scss并添加一些参数：

@import "../node\_modules/ag-grid-community/src/styles/ag-grid.scss";

@import "../node\_modules/ag-grid-community/src/styles/ag-theme-alpine/sass/ag-theme-alpine-mixin.scss";

.ag-theme-alpine {

@include ag-theme-alpine((

odd-row-background-color: #CFD8DC

));

}

您可以在此处找到主题参数的完整列表。

如果一切都配置正确，则网格的第二行将变暗。恭喜你！现在，您现在知道可以随意调整网格外观了-还有几十个Sass变量，可用于控制字体系列和大小，边框颜色，标题背景颜色，甚至单元格和列中的间距。主题文档部分中提供了完整的主题参数列表。

组件

为了使ag-Grid能够使用您的Angular组件，您需要在顶层模块中提供它们：

@NgModule({

imports: [

AgGridModule.withComponents(

[

*SquareComponent*,

CubeComponent,

// ...other components

然后，您可以将这些组件用作编辑器，渲染器或过滤器。例如，要将Angular组件用作单元渲染器，您可以执行以下操作：

let colDefs = [

{

headerName: "Square Component",

field: "value",

cellRendererFramework: *SquareComponent*,

},

...other column definitions

]

子组件与父组件的沟通

有多种方法可以管理Angular中的组件通信（共享服务，局部变量等），但是您通常需要一种简单的方法来让“父”组件知道“子”组件上发生了某些事情。在这种情况下，最简单的方法是使用gridOptions.context持有对父项的引用，子项随后可以访问该引用。

// 在父组件中——该组件托管ag-grid-angular，并指定在网格中使用哪些angular组件

constructor() {

this.gridOptions = <GridOptions>{

context: {

componentParent: this

}

};

this.gridOptions.rowData = this.createRowData();

this.gridOptions.columnDefs = this.createColumnDefs();

}

// 在子组件中——在网格中动态创建的angular组件

// 父组件可以按如下方式访问:

this.params.context.componentParent

请注意，尽管我们在componentParent这里使用了属性名，但它可以是任何东西-重点是您可以使用该context机制在组件之间共享信息。

1 - 父组件

<https://github.com/ag-grid/ag-grid-angular-example/blob/master/systemjs_aot/app/dynamic-component-example/dynamic.component.ts>

2 - 子组件

<https://github.com/ag-grid/ag-grid-angular-example/blob/master/systemjs_aot/app/dynamic-component-example/child-message.component.ts>

筛选

网格可以使用过滤显示提供的行的子集。

1 列过滤器：列过滤器出现在列菜单和/或“过滤器工具面板”中。设置在列上的过滤器仅使用该列中的数据进行过滤。

2 快速过滤器：快速过滤器是给网格的一段文本（通常用户会在应用程序的某个位置键入它），用于使用网格中所有列中的数据过滤行。

3 外部过滤器：外部过滤器是应用程序独立于网格进行的任何过滤而过滤出行的一种机制。

最小高度与自动高度

显示行自动高度时，最小高度为50px。这是为了美观，特别是当网格中没有行时，允许显示“no rows”消息的空间，否则这个消息将覆盖在头部，这看起来不太好。  
当使用auto-height时，不可能指定最大高度。

Flexbox容器内  
默认情况下，网格运行一个计时器来监视其容器大小并相应地调整UI的大小。这可能会干扰display: flex set元素的默认行为。简单的解决方法是在网格元素的父元素中添加overflow: hidden。  
在新选项卡中打开下面的示例并调整窗口大小，以查看网格实例如何相应地调整大小。