|  |  |  |  |
| --- | --- | --- | --- |
| Person | Role | Date | Notes |
| Vinoth Kumar S | Writer | May 18, 2020 |  |
|  |  |  |  |

Getting Started with the Blazor HeatMap Chart

Hi everyone, welcome to this tutorial on how to get started with the Blazor HeatMap Chart component from Syncfusion. Our Blazor HeatMap Chart display simple or large matrix data graphically through color variations in SVG or canvas. In this video, I’ll explain you how easy it is to add a Blazor HeatMap Chart component from Syncfusion to a server-side Blazor application. I will also show you how to configure few of its basic features like setting data points, bubble heat map chart, axis, palette, dimensions, and tooltip. To explore the other features available in Syncfusion Blazor Bullet Chart component, refer the website link shown in the above card.

Let me start with the application creation process. I can create Blazor application using either Visual Studio 2019 or Visual Studio Code. In this video, I’ll show you how to create a server-side Blazor application using Visual Studio 2019. Make sure that you have installed it along with .NET Core SDK 3.1.3, or later. If you want to know more about the Blazor server app and how to add syncfusion components to it, then you can check the video link that is shown in the above card.

Now, open Visual Studio 2019 and choose **Create a new project** from the dashboard. Select **Blazor App** from the templates and click the **Next** button. Here, I’ll name my project and click the Create button.

Here, I will choose the **Blazor Server App** and click the Create button.

The project has been created and is ready for use. Now, I will show you the step-by-step process for adding a HeatMap Chart to it.

To add and start using a HeatMap Chart, first I need to install the necessary Syncfusion Blazor packages. For that, I’ll open the NuGet package manager and I search for **Syncfusion Blazor** and install it into my application.

Once installed, you can see that the Blazor packages have been added to this application’s packages folder. Next, to render a HeatMap Chart, first I’ll navigate to **\_imports.razor** page and import the namespace **Syncfusion.Blazor.HeatMap**.

Then I need to register the Syncfusion Blazor service. For that, I’ll navigate to the **Program.cs** file and add the **Syncfusion.Blazor** namespace. And then I add the **Syncfusion Blazor Service** method “**AddSyncfusionBlazor**” inside the **Main** method. Within the same method, I register the Syncfusion license by passing the trial license key in the RegisterLicense() method.

Next, I will navigate to the **index.html** page within WWWroot folder to add a CSS reference from the installed Syncfusion Blazor package.

Now, the configuration steps are done. Let me continue to add the Blazor HeatMap Chart now. For that I’ll open the index component, and then I’ll include the SfHeatMap tag. To visualize the HeatMap chart in browser page, I will bind data to it using the data adaptor support. data can be bound to the heat map using **DataSource** property. I will declare public variable HeatMapData of object type. Then I define the method DefaultData() of type two dimensional array of int. I declare variable datasource and assign the values. And then I return the datasource variable See here the data points that are null are considered as EmotyPoints. I define **OnInitialized** method and assign the **DefaultData** method to **HeatMapData** variable. Now assign this variable to the **DataSource** property of HeatMap Chart. See the output, HeatMap chart is rendered with provided data. See that, default palettes are rendered for the data points with null value.

Next, I will show you how to add the title to the HeatMap Chart. I set the title, by adding **HeatMapTitle** property and set **Text** property value as “Sales Revenue”. See the output, the chart rendered with title.

Now. I will show you how to set the X and Y axis label to the Heat Map Chart. Here you can set the three types of axis – Category, Numeric and Datetime axis. Let me add the category axis first. For that I declare variable of XAxis of type string array and define the array values. In the sample way, I will declare the YAxis variable with array values. Now I will assign these variables to the **Labels** property of HeatMapXAxis and HeatMapYAxis tag with **Type** property value as category. See the output, the Heat Map Chart rendered with Category labels in X and Y axis. Here I can also render the Numeric labels simply by changing the type as Numeric in HeatMapXAxis and HeatMapYAxis tag. See the output, Heat Map chart rendered with Numeric axis.

Next, I will show you how to format the axis in Heat Map chart. Here I will include the **LabelFormat** property in HeatMapYAxis tag and set its value to ${value}. See the output, currency format is applied to the Y-axis.

Data points represent the data source values with Gradient or Fixed colors in the heat map. I will show you how to change the palette types in HeatMap Chart - Gradient or Fixed. Let see about the Gradient paleete type. The heat map calculates all the gradient colors between the start and end colors for all distinct data values. Default start color and end color will be considered for gradient calculation if the colors are not defined. Here I will set the **Type** property value as Gradient inside the HeatMapPaletteSettings tag. Then I will include the HeatMapPalette tag inside the HeatMapPaletteSettings tag and specify the color value using **Color** property. In the same way I define two more colors. See the output, HeatMap Chart rendered with gradient palette type.

In fixed palette type, solid colors are applied to the heat map cells. The data values can be grouped based on the number of colors defined for the heat map. The palette type should be defined as **Fixed** for the **Type** property in the **HeatMapPaletteSettings** property. See the output, HeatMap Chart rendered with fixed palette type.

Next, I will show you how to render Bubble heat map. The data points can be represented in the bubble along with its attributes by setting the **TileType** property to **Bubble** inside **HeatMapCellSettings** tag. Here data points can be displayed as bubble size, bubble colors, or sector. Here I will include the **BubbleType** property and set its value to Size. See the output, Bubble chart rendered with Type as Bubble size.

Next, I will show you how to set dimension for Heat Map Chart. I can specify the dimension in pixel or in percentage. Here I will include the **Height** and **Width** properties with some pixel values in SfHeatMap tag. See the output, Heat Map chart rendered based on the dimensions specified.

Finally, I will show how to set tooltip in Heat Map chart just by enabling the **ShowTooltip** property values in SfHeatMap tag. See the output, tooltip in shown in Heat Map Chart.

Now let me make a quick summary of what we have seen so far. I have shown you how to create a Blazor server app and add a Bullet Chart component to it. And then I showed how to configure a few of its basic features like setting data points, bubble heat map chart, axis, palette, dimensions, and tooltip.

You can download this working example from the GitHub link in the description for this video below. You can also see about getting a free license key to use our Blazor products if you’re eligible for our Community License.

Thank you for watching this video. If you found this video useful, give a like and subscribe to our channel. Thank you.