Galan Maria Lavinia

Group 248/2

Bar Chart Web Component

Requirement

Build a software system that will have to illustrate the creation of a new component for a given framework.

Project

I created a web component called “my-bar-chart” for Angular framework. It can create custom bar charts based on the provided data.

About Angular

Angular is a development platform, built on TypeScript. As a platform, Angular includes component-based framework for building scalable web applications, a collection of well-integrated libraries that cover a wide variety of features, including routing, forms management, client-server communication etc. as well as a suite of developer tools to help in developing, building, testing, and updating the code.

In Angular, components are the building blocks that compose an application. A component includes a TypeScript class with a @Component() decorator, an HTML template, and styles. The @Component() decorator specifies the following Angular-specific information: a CSS selector that defines how the component is used in a template (HTML elements in a template that match this selector become instances of the component), an HTML template that instructs Angular how to render the component and an optional set of CSS styles that define the appearance of the template's HTML elements.

My Bar Chart

The purpose of this new component is to easily integrate various custom bar charts inside a web page. It consists of three main files: a Typescript file, an HTML file and a CSS file.

The typescript file (bar-chart.component.ts) includes, as described above, the @Component decorator that specifies its selector (how the component will be named within a HTML template) – in this case it is “my-bar-chart”, an HTML template url – the relative path to bar-chart.component.html file – and the relative path to bar-chart.component.css, denoting the files with the used CSS styles.

```typescript
@Component({
  selector: 'my-bar-chart',
  templateUrl: './bar-chart.component.html',
  styleUrls: ['./bar-chart.component.css']
})
export class BarChartComponent implements OnInit {
  @Input() data: Array<any>;
  @Input() theme?: String;
  @Input() direction?: String;

  total = 0;
  maxHeight = 300;
  lineWidth = "";
```

bar-chart.component.ts

The bar chart receives as input when used in an HTML template three values, out of which two are optional. The data array contains all the data necessary to create the bar-chart, namely objects with at least two fields – the label and the value (the numerical value, e.g. the percentage); other fields may be the description associated with each entry or the colour desired for that specific bar. Another input value is the theme – it can be dark or light (in the case of the dark theme, the bars will be black; for the light theme they will be white. If the theme is missing, they will be grey). Furthermore, the direction can be specified via direction input value – denoting if the orientation of the bars will be vertical or horizontal.

The ngOnInit() method initializes the variables: the total is used in order to know what dimension each bar should have (related to the overall proportions) and the line width (useful for vertical bars chart) is computed based on the total number of entries in the data array. This method parses the array and computes the sum of all values; after that, it sets for each entry in the array a size attribute corresponding to the relative dimension that the bar should have – transposed in style.height. It also sets a color attribute (the default according to the theme) if there was no other specified in input.

Each bar has a click event which triggers the setDescription(i: number) method followed by showing a dialog. The setDescription(i: number) method populates the text div in the dialog box with the description associated with the i-th element in the data array. If no description was provided, an appropriate message will be displayed on the dialog.

The HTML template file instructs Angular how to render the component. It contains a dialog element that will be opened when a bar was clicked, as I mentioned before. The dialog includes a text (the description of the data associated with that bar) and an OK button that simply closes the dialog. The file also provides the necessary code for displaying the bar – for each element in the data array, it displays a div having the class bars – referring to the value and the bar itself. The NgFor structural directive was used; it renders a template for each item in a collection. It dynamically specifies the color of each div from verticalBar class and its height (set in the NgOnInit). Moreover, depending on the isVertical attribute of the BarChartComponent class, a different type of chart will be shown (only the css and the arrangement are different).

```
<dialog #myDialog>
  <div class="centered" id="dataDescription"></div>
  <br>
  <button class="dialogButton" (click)="myDialog.close()">
    OK
  </button>
</dialog>


<div class="myChart" *ngIf="isVertical">
  <div class="content">
    <div class="bars" *ngFor="let item of data; let i = index" [attr.data-index]="i">
      <span>{{item.value}}</span>
      <div class="verticalBar" [style.background-color]="item.color"
           [style.height]="item.size" (click)="setDescription(i); myDialog.show()"></div>
    </div>
  </div>
  <div class="line" [style.width]="lineWidth"></div>
  <div class="nameLabels">
    <span *ngFor="let item of data">{{item.label}}</span>
  </div>
</div>
```

bar-chart.component.html

The CSS file defines the appearance of the HTML elements. It defines several classes used in creating the bars either horizontally or vertically – they were mainly created using flex, together with the dynamical height present in the HTML file. Moreover, it defines also the style of the dialog box containing the description.

To use this new component, I defined in the file app.component.ts an array of data, as follows:

```
public eurovisionScores : Array<any> = [
  {label : "Finland", value: 526, description: "The Finnish singer Kaarija got the first place on televote. His song was Cha Cha Cha."},
  {label : "Israel", value: 326, description: "Noa Kirel represented Israel with the song 'Unicorn'.", color:"pink"},
  {label : "Sweden", value: 583, description: "Loreen won the Eurovision Song Contest with her song, 'Tattoo'."},
  {label : "Italy", value: 350, description: "The song 'Due Vite' by the Italian singer Marco Mengoni was an emotional and beautiful ballad."}
  {label : "Norway", value: 268},
  {label : "Moldova", value: 96, description: "Pasha Parfeni got the 18th place with his song 'Soarele si luna'."}
];
public theme = "dark";
```

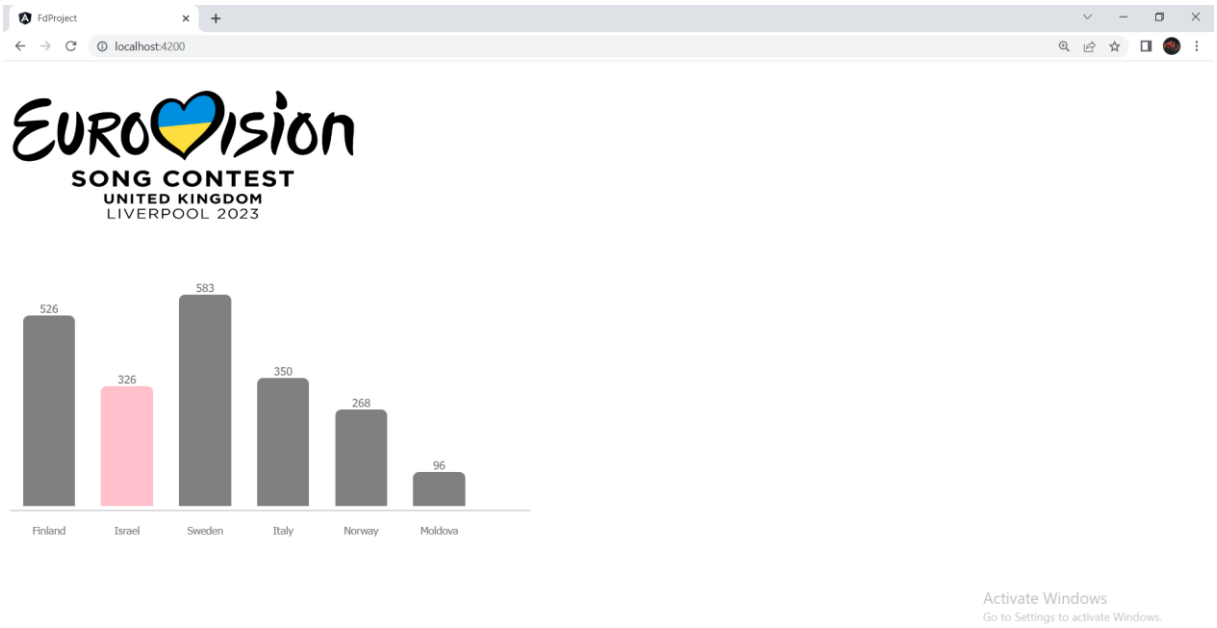
app.component.ts

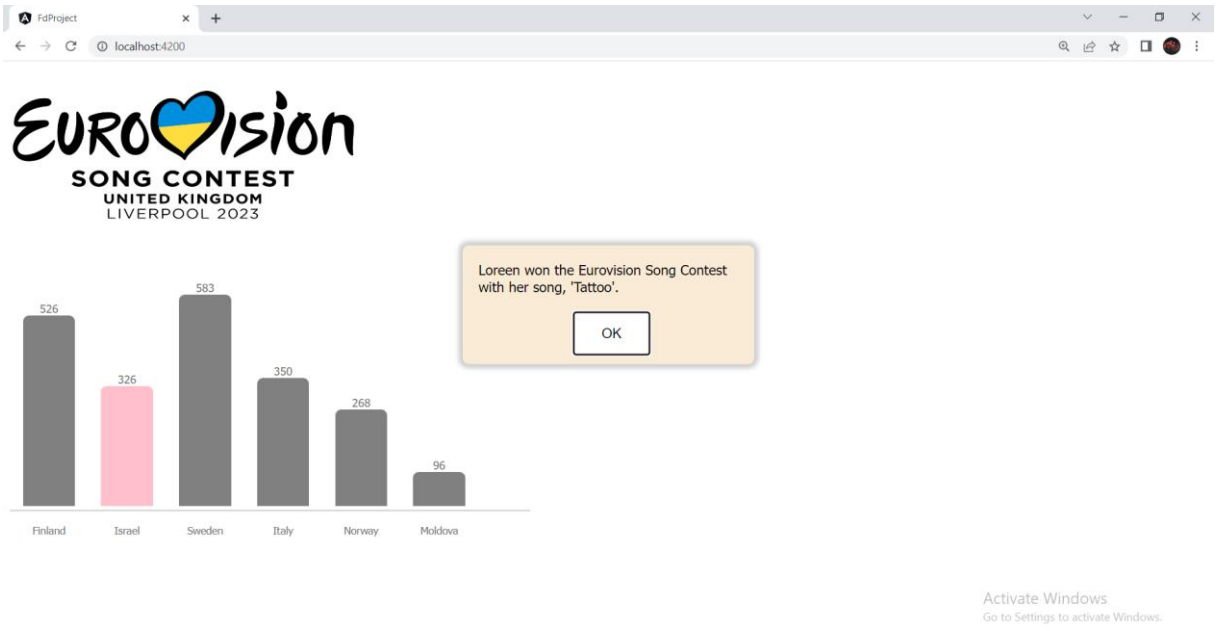
Finally, “my-bar-chart” can be used in another HTML file, integrating the new component into another one. The objects in the data array must contain the fields label and value, while description and color are optional.

```
<div>
  <my-bar-chart [data]=eurovisionScores></my-bar-chart>
</div>
```

app.component.html

The following screenshots are taken from the resulting web application.





To conclude, “my-bar-chart” is a component for creating custom bar charts and display the description of each bar. It supports different combinations of colors, numbers of items and so on and it can easily be integrated inside another component of an web application, as it was shown above.