pie-chart-comp-d3

pie-chart-comp-d3 is a Vue.js (>= 2.5) web component that draws svg pie/donut charts. **pie-chart-comp-d3** depends on the vue.js, various modules from d3.js, along with table-comp from the deandevl repositories. The dependencies can be installed via npm install with the included `package.json` file in the demo folder. **pie-chart-comp-d3** offers several features including:

- adjustable inner radius controls layout as pie (inner radius = 0) or donut (inner radius > 0)
- main titles are draggable
- legend in the form of a table that can optionally be hidden
- tooltip for displaying a slice's value on mouse hover
- slices can be selected programmically
- mouse click on either a table row or pie slice highlights the selection on both table and pie
- an event containing slice information is emitted on a slice mouse click
- control over format of value output
- adjustable margins around the chart
- CSS variables are provided for easily controlling chart colors, backgrounds and font sizes

pie-chart-comp-d3 can be installed via with the included `package.json` file for a local installation via the npm install command. **pie-chart-comp-d3** depends on some d3 modules and the vue.js framework. A demo folder is provided that used Parcel together with its associated `package.json` file to bundle together **pie-chart-comp-d3** along with its vue.js /d3 dependencies for a simple application. Further details are provided below for running the demo.

Props

A prop in Vue.js is a custom attribute for passing information from a parent component hosting **pie-chart-comp-d3** instance(s) to an **pie-chart-comp-d3** as a child component. **pie-chart-comp-d3** has the following props for a parent to bind and send information to:

- chart_data -- an array of javascript objects with pairs of variable names for keys and a string/numeric value as the keys ' values
- keys -- an object for defining the data keys for text, value, and color (see below)
- slice_index -- a number that selects and highlights the slice from a specific chart_data array index
- value_format -- a string defining the d3 value format (see d3.format)
- outer_radius -- outer radius of the pie (default: 200)
- inner_radius -- inner radius of the pie (default: 0)
- title_1, title_2 -- strings defining the main draggable chart titles
- margin_top -- chart's top margin (default: 90)
- margin_left -- chart's left margin (default: 60)
- margin_bottom -- chart's bottom margin (default: 40)
- show_legend -- boolean to control displaying the legend
- css_variables -- a javascript object that defines the css variables (see below)

Each row of `chart_data` is an object with keys for variable text, value, and color. The `slices` property is an object with 3 keys for identifying the data keys for text, value, and color. Below is a portion of `chart_data` from the demo -- an array of objects:

```
[
    {"age":"<5","population":2704659,"color":"#98abc5"},
    {"age":"5-13","population":4499890,"color":"#8a89a6"},
    {"age":"14-17","population":2159981,"color":"#7b6888"},
    ,,,
    ,,,
```

Also from the demo below is the `keys` property identifying the keys from the above data for text, value, and color:

```
keys: {text_key: 'age',value_key: 'population',color_key: 'color'}
```

Styling

The **css_variables** prop is a javascript object that contains any combination of css variable names as keys and associated values. The following list are the css variable names along with their default values for a quick styling of **pie-chart-comp-d3**:

```
  {
      pie_chart_compD3_font_family: Verdana,serif,
      pie_chart_compD3_color: black,
      pie_chart_compD3_background_color: white,

      pie_chart_compD3_slice_text_color: black,
      pie_chart_compD3_tooltip_color: black
  }
```

Events

pie-chart-comp-d3 has one event that notifies the parent component of the current clicked slice.

pie-chart-comp-d3 emits the following single named event:

```
'piechartcompd3_slice_clicked' -- returns an object containing the current slice
index, slice text, slice value, slice percent, and slice color.
```

The parent component can listen to the above event and provide a callback for further processing. Events emitted from a child component back to the parent is explained at Vue Custom Events.

The demonstration shows how the event can be incorporated.

Demonstration

One demonstration of **pie-chart-comp-d3** is provided in the folder named `demo`. It can be viewed by hosting the `index.html` file in the `dist` folder.

As a suggestion, install http-server locally/globally via npm then enter the command `http-server` in the **pie-chart-comp-d3** `dist` directory. From a browser enter the url: `localhost:8080/` to view the demo.

The demo folder contains a `package.json` file that can be used to setup dependencies for this demo and as a template for other applications using **pie-chart-comp-d3**.

The following is the setup for this chart contained in the `App.vue` file where `keys` is a reference to the `keys` in our above example:

```
    <pie-chart-comp-d3
        title_1="Distribution Across Ages"
        :title_2="title_2"
        inner_radius="80"
        value_format=".3s"
        :show_legend="show_legend"
        :chart_data="chart_data"
        :keys="keys"
        :css_variables="css_variables"
        v-on:piechartcompd3_slice_clicked="value => set_current(value)">
        <svg class="svg_chart_1"></svg>
      </pie-chart-comp-d3>
```

Note how the `pie-chart-comp-d3` tag wraps around the `<svg>` element. It is important that a class be assigned to the `svg` for **pie-chart-comp-d3** to locate the element. Among **pie-chart-comp-d3**'s attributes, it makes a reference to `chart_data` for the `chart_data` attribute. `chart_data` is set using a function called `read_csv` (from the d3fetchmodule module). Below is some of the code for reading the data.csv file:

```
read_data: function(){
      const convert = [
        {field: 'population', type: 'linear'},
      ];
      const get_data = async () => {
        try{
          this.chart_data = await read_csv('data/ages_population.csv', convert);
          //debug
          console.log(JSON.stringify(this.chart_data));
        }catch(e){
          console.log(e);
        }
      };
      get_data();
    }
```