Instructions for use

Programmation de composants Angular pour la mise en œuvre de tests utilisateurs

Benjamin Mitton

Cédric Pinard

Université Grenoble Alpes

Summary

Introduction

This document presents the operations to be carried out to use the basic-linechart library. It contains of the information to launch the library and even a link to find some example. It is assumed that the reader of this document has already read the installation manual.

If you are a developer, you can read the whole document to implement our library.

If you are a simple user that want to try it, you can read the last part “Use” and go on this website to try it : https://projet-ter-mitton-pinard.github.io/

Implementation

- In your app.module.ts, you must add BasicLinechartModule to imports of @NgModule.
- In your app.component.html, you can add the component : <lib-basic- linechart></lib-basic-linechart>

Parameters of the component

No parameters are required.

- Input [data]: Data[] default value : [], data displayed in the component (specified Data in the section below)

- Input [width]: number default value : 900, width of the component

- Input [height]: number default value : 200, height of the component

- Input [domain]: [number,number] default value : [0,0], domain of value (only for continuous values)

- Input [range]: [number,number] default value : [0,0], range of timestamp that we display in component

- Input [currenTime]: number default value : 0, timestamp for the current time line

- Input [speedZoom]: number]0;1] default value : 0.2

- Output (rangeChange): [number,number] to bind with a function in your app, to synchronize components ranges

- Output (currenTimeChange): number to bind with a function in your app, to synchronize components currentTime

/! Don't mix dataset with different value's type (continuous, positive integer) in one component.

/! Don't mix dataset with different range of timestamp in one component.

/! Don't bind range on components that have dataset with different ranges of timestamp

Interface Data

Represents one dataset. You can add an array of dataset in the component.

```
interface Data {
    label: string;
    values: [number,number][]; //[timestamp,value]
    color: string;
    style: "line" | "area" | "both";
    interpolation: "linear" | "step";
}
```

DataService

Contains function parseBool that you can use in generateData for boolean value.

Contains function generateData, which you can use to parse Data from a dataset str :

public generateData(str : string, label : string, color : string, style : "both"|"line"|"area", interpolation : "step"|"linear", f : (s:string)=>number): Data[]

/! str format example :

‘"2016-07-25 15:47:24,459";"PC6";"OFF"

"2016-07-25 19:47:24,459";"PC6";"ON"’

/! Fill parameter f with parseBool or parseFloat

Examples :
- generateData("PC6","#124568","both", "step", parseBool)
- generateData("Temperature_Salon", "purple", "line", "linear", parseFloat)

Contains dataExamples : Data[]. You can import them to test the component (show in the example below).

Example (tutorial)

app.component.ts

Write in the main class :

```
public data1:Data[]=[];
public data2:Data[]=[];
public data3:Data[]=[];
public data4:Data[]=[];
public data5:Data[]=[];
public data6:Data[]=[];
public datatest:Data[]=[];
public range: [number, number] = [0,0];
public currentTime : number =0;
public range2: [number, number] = [0,0];
public currentTime2 : number =0;
constructor(data : DataService){
  this.data1=data.dataExample1;
  this.data2=data.dataExample2;
  this.data3=data.dataExample3;
  this.data4=data.dataExample4;
  this.data5=data.dataExample5;
  this.data6=data.dataExample6;
}
```

```
  public updateRange(rangeChange: [number,number]){
    this.range=rangeChange;
  }
  public updateCurrentTime(currentTimeChange: number ){
    this.currentTime=currentTimeChange;
  }
  public updateRange2(rangeChange: [number,number]){
    this.range2=rangeChange;
  }
  public updateCurrentTime2(currentTimeChange: number ){
    this.currentTime2=currentTimeChange;
  }
  public change(i: number){
    if(i==1) this.datatest = this.data4;
    if(i==2) this.datatest = this.data5;
    if(i==3) this.datatest = this.data6;
  }
```

app.component.html

Write :

```
<lib-basic-linechart [data]=data2 [range]=range
        (rangeChange)="updateRange($event)" [currentTime]=currentTime
        (currentTimeChange)="updateCurrentTime($event)"></lib-basic-linechart>
<lib-basic-linechart [data]=data1 [domain]=[0,30] [range]=range
        (rangeChange)="updateRange($event)" [currentTime]=currentTime
        (currentTimeChange)="updateCurrentTime($event)"></lib-basic-linechart>
<lib-basic-linechart [width] = "1200" [height]="200" [data]=data3 [range]=range
        (rangeChange)="updateRange($event)" [currentTime]=currentTime
        (currentTimeChange)="updateCurrentTime($event)"></lib-basic-linechart>
```

```
<lib-basic-linechart [speedZoom]=0.7 [data]=data4 [range]=range2
        (rangeChange)="updateRange2($event)" [currentTime]=currentTime2
        (currentTimeChange)="updateCurrentTime2($event)"></lib-basic-linechart>
<lib-basic-linechart [speedZoom]=0.7 [data]=datatest [domain]=[26,27]
        [range]=range2 (rangeChange)="updateRange2($event)"
        [currentTime]=currentTime2
        (currentTimeChange)="updateCurrentTime2($event)"> </lib-basic-linechart>
    <button (click)='change(1)'>Data 4</button>
    <button (click)='change(2)'>Data 5</button>
    <button (click)='change(3)'>Data 6</button>
```

There are some examples implemented on :

https://projet-ter-mitton-pinard.github.io/

Use

On the linechart component you can :

- zoom with ctrl+wheel

- move the horizontal scrollbar, to move the range of timestamp

- drag and drop the vertical current-time line to move it