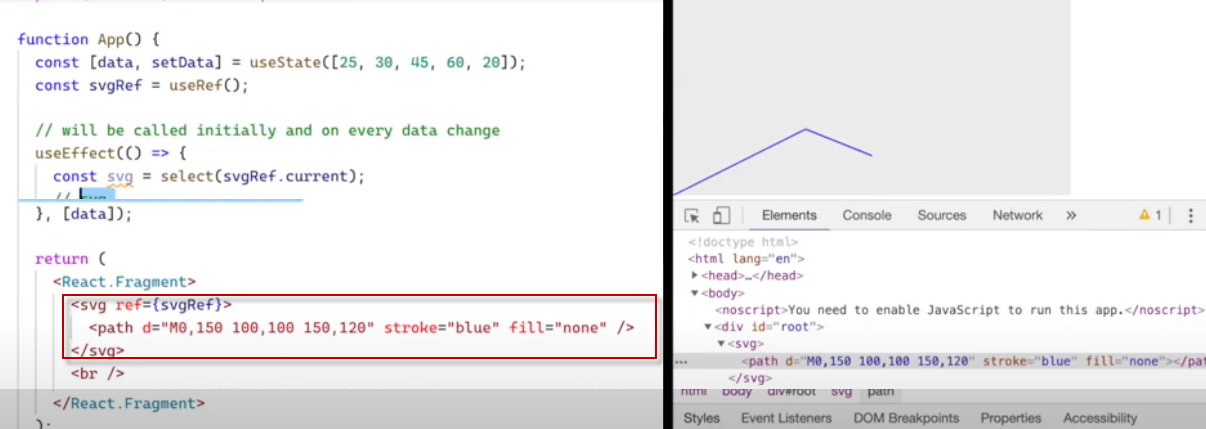
02 Curved line chart

<https://www.youtube.com/watch?v=hR8xtl_IbCw>

polyline or path elements draw lines with D3, this video uses path

the “d” defines the path, where it starts, goes and ends.



// https://www.youtube.com/watch?v=hR8xtl\_IbCw

import React, { useState, useRef, useEffect } from "react";

//  need to import line from d3 to use to define the value of d to make use  path element, curvedCardinal for that type of line

import { select, line, curveCardinal } from "d3";

*function* App() {

*const* [data, setData] = useState([25, 30, 45, 60, 20, 75]);

*const* svgRef = useRef();

  useEffect(() *=>* {

    //

*const* svg = select(svgRef.current);

    // this myline will be passed in as values for the "d" attribute of the line

    // the d element defines where path is drawn where it starts, where it goes and where it ends

    // myline is a function that will generate the d value based on the array of data it gets

*const* myLine = line()

      // based on the value of the index in the array

      .x((*value*, *index*) *=>* index \* 50)

      // y will be the value of the element in the array

      // doing the 150 - value will position the line starting at the bottom of the svg

      .y((*value*) *=>* 150 - value)

      // makes a nicely smooth and curvy line. This needs to be a d3 import

      .curve(curveCardinal);

    svg

      // path draws a line in D3, select all the path elements found in the SVG

      .selectAll("path")

      // syncronize data from the data array

      // need to put the data array in another array here so that D3 dosent

      // generate a new path element for every element in the data array

      // just want one path element for the entier array

      .data([data])

      .join("path")

      // attach for every entering and updating element

      // the d element defines where path is drawn where it starts, where it goes and where it ends

      // this call back will recive the entier data array as an argument

      // returns the d and attaches it to the path element

      .attr("d", (*value*) *=>* myLine(value))

      // a line so no fill

      .attr("fill", "none")

      // need stroke to see it

      .attr("stroke", "blue");

  }, [data]);

  return (

    <React.Fragment>

      <svg ref={svgRef}></svg>

      <br></br>

      <button onClick={() *=>* setData(data.map((*value*) *=>* value + 5))}>

        Update Data

      </button>

      <button onClick={() *=>* setData(data.filter((*value*) *=>* value < 35))}>

        Filter Data

      </button>

    </React.Fragment>

  );

}

export default App;