RespVis: A D3 Extension for Responsive SVG Charts

Keith Andrews
Graz University of Technology
Graz, Austria
kandrews@tugraz.at

David Egger
Graz University of Technology
Graz, Austria
david.egger@student.tugraz.at

Peter Oberrauner
Graz University of Technology
Graz, Austria
oberrauner.peter@gmail.com

***Abstract*—RespVis is an open-source JavaScript library which extends D3 to support the creation of responsive SVG charts. Each chart is a composite SVG document, whose elements can be positioned and styled with CSS. Chart authors can use CSS media queries and CSS Flexbox and Grid syntax to (re)position chart components such as title, legend, axes, and the chart itself in a responsive way. Example charts illustrate how to implement other responsive patterns using CSS and JavaScript, such as rotating labels, thinning out tick marks, or flipping a chart by 90°.**

RespVis is implemented in TypeScript as an extension of D3. It uses a novel custom layouter to enable the various SVG chart components to be positioned via CSS layout mechanisms, which are normally reserved for HTML elements.

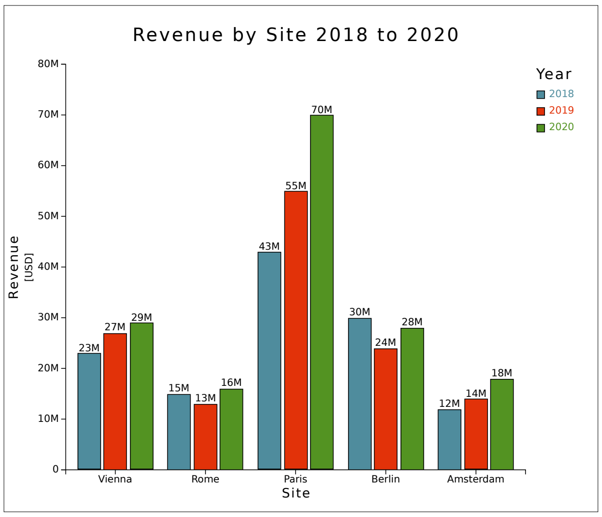
***Index Terms*—responsive, visualisation, chart, SVG, CSS, flexbox, grid, component, layout, pattern, D3, TypeScript.**

I. Introduction

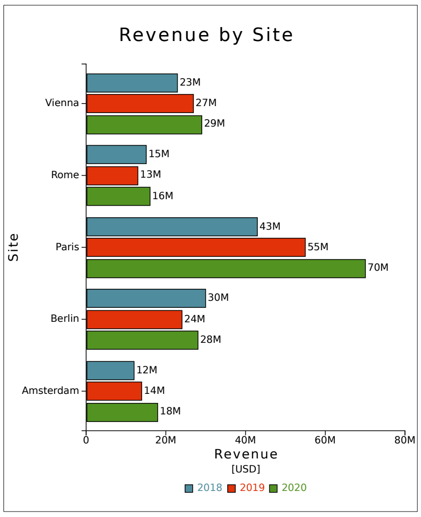
The predominance of the responsive web design paradigm, whereby web pages respond and adapt to the size and characteristics of the display device, has implications for web-based charts and graphics. To be embedded in responsive web pages, these too must become responsive, adapting to the amount of space available, and supporting whatever interaction modalities (pointer, keyboard, touch) are available on the end user's device. Note that the same chart code is served to all devices; the responsive logic is embedded within the chart.

Fig. 1 shows an example of a responsive grouped bar chart. On a wider screen, the legend is positioned to the right of the chart and a longer chart title can be used. On a narrower screen, the chart is rotated 90 degrees, the legend is positioned beneath the x-axis, and a shorter title is used.

The RespVis framework is designed to make it easier for chart builders to create responsive charts in JavaScript. It is open-source and is hosted on GitHub [1]. The library is built in ESM, CommonJS, and IIFE formats. Examples of common charts are provided, including line chart, multi-line chart, bar chart, grouped bar chart, stacked bar chart, and scatterplot. Live demos are available of both the current stable master branch [2] and the current developer branch [3]. To see the responsive nature of the charts in action, open one of the examples, then grab the side of the browser window and make it narrower and wider.



(a) Wide (>40rem).



(b) Narrow (<40rem).

Fig. 1: A responsive grouped bar chart created with RespVis. On a wider screen, the legend is positioned off to the right. On a narrower screen, the chart is rotated 90°, the legend is positioned beneath the x-axis, and the title is shortened.

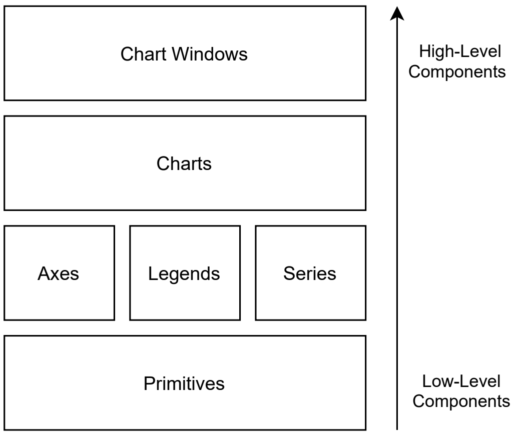
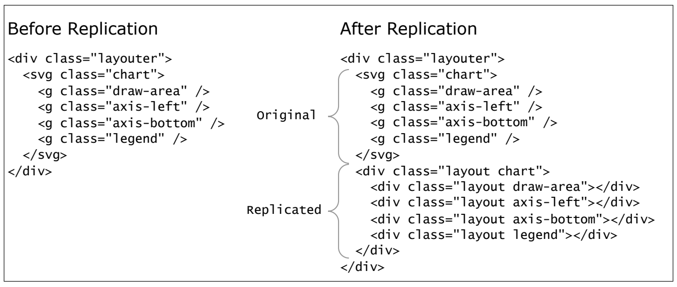


Fig. 2: The four layers of components in the RespVis library. Higher layers contain increasingly higher-level components.


Before Replication
After Replication
Original
Replicated

```
<div class="layouter">
  <svg class="chart">
    <g class="draw-area" />
    <g class="axis-left" />
    <g class="axis-bottom" />
    <g class="legend" />
  </svg>
</div>
```

```
<div class="layouter">
  <svg class="chart">
    <g class="draw-area" />
    <g class="axis-left" />
    <g class="axis-bottom" />
    <g class="legend" />
  </svg>
  <div class="layout chart">
    <div class="layout draw-area"></div>
    <div class="layout axis-left"></div>
    <div class="layout axis-bottom"></div>
    <div class="layout legend"></div>
  </div>
</div>
```

Fig. 3: The RespVis Layouter internally replicates a nested structure of SVG `<g>` elements with an equivalent nested structure of HTML `<div>` elements.

II. Responsive Data Visualisation

As outlined by Andrews [4, 5], responsive data visualisation applies the principles of responsive web design to web-based charts and visualisations. A responsive data visualisation adapts (responds) to the available display space and the characteristics of the display device. Some early prototypes can be seen at Andrews and Smrdel [6]. A broader view is taken by Horak et al. [7], who consider factors such as posture, how a device is held, and environmental factors. Hoffswell et al. [8] and Kim et al. [9] both give extensive surveys of responsive visualisation techniques, grouping them into categories of responsive patterns.

III. RespVis

RespVis is implemented in TypeScript as an extension of D3 [10]. D3 is a very popular JavaScript library, widely used to produce information visualisations and charts inside the web browser by associating SVG nodes with data items. RespVis uses a four-layer component hierarchy, illustrated in Fig. 2. Components in higher layers are at an increasingly higher level of abstraction.

```
@media (max-width: 40rem) {
  .chart {
    grid-template:
      'header header'
      'axis-left draw-area' 1fr
      '. axis-bottom' auto
      '. legend' auto / auto 1fr;
  }
}
```

Listing 1: Using CSS Grid syntax to responsively reposition the SVG chart components (title, axes, legend, and drawing area) of the grouped bar chart shown in Fig. 1b for a narrower screen.

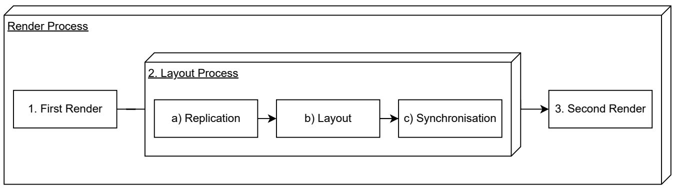


Fig. 4: The RespVis Layouter uses the browser to lay out the HTML `<div>` structure, then applies the calculated coordinates back to the original SVG group structure.

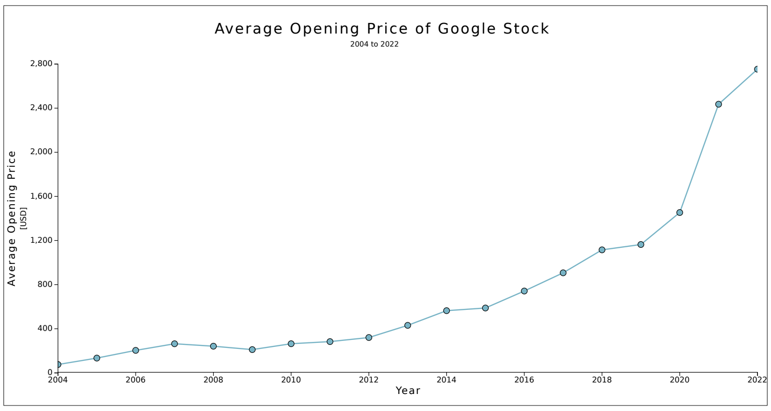
One of the issues in building responsive charts is the responsive placement of chart components such as the title, legend, and axes. CSS has two powerful layout mechanisms for the placement of HTML elements: Flexbox and Grid. However, these are only applied to HTML elements. Having access to them for SVG elements would greatly simplify the task of responsive re-layout for SVG chart components.

RespVis features a custom layouter which makes Flexbox and Grid (and indeed any CSS layout mechanism supported by the browser) available for SVG elements. The RespVis Layouter internally replicates a nested structure of SVG `<g>` elements with an equivalent nested structure of HTML `<div>` elements, as shown in Fig. 3.

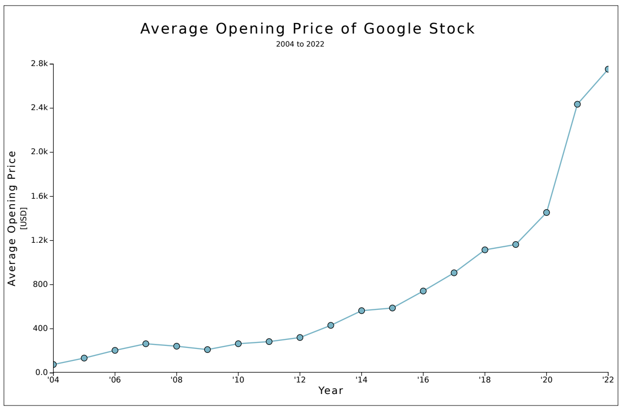
With this in place, the chart author can use Flexbox or Grid syntax to responsively position components of the SVG chart, such as title, axes, legend, and the drawing area itself, as shown in Listing 1.

The RespVis Layouter then renders the chart in a two-stage process, shown in Fig. 4, where the browser is used to calculate the layout of the replicated HTML structure. The calculated coordinates are then transferred over to the original SVG group structure. This way, SVG chart components such as title, legend, axes, and the chart drawing area itself can be responsively repositioned using standard CSS mechanisms like Flexbox and Grid.

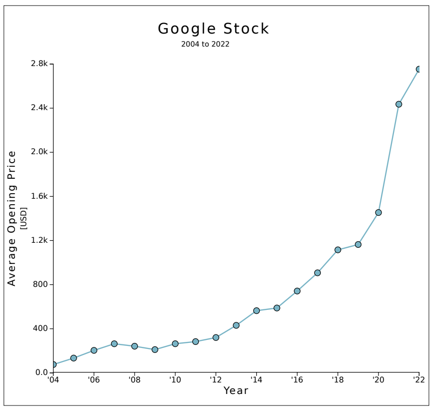
Other responsive patterns, such as rotating labels, shortening titles or labels, thinning out tick marks, flipping a chart by 90°, or removing certain elements can be implemented using a mixture of standard CSS mechanisms like media queries



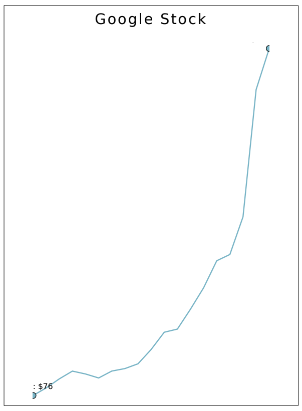
(a) Very wide (>80rem).



(b) Wide (60–80rem).

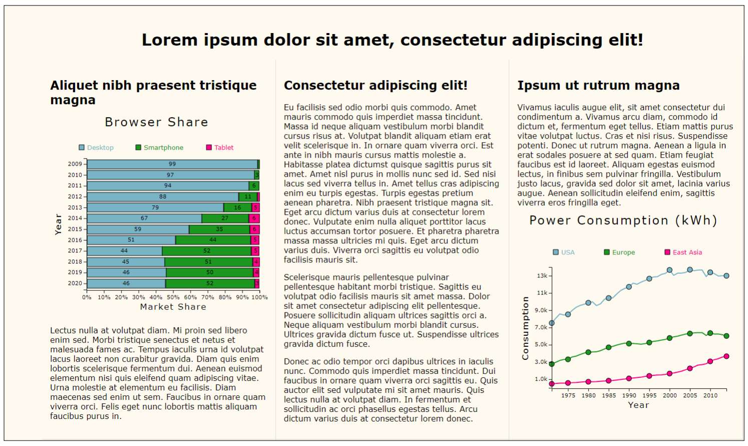


(c) Medium width (40–60rem).



(d) Narrow (<40rem).

Fig. 5: A responsive line chart. As the available space becomes narrower, the chart title and tick labels are shortened. At the narrowest width, the axes are removed and the line chart becomes a sparkline.



Lorem ipsum dolor sit amet, consectetur adipiscing elit!

Aliquet nibh praesent tristique magna

Year	Desktop	Smartphone	Tablet
2009	99		
2010	97	3	
2011	94	6	
2012	88	11	
2013	79	16	5
2014	67	27	6
2015	59	35	6
2016	51	44	5
2017	44	52	5
2018	45	51	4
2019	46	50	4
2020	46	52	3

Lectus nulla at volutpat diam. Mi proin sed libero enim sed. Morbi tristique senectus et netus et malesuada fames ac. Tempus iaculis urna id volutpat lacus laoreet non curabitur gravida. Diam quis enim lobortis scelerisque fermentum dui. Aenean euismod elementum nisi quis eleifend quam adipiscing vitae. Urna molestie at elementum eu facilisis. Diam maecenas sed enim ut sem. Faucibus in ornare quam viverra orci. Felis eget nunc lobortis mattis aliquam faucibus purus in.

Consectetur adipiscing elit!

Eu facilisis sed odio morbi quis commodo. Amet mauris commodo quis imperdiet massa tincidunt. Massa id neque aliquam vestibulum morbi blandit cursus risus at. Volutpat blandit aliquam etiam erat velit scelerisque in. In ornare quam viverra orci. Est ante in nibh mauris cursus mattis molestie a. Habitasse platea dictumst quisque sagittis purus sit amet. Amet nisl purus in mollis nunc sed id. Sed nisi lacus sed viverra tellus in. Amet tellus cras adipiscing enim eu turpis egestas. Turpis egestas pretium aenean pharetra. Nibh praesent tristique magna sit. Eget arcu dictum varius duis at consectetur lorem donec. Vulputate enim nulla aliquet porttitor lacus luctus accumsan tortor posuere. Et pharetra pharetra massa massa ultricies mi quis. Eget arcu dictum varius duis. Viverra orci sagittis eu volutpat odio facilisis mauris sit.

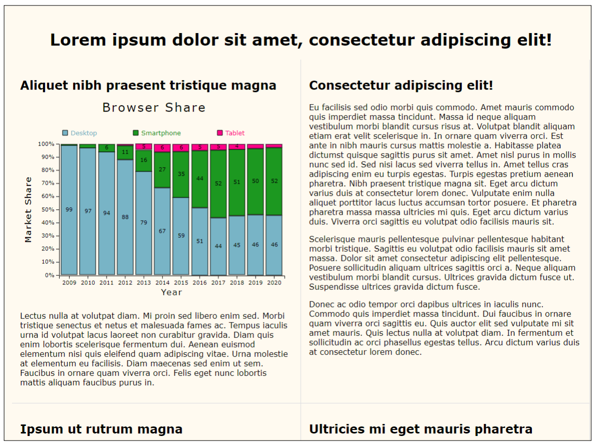
Scelerisque mauris pellentesque pulvinar pellentesque habitant morbi tristique. Sagittis eu volutpat odio facilisis mauris sit amet massa. Dolor sit amet consectetur adipiscing elit pellentesque. Posuere sollicitudin aliquam ultrices sagittis orci a. Neque aliquam vestibulum morbi blandit cursus. Ultrices gravida dictum fusce ut. Suspendisse ultrices gravida dictum fusce.

Donec ac odio tempor orci dapibus ultrices in iaculis nunc. Commodo quis imperdiet massa tincidunt. Dui faucibus in ornare quam viverra orci sagittis eu. Quis auctor elit sed vulputate mi sit amet mauris. Quis lectus nulla at volutpat diam. In fermentum et sollicitudin ac orci phasellus egestas tellus. Arcu dictum varius duis at consectetur lorem donec.

Ipsum ut rutrum magna

Vivamus iaculis augue elit, sit amet consectetur dui condimentum a. Vivamus arcu diam, commodo id dictum et, fermentum eget tellus. Etiam mattis purus vitae volutpat luctus. Cras et nisi risus. Suspendisse potenti. Donec ut rutrum magna. Aenean a ligula in erat sodales posuere at sed quam. Etiam feugiat faucibus est id laoreet. Aliquam egestas euismod lectus, in finibus sem pulvinar fringilla. Vestibulum justo lacus, gravida sed dolor sit amet, lacinia varius augue. Aenean sollicitudin eleifend enim, sagittis viverra eros fringilla eget.

(a) Wide (3 columns).



Lorem ipsum dolor sit amet, consectetur adipiscing elit!

Aliquet nibh praesent tristique magna

Year	Desktop	Smartphone	Tablet
2009	99		
2010	97		
2011	94	6	
2012	88	11	
2013	79	16	5
2014	67	27	6
2015	59	35	6
2016	51	44	5
2017	44	52	5
2018	45	51	4
2019	46	50	
2020	46	52	

Lectus nulla at volutpat diam. Mi proin sed libero enim sed. Morbi tristique senectus et netus et malesuada fames ac. Tempus iaculis urna id volutpat lacus laoreet non curabitur gravida. Diam quis enim lobortis scelerisque fermentum dui. Aenean euismod elementum nisi quis eleifend quam adipiscing vitae. Urna molestie at elementum eu facilisis. Diam maecenas sed enim ut sem. Faucibus in ornare quam viverra orci. Felis eget nunc lobortis mattis aliquam faucibus purus in.

Consectetur adipiscing elit!

Eu facilisis sed odio morbi quis commodo. Amet mauris commodo quis imperdiet massa tincidunt. Massa id neque aliquam vestibulum morbi blandit cursus risus at. Volutpat blandit aliquam etiam erat velit scelerisque in. In ornare quam viverra orci. Est ante in nibh mauris cursus mattis molestie a. Habitasse platea dictumst quisque sagittis purus sit amet. Amet nisl purus in mollis nunc sed id. Sed nisi lacus sed viverra tellus in. Amet tellus cras adipiscing enim eu turpis egestas. Turpis egestas pretium aenean pharetra. Nibh praesent tristique magna sit. Eget arcu dictum varius duis at consectetur lorem donec. Vulputate enim nulla aliquet porttitor lacus luctus accumsan tortor posuere. Et pharetra pharetra massa massa ultricies mi quis. Eget arcu dictum varius duis. Viverra orci sagittis eu volutpat odio facilisis mauris sit.

Scelerisque mauris pellentesque pulvinar pellentesque habitant morbi tristique. Sagittis eu volutpat odio facilisis mauris sit amet massa. Dolor sit amet consectetur adipiscing elit pellentesque. Posuere sollicitudin aliquam ultrices sagittis orci a. Neque aliquam vestibulum morbi blandit cursus. Ultrices gravida dictum fusce ut. Suspendisse ultrices gravida dictum fusce.

Donec ac odio tempor orci dapibus ultrices in iaculis nunc. Commodo quis imperdiet massa tincidunt. Dui faucibus in ornare quam viverra orci sagittis eu. Quis auctor elit sed vulputate mi sit amet mauris. Quis lectus nulla at volutpat diam. In fermentum et sollicitudin ac orci phasellus egestas tellus. Arcu dictum varius duis at consectetur lorem donec.

Ipsum ut rutrum magna

Ultricies mi eget mauris pharetra

(b) Narrow (2 columns).

Fig. 6: A simulated newspaper article. The charts are sized using CSS container queries. On a wider screen, a 3-column layout means proportionately less space for each chart. In the 2-column layout, the stacked bar chart adapts to having more room.

and container queries and pieces of custom JavaScript. The example charts show how to do this for a number of common responsive patterns.

IV. RespVis Usage

RespVis provides templates for responsive versions of a number of common chart types: line chart, multi-line chart, bar chart, grouped bar chart, stacked bar chart, and scatterplot. Fig. 5 shows a responsive line chart built with RespVis. As the available space becomes narrower, the chart title and tick labels are shortened. At the narrowest width, the axes are removed and the line chart becomes a sparkline.

Fig. 6 shows a simulated newspaper article. The charts are sized using CSS container queries. On a wider screen, a 3-column layout actually means proportionately less space for each chart. The stacked bar chart adapts by rotating itself by 90°. In the 2-column layout, the stacked bar chart takes on its more traditional form with vertical bars.

Furthermore, RespVis has built-in functionality to filter and select dimensions where several are available (for example, in a grouped bar chart), and to export whatever the current view is to an SVG file.

V. Outlook and Future Work

The RespVis framework is currently being extended to cover more chart types, including parallel coordinates. It will also be extended to make better use of touch and keyboard events where supported by the end user's device. The programming interface for chart builders needs some refactoring. Better use could also be made of container queries, now that they are widely available in modern web browsers. Although the main target users of RespVis are chart developers (chart builders), some user testing of the interactivity and functionality provided by RespVis charts to end users of the charts would also be beneficial and is planned.

VI. Concluding Remarks

RespVis is an open-source JavaScript library which extends D3 to support the creation of responsive SVG charts. The main contribution of RespVis is its custom layouter for the responsive placement of chart components such as title, legend, and axes using standard CSS layout mechanisms like Flexbox and Grid. In addition, the RespVis example charts illustrate how to implement a number of responsive patterns using CSS and JavaScript, such as rotating labels, shortening titles or labels, thinning out tick marks, flipping a chart by 90°, or removing certain elements.

Acknowledgements

Peter Oberrauner built the first version of RespVis as part of his Master's thesis [11]. The software is currently being updated and extended by David Egger.

References

[1] Peter Oberrauner, David Egger, and Keith Andrews. *RespVis*. Software Repository. June 11, 2023. https://github.com/tugraz-isds/respvis.

[2] David Egger, Peter Oberrauner, and Keith Andrews. *RespVis Stable Demo*. June 11, 2023. https://respvis.netlify.app/.

[3] David Egger, Peter Oberrauner, and Keith Andrews. *RespVis Dev Demo*. June 11, 2023. https://respvis-dev.netlify.app/.

[4] Keith Andrews. "Responsive Data Visualisation". In: *Graphical Web 2016 Talk* (Met Office, Exeter, UK). Nov. 2, 2016. https://youtu.be/cQKzpKfea-E.

[5] Keith Andrews. "Responsive Visualisation". In: *CHI 2018 Workshop on Data Visualization on Mobile Devices (MobileVis 2018)* (Montréal, Québec, Canada). Apr. 21, 2018. https://mobilevis.github.io/assets/mobilevis2018_paper_4.pdf.

[6] Keith Andrews and Aleš Smrdel. *Responsive Data Visualisation*. June 7, 2017. https://projects.isds.tugraz.at/respvis/.

[7] Tom Horak, Wolfgang Aigner, Matthew Brehmer, Alark Joshi, and Christian Tominski. "Responsive Visualization Design for Mobile Devices". In: *Mobile Data Visualization*. Edited by Bongshin Lee, Raimund Dachselt, Petra Isenberg, and Eun Kyoung Choe. CRC Press, Dec. 23, 2021. Chapter 2, pages 33–65. ISBN 0367534711. doi:10.1201/9781003090823-2. https://imld.de/cnt/uploads/Horak2021_MobileDataVisBook_Chap02_Responsive.pdf.

[8] Jane Hoffswell, Wilmot Li, and Zhicheng Liu. "Techniques for Flexible Responsive Visualization Design". In: *Proc. ACM Conference on Human Factors in Computing Systems (CHI 2020)* (Online). ACM, Apr. 25, 2020, pages 1–13. doi:10.1145/3313831.3376777. https://jhoffswell.github.io/.

[9] Hyeok Kim, Dominik Moritz, and Jessica Hullman. "Design Patterns and Trade-Offs in Responsive Visualization for Communication". In: *Computer Graphics Forum* 40.3 (June 2021), pages 459–470. ISSN 1467-8659. doi:10.1111/cgf.14321. https://arxiv.org/abs/2104.07724.

[10] Michael Bostock. *D3.js Data-Driven Documents*. June 11, 2023. http://d3js.org/ (visited on 06/11/2023).

[11] Peter Oberrauner. "RespVis: A Browser-Based, D3 Extension Library for Creating Responsive SVG Charts". Master's thesis. Graz University of Technology, Austria, May 12, 2022. 131 pages. https://ftp.isds.tugraz.at/pub/theses/poberrauner-2022-msc.pdf.