Effective Data Visualizations on Mobile Devices

Gauri Joshi, Prof. Alark Joshi

Abstract—Demand for data visualization on mobile devices is on the rise in every sector. One of the challenges in providing uninterrupted access to visualized information is the drastic reduction in screen size when moving from a desktop or laptop to a tablet or to a smartphone. Best practices for print or large screen graph presentations are unsuitable. Chart titles, axis labels, and other graph elements on a small screen are clutter rather than useful information. [1]

Index Terms— Data Visulizations, Effective Visulizations, Visulizations on Mobile Devices

INTRODUCTION

To deal with the issue of multiple screen sizes and operating systems, we planned to use the principle of responsive design (RD) along with the effective visualization design. The main idea behind this approach is to give control to a user to play with visualization to gain more insights and information from the data. For e.g. the first screen will only show the summary of the data and if user is interested in any subpart of the visualization s/he can click on it to get more details and it applies till the data reaches to granular level.

APPROACH

To reach this approach we tried many options to make visualizations better and useful. We have listed down milestones of our research below

1. Researched how to implement responsive designs. We divided reading into 2 different sections.
   - Responsive web applications
   - Responsive Data Visualizations for the web
2. Explore different options to make any visualization responsive. (Using React + Redux, Plain JS + D3 + CSS, Chart Libraries)
3. First step was to implement simple responsive Bar Charts to get an idea how responsive data visualization works.
4. Implemented few other plots like Line Charts, Scatter Plots, Calendar View etc. to get more insights on how we can use different combination of visualizations to make effective visualizations on mobile.
5. At the end we decided to explore Calendar View Charts in more details. We developed an application to explore time series data using Calendar View, Bar and Line Charts.
6. Basic idea behind this approach is simple. Give user control to see data in detail.

METHODOLOGY

To implement above-mentioned approach we followed below steps:

1. Dummy data creation
   In the current implementation data is created using plain vanilla javascript. Its simple data consists of temperatures taken at different times in a day for few years.
2. Convert data into required format
   Date is later divided in Year, Month and Day format for different visualizations.
3. Implement Bar Chart, Calendar Heatmap and Line Chart
   More details about implementation are mentioned in the Implementation Section.

Application is broadly divided into 3 modules.

- Year wise temp data summary – Used Bar Chart to show yearly avg. temp
- Month wise temp data summary – Used Calendar Heatmap/View to show monthly avg. temp
- Day wise temp data summary – Used Line Chart to display temp for that particular day

IMPLEMENTATION

As mentioned in the methodology section we have used Bar Chart, Calendar Heatmap and Line Chart for our approach. It's up to developer which data visualizations they want to use we chose these three categories because it was suitable for our use case. We worked on a very simple use case to show details of avg. temp for different years and if user is interested in learning more about the data in particular year show details of temp in months format for that particular year. If user is still interested in learning more about data for particular day s/he can choose particular day to view details about it. We have used starter code mentioned in the references it does something similar but for different use case [2]. We have modified it according to our use case and charts specifications. To sum up the idea we implemented is every use case flow can be converted into summary to granular data level flow and based on user's interest s/he can control the view of data visualization. It's particularly useful when user tries to seek information out of data on mobile devices.

RESULTS

To execute our demo application user needs to follow below steps

Prerequisites

- Install latest version of Node.js on your machine.
- Once Node.js is installed you can use 'npm install http-server -g' command to install http-server on machine. http-server is a simple, zero-configuration command-line http server.

Once prerequisites are done follow below steps

1. Download code on your machine using git clone command mentioned below
   git clone
   (Run this command on the terminal in the directory where you want to pull the code)
2. Go to folder and run 'npm install' command. This command will download and install required packages to run this application.
3. Run 'http-server .' to run the application.

Screenshots of an application

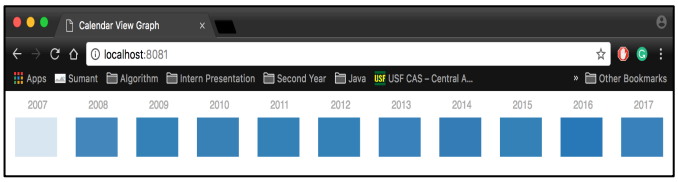

Fig. 1. Year wise avg. temp

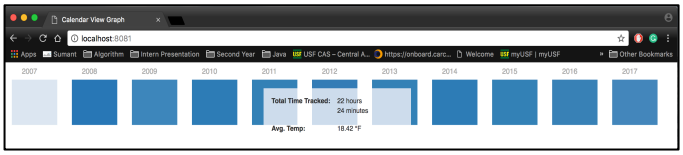

Fig. 2. Year wise avg. temp with tooltip

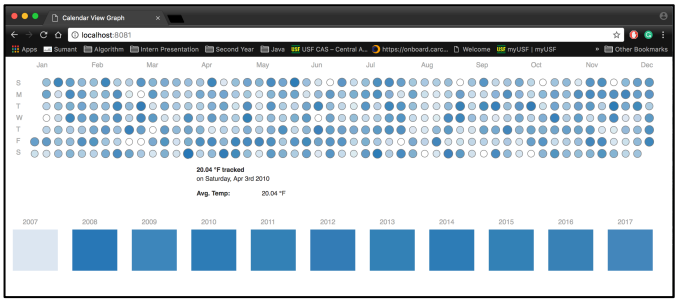

Fig. 3. Month wise avg. temp with tooltip

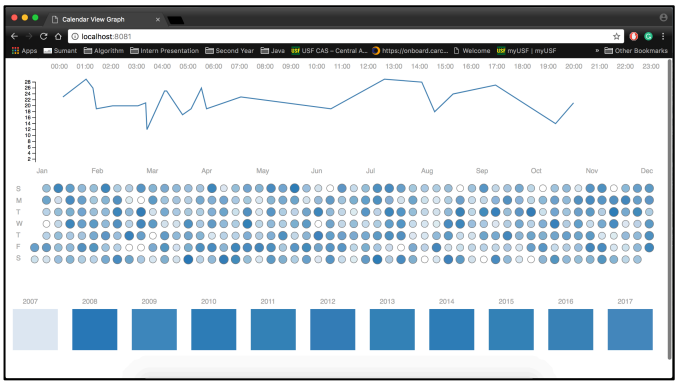

Fig. 4. Day wise avg. temp along with Month and Year view

Conclusion

Though this approach really worked well to visualize time series data effectively on Desktop and Mobile Screens the work can be extended to different datasets. Also selection of suitable visualizations and implementation of them can be automated for different use cases. In this particular application we can add few other options to see different trends in data for e.g. along with avg. temp we can add toggle button to visualize max temp. Also this application can be extended to test against any real-time time-series data.

Acknowledgments

The authors wish to thank A, B, C. This work was supported in part by a grant from XYZ.

References

[1] http://www.dummies.com/programming/big-data/big-data-visualization/data-visualization-designing-for-mobile-devices/
[2] Starter Code - https://github.com/g1eb/calendar-heatmap
[3] https://bl.ocks.org/mbostock/4063318
[4] https://github.com/DKirwan/calendar-heatmap
[5] http-server - https://www.npmjs.com/package/http-server
[6] http://eyeseast.github.io/visible-data/2013/08/28/responsive-charts-with-d3/
[7] https://thenextweb.com/dd/2015/06/12/20-best-javascript-chart-libraries/
[8] http://gionkunz.github.io/chartist-js/index.html
[9] https://medium.com/@caspg/responsive-chart-with-react-and-d3v4-afd717e57583
[10] https://blog.webkid.io/responsive-chart-usability-d3/
[11] https://bl.ocks.org/jfsiii/65bb6ffd9ad25f8d3cac83f6e77cbf65
[12] https://developer.mozilla.org/en-US/docs/Web/CSS/Media_Queries/Using_media_queries
[13] https://css-tricks.com/snippets/css/media-queries-for-standard-devices/
[14] https://www.html5rocks.com/en/mobile/responsivedesign/
[15] https://www.packtpub.com/books/content/web-development-react-and-bootstrap
[16] http://eyeseast.github.io/visible-data/2013/08/28/responsive-charts-with-d3/
[17] https://thenextweb.com/dd/2015/06/12/20-best-javascript-chart-libraries/
[18] http://gionkunz.github.io/chartist-js/index.html
[19] https://medium.com/@caspg/responsive-chart-with-react-and-d3v4-afd717e57583
[20] https://blog.webkid.io/responsive-chart-usability-d3/
[21] https://bl.ocks.org/jfsiii/65bb6ffd9ad25f8d3cac83f6e77cbf65
[22] https://bl.ocks.org/mbostock/3885705
[23] https://bl.ocks.org/d3noob/402dd382a51a4f6eea487f9a35566de0
[24] https://bl.ocks.org/mbostock/3883245