1. Abstract

In this work we present Protoviewer, a web-based design and collaborative data visualization environment designed around Protovis. Protoviewer aims to lower cognitive effort as well as the significant gulf of execution in designing effective and powerful visualizations by providing accurate categorizations of data while providing the user with the fullest control over its rendering.

2. Introduction

Developing any form of data visualization requires, at its simplest, identifying the tool to use, selecting the data to represent, and choosing the type of representation for that data. Many web-based toolkits and libraries are available, and often the choice of which to use is driven by the answers to the question of accessibility, efficiency and expressiveness, as highlighted by Bostock & Heer [1]. Examples are a powerful tool for determining expressiveness, while user familiarity with data manipulation and programming are key concerns regarding accessibility and efficiency.

We expand upon the previous work of one web-based toolkit, Protovis, which was developed to simplify the design process by increasing accessibility while maximizing expressiveness. It decomposes visualizations into individual components known as marks, which are then defined and associated with data, as well as to each other mark in the design. Similarly, Protoviewer decomposes the generation of visualizations into three separate steps: date manipulation, visualization selection, and iterative customization.

3. Approach

The layout divides the workspace into specific areas for data, design and code. Data input can be in either CSV or JSON format. Protoviewer suggests suitable visualizations based on the columns the user selects as desirable to display. This is largely similar to the methodologies adopted in Tableau’s “Show Me Alternatives” view. For example:

- 2Q: scatter plot, line chart, area chart
- 1Q, > 2C: stacked bars
- 1D, > 1C: candlestick, discrete lines
- 1C, > 1Q: grouped bars

The middle panel is allows users to directly manipulate the suggested visualization. Clicking on any Protovis mark will show the data associated with the mark, if available, as well as highlight the particular line in the code used to generate that mark. The data point can also be modified in-line.

Lastly, the right panel contains the JavaScript code used to render the visualization, as well as extra tools designed to add or manipulate the code. Any code change made in this panel is immediately processed and reflected in the design panel. Syntax is automatically highlighted for readability.

Informal feedback from the Protovis user community has been very positive. In particular, recognition was made regarding the need for a collaborative design space as well as an opportunity for rapid deployment and easy code modification in order to overcome the abstractions that hinder data exploration.

References