D3.JS: Data-Driven Documents

Roland van Dierendonck
Leiden University
rvdierendonck@gmail.com

Sam van Tienhoven
Leiden University
sammieboy12@gmail.com

Thiago Elid
Leiden University
thiago.elid@gmail.com

ABSTRACT

In this paper, we demonstrate the power of D3, an open-source JavaScript library which provides a toolkit for data visualization in web browsers. Since it makes use of major web standards, it has a great compatibility with web browsers making it easy to be viewed without the need of any additional plug-in. Moreover, we try to give a broad view over D3 given its superb flexibility, which makes it not only a charting library mostly used for data visualization, but also a drawing library with unimaginable applications in the web environment. We provide a demonstration on how to add the D3 library offer a brief guide for beginners to set first steps with data driven visualizations.

1. PURPOSE, CONTEXT AND HISTORY

Designs for envisioning quantitative information have a long history [20, 28]. Diagrams, maps and other graphical forms play an essential role in our reasoning about large quantitative datasets. Since the advent of the World Wide Web, the amount of available data has grown exponentially, as did the possibilities for visualizing and interacting with this data [24]. As a consequence, web browsers are increasingly considered to be a valid base for data visualization toolkits.

Data-Driven Documents (D3) is a free, open source JavaScript (JS) library released in 2011 by Heer, Ogievetsky and Bostock, then part of the Stanford Visualization Group. It provides a toolkit for data visualization in web browsers, using web standards: Cascading Style Sheets (CSS) for style, Scalable Vector Graphics (SVG) for vector graphics, HyperText Markup Language (HTML) for content and JavaScript for interactivity [2].

Using D3, the Document Object Model (DOM) can be efficiently manipulated on the basis of selected data, constructing and updating document elements. D3 bears resemblance to document transformers such as jQuery [18] that simplify the act of document transformation compared to the tedious W3C DOM API inherent in web browsers. D3 does not introduce novel representations of graphical elements, but uses the DOM's standard SVG syntax, that shows similarities with the graphical abstractions used in graphics libraries such as Processing and Raphaël. In addition, D3 also has helper modules that enable more complex visualizations comparable to toolkit-specific visualizations of systems such as Protovis [22].

Here a timeline of data visualization technologies developed by the Stanford Visualization Group and directly preceding D3 is presented. In recent years, the number of data visualization and graphing JavaScript libraries has exploded, an overview of which can be found here [16].

- 2005: Prefuse, a Java-based visualization toolkit developed, rendering visualizations within browsers via Java plug-in [21].
- 2007: Flare is an Actionscript library for data visualization, rendered within browsers via Flash plug-in [11].
- 2009: Protovis is launched, a JavaScript library to generate SVG graphics, using special representations [21].
- 2011: D3.js is released.

2. OPERATING PRINCIPLES

D3 is a JavaScript library. This means it is essentially a collection of prewritten JavaScript code. The D3 library was built to make working with HTML, CSS and SVG in conjunction with data easier. To understand how D3 works and what D3 is useful for, we must first have a basic knowledge of these three technologies and know what JavaScript is. We will discuss them briefly.

HTML [15] (HyperText Markup Language) is the standard language used to create websites. HTML consists of elements which are created with tags. For instance a header element is created by using a header tag. A tag typically is a letter or short words between angle brackets. An element is often denoted by an opening tag and a closing tag. The tags used to create a header element are <h> (opening tag) and </h> (closing tag). There are about 100 different tags which can be used to create a website. It is important to realize that HTML is a markup language and not a programming language. It is only used to define the content and structure of a website, and does not define any functionality of a website.

CSS [6] (Cascading Style Sheets) is a language often used in conjunction with HTML. CSS enables a designer to separate content and style. A piece of CSS code defines what the style of a specific HTML element must look like. For instance, you can define that all text within all headers must be purple or that the font size of one specific header should be bigger than the other headers. One of the biggest advantages of CSS is that you remove a lot of clutter from your HTML text.

SVG [23] (Scalable Vector Graphics) is a format for interactive two-dimensional images. SVG images are created by a simple language and can be rescaled without losing any quality. They have many advantages over other image types such as JPEG, which are traditional pixel based image formats. SVG images are created by describing the image instead of individual pixels values, which leads to a smaller and more flexible image format.

JavaScript [17] is a programming language based on C (and not on Java as one would suspect) which can be used within web pages to add more complexity and dynamic content to a website. Where HTML is used to define the content of a website and CSS is used to define the style of a website, JavaScript is used to define the behavior of a website. Both SVG and JavaScript can be used within HTML by using using a SVG tag or script tag respectively.

Creating a dynamic website with the above mentioned technologies can be tedious or complex. D3 is a library of JavaScript code which aims to make this easier. For instance, it enables the user to select all HTML elements with a certain tag and change their style, with one simple line of code. Something which is not possible without D3. It can also be used to create SVG images and style them and offers a possibility to create smooth transitions between styles.

Next we will discuss some of the key functionalities of D3: selections, transitions and update, enter and exit functions [9]. Selections allow the user to select and manipulate HTML elements in a very simple way. You can select an element by tag, class, ID, attribute value or containment and then perform an operation on your selection. If you want to perform more operations you can chain operations after you select an element. Moreover, you can select multiple elements at once by using the selectAll() method. Without D3 a programmer would have to iterate multiple times to perform the same action as selectAll() does. To perform the same operation as shown in example 1 without D3 would require a more complex code.

```
A selection with D3: (example 1)

d3.select("body")
.style("background-color", "black");
.style("color", “red");
```

Another key function of D3 is that it supports smooth transitions between styles. This allows for the fluent changes within graphs and images which D3 is known for. D3 creates transitions by interpolating. Take a look at example 2, it shows the code to create a smooth transition in background color from white to black.

```
A transition with D3: (example 2)
d3.select(“body")
.transition()
.duration(500)
.style("background-color", "black");
```

Finally we would like to discuss the update, enter and exit functions. Update, enter and exit functions can be performed on a selection to create an update, enter or exit selection, respectively (Figure 1). D3 is especially suited for working with data. This data can be dynamic, new data can be added or old data removed. Before D3 can work with data, data first has to be bound to elements in the DOM. This is done by the update function. The update function binds data to already existing elements in the DOM. If you want to bind more data items than available elements, the enter function comes into play.

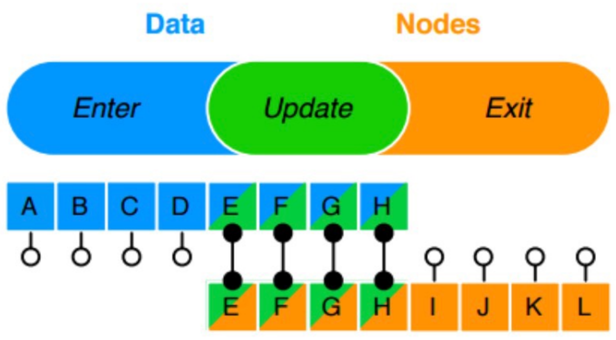


Figure 1: Update, Enter and Exit Functions in D3. [2]

The enter selection stores excess data items. If you now perform an update function the data that is already bound to elements in the DOM is updated. For the data stored in the enter selection, new elements are created and the data is then bound to these elements. Just as the enter function creates new elements, the exit function deletes elements that are no longer used. When an exit function is performed, data that is about to leave the dataset is selected. When a new update is performed the data is removed and the empty elements are deleted. For an example of the enter function see section 6, for a graphical representation of the update, enter and exit functions see figure 1 above.

3. STRENGTHS AND WEAKNESSES

Being an open source project, D3 is free to use for everyone, which puts it apart from similar data visualization JavaScript libraries such as amCharts [1] and FusionCharts [12], for which the user needs to buy a pricey license. Furthermore, its development is supported by a big online community and it has extensive documentation, which is also supplemented by the lengthy documentation of web standards like SVG. Moreover, multiple tutorials for D3 can be found online, including one by Scott Muray [8], and a website called Dashing D3.js [10].

D3 is very flexible, and has a great expressiveness, a user is able to create many forms through different routes. As it is a drawing library, and not just a charting library, it's functionally reaches beyond conventional data visualization. Moreover, D3 allows interaction and animation, as opposed to mere static visualizations.

Only one line of code is enough to start using D3, which is a strength compared to other data visualization kits, which often require a long installation procedure and require updating. The compatibility of D3 with web standards and the usage of their widespread syntax have the advantage that visualization can be shared and viewed without the need for additional plug-ins, such as Flash. This makes it different from previous libraries such as Flare, which required Flash to render [11]. Moreover, as it is based on JavaScript, it is compatible with browser's built-in debugger, which facilitates fast and easy debugging.

Although D3 offers many possibilities, it is said that the learning curve can be fairly steep [2]. This is partly explained because of the need for in depth knowledge of web standards, especially of SVG. Furthermore, the many helper modules require studying too. This relative weakness explains the popularity of libraries such as C3, a library for reusable charts build on top of D3, but with a simpler syntax [5].

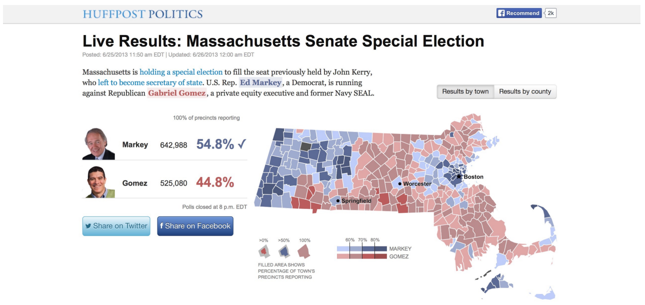
Another weakness of D3 is that it is less efficient than toolkits with custom graphs. Contrary to other JavaScript libraries such as Google Charts [14], D3 does not offer pre-built charts. In order to increase efficiency and ease of use, JavaScript libraries with simple charting options have been built on top of D3. An example is uvCharts, an open source JavaScript library including 12 customizable standard chart types [29]. In a situation where one wishes to create a standard graph, it might be more efficient to use a library such as uvCharts. But when you want to build graphs that are complex or unconventional in either visualization or interaction, D3 offers far more possibilities.

4. TYPICAL APPLICATIONS

4.1 DATA VISUALIZATION

The D3 website displays examples of data visualizations done with D3, that interactively run inside web browsers and are available for anyone who wants to visualize data [7]. Many types of examples are accompanied by a full documentation of how to manipulate documents based on data. Among all of them, it is worth citing the most used types of data visualization such as area chart, line chart, multi series line chart, bar chart, scatter-plot, donut chart and pie chart.

These types of data visualization can be used on a project basis given its facility to implement and manipulate information on the fly. Among those who use D3 in their projects, newspapers like The Guardian [13], The Huffington Post [4] and The New York Times [3] are the most famous ones to provide a good image of D3. They show that this powerful library can help ordinary users to to transform complex data into easy and clear information. D3 also allows information to be seen in real time, figure 2 shows that D3 can be constantly updated with data giving users a better understanding on how elections are happening.


HUFFPOST POLITICS

Live Results: Massachusetts Senate Special Election

Posted: 6/25/2013 11:50 am EDT | Updated: 6/26/2013 12:00 am EDT

Massachusetts is holding a special election to fill the seat previously held by John Kerry, who left to become secretary of state. U.S. Rep. Ed Markey, a Democrat, is running against Republican Gabriel Gomez, a private equity executive and former Navy SEAL.

100% of precincts reporting

Markey	642,988	54.8% ✓
Gomez	525,080	44.8%

Polls closed at 8 p.m. EDT

FILLED AREA SHOWS PERCENTAGE OF TOWN'S PRECINCTS REPORTING

Figure 2: The Huffington Post.

4.2 WIMBLEDON 2013 DATA VISUALIZATION

The use of D3 is not limited to digital newspapers, one can find thousands of examples spread on the web. One of the best examples we have found is a series of ten different data visualizations of the Wimbledon tennis tournament. The data charts made by Peter Cook, display information about the event played back in 2013 [30]. Among the the charts created was a circular match tree, a scatter-plot, a bar chart with negative values and a bubble chart with arrows.

The original data was collected from the British tennis data website and it is possible to see, among other things, how many matches were played by each player and who played who in the tournament. In the circular match tree (Figure 3), it is possible to see who played against who and what the result was of the match or round, by hovering your mouse over it. The winner is in the center of the circle.

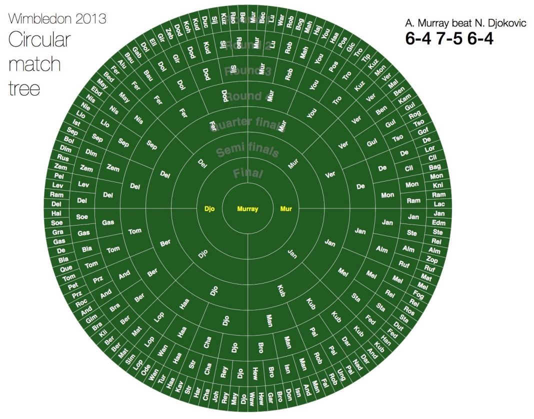


Figure 3: Wimbledon 2013: Circular Match Tree.

5. SURPRISING APPLICATIONS

5.1 ADOBE'S GRAPHICAL WEB EXPERIMENT

Because the scope of visualizations made with D3 is very wide, it appears hard to find surprising applications. However, an example of surprising usage of D3 is Adobe's Graphical Web Experiment, a website built specifically to show the possibilities of different technologies (Figure 4) [25]. For instance, CSS is often used to provide colors to websites, while SVG is capable of creating complex vector shapes that scale infinitely. Lastly, with D3 it is possible to animate and interpolate paths for the SVG characters as well as the curves of the terrain in the scenery of the website. All these technologies used together provide an interactive environment just like Adobe Flash used to do in the early years of the Internet. Although D3 can perfectly be used for this sort of application it was not originally intended as an animation technology for web browsers, therefore this might be considered a surprising application.

Figure 4: The Quest for the Graphical Web.

5.2 THE VIRTUAL BRAIN SIMULATOR

The Virtual Brain [26] is an international open-source project aiming to find out more about human brains by collecting data from real brains. Based on that data they represent thousands of virtual brains under thousands of different scenarios simultaneously (Figure 5). The application provides a platform for professionals such as doctors, scientists and nurses to work together and pave the way for further discoveries in neuroscience. The graphical user interface (GUI) is web based and makes use of Python, HTML5, CSS3, JavaScript, WebGL and D3 together with lots of sister-projects in the neuroscience community, to provide an intuitive and responsive interface that can be locally and remotely accessed [19]. The main system is accessible through a simple web browser, making it very easy to upload imaging data, running 3D-animated simulations in WebGL and D3 and getting results back. By doing so, the entire community can participate in hospitals, laboratories and clinics without having to host and run their own supercomputers.

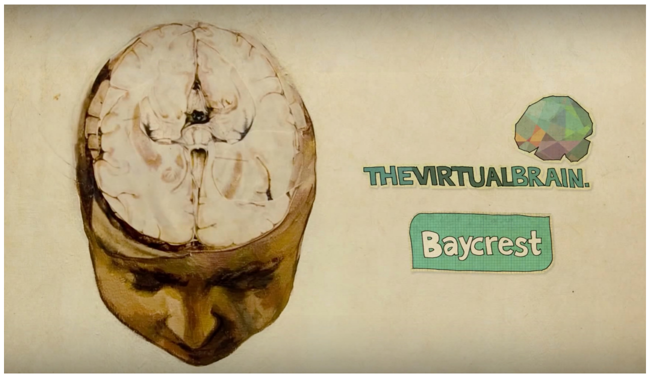


Figure 5: The Virtual Brain Simulator.

6. GETTING STARTED

Getting started with D3 is quick and easy. In this getting started guide we'll take a look at how to handle a simple dataset and make a simple visualization. After all, this is D3's main strength. We'll first discuss some prerequisites. You need to have basic knowledge of HTML, CSS, SVG and JavaScript. All you need to have installed is a text editor (Notepad suffices) and an internet browser. To access the D3 library one can either download a zip file from the D3 website or include a line of code in your file which links directly to the latest release. We start by creating a very basic HTML file and including the line of code which links to the D3 library in the header.

```
<html>
   <head>
      <script    src="https://
cdnjs.cloudflare.com/ajax/libs/
d3/3.5.5/d3.min.js" charset="utf-8">
      </script>
   </head>
   <body>
      <!-- body content here -->
   </body>
</html>
```

As D3 is javascript code it should be bookmarked by <script> tags. To test the examples in chapter 2 just copy the code to the body of the HTML file and put it between <script> tags. Now we will take a look at adding some data. Let's take a simple dataset: var data = [10, 3, 7, 7, 1]; and try to create a bar chart out of it. Before we can do anything with this data we must first bind it to the DOM. In our case it is logical to bind the data to the bars of the bar chart. The bars of the bar chart are nothing more than styled divs (div is short for division, an HTML element which in practice is nothing more than an empty rectangle) the style being defined in the head using CSS.

```
<html>
   <head>
     <script    src="https://
cdnjs.cloudflare.com/ajax/libs/
d3/3.5.5/d3.min.js" charset="utf-8"
      </script>
      <style>
         div.bar {
            width: 50px;
            margin-right: 5px;
            background-color: green;}
      </style>
   </head>
   <body>
      <script>
      var data = [10, 3, 7, 7, 1];
      d3.select("body")
      .selectAll("div")
      .data(data)
      .enter()
      .append("div")
      .attr("class", "bar")
      .style("height", function(d) {
            var barH = d*20;
            return barH + "px";});
      </script>
   </body>
</html>
```

We want to bind each point of data to a separate div, but we haven't created any divs yet. That is where the enter function comes into play. The enter function creates a div for each point of data in the body of the page. The .append and .attr functions make sure that the divs have the bar class we defined in the head, assigned to them. The .style function contains function(d), where d is a single point of data. This function calculates the height for each bar, as it has to correlate to the data point.

The result should look like this (Figure 6).

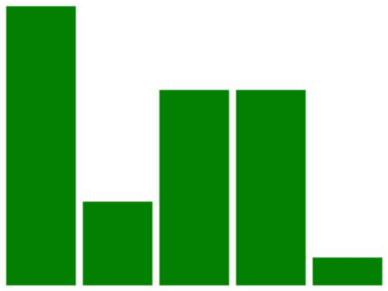


Figure 6: Getting Started with D3.

7. FINAL THOUGHTS

JavaScript has become the most important programming languages for the web in the last decade or so. This has caused a big shift in the direction of web development getting to the point where plug-ins are less and less needed for creating interactivity and action. Therefore, we foresee developers trying to use the power of the browser in itself at its best and avoid external plug-ins. This reduces the likelihood of bugs or incompatibility issues as well the need to update.

Additionally, developers can work with existing standards that are already being used on the web. This increases the compatibility of the D3 library with other popular technologies like HTML, CSS and JavaScript in general, making it even more easy to extend the work and making data visualization available to everyone. Finally, big data [27] is becoming the 'new oil' of the 20th as we all spend a large percentage of our lives online. We strongly believe that libraries like D3 will be frequently and creatively used by the community of web designers, web developers, computer scientists and computer engineers and shows the general direction web technologies are heading in.

8. REFERENCES

1. amCharts. http://www.amcharts.com/

2. Bostock, M.l., Ogievetsky V. and Heer J. D^3 data-driven documents. In Visualization and Computer Graphics, IEEE Transactions on (2011), 2301-2309.

3. Bostock. M & Carter, S. Over the Decades, How States Have Shifted. *Nytimes.com*. (Retrieved June 2015). http://www.nytimes.com/interactive/2012/10/15/us/politics/swing-history.html?_r=0

4. Bycoffe, A.& Boice, J. Massachusetts Senate Special Election: Live Results. *Huffingtonpost.com*. (Retrieved June 2015).

5. C3.
http://c3js.org/

6. CSS Introduction.
http://www.w3schools.com/css/css_intro.asp

7. D3 Gallery on GitHub
https://github.com/mbostock/d3/wiki/Gallery/

8. D3 Tutorials.
ttp://alignedleft.com/tutorials/d3

9. D3.
http://d3js.org/

10. Dashing D3 js.
https://www.dashingd3js.com/table-of-contents

11. Flare.
http://flare.prefuse.org/

12. FusionCharts.
http://www.fusioncharts.com/

13. Goldenberg, S. Alaska Villages Frontline Global Warming. *TheGuardian.co.uk*. (Retrieved June 2015).
http://www.theguardian.com/environment/interactive/2013/may/14/alaska-villages-frontline-global-warming

14. Google Charts.
https://developers.google.com/chart/

15. HTML Introduction.
http://www.w3schools.com/html/html_intro.asp

16. JavaScript Graphs and Charts libraries.
http://socialcompare.com/en/comparison/javascript-graphs-and-charts-libraries

17. JavaScript Tutorial.
http://www.w3schools.com/js/default.asp

18. jQuery.
https://jquery.com/

19. Knock, L.P.S, Woodman, S.A., Domide M.M., Mersmann, L. McIntosh J., Jirsa, V. The Virtual Brain: a simulator of primate brain network dynamics. Frontiers in neuroinformatics, 7: 10 (2013).

20. Michael., F. A brief history of data visualization. In Handbook of data visualization. Springer Berlin, Heidelberg, Germany, 2008. 15-56.

21. Prefuse.
http://prefuse.org/

22. Protovis.
http://mbostock.github.io/protovis/

23. SVG Tutorial.
http://www.w3schools.com/svg/

24. The Evolution of the Web.
http://www.evolutionoftheweb.com/

25. The Quest for the Graphical Web.
http://thegraphicalweb.com/

26. The Virtual Brain.
http://thevirtualbrain.org/tvb/zwei

27. Toonders, J., Data is the New Oil of the Digital Economy, *Wired.com* (Retrieved June, 2015)

28. Tufte, E. R., and Graves-Morris, G. R., The visual display of quantitative information. Graphics press, Cheshire, CT, USA 1983.

29. uvCharts.
http://imaginea.github.io/uvCharts/

30. Wimbledon 2013 Data Visualizations.
http://charts.animateddata.co.uk/tennis/