Showcase of d3.js and its possibilities in infographics using refugee data of the current Ukraine conflict

Luis Rothenhäusler (20202459)

Brandenburg, June 2, 2022

Abstract - German

Englische Arbeiten brauchen eine Zusammenfassung auf Deutsch. Mal abgesehen davon, dass wenn die Zusammenfassung interessant ist man ohne English eh nicht weiter kommt...

Contents

1. Introduction

Describe the background of the thesis, why it is important, what do we want to achieve. Why is this thesis? What do we want to do? What is the status quo? What benefits will result from this thesis? Something about the importance of infographics and comprehendible data.

The postmodern world produces huge amounts of data every second. Analyzing this data leads to better-informed decision-making in every sector. Yet the wast amounts of created data is often hard to comprehend with the human mind. Data visualization is about finding ways to represent this data in visually appealing, yet easily understandable visual representations. Doing this quickly and always up to date can be crucial. There are many tools available to help with the creation of infographics. Some of these tools have a graphical-user-interface, are programming based. This thesis will be a deep dive into the possibilities of one of these tools, the 'd3.js'(D3) library for JavaScript. To show its capabilities and potential D3 will be used to create a showcase containing several different graphics.

2. Basics

Everything necessary to understand the implementation as well as anything which is done beforehand, will come up here

In the following, all concepts, technologies and required backgrounds for understanding this thesis are explained.

2.1 Data

Well talk about data a bit. Where does it come from? How is it structured? What kind of attributes? What even are attributes?

Since ancient times, humans have recorded data. Recording the ins and outs of available resources was one of the driving factors behind the conceptualization of writing. (TODO: Check sources of the beer brewing video series) With the introduction of computers the amounts of gathered data have grown drastically. Vast amounts of data are gathered across all aspects of life.

2.1.1 Types of Data

Even though data comes from a huge variety of sources and can express a plethora of things, there are only four different types of data. They are split into equal pairs. Two types of categorical data and two types of numerical data. In the following each of the types of data will be explained.

Categorical

What is categorical data? Nominal and ordinal data

Categorical or qualitative data is information collected in groups. It is often of descriptive nature. Whilst the values can be represented in numbers, they do not allow for arithmetic operations. There are two types of categorical data. Nominal and ordinal data.

Nominal data is mostly descriptive in nature. They are independent and have no inherit order. Examples are ’Country’, ’Color’, ’Brand’.

Ordinal data is mostly similar to nominal data. Yet the data does have some sort of internal order. For example different dates each describe a day, but one day also comes after another.

Numeric

What is numeric data? Continuous and Discrete

Numeric or quantitative data is all data expressed in numbers, where numbers do not represent categories. It allows for arithmetical operations and can be split into Discrete and Continuous data.

Discrete data can only take certain defined values. This usually means whole numbers to represent things that can not be split up further. Like the ’Number of Refugees’ or ’Tickets sold’. Discrete data is countable.

Continuous data can be measured. It can have any real number as value. Therefore fractions are possible as well. For example when measuring the temperature, or the length or weight of an object.

2.1.2 Datasets

What are our datasets about? Where do they come from?

Datasets are a collection of several data-points. each of these data-points is made up of different attributes. Each attribute corresponds to a specific data-type.

In this thesis, two data-sets are used. They are both from UNHCR[1]. The first data-set contains information about the total number of refugees per country[2]. The second dataset is about the total cumulative total amount of refugees per day[3]. Both data-sets are in the JSON format.

Preprocessing

What is done in preprocessing? Python script which removes all excess / maybe do that in JS as well..?

As both of the used data-sets have a lot of ’unnecessary’ information, they are both preprocessed. In both cases the data-sets are read and the valuable information extracted and saved in the csv format. Both of the newly created csv files have two columns and one header row. The resulting csv of the refugees per country dataset, contains the two columns of country and refugees. The other resulting csv contains a column for the date and one for the cumulative refugees. The data-set about the refugees per country can also easily be converted into using percentages. After adding up the total amount of refugees from each data-point, one can convert the absolute number of refugees into percent.

Data Types

Which data types can be found in our data-sets? Where?

The two chosen data-sets already cover most of the data-types. The refugees per country dataset contains two attributes per data-point. The country is a categorical attribute. The number of refugees is discrete. When converting this data-set into using percentages, the percentage of refugees becomes a continuous attribute. The In the refugees per date data-set, the amount of refugees is still a discrete attribute. The date itself is an ordinal attribute though. As one day clearly comes before and after another day.

Choosing data-sets which cover all types of data-types was an important consideration. Different data-types can have different ways of representation, as well as different ways of implementation on the programming side of things.

2.1.3 Data Visualization

What is it? Where is it? Why is it important?

Data visualization is the process of turning data into graphical representations. As the vast amounts of data which are gathered in charts and databases are often hard to comprehend with the human mind and might require a big amounts of space to directly represent, it is often desired to turn these data-sets into a more easily understandable formats. Therefore data is turned into diagrams. We constantly come across the results of data visualization in everyday life. They can be commonly found across all kinds of news sources, but also in reports, information campaigns or as part of user-interfaces in machinery or control systems.

2.2 Diagrams

What diagrams exists? Which are the most common? What possibilities do they offer for encoding data? Which considerations for readability? Why do some diagrams not make as much sense? Which considerations where made for fulfilling the showcase requirements?

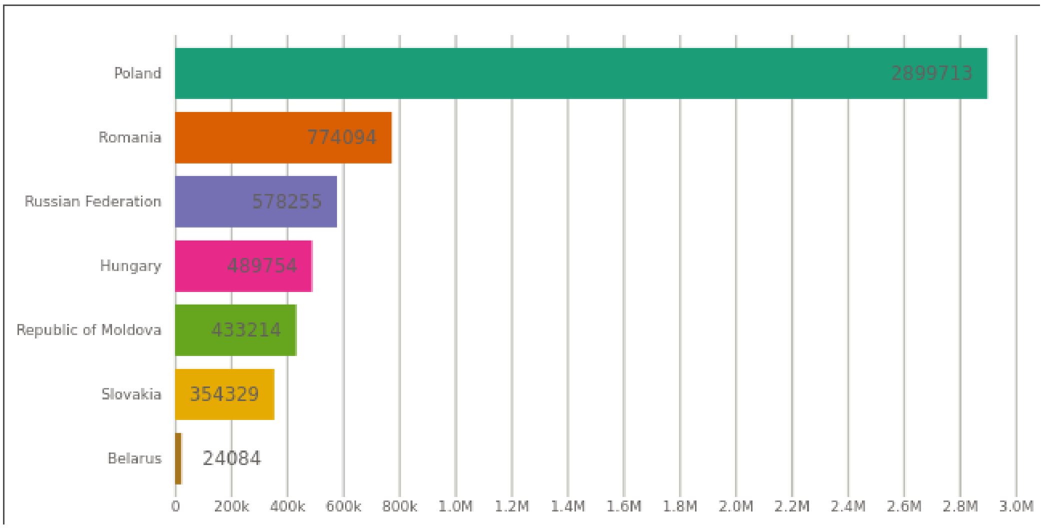

Figure 2.1: This is a bar-chart (TODO: which needs a frame..?)

Diagrams exists in many different shapes and sizes. There are many factors which should be taken into consideration when deciding which diagram to choose. There are usually many ways to represent the desired data. Some of the most common diagrams include bar-charts, pie-charts and scatter-plots (TODO: Find source (lel))

All diagrams use a combination of marks and channels to encode data. Marks usually depend on the type of data. They are usually single points, lines or areas in a diagram. Channels depend on the singular data-points and influence the marks. The most common channels are position, color and size. It is important to note, that not all channels work with all data-types.

To understand this more easily, we can look at diagram 2.2. This diagram uses lines as marks. The y-position, as well as the hue, are both used to encodes the same categorical attribute. The area/size encodes a discrete attribute. The donut chart(2.2) instead used areas as marks. Their size encodes a discrete, whilst the hue encodes a categorical attribute.

Different combinations of marks and channels can influence the readability and the correctness of how the represented data is comprehended [4] [5].

2.3 D3.js

This is all about d3. What is it? Where does it come from? What is it used for? Who uses it? Why should it be used? How does it work? Enter, update and exit pattern. Something about the modular structure of D3 as

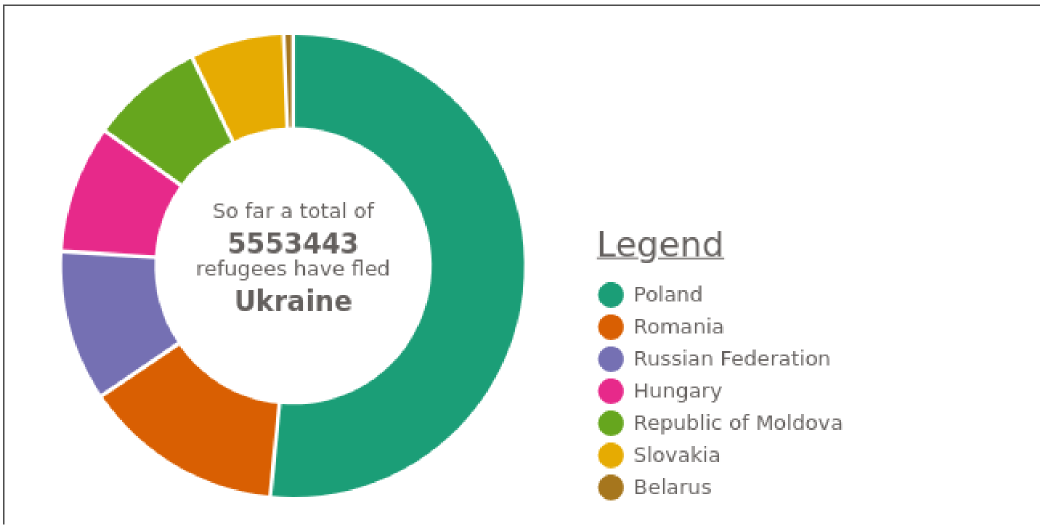

Figure 2.2: This is a donut-chart (TODO: which needs a frame..?)

well. Might be worth mentioning ”observables” as well.

”D3.js is a JavaScript library for manipulating documents based on data. D3 helps you bring data to life using HTML, SVG, and CSS.”[6]. The name D3 is short for data-driven documents. The D3 library was originally created by Mike Bostock and is published under the BSD-3-Clause open-source license.

2.3.1 How does it work?

General functioning of D3.

”D3 allows you to bind arbitrary data to a Document Object Model (DOM), and then apply data-driven transformations to the document.”[6]. There are three main concepts that make up the core of D3. Selections, data joins and the general update pattern. Importing the d3 library into a project allows access to the `d3` namespace.

Selections

What are they? Why are they useful?

All operations in D3 run on an arbitrary collection of nodes these collections of nodes are called selections. There are two functions in D3 to create a new selection: `d3.select("selector")` and `d3.selectAll("selector")`. Both functions require a selector for identifying the appropriate elements. The selectors are defined in the W3C Selectors API[7]. It is possible to directly access the DOM elements as the selections are only collections of nodes. But there are also many predefined functions for modifying the nodes properties. This includes the modification of attributes and styles, as well as event handling. Selections can also be extended or shrunken by adding or removing nodes, or by combining multiple selections.

Data Joins

What are they and why are they important?

Data joins are a key feature of D3. They link up a data-point to a specific DOM element. To create a data-join, one has to call the `.data(dataset)` function on a selection. It takes a dataset, an array of data-points, as parameter. This will bind the data elements to the nodes in the selection. When we want to create diagrams which can respond to data changes over time, it is usually also a good idea to specify a identifier function for the data-points. As the default identifier corresponds to the index of the data-point. This can lead to unexpected behavior when the dataset is modified. It is not important for the number of data-points to match up with the number of nodes in the selection. This is taken care of through the general update pattern.

General Update Pattern

What is it? What can it do? Describe data joins and dom element links.

The general update pattern is another core concept of D3. Every time a data join is created or updated, it comes into play. There are three possible cases that can happen, when a data join is invoked. The first case, which usually happens as the data join is first created, is, that there are more data-points than nodes. In this case, all the data-points without a linked node are called the enter selection. The second case is about data-points which already have a linked DOM element. They make up the update selection. The last case , where a data-point which was already linked to a DOM element has been removed, is called the exit selection. For each three of the resulting sub-selections, we can define the behavior. For the entry selection, we usually want to create a new DOM element to represent the data-point. When the goal is to create only static representations, it is even enough to only implement the entry functionality. If we want to allow for data to change over time, we probably also want to implement the update and exit selections behaviors.

Modules

The way D3 is split up into modules, the core package and what kind of extensions are there.

3. Implementation

How are the chosen diagrams implemented? Which D3 modules have been used? How was the implementation done?

3.1 Showcase

How is the showcase structured? How can you get there? Why does it exist? Who might benefit? How can you reuse a part the interesting parts?

3.1.1 Integration of each diagram

How is each diagram integrated? How can you access them? Where can you grab them standalone?

3.1.2 Data Updates

How can you simulate data changes? Why is this useful?

3.2 Diagrams

describe all the diagrams and why they are special and what makes them tick

3.2.1 Bar Chart

How does it work? Which d3 features does it use? how do they work?

Pie chart

How does it work? Which d3 features does it use? how do they work?

Tree map

How does it work? Which d3 features does it use? how do they work?

Sankey

How does it work? Which d3 features does it use? how do they work?

Area graph

How does it work? Which d3 features does it use? how do they work?

Circle graph

How does it work? Which d3 features does it use? how do they work?

4. Conclusion

How well did it work? Was it worth the effort? What could be improved?

Bibliography

[1] UNHCR, “Operational data portal,” accessed:12.05.2022. [Online]. Available: https://data2.unhcr.org/en/situations/ukraine

[2] ——, “Refugees per country,” accessed:12.05.2022. [Online]. Available: https://data.unhcr.org/population/get/sublocation?widget_id=312121&sv_id=54&populatio
01-01

[3] ——, “Refugees per day,” accessed:12.05.2022. [Online]. Available: https://data.unhcr.org/population/get/timeseries?widget_id=312123&sv_id=54&population_
01-01

[4] J. Heer and M. Bostock, “Crowdsourcing graphical perception: using mechanical turk to assess visualization design,” in *Proceedings of the SIGCHI conference on human factors in computing systems*, 2010, pp. 203–212.

[5] J. Mackinlay, “Automating the design of graphical presentations of relational information,” *Acm Transactions On Graphics (Tog)*, vol. 5, no. 2, pp. 110–141, 1986.

[6] M. Bostock, “Data-driven documents,” accessed:31.03.2022. [Online]. Available: https://d3js.org/

[7] A. v. Kesteren and L. Hunt. [Online]. Available: https://www.w3.org/TR/selectors-api/

5. Appendix

I guess this should contain all the source code. Maybe there are ways to import it automatically too?