Accessible Visualisation

Group 1

Dominik Bramreiter, Markus Ebner, Philipp Leitner, and Christoph Lipautz

Institute of Interactive Systems and Data Science (ISDS),
Graz University of Technology
A-8010 Graz, Austria

19 May 2017

Abstract

The document at hand gives an overview of strategies and techniques in order to provide content of information visualisation to visual impaired people.

Providing accessible content is an everyday challenge to professions like web-, mobile-, and content developers. With the release of the recommendation of WAI-ARIA in 2014, many issues with handling graphical content received a recommended way to present. On specific detail with info graphics, data charts and SVG, this survey discuss common problems, challenges and best practices in order to ensure graphical described content is accessible to visual impaired.

Further tools for checking and automated testing of informational visualizations are listed and described.

Contents

List of Figures

List of Tables

List of Listings

Chapter 1

Motivation

Providing accessible content is an ongoing challenge for developers and content maintainers. Offering an accessible design on your web pages enables people with disabilities to understand, navigate, and interact with your content. Web developers have to understand that accessibility is not a barrier which makes your product ugly, cluttered or boring, it makes it innovative and open to a wider range of people worldwide.

Understanding the constraints of disabilities is the first step. Any graphical content can be inaccessible in various aspects. A user interface to interact with a data chart can have difficulties for a user with cognitive disabilities, or a reader that is limited to keyboard navigation only for ongoing disability or a temporal like a broken arm.

To enhance chances for people with disabilities the World Wide Web Consortium (W3C) released Web Content Accessibility Guidelines (WCAG), in the latest version 2.0. This set of constraints have been produced as a port of the W3C Web Accessibility Initiative (WAI). The document covers a wide range of recommendations for making content on the web accessible for people with disabilities. The latest version of the document has been reviewed by W3C Members, software developers and other interested parties. [WAI, 2008] A overall good introduction to web accessibility is provided by the Web Accessibility In Mind (WebAIM), a non-profit organization who is one of the leading providers of web accessibility expertise.

In this chapter we will provide an introduction to Web Accessibility in general, its principles and standards available on the web as well as starting points for information about accessibility provided by organizations. Although, accessibility is not limited to visual impaired, the following chapters focus on two disabilities: color blindness and total blindness. Further, examples will be shown to give a first impression on the importance and implementation of accessible design.

1.1 How to Implement Web Accessibility?

Every designer and developer can make the Web content accessible, but before they must understand accessibility and be committed to it, as a person and/or a company. Further, you have to learn how to make content accessible and what tools are available on the market. In this chapter we present some tips and visions provided by WebAIM.

1.1.1 Commitment to Accessible Design

According to [WebAIM, 2016a], commitment to accessible design has three main points:

Awareness: Most web developers are not aware of this issue and therefore many web sites have accessibility problems. Being committed to accessible design is the first step.

Leadership: Not only the web developers have to understand the issue. Also the leadership of companies and organizations need to show their commitment to web accessibility.

Policies and Procedures: When both mentioned parties are fond with the idea policies and procedures are needed so that it get not lost in the daily routine. Creating an internal policy may be the best way of ensure web accessibility in the future.

1.1.2 Technical Knowledge

Web developers sometimes think that providing accessible web content is more expensive and time-consuming, which is not true. The basis can be learned after a view days. Available online resources are providing enough material on this topic. [WebAIM, 2016a]

1.1.3 Law

Many international laws are addressing accessibility. In the USA this is regulated by the The Americans with Disabilities Act (ADA) and the Rehabilitation Act of 1973 (Sections 504 and Section 508). [WebAIM, 2016a]

The European Union (EU) took actions against the “Digital Divide”, so that people with disabilities are no longer in danger of being excluded from the society or discriminated. The EU supports the Web Accessibility Initiative principle (WAI-principle) from Worldwide Web Consortium (W3C). In Austria this is defined in the “Behindertengleichstellungsgesetz” and “E-Government-Gesetz”. [Bundeskanzleramt, 2017]

1.1.4 Standards

The Web Content Accessibility Guidelines (WCAG) are providing guidelines, developed by the W3C, and are based on four principles [WebAIM, 2016a]:

Perceivable: Think about how people can consume the provided web content. The most relevant senses are sight, hearing, and touch.

Operable: Not everybody is using the keyboard or the mouse to navigate. Alternative devices are used by people with disabilities.

Understandable: Provide clear content, easy language and functionality.

Robust: Keep in mind that different technologies can access the content. Use technologies according to their specifications to ensure operability (e.g. to support screen readers).

1.2 Principles of Accessible Design

In this chapter a list of key principles for accessible design are provided. Although, not all accessibility issues are mentioned, you can provide an overall better accessible website to your audience by addressing this points. Most of them are easy to use and integrate in the website without changing the overall look of the site [WebAIM, 2016b]:

Provide appropriate alternative text: Especially helpful for people who rely on assistive technology such as screen readers.

Provide appropriate document structure: Stick to the web standards and structure of web pages as well as guidelines for accessible visualization. Also include keyboard navigation within the page.

Provide headers for data tables: Associate data cells with the appropriate header for making the easier to understand for screen readers and easy to navigate.

Ensure users can complete and submit all forms: Provide labels and assure that they are correctly associated with the fields. Also think about how to present errors in the form to the users.

Ensure links make sense out of context: Avoid using “click here” or “more”. Use better descriptions. If e.g. a screen reader is only reading the links of the page the user will be confused with this type of link description.

Caption and/or provide transcripts for media: Add captions and transcripts to them to make them accessible.

Ensure accessibility of non-HTML content, like PDF files, MS Word documents, and PowerPoint presentations: Make them as assessable as possible, like the rest of the page. If it is not possible, think about providing a html version or try to make them accessible.

Allow users to skip repetitive elements on the page: Provide a possibility to skip the navigation and jump to the main content. You may use links with appropriate description for it.

Do not rely on color alone to convey meaning: This information may be inaccessible for color blind people or unavailable for blind people who are using the screen reader.

Make sure content is clearly written and easy to read: A clear structure, simple text and appropriate headings and description makes the life for every user easier.

Design to standards: Use Cascading Style Sheets (CSS) to separate the content from its presentation. A more robust and accessible page is a good way for search engine optimization.

Chapter 2

Visual Impairment and Blindness

An estimation of the World Health Organization (WHO) suggests that over 285 million people worldwide are affected by visual impairment. Over 39 million are blind and over 246 million have low vision [WHO, 2014]. Although, the numbers may be even higher than expected because the WHO estimation is based on best-corrected visual acuity [L. Dandona and R. Dandona, 2006]. The visual impairment compromises the people in the ability to effectively perform color and visualization-related tasks. The key problem is that the colors most people see as different will look the same [Machado et al., 2009].

2.1 Blindness

Over 39 million people worldwide are blind [WHO, 2014]. Although, some of them have a degree of vision, blind people do not use their eyes to access content on the web. A monitor and a mouse are not much of use for them. As mentioned in the previous chapter, as a web developer you have to provide accessible design to make the web access possible for people who are blind and support the assistive technologies they are using. [WebAIM, 2013a]

2.1.1 Assistive Technology for Blind

Screen Readers: Text is converted into synthesized speech. With this Text-To-Speech engine blind people can listen to the provided web content. Only a robust and well structured web page can provide easy access to the content. Further, a screen reader is also able to support deaf and blind people by converting text into Braile characters. Those people then can feel the web content. [WebAIM, 2013a] The Stanford Online Accessibility Program offers Tools for Screen Reader Testing on their web-page: `https://soap.stanford.edu/tips-and-tools/screen-reader-testing`. There are also some limitations to mention with screen readers [WebAIM, 2013a]:

- Image: Describing an image is not possible. Therefore a text must be provided as a substitute for that image. If there is no alt text for the image it is not possible for the screen reader to convey the meaning of the image.
- Layout: A visual user can quickly look over the web page and realize its structure. A screen reader can not perform that task. Extraneous content such as advertisements can not be skipped intelligently.

Haptic Interaction: An enhancement which allows users "to touch" and feel the simulated objects they interact with. Those devices are stimulating the senses of touch such as the sensory capabilities within your hand. A haptic interface is a force-reflecting device with the power to let the user touch, feel, and manipulate objects in a specific environment. [Chimurkar and Bagdi, 2014] By the aid of the haptic interface the user can perform tasks which are normally performed by using hands in the real world. Appropriate tactual images are presented to the user. [Srinivasan, 1995]

ethnic group	male	female
Caucasians	7.9%	0.42%
Asians	4.2%	0.58%
Africans	2.6%	0.54%

Table 2.1: CVD among different ethnic groups worldwide, based on [Machado et al., 2009], [Rigden, 1999], [Gegenfurtner and Sharpe, 1999, page 3-51].

2.2 Color Blindness

A normal human color vision is called normal trichromacy and requires three retinal photoreceptors which are called L, M, and S cones. Together they form the LMS color space of the human eye, which is named after the sensitivity of long (L), medium (M) and short (S) wavelengths. If the sensitivity is shifted to a different band of spectrum, the condition is called anomalous trichromacy [Machado et al., 2009]. It is further classified in three types [Machado et al., 2009]:

- protanomaly - reduced sensitivity to red light (common),
- deuteranomaly - reduced sensitivity to green light (most common), and
- tritanomaly - reduced sensitivity to blue light (rare)

Total color blindness is called arachomatopsia and is very rare. The most common case of Color Vision Deficiency (CVD) involves the L and M cones and is known as red-green CVD. The male population has a significantly higher chance of being affected than the female population, as shown in table 2.1

The private and professional lives of people which are influenced by e.g. red-green CVD is more difficult if they have to perform tasks which are color and visualization-related [Machado et al., 2009]. Understanding this has a significant practical importance in developing accessible visualizations.

In case of red-green CVD the color is often indistinguishable. In general this is not a problem. Although, if the color is used to convey information some changes have to be made, otherwise the information is indistinguishable. In this cases, to make the information accessible, you have to change it or provide additional means, e.g. provide an explanation in the text itself. [WebAIM, 2013b]

2.2.1 Examples of CVDs

Two examples of different CVDs are shown: Figure 2.1 shows *The Starry Night* and Figure 2.2 the *Walk in the moonlight* from the artist Van Gogh (both are Public Domain). The top left corner shows the original version of the picture. Top right the vision if you have protanopia, bottom left deuteranopia and bottom right tritanopia.

2.2.2 CVD - Testing

Figure 2.3 shows a Ishihara color test plate. The number 74 is seen by people with normal vision, 21 by many color blind people. People with total color blindness may not see any number on the color test plate.

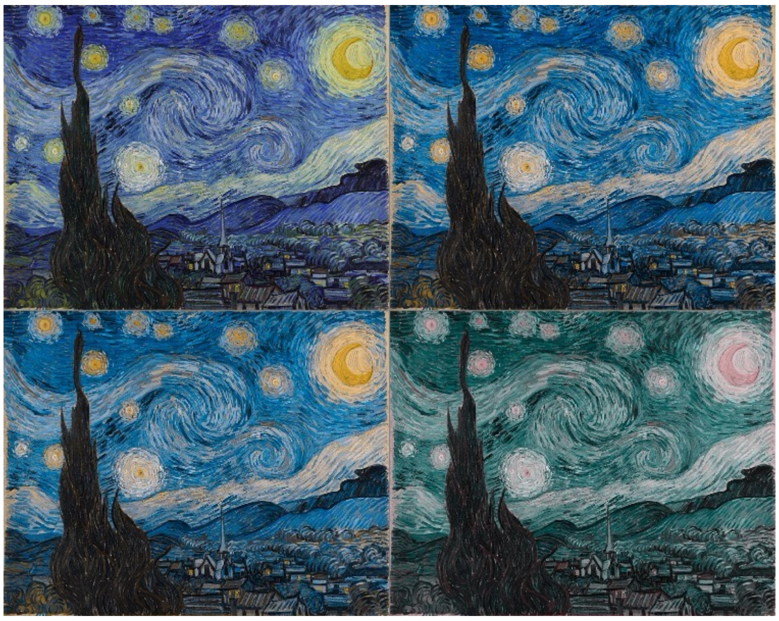


Figure 2.1: Van Gogh - Starry Night. Vision from top left to bottom right: original, protanopia, deuteranopia, tritanopia. Original image from van Gogh [1889] in the public domain, created with Chromatic Vision Simulator by Asada [2013].

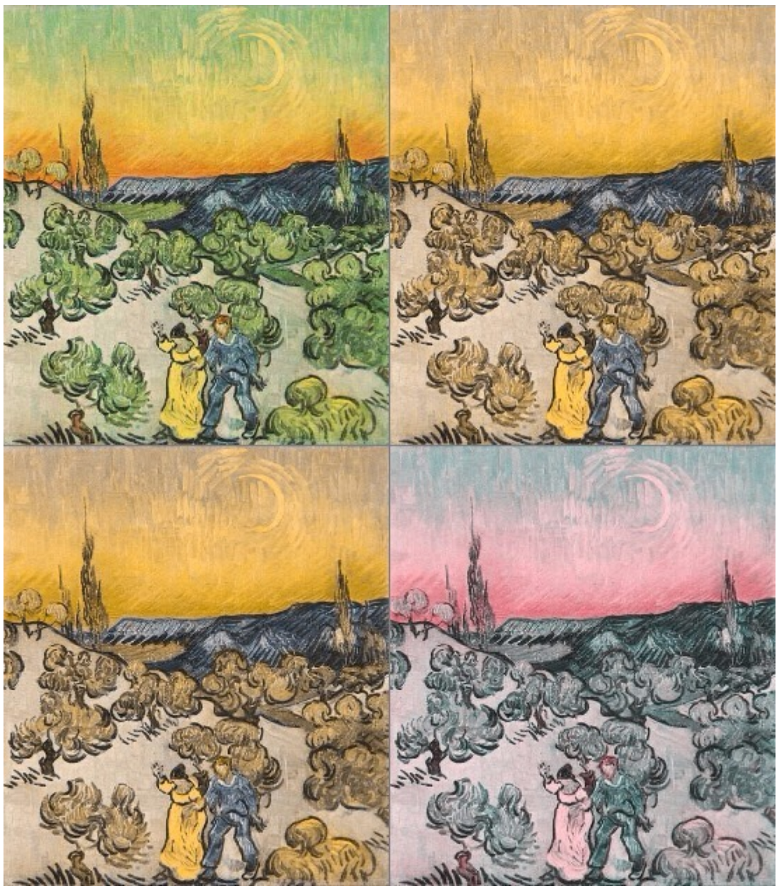


Figure 2.2: Van Gogh - Walk in the moonlight. Vision from top left to bottom right: original, protanopia, deuteranopia, tritanopia. Original image from van Gogh [1890] in the public domain, created with Chromatic Vision Simulator by Asada [2013].

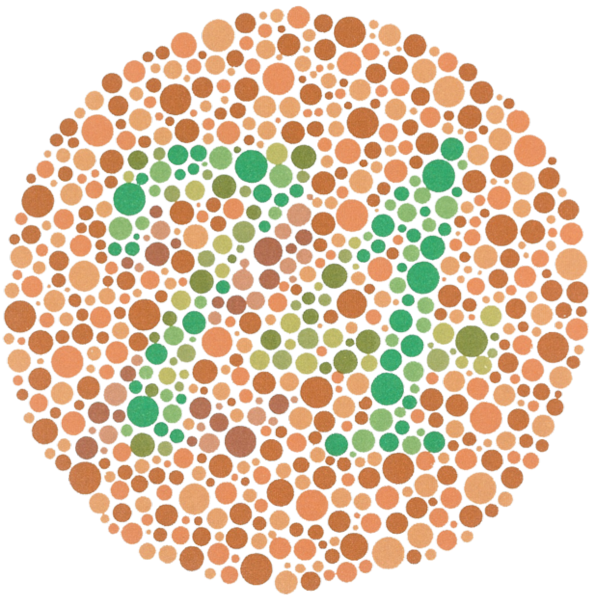


Figure 2.3: Example of an Ishihara color test plate. Image from Wikimedia Commons [2010] in the public domain.

Chapter 3

Accessible Infographics and Data Charts

The colors in an infographic or data chart are an important factor for readability and accessibility for people. The differences between the bars or lines in a data chart should also be accessible if the reader is color blind or people use different color settings for their screens. Therefore, it is important that the colors and lines differ enough from each other. To make these differences also accessible if the people can not see the colors, or the graphic is printed in only black and white, it is important to make sure that the intensity between the different colors or the darkness between the different parts are still accessible. For Example, different styles for the lines, or different symbols for the points help making charts accessible.

Here are some handy rules to avoid color blindness pitfalls according to Bigmann [2013]:

Avoid color combinations which are hard for color blind people: E.g. Green & Red; Green & Brown; Blue & Purple;

Make it monochrome: Using various shades of a single color instead of multiple colors

Use high contrast: CVD people can still perceive contrast, as well as differences in hue, saturation and brightness.

Use thicker lines: Some mildly color blind people are able to see a color, but only if there is a sufficient "mass" of it.

Do not assume colors will signal emotions in and of themselves: If you are using red to signal "bad, warning", or "watch out", consider adding another symbolic element to get the point across to color blind viewers.

Use texture instead: Especially in maps and infographics

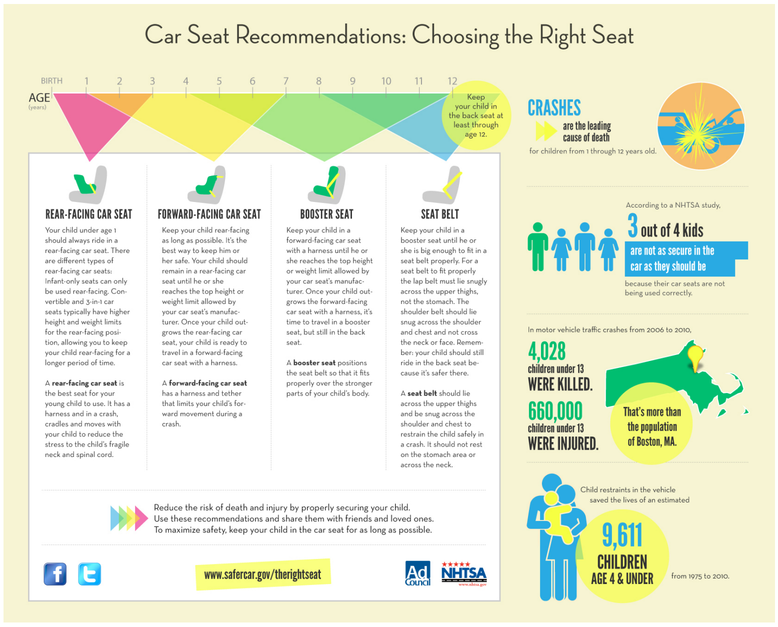
3.1 Infographic

In Infographics and also Maps the color is often used to differ between the information. It is crucial to the appeal and often carries symbolic information.

Infographics are a good choice if you want to deliver a message to everyone. Around 80% of the daily information is transmitted by the brain visually [Morton, 2010]. Most of the people do not read a text without an appropriate image which presents the information in an easy way. If you want to create an infographic keep in mind that it is not easy to make. It will cost a lot of time to get from the idea over research to the story and the appropriate design. But the success speaks for itself, it can go viral and gain attention on social media channels like twitter or Facebook [Wilkinson et al., 2015].

3.1.1 Example Infographic

Figure 3.1 shows an infographic example about car seat recommendations. The top shows the different age domains and the appropriate car seat for that age. Further information is provided in the different areas: Age 0 to 3: rear-facing car seat, age 1 to 7: forward-facing car seat, age 4 to 12: booster seat and age 8 to 13: you can use the seat belt, if appropriate.



Car Seat Recommendations: Choosing the Right Seat

Keep your child in the back seat at least through age 12.

REAR-FACING CAR SEAT

Your child under age 1 should always ride in a rear-facing car seat. There are different types of rear-facing car seats: Infant-only seats can only be used rear-facing. Convertible and 3-in-1 car seats typically have higher height and weight limits for the rear-facing position, allowing you to keep your child rear-facing for a longer period of time.

A **rear-facing car seat** is the best seat for your young child to use. It has a harness and in a crash, cradles and moves with your child to reduce the stress to the child's fragile neck and spinal cord.

FORWARD-FACING CAR SEAT

Keep your child rear-facing as long as possible. It's the best way to keep him or her safe. Your child should remain in a rear-facing car seat until he or she reaches the top height or weight limit allowed by your car seat's manufacturer. Once your child outgrows the rear-facing car seat, your child is ready to travel in a forward-facing car seat with a harness.

A **forward-facing car seat** has a harness and tether that limits your child's forward movement during a crash.

BOOSTER SEAT

Keep your child in a forward-facing car seat with a harness until he or she reaches the top height or weight limit allowed by your car seat's manufacturer. Once your child outgrows the forward-facing car seat with a harness, it's time to travel in a booster seat, but still in the back seat.

A **booster seat** positions the seat belt so that it fits properly over the stronger parts of your child's body.

SEAT BELT

Keep your child in a booster seat until he or she is big enough to fit in a seat belt properly. For a seat belt to fit properly the lap belt must lie snugly across the upper thighs, not the stomach. The shoulder belt should lie snug across the shoulder and chest and not cross the neck or face. Remember: your child should still ride in the back seat because it's safer there.

A **seat belt** should lie across the upper thighs and be snug across the shoulder and chest to restrain the child safely in a crash. It should not rest on the stomach area or across the neck.

Reduce the risk of death and injury by properly securing your child.
Use these recommendations and share them with friends and loved ones.
To maximize safety, keep your child in the car seat for as long as possible.

CRASHES are the leading cause of death for children from 1 through 12 years old.

According to a NHTSA study, 3 out of 4 kids are not as secure in the car as they should be because their car seats are not being used correctly.

In motor vehicle traffic crashes from 2006 to 2010,

4,028 children under 13 WERE KILLED.

660,000 children under 13 WERE INJURED.

That's more than the population of Boston, MA.

Child restraints in the vehicle saved the lives of an estimated 9,611 CHILDREN AGE 4 & UNDER from 1975 to 2010.

Figure 3.1: Infographic - car seat recommendations: Choosing the right seat. Image from National Highway Traffic Safety Administration [2012] in the public domain.

3.1.2 Infographics and Accessibility

An infographics is more or less an image. The same way as you provide alternative text to an image you can provide alternative text to an infographic. So the complex information can be consumed by people with disabilities. There are 2 methods of making them accessible, with a text alternative or with HTML/CSS [Fernandez, 2012]:

Text alternative: When you do so please consider the following points:

- Purpose: Find the parts of the infographic which needs to be explained in text form.
- Hierarchy: Group the alternative text into logical parts. Follow the order of the visual representations of the infographics.
- Meaningful images: Make sure that the context provided by visual cues is included.

HTML/CSS: You can use HTML and CSS for creating an infographic instead of an image file. Using this approach you can provide real text instead of alt-text for the image. An example of such an approach can be found in the blog from Throup [2012].

With the second method the text information is created in HTML, styling is made via CSS. This makes the infographic itself accessible. This method also has the benefit of working well together with an screen reader. Unfortunately, the second method is not very common. It is easier to create an infographic with a graphics package than using HTML/CSS due to the higher workload by paying attention to the accessibility requirements such as reading order, headings and so on. [Fernandez, 2012]

3.1.3 Example Map

This examples shows a metro map of Tokyo. Figure 3.2 shows a color rich version. You can find the metro lines or city districts and the directions quite easily. Figure 3.3 shows a greyscale version of the same map to get an impression on how difficult it is for people with color vision deficiency to differ between the lines.

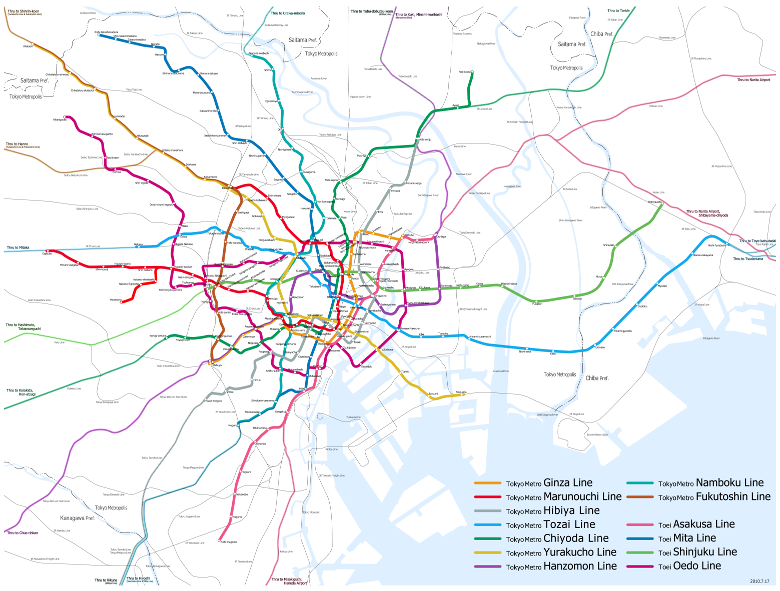


Figure 3.2: Map of Tokyo Metro lines and Toei Subway lines in color. Image from Hisagi [2010] under the (CC BY-SA 3.0).

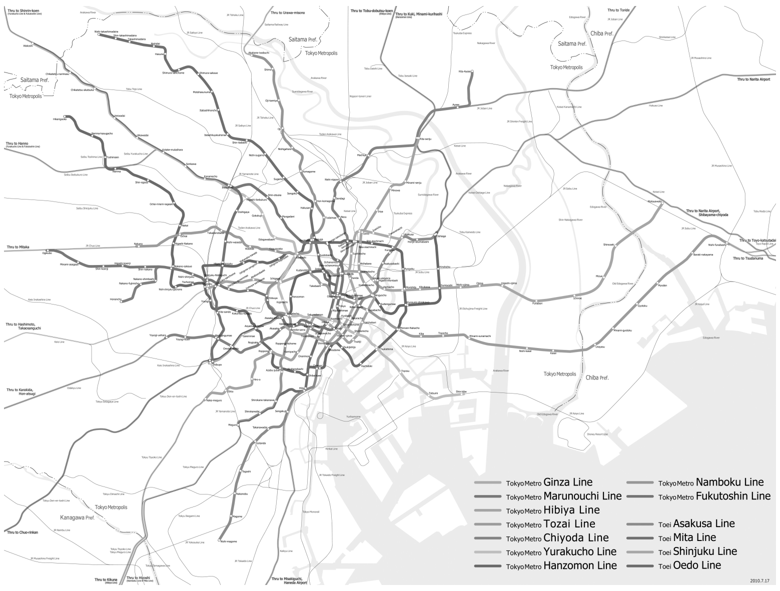


Figure 3.3: Map of Tokyo Metro lines and Toei Subway lines in greyscale. Image from Hisagi [2010] under the (CC BY-SA 3.0). Grayscale created with Adobe Photosthop CS5.

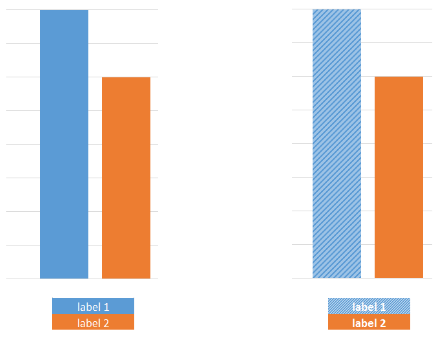


Figure 3.4: Bar chart with colors, left: common version (bad), right: with texture (better). Created by the authors using Microsoft Excel 2013.

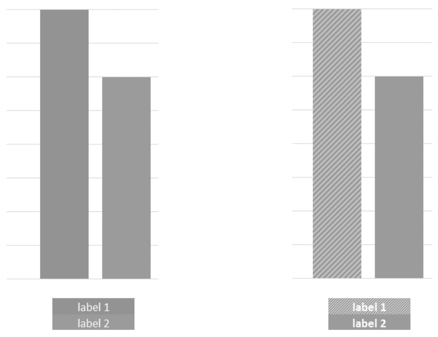


Figure 3.5: Bar chart in greyscale, left: common version (very bad), right: with texture (better). Created by the authors using Microsoft Excel 2013. Grayscale created with Adobe Photosthop CS5.

3.2 Data Charts

For screen readers charts are only images and they can not read them without extra information properly. To make the information accessible for screen readers they need a description ("@longdesc") or alternative text, which describes the data chart. This description text should tell in some lines what the data chart is about and what it should show to the user. Often, it is also helpful to give the data that is visualized by the image as table.

Scalable Vector Graphics (SVG) and Accessible Rich Internet Applications (ARIA), both allow to use titles and descriptions within the images and graphics. Thanks to these description tags, which are readable for screen readers the authors of the charts can directly link describing information to the chart elements. When the screen reader goes over the SVG it can read this information and explain the graphic to the user.

3.2.1 Example - Texture

This examples show how easy and beneficial it is to use texture in graphs. Figure 3.4 shows a common used graph on the left and a version with texture on the right. Figure 3.5 is the same as the first, but in greyscale vision. Here you can see the difference and how important it is to provide accessibility in graphs.

Chapter 4

WAI-ARIA Accessible SVG

In March 2014 the Web Accessibility Initiative (WAI) a working group of the World Wide Web Consortium (W3C) released the version 1.0 of the Accessible Rich Internet Applications (ARIA) Suite. WAI-ARIA is a W3C recommended web standard. The suite consists of extensions to HTML via attributes in order to extend information on a content given to improve accessibility for user agents and Assistive Technologies (AT).

4.1 Using SVG in HTML Documents

A Scalable Vector Graphics (SVG) is a XML based graphical content description. It can be used in HTML documents in several ways: Using it directly inside an HTML `img` tag, using it inside a CSS `background-image` style definition or embedding the SVG directly into the document namely used as inline svg. Whereas the first two methods mentioned serve as design content, any graphical content that stores information should be implemented as inline svg only. The `img` and `background-image` are treated as one single graphical content, as the name suggests, rather than structured meta content.

SVG images have several benefits over other HTML image formats like PNG, GIF, JPEG, or canvas elements. As the name suggests, a SVG element is scalable and therefore resolution independent. Further the single components of an image are object elements themself and can be directly addressed in the Document Object Model (DOM) of a HTML document.

4.2 WAI-ARIA Graphics Module

The current version of the WAI-ARIA Graphics Module is with September 2016 not yet in a final release but a working draft. The following short list gives an overview of WAI-ARIA attributes that can be used to improve accessibility of SVG content. A complete list can be found in WAI-ARIA specifications and within the graphics module attributes and usage may change to a findal release.

role: Defines the purpose of a given object element on a document node level. Note that the role definition is, in contrary to any other aria attributes, not prefixed by `aria-`. In a graph common values are used like chart, heading, dataset, datapoint, datavalue, axislabel, yaxis and xaxis.

aria-roledescription: Provides a more specific description of the chart itself or nested group structure.

aria-labelledby: Relates to an HTML identifier `id` inside the same document that contains a label to the given element.

aria-describedby: Analog to the `aria-labelled` attribute, a description is referenced.

aria-minvalue, aria-maxvalue: Define the minimum and maximum value of a data range.

aria-live: Inform the user agent if the chart is going to be updated.

aria-haspopup: Tells if any tooltips or other hidden contents may appear within interaction.

aria-datatype: Can be used to describe a given type of data, usually used at chart axes to provide information on the scale. Examples of valid types are: count, ordinal, datetime, duration, labels, and others.

aria-charttype: Used in describler and suggested by Doug Schepers but not currently documented and implemented in WAI-ARIA [D. Schepers, 2015].

4.3 An Example Improvement of an SVG

Listing 4.1 shows an example image of a line chart representing some currency rates of EURO to Britisch pounds from June 2016. The sourcecode of the graphic is handwritten, no image processing software was used.

The following section will go through the original sourcecode, explain structure and elements, and provide a step by step tutorial on how to improve the given graphic.

After the initial SVG definition, it provides some CSS styling rules to colorize lines, define the stroke width to be used and determines the font to be used for the labeling text elements.

A text element provides a title to be statically placed in the upper right corner of the graphic. The main information of the visualisation, the given data, is represented by a single polyline that stops at recalculated values in scale to the viewbox.

The following objects are used to draw axes including a scale and attaching corresponding labels to them. Figure 4.1 shows how the vector graphic is displayed when rendered with a user agent or any image viewer supporting SVG. Using an assistive technology like a screen reader on the current state of the graphic there is no output on this state of the image at all. Not only the image has no narrative text except the text element that is not recognized as title or description, it even cannot be focused when for example an user navigates a document with the keyboard.

```
<svg
  version="1.1"
  baseProfile="full"
  xmlns="http://www.w3.org/2000/svg"
  xmlns:xlink="http://www.w3.org/1999/xlink"
  viewBox="0 0 125 125"
>
  <!--
  With a scale between 1.21 and 1.31:
    2016-06-22, 1.3056 - results to 0.965%, inverted without offset: 4.4
    2016-06-23, 1.3091 - results to 0.990%, inverted without offset: 0.1
    2016-06-24, 1.2319 - results to 0.219%, inverted without offset: 78.1
    2016-06-25, 1.2319 - results to 0.219%, inverted without offset: 78.1
    2016-06-26, 1.2162 - results to 0.062%, inverted without offset: 93.8
  -->
  <g>
    <defs>
      <style type="text/css">
        <![CDATA[
        .chart-title { fill: black; font-family: sans-serif; font-size: 28%; }
        .chart-label { fill: black; font-family: sans-serif; font-size: 18%; }
        .chart-data  { fill: none; stroke: blue;  stroke-width: 1; }
        .axes        { fill: none; stroke: black; stroke-width: 1; }
        ]]>
      </style>
    </defs>
    <text x="60" y="10" class="chart-title">GB to EUR rates 2016-06</text>

    <polyline points="31.66,9.4 48.33,5.1 65,83.1 81.66,83.1 98.33,98.8"
      class="chart-data" />

    <polyline points="15,4.5 15,105 115.5,105" class="axes" />

    <line x1="14.5" y1="5"   x2="12.5" y2="5"   class="axes" />
    <line x1="14.5" y1="30"  x2="12.5" y2="30"  class="axes" />
    <line x1="14.5" y1="55"  x2="12.5" y2="55"  class="axes" />
    <line x1="14.5" y1="80"  x2="12.5" y2="80"  class="axes" />
    <line x1="14.5" y1="105" x2="12.5" y2="105" class="axes" />
    <text role="axislabel" x="4" y="5.75" class="chart-label">1.31</text>
    <text role="axislabel" x="4" y="55.75" class="chart-label">1.26</text>
    <text role="axislabel" x="4" y="106.75" class="chart-label">1.21</text>

    <line x1="31.66" y1="105.5" x2="31.66" y2="107.5" class="axes" />
    <line x1="48.33" y1="105.5" x2="48.33" y2="107.5" class="axes" />
    <line x1="65"    y1="105.5" x2="65"    y2="107.5" class="axes" />
    <line x1="81.66" y1="105.5" x2="81.66" y2="107.5" class="axes" />
    <line x1="98.33" y1="105.5" x2="98.33" y2="107.5" class="axes" />

    <text x="117"   y="106" class="chart-label">Date</text>
    <text x="24.16" y="112" class="chart-label">2016-06-22</text>
    <text x="40.83" y="112" class="chart-label">2016-06-23</text>
    <text x="57.5"  y="112" class="chart-label">2016-06-24</text>
    <text x="74.16" y="112" class="chart-label">2016-06-25</text>
    <text x="90.83" y="112" class="chart-label">2016-06-26</text>
  </g>
</svg>
```

Listing 4.1: A scalable vector graphic line chart.

The first step in improving the SVG therefore, is to set a title and a description. The title should be the first element of the SVG and give a very brief text-only overview. The description instead goes into further detail and presents a verbose textual story about the given graphic. Some user agents may tend to use the title as a tooltip that is shown on hovering the graphic with the mouse cursor. The description is presented on request. It can be assumed that a screen reader will read out the title for the user and offer to read the description if selected via user input. Listing 4.2 displays the main `title` and `descr` tag used for the improved SVG example. Note that the title and description tag are part of available SVG elements specification and not extended attributes offered by WAI-ARIA.

In order to further provide meta information about the data represented in the informational visualisation, WAI-ARIA is used. In further detail roles are defined to determine the type of an object, attributes describe the state or a certain property of a document node, and relational attributes link specific content to its corresponding label and descriptions.

A continued implementation is shown in listing 4.3. On the root level the SVG `g` tag is used to specify that the elements included are grouped together. To identify the group as a chart the role of a chart is applied and the charttype set to line. Additionally a tabindex is set in order to make the element focusable and therefore

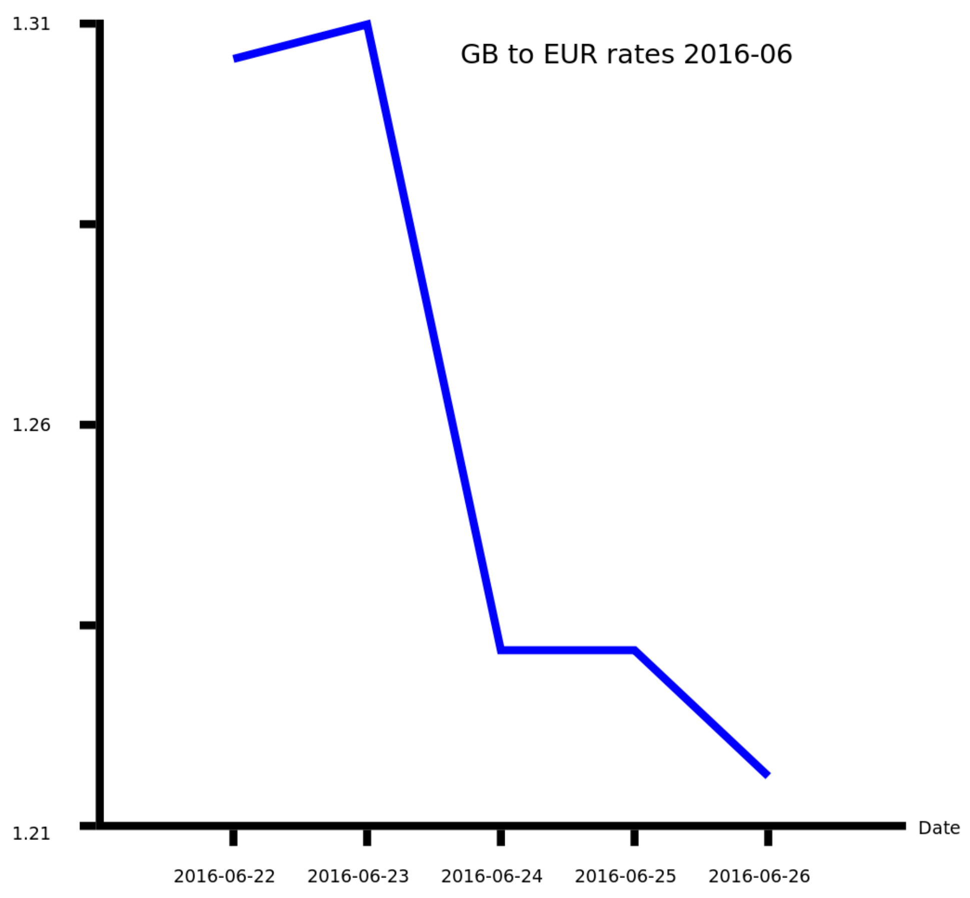


Figure 4.1: Line chart presenting the rate of EURO to British pounds in June 2016, before and after the british referendum to leave the European Union. Created by the authors.

```
<svg
  version="1.1"
  baseProfile="full"
  xmlns="http://www.w3.org/2000/svg"
  xmlns:xlink="http://www.w3.org/1999/xlink"
  viewBox="0 0 125 125"
>
  <title>Line chart of GB to EUR rates in June 2016</title>
  <desc>An example of an accessible scalable vector graphic presenting
     a line
  chart with the currency comparison of british pound GB to euro EUR
     before and
  after the british referendum for the brexit in july 2016.</desc>
  <!-- ... -->
</svg>
```

Listing 4.2: Using a title and description tag to provide basic overview of the informational content.

```
<svg
  <!-- ... -->
  </defs>
  <g role="chart" aria-charttype="line" tabindex="0">
    <text role="heading" x="60" y="10" class="chart-title">GB to EUR
        rates
      2016-06</text>
  <!-- ... -->
</svg>
```

Listing 4.3: Group the chart elements together, define a proper role and type of chart and declare the main label to be the header.

accessible for any user that is limited to keyboard navigation. As the element can now be focused, a screen reader can also already provide the first useful information about the graphical content by reading out the titles and descriptions given. When setting a tabindex it is a good practice to always use the value of zero. The value given represents the order of how elements in a document will be focused. With many objects or dynamic content, this can be very complex and break a useful flow of navigation when interacting via keyboard. By using a value of zero only, the navigational order will remain in the order of the document elements defined in the sourcecode.

The y axis as shown in listing 4.4 serves the graphic to show a scale for the rate values. Three values are labeled and a title for the axis is present, the polyline for the frame of the axes does stay the same as it only serves as a visual support and does not hold any content information. In order to make it accessible the axis is grouped and a role of "yaxis" is declared on it, all elements that are bound to the y axis are grouped together and the main labels text element is declared to be of the role heading. Another tabindex attribute ensures this content is focusable for any user with keyboard navigation. As mentioned in the previous step, a tabindex value of zero is used so that the order of the navigation is not mixed. Any present label of the scale is grouped together with the corresponding axis and declared to represent an axislabel via the role attribute. As the y axis uses a fixed numerical range a minimum and maximum value is additionally defined using aria-valuemin and area-valuemax. This optional choice to enhance the level of detail can come in handy for an user agent that may not just read description and labels but summarize information that is easy to capture visually but harder to follow if just using text like the range or an average of the numerical scale.

In a similar way to the improvements on the y axis, the x axis shown in listing 4.5 is adapted to group the specific points with the corresponding label. A tabindex is set on the axis role object and the heading declared. As the datapoints should be linked to the data, every label receives an unique identifier whereas a format of `x-<yearmonthday>` was choosen to ensure uniqueness.

In the last step of the improvement meta information is added to the datapoints. The first change in listing 4.6 is the declaration of the collected datapoints to group as a dataset role. In difference to the axes every single datapoint has a tabindex and is linked to the associated label using the `aria-labelledby` attribute with a value of a unique identifer. To additionally improve visual feedback every datapoint has a new small black circle element in order to see on first sight what comes next when using keyboad navigation only.

When comparing the original output of an assistive technology, the changes done have several effects. The axes, datapoints and the chart itself can now be accessed using a keyboard navigation only, that in example is the case when using a screen reader. On each navigational stop, headings, titles, descriptions, and labels can now be interpreted and read out to the user. The data definition of axes allow user agents to summarize details of the scales used and provide this optional information. Datapoints are linked to the date labels so that a screen

```
<svg
  <!-- ... -->
    <polyline points="15,4.5 15,105 115.5,105" class="axis" />

    <g role="yaxis" aria-valuemin="1.21" aria-valuemax="1.31" tabindex="0">
      <text role="heading" x="10.5" y="3" class="chart-label">GB/EUR</text>
      <g>
        <text role="axislabel" x="4" y="5.75" class="chart-label">1.31</text>
        <line x1="14.5" y1="5" x2="12.5" y2="5" class="axis" />
      </g>
      <line x1="14.5" y1="30" x2="12.5" y2="30" class="axis" />
      <g>
        <text role="axislabel" x="4" y="55.75" class="chart-label">1.26</text>
        <line x1="14.5" y1="55" x2="12.5" y2="55" class="axis" />
      </g>
      <line x1="14.5" y1="80" x2="12.5" y2="80" class="axis" />
      <g>
        <text role="axislabel" x="4" y="106.75" class="chart-label">1.21</text>
        <line x1="14.5" y1="105" x2="12.5" y2="105" class="axis" />
      </g>
    </g>
  <!-- ... -->
</svg>
```

Listing 4.4: On the y axis define the scale labels and make it focusable.

reader will read out the third datapoint from listing 4.6 as “June 24th 2016, 1.2319”. With minor adaptions to the original vector graphic, using detailed descriptions and WAI-ARIA attributes, the graphical information is now accessible to a much wider range of users.

```
<svg
  <!-- ... -->
    <g role="xaxis" tabindex="0">
      <text role="heading" x="117" y="106" class="chart-label">Date</text>
      <g>
        <text id="x-20160622" role="axislabel" x="24.16" y="112"
          class="chart-label">2016-06-22</text>
        <line x1="31.66" y1="105.5" x2="31.66" y2="107.5" class="axis" />
      </g>
      <g>
        <text id="x-20160623" role="axislabel" x="40.83" y="112"
          class="chart-label">2016-06-23</text>
        <line x1="48.33" y1="105.5" x2="48.33" y2="107.5" class="axis" />
      </g>
      <g>
        <text id="x-20160624" role="axislabel" x="57.5" y="112"
          class="chart-label">2016-06-24</text>
        <line x1="65" y1="105.5" x2="65" y2="107.5" class="axis" />
      </g>
      <g>
        <text id="x-20160625" role="axislabel" x="74.16" y="112"
          class="chart-label">2016-06-25</text>
        <line x1="81.66" y1="105.5" x2="81.66" y2="107.5" class="axis" />
      </g>
      <g>
        <text id="x-20160626" role="axislabel" x="90.83" y="112"
          class="chart-label">2016-06-26</text>
        <line x1="98.33" y1="105.5" x2="98.33" y2="107.5" class="axis" />
      </g>
    </g>
  <!-- ... -->
</svg>
```

Listing 4.5: The x axis is limited to the dates from 22. June to 26. June 2016. Those points are grouped on the axis with their label and receive a unique identifier.

```
<svg
  <!-- ... -->
    <g id="dataarea" role="dataset">
      <polyline points="31.66,9.4 48.33,5.1 65,83.1 81.66,83.1
          98.33,98.8"
        class="chart-data" />
      <g role="datapoint" tabindex="0">
        <title role="datavalue" aria-labelledby="x-20160622">1.3056</
            title>
        <circle cx="31.66" cy="9.4" r="1" fill="black" />
      </g>
      <g role="datapoint" tabindex="0">
        <title role="datavalue" aria-labelledby="x-20160623">1.3091</
            title>
        <circle cx="48.33" cy="5.1" r="1" fill="black" />
      </g>
      <g role="datapoint" tabindex="0">
        <title role="datavalue" aria-labelledby="x-20160624">1.2319</
            title>
        <circle cx="65" cy="83.1" r="1" fill="black" />
      </g>
      <g role="datapoint" tabindex="0">
        <title role="datavalue" aria-labelledby="x-20160625">1.2319</
            title>
        <circle cx="81.66" cy="83.1" r="1" fill="black" />
      </g>
      <g role="datapoint" tabindex="0">
        <title role="datavalue" aria-labelledby="x-20160626">1.2162</
            title>
        <circle cx="98.33" cy="98.8" r="1" fill="black" />
      </g>
    </g>
  <!-- ... -->
</svg>
```

Listing 4.6: Inside the grouped datapoints, defined to be of role dataset, each point is uniquely linked to a specific label from the x axis.

Chapter 5

Tools

Accessibility gets more and more important, developers, web developers and designers of graphics, must make their work accessible and be able to test their software, websites or graphics. They need to add accessibility features and need authoring tools that allows them to add them. Each screen reader works different and support different accessibility features, which makes it necessary for developers to test their work with different screen readers or use special tools that let them test their work.

5.1 Support in Design Tools

Making SVGs and SVG charts accessible for screen readers, requires a lot of work by the authors, using the description and title tags in their SVG graphics. Using the SVG elements to describe the elements in an SVG icon, graphic or chart, make the graphics accessible for screen readers or only text browsers, which would only see an image. Extending the SVG file with these describing elements, can mean a lot of work if they are added by hand into the SVG code.

To avoid writing the SVG code by hand, the designers use authoring tools as the commercial Adobe Illustrator or the open source software Inkscape. Both tools allow the authors to add titles and descriptions during their workflow. Adobe Illustrator is a big software solution for designers, and allows many different options to export, or safe the files. However, all these options are sometimes the reason why it is complicated to find the right settings to include all the wanted SVG descriptions in the exported files. Some export versions only allow to export the SVG files with a lot of extra information, what makes the exports big and not very usable for the internet [D. A. Schepers, 2013].

Designers of SVG graphics do often use different SVG graphics and pull them together into one new graphic. But these graphics often don't have accessibility features, what would make them describable. Therefore, library's for SVG graphics and authors should describe their graphics and make them accessible. However, many screen readers still have problems supporting the SVG "desc" and "title" elements properly, therefore it is recommended to use WAI-ARIA to make the graphics and charts even more accessible for all the different used screen readers.

5.2 Screen Reader and Screen Reader Testing Tools

Screen readers are a tool for blind people or people with visual disabilities, which can read the text of documents, websites or other programs out loud. This allows the users to navigate and use a computer without being able to see the information on the screen. Users of screen reading software usually navigate without a mouse, instead they use shortcut-keys on a keyboard or Braille keyboard. All the shortcuts, that are required to navigate and use the features of a screen reader make them complicated for people who are not familiar with them. Therefore, a lot of practice is required, which often needs several hours to get used to how a screen reader works, and how to use it.

Screen reader	%
JAWS	30.2%
ZoomText	22.2%
Windows-Eyes	20.7%
NVDA	14.6%
VoiceOver	7.6%
Others	4.7%

Table 5.1: Primary used Screen Readers according to WebAIM [2015].

Screen reader	%
JAWS	43.7%
NVDA	41.4%
VoiceOver	30.9%
Window-Eyes	29.6%
ZoomText	27.5%

Table 5.2: Commonly used Screen Readers according to WebAIM [2015].

The afford needed to use screen readers, also makes it harder for developers, to test their websites with them. Tools like Describler, that allow the users to see how accessible their SVGs are and to optimize the accessibility of them, shall help the developers with their work. However, each screen reader works different and the most widely used screen readers as JAWS, NVDA, Voice over, ZoomText or Window-Eyes, are all only running on one specific operating system. JAWS, NVDA, Window-Eyes, and ZoomText only run on a Microsoft Windows operating system, whereas VoiceOver was developed by Apple and run only on their MacOS and iOS devices.

Some of the screen readers as NVDA and ZoomText are also available as portable version and can be used from an USB stick. This allows the users to carry the screen reader with them and use it on different computers, for example at work and at home. However, a lot of people must use more than one screen reading software, when they want to work with their mobile devices or on computers with different operating systems.

JAWS was the most important screen reader a few years ago, as ZoomText it is a commercial software, whereas NVDA is an open source software and can be downloaded and used for free. Apples VoiceOver comes with every Apple computer or mobile device and has no extra cost to use it. Window-eye is also a commercial software, but it has a cooperation with Microsoft to make it available free for users of Microsoft Office 2010 and newer.

According to a survey from [WebAIM, 2015] in 2015, JAWS was the most used screen reader in 2009, with over 74% of the screen reader users using it. In the last years that number went down, in 2015 there have been 30.2% using JAWS as their main screen reader, 22.2% were using ZoomText, and Window-Eye was used by 20.7% what might be related, that Microsoft made it free for users of Microsoft Office 2010 and newer. NVDA was the main screen reader for 14.6% of the users and Apples VoiceOver was only used by 7.6%, as shown in table (see table 5.1) which uses the data from [WebAIM, 2015].

As shown in table 5.2 from [WebAIM, 2015], which shows the percentage of participants in their survey, that commonly uses this Screenreaders, JAWS is used by 43.7% of the people, NVDA is used by 41.4% of the people, Voice Over by 30.9%, Window-Eyes by 29.6% and ZoomText by 27.5%. It also shows that this five screen readers are all very popular.

5.2.1 NonVisual Desktop Access

NonVisual Desktop Access (NVDA) is a free and open source screen reader software developed by Michael Curran and James The, who are both fully blind. They founded NV Access, a registered charity from Australia, to support and develop NVDA and the software can be downloaded from their website [NVAccess, 2017]. The fact, that the software is open source, makes the code accessible for others, developers can extend the software or translate it into other languages. This made it possible for volunteers to translate the software into more than 43 languages and makes NVDA be used in over 120 countries.

NVDA runs only on computers with a Microsoft Windows operating system, however, it does not need any additional video card drivers to be installed with as some other screen reader software. For users who do not want to install NVDA onto their system, there is a portable version available as well. The portable version allows the users to run the screen reader from an USB stick, which makes it useable on different computers at work or at home without an additional installation.

The fact that NVDA is free, makes it useful for developers who want to test the accessibility of their websites. NVDA was used for accessibility testing by companies like Microsoft, Yahoo or Google. Even so it works with other internet browsers too, NVDA works currently best with Mozilla Firefox.

As other screen readers too, NVDA has still some issues with SVG accessibility when using nested title and description tags. Some of the problems are, that it does not recognize the description element or that it reads the SVG title text twice [Halter, 2016].

5.2.2 Job Access with Speech

Job Access With Speech (JAWS) is a commercial screen reader developed by Freedom Scientific. It was the most used screen reader software in the last years and more than 74% of the users of screen readers used it in 2009 [WebAIM, 2015]. In the last years, its popularity decreased some but it is still one of the most used screen readers. JAWS is translated into 17 languages and distributed in more than 50 countries.

JAWS only can be used on computers with a Microsoft Windows operating system, and the newest version of JAWS only works with Windows 8.1 or Windows 10. It is possible to active remote access which provides support for Remote Desktop, or other software and allows the user to connect and use the screen reader from another computer.

The JAWS screen reader software is available in three commercial versions, the JAWS Timed License, a 90 Days Version for 179 USD, the JAWS Home Edition for 900 USD and the JAWS Pro version for 1,000 USD, which is needed for use in work or commercial environments. JAWS does not just provide speech output but also provides Braille output for the most popular computer applications. It supports Firefox, Internet Explorer and many other commonly used computer software. Nevertheless, it is best optimized for the use with the Internet Explorer. JAWS is also available as a free DEMO version which allows 40 minute usage before it needs a restart. Freedom Scientific prefers the developers not to use this version for testing, however, they can contact them if they are interested in testing with JAWS.

In their latest JAWS 18 version they support several ARIA 1.1 features, as aria-modal, aria-current and aria placeholder [FreedomScientific, 2017].

5.2.3 VoiceOver

VoiceOver is a screen reader software that is already integrated in the macOS (MAC OperatingSystm) and therefore free for everyone with a Mac version newer to Mac OS X 10.4 or an iPhone 3GS or newer. By pressing “cmd + F5” it can be activated and deactivated. VoiceOver can read the text on the screen, can be used to navigate and can print to a Braille Text device. It only is available for Apple devices that run MacOS or iOS as their operating system. VoiceOver also allows the use of special VoiceOver-gestures when its used together with a multi-touch-trackpad. The iOS version for iPhones and iPads uses only gestures for the interaction and can leave the devices screen black and just work with gestures and speech output. Users can also use Apples Siri to communicate with the screen reader.

Depending on the used Apple Device (Apple TV, Appple, Watch, iPhone, Mac, ...) the number of supported languages varies, however, for Mac there is support for over 30 languages. The quality of the SVG accessibility support from VoiceOver also variates between MacOS and the iOS version. Where the Mac version does not recognize the desc element in SVGs, the iOS version does not recognize the title element either

[Halter, 2016].

Most users of screen readers are still use a Microsoft Windows operating system, which is still a reason why many sides are not optimized for Apples VoiceOver screen reader. The VoiceOver screen reader also uses different shortcut commands. Due to this different shortcut-keys they don't need different modes as windows screen readers have them [Whiting, 2016], which results in different behaviours when testing the accessibility for a side on a Mac or Windows Computer.

5.2.4 Window-Eyes

Window-Eyes is a commercial screen reader software developed by GW Micro, which requires a Microsoft Windows operating system to be installed on. The software is distributed over local partners, however, for users of Microsoft Office 2010 and newer, Microsoft has a partnership with GW Micro, that allows them to download the screen reader from the "Window-Eyes for Office" website [VFO Group, 2017] and is available in 19 languages. In the USA and Canada, GW Micro stopped to sale their Window-Eyes Screen reader and offered their users migrations to JAWS 18 [Ai Squared, 2017] last visited: May 15, 2017}. Existing versions can still be used and the free version for Microsoft Office users will still be available.

Window-Eyes uses standard Microsoft controls and every software using this controls is supported with only less extra configurations needed. It supports Microsoft Word, Excel, PowerPoint and Microsoft Outlook since 2007. Window-Eyes also advanced web-browsing support for Internet Explorer and Mozilla Firefox. The screen reader has two output modes, it allows to output as speech or on a Braille display and can switch between them on the fly. There is also support for ARIA, which includes landmarks, live regions and more.

5.2.5 ZoomText

ZoomText is a commercial screen magnifier and can be purchase with an included screen reader, it is developed by AI Squared. It is only available for Mircosoft Windows operating systems. It can be purchased on their website as an installation version or as a portable version on an USB Stick. The ZoomText Magnifier Version 11 costs 695 Euro and the USB stick version comes for 795 Euro and allows the users to take the screen reader with them and use it on different computers with a Microsoft Windows operating system.

ZoomText also includes a screen magnifier, which allows the user to zoom the screen and make the information on the display bigger. This helps people who are not blind but have visual impairments, to still use the computer, but also have support of the screen reader if needed.

5.3 SVG Testing using Describler

Describler is an open-source software application that can be used to test how meta information inside a SVG may be interpreted by an assistive technology. The sourcecode was created by Doug Schepers who worked in the W3C working groups from 2007 to 2017 with a special focus on SVG and accessibility. The software can be executed on any website, sourcecode is available on github [**escribler-Git-Hub**] and a published version at [D. A. Schepers, 2017]. There is no requirement to install any additional software, it can be run by using any web browser. Besides HTML for the website layout and CSS used to style the controls, the application uses only JavaScript to handle the SVG files uploaded, parse them, generate and optimize text to speak out, generate sound for speech and sonification. Later is a method used to generate an audio signal and pitch the audio by the flow of a chart.

The Describler software also provides a tree of all tags, titles and descriptions, the a developer can browse to test and find out that tags are missing in order to make SVGs more accessible. It is easy to use and its focus is on making SVGs and especially SVG data visualizations accessible for blind people. In comparison to the current state of screen reader available for testing in section Screen Reader and Screen Reader Testing Tools the application has better result when interpreting titles, descriptions and WAI-Aria attributes.

Bibliography

Ai Squared [2017]. *WindowEyes migrate to JAWS 18*. 2017. `http://gwmicro.com/window-eyes/migrate/` (cited on page 24).

Asada, Kazunori [2013]. *Chromatic Vision Simulator*. 2013. `http://asada.tukusi.ne.jp/cvsimulator/e/index.html` (cited on page 7).

Bigmann, Alex [2013]. *Why all designers need to understand color blindness*. 2013. `https://en.99designs.at/blog/tips/designers-need-to-understand-color-blindness/` (cited on page 9).

Bundeskanzleramt [2017]. *Barrierefreies Web*. 2017. `https://digitales.oesterreich.gv.at/barrierefreies-web-zugang-fur-alle` (cited on page 2).

Chimurkar, Renuka M. and Vijay Bagdi [2014]. "HAPTICS TECHNOLOGY: (sense of touch)". *International Journal Of Engineering And Computer Science* 3.11 [11th Nov 2014], pages 9188–9192. `https://ijecs.in/issue/v3-i11/47%20ijecs.pdf` (cited on page 5).

Dandona, Lalit and Rakhi Dandona [2006]. "What is the global burden of visual impairment?" *BMC medicine* 4.1 [16th Mar 2006], page 6. doi:10.1186/1741-7015-4-6. `https://bmcmedicine.biomedcentral.com/articles/10.1186/1741-7015-4-6` (cited on page 5).

Fernandez, Clarizza [2012]. *How to create an accessible infographic*. 27th Sep 2012. `http://accessiq.org/create/content/how-to-create-an-accessible-infographic` (cited on page 10).

FreedomScientific [2017]. *What's New in JAWS 18 Screen Reading Software*. Apr 2017. `http://freedomscientific.com/Downloads/jaws/JAWSWhatsNew` (cited on page 23).

Gegenfurtner, Karl R and Lindsay T Sharpe [1999]. *Color Vision: From Genes to Perception*. Cambridge, UK: Cambridge University Press, 1999, page 492. ISBN 0-521-59053-1. `http://allpsych.uni-giessen.de/karl/colbook/index.html` (cited on page 6).

Halter, Adina [2016]. *Inline SVG used for buttons and links*. Dec 2016. `https://haltersweb.github.io/Accessibility/svg.html` (cited on pages 23–24).

Hisagi [2010]. *Map of Tokyo Metro lines and Toei Subway lines*. 23rd Aug 2010. `https://commons.wikimedia.org/wiki/File:Tokyo_metro_map_en.png` (cited on page 11).

Machado, Gustavo M, Manuel M Oliveira and Leandro AF Fernandes [2009]. "A physiologically-based model for simulation of color vision deficiency". *IEEE Transactions on Visualization and Computer Graphics* 15.6 [23rd Oct 2009], pages 1291–1298. doi:10.1109/TVCG.2009.113. `http://ieeexplore.ieee.org/document/5290741/` (cited on pages 5–6).

Morton, Jill [2010]. "Why color matters". *COLORCOM* [2010]. `http://colorcom.com/research/why-color-matters` (cited on page 9).

National Highway Traffic Safety Administration [2012]. *Car Seat Recommendations: Choosing the Right Seat*. 6th Apr 2012. `https://commons.wikimedia.org/wiki/File:Infographic_car_seat_recommendations_choosing_right_seat.jpg` (cited on page 10).

NVAccess [2017]. *NVAccess Website*. 2017. `http://nvaccess.org` (cited on page 23).

Rigden, Christine [1999]. "'The Eye of the Beholder'-Designing for Colour-Blind Users". *British Telecommunications Engineering* 17 [1999], pages 291–295. `http://colinpurrington.com/wp-content/uploads/2011/09/Rigden19991.pdf` (cited on page 6).

Schepers, Doug [2015]. *ARIA Dataviz Vocabulary Strawman*. Jan 2015. `https://lists.w3.org/Archives/Public/public-svg-a11y/2015Jan/0006.html` (cited on page 14).

Schepers, Douglas Alan [2013]. *Current State of Authoring Accessible SVG*. Sep 2013. `http://schepers.cc/authoring-accessible-svg` (cited on page 21).

Schepers, Douglas Alan [2017]. *Describler Website*. 2017. `http://describler.com` (cited on page 24).

Srinivasan, Mandayam A [1995]. "What is haptics?" *Laboratory for Human and Machine Haptics: The Touch Lab, Massachusetts Institute of Technology* [1995], pages 1–11. `http://geomagic.com/files/7713/4857/8044/what_is_haptics.pdf` (cited on page 5).

Throup, Chris [2012]. *Accessible infographics*. 14th Apr 2012. `http://chris.throup.org.uk/2012/04/14/accessible-infographics/` (cited on page 10).

Van Gogh, Vincent [1889]. *The Starry Night*. 1889. `https://commons.wikimedia.org/wiki/File:Van_Gogh_-_Starry_Night_-_Google_Art_Project.jpg` (cited on page 7).

Van Gogh, Vincent [1890]. *Walk in the moonlight*. 1890. `https://de.wikipedia.org/wiki/Datei:Van_gogh-spaziergang_im_mondlicht.jpg` (cited on page 7).

VFO Group [2017]. *Window-Eyes for Office*. 2017. `http://windoweyesforoffice.com` (cited on page 24).

WAI [2008]. *Web Content Accessibility Guidelines (WCAG) 2.0*. Dec 2008. `https://w3.org/TR/WCAG20/` (cited on page 1).

WebAIM [2013a]. *How Blind People Use the Web*. 28th Aug 2013. `http://webaim.org/articles/visual/blind` (cited on page 5).

WebAIM [2013b]. *Visual Disabilities*. 28th Aug 2013. `http://webaim.org/articles/visual/colorblind` (cited on page 6).

WebAIM [2015]. *Screen Reader User Survey 6 Results*. 28th Aug 2015. `http://webaim.org/projects/screenreadersurvey6` (cited on pages 22–23).

WebAIM [2016a]. *Implementing Web Accessibility*. 15th Mar 2016. `http://webaim.org/intro/#implementing` (cited on pages 1–2).

WebAIM [2016b]. *Principles of Accessible Design*. 15th Mar 2016. `http://webaim.org/intro/#principles` (cited on page 2).

Whiting, Jon [2016]. *Three things you should know before using VoiceOver for testing*. 31st Aug 2016. `http://webaim.org/blog/three-things-voiceover` (cited on page 24).

WHO [2014]. *Visual impairment and blindness, Fact Sheet number 282*. Aug 2014. `http://.who.int/mediacentre/factsheets/fs282/en/` (cited on page 5).

Wikimedia Commons [2010]. *Ishihara Plate 9*. 21st Jul 2010. `https://commons.wikimedia.org/wiki/File%3AIshihara_9.png` (cited on page 8).

Wilkinson, Sarah E, Marnique Y Basto, Greta Perovic, Nathan Lawrentschuk and Declan G Murphy [2015]. "The social media revolution is changing the conference experience: analytics and trends from eight international meetings". *BJU international* 115.5 [2015], pages 839–846. doi:10.1111/bju.12910. `https://ncbi.nlm.nih.gov/pubmed/25130687` (cited on page 9).