# Slide Decks in HTML

Patrick Hipp

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

25 Jan 2019

**Abstract**

Slide decks built for the web can be flexible and feature-rich for presenters, while at the same time being viewable in any web browser on any device worldwide. They are a viable alternative to proprietary presentation software such as PowerPoint and Keynote and offer unique features such as live code execution.

This survey discusses the history and development of packages for web-based slide decks and gives an overview of currently available solutions. The four main approaches described are: text-based, JavaScript-based, hosted, and responsive slide decks.

# Contents

# List of Figures

# List of Tables

# List of Listings

Chapter 1

# Introduction

Presentations can have a huge impact on the world. On the 5th of February 2013, U.S. Secretary of State Colin Powell went before the Security Council of the United Nations to give a presentation on the development of weapons of mass destruction in Iraq [Brock 2017]. His 45 PowerPoint slides were well-organised and convincing and are considered to be "one of the most famous PowerPoint presentations of all time."

## 1.1 The Origins of Slide Shows

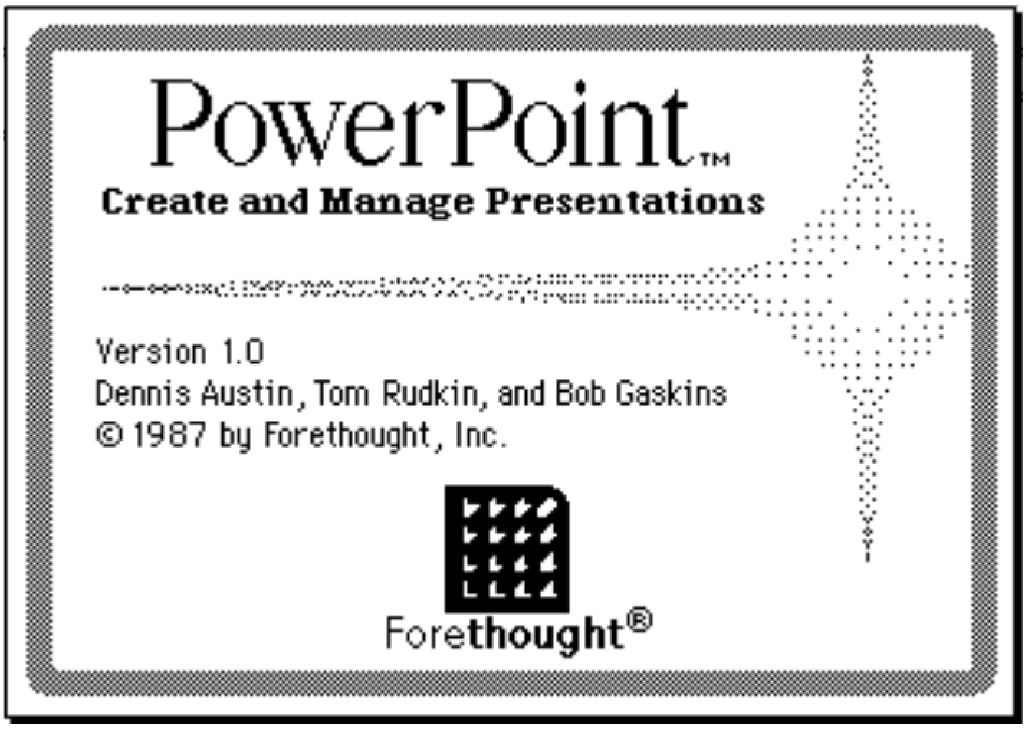
In the 1980s, presentation slides were typically made in one of two ways [Gaskins 2012, pages 15–19]:

- *Overhead transparencies*: Black and white slides were designed on a computer, printed onto paper on a laser printer, then copied onto overhead transparencies (clear plastic sheets) to be shown on an overhead projector.
- *35mm slides*: Colour slides were prepared on a workstation as say GIF images, then transferred (via expensive hardware) to individual 35 mm photographic slides. Once framed, these were sorted into a slide rack and shown using a slide projector.

At the time, laser printers and copiers could only produce black and white images, while preparing 35mm slides was expensive and time-consuming.

PowerPoint, shown in Figures 1.1 and 1.2, changed all that and is now the most well-known presentation application. Microsoft estimates that 1.2 billion copies of PowerPoint now exist [Brock 2017, page 44]. Originally released in April 1987 by Forethought Inc. for the Macintosh [Gaskins 2012, page 16], PowerPoint 1.0 quickly made a name for itself, with sales of $ 1 million in its first month [Brock 2017, page 49]. At launch, PowerPoint was limited to black and white for overhead transparencies [Phi 2018], but support for color 35mm slides was added in version 2.0 [Gaskins 2012, page 17]. Forethought was bought by Microsoft for $ 14 million just three months after the initial release of PowerPoint. PowerPoint made presentation software accessible to the public and defined some essential features for such applications.

Apple's Keynote [Apple 2018] presentation software was originally custom-built for Steve Jobs himself to use during his Apple keynote talks [Roemmele 2013]. In January 2003, Keynote was released for MacOS [Apple 2003], as an alternative to PowerPoint. It is currently part of the iWork suite and is now free for both MacOS and iOS [Clover 2017]. Keynote comes with built-in support for Apple remotes and iCloud synchronisation, is cross-compatible with PowerPoint, and shares similar features.



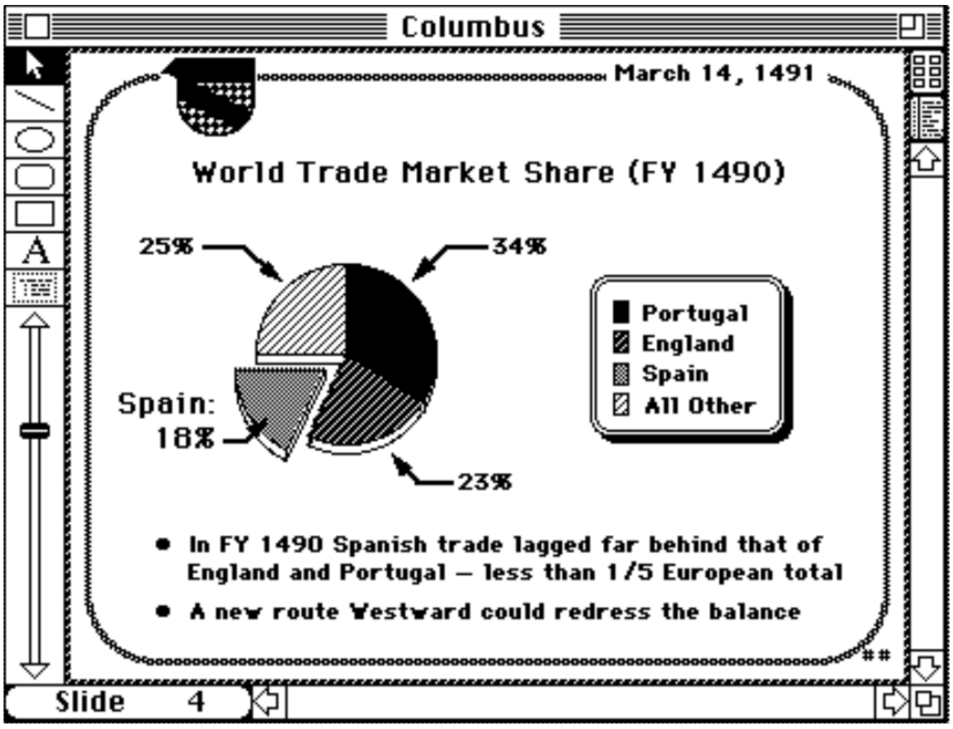
PowerPoint™

Create and Manage Presentations

Version 1.0
Dennis Austin, Tom Rudkin, and Bob Gaskins
© 1987 by Forethought, Inc.

Forethought®

**Figure 1.1:** The splash screen of PowerPoint 1.0 by Forethought, Inc. [Screenshot made by the author using emulation [Friend 2018].]



Columbus

March 14, 1491

World Trade Market Share (FY 1490)

- In FY 1490 Spanish trade lagged far behind that of England and Portugal — less than 1/5 European total
- A new route Westward could redress the balance

Slide 4

**Figure 1.2:** A slide of an example presentation included in PowerPoint 1.0. [Screenshot made by the author using emulation [Friend 2018].]

| Top Ten Platforms | | |
|---|---|---|
| 1 | Android 7 | 15.46% |
| 2 | Windows 7 | 13.58% |
| 3 | Windows 10 | 13.05% |
| 4 | Android 6 | 10.83% |
| 5 | iOS 11 | 9.97% |
| 6 | Android 8 | 8.67% |
| 7 | Android 5 | 8.41% |
| 8 | Android 4 | 4.14% |
| 9 | Mac OS X | 2.91% |
| 10 | Windows 8.1 | 2.50% |

| Top Ten Screen Resolutions | | |
|---|---|---|
| 1 | 640x360 | 29.96% |
| 2 | 1366x768 | 10.79% |
| 3 | 1920x1080 | 5.76% |
| 4 | 1024x768 | 5.53% |
| 5 | 667x375 | 5.09% |
| 6 | 736x414 | 2.47% |
| 7 | 1440x900 | 2.33% |
| 8 | 1600x900 | 2.27% |
| 9 | 732x412 | 2.17% |
| 10 | 1280x800 | 2.09% |

**Table 1.1:** The top ten platforms and screen resolutions globally in September 2018 [W3Counter 2018].

## 1.2 Slide Guidelines

Slides are the building blocks of every presentation. They can have a variety of different layouts, but most commonly they consist of a heading followed by text, bulleted lists, charts, images, or videos. The purpose of a slide is often to present information in small, easily digestible pieces, which then build up to form a larger picture. Gabrielle [2010] distinguishes between ballroom style presentations, briefing decks, discussion decks, and reading decks, depending on the size of the audience and the amount of interaction between speaker and members of the audience.

Doumont [Doumont 2002; Doumont 2005; Doumont 2009] established three (four) rules for professional communication: 0) define your purpose, 1) adapt to your audience, 2) maximise the signal-to-noise ratio, and 3) use effective redundancy. Slides in a presentation should reinforce the message, focusing not on providing every detail, but rather on the implications that follow from them. Effective slides should not compete for attention with the speaker. This can be achieved by reducing the amount of text as far as possible and avoiding purely decorative graphics or backgrounds. For example, clever rephrasing should be used to reduce bulleted list items to no more than two lines of text. If detailed tables, charts, or graphics would be helpful to promote the message, they can be distributed to audience members in the form of a handout.

Alley [2013] presents some guidelines for creating and giving scientific presentations. He points out that following PowerPoint's default slide layout leads to a topic–subtopic approach, which is often unsuitable for scientific purposes. In a topic–subtopic approach, the headline dictates the main topic and is followed by a bulleted list which further explores it [Alley 2013, page 110]. Research has shown that assertion–evidence slides lead to better comprehension of complex topics [Alley 2013, page 117]. In such slides, most of the words are found in the headline, presenting an assertion in the form of a sentence, while visual aids are used as evidence to support it [Alley 2013, page 121].

## 1.3 Modern Challenges

Over the last few years, a technological arms race took place between industry giants, causing mobile devices and web browsers to constantly evolve and change shape. This in turn directly affects the way these devices and browsers are used and how they are expected to behave in certain situations. As shown in to Figure 1.3, Google Chrome, Safari, UC Browser and Mozilla Firefox are currently the most used web browsers, with Chrome leading by a wide margin. The strong dominance of Chrome in the web browser market share is likely due to the fact that Android is now the leading operating system globally, as can be seen in Table 1.1, and it ships with Google Chrome preinstalled as the default web browser.

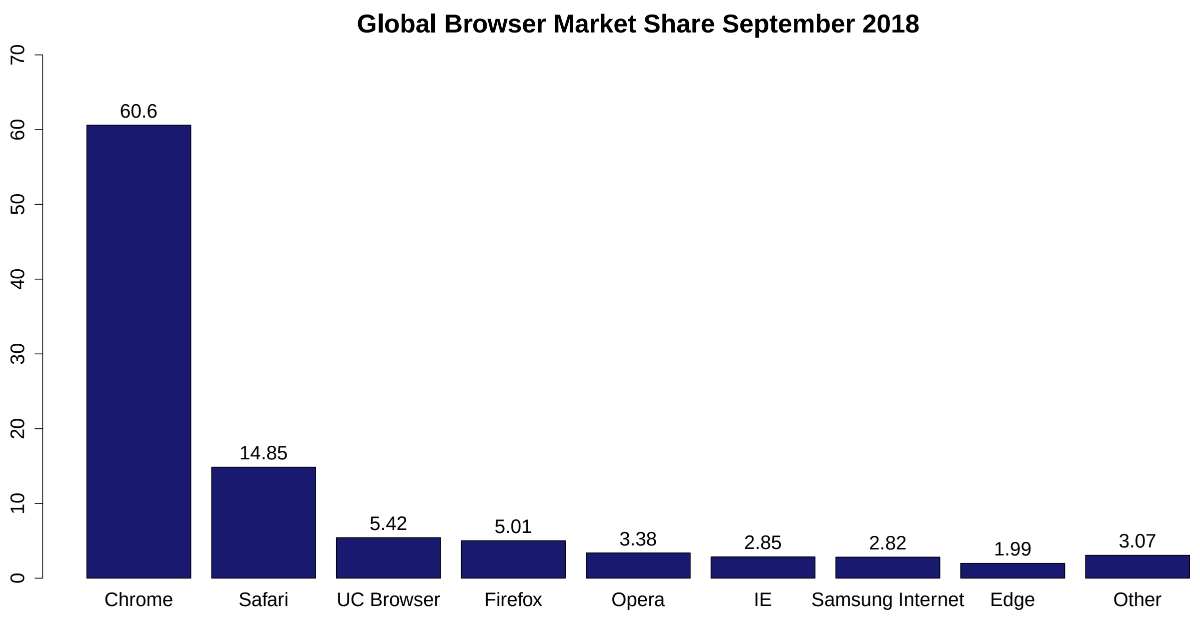


**Figure 1.3:** Browser market share over all platforms from September 2018. [Diagram made by the author based on data from StatCounter [2018].]

Developing an application for multiple devices, screen resolutions, and web browsers requires special effort, but is now considered to be best practice. Some of the challenges include:

- Support for all modern browsers: Generally speaking this includes Google Chrome, Mozilla Firefox, Safari, Microsoft Edge, and Opera.
- Support for all modern devices: Devices can be broadly categorised into desktop, tablet, and smartphone, but hybrids also exists.
- The ability to adapt to changes in the device environment: For example, detecting and utilising newly plugged-in peripheral devices, such as a mouse and keyboard, or redrawing the screen after device orientation has changed.
- Sharing and communication between different platforms.
- Accessibility: Intuitive controls, suitable for all kinds of users.
- Inclusion of third-party software.
- Resilient and responsive design.

For presentation software, the above are important considerations, but some additional problems have to be dealt with. These include zooming for images and graphs (especially on smaller screens), having the ability to print a slidedeck, navigation, speaker notes, and many more.

Fortunately, a combination of Hyper Text Markup Language (HTML), Cascading Style Sheets (CSS), and JavaScript can be used to tackle these problems. HTML is the foundation of everything that is built on the web [Keith and Andrew 2016]. It is a markup language which defines the building blocks making up a website, such as headings, paragraphs and images. CSS makes it possible to style those elements dynamically and change where and how they are displayed on the screen [Cederholm 2015]. JavaScript is a programming language which can modify the behaviour of a web site [Marquis 2016]. In short, HTML defines the content of a website, CSS defines the layout of that content, and JavaScript adds interactivity. Almost all of the solutions described in this survey make at least some use of each of these three fundamental web technologies.

## 1.4 Survey Structure

This survey gives an insight into the history of slide decks for the web. A plethora of web-based slideshow solutions are grouped into four categories based on their dominant design approach, and the contents of each group is discussed in chronological order. The categories are: text-based, JavaScript-based, hosted, and responsive slide decks. The detailed structure of this survey is as follows:

- Chapter 2 deals with some early systems and outliers that laid the ground work for modern slide decks.
- Chapter 3 discusses text-based slide decks, which take their input from plain text or markdown files.
- Chapter 4 looks at feature-rich slide decks based on JavaScript.
- Hosted slide decks are discussed in Chapter 5. They come in the form of neatly packaged web applications providing all the tools necessary to create slideshows.
- Chapter 6 presents state-of-the-art design approaches for responsive slide decks.
- Chapter 7 gives some concluding remarks.
- Finally, Appendix A covers some code to slide examples, recreating the same slide with each slideshow application.

# Chapter 2

# Early Systems and Outliers

This chapter discusses the early days of presentation software for the web and describes the most important applications of that era. It also briefly covers some outliers using technologies like flash or video encoding software.

## 2.1 Slidy and Slidy2

In 2005, Dave Raggett, a member of the W3C team, developed one of the first libraries for HTML-based slide presentations called Slidy [Raggett 2005]. Slidy turns an HTML file into a slideshow by including two files: a CSS style sheet to regulate the style of the presentation, and a JavaScript file to add presentation functionality. Additionally, an optional CSS file containing a print style sheet can be included. Slides are written in HTML and the contents of each slide is placed inside `<div class="slide">...</div>` elements.

Features include basic keyboard navigation, font size scaling, handout notes, incremental bullet lists, automatic numbering of slides, the ability to link to a specific slide, and a print style sheet. Disabling JavaScript or CSS displays all the slides, so in case of an error the main content can still be accessed.

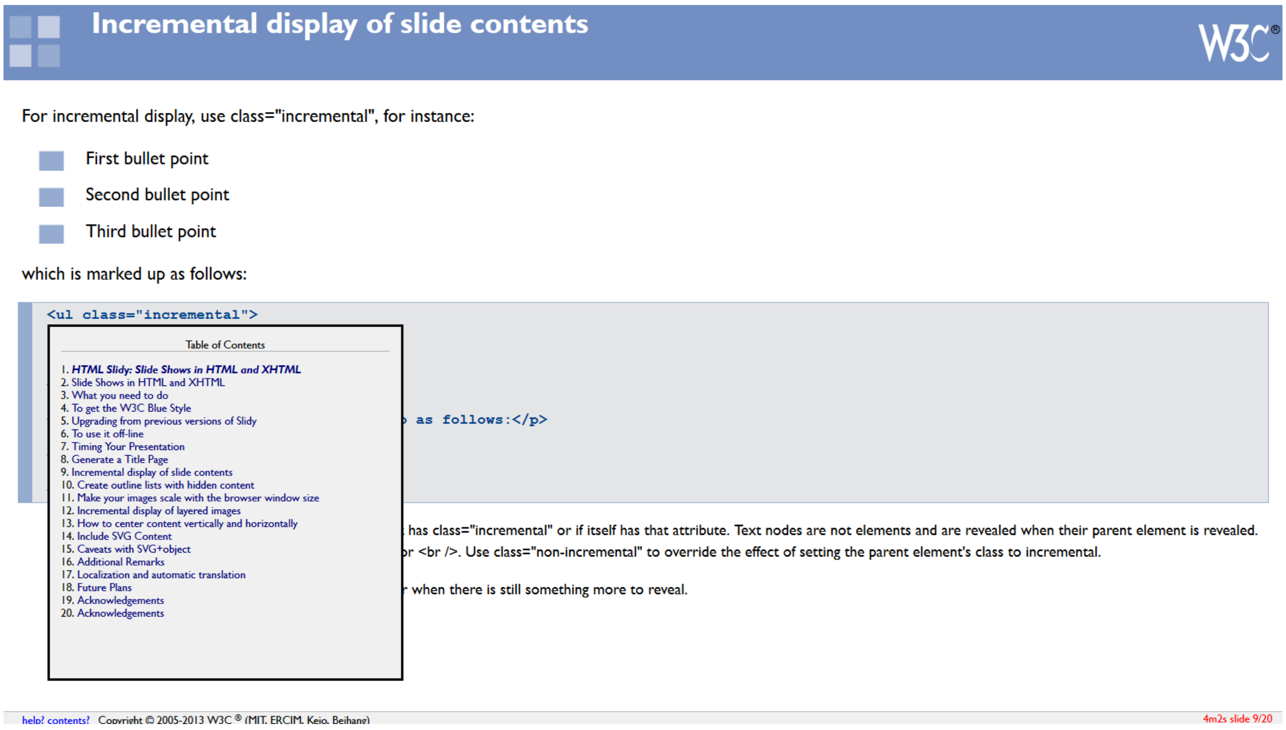
Raggett [2006a] later released an improved version of Slidy called Slidy2, which adds more functionality. The most important addition is an auto-generated table of contents, which takes the title of each slide and supports quicker navigation. An example slide with the table of contents can be seen in Figure 2.1, generated from the code excerpt shown in Listing 2.1. Other new features are the addition of a timer and localisation support. It is also fully backwards compatible with the first version of Slidy.

Slidy and Slidy2 were designed for use in a desktop environment, they do not scale well on narrower screens. Since keyboard shortcuts are typically unavailable on a mobile device, navigation quickly becomes a struggle. The footer is especially problematic on a touchscreen, since it only takes up a tiny sliver of the screen, making the links extremely difficult to use.

Further development plans for Slidy included a "what you see is what you get" editor, remote control of slides using HTTP requests, and the ability to import and export from or to other slide formats like PowerPoint and Open Office [Raggett 2006b], but these were never implemented.

```html
<div class="slide">
  <h1>Incremental display of slide contents</h1>

  <p>For incremental display, use class="incremental", for instance:</p>

  <ul class="incremental">
    <li>First bullet point</li>
    <li>Second bullet point</li>
    <li>Third bullet point</li>
  </ul>
  ...
</div>
```

**Listing 2.1:** An example slide in Slidy2. The class `incremental` can be used for incremental display of bullet items.



## Incremental display of slide contents

For incremental display, use class="incremental", for instance:

- First bullet point
- Second bullet point
- Third bullet point

which is marked up as follows:

```
<ul class="incremental">
```

```
 as follows:</p>
```

Table of Contents

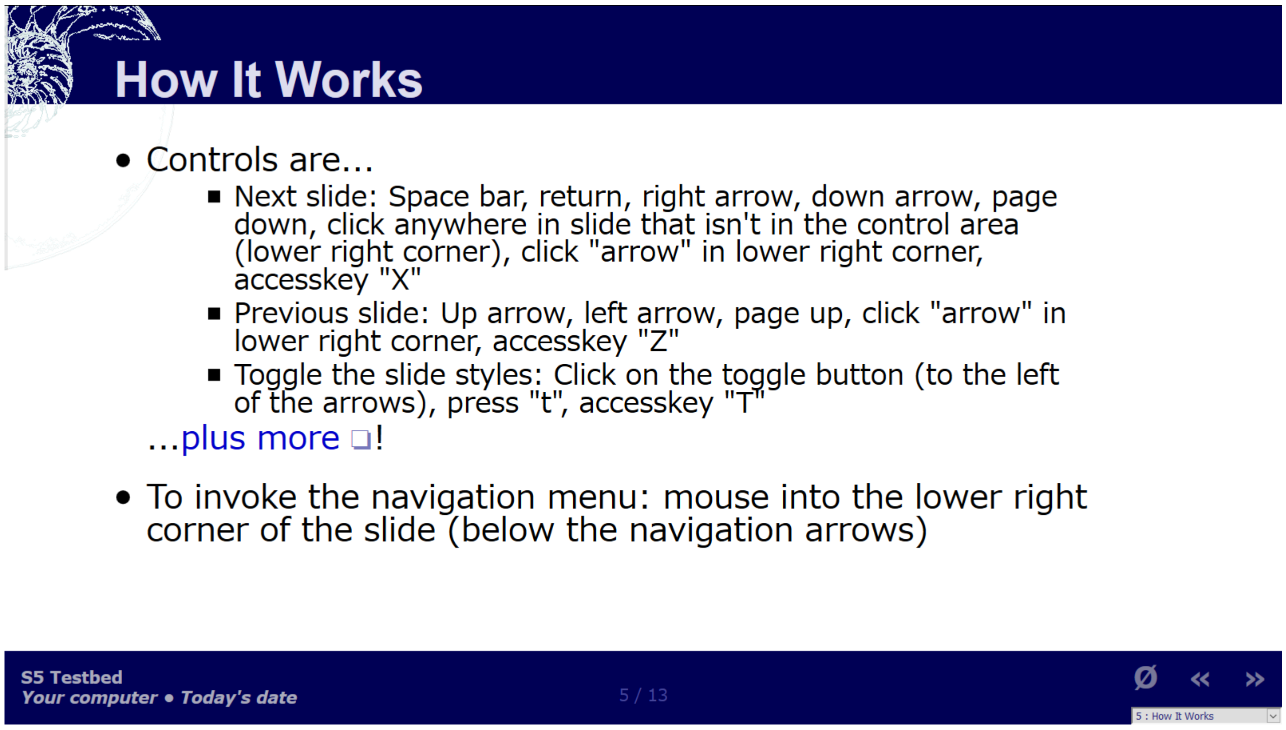
has class="incremental" or if itself has that attribute. Text nodes are not elements and are revealed when their parent element is revealed.
r <br />. Use class="non-incremental" to override the effect of setting the parent element's class to incremental.

r when there is still something more to reveal.

help? contents? Copyright © 2005-2013 W3C ® (MIT, ERCIM, Keio, Beihang)

4m2s slide 9/20

**Figure 2.1:** Slidy2 features an auto-generated table of contents, which is displayed by pressing `T` on the keyboard or by clicking the `contents?` link in the footer. [Screenshot made by the author from Slidy2 [Raggett 2006a].]



How It Works

- Controls are...
  - Next slide: Space bar, return, right arrow, down arrow, page down, click anywhere in slide that isn't in the control area (lower right corner), click "arrow" in lower right corner, accesskey "X"
  - Previous slide: Up arrow, left arrow, page up, click "arrow" in lower right corner, accesskey "Z"
  - Toggle the slide styles: Click on the toggle button (to the left of the arrows), press "t", accesskey "T"

  ...plus more ❏!
- To invoke the navigation menu: mouse into the lower right corner of the slide (below the navigation arrows)

S5 Testbed
*Your computer • Today's date*

**Figure 2.2:** S5 slides can be navigated by keyboard and mouse controls. The on-screen buttons and drop-down menu only become visible when hovering over the bottom right half of the footer. [Screenshot made by the author from S5 [Meyer 2005b].]

## 2.2 S5

S5 stands for **S**imple **S**tandards-based **S**lide **S**how **S**ystem and was developed by Eric Meyer [2005a] in 2005. Its features, code, and usage requirements are similar to Slidy and some additional slide themes are provided. A source code example of S5 is shown in Listing 2.2. Navigation is handled through keyboard shortcuts, mouse clicks, and a footer containing buttons and a drop-down menu, which only becomes visible when moving the mouse pointer near the bottom right half of the screen, as shown in Figure 2.2.

Navigation on mobile devices is extremely tedious. Fortunately, touch events in the browser automatically call equivalent mouse events, which makes it possible to advance slides in S5. However, in order to navigate to a previous slide the footer still needs to be accessed, which is difficult without mouse hover.

## 2.3 S6

Development of S6 started by Gerald Bauer [2013] in the form of a rewrite of S5 focusing on easy extensibility through plugins. It makes use of the popular JavaScript library JQuery. It supports both classic and modern HTML elements. S6 is one of the formats generated by the Ruby command line tool S9, a text-based slide deck solution discussed in Section 3.1. A code example of S6 is shown in Listing 2.2.

```
<div class="slide">
  <h1>Navigation</h1>

  <ul>
    <li>Next slide: Space bar, return, right arrow...</li>
    <li>Previous slide: Up arrow, left arrow, page up...</li>
    <li>Toggle the slide styles: Click on the toggle button...</li>
  </ul>
</div>
```

**Listing 2.2:** An example slide in S5 and S6. S6 also supports `section` instead of `div`.

## 2.4 Diascope

In 2006, Martin Stoll created Diascope [Stoll 2016b], a slideshow generator for Linux which uses video encoding to render presentations. Diascope comes in the form of a binary executable and focuses primarily on smooth transitions, panning, zooming, and audio effects. The output of diascope is an MPEG2, MPEG4 or FLV video file ready for playing or embedding on a web page.

A short code example of a diascope presentation can be seen in Listing 2.3. It is a very powerful tool, but requires knowledge of video editing software and even small changes to the code require a re-rendering of the output video file, making it unsuitable for inexperienced users.

## 2.5 Flash Presentations

Adobe Flash [Chun 2014] is a powerful tool for creating standalone interactive multimedia applications which can be embedded into a web site. Following its release in 1996, it became omnipresent on the web, with many of the more impressive looking websites making use of it to some degree. With support for mouse and keyboard interactions and inbuilt tools to easily create animations from keyframes, Adobe Flash can be used to create slideshows, which can be displayed on a web page with the flash plug-in. Such slideshows can also be exported as flash video for embedding in a web page.

In 2010 Steve Jobs [2010] described the shortcomings of Flash in a blog post titled "Thoughts on Flash":

> “Flash was created during the PC era – for PCs and mice. Flash is a successful business for Adobe, and we can understand why they want to push it beyond PCs. But the mobile era is about low power devices, touch interfaces and open web standards – all areas where Flash falls short. [· · · ] New open standards created in the mobile era, such as HTML5, will win on mobile devices (and PCs too).”

Shortly after, Apple announced that its products would no longer support Flash. Most developers today have switched away from Flash, and it will be officially retired in 2020 [Barrett 2017].

```
format pal quality=1 interlaced mpeg2sound=mp2 mpeg2
base template_hq_i /tmp
set resize=resize
set font=PenguinAttack size=78 fill=black strokewidth=0

set dur=sec

# audio mysong.wav fade=0,2

# Title slide

create 2 white title=0,1,1,0 "Diascope\nSlideshow\nGenerator"
trns 0.3 luma dir=ro

set font=ArialB fill=gray stroke=white strokewidth=1

# Main show

create 0.8 blue title=0.8 "diascope.sf.net"
trns 0.4 luma dir=td sharp=1
create 0.8 red title=0.8 "diascope.sf.net"
trns 0.4 luma dir=td sharp=1
create 0.8 yellow title=0.8 "diascope.sf.net"
trns 0.4 luma dir=td sharp=1
create 0.8 blue title=0.8 "diascope.sf.net"

trns 0.3 luma dir=ri
audio silence

# End

create 1 black
```

**Listing 2.3:** An excerpt from a simple diascope source file [Stoll 2016a]. First, rendering parameters are set, followed by some basic slides with transitions.

# Chapter 3

# Text-Based Slide Decks

Unlike other approaches, text-based slide decks do not require slide creators to possess any knowledge of HTML, CSS (styling), or JavaScript (scripting). Slides are created by editing plain text files [Unicode 2018] or markdown files [Gruber 2004], which are then turned into a web-based slideshow by an application. Markdown is a syntax for structuring plain text to be converted to HTML or other formats. The main advantage of this approach is the fast and easy way to write and edit slides, but the trade-off is less flexibility in terms of styling and interaction.

## 3.1 Slide Show (S9)

S9 is a command line tool written in Ruby by Gerald Bauer [2011]. It can turn a markdown file into a variety of different slide decks, such as Slidy (Section 2.1), S5 (Section 2.2), and S6 (Section 2.3), among others. An example slide is shown in Listing 3.1. The user has to download and install template packs for each of these output formats, and can choose from a variety of theme packs. To install the S6 template pack, the command:

```
$ slideshow install s6blank
```

is run from the command line. A list of installed templates can be viewed with `slideshow list`. To build a slide show from a file titled `slides.text`, the following command has to be executed:

```
$ slideshow build slides.text -t s6blank

=> Preparing slideshow 'slides.html'...
=> Done.
```

More experienced slide creators can include a variety of plugins and helpers in their presentations, or write their own scripts using Ruby. These can be used, for example, to provide syntax highlighting. It is also possible to create or edit templates and themes.

## 3.2 Slidedown

Slidedown [Nakajima and Croak 2012] is a very lightweight and minimalistic Ruby tool for creating basic slideshows from markdown. A slideshow is generated by running the command:

```
$ slidedown slides.md > slides.html
```

The created presentations only support barebones keyboard controls: the left and right arrow keys are used for navigation and pressing escape exits the presentation. Custom style sheets are automatically included, if

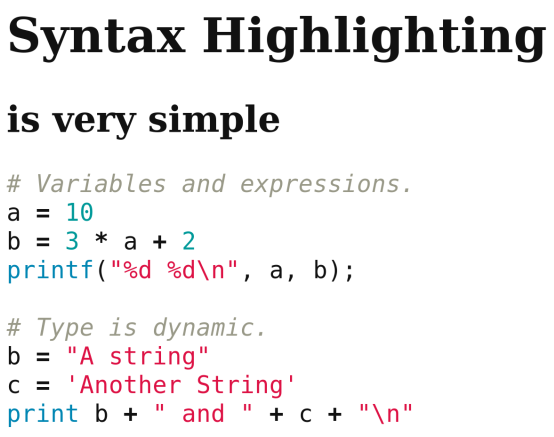
```
# Slide Heading

- First bullet point
- Second bullet point
- Third bullet point
```

**Listing 3.1:** An example slide in S9 using Markdown. # is used for headings and - or * for lists.

```
!SLIDE code

# Syntax Highlighting

## is very simple

@@@ ruby
    # Variables and expressions.
    a = 10
    b = 3 * a + 2
    printf("%d %d\n", a, b);

    # Type is dynamic.
    b = "A string"
    c = 'Another String'
    print b + " and " + c + "\n"
@@@
```

**Listing 3.2:** In Slidedown, Syntax highlighting is performed on code placed inside @@@ blocks. After the block definition, the programming language can be specified.

they are placed in the same directory as the presentation markdown file, and support for syntax highlighting is provided. Some example code and the resulting slide are shown in Listing 3.2 and Figure 3.1.



Syntax Highlighting

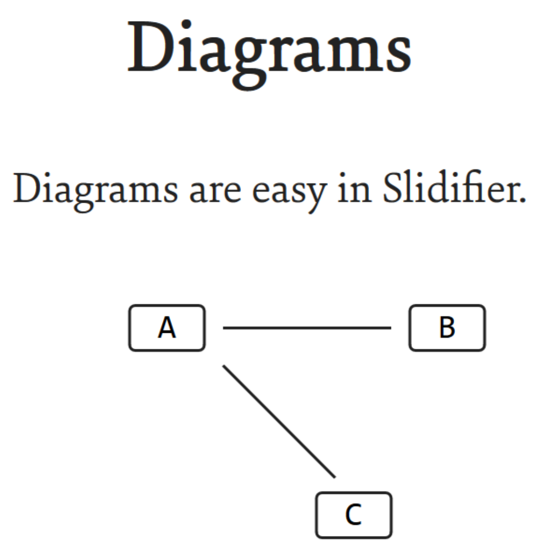
is very simple

```
# Variables and expressions.
a = 10
b = 3 * a + 2
printf("%d %d\n", a, b);

# Type is dynamic.
b = "A string"
c = 'Another String'
print b + " and " + c + "\n"
```

**Figure 3.1:** The code from Listing 3.2 turned into a slide with Slidedown. [Screenshot made by the author from Slidedown [Nakajima and Croak 2012].]

```
---

# Diagrams

Diagrams are easy in Slidifier.

\\diagram
  [ A ]*1       1*[ B ]
         *2



               *2
              [ C ]
\\diagram

---
```

**Listing 3.3:** Source code for a slide with a diagram in Slidifier. Connecting lines are assigned numbers and connected using the * character; labels are enclosed in square brackets.



Diagrams

Diagrams are easy in Slidifier.

**Figure 3.2:** The output slide, generated by Slidifier from the source code in Listing 3.3. [Screenshot made by the author from Slidifier [Ludvigsen 2016b].]

## 3.3 Slidifier

Rather than relying solely on markdown, Slidifier [Ludvigsen 2016a] defines its own simple plain text syntax. A line containing `---` is used to separate slides and inline HTML can be used extend the plain text, for example to include or embed an image. A unique feature of Slidifier is its inbuilt support for creating simple diagrams. The example in Listing 3.3 and Figure 3.2 illustrates this. A primitive form of keyword highlighting is supported as well.

## 3.4 Remark

Remark [Bang 2018b] uses a mixture of HTML and markdown and provides many advanced features, such as a synchronised display, speaker notes, and a presenter mode, as shown in Listing 3.4 and Figure 3.3. Three dashes separate slides and three question marks introduce speaker notes. In presenter mode, a timer and speaker notes are displayed on the right and the current and next slide are displayed on the left. Remark supports touch gestures such as swipe to navigate and slide scaling for various screen sizes. There are some external tools for Remark, most noteworthy a PDF generator and an online "what you see is what you get" editor.

A Remark input file starts in HTML and allows for styles to be defined in the HTML header. The body then contains a `<textarea id="source">` tag, in which the markdown text for slides is placed. Afterwards, Remark is called from within a script element as follows:

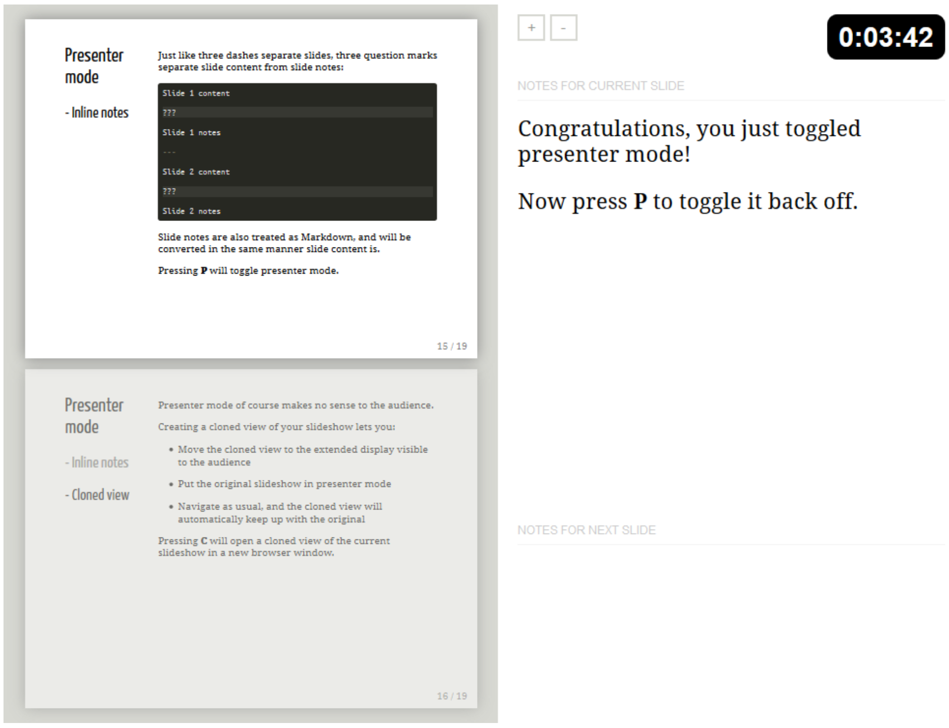
```
var slideshow = remark.create();
```

Remark further extends the markdown syntax with slide properties. These can be placed at the beginning of a slide using a `key: value` syntax. The properties are:

- `name`: Assigns a name to a slide in order to link to it directly using the browser window's hash after the URL.
- `class`: Adds a comma-separated list of CSS classes to the slide.
- `background-image`: Sets a background image for the specific slide.
- `count`: Slides where the count property is set to false will not influence the total number of slides. All incremental slides set this property to false by default.
- `template`: This property can be used to inherit the properties of another slide.
- `layout`: If set to true, the slide properties will be carried over to all subsequent slides, until layout is set to false again.
- `exclude`: Fully excludes a file from rendering.

```
<textarea id="source">
---
.left-column[
  ## Presenter mode
  ### - Inline notes
]
.right-column[
Just like three dashes separate slides,
three question marks separate slide content from slide notes:

Slide notes are also treated as Markdown, and will be converted in the
same manner slide content is.

Pressing __P__ will toggle presenter mode.
]
???
Congratulations, you just toggled presenter mode!

Now press __P__ to toggle it back off.
---
</textarea>
```

**Listing 3.4:** In Remark, classes can be applied by `.class[]`, such as `.right-column[]`, and speaker notes are defined after three question marks.



## Presenter mode

- Inline notes

Just like three dashes separate slides, three question marks separate slide content from slide notes:

```
Slide 1 content
???
Slide 1 notes
---
Slide 2 content
???
Slide 2 notes
```

Slide notes are also treated as Markdown, and will be converted in the same manner slide content is.

Pressing **P** will toggle presenter mode.

15 / 19

## Presenter mode

- Inline notes

- Cloned view

Presenter mode of course makes no sense to the audience.

Creating a cloned view of your slideshow lets you:

- Move the cloned view to the extended display visible to the audience
- Put the original slideshow in presenter mode
- Navigate as usual, and the cloned view will automatically keep up with the original

Pressing **C** will open a cloned view of the current slideshow in a new browser window.

16 / 19

NOTES FOR CURRENT SLIDE

Congratulations, you just toggled presenter mode!

Now press **P** to toggle it back off.

NOTES FOR NEXT SLIDE

**Figure 3.3:** Pressing P in Remark toggles presenter mode. A separate synchronised window (called a cloned view) can be opened by pressing C, which can then be displayed on a projector. [Screenshot made by the author from Remark.js [Bang 2018a].]

# Chapter 4

# JavaScript-Based Slide Decks

Slide decks built with JavaScript offer a broad spectrum of functionality, rely heavily on scripting, and often make use of third-party packages. Many of these slide decks are listed in a public collection [NPM 2018] of open source packages for Node.js, and can be easily installed using a package manager. These packages are modern and feature-rich, but there are also some minimalistic slide decks which try to reduce the number of external dependencies. A possible downside to this approach is that the scripts are often computationally expensive.

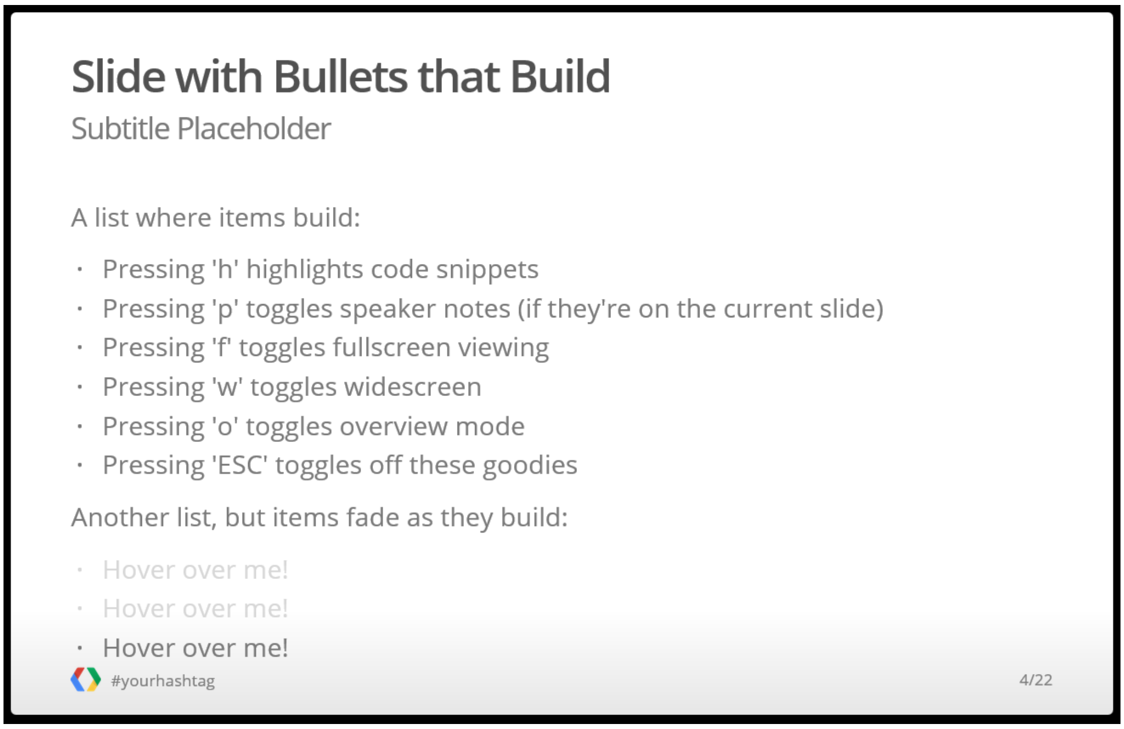
## 4.1 Google I/O Slides

Google I/O Slides [Mahém et al. 2013] is an HTML template developed for Google I/O in 2011. Originally, it went under the name html5slides. It supports many of the standard features of slide decks such as keyboard navigation, syntax highlighting, and touch controls. A list of special keyboard shortcuts can be seen in Figure 4.1. A file called `slide_config.js` is used to configure the slide template, including title, fonts, themes, authors, and various options. The template makes use of external tools such as compass, a Ruby tool for converting SASS into CSS, and Python to convert markdown into HTML. Slide content is placed inside `<slide>` elements.

Development was abandoned in 2015, likely in favour of Google Slides (see Section 5.1), a hosted slide deck approach. At the time this survey was written, all the repositories for Google I/O Slides have been either partially or fully removed, but a fork of the project is still available on GitHub [Rota 2015].

## 4.2 Fathom.js

Fathom.js [Dalgleish 2015] uses the jQuery Mobile library to optimise slide decks for mobile devices. It provides a list of event handlers for user actions like tap, swipe left, swipe right, scroll start, scroll stop, and orientation change. A special feature of Fathom.js is the ability to synchronise a slideshow to an external video of a presentation. This is achieved by providing a timeline of slide transitions and a video object, as can be seen in Listing 4.1. The project page is no longer maintained and recommends using Bespoke.js (see Section 4.5).



Slide with Bullets that Build

Subtitle Placeholder

A list where items build:

- Pressing 'h' highlights code snippets
- Pressing 'p' toggles speaker notes (if they're on the current slide)
- Pressing 'f' toggles fullscreen viewing
- Pressing 'w' toggles widescreen
- Pressing 'o' toggles overview mode
- Pressing 'ESC' toggles off these goodies

Another list, but items fade as they build:

- Hover over me!
- Hover over me!
- Hover over me!

#yourhashtag 4/22

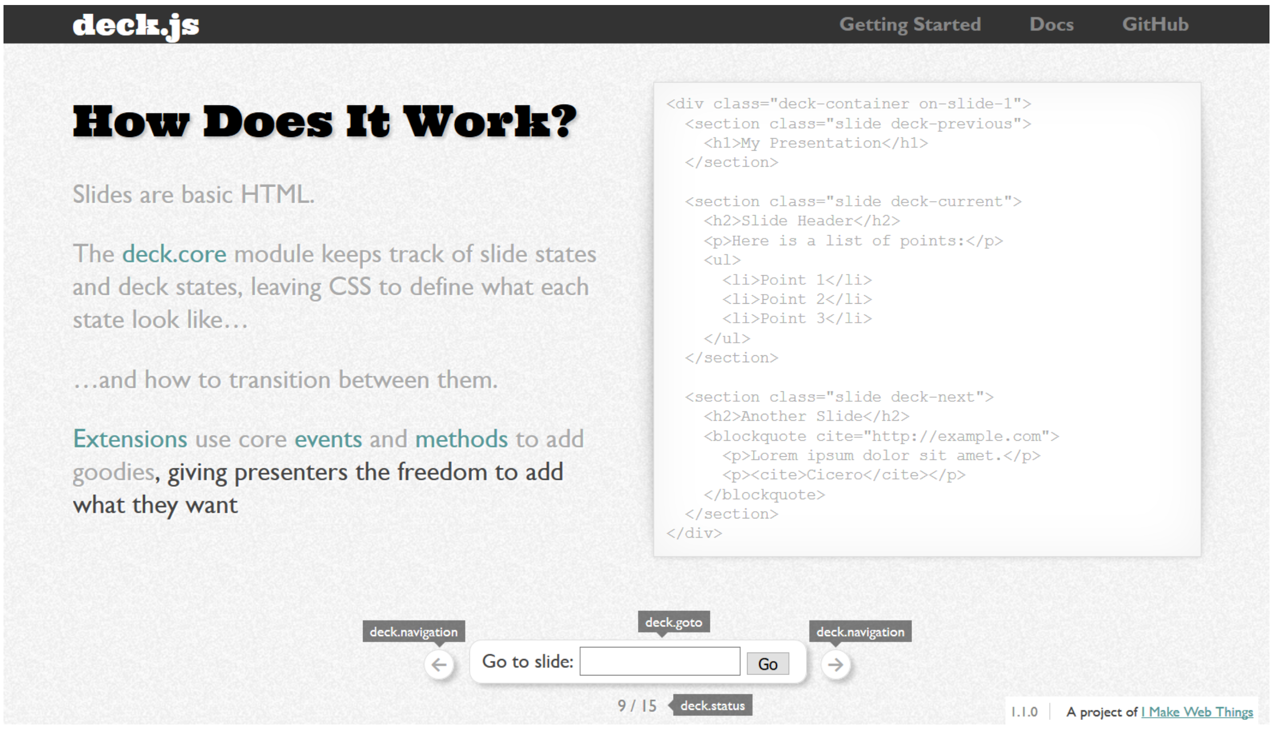
**Figure 4.1:** As well as standard navigation, Google I/O Slides supports keyboard shortcuts for visual alterations. [Screenshot made by the author from Google I/O Slides [Rota 2015].]

```
$('#presentation').fathom({
  timeline: [ 0, 5, 20, 30, 40, 50, '1:00', '1:15', 90, 120, 155 ],
  video: {
    source: 'vimeo',
    id: '16917950',
    parent: '#vimeo',
    autoplay: true
  }
});
```

**Listing 4.1:** Fathom.js can synchronise a slide deck to an external video using a timeline of slide transitions. The timeline parameter supports both numbers and strings.

## 4.3 Deck.js

Developed by Caleb Troughton [2016a], Deck.js is a modular and flexible JavaScript framework for generating customised slide decks. It is based on jQuery and Modernizr. The framework includes a file named `boilerplate.html` which serves as a starting point for slide creators. The main functionality is provided by the `deck.core` module, which defines various states for the slide deck that can be customised using CSS. Additional functionality is provided by extensions, which use core events and methods in various ways. An example of a navigation bar extension can be seen in Figure 4.2 and Listing 4.2. Using both the core module and extensions, slide creators can customise their presentation to include what they need and leave out what they do not. Deck.js has in-depth documentation [Troughton 2016b] and a wiki page [Troughton 2016d], where people can share their themes, extensions, and tools.



How Does It Work?

Slides are basic HTML.

The deck.core module keeps track of slide states and deck states, leaving CSS to define what each state look like…

…and how to transition between them.

Extensions use core events and methods to add goodies, giving presenters the freedom to add what they want

```
<div class="deck-container on-slide-1">
  <section class="slide deck-previous">
    <h1>My Presentation</h1>
  </section>

  <section class="slide deck-current">
    <h2>Slide Header</h2>
    <p>Here is a list of points:</p>
    <ul>
      <li>Point 1</li>
      <li>Point 2</li>
      <li>Point 3</li>
    </ul>
  </section>

  <section class="slide deck-next">
    <h2>Another Slide</h2>
    <blockquote cite="http://example.com">
      <p>Lorem ipsum dolor sit amet.</p>
      <p><cite>Cicero</cite></p>
    </blockquote>
  </section>
</div>
```

**Figure 4.2:** An extension for Deck.js provides a navigation bar, which interfaces to the core module's navigation, goto, and status functionality. [Screenshot made by the author from Deck.js [Troughton 2016c].]

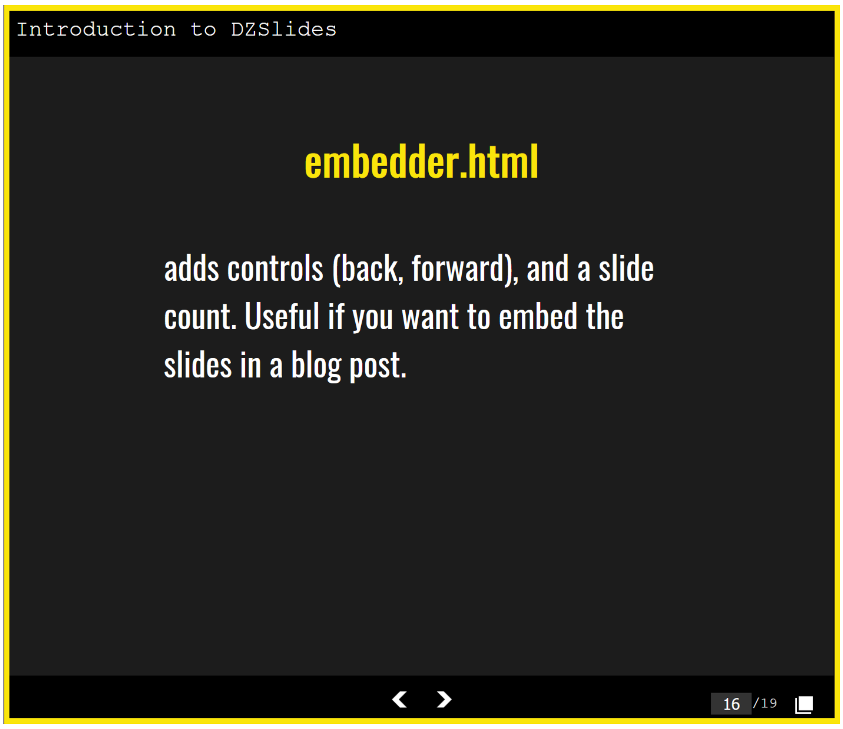
## 4.4 DZSlides

DZSlides [Rouget 2017a] is a single, ready-to-use HTML file template for creating slideshows. Both the CSS and JavaScript are already included in the file, making it easier to handle for slide creators. By using so-called *shells*, the presentation can be embedded or used in a presenter mode. An example of embedded mode can be seen in Figure 4.3. These shells are separate HTML files which link together with the original presentation. In order for these extensions to function, DZSlides implements a messaging system. The main template file can either receive or send a set of predefined messages. These include simple commands such as navigation, but also more advanced commands which can be used to register events or obtain the mouse cursor position.

The slides automatically adjust their scaling along with the browser window, but they are internally fixed at a virtual resolution of `800x600`, making the presentation framework not fully responsive. Basic keyboard navigation is supported along with mobile-friendly touch controls.

```
<section class="slide">
  <h1>Slide</h1>

  <ul>
    <li>First bullet point</li>
    <li>Second bullet point</li>
    <li>Third bullet point</li>
  </ul>
</section>

<!-- Core Module (Required) -->
<script src="core/deck.core.js"></script>

<!-- Extensions -->
<script src="extensions/menu/deck.menu.js"></script>
<script src="extensions/goto/deck.goto.js"></script>
<script src="extensions/status/deck.status.js"></script>
<script src="extensions/navigation/deck.navigation.js"></script>
<script src="extensions/scale/deck.scale.js"></script>

<!-- Finally, initialise the deck. -->
<script>
  $(function() { $.deck('.slide'); });
</script>
```

**Listing 4.2:** An example slide setup in Deck.js. After including the core module, additional functionality can be added through extensions.



Introduction to DZSlides

embedder.html

adds controls (back, forward), and a slide count. Useful if you want to embed the slides in a blog post.

**Figure 4.3:** In DZSlides, the `embedder.html` shell embeds the presentation into another web page and adds navigation controls in the form of a footer. The rightmost button on the footer opens a pop-up window to the embedded slideshow. [Screenshot made by the author from DZSlides [Rouget 2017b].]

```
// HTML Body

<article id="presentation">
  <section>Slide 1</section>
  <section>Slide 2</section>
  <section>Slide 3</section>
</article>

// JavaScript

var bespoke = require('bespoke'),
  cube = require('bespoke-theme-cube'),
  keys = require('bespoke-keys'),
  touch = require('bespoke-touch');

var deck = bespoke.from('#presentation', [
  cube(),
  keys(),
  touch()
]);
```

**Listing 4.3:** Bespoke.js is a modular slide deck library, which uses a plugin ecosystem to provide additional functionality. In this example, a theme and two plugins are loaded.

## 4.5 Bespoke.js

Bespoke.js [Dalgleish 2018a] is a minimalist modular presentation library. The main package only provides very basic functionality and a simple control API, and everything else is managed via plugins. Because of the fragmented nature of the package, it is recommended to use a build tool such as gulp to pull all the required plugins together and generate the presentation. Currently, 21 themes and 75 plugins are available. For example, bespoke-touch adds touch controls and bespoke-math adds support for LaTeX formulas. A custom theme can be seen in Figure 4.4

To include a theme or a plugin, a simple `require('...')` command is needed to load the package, followed by a call to its constructor in the plugin array of Bespoke's main function. Plugins interact with the API of Bespoke, which contains a number of predefined variables and helper functions for navigation and events. Listing 4.3 shows some example code for generating a presentation with the cube theme and support for keyboard and touch controls.

## 4.6 Inspire.js

Formerly known as the CSS-based SlideShow System (CSSS), Inspire.js [Verou 2018a] aims to be a lean and minimalistic slideshow tool focusing on reusability and extendibility using automatically loaded JavaScript extensions. The renaming of the tool took place in September 2018 and is currently still ongoing. Unique features of Inspire.js are the ability to reuse a previously defined slide by its id, support for annotated videos, and support for live code examples. An example of video annotation can be seen in Listing 4.4 and Figure 4.5.

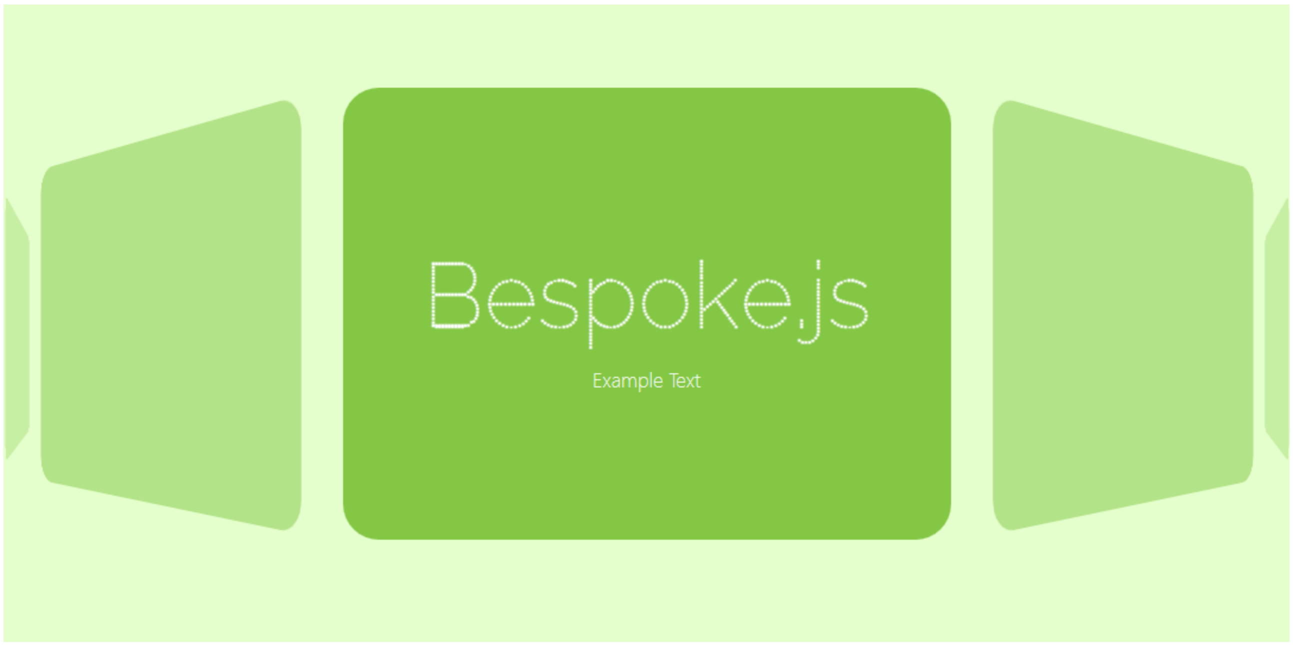
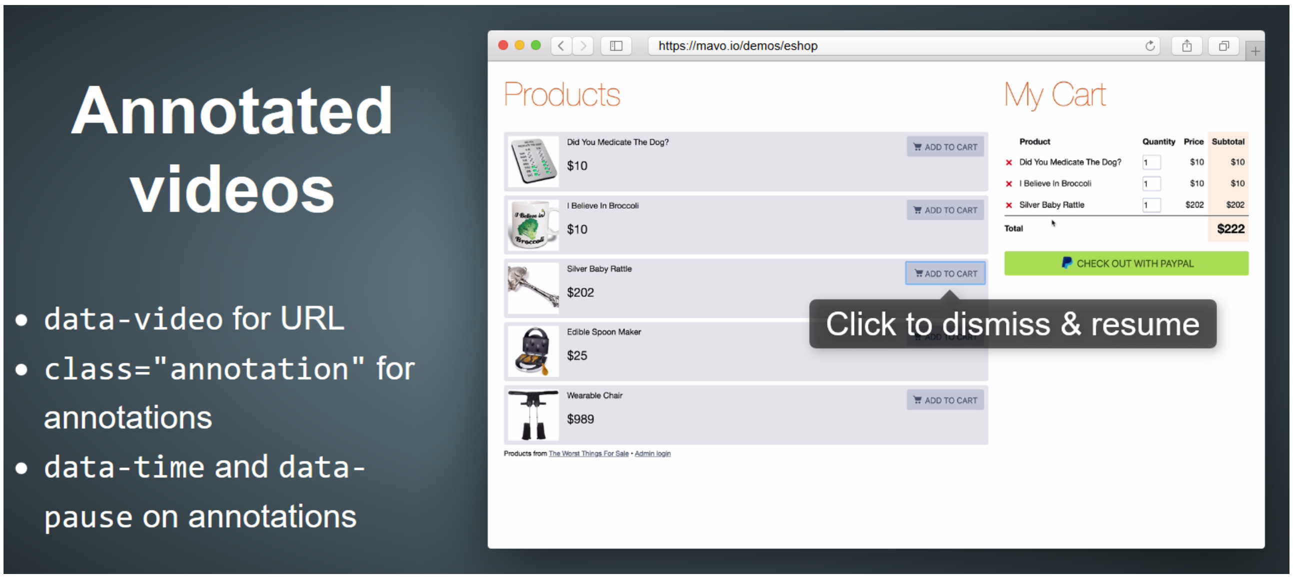


**Figure 4.4:** Bespoke.js currently features 21 custom themes. The theme here is `carousel`. [Screenshot made by the author from Bespoke.js [Dalgleish 2018b].]

```
<article class="slide" data-video="https://mavo.io/demos/eshop/video.mp4" style="--
    url: 'https://mavo.io/demos/eshop'" id="media-plugin">
  <div>
    <h1>Annotated videos</h1>
    <ul>
      <li><code>data-video</code> for URL</li>
      <li><code>class="annotation"</code> for annotations</li>
      <li><code>data-time</code> and <code>data-pause</code> on annotations</li>
    </ul>
  </div>
  <span class="annotation top-pointer" style="top: 44%; left: 59.5%; --pointer-left:
      4em;" data-time="3000 to 3500" data-pause>Click to dismiss & resume</span>
  <span class="annotation top-pointer" style="top: 26%; left: 67.1%" data-time="5200
      to 5700" data-pause="2000">Auto-resume after 2sec</span>
</article>
```

**Listing 4.4:** Inspire.js uses the `annotation` class with the attributes `data-time` or `data-pause` to display video annotations.



Annotated videos

- data-video for URL
- class="annotation" for annotations
- data-time and data-pause on annotations

https://mavo.io/demos/eshop
Products
Did You Medicate The Dog?
$10
ADD TO CART
I Believe In Broccoli
$10
ADD TO CART
Silver Baby Rattle
$202
ADD TO CART
Edible Spoon Maker
$25
Wearable Chair
$989
ADD TO CART
Products from The Worst Things For Sale • Admin login
My Cart
Product
Quantity
Price
Subtotal
Did You Medicate The Dog? 1 $10 $10
I Believe In Broccoli 1 $10 $10
Silver Baby Rattle 1 $202 $202
Total $222
CHECK OUT WITH PAYPAL
Click to dismiss & resume

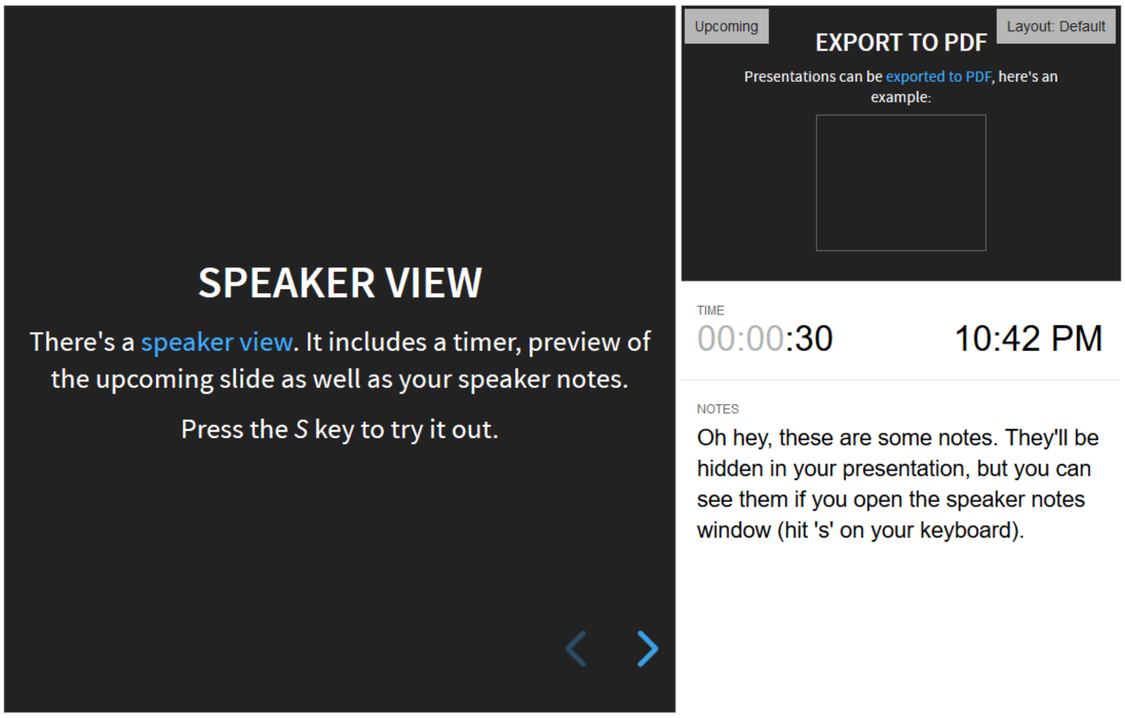
**Figure 4.5:** In Inspire.js, annotations temporarily pause a video to show text, which can either automatically disappear after a set time or be dismissed by mouse click. [Screenshot made by the author from Inspire.js [Verou 2018b].]

## 4.7 Reveal.js

With a broad range of functionality and elegance, Reveal.js [Hattab 2018a] is a highly advanced presentation framework. It supports vertically nested slides, markdown content, keyboard and touch controls, PDF export, speaker notes, presenter mode, math functions, lazy loading, and more. An example presentation in presenter mode can be seen in Figure 4.6. Many of these features are provided by third party scripts, for example markdown interpretation and syntax highlighting. Slideshows using Reveal.js can also be created and published through an online platform [Slides 2018] which provides a graphical slide editor.

Reveal.js can be configured both during initialisation and at runtime and comes with a long list of options for each of its features. Configuration is entirely optional and hides some interesting functionality. For cultures which write and read from right to left, the direction of the presentation can be configured. Even keyboard shortcuts can be remapped manually, as shown in Listing 4.5. A full list of these options are provided in the manual, the most important ones are:

- `controls`: If set to true, displays the arrow controls.
- `history`: If set to true, pushes slide changes into the browser history.
- `loop`: If set to true, loops the presentation.
- `autoSlide`: Automatically advances a slide after a set number of milliseconds.
- `rtl`: If set to true, changes the presentation direction to right-to-left.
- `previewLinks`: Adds an iframe preview window for links.
- `transition`: Changes the transition style between `none`, `fade`, `slide`, `convex`, `concave` or `zoom`.


SPEAKER VIEW

There's a speaker view. It includes a timer, preview of the upcoming slide as well as your speaker notes.

Press the *S* key to try it out.

EXPORT TO PDF

Presentations can be exported to PDF, here's an example:

TIME
00:00:30 10:42 PM

NOTES
Oh hey, these are some notes. They'll be hidden in your presentation, but you can see them if you open the speaker notes window (hit 's' on your keyboard).

**Figure 4.6:** Reveal.js calls its presenter mode feature "Speaker View". It shows the current and next slide, speaker notes, and a timer. [Screenshot made by the author from Reveal.js [Hattab 2018b].]

```
<section>
  <h1>Slide</h1>

  <ul>
    <li>First bullet point</li>
    <li>Second bullet point</li>
    <li>Third bullet point</li>
  </ul>
</section>

<script>
  Reveal.initialize({
    keyboard: true,
    rtl: true,
    transition: 'slide',
    transitionSpeed: 'fast'
  });

  Reveal.configure({
    keyboard: {
      13: 'next',
      27: function() {},
      32: null
    }
  });
</script>
```

**Listing 4.5:** The example code above initialises Reveal.js in right-to-left mode and remaps some keyboard shortcuts. To unmap defaults, set a shortcut to `null`.

```
<section class="slide">
  <h1>Key features</h1>
  <ul>
    <li>Built on HTML, CSS and JavaScript</li>
    <li>Works in all modern browsers</li>
  </ul>
</section>
```

**Listing 4.6:** An example slide in Shower.

## 4.8 Shower

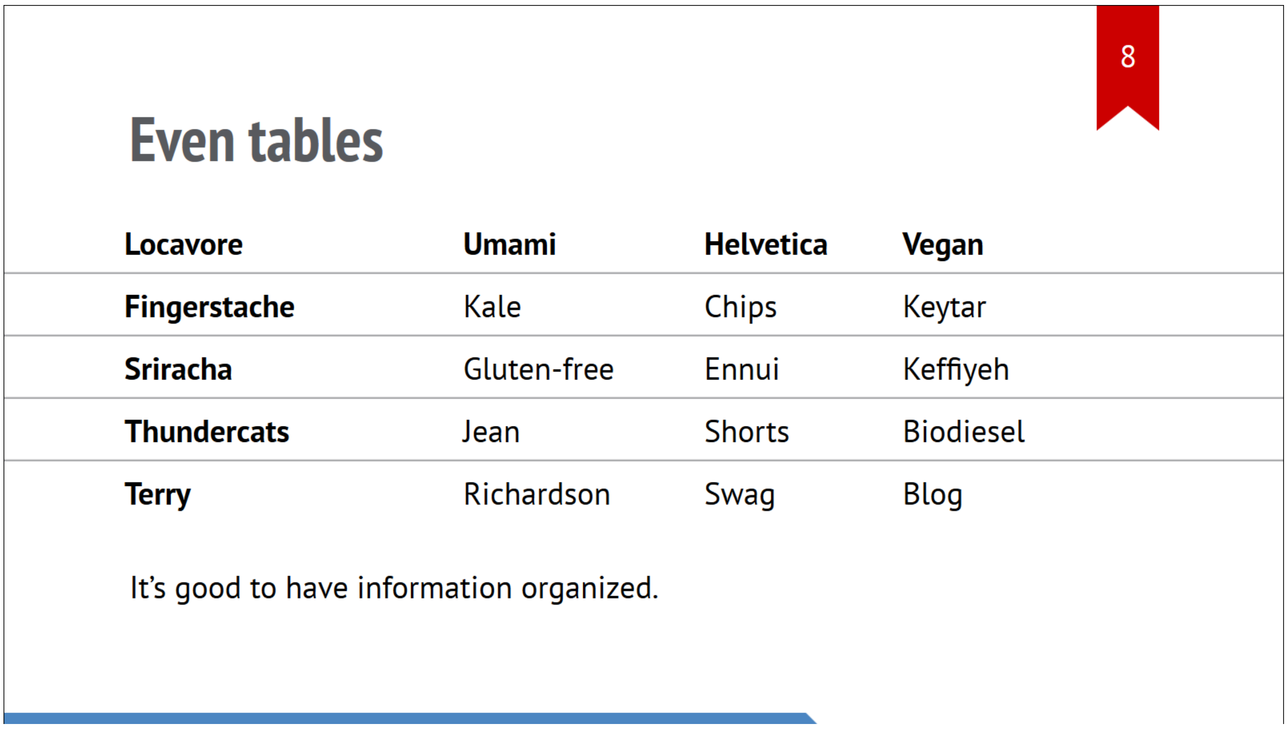
Resembling the look of PowerPoint, Shower [Makeev 2018a] tries to recreate classical printable slideshows. An example slide can be seen in Figure 4.7. Unlike most other slide decks, Shower also provides a means for slide creators to deploy and host their presentations using Netlify [2018], in addition to being able to self-host. This automatically triggers a rebuild when shower is updated and no extra tools need to be installed for hosting the presentation.

The tool comes bundled with a "template archive" containing the two main themes, Ribbon and Material, and the core files. Shower can also be downloaded using npm. This includes a gulp configuration file with predefined tasks: `prepare`, which builds the slide deck, and `publish`, which handles the upload process to Netlify. An example slide is shown in Listing 4.6.

## 4.9 Impress.js

Impress.js [Szopka and Ingo 2018a] is a presentation framework unlike traditional slide decks. It is heavily inspired by Prezi (see Section 5.2) and focuses on stunning transitions and animations, which are made possible by CSS3 transformations. Impress.js supports translation, scaling, and rotation in both 2D and 3D. The entire presentation is placed on an infinite canvas, which is then traversed by a camera. An example of its bird's-eye view can be seen in Figure 4.8.

In order to achieve this, every slide has to provide additional parameters for position, rotation, and scaling as HTML data attributes, which can be tedious to maintain. An example can be seen in Listing 4.7. Impress.js, as the name implies, is designed to impress with stunning visuals, and is therefore computationally expensive. It is intended to be presented on a powerful computer with a modern browser and does not guarantee mobile device support. Impress.js does not rely on third party packages, but a jQuery port called Jmpress.js [Young and Koppers 2016] is available.



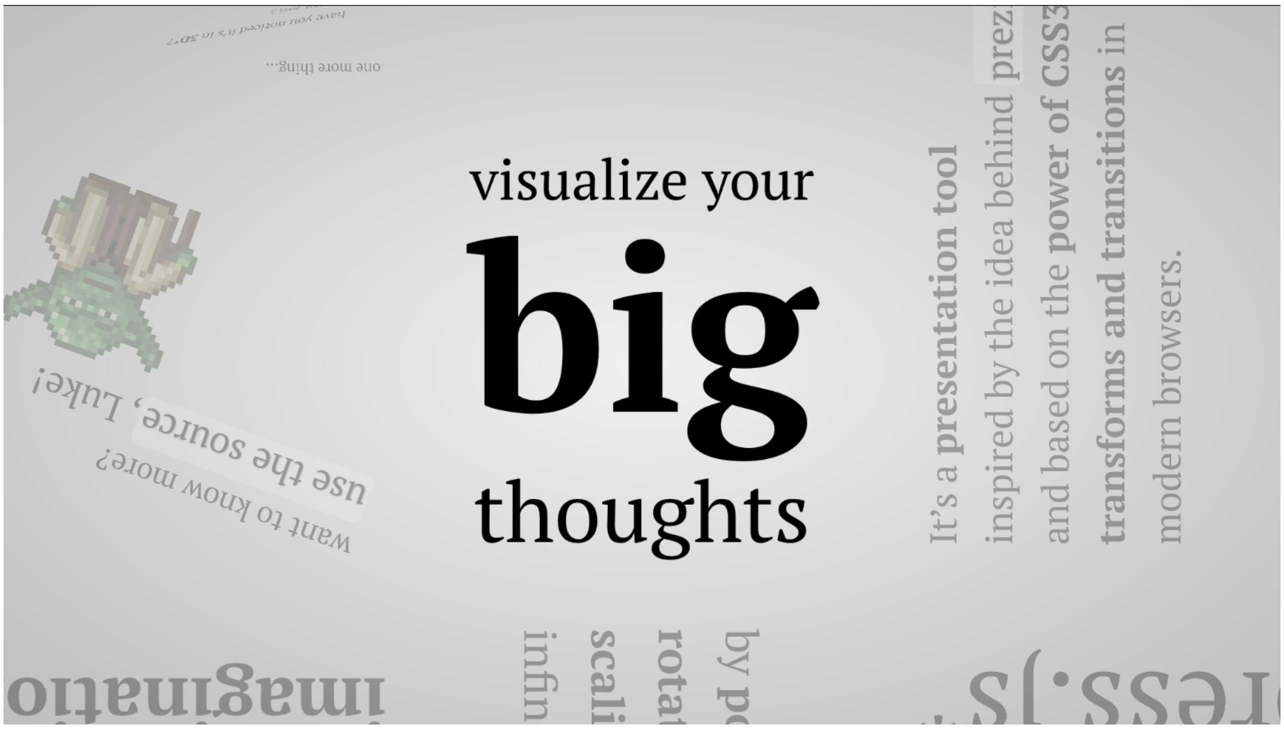
8

## Even tables

| Locavore | Umami | Helvetica | Vegan |
|---|---|---|---|
| **Fingerstache** | Kale | Chips | Keytar |
| **Sriracha** | Gluten-free | Ennui | Keffiyeh |
| **Thundercats** | Jean | Shorts | Biodiesel |
| **Terry** | Richardson | Swag | Blog |

It’s good to have information organized.

**Figure 4.7:** An example slide with stylised tables in Shower. The progress bar is visible at the bottom and the current slide number can be seen in the upper right corner. [Screenshot made by the author from Shower [Makeev 2018b].]



visualize your

big

thoughts

It’s a **presentation tool**
inspired by the idea behind prezi
and based on the **power of CSS3**
**transforms and transitions** in
modern browsers.

**Figure 4.8:** Impress.js places all the slides on an infinite canvas and then traverses it with a camera. [Screenshot made by the author from Impress.js [Szopka and Ingo 2018b].]

```
<div class="step" data-x="6200" data-y="4300" data-z="-100" data-rotate-x="-40"
     data-rotate-y="10" data-scale="2">
  <ul>
    <li>First bullet point</li>
    <li>Second bullet point</li>
    <li>Third bullet point</li>
  </ul>
</div>
```

**Listing 4.7:** In Impress.js animations are handled with HTML data attributes. The position of the centre of a slide is defined by `data-x`, `data-y`, and `data-z`. Rotation can be adjusted with `data-rotate-x`, `data-rotate-y`, and `data-rotate-z`. The scale is defined by `data-scale` and defaults to 1.

# Chapter 5

# Hosted Slide Decks

Neatly packed into a web application, hosted slide decks take a drastically different approach from those previously covered in this survey. They not only host the finished presentations for viewing, but also the tools for editing them. While offering some unique advantages, this approach also has its downsides. Some of the tools listed in this section require a monthly subscription fee. Often, the slide decks can only be viewed with a stable internet connection, offline access and the ability to self-host is not provided.

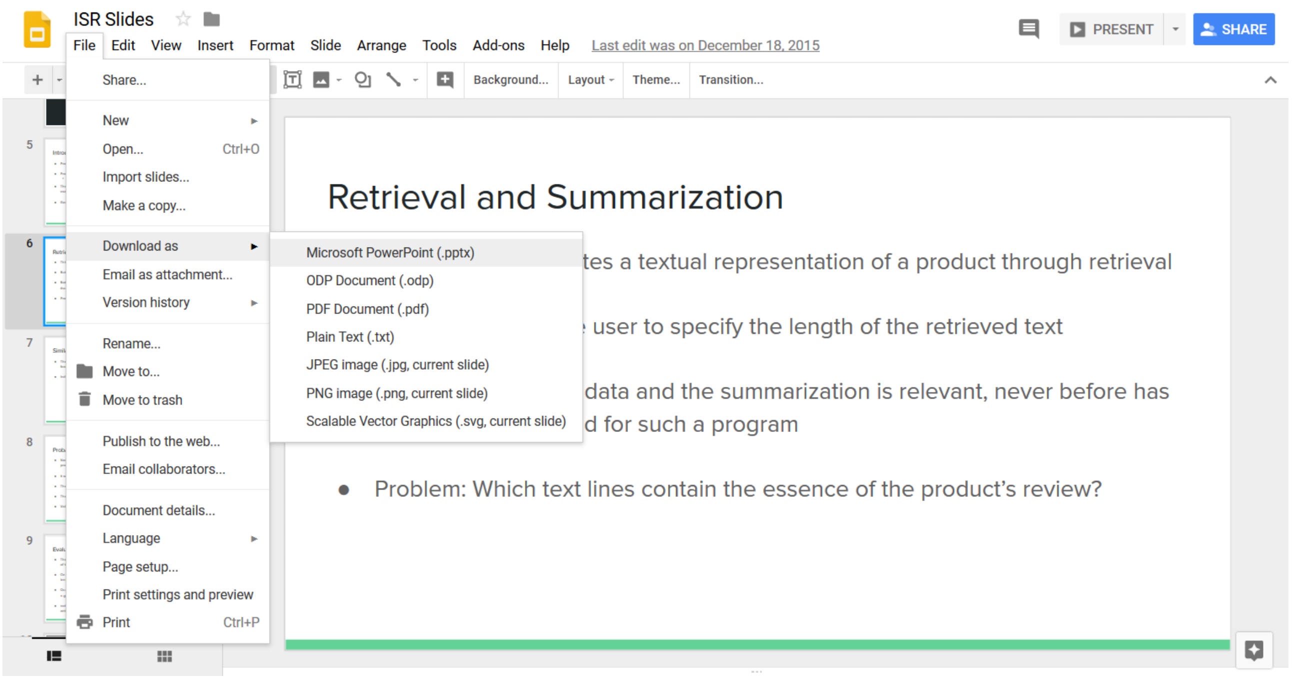
## 5.1 Google Slides

Google hosts a free online office suite as part of Google Drive. This includes Google Slides [Google 2006], an online presentation tool compatible with PowerPoint, which means users can import from and export to the PowerPoint file format freely. An example of the export menu is shown in Figure 5.1. The features of Google Slides mostly mirror those of PowerPoint.

Google Slides provides a set of templates and layouts and focuses on simple and fast to create slideshows with basic functionality. The largest advantage of working in Google Slides is that multiple slide creators can edit a presentation simultaneously. Every change is committed to Google Drive's cloud servers immediately and slides can even be edited as they are presented. An offline version does not exist.

## 5.2 Prezi

Prezi [Halácsy et al. 2018a] takes a camera with a bird's-eye view over an infinitely expandable canvas and moves it between fixed points. It was developed as a standalone app for iOS in 2011. An Android and a web browser version using Flash were released later on, and a HTML5-based hosted version was released in 2017 called "Prezi Next". Prezi provides public and reusable template slide decks on their website, an example of which is shown in Figure 5.2. With the acquisition of Infogram, a data visualisation platform, Prezi expanded its PowerPoint-like editor with several tools for charts and figures. Beyond that, it provides analysis tools for slide creators, showing how much time viewers spend on each slide, how many people viewed the presentation, and how many of them shared it. Unfortunately, the analysis tools are only available for Premium users and offline access is only provided to Plus users and higher. A list of the available account plans can be seen in Figure 5.3.



Retrieval and Summarization

tes a textual representation of a product through retrieval

user to specify the length of the retrieved text

data and the summarization is relevant, never before has
d for such a program

- Problem: Which text lines contain the essence of the product's review?

**Figure 5.1:** Google Slides supports exporting to various file formats, including PowerPoint and PDF. [Screenshot made by the author from Google Slides.]

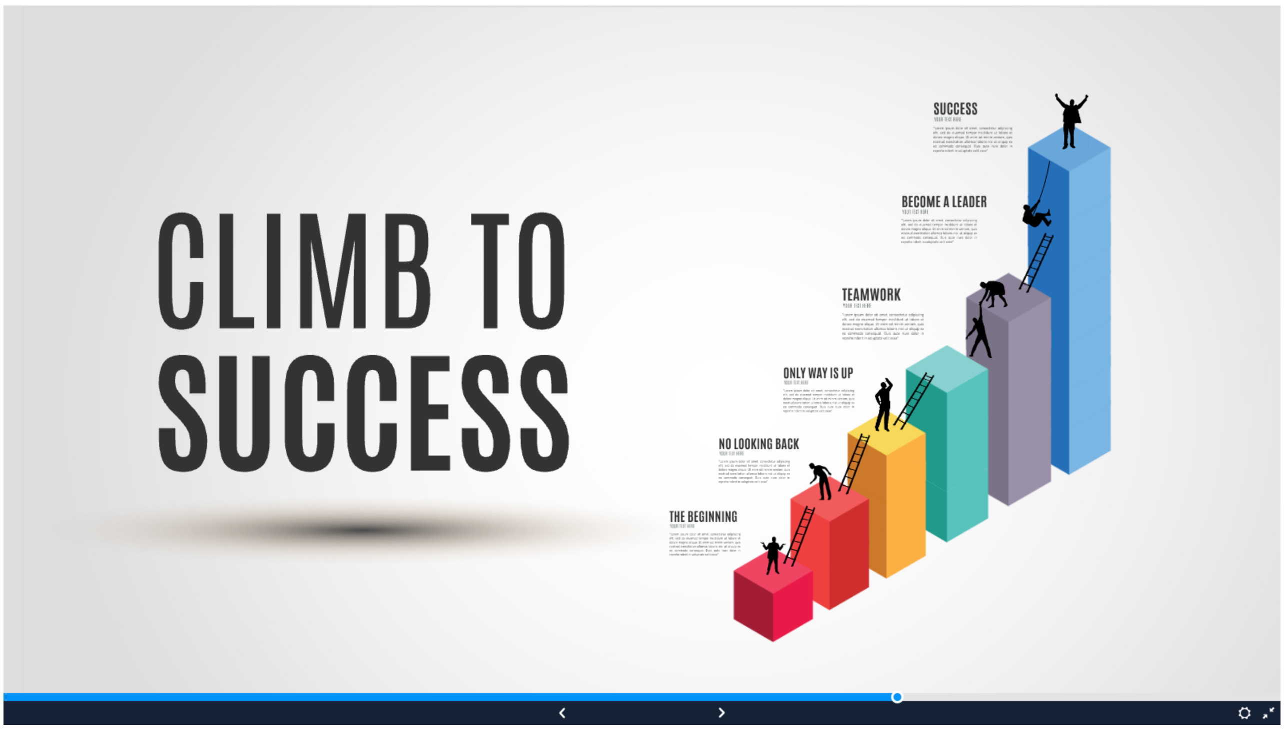
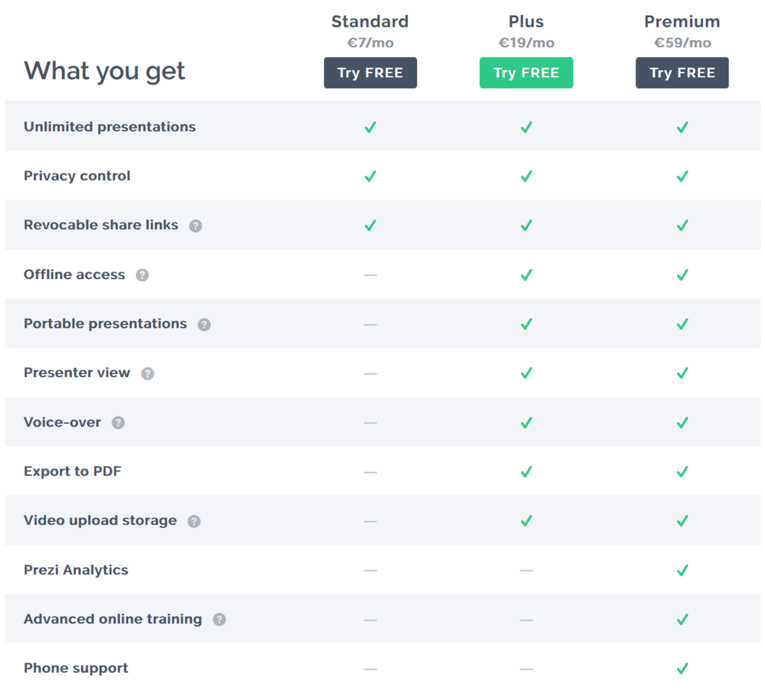


**Figure 5.2:** A slide showing the entire canvas of Prezi's "Climb to Success" template. [Screenshot made by the author from Prezi.com [Prezibase 2015].]



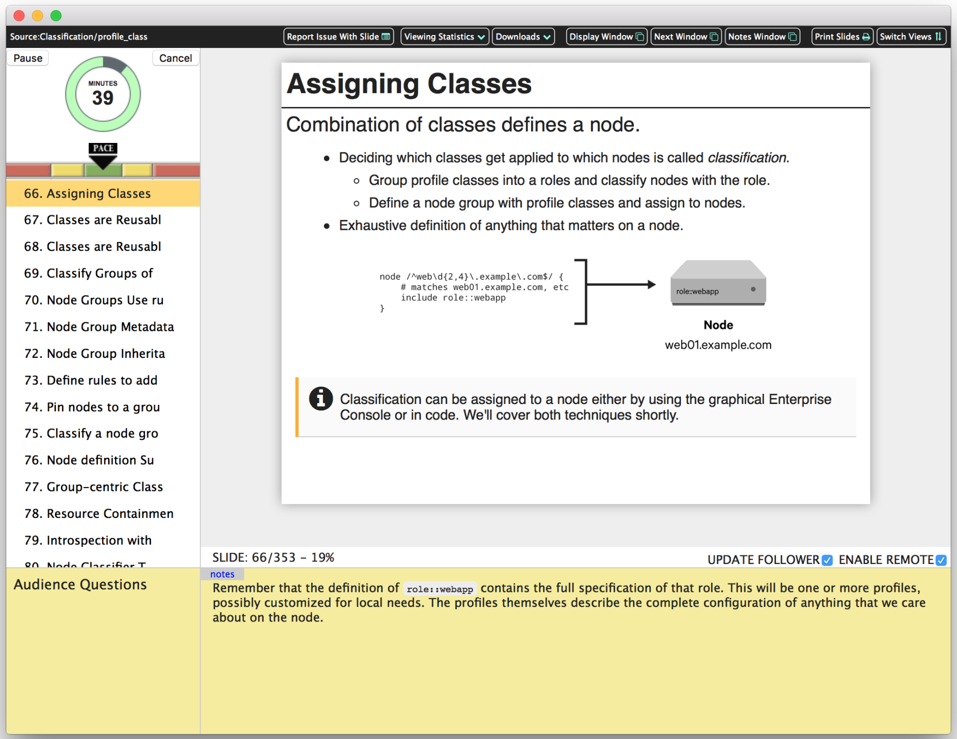
| What you get | Standard €7/mo Try FREE | Plus €19/mo Try FREE | Premium €59/mo Try FREE |
|---|---|---|---|
| Unlimited presentations | ✓ | ✓ | ✓ |
| Privacy control | ✓ | ✓ | ✓ |
| Revocable share links | ✓ | ✓ | ✓ |
| Offline access | — | ✓ | ✓ |
| Portable presentations | — | ✓ | ✓ |
| Presenter view | — | ✓ | ✓ |
| Voice-over | — | ✓ | ✓ |
| Export to PDF | — | ✓ | ✓ |
| Video upload storage | — | ✓ | ✓ |
| Prezi Analytics | — | — | ✓ |
| Advanced online training | — | — | ✓ |
| Phone support | — | — | ✓ |

**Figure 5.3:** A list of Prezi features included with each subscription plan. Features like offline access, portable presentations, presenter view, and exporting to PDF are not included in the Standard plan, and analysis tools are only included in the Premium plan. [Screenshot made by the author from Prezi.com [Halácsy et al. 2018b].]

## 5.3 Showoff

"Don't just present; interact with your audience!" is the core idea behind Showoff [Puppetlabs 2018]. It supports markdown, live code execution, speaker notes, and a presenter mode. Unlike traditional slide decks, Showoff also includes live quizzes, polls, and feedback questions in order to boost audience interaction. An example can be seen in Figure 5.4. Showoff is written in Ruby and can be easily installed like any other Ruby gem.

Showoff is intended for use in a face-to-face classroom setting. To use its interactive features, the speaker executes the command `showoff serve` and distributes the local IP address URL to members of the audience. For these live viewers, the slideshow is synchronised to the main presentation or can be navigated independently. It is also possible to create a slide deck for deployment to a web server, but without the interactive audience features.



66. Assigning Classes
67. Classes are Reusabl
68. Classes are Reusabl
69. Classify Groups of
70. Node Groups Use ru
71. Node Group Metadata
72. Node Group Inherita
73. Define rules to add
74. Pin nodes to a grou
75. Classify a node gro
76. Node definition Su
77. Group-centric Class
78. Resource Containmen
79. Introspection with
80. Node Classifier T

Audience Questions

## Assigning Classes

Combination of classes defines a node.

- Deciding which classes get applied to which nodes is called *classification*.
  - Group profile classes into a roles and classify nodes with the role.
  - Define a node group with profile classes and assign to nodes.
- Exhaustive definition of anything that matters on a node.

```
node /^web\d{2,4}\.example\.com$/ {
    # matches web01.example.com, etc
    include role::webapp
}
```

Classification can be assigned to a node either by using the graphical Enterprise Console or in code. We'll cover both techniques shortly.

SLIDE: 66/353 – 19%

UPDATE FOLLOWER ✓ ENABLE REMOTE ✓

notes

Remember that the definition of `role::webapp` contains the full specification of that role. This will be one or more profiles, possibly customized for local needs. The profiles themselves describe the complete configuration of anything that we care about on the node.

**Figure 5.4:** Showoff encourages interaction with the audience. The person hosting a Showoff presentation can see a list of audience questions at the bottom of the screen. [Screenshot taken from Showoff [Puppetlabs 2018] (MIT License).]

# Chapter 6

# Responsive Slide Decks

Ethan Marcotte [2014] describes the three main pillars of Responsive Web Design as flexible grids, fluid images, and media queries. As the name implies, using this approach makes it possible to respond to changes in the browser environment and adjust how a web site or application is displayed accordingly. This flexible and device-independent design is the key to developing long term web applications which can run on a multitude of display sizes and sniff for features rather than requiring them. Responsive slide decks can thus tackle the modern challenges described in Section 1.3.

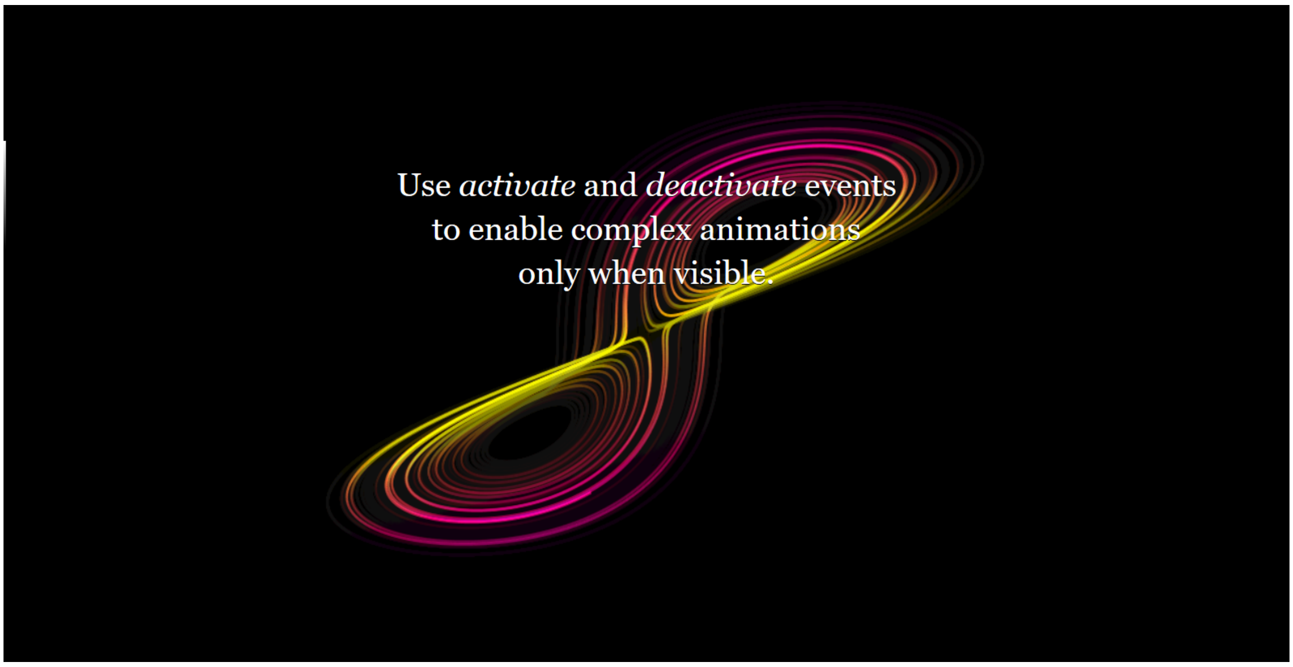
## 6.1 Stack

Stack [Bostock 2015a] takes a unique approach to slide decks by stacking slides from top to bottom, rather than relying on the usual left to right transitions. Slide navigation is handled primarily by scrolling, which is a basic feature supported by every browser on every device. Relying on scrolling allows Stack to support touch controls without explicitly coding for them. A small bar is displayed on the left side of the screen indicating the distance which needs to be scrolled for the next slide to appear. In addition to mouse wheel, touch pad, or touch screen, basic keyboard controls are also supported.

Stack resizes the content of slides dynamically with CSS media queries and JavaScript, maintaining a constant aspect ratio. The framework also provides two events called “activate” and “deactivate” to execute certain content only when a specific slide is displayed. This can be used to prevent expensive scripts or animations from executing while they are not visible. An example can be seen in Figure 6.1 and Listing 6.1.

## 6.2 Scrolldeck.js

Scrolldeck.js [Polacek 2015b] builds on the idea behind Stack (see Section 6.1) and enhances it using jQuery and Scrollorama. This includes support for animations, incremental lists, parallax scrolling, and improvements in the responsive scaling of content. The slides are no longer fixed to a constant aspect ratio and scale more freely. An optional sticky header is also supported, which displays the title and some user-defined navigation shortcuts. An example of how to setup Scrolldeck.js can be seen in Listing 6.2. Parallax scrolling is shown in Figure 6.2.

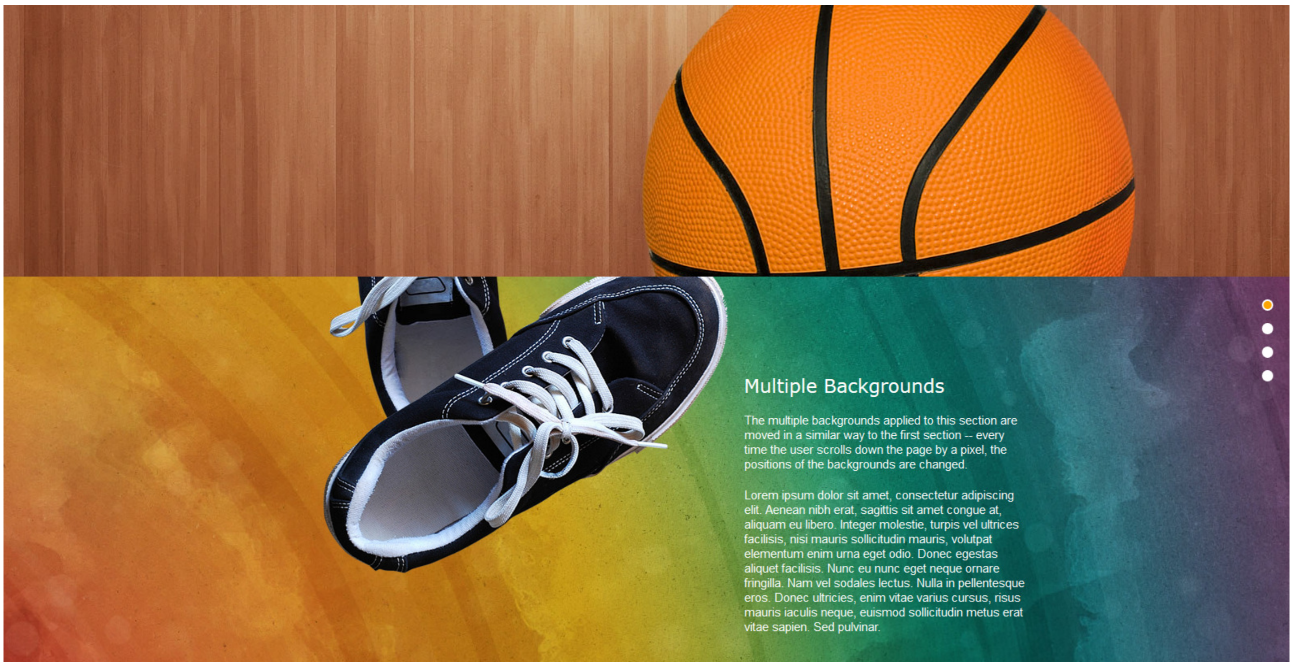


Use *activate* and *deactivate* events
to enable complex animations
only when visible.

**Figure 6.1:** In order to conserve power, Stack provides the events *activate* and *deactivate*. With these events, expensive calculations can be disabled unless a specific slide is displayed. [Screenshot made by the author from Stack [Bostock 2015b].]

```
<section>
  <h1>Slide</h1>

  <ul>
    <li>First bullet point</li>
    <li>Second bullet point</li>
    <li>Third bullet point</li>
  </ul>
</section>

var mystack = stack()
    .on("activate", activate)
    .on("deactivate", deactivate);

function activate(d, i) {
  if (i === 3) start();
}

function deactivate(d, i) {
  if (i === 3) stop();
}
```

**Listing 6.1:** An example slide in Stack. On a slide transition, the events “activate” and “deactivate” are called. In this example, special functions are called for the slide with index 3.



Multiple Backgrounds

The multiple backgrounds applied to this section are moved in a similar way to the first section -- every time the user scrolls down the page by a pixel, the positions of the backgrounds are changed.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean nibh erat, sagittis sit amet congue at, aliquam eu libero. Integer molestie, turpis vel ultrices facilisis, nisi mauris sollicitudin mauris, volutpat elementum enim urna eget odio. Donec egestas aliquet facilisis. Nunc eu nunc eget neque ornare fringilla. Nam vel sodales lectus. Nulla in pellentesque eros. Donec ultricies, enim vitae varius cursus, risus mauris iaculis neque, euismod sollicitudin metus erat vitae sapien. Sed pulvinar.

**Figure 6.2:** Scrolldeck.js can blend together two slides using parallax scrolling. [Screenshot made by the author from Scrolldeck.js [Polacek 2015a].]

```
<!-- Load the required dependencies and scrolldeck.js -->
<script src="js/jquery-1.8.2.min.js"></script>
<script src="js/jquery.scrollTo-1.4.3.1.min.js"></script>
<script src="js/jquery.scrollorama.js"></script>
<script src="js/jquery.easing.1.3.js"></script>
<script src="js/jquery.scrolldeck.js"></script>

<div class="slide">
  <h1>Slide</h1>

  <ul>
    <li>First bullet point</li>
    <li>Second bullet point</li>
    <li>Third bullet point</li>
  </ul>
</div>

<!-- Initialize scrolldeck.js -->
<script>
  $(document).ready(function() {
    var deck = new $.scrolldeck({
      buttons: '.nav-button',
      slides: '.slide',
      duration: 600,
      easing: 'easeInOutExpo',
    offset: 0
    });
  });
</script>
```

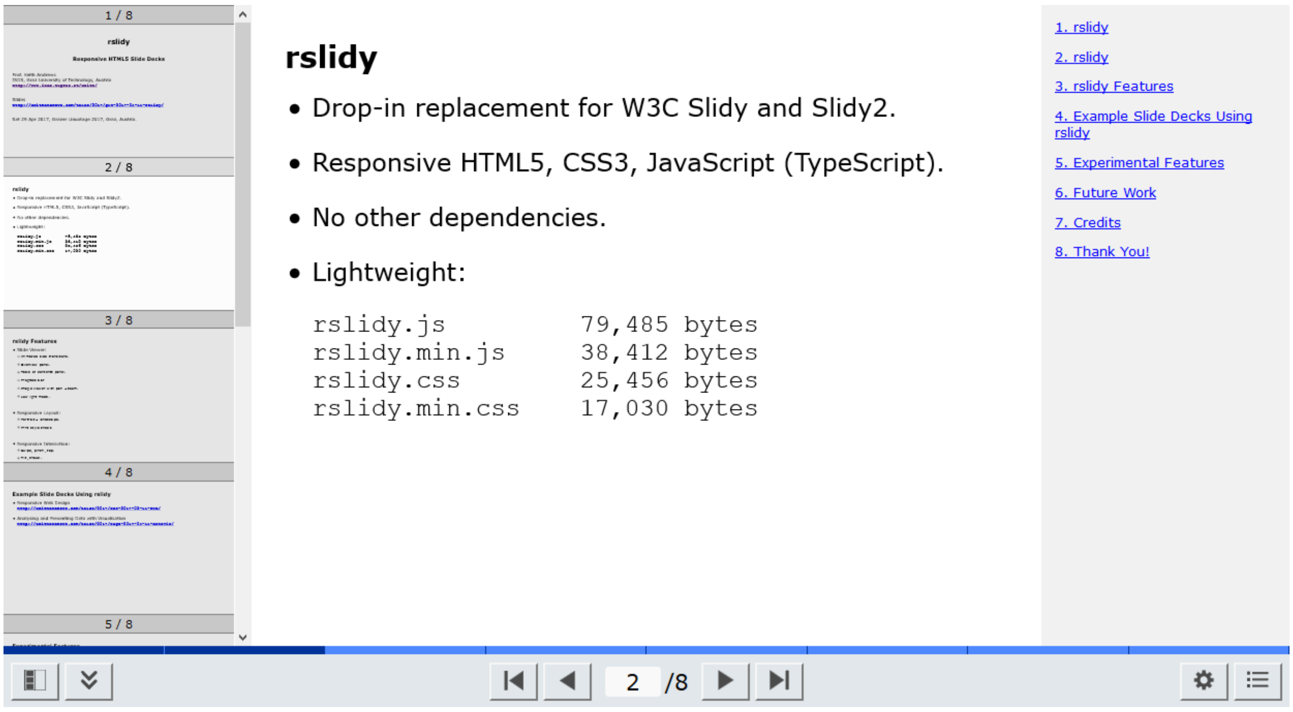
**Listing 6.2:** Scrolldeck.js requires the scripts `jquery`, `scrollTo`, `scrollorama` and `easing` to be linked before its own script file. Initialisation and configuration is then handled in the `$(document).ready` function.

```
<div class="slide">
  <h2>rslidy</h2>

  <ul>
    <li>Drop-in replacement for W3C Slidy and Slidy2.</li>

    <li>Responsive HTML5, CSS3, JavaScript (TypeScript).</li>

    <li>No other dependencies.</li>

    <li>Lightweight:
      <pre>
        rslidy.js         79,485 bytes
        rslidy.min.js     38,412 bytes
        rslidy.css        25,456 bytes
        rslidy.min.css    17,030 bytes
      </pre>
    </li>
  </ul>
</div>
```

**Listing 6.3:** Slides in Rslidy are defined inside either `<div class="slide">` or `<section>` elements.

## 6.3 Rslidy

Rslidy [Schofnegger 2015] is a lightweight and responsive slide deck currently in development at Graz University of Technology. It is backwards compatible with Slidy and Slidy2 (see Section 2.1). Rslidy is feature-rich, responsive, resilient, and supports a variety of input methods. The code from Listing 6.3 turns into the slide shown in Figure 6.3, showcasing Rslidy's interface. Some of its features include: slide transitions, a navigable progress bar, an image viewer with pan and zoom controls, low light mode, print style sheet, navigation with touch, tilt, shake and swipe controls, a table of contents, and an overview with slide thumbnails.



## rslidy

- Drop-in replacement for W3C Slidy and Slidy2.
- Responsive HTML5, CSS3, JavaScript (TypeScript).
- No other dependencies.
- Lightweight:

```
rslidy.js         79,485 bytes
rslidy.min.js     38,412 bytes
rslidy.css        25,456 bytes
rslidy.min.css    17,030 bytes
```

1. rslidy
2. rslidy
3. rslidy Features
4. Example Slide Decks Using rslidy
5. Experimental Features
6. Future Work
7. Credits
8. Thank You!

**Figure 6.3:** Rslidy has a status bar, an overview showing slide thumbnails, and a table of contents, all of which can be hidden. [Screenshot made by the author from Rslidy.]

# Chapter 7

# Concluding Remarks

It is important to keep in mind that many of the tools reviewed in this survey fit into more than one of the main categories and the code of earlier approaches is often supported by more modern approaches. The trinity of web development (HTML, CSS and JavaScript) makes it possible to ditch proprietary slide decks and allows for more open, creative, and varied approaches. A major advantage of web-based slide decks is the ability to execute code directly at runtime. However, where web-based slide tools are developed by smaller teams, bugs and issues may be left unfixed for long periods of time.

In the future, there will be an even greater focus on collaboration and sharing, as can be seen by the most popular tools in use right now. Web browsers and handheld devices are constantly evolving and changing shape, pushing developers towards responsive design approaches. As Ethan Marcotte [2014] writes in his book on responsive web design:

> "Web design is about asking the right questions. And really, that's what responsive web design is: a possible solution, a way to more fully design for the web's inherent flexibility. [· · · ] If we're willing to research the needs of our users, and apply those ingredients carefully, then responsive web design is a powerful approach indeed."

Asking the right questions is important for presentation software as well and many possible solutions have been showcased in this survey. Although they are not perfect by any means, slide decks built for the web are a viable alternative for creating state-of-the-art slideshows.

# Appendix A

# Code Examples

In this part of the survey, a model slide is recreated using the default settings of each slide deck, showing their similarities as well as their differences. Since many slide decks can run the same code, a Listing may only be needed once to produce several example figures. The model slide consists of a heading, a paragraph, and a bulleted list.

The code from Listing A.1, using HTML elements contained in `<div class="slide">...</div>`, is valid for:

- Slidy & Slidy2, covered in Section 2.1. The model slide is shown in Figure A.1.
- S5, covered in Section 2.2. The model slide is shown in Figure A.2.
- S6, covered in Section 2.3. The model slide is shown in Figure A.7.
- Fathom.js, covered in Section 4.2. The model slide is shown in Figure A.3.
- Inspire.js, covered in Section 4.6. The model slide is shown in Figure A.4.
- Scrolldeck.js, covered in Section 6.2. The model slide is shown in Figure A.5.
- Rslidy, covered in Section 6.3. The model slide is shown in Figure A.6.

The code from Listing A.2, using HTML5 elements contained in `<section class="slide">...</section>`, is valid for:

- S6, covered in Section 2.3. The model slide is shown in Figure A.7.
- Deck.js, covered in Section 4.3. The model slide is shown in Figure A.8.
- DZSlides, covered in Section 4.4. The model slide is shown in Figure A.9.
- Bespoke.js, covered in Section 4.5. The model slide is shown in Figure A.10.
- Inspire.js, covered in Section 4.6. The model slide is shown in Figure A.4.
- Reveal.js, covered in Section 4.7. The model slide is shown in Figure A.11.
- Shower, covered in Section 4.8. The model slide is shown in Figure A.12.
- Stack, covered in Section 6.1. The model slide is shown in Figure A.13.
- Rslidy, covered in Section 6.3. The model slide is shown in Figure A.6.

```html
<div class="slide">
  <h1>Slide Types</h1>

  <p>The main types of slides are:</p>

  <ul>
    <li>Slides using valid HTML(5).</li>
    <li>Slides using Markdown.</li>
    <li>Slides using something else entirely.</li>
  </ul>
</div>
```

**Listing A.1:** The model slide in HTML, using `<div class="slide">...</div>` to hold the slide content.

```html
<section class="slide">
  <h1>Slide Types</h1>

  <p>The main types of slides are:</p>

  <ul>
    <li>Slides using valid HTML(5).</li>
    <li>Slides using Markdown.</li>
    <li>Slides using something else entirely.</li>
  </ul>
</section>
```

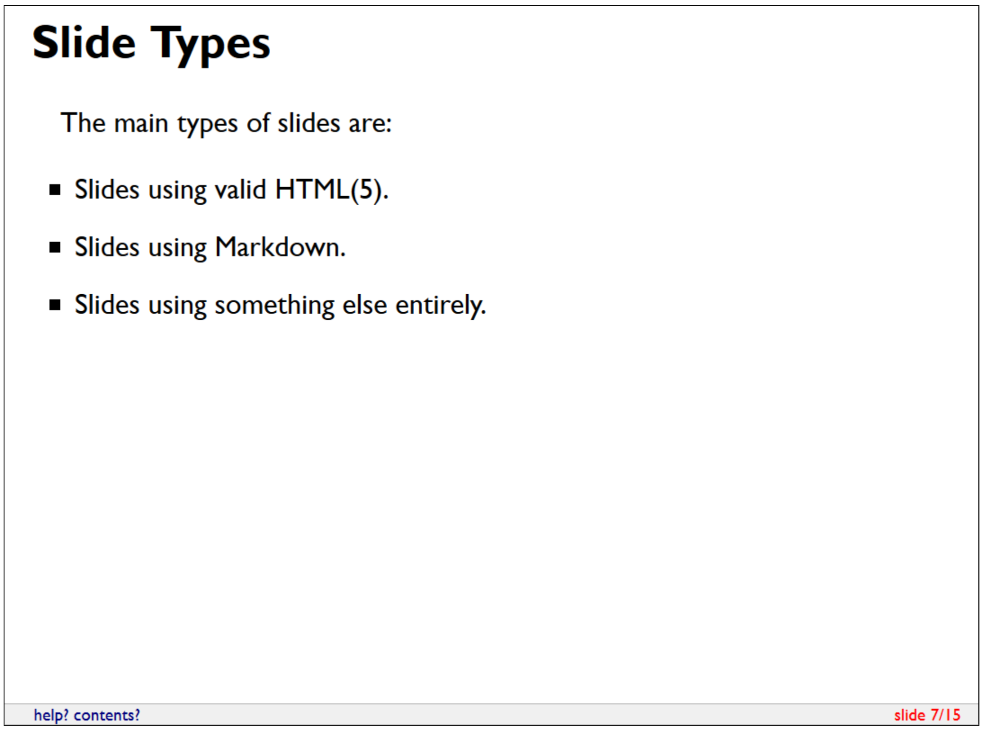
**Listing A.2:** The model slide in HTML5, using `<section class="slide">...</section>` to hold the slide content.

```markdown
# Slide Types

The main types of slides are:

- Slides using valid HTML(5).
- Slides using Markdown.
- Slides using something else entirely.
```

**Listing A.3:** The model slide in Markdown, using # for headings and - or * for lists.

The code from Listing A.3, using Markdown, is valid for:

- S9, covered in Section 3.1. S9 can generate the model slide for S5, S6, Slidy, and others.
- Slidedown, covered in Section 3.2. The model slide is shown in Figure A.14.
- Slidifier, covered in Section 3.3. The model slide is shown in Figure A.15.
- Remark, covered in Section 3.4. The model slide is shown in Figure A.16.
- Showoff, covered in Section 5.3. The model slide is shown in Figure A.17.

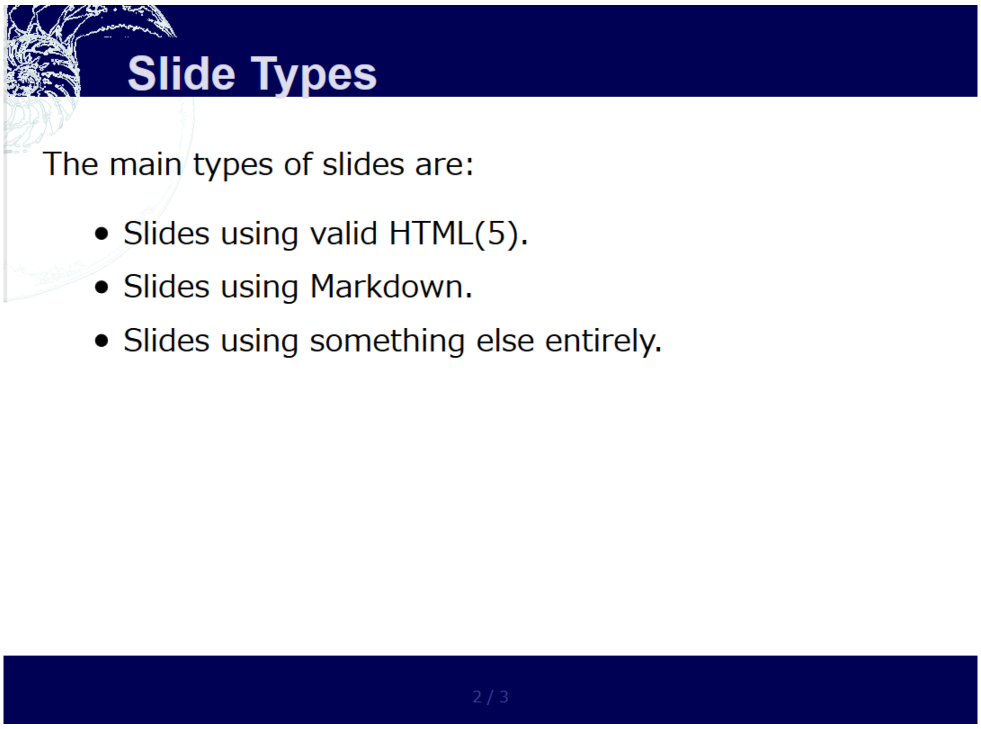


## Slide Types

The main types of slides are:

- Slides using valid HTML(5).
- Slides using Markdown.
- Slides using something else entirely.

**Figure A.1:** The code from Listing A.1 in Slidy2. [Screenshot made by the author.]



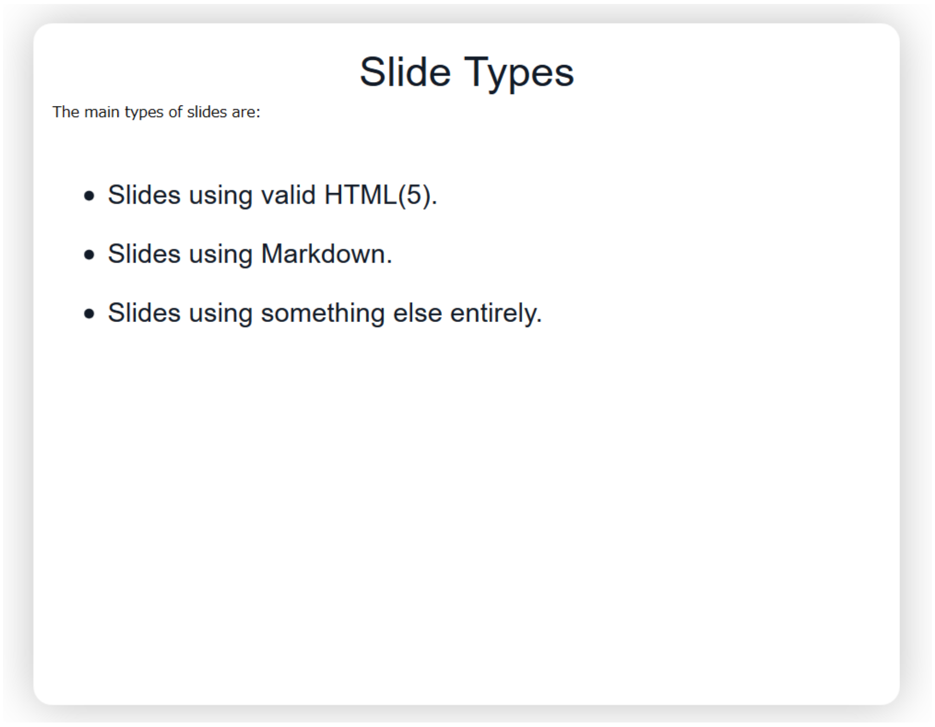
Slide Types

The main types of slides are:

- Slides using valid HTML(5).
- Slides using Markdown.
- Slides using something else entirely.

2 / 3

**Figure A.2:** The code from Listing A.1 in S5. [Screenshot made by the author.]

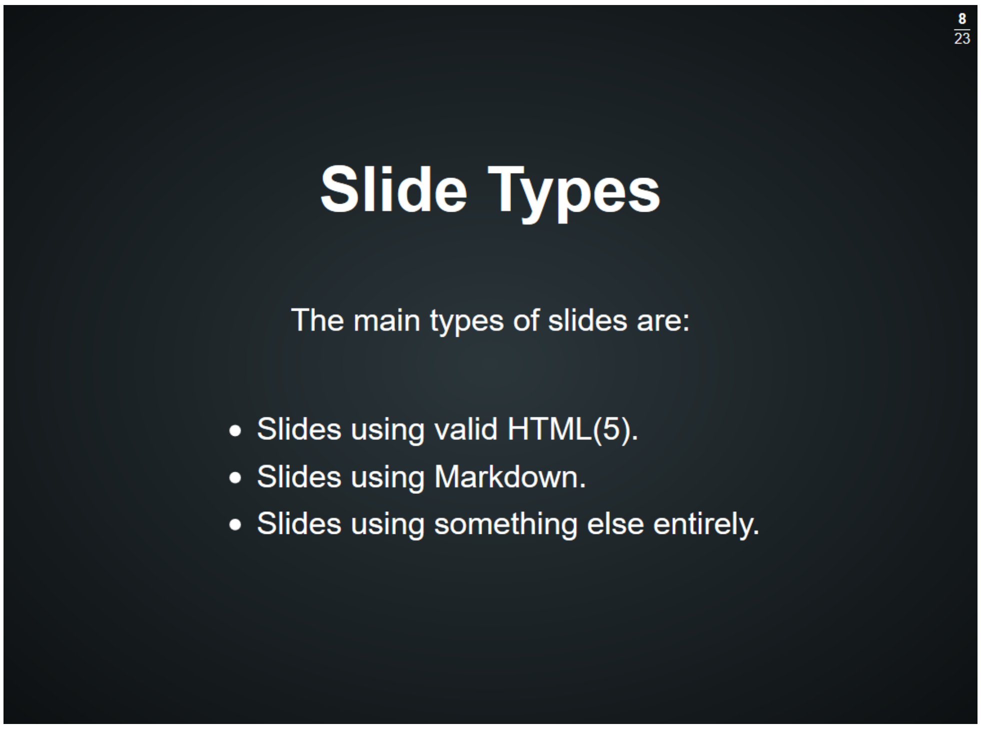


Slide Types

The main types of slides are:

- Slides using valid HTML(5).
- Slides using Markdown.
- Slides using something else entirely.

**Figure A.3:** The code from Listing A.1 in Fathom.js. [Screenshot made by the author.]

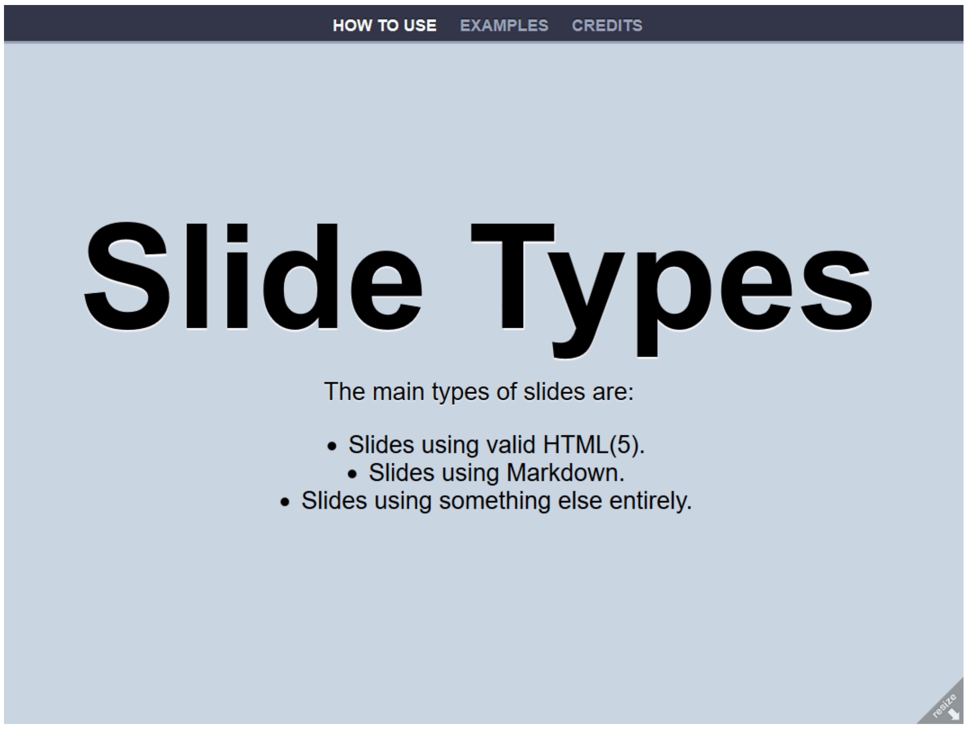


## Slide Types

The main types of slides are:

- Slides using valid HTML(5).
- Slides using Markdown.
- Slides using something else entirely.

**Figure A.4:** The code from Listing A.1 or Listing A.2 in Inspire.js. [Screenshot made by the author.]

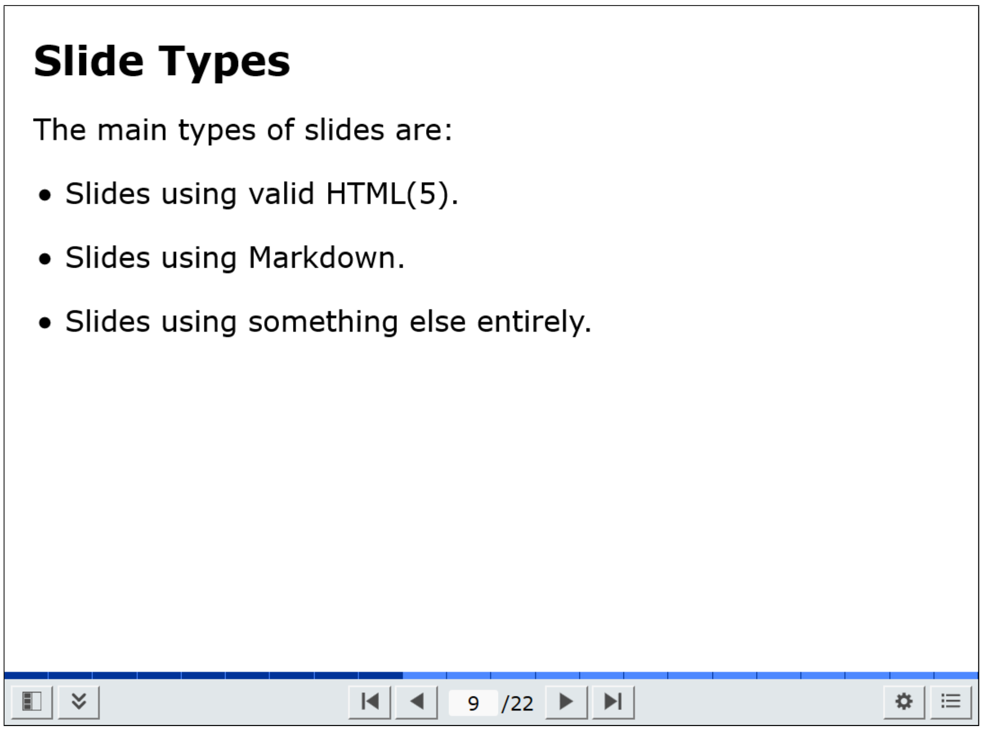


## Slide Types

The main types of slides are:

- Slides using valid HTML(5).
- Slides using Markdown.
- Slides using something else entirely.

**Figure A.5:** The code from Listing A.1 in Scrolldeck.js. [Screenshot made by the author.]

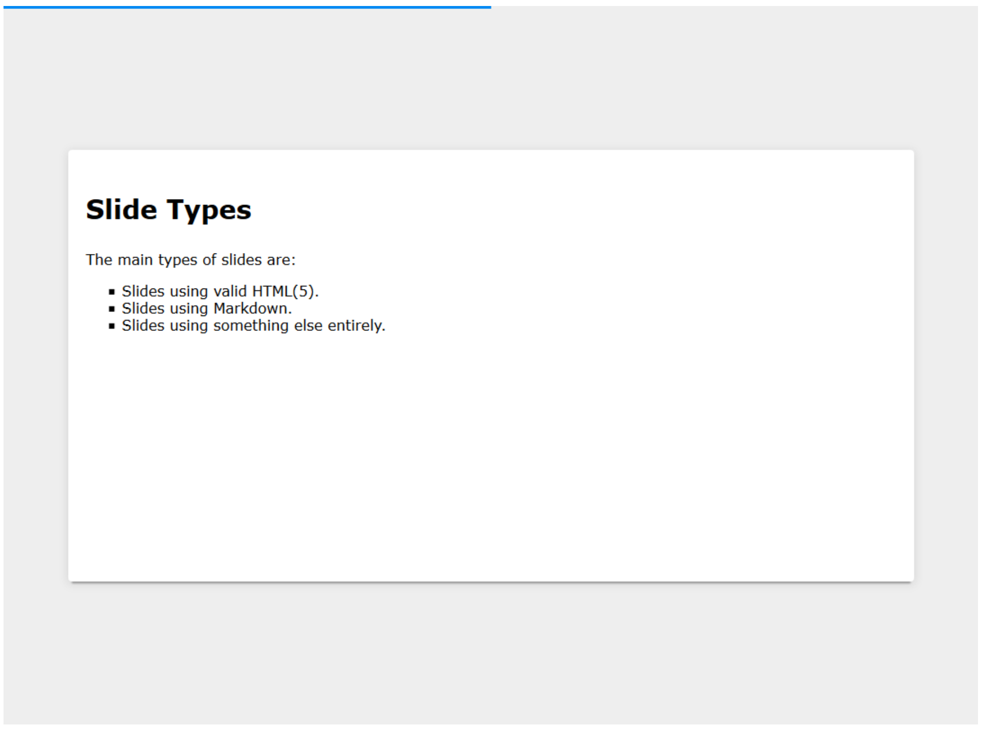


Slide Types

The main types of slides are:

- Slides using valid HTML(5).
- Slides using Markdown.
- Slides using something else entirely.

**Figure A.6:** The code from Listing A.1 or Listing A.2 in Rslidy. [Screenshot made by the author.]

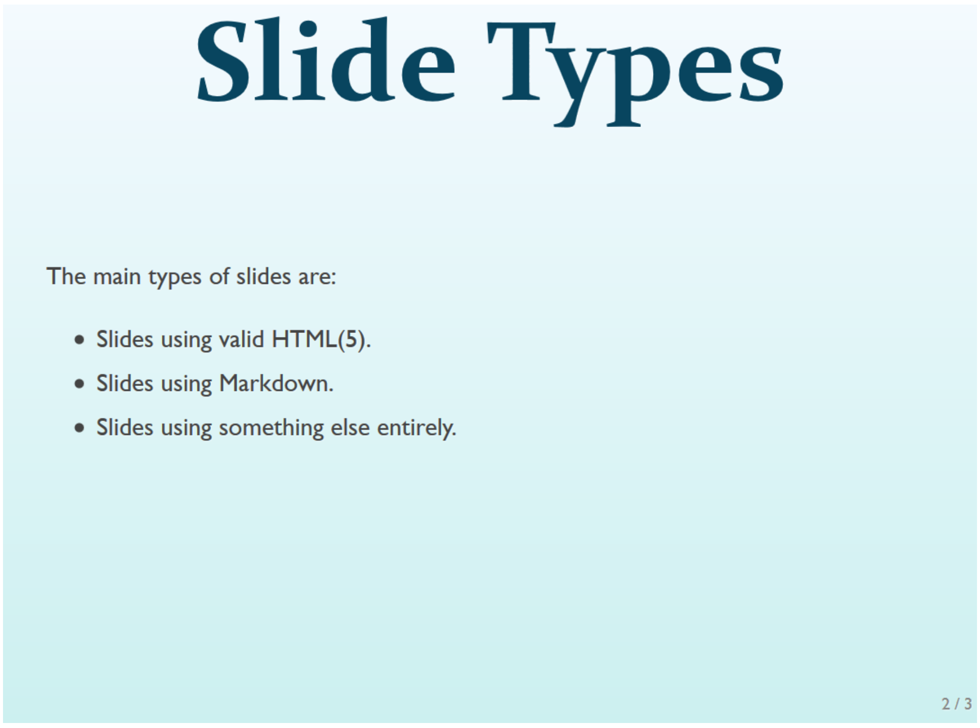


Slide Types

The main types of slides are:

- Slides using valid HTML(5).
- Slides using Markdown.
- Slides using something else entirely.

**Figure A.7:** The code from Listing A.1 or Listing A.2 in S6. [Screenshot made by the author.]



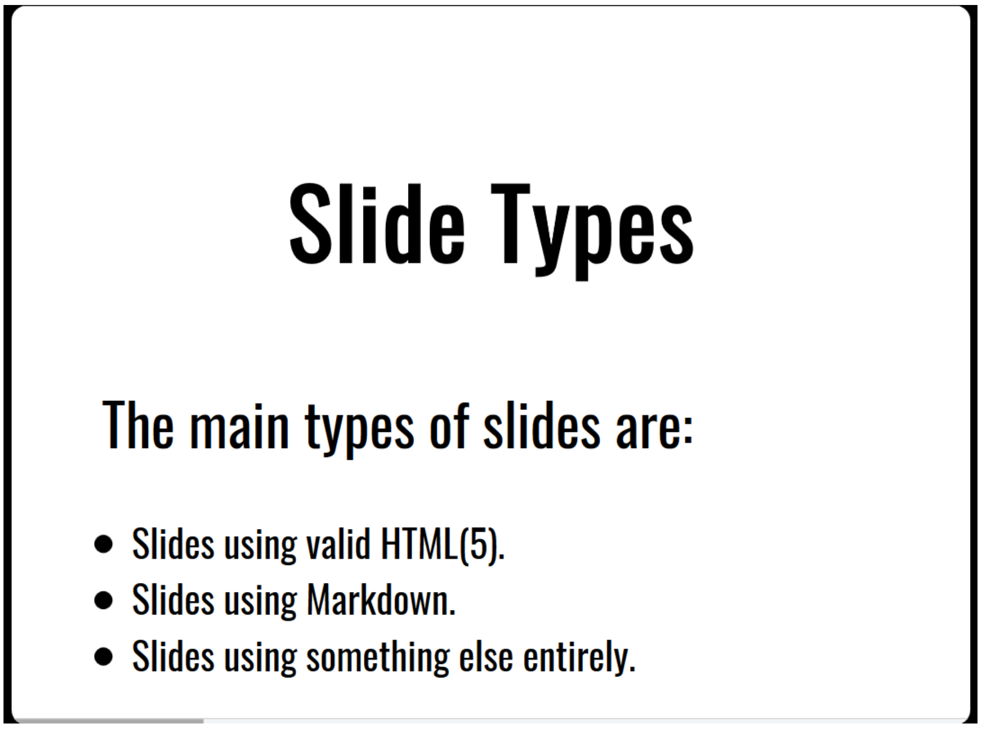
## Slide Types

The main types of slides are:

- Slides using valid HTML(5).
- Slides using Markdown.
- Slides using something else entirely.

2 / 3

**Figure A.8:** The code from Listing A.2 in Deck.js. [Screenshot made by the author.]

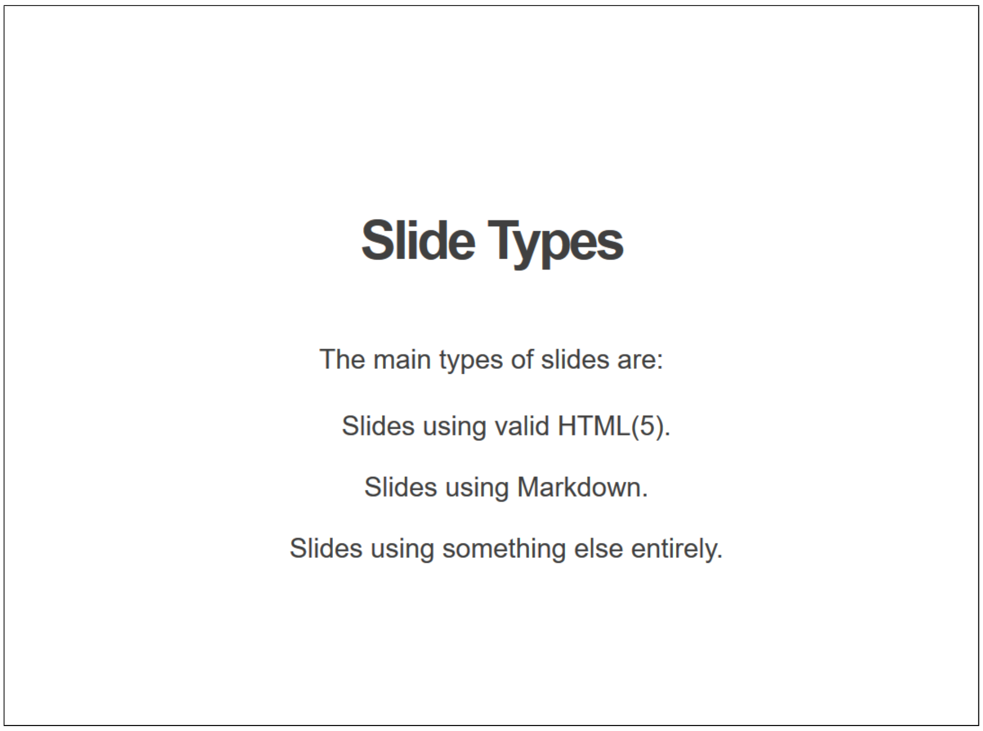


## Slide Types

The main types of slides are:

- Slides using valid HTML(5).
- Slides using Markdown.
- Slides using something else entirely.

**Figure A.9:** The code from Listing A.2 in DZSlides. [Screenshot made by the author.]



Slide Types

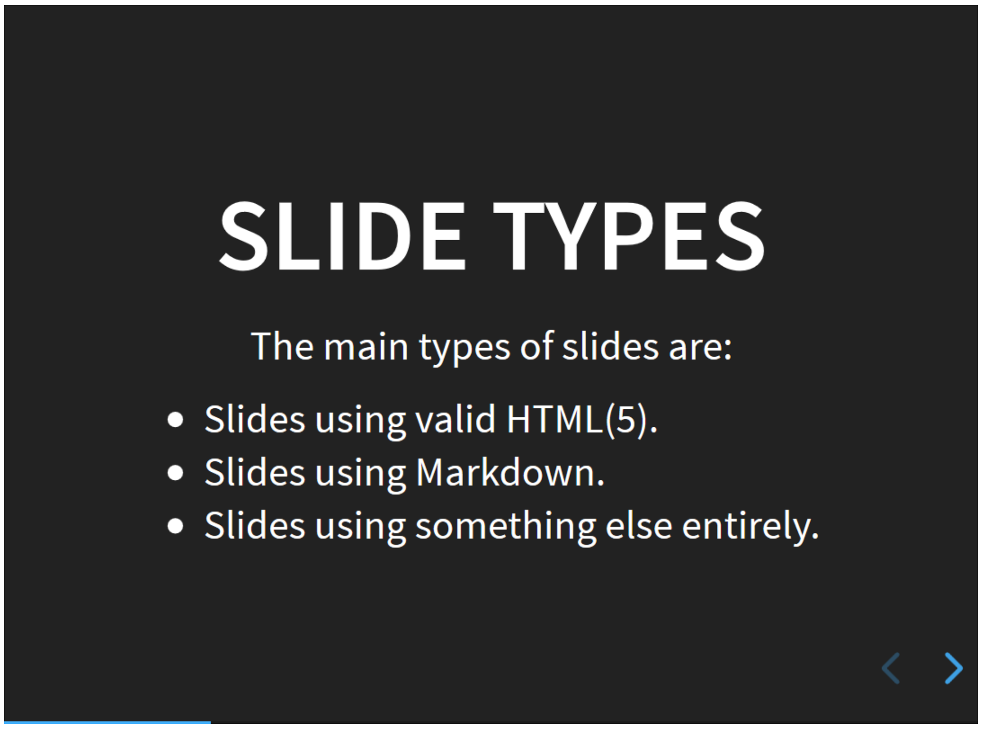
The main types of slides are:

Slides using valid HTML(5).

Slides using Markdown.

Slides using something else entirely.

**Figure A.10:** The code from Listing A.2 in Bespoke.js. [Screenshot made by the author.]

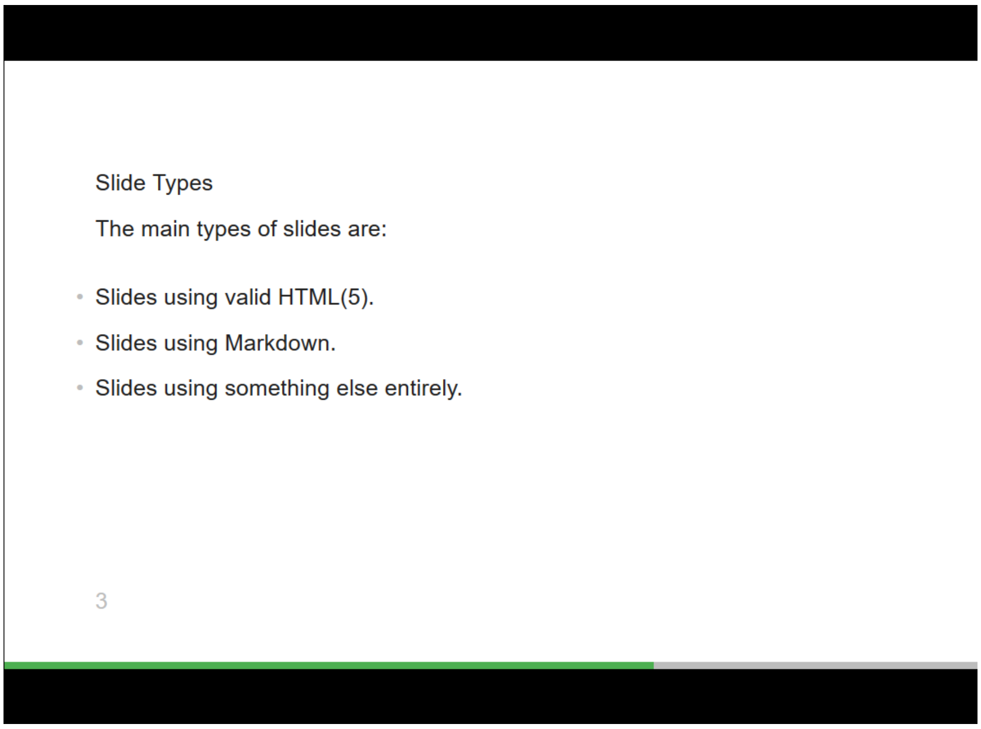


SLIDE TYPES

The main types of slides are:

- Slides using valid HTML(5).
- Slides using Markdown.
- Slides using something else entirely.

**Figure A.11:** The code from Listing A.2 in Reveal.js. [Screenshot made by the author.]



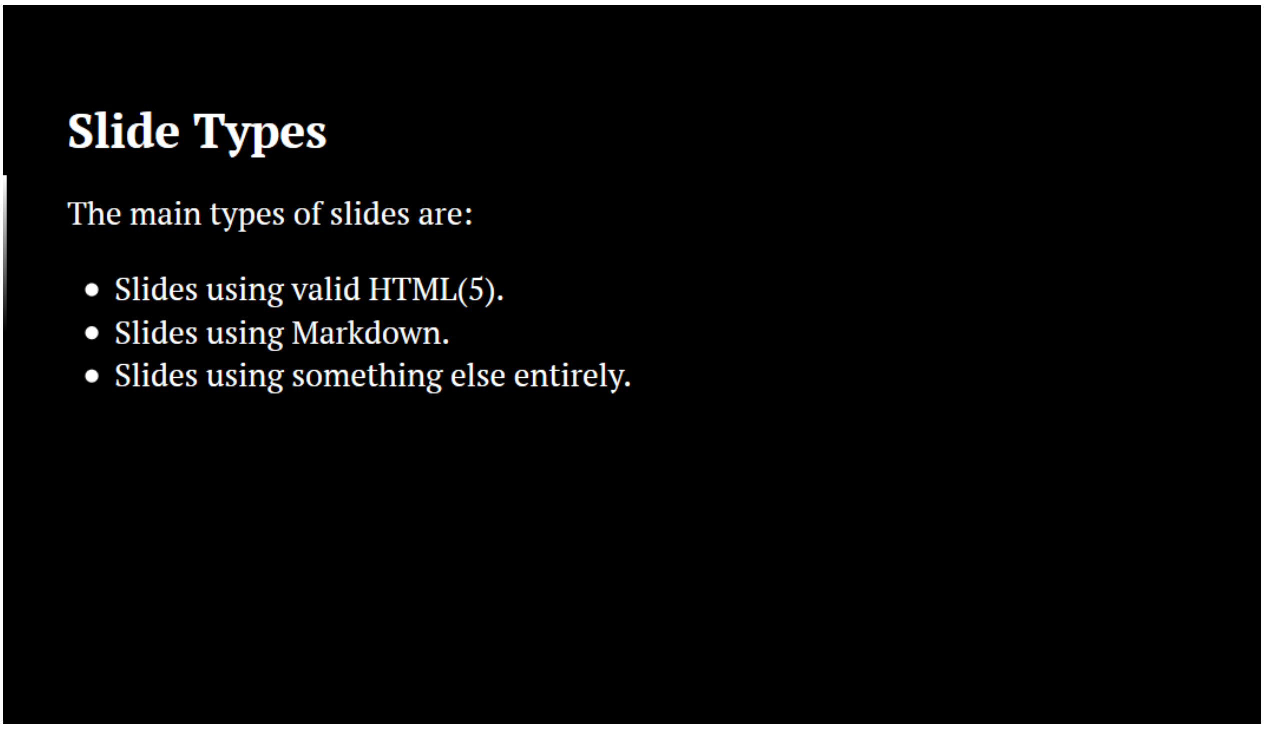
Slide Types

The main types of slides are:

- Slides using valid HTML(5).
- Slides using Markdown.
- Slides using something else entirely.

3

**Figure A.12:** The code from Listing A.2 in Shower. [Screenshot made by the author.]

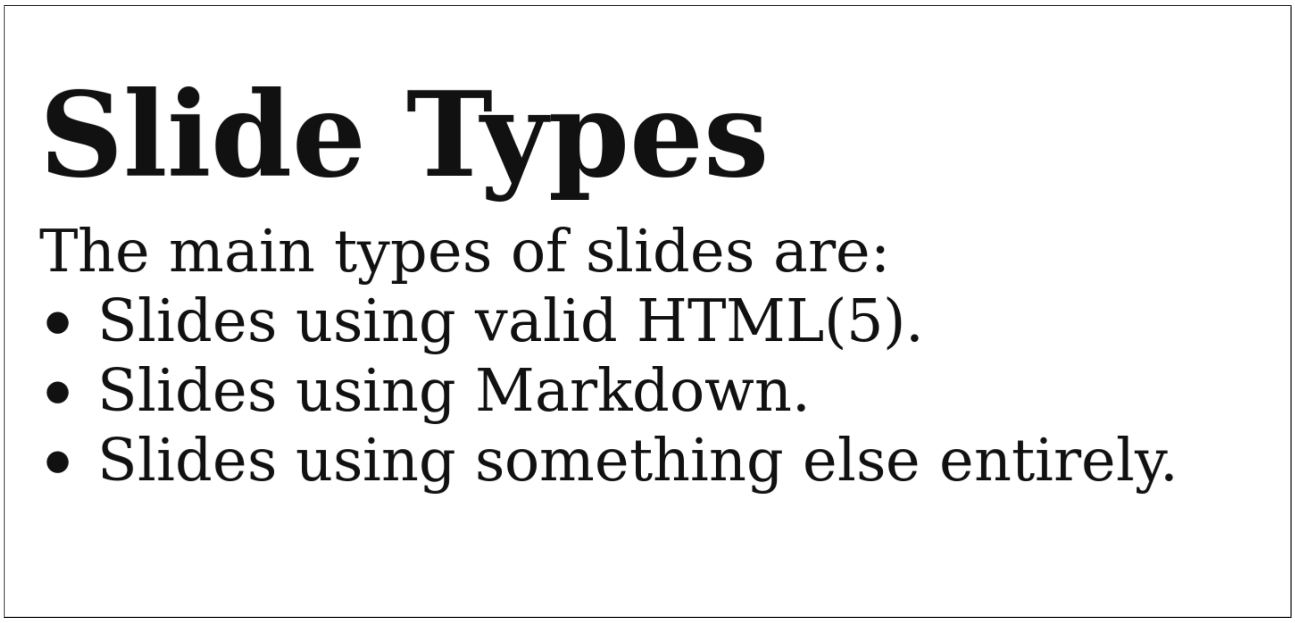


Slide Types

The main types of slides are:

- Slides using valid HTML(5).
- Slides using Markdown.
- Slides using something else entirely.

**Figure A.13:** The code from Listing A.2 in Stack. [Screenshot made by the author.]

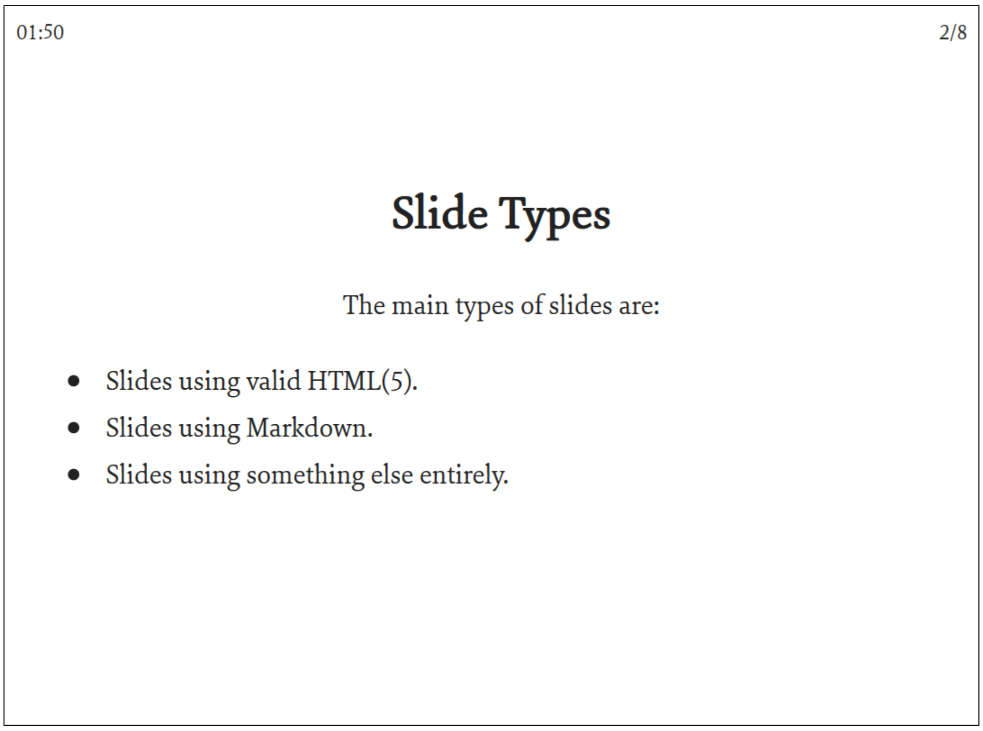


Slide Types

The main types of slides are:

- Slides using valid HTML(5).
- Slides using Markdown.
- Slides using something else entirely.

**Figure A.14:** The code from Listing A.3 in Slidedown. [Screenshot made by the author.]



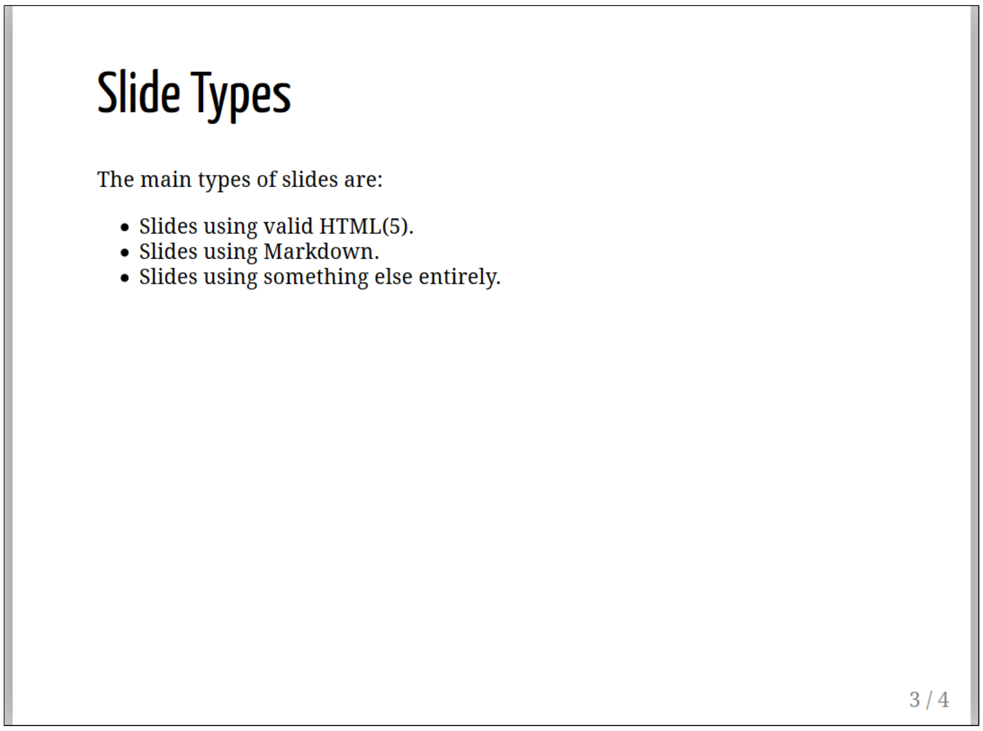
01:50 2/8

Slide Types

The main types of slides are:

- Slides using valid HTML(5).
- Slides using Markdown.
- Slides using something else entirely.

**Figure A.15:** The code from Listing A.3 in Slidifier. [Screenshot made by the author.]



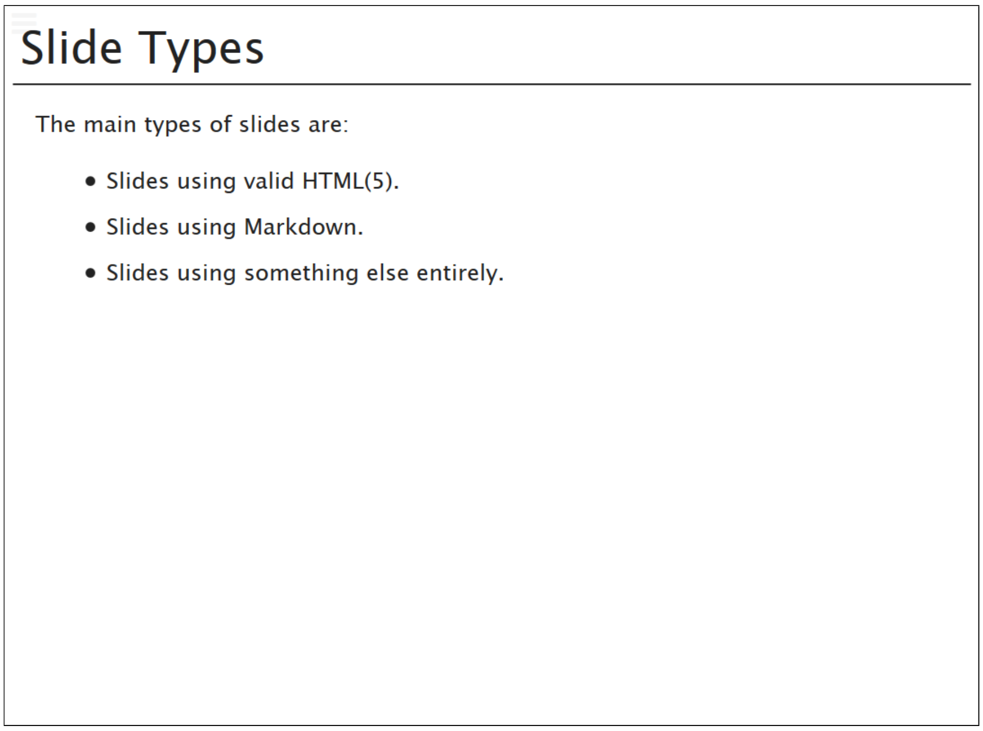
## Slide Types

The main types of slides are:

- Slides using valid HTML(5).
- Slides using Markdown.
- Slides using something else entirely.

3 / 4

**Figure A.16:** The code from Listing A.3 in Remark. [Screenshot made by the author.]

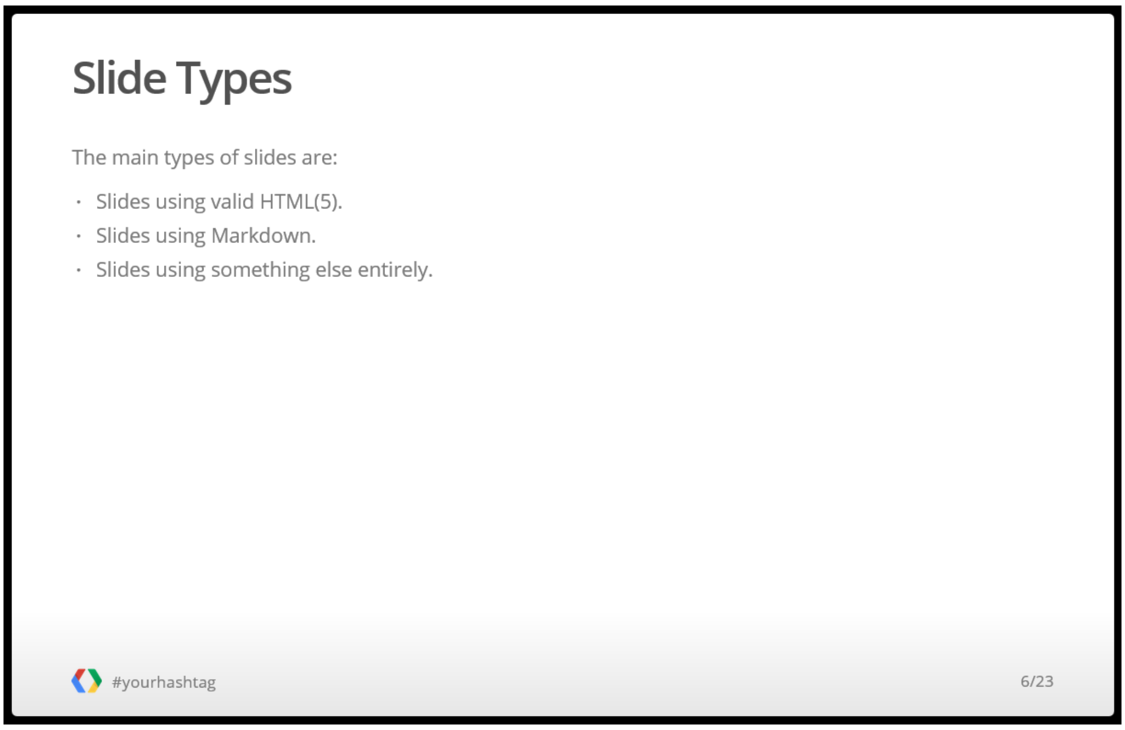


## Slide Types

The main types of slides are:

- Slides using valid HTML(5).
- Slides using Markdown.
- Slides using something else entirely.

**Figure A.17:** The code from Listing A.3 in Showoff. [Screenshot made by the author.]

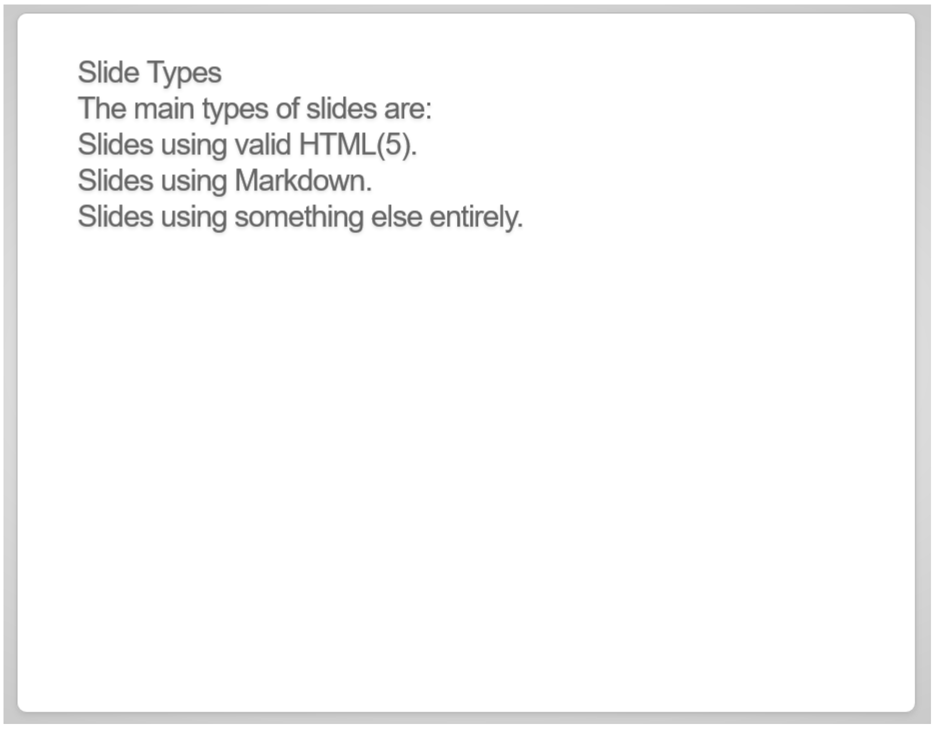


## Slide Types

The main types of slides are:

- Slides using valid HTML(5).
- Slides using Markdown.
- Slides using something else entirely.

**Figure A.18:** The model slide in Google I/O Slides. Google I/O Slides, covered in Section 4.1, uses `<slide>` instead of `<div>` or `<section>` to define slides, which is not a valid HTML element. [Screenshot made by the author.]



Slide Types
The main types of slides are:
Slides using valid HTML(5).
Slides using Markdown.
Slides using something else entirely.

**Figure A.19:** The model slide in Impress.js. Impress.js, covered in Section 4.9, uses `<div class="step">` instead of using `<div class="slide">`, as is shown in Listing 4.7. [Screenshot made by the author.]

# Bibliography

Alley, Michael [2013]. *The Craft of Scientific Presentations*. 2nd Edition. Springer, 30 Apr 2013. 286 pages. ISBN 1441982787 (cited on page 3).

Apple [2003]. *Apple Unveils Keynote*. Press Release. 07 Jan 2003. `https://apple.com/newsroom/2003/01/07Apple-Unveils-Keynote/` (cited on page 1).

Apple [2018]. *Keynote*. 18 Oct 2018. `https://apple.com/lae/keynote/` (cited on page 1).

Bang, Ole Petter [2018a]. *Remark Example*. 07 May 2018. `https://remarkjs.com` (cited on page 18).

Bang, Ole Petter [2018b]. *Remark: A Simple, In-Browser, Markdown-Driven Slideshow Tool*. GitHub. 07 May 2018. `https://github.com/gnab/remark` (cited on page 17).

Barrett, Brian [2017]. *Adobe Finally Kills Flash Dead*. Wired. 25 Jul 2017. `https://wired.com/story/adobe-finally-kills-flash-dead` (cited on page 10).

Bauer, Gerald [2011]. *Slide Show (S9) Guide*. 09 Jul 2011. `https://slideshow-s9.github.io/` (cited on page 13).

Bauer, Gerald [2013]. *S6 – Slide Show Templates Using HTML5, CSS3 'n' JavaScript*. 01 Jul 2013. `https://slidekit.github.io/` (cited on page 9).

Bostock, Mike [2015a]. *Stack*. GitHub. 19 Mar 2015. `https://github.com/mbostock/stack` (cited on page 35).

Bostock, Mike [2015b]. *Stack Example*. GitHub. 19 Mar 2015. `https://mbostock.github.io/stack` (cited on page 36).

Brock, David C. [2017]. *The Improbable Origins of Powerpoint*. IEEE Spectrum 54.11 (02 Nov 2017), pages 42–49. doi:10.1109/mspec.2017.8093800 (cited on page 1).

Cederholm, Dan [2015]. *CSS3 for Web Designers*. 2nd Edition. A Book Apart, 24 Feb 2015. 139 pages. ISBN 1937557200 (cited on page 4).

Chun, Russell [2014]. *Adobe Flash Professional CC*. Classroom in a Book. 11 Aug 2014. 384 pages. ISBN 0133927105 (cited on page 10).

Clover, Juli [2017]. *Apple Makes iMovie, GarageBand, and iWork Apps for Mac and iOS Free for All Users*. MacRumors. 18 Apr 2017. `https://macrumors.com/2017/04/18/apple-imovie-garageband-iwork-free-for-all-users/` (cited on page 1).

Dalgleish, Mark [2015]. *Fathom.js*. GitHub. 06 Aug 2015. `https://github.com/markdalgleish/fathom` (cited on page 19).

Dalgleish, Mark [2018a]. *Bespoke.js*. GitHub. 18 Jan 2018. `https://github.com/bespokejs/bespoke` (cited on page 23).

Dalgleish, Mark [2018b]. *Bespoke.js Example*. 18 Jan 2018. `http://markdalgleish.com/projects/bespoke.js` (cited on page 24).

Doumont, Jean-luc [2002]. *The Three Laws of Professional Communication*. IEEE Transactions on Professional Communication 45.4 (Dec 2002), pages 291–296. ISSN 0361-1434. doi:10.1109/TPC.2002.805164 (cited on page 3).

Doumont, Jean-luc [2005]. *The Cognitive Style of PowerPoint: Slides Are Not All Evil*. Technical Communication 52.1 (01 Feb 2005), pages 64–70. ISSN 0049-3155 (cited on page 3).

Doumont, Jean-luc [2009]. *Trees, Maps, and Theorems*. Principiae, Jan 2009. 178 pages. ISBN 9081367706. `http://treesmapsandtheorems.com/` (cited on page 3).

Friend, James [2018]. *PCE.js – Emulates Mac Plus, PC, & Atari ST in asm.js*. GitHub. 24 Nov 2018. `https://github.com/jsdf/pce` (cited on page 2).

Gabrielle, Bruce R. [2010]. *Speaking PowerPoint: The New Language of Business*. Insights Publishing, 10 Oct 2010. ISBN 098423604X (cited on page 3).

Gaskins, Robert [2012]. *Sweating Bullets: Notes about Inventing PowerPoint*. Vinland Books, 12 Apr 2012. ISBN 0985142421. `https://robertgaskins.com/powerpoint-history/sweating-bullets/gaskins-sweating-bullets-webpdf-isbn-9780985142414.pdf` (cited on page 1).

Google [2006]. *Google Slides*. 09 Mar 2006. `https://google.com/slides/about` (cited on page 31).

Gruber, John [2004]. *Markdown*. 17 Dec 2004. `https://daringfireball.net/projects/markdown` (cited on page 13).

Halácsy, Péter, Peter Arvai, and Adam Somlai-Fischer [2018a]. *Prezi – Online Presentation Tools*. Nov 2018. `https://prezi.com/` (cited on page 31).

Halácsy, Péter, Peter Arvai, and Adam Somlai-Fischer [2018b]. *Prezi – Pricing*. Nov 2018. `https://prezi.com/pricing` (cited on page 33).

Hattab, Hakim El [2018a]. *Reveal.js*. GitHub. 03 Oct 2018. `https://github.com/hakimel/reveal.js` (cited on page 25).

Hattab, Hakim El [2018b]. *Reveal.js – The HTML Presentation Framework*. 03 Oct 2018. `https://revealjs.com` (cited on page 26).

Jobs, Steve [2010]. *Thoughts on Flash*. Apple. Apr 2010. `https://apple.com/hotnews/thoughts-on-flash` (cited on page 10).

Keith, Jeremy and Rachel Andrew [2016]. *HTML5 for Web Designers*. 2nd Edition. A Book Apart, 17 Feb 2016. 92 pages. ISBN 1937557243 (cited on page 4).

Ludvigsen, Holger [2016a]. *Slidifier*. GitHub. 27 Jan 2016. `https://github.com/holgerl/Slidifier` (cited on page 17).

Ludvigsen, Holger [2016b]. *Slidifier Example*. 27 Jan 2016. `http://slidifier.com/slidifier.html` (cited on page 16).

Mahém, Luke, Marcin Wichary, and Eric Bidelman [2013]. *Google HTML5 Slides*. 08 Apr 2013. `https://code.google.com/archive/p/io-2013-slides` (cited on page 19).

Makeev, Vadim [2018a]. *Shower*. GitHub. 08 Oct 2018. `https://github.com/shower/shower` (cited on page 27).

Makeev, Vadim [2018b]. *Shower Presentation Engine*. 08 Oct 2018. `https://shwr.me` (cited on page 28).

Marcotte, Ethan [2014]. *Responsive Web Design*. 2nd Edition. A Book Apart, 02 Dec 2014. ISBN 1937557189 (cited on pages 35, 41).

Marquis, Mat [2016]. *JavaScript For Web Designers*. A Book Apart, 28 Sep 2016. 135 pages. ISBN 1937557464 (cited on page 4).

Meyer, Eric [2005a]. *S5: A Simple Standards-Based Slide Show System*. 28 Jul 2005. `https://meyerweb.com/eric/tools/s5` (cited on page 9).

Meyer, Eric [2005b]. *S5: An Introduction*. 28 Jul 2005. `https://meyerweb.com/eric/tools/s5/s5-intro.html` (cited on page 9).

Nakajima, Pat and Dan Croak [2012]. *Slidedown*. GitHub. 16 Feb 2012. `https://github.com/nakajima/slidedown` (cited on pages 13, 15).

Netlify [2018]. *All-in-One Platform For Automating Modern Web Projects*. 09 Nov 2018. `https://app.netlify.com/start/deploy?repository=https://github.com/shower/shower` (cited on page 27).

NPM [2018]. *NPM Presentation Packages*. 29 Nov 2018. `https://npmjs.com/search?q=presentation` (cited on page 19).

Phi, Jay [2018]. *The Origins of PowerPoint*. 30 Sep 2018. `https://modernslide.com/the-origins-of-powerpoint/` (cited on page 1).

Polacek, John [2015a]. *jQuery Scrolldeck Parallax Plugin Demo*. GitHub. 23 Dec 2015. `https://johnpolacek.github.io/scrolldeck.js/decks/parallax` (cited on page 37).

Polacek, John [2015b]. *Scrolldeck.js*. GitHub. 23 Dec 2015. `https://github.com/johnpolacek/scrolldeck.js` (cited on page 35).

Prezibase [2015]. *Climb to Success – Prezi Template*. 13 Dec 2015. `https://prezi.com/koluwfk7mcq_/climb-to-success-prezi-template` (cited on page 32).

Puppetlabs [2018]. *Showoff*. GitHub. 17 Jul 2018. `https://github.com/puppetlabs/showoff` (cited on pages 33–34).

Raggett, Dave [2005]. *HTML Slidy: Slide Shows in XHTML*. Mar 2005. `https://w3.org/2005/03/slideshow.html` (cited on page 7).

Raggett, Dave [2006a]. *HTML Slidy: Slide Shows in HTML and XHTML*. 2006. `https://w3.org/Talks/Tools/Slidy2` (cited on pages 7–8).

Raggett, Dave [2006b]. *Slidy – A Web-Based Alternative to PowerPoint*. Slides from XTech, Amsterdam. 19 May 2006. `https://w3.org/2006/05/Slidy-XTech` (cited on page 7).

Roemmele, Brian [2013]. *Did Apple buy or create the Keynote software?* Quora. 21 Aug 2013. `https://quora.com/Apple-company/Did-Apple-buy-or-create-the-Keynote-software/answers/2994640` (cited on page 1).

Rota, Andrew [2015]. *Google-IO-Slides-Fork*. GitHub. 22 Mar 2015. `https://github.com/andrewrota/google-io-slides-fork` (cited on pages 19–20).

Rouget, Paul [2017a]. *DZSlides*. GitHub. 30 Oct 2017. `https://github.com/paulrouget/dzslides` (cited on page 21).

Rouget, Paul [2017b]. *DZSlides Example*. 30 Oct 2017. `http://paulrouget.com/dzslides` (cited on page 22).

Schofnegger, Markus [2015]. *Rslidy: Responsive Presentation Slides in HTML5 and CSS3*. 02 Nov 2015. `https://ftp.isds.tugraz.at/pub/theses/mschofnegger-2015-bsc.pdf` (cited on page 38).

Slides [2018]. *Slides – Create and Share Presentations online*. 03 Oct 2018. `https://slides.com` (cited on page 25).

StatCounter [2018]. *Browser Market Share Worldwide*. 09 Nov 2018. `http://gs.statcounter.com/browser-market-share` (cited on page 4).

Stoll, Martin [2016a]. *Diascope Example*. 30 Apr 2016. `http://diascope.sourceforge.net/resources/template.txt` (cited on page 11).

Stoll, Martin [2016b]. *Diascope Homepage*. 30 Apr 2016. `http://diascope.sourceforge.net/` (cited on page 10).

Szopka, Bartek and Henrik Ingo [2018a]. *Impress.js*. GitHub. 22 Oct 2018. `https://github.com/impress/impress.js` (cited on page 27).

Szopka, Bartek and Henrik Ingo [2018b]. *Impress.js Example*. 22 Oct 2018. `https://impress.js.org` (cited on page 28).

Troughton, Caleb [2016a]. *Deck.js*. GitHub. 04 May 2016. `https://github.com/imakewebthings/deck.js` (cited on page 20).

Troughton, Caleb [2016b]. *Deck.js Docs*. 04 May 2016. `http://imakewebthings.com/deck.js/docs` (cited on page 20).

Troughton, Caleb [2016c]. *Deck.js Example*. 04 May 2016. `http://imakewebthings.com/deck.js` (cited on page 21).

Troughton, Caleb [2016d]. *Deck.js Wiki*. GitHub. 04 May 2016. `https://github.com/imakewebthings/deck.js/wiki` (cited on page 20).

Unicode [2018]. *Unicode Version 11*. 05 Jun 2018. `https://unicode.org/versions/Unicode11.0.0/` (cited on page 13).

Verou, Lea [2018a]. *Inspire.js*. GitHub. 29 Sep 2018. `https://github.com/LeaVerou/inspire.js` (cited on page 23).

Verou, Lea [2018b]. *Inspire.js: A Brief Introduction*. 29 Sep 2018. `https://inspirejs.org` (cited on page 25).

W3Counter [2018]. *Browser & Platform Market Share*. W3Counter: Global Web Stats. 01 Oct 2018. `https://w3counter.com/globalstats.php?year=2018&month=9` (cited on page 3).

Young, Kyle Robinson and Tobias Koppers [2016]. *Jmpress.js*. GitHub. 20 Apr 2016. `https://github.com/jmpressjs/jmpress.js` (cited on page 27).