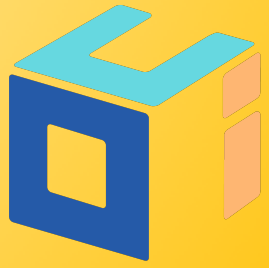
Modern Web Dev. Jump Start

slides at https://github.com/mvolkmann/talks
code at https://github.com/mvolkmann/web-dev-jump-start

R. Mark Volkmann
Object Computing, Inc.
http://objectcomputing.com
Email: mark@objectcomputing.com
Twitter: @mark_volkmann
GitHub: mvolkmann



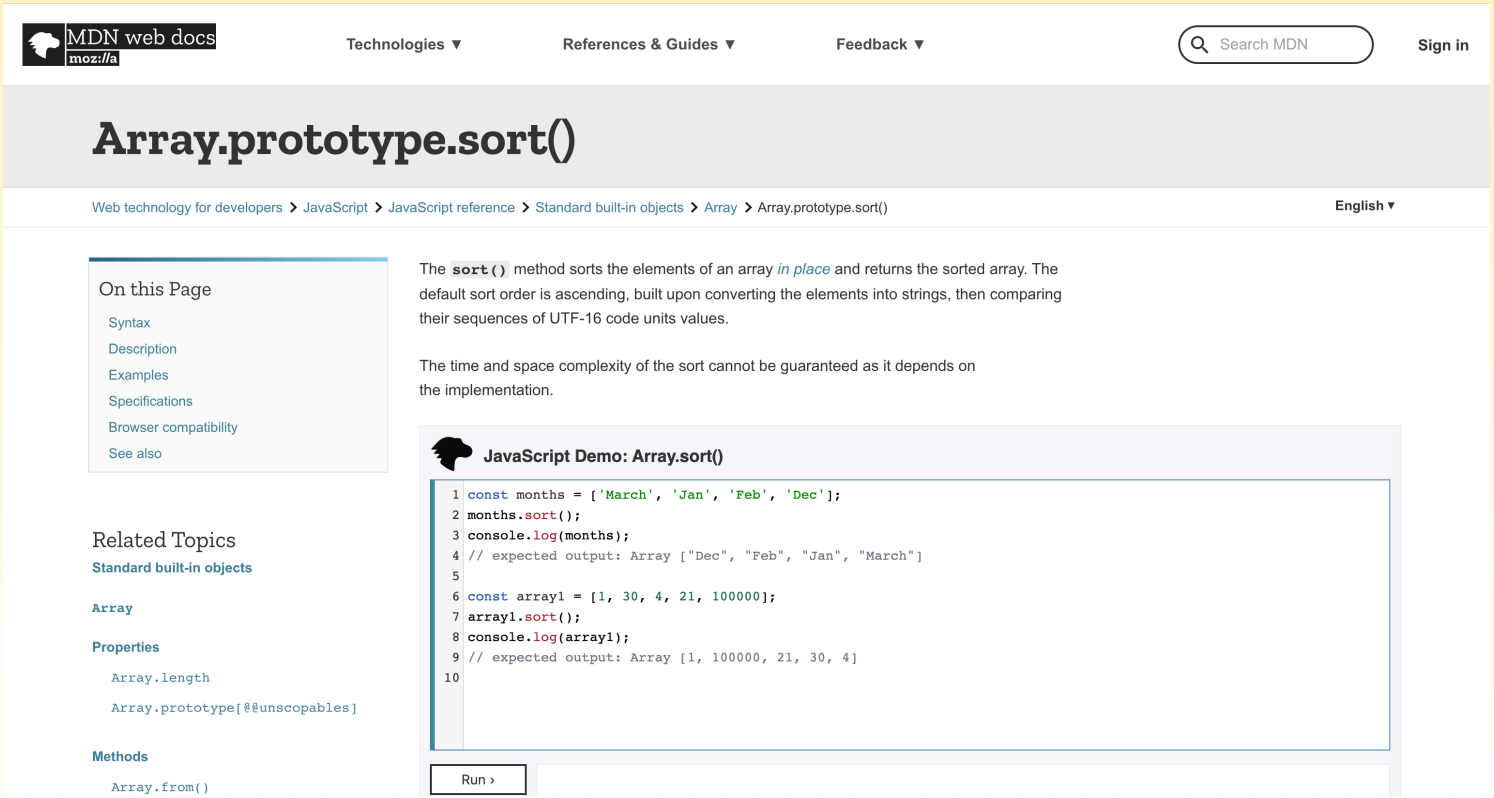
OCI | TRAINING

Topics

- Important resources
- Hosting web sites for free using GitHub Pages
- Lighthouse tool for auditing web sites and web apps
- Minimum HTML structure for a good Lighthouse score
- CSS tips including flexbox and grid layout
- Browser DOM overview
- Sample app with plain HTML, CSS, and JavaScript
- Svelte and the same sample app

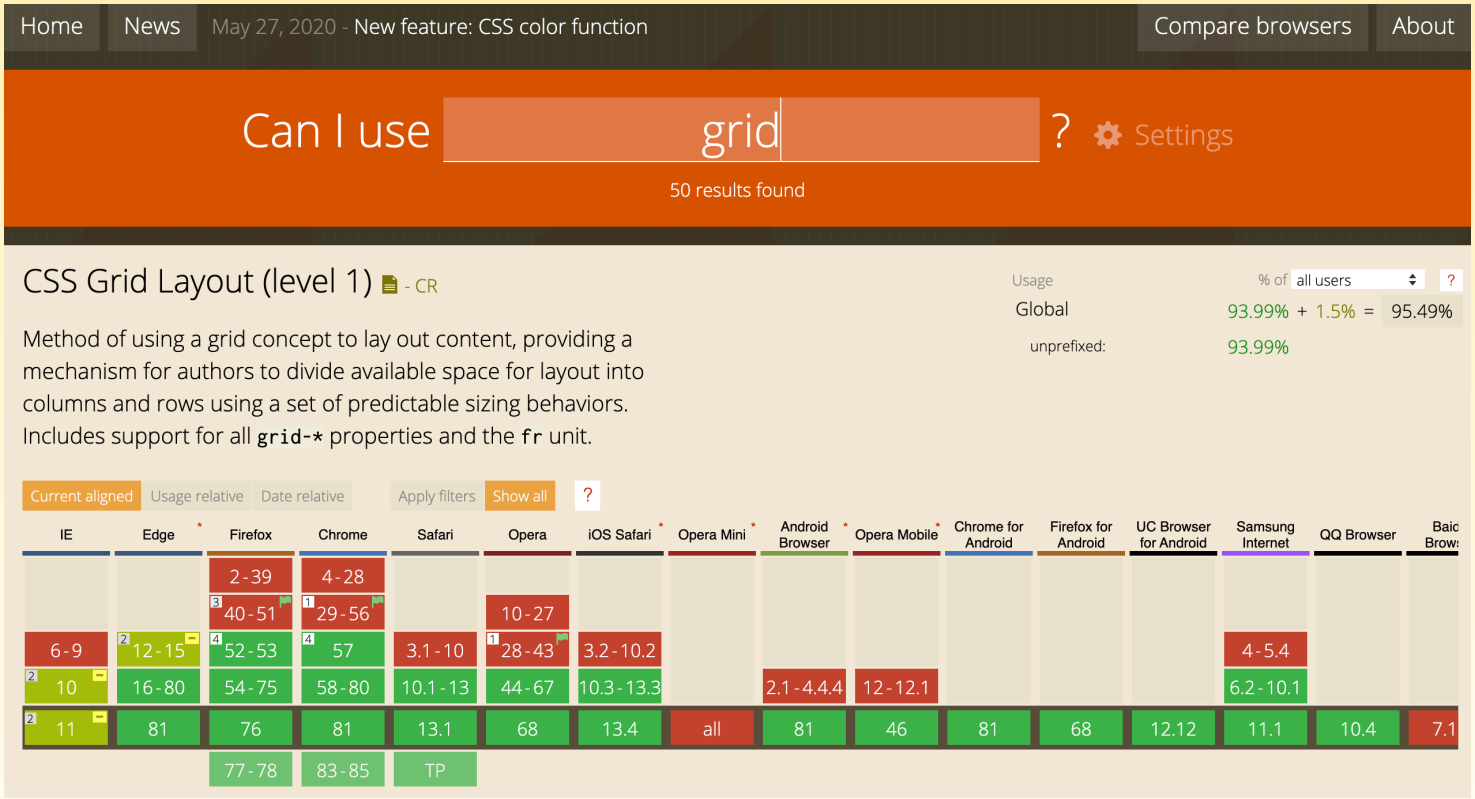
Important Resources ...

- Mozilla Developer Network (MDN)
  - start web searches with "mdn"
  - ex. **`mdn array sort`**
  - finds reliable information quickly



… Important Resources

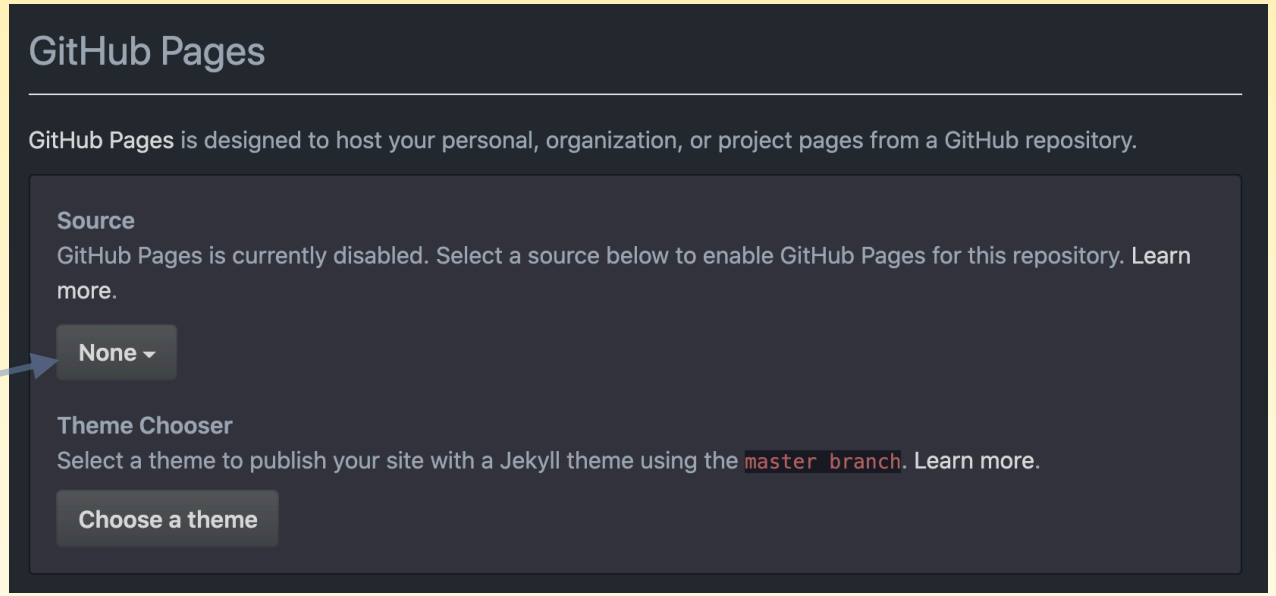
- Can I Use - https://caniuse.com
  - fast way to determine browser support for a specific feature
    - ex. search for "grid"
  - provides details on deviations from standard



- Podcasts
  - Syntax, JS Party, Real Talk JavaScript, ShopTalk, The Changelog, JavaScript Jabber

GitHub Pages ...

- Provides a very easy, free way to host static web sites
- Steps
  - create a GitHub repository
  - add static assets such as HTML, CSS, JavaScript, and image files
  - browse GitHub repository
  - click "Settings" near upper-right
  - scroll to "GitHub Pages" section
  - in "Source" drop-down, select "master branch"
  - wait about 30 seconds
  - browse **`https://username.github.io/repo-name/`**



… GitHub Pages

- To make changes to site
  - modify existing files and add new ones
  - commit changes
  - push to master branch
  - site will update automatically in about 30 seconds

Lighthouse ...

- Free tool used by Chrome DevTools on "Lighthouse" tab (was "Audits")
  - can also be run from a command line or a Node application
- Audits many aspects of a web app including
  - performance
  - progressive web app metrics
  - accessibility
  - search engine optimization (SEO)
- Provides suggestions on improving the app
- https://developers.google.com/web/tools/lighthouse

… Lighthouse

- Steps to use
  1. open web site to be tested in Chrome
  2. open DevTools
  3. click "Lighthouse" tab
  4. check checkboxes for categories to be tested (typically all of them)
  5. press "Generate report" button
  6. to keep current test results and open a new tab for the next test, press the "+" in the upper-left corner of Audits tab
  7. visit another page or state of the site
  8. press "Generate report" button again
  9. repeat steps 6 to 8 for each page/state of the site, noting reported issues for each

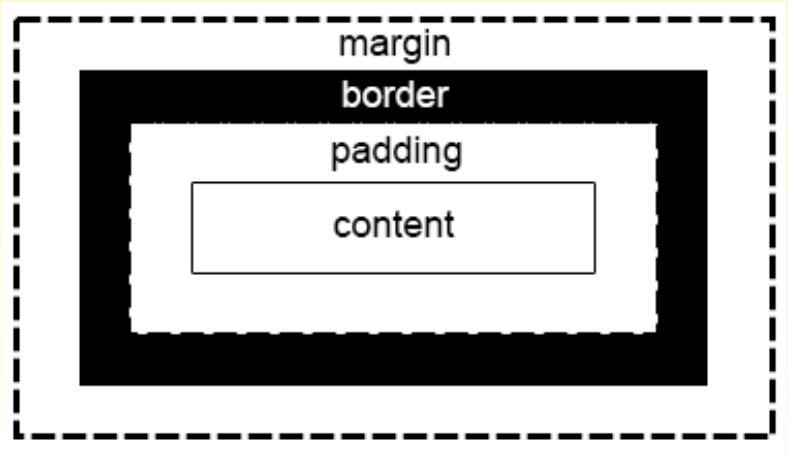
Minimum HTML

- This HTML gets a perfect Lighthouse score for both desktop and mobile

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Simple Site</title>
    <meta charset="utf-8" />
    <meta name="description" content="simple site" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
  </head>
  <body>
    Welcome to my simple site!
  </body>
</html>
```

CSS Tips

- 3 ways to identify target elements
  - element name
    - ex. `section`
  - id (#)
    - ex. `#login-form`
  - CSS class name (dot)
    - ex. `.sport-name`
- Box model



`box-sizing` defaults to `content-box`, but can be set to `border-box`

- Media queries

```
<style>
  @media (max-width: 760px) {
    .container {
      flex-direction: column;
    }
  }
</style>
```

- Transitions

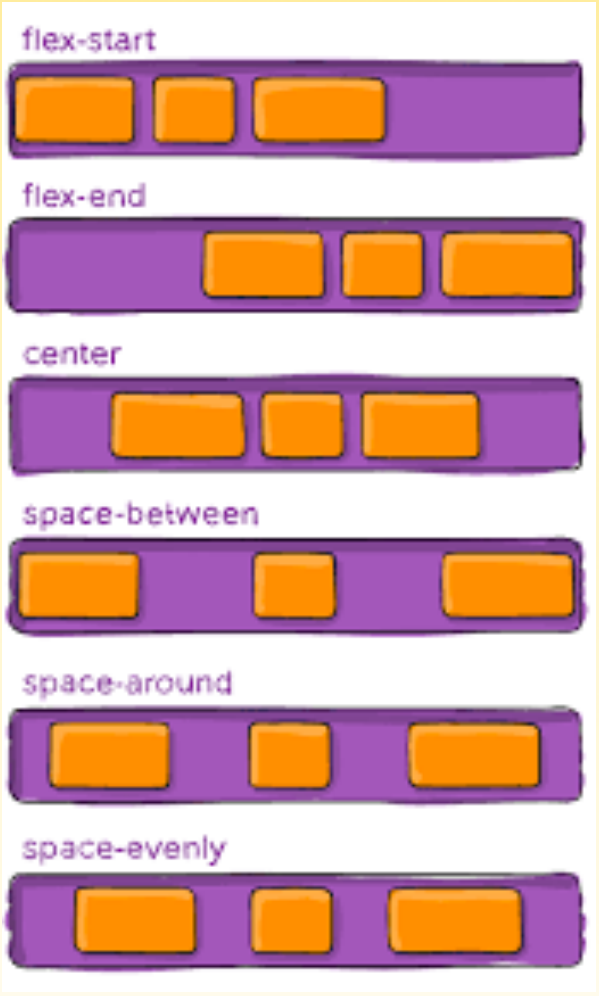
```
.arrow {
  transition-duration: 0.5s;
  transition-property: transform;
  transform: rotate(0deg);
}

.arrow.selected {
  transform: rotate(90deg);
}
```

we will see this in action later

CSS Flexbox

- One-dimensional element layout
- To use, **`display: flex`**
- **`justify-content`** specifies alignment on major axis (defaults to row)
  - commonly used values are **`center`**, **`flex-start`**, **`flex-end`**, **`space-between`**, **`space-around`**, and **`space-evenly`**
- **`align-items`** specifies alignment on minor axis
  - values are **`center`**, **`flex-start`**, and **`flex-end`**
- **`flex-direction`** values are **`row`** (default) and **`column`** (swaps major and minor axis)
- Example

```
.row {
  display: flex;
  justify-content: space-between;
  align-items: center;
}
```



CSS Grid

- Two-dimensional element layout
- To use, **`display: grid`**
- **`grid-template-columns`**
  specifies number of columns and their widths
- **`grid-template-rows`**
  specifies number of rows and their heights
- **`grid-template-areas`**
  associates a name with each grid cell
- **`grid-area`**
  associates a grid name with an element
- **`grid-gap`**
  specifies space between cells

The special unit **`fr`** (stands for fractional unit) can be used for column widths and row heights. For example,
`grid-template-columns: 100px 1fr 2fr;`
creates three columns where
the first is 100px wide,
the second is 1/3 of the remaining space, and
the third is 2/3 of the remaining space.

CSS Grid Demo ...

... CSS Grid Demo ...

demo.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>CSS Grid Demo</title>
    <meta charset="utf-8">
    <meta name="description" content="CSS Grid Demo">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="demo.css">
  </head>
  <body>
    <header>
      <h1>CSS Grid Demo</h1>
    </header>
    <nav>
      <a href="./beverly-hillbillies.html" target="frame">Beverly Hillbillies</a>
      <a href="./friends.html" target="frame">Friends</a>
      <a href="./grease.html" target="frame">Grease</a>
    </nav>
    <iframe name="frame" src="welcome.html" title="main content"></iframe>
    <footer>&copy; CSS Grid Demo, 2020</footer>
  </body>
</html>
```

... CSS Grid Demo ...

demo.css

```
body {
  --footer-height: 60px;
  --header-height: 140px;
  --nav-width: 245px;
  --primary-color: cornflowerblue;
  --secondary-color: orange;
  --tertiary-color: white;

  display: grid;
  grid-template-columns: var(--nav-width) 1fr;
  grid-template-rows: var(--header-height) 1fr var(--footer-height);
  grid-template-areas:
    'header header'
    'nav main'
    'footer footer';

  color: white;
  font-family: sans-serif;
  height: 100vh;
  margin: 0;
  width: 100vw;
}
```

CSS variables

vh = view height
vw = view width

... CSS Grid Demo

demo.css

```
footer {
  display: flex;
  align-items: center;

  background-color: black;
  grid-area: footer;
  padding-left: 20px;
}

header {
  display: flex;
  justify-content: center;
  align-items: center;

  background-color: var(--primary-color);
  grid-area: header;
  margin: 0;
  padding: 1rem;
}
```

demo.css

```
iframe {
  background-color: var(--tertiary-color);
  border: none;
  grid-area: main;
  height: 100%;
  width: 100%;
}

nav {
  background-color: var(--secondary-color);
  color: white;
  grid-area: nav;
  overflow: auto;
  padding: 1rem;
}

nav > a {
  color: white;
  display: block;
  font-size: 1.5rem;
  margin: 0.5rem 0;
  text-decoration: none;
}
```

Document Object Model (DOM)

- A tree of JavaScript objects that represent what the browser will render
- Objects belong to classes that define properties and methods
- Primary classes are
  `Window`, `Document`, `Node`, `NodeList`, `Element`, and `Text`
- Modifying the DOM changes what the browser renders

DOM Highlights

- **Document**
  - `document.createElement(name)`
- **Node**
  - base class of **Element**
- **Element**
  - `element.textContent = text;`
  - `element.innerHTML = htmlText;`
  - `element.parentElement`
  - `element.children`
  - `parentElement.append(childElement)`
  - `parentElement.remove(childElement)`
  - `element.classList.add('class-name');`
  - `element.classList.remove('class-name');`
  - `element.classList.toggle('class-name');`
  - `element.addEventListener('event-name', handlerFunction);`

similar to **textContent**, but parsed for HTML content

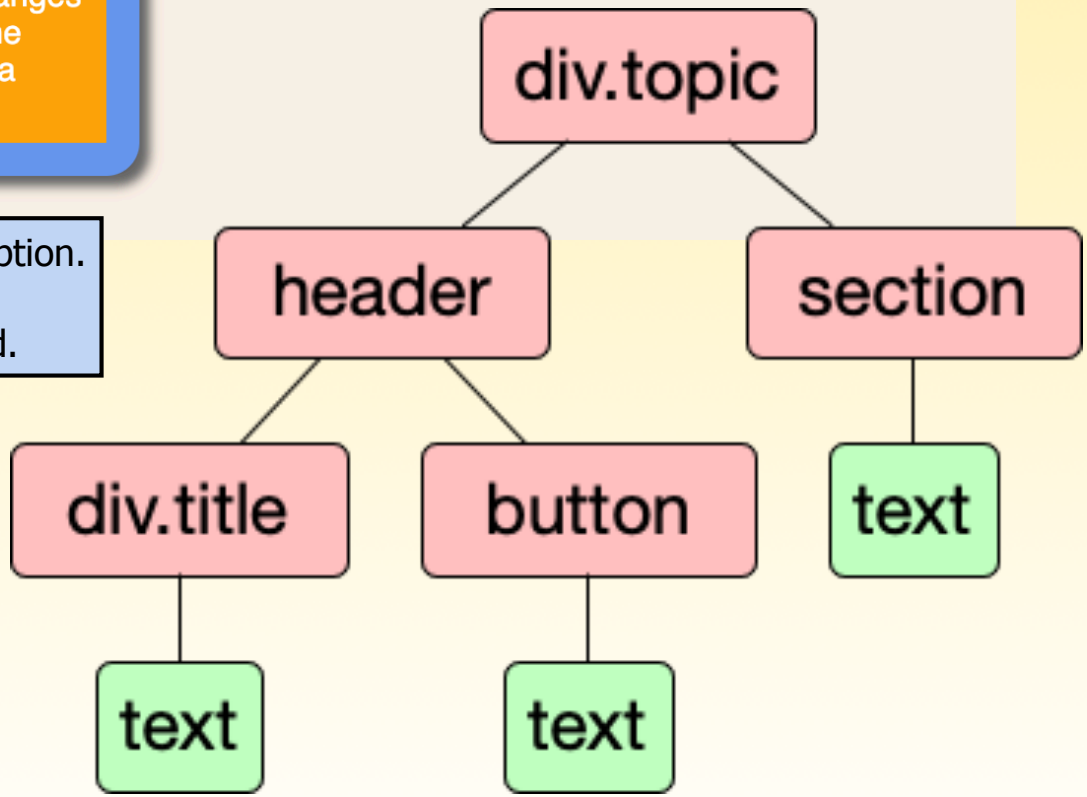
HTML Demo

CSS Transitions ▼

CSS transitions provide a way to control animation speed when changing CSS properties. Instead of having property changes take effect immediately, you can cause the changes in a property to take place over a period of time.

CSS Flexible Box Layout ▶

Clicking a triangle shows the description. The triangle rotates and the description is gradually revealed.



Demo HTML

demo.html

```
<DOCTYPE html>
<html lang="en">
  <head>
    <title>CSS Transitions</title>
    <meta charset="utf-8" />
    <meta name="description" content="CSS transitions" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />

    <script>
      <!-- See "Demo Script" slides ahead. -->
    </script>

    <style>
      <!-- See "Demo Style" slide ahead. -->
    </style>
  </head>
  <body>
  </body>
</html>
```

will populate in JavaScript code

Demo Script ...

This builds the DOM for a "topic" and appends it to the **body**.

```
function getTopic(titleText, descriptionText) {
  const topic = document.createElement('div');
  topic.classList.add('topic');

  const header = document.createElement('header');
  topic.append(header);

  const title = document.createElement('div');
  title.classList.add('title');
  title.textContent = titleText;
  header.append(title);

  const button = document.createElement('button');
  button.classList.add('toggle');
  button.innerHTML = '&#x25B6;';
  button.addEventListener('click', handleClick);
  header.append(button);

  const section = document.createElement('section');
  section.textContent = descriptionText;
  topic.append(section);

  return topic;
}
```

Unicode triangle pointing right

defined on next slide

... Demo Script

```
function handleClick(event) {
  const {target} = event;
  const topic = target.parentElement.parentElement;
  topic.classList.toggle('show');
}

let description = `
  CSS transitions provide a way to control
  animation speed when changing CSS properties.
  Instead of having property changes take effect immediately,
  you can cause the changes in a property
  to take place over a period of time.
`;
document.body.append(getTopic('CSS Transitions', description));

description = `
  CSS Flexible Box Layout is a module of CSS that defines a CSS box model
  optimized for user interface design, and the layout of items in
  one dimension.  In the flex layout model, the children of a
  flex container can be laid out in any direction, and can "flex" their
  sizes, either growing to fill unused space or shrinking to avoid
  overflowing the parent.  Both horizontal and vertical alignment of
  the children can be easily manipulated.
`;
document.body.append(getTopic('CSS Flexible Box Layout', description));
```

The CSS expects the **show** class to be added to the topic **div**.

Demo Style

```
body {
  background-color: linen;
  font-family: sans-serif;
  padding: 1rem;
}

.topic {
  box-shadow: 5px 5px 5px #666;
  display: inline-block;
  background-color: cornflowerblue;
  border-radius: 1rem;
  color: white;
  margin: 0 1rem 1rem 0;
  padding: 1rem;
  vertical-align: top;
  width: 350px;
}

.topic > header {
  display: flex;
  justify-content: space-between;
  align-items: center;

  font-size: 1.5rem;
  font-weight: bold;
}
```

```
.topic > header > button {
  background-color: transparent;
  border: none;
  color: inherit;
  font-size: 2rem;
  outline: none;
  transition-duration: 0.5s;
  transition-property: transform;
  transform: rotate(0deg);
}

.topic.show > header > button {
  transform: rotate(90deg);
}
```

```
.topic > section {
  background-color: orange;
  margin-top: 0;
  max-height: 0;
  padding: 0 0.5rem;
  overflow-y: hidden;
  transition-duration: 0.5s;
  transition-property:
    margin-top, max-height, padding;
}

.topic.show > section {
  margin-top: 1rem;
  max-height: 100vh;
  padding-bottom: 0.5rem;
  padding-top: 0.5rem;
}
```

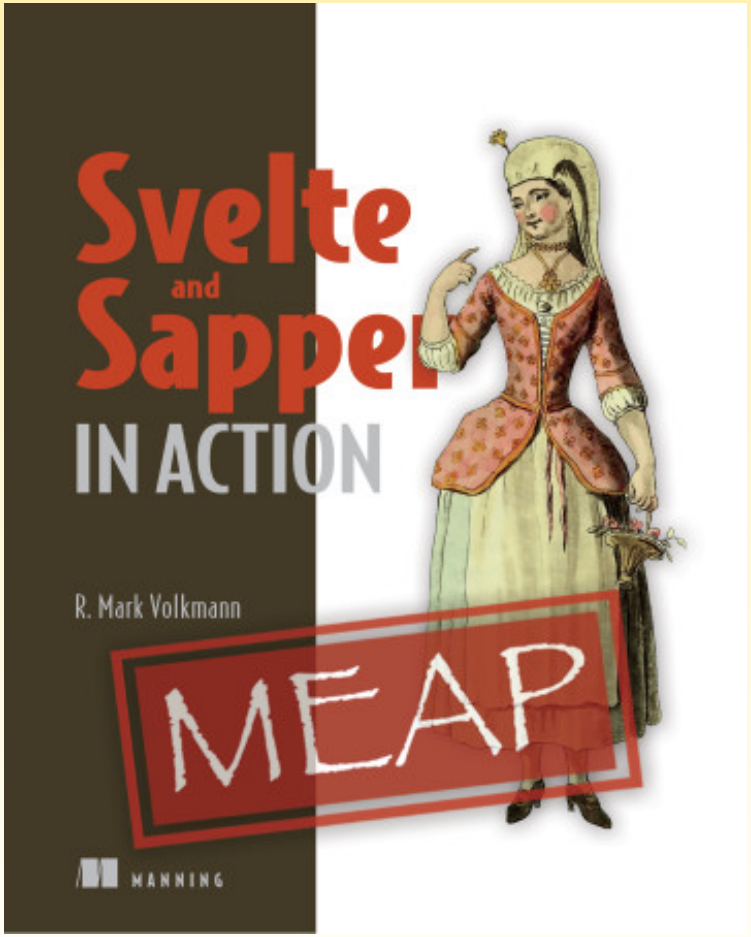
no limit

Eleventy

- Recommended option for building static web sites
- Wish I had time to cover this!
- See https://mvolkmann.github.io/blog/topics/#/blog/eleventy/

Svelte Intro.

- Recommended framework for building dynamic web applications



Getting Started With Svelte

- **`npx degit sveltejs/template app-name`**
- **`cd app-name`**
- **`npm install`**
- **`npm run dev`**
- browse localhost:5000

Svelte `Topic` Component

src/Topic.svelte

```
<script>
  export let title;
  export let description;

  let show = false;

  $: classes = 'topic' + (show ? ' show' : '');

  const handleClick = () => (show = !show);
</script>

<div class={classes}>
  <header>
    <div class="title">{title}</div>
    <button class="toggle" on:click={handleClick}>&#x25b6;</button>
  </header>
  <section>{description}</section>
</div>

<style>
  <!-- Same CSS rules as before. -->
<style>
```

reactive declaration

Svelte App Component ...

src/App.svelte

```
<script>
  import Topic from './Topic.svelte';

  const topics = [
    {
      title: 'CSS Transitions',
      description: `
        CSS transitions provide a way to control
        animation speed when changing CSS properties.
        Instead of having property changes take effect immediately,
        you can cause the changes in a property
        to take place over a period of time.
      `
    },
    {
      title: 'CSS Flexible Box Layout',
      description: `
        CSS Flexible Box Layout is a module of CSS that defines a CSS box model
        optimized for user interface design, and the layout of items in
        one dimension.  In the flex layout model, the children of a
        flex container can be laid out in any direction, and can "flex" their
        sizes, either growing to fill unused space or shrinking to avoid
        overflowing the parent.  Both horizontal and vertical alignment of
        the children can be easily manipulated.
      `
    }
  ];
</script>
```

... Svelte App Component

```
{#each topics as {title, description}}
  <Topic {title} {description} />
{/each}

<style>
  :global(body) {
    background-color: linen;
    font-family: sans-serif;
    padding: 1rem;
  }

  @media (max-width: 640px) {
    :global(body) {
      display: flex;
      flex-direction: column;
    }
  }
</style>
```

for better mobile layout

Conclusion

- Use MDN and caniuse to find answers to your web dev. questions
- Listen to great podcasts
- Host web sites for free using GitHub Pages
- Keep improving your web sites and apps until they achieve good Lighthouse scores
  - shoot for all 100's
- Rock some CSS, including flexbox/grid layout and transitions
- Learn more about Svelte