07: Web Dev Studio

Review

Semantic HTML

CSS Selectors and Styling

CSS Box Model

Layouts with Flexbox and Grids

Working with External Libraries

External CSS Libraries

- **What are CSS libraries?**
  - Pre-written collections of styles, effects, and animations.
- **Benefits of CSS libraries**
  - Speed up development.
  - Ensure consistency across designs.
  - Offer responsive, cross-browser-compatible styles.
- **Popular libraries we'll cover**: Bootstrap, Tailwind CSS, Animate.css, and Hover.css.

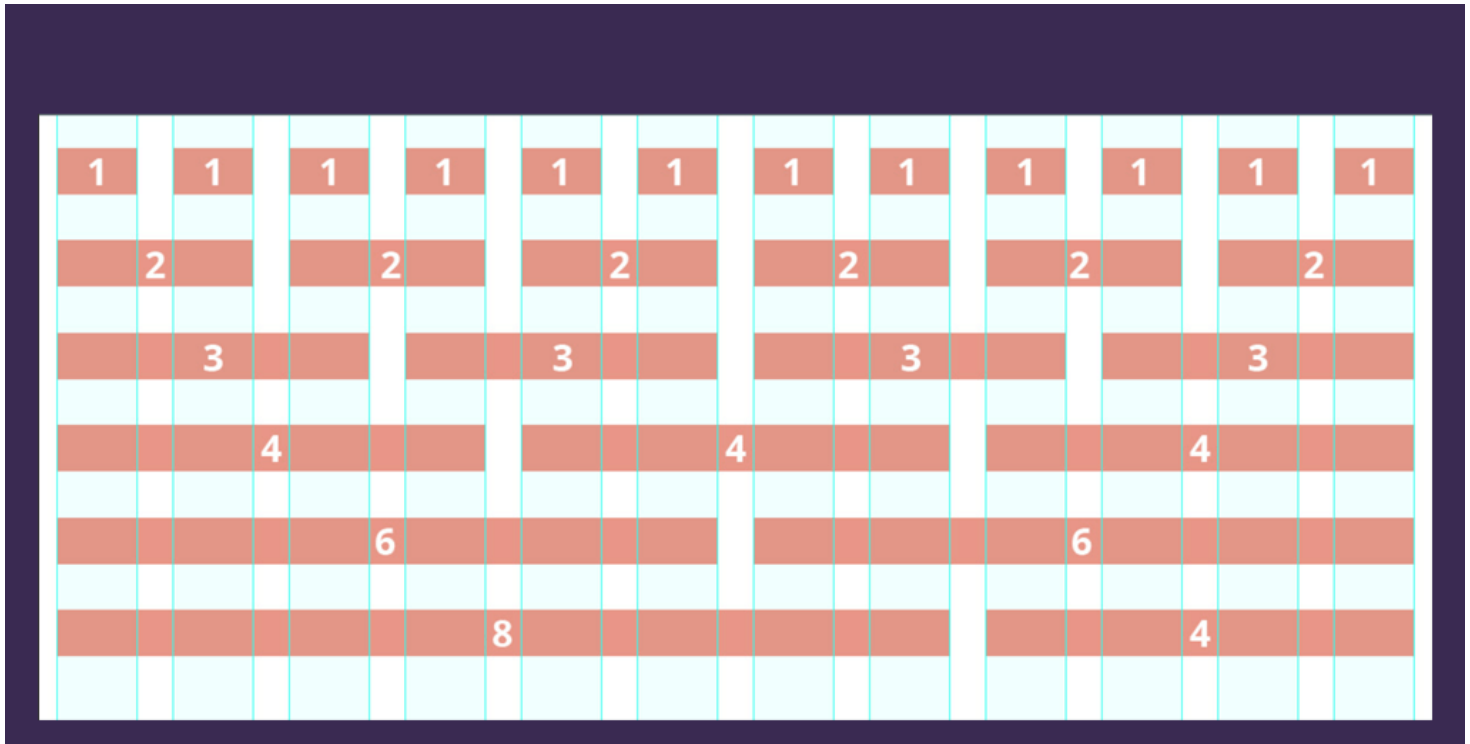
Including External CSS

- External CSS can be linked with the <link> tag just like our local CSS files
- Instead, most commonly we point the source to a CDN that's hosting the file

```
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css">
```

Bootrstrap

- Bootstrap is one of the most popular CSS libraries for creating consistent components and grid-based layouts
- It uses a 12-column grid system to create layouts



Bootstrap Components

- Bootstrap also offers a pre-defined set of styles and components with its library, along with variants of each component

TailwindCSS

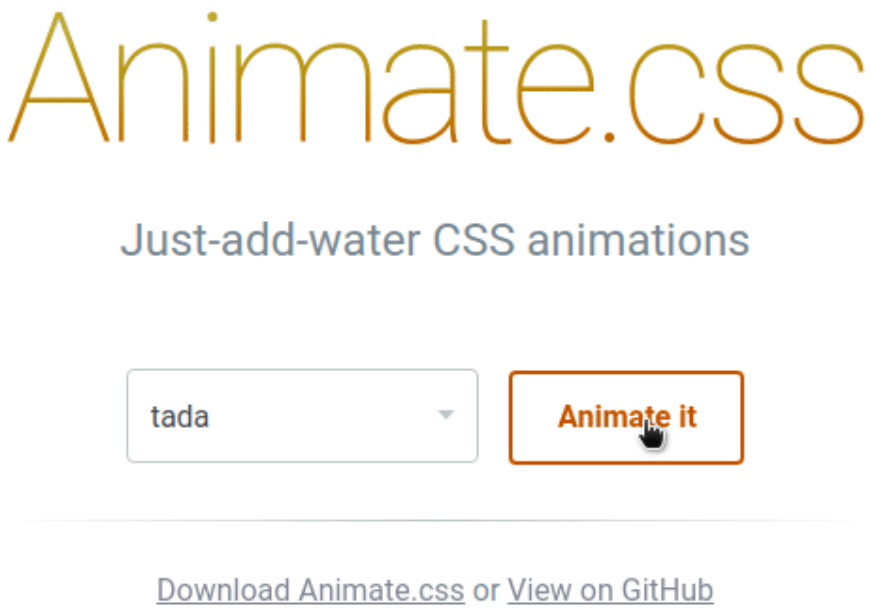
- Tailwind is a utility-based approach to defining styles
- It offers a short-hand way to write regular CSS properties within the HTML
- These classes are then compiled to output vanilla CSS
- Documentation: https://tailwindcss.com/

TailwindCSS

```
<button class="bg-blue-500 hover:bg-blue-700 text-white font-bold py-2 px-4 rounded">
Button
</button>
```

Animate.CSS

- Easy to use pre-defined animations
- Documentation: https://animate.style/

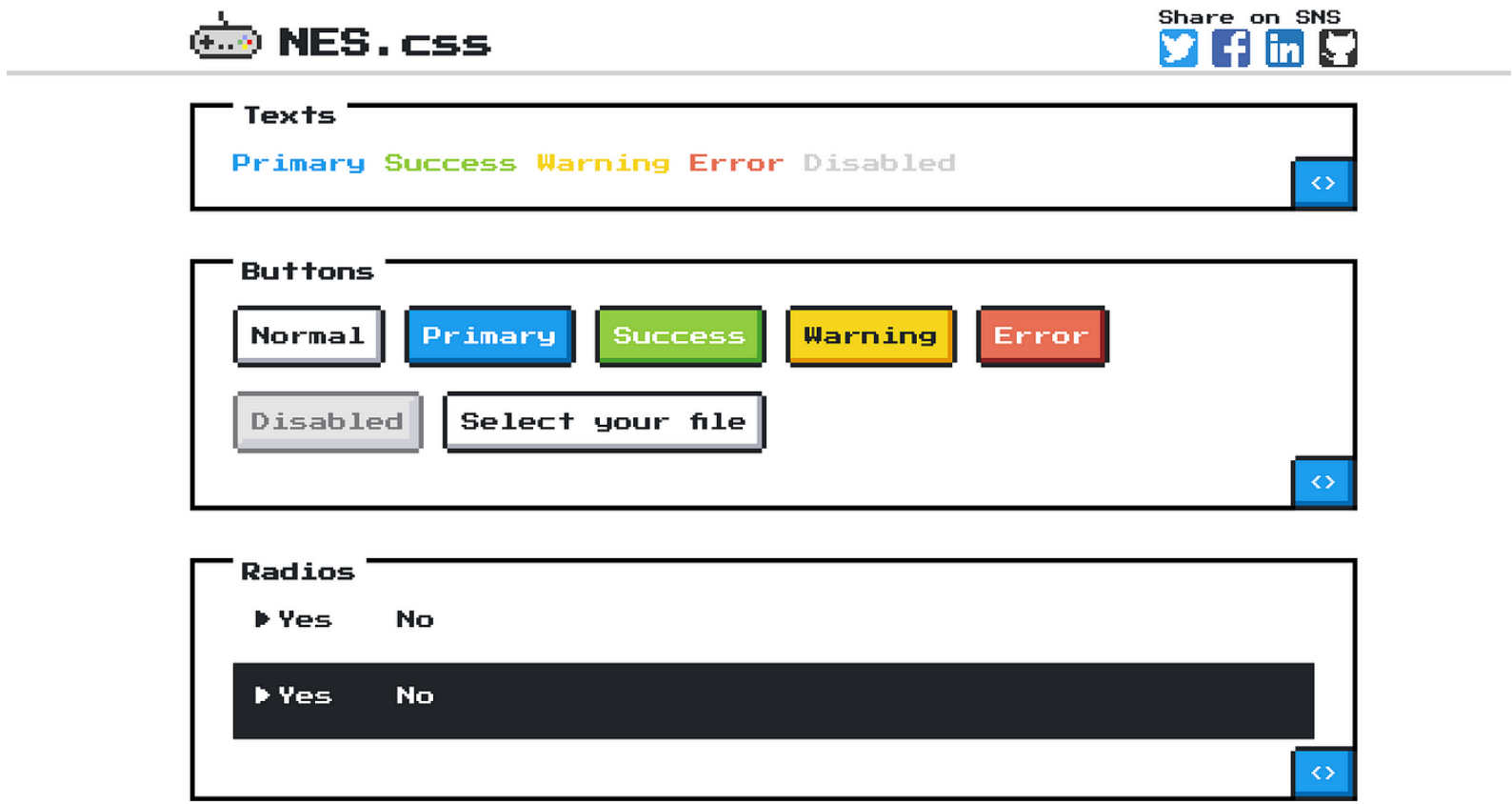


Hover.css

- Hover effects and transitions
- Documentation: https://ianlunn.github.io/Hover/

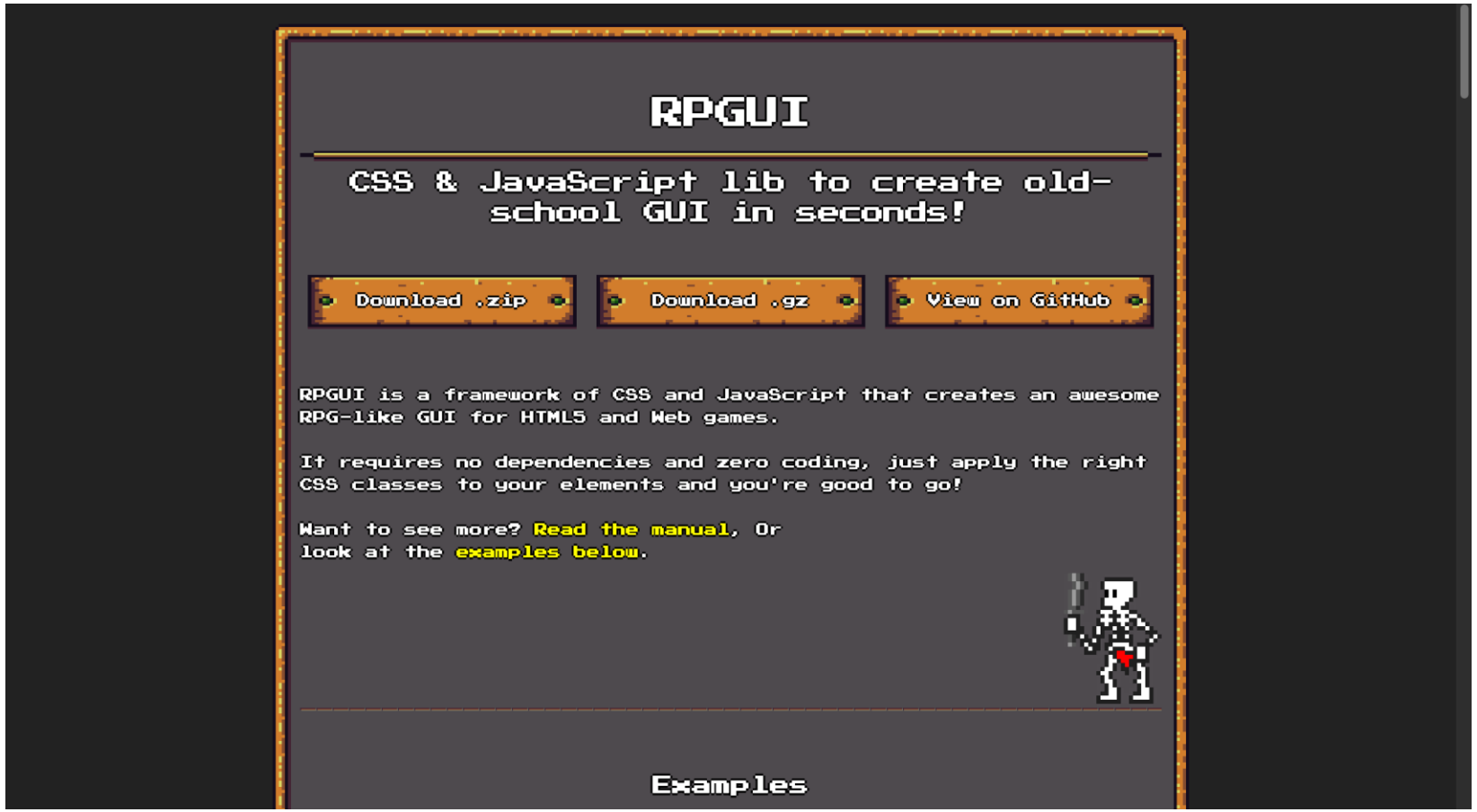
Specialized CSS Libraries

- NES.css - https://nostalgic-css.github.io/NES.css/



Specialized CSS Libraries

- RPGUI - https://ronenness.github.io/RPGUI/



Specialized CSS Libraries

- 98.css - https://jdan.github.io/98.css/

Specialized CSS Libraries

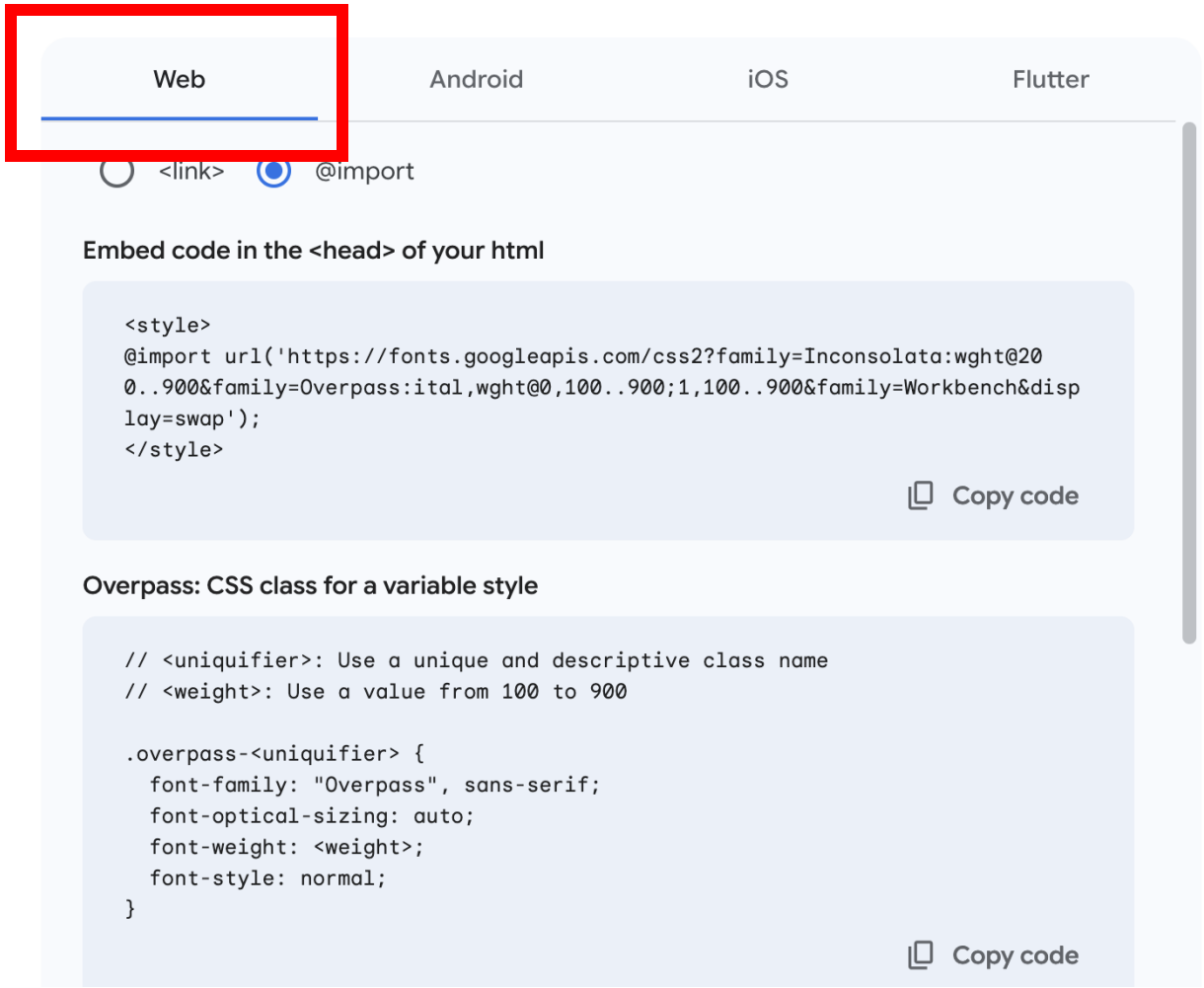
- XP.css - https://botoxparty.github.io/XP.css/

Using External Webfonts Fonts

- Collections
  - Google Fonts (free)
  - Adobe Fonts (paid)
  - Fontshare (free)

- Independent Font Foundries

Using External Webfonts Fonts

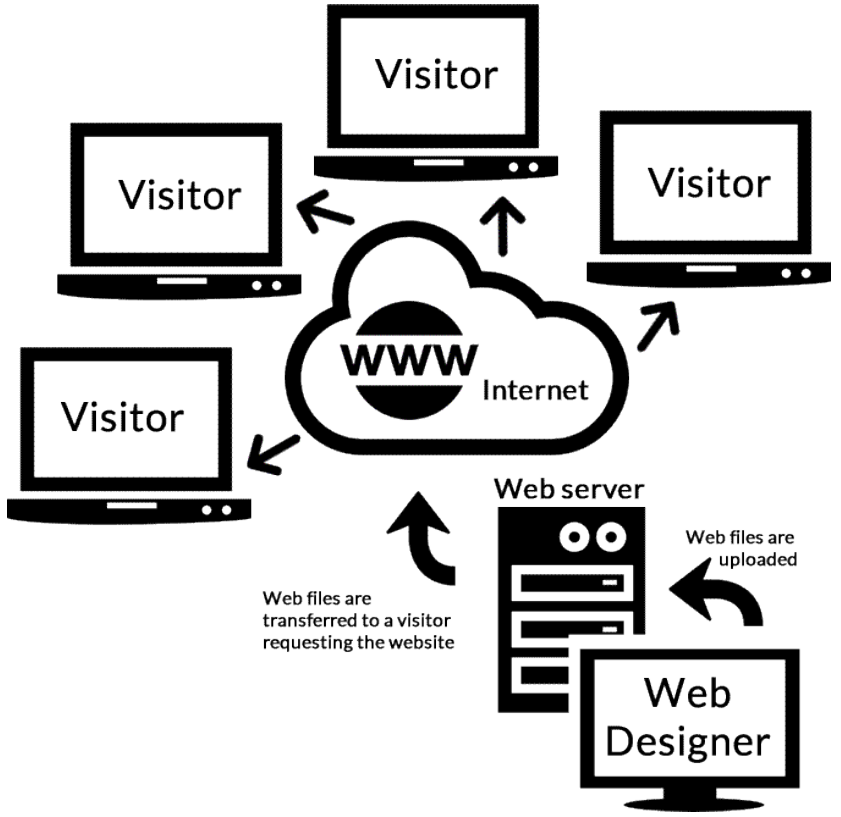


JavaScript & Document Object Model

How does a website work?

- What happens when I visit a website?

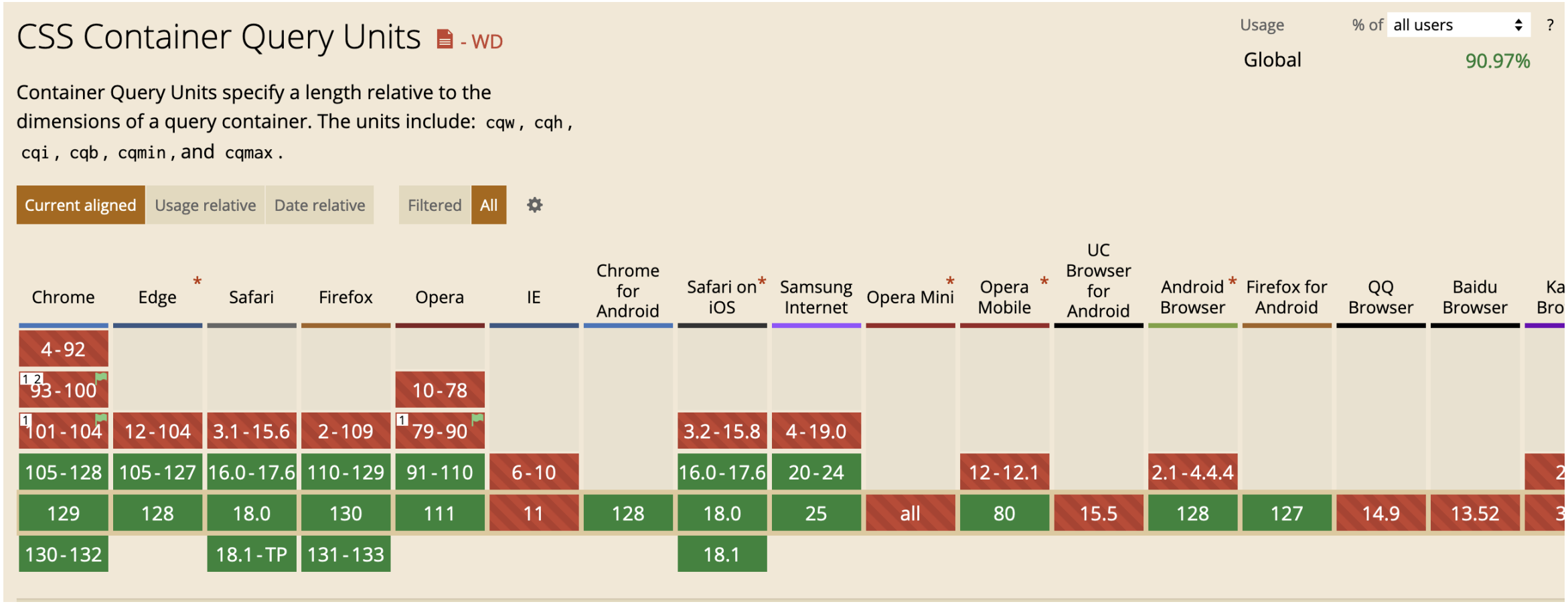
- What is ‘client’?
- What is ‘server’?



Browsers' Role

- Browsers are responsible to render our code as websites to users
- Different browsers use different browser 'engines'
- Engines are an implementation of standard conventions

Why should I care? Compatibility!



CSS Container Query Units - WD

Usage: % of all users — Global: 90.97%

Container Query Units specify a length relative to the dimensions of a query container. The units include: `cqw`, `cqh`, `cqi`, `cqb`, `cqmin`, and `cqmax`.

Current aligned | Usage relative | Date relative | Filtered | All

Chrome	Edge *	Safari	Firefox	Opera	IE	Chrome for Android	Safari on iOS *	Samsung Internet	Opera Mini *	Opera Mobile *	UC Browser for Android	Android Browser *	Firefox for Android	QQ Browser	Baidu Browser	Ka Bro
4-92																
1 2 93-100				10-78												
1 101-104	12-104	3.1-15.6	2-109	1 79-90			3.2-15.8	4-19.0								
105-128	105-127	16.0-17.6	110-129	91-110	6-10		16.0-17.6	20-24		12-12.1		2.1-4.4.4				2
129	128	18.0	130	111	11	128	18.0	25	all	80	15.5	128	127	14.9	13.52	3
130-132		18.1-TP	131-133				18.1									

View Transition example

Article 1 Article 2

Article 1 content

photo credit: Michael Kirsh

Lorem ipsum dolor sit amet, consectetur adipiscing elit
facilisis vel mauris in, luctus semper turpis. Aliquam lob
feugiat. Nulla aliquet ante laoreet enim maximus mollis

View Tr

Limi

Provides a m
between diff
contents in a
transitions, s
is being plan

Current aligned | Usage relative | Date relative | Filtered

Open Bug 1909173 (dt-view-transition) Opened 2 months ago Updated 2 days ago

[META] Add support for CSS View Transitions level 1 (SPA)

Categories

Product: DevTools
Component: Inspector

Type: task
Priority: Not set Severity: --

Tracking

Status: NEW

People (Reporter: nchevobbe, Unassigned)

References (Blocks 1 open bug)

Details (Keywords: meta)

Bottom ↓ | Tags | Timeline

Nicolas Chevobbe [:nchevobbe] Reporter
Description • 2 months ago

CSS view transition for single page application is going to be implemented in the coming months. Let's have this bug to cover needed in DevTools to avoid issues / developer confusion

Chrome	Edge *	Safari	Firefox	Opera	IE	Chrome for Android	Safari on iOS *	Samsung Internet	Opera Mini *	Opera Mobile *	Browser for Android	Android Browser *	Firefox for Android	QQ Browser	Baidu Browser	Ka Bro
4-110	12-110			10-96				4-22								
111-128	111-127	3.1-17.6	2-129	97-110	6-10		3.2-17.6	23-24		12-12.1		2.1-4.4.4				2
129	128	18.0	130	111	11	128	18.0	25	all	80	15.5	128	127	14.9	13.52	3
130-132		18.1-TP	131-133				18.1									

Browser Engines

- WebKit by Apple
  - Safari
- Gecko by Mozilla Foundation
  - Firefox
- Blink by Google
  - Google Chrome
  - Microsoft Edge
  - Arc
  - Brave
  - Opera

- Each browser engine is equipped with its own JavaScript engine
- JavaScript engines run along with the rendering engines via the Document Object Model to enable dynamic interactions on the web page

JavaScript Engines

- WebKit by Apple uses JavaScript Core
  - Safari
- Gecko by Mozilla Foundation uses Spider Monkey
  - Firefox
- Blink by Google uses V8
  - Google Chrome
  - Microsoft Edge
  - Arc
  - Brave
  - Opera

V8 engine also powers the NodeJS runtime

JavaScript Intro

A variable is a named spot in code that stores a value you can use or change.

Declaring a variable:

```
let x = 100; // number

let name = "cci"; // string

let isThursday = true // boolean
```

JavaScript Intro

let vs const

```
let x = 100;
```

```
x = 200;
```

```
const x = 100;
```

```
x = 200;
```

JavaScript Intro

Functions

```
function sayHello() {
  alert("Hello!");
}
```

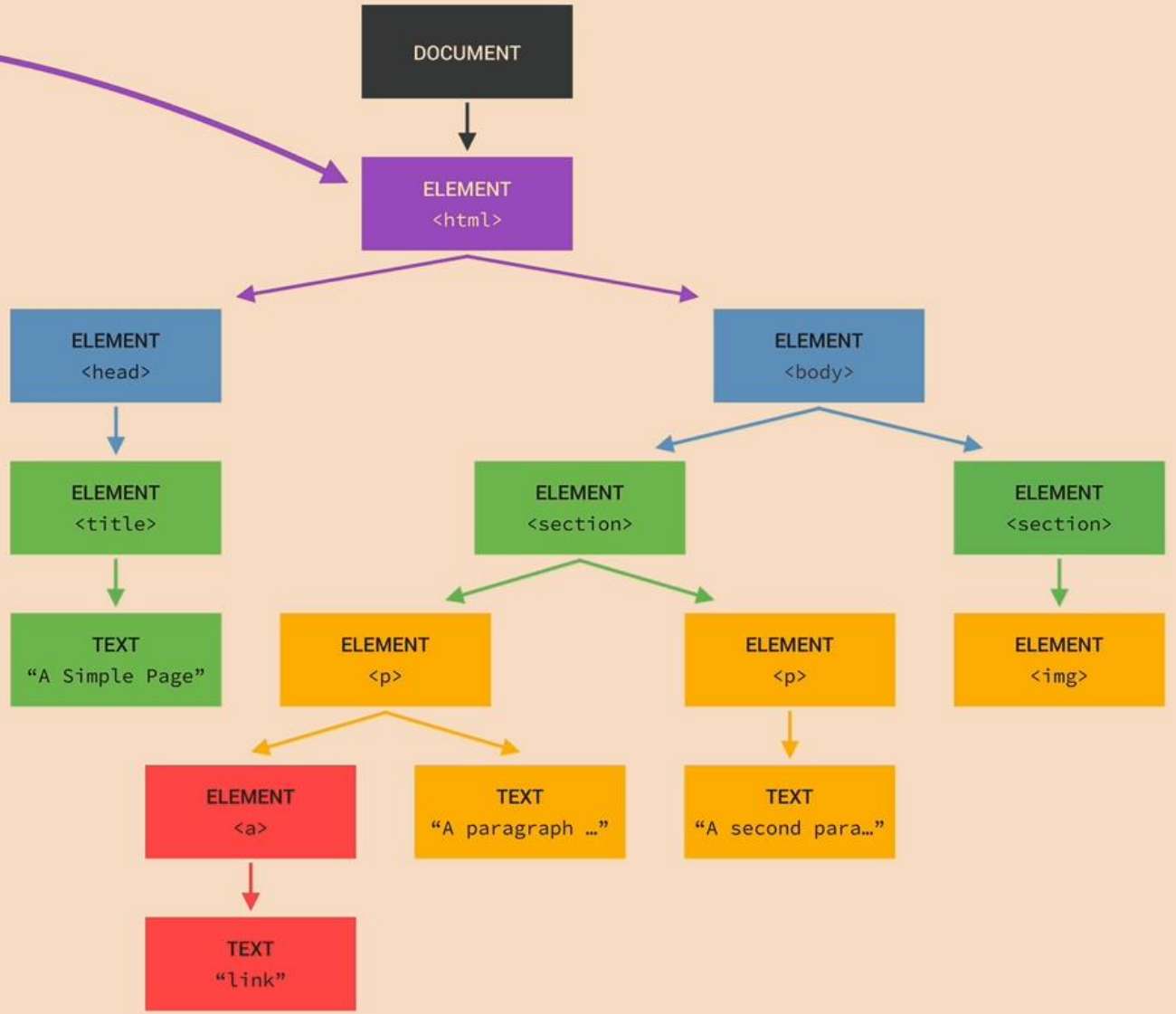
```
sayHello();
```

Running the function (function execution)

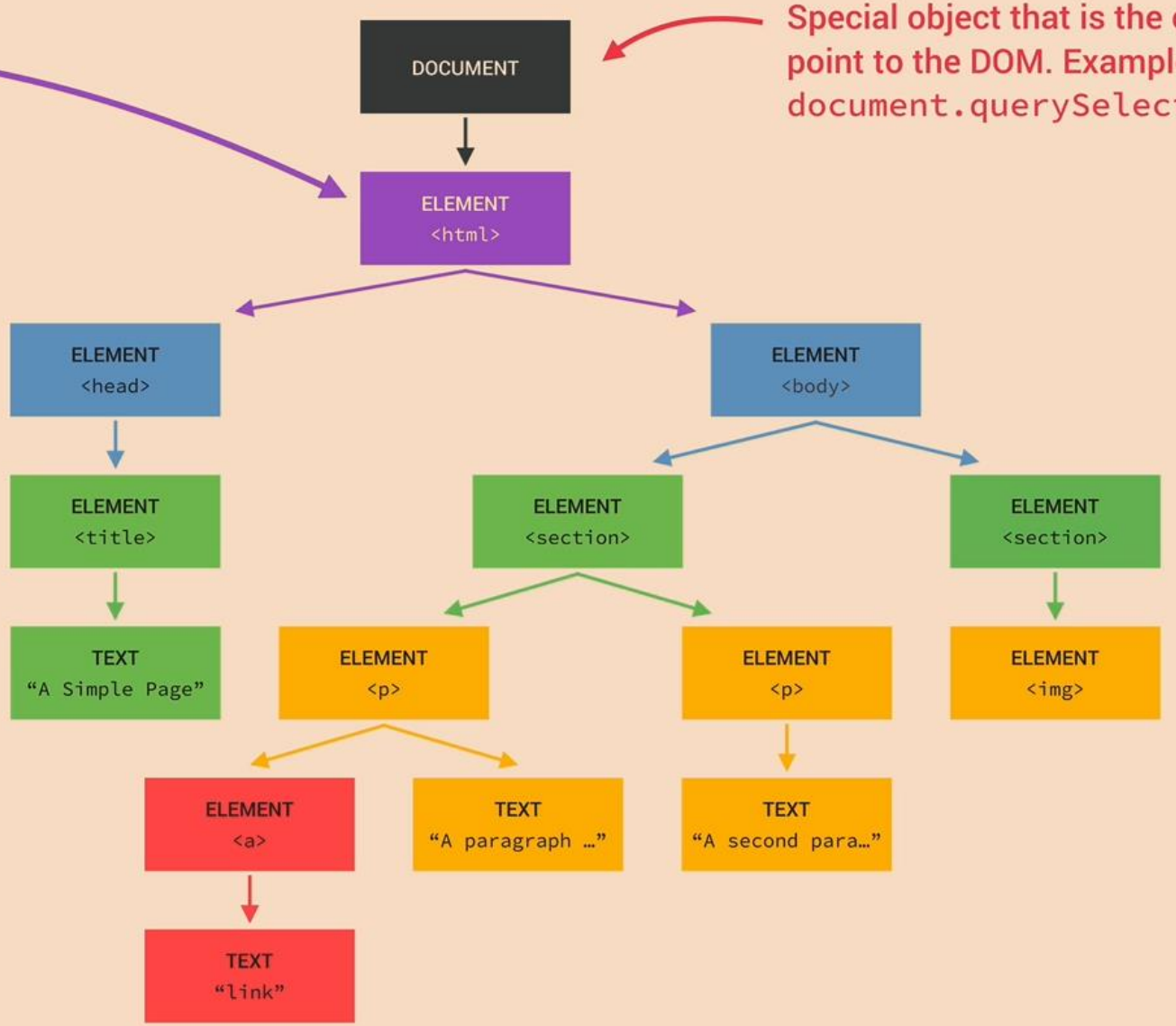
Document Object Model

- The **Document Object Model (DOM)** represents an HTML page as a structured tree of nodes.
- JavaScript uses the DOM to access and modify page elements, making web pages interactive.

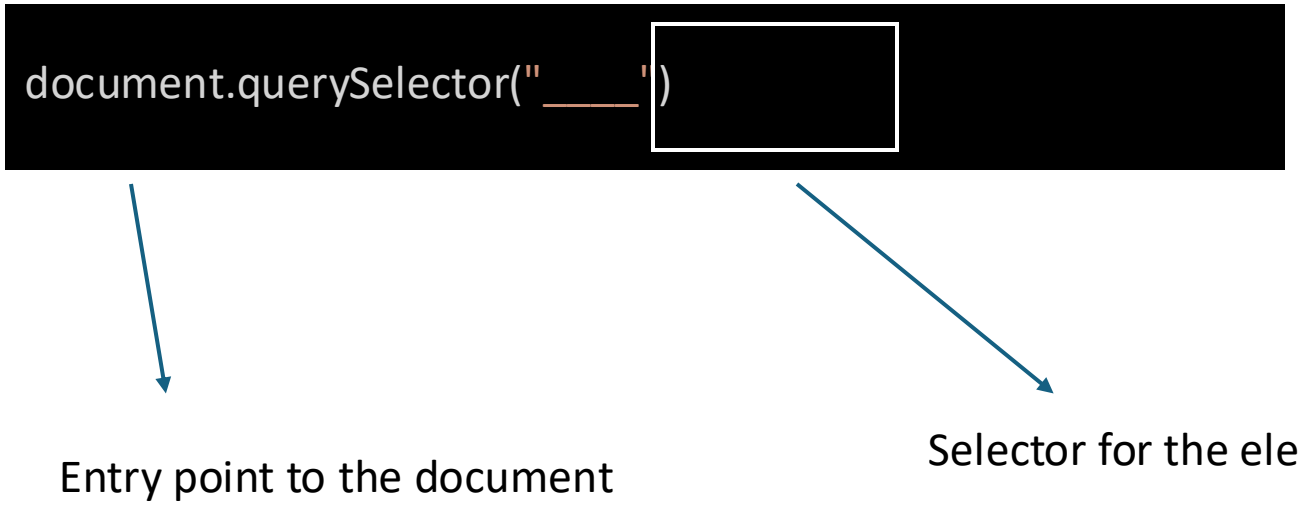
```
<html>
  <head>
    <title>A Simple Page</title>
  </head>
  <body>
    <section>
      <p>A paragraph with a <a>link</a></p>
      <p>A second paragraph</p>
    </section>
    <section>
      <img src="dom.png" alt="The DOM" />
    </section>
  </body>
</html>
```



```html
<html>
  <head>
    <title>A Simple Page</title>
  </head>
  <body>
    <section>
      <p>A paragraph with a <a>link</a></p>
      <p>A second paragraph</p>
    </section>
    <section>
      <img src="dom.png" alt="The DOM" />
    </section>
  </body>
</html>
```



Manipulating Documents



Manipulating Documents

querySelector method takes CSS-like selectors to access an element

for ID

. for classes

Manipulating Documents

querySelector method takes CSS-like selectors to access an element

for ID

. for classes

HTML

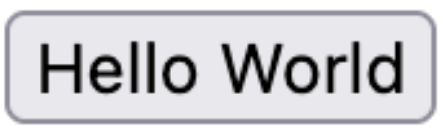
```
<button id="submit-btn">Clickclick</button>
```

JavaScript

```
document.querySelector("#submit-btn")
```

```
const myButton = document.querySelector("#submit-btn");

myButton.textContent = "Hello World";
```



Working with CSS Classes

```
myButton.classList.add('highlight')

myButton.classList.remove('highlight’)
```

Creating New Elements

```
const myNewButton = document.createElement("button")


myNewButton.textContent = "Hello again"


document.body.append(myNewButton)
```

Creating New Elements

```
const myNewButton = document.createElement("button")


myNewButton.textContent = "Hello again"


document.body.append(myNewButton)
```

Adds the element at the end of body

```
<div id="about-section"></div>
```

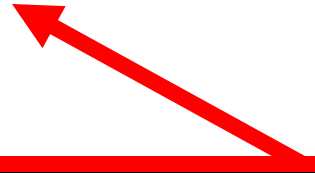
```
const sectionContainer = document.querySelector("#about-section")

const aboutHeading = document.createElement('h2')

aboutHeading.textContent = "About Me"

sectionContainer.append(aboutHeading)
```

```
<div id="about-section"></div>
```



```
const sectionContainer = document.querySelector("#about-section")

const aboutHeading = document.createElement('h2')

aboutHeading.textContent = "About Me"

sectionContainer.append(aboutHeading)
```

Adds the h2 tag at the end of

Walk-through