HTML, CSS, and JavaScript

Rainforest

- Overview
  - Frontend
  - Backend
  - DevOps
- HTML
- CSS
- JavaScript

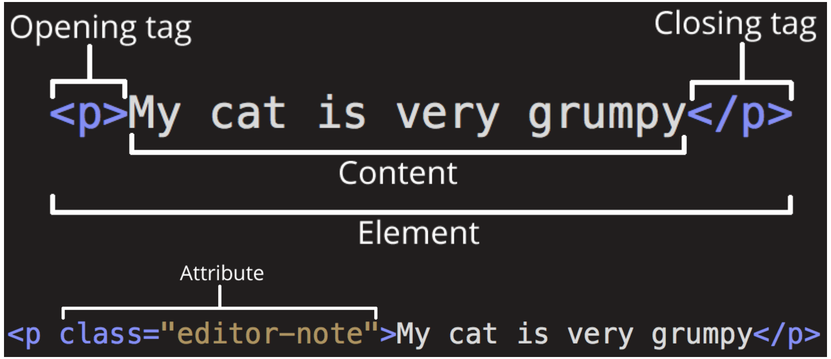
Overview

HTML

- Hypertext Markup Language,
- markup language
-
- nesting element

```
<ul id="todo__items" class="todo__items">...</ul>
```

- `` `<ul` ``: opening tag
- `` `</ul>` ``: closing tag
- `` `...` ``: content
- `` `id` ``, `` `class` ``, ...: attributes



```
<p>My cat is very grumpy</p>
<p class="editor-note">My cat is very grumpy</p>
```

HTML

```html
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My test page</title>
  </head>
  <body>
    <nav><img src="images/logo.png" alt="logo"></nav>
    <aside>
      <ul>
        <li>Item 01</li>
        <li>Item 02</li>
        ...
      </ul>
    </aside>
    <main>
      <p>
        This is demo text!
      </p>
    </main>
    <footer>Rainforest@2022</footer>
  </body>
</html>
```

- head
  -
    - CSS
    - JavaScript
  -
    - title:
    - favicon
    - description
    - ...
- body
  - nav
  - aside
  - main
  - footer
  - ...

HTML elements

`` `div` ``

- the generic container for flow content
- no effect on the content or layout until styled in some way using CSS

- heading: h1, h2, ..., h6
- paragraph: p
- link: a
- span
  - generic inline container for phrasing content

> Block-level elements v.s. Inline elements

- Unordered lists: `` `ul -> li` ``
- Ordered lists: `` `ol -> li` ``

Heading 1

Heading 2

Heading 3

Heading 4

Heading 5

HEADING 6

paragraph paragraph paragraph paragraph paragraph paragraph paragraph

Rainforest

paragraph paragraph paragraph paragraph span with red text paragraph paragraph

- Unordered List Item 1
- Unordered List Item 2
- Unordered List Item 3
- ...

1. Ordered List Item 1
2. Ordered List Item 2
3. Ordered List Item 3
4. ...

HTML elements(Cont.)

`img`

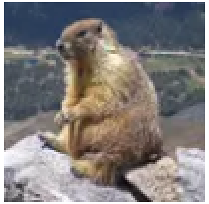
`table`

`button`

`form`

- `label`
- `input`
- `select`, `option`
- `textarea`

> Further Reading



Column 1	Column 2	...
Cell 1	Cell 2	...
...	...	...

Button

text input

number input

password input

Lebel for Select

Option 1

textarea

ARIA

- Accessible Rich Internet Applications
- a set of roles and attributes that define ways to make web content and web applications (especially those developed with JavaScript) more accessible to people with disabilities.

Further Reading

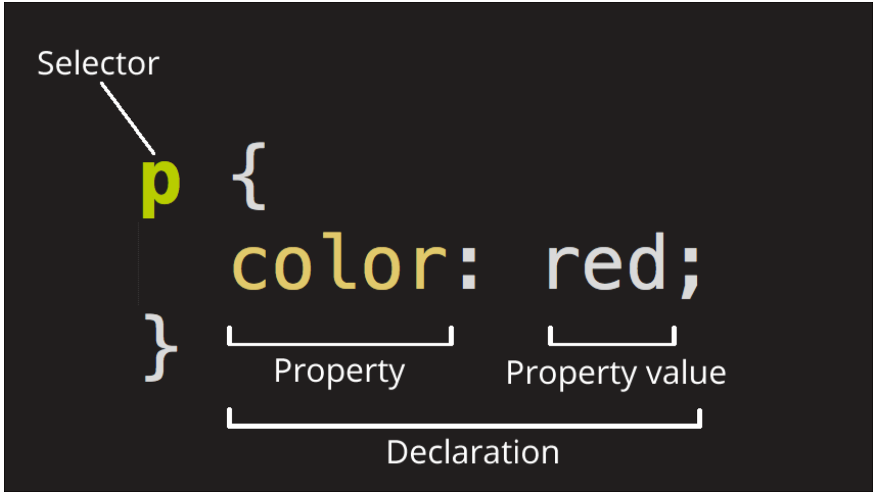
`` `data-*` ``

allow us to store extra information on standard, semantic HTML elements without other hacks such as non-standard attributes, or extra properties on DOM.

Further Reading

CSS

- Cascading Stylesheets,
- style sheet language
- Selector
- Declaration = (Properties) + (Property value)



Selector

```
p {
  color: red;
}
```

Property

Property value

Declaration

Selector

```
<img id='root' class='class1' src='/ha.png' />
```

-

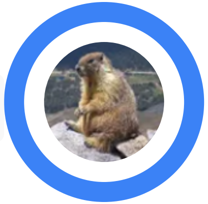
  ```
  img {
    width: 100px;
    height: 100px;
  }
  ```

- ID

  ```
  #root {
    border-radius: 50px;
  }
  ```

- Class

  ```
  .class1 {
    border: 10px solid blue;
  }
  ```



-

  ```
  [src] {
    padding: 10px;
  }
  ```

- class Pseudo-class selector

  ```
  img:hover {
    cursor: pointer;
  }
  ```

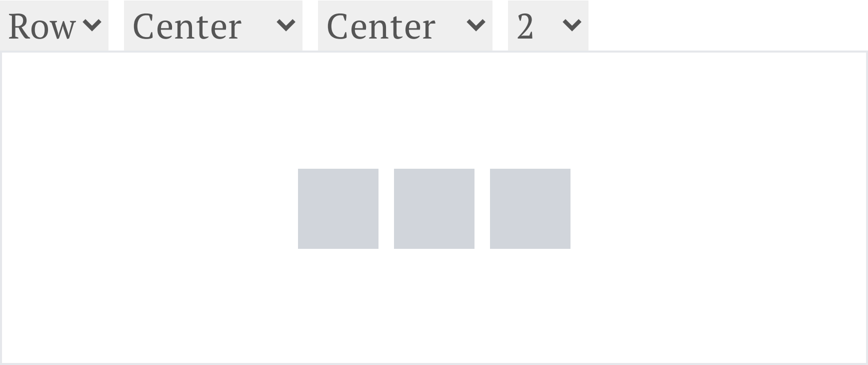
> Further Reading

Position

- relative
- absolute
  - removed from the normal document flow
  - no space is created for the element in the page layout
  - positioned relative to its **closest positioned ancestor**
- fixed
  - removed from the normal document flow
  - no space is created for the element in the page layout
  - positioned relative to **the initial containing block** established by the viewport
- sticky
  - offset relative to its nearest scrolling ancestor and containing block (nearest block-level ancestor)

Display

- block: generating line breaks both before and after the element when in the normal flow.
- inline-block
- flex
  - CSS Flexible Box Layout

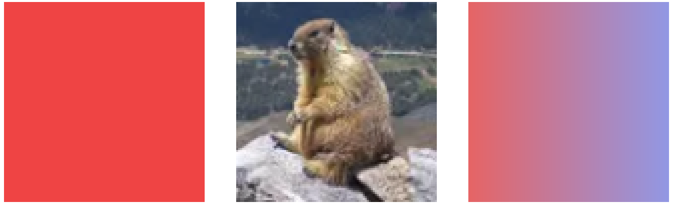


- grid
  - Grid Layout

Colors

- background

```
background: red;
background: url('/ha.png');
background-image: linear-gradient(#e66465, #9198e5);
```



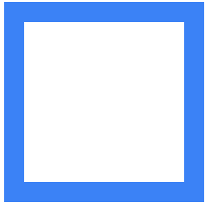
- text

```
color: red;
```

Green Text

- border

```
border: 10px solid blue;
```



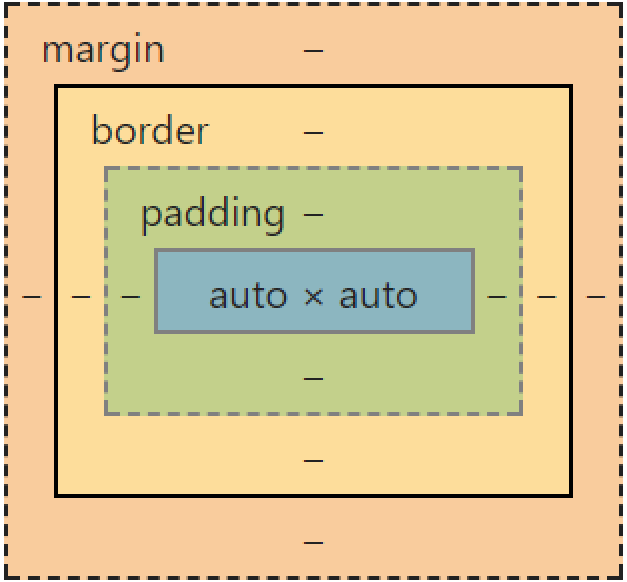
Text

- font-family
- font-size
- font-weight

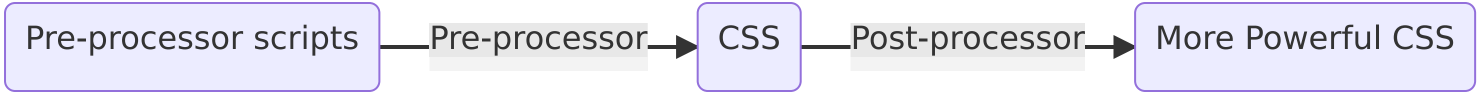
Sizing

- width
- height
- padding
- margin
- box-sizing
  - border-box
  - content-box

```
width: 100px;
height: 100px;
padding: 8px;
margin: 4px;
box-sizing: border-box | content-box;
```



Pre or Post-processor



- Pre-processor
  - Sass, stylus, less,
  - Take a language that compiles to CSS

```
$font-stack: Helvetica, sans-serif;
$primary-color: #333;

body {
  font: 100% $font-stack;
  color: $primary-color;
}
```

→

```
body {
  font: 100% Helvetica, sans-serif;
  color: #333;
}
```

- Post-processor
  -
  - Align and refurbish CSS to have the best possible outcome for today’s browsers.

> Deconfusing Pre- and Post-processing

JavaScript

- JavaScript is bananas

```
+ 'a'
// NaN, Not a number
'b' + 'a'
// 'ba'
'b' + 'a' + + 'a' + 'a'
// 'baNaNa'
('b' + 'a' + + 'a' + 'a').toLowerCase()
// 'banana'
```

- == v.s. ===
  - ==: operator attempts to convert them to the same type before comparing.
  - ===: strict equality operators

- Shallow Copy v.s. Deep Copy
  - shallow copy is copy by reference
  - deep copy

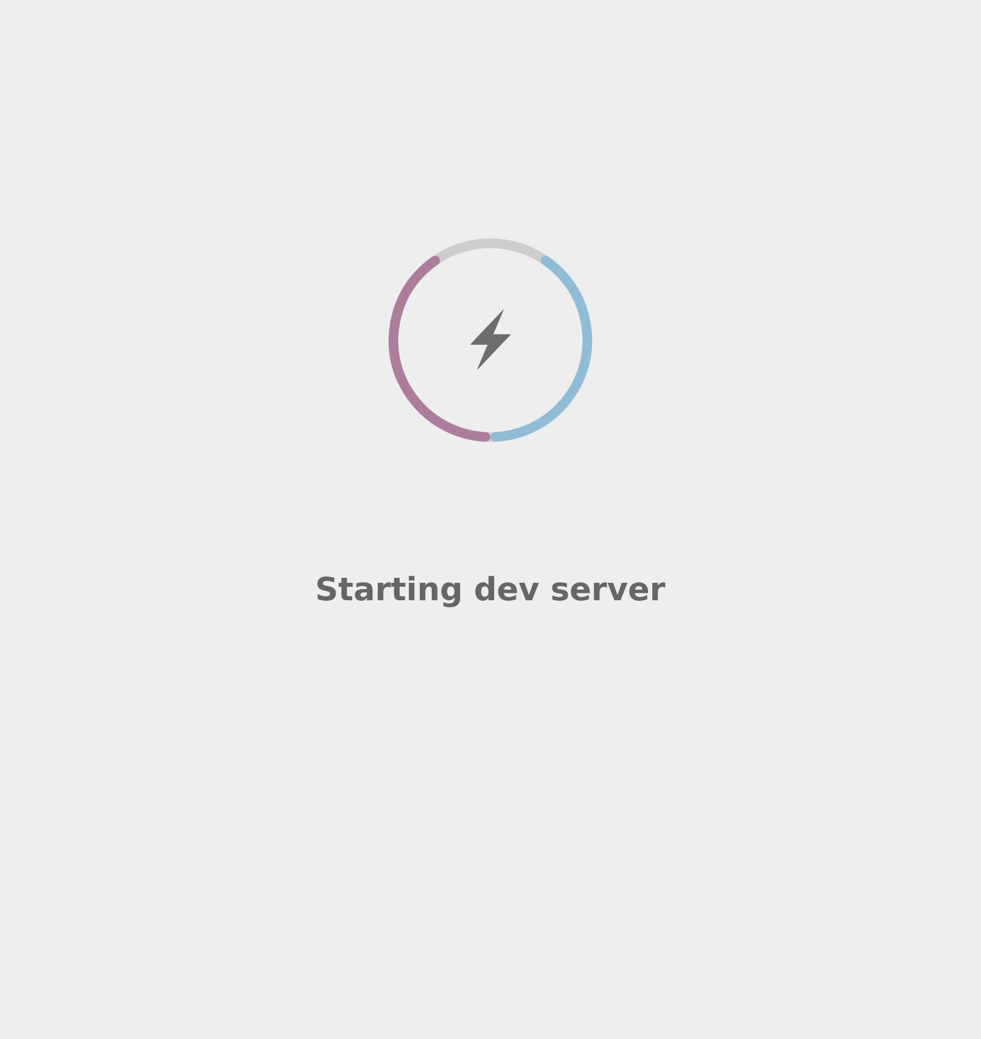
```
const data = { a: 'test' }
const b = Object.assign({}, data)
const c = {...a}
```

- JS HTML DOM
- JS Web APIs
- TypeScript: strongly typed programming language that builds on JavaScript

- Wappalyzer
- Inspect Element
  - Elements
  - Console
  - Storage
  - (Sources)

- Visual Studio Code + Web Server
  - Live Server
- Codepen
- Stackblitz

Demo



Starting dev server