DWARVES FOUNDATION

Basic HTML + CSS + JavaScript

Khac Vy
@trankhac_vy
vincenzo@d.foundation

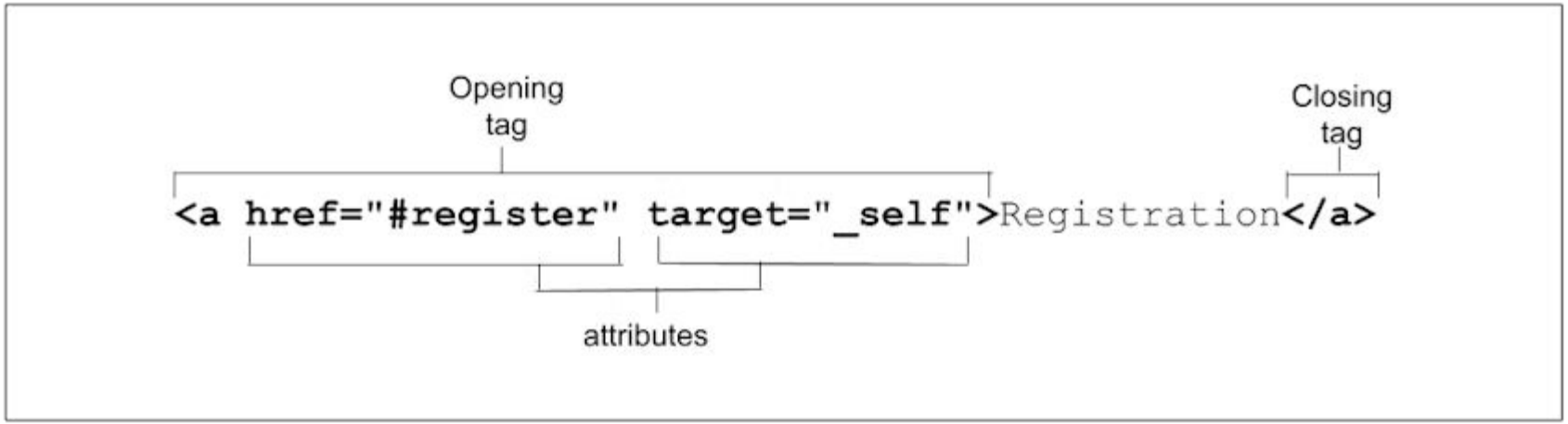
Agenda

1. Semantic HTML
2. CSS Fundamentals
3. Introduction to JavaScript
4. Combining HTML, CSS, and JavaScript
5. Q&A

Introduction to HTML

- HTML stands for Hypertext Markup language.
- A markup language is a set of markup tags.
- HTML is **not** case sensitive language.
- HTML documents are described by HTML tags.
- Opening tag can carry attributes.

HTML Elements



HTML Document Structure

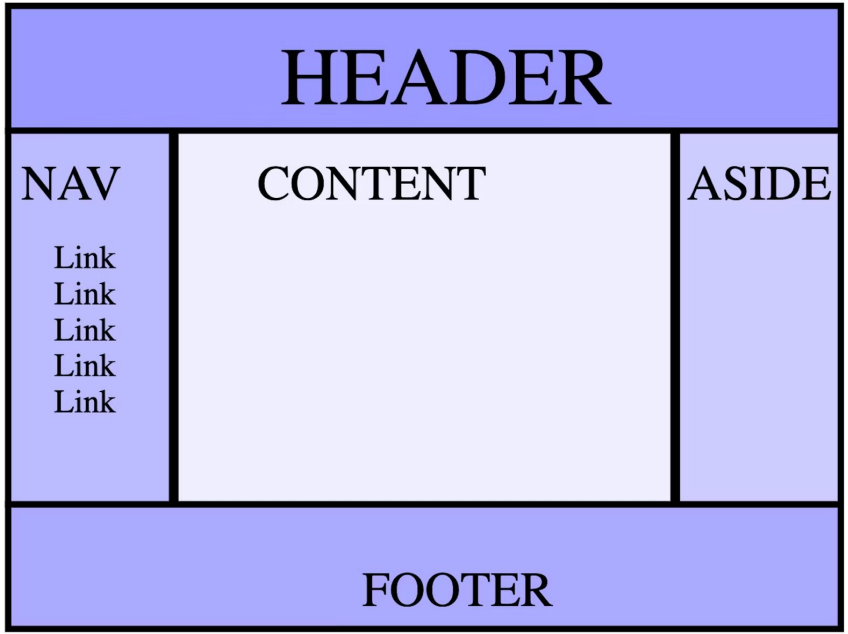
```html
<!DOCTYPE html>
<html lang="en-US">
  <head>
  </head>
  <body>
  </body>
</html>
```

Semantic HTML

- A semantic element clearly describes its meaning to both the browser and the developer.
- Writing semantic HTML means using HTML elements to structure your content based on each element's meaning, not its appearance.
- Examples of non-semantic elements: and - Tells nothing about its content.
- Examples of semantic elements: <form>, <table>, and <article> - Clearly defines its content.

Page layout

```
<body>
  <header>Header</header>
  <nav>Nav</nav>
  <main>
    <article>First post</article>
    <article>Second post</article>
  </main>
  <aside>Aside</aside>
  <footer>Footer</footer>
</body>
```



Headings: <h1>-<h6>

- Do not use heading elements to resize text.
- Do not skip heading levels: always start from <h1>, followed by <h2> and so on.
- Avoid using multiple <h1> elements on one page

```
<h1>Heading level 1</h1>

<h2>Heading level 2</h2>

<h3>Heading level 3</h3>

<h4>Heading level 4</h4>

<h5>Heading level 5</h5>

<h6>Heading level 6</h6>
```

DWARVES FOUNDATION

Text

- paragraph
- <blockquote>blockquote</blockquote>
- <q>q</q>
- <cite>blockquote</cite>

Link

```html
<a href="https://dwarves.foundation/">Dwarves foundation/</a>

<a href="https://dwarves.foundation/" target="_blank">Dwarves foundation/</a>

<a href="#about">About</a>

<a href="mailto:vincenzo@d.foundation">Email Vincenzo</a>

<a href="tel:0123456789">Call Vy</a>
```

List

```
// Unordered lists
<ul>
  <li>Blender</li>
  <li>Toaster</li>
  <li>Vacuum</li>
</ul>

// Ordered lists
<ol>
  <li>Blender</li>
  <li>Toaster</li>
  <li>Vacuum</li>
</ol>
```

- Blender
- Toaster
- Vacuum

1. Blender
2. Toaster
3. Vacuum

Navigation

```html
<nav aria-label="On this page">
  <div>On this page</div>
  <div>
    <ul>
      <li>
        <a href="#skip">Skip to content link</a>
      </li>
      <li>
        <a href="#toc">Table of contents</a>
      </li>
      <li>
        <a href="#bc">Page breadcrumbs</a>
      </li>
      <li>
        <a href="#ln">Local navigation</a>
      </li>
      <li>
        <a href="#global">Global navigation</a>
      </li>
    </ul>
  </div>
</nav>
```

On this page

- Skip to content link
- Table of contents
- Page breadcrumbs
- Local navigation
- Global navigation

Table

```html
<table>
  <caption>MLW Alumni</caption>
  <thead>
    <tr>
      <th>Name</th>
      <th>Destiny</th>
      <th>Year</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <th>Hal Gibrah</th>
      <td>Calculator</td>
      <td>2020</td>
    </tr>
    <tr>
      <th>Cathy Terr</th>
      <td>Waste disposal</td>
      <td>2018</td>
    </tr>
    <tr>
      <th>Lou Minious</th>
      <td>Lightbulb</td>
      <td>1956</td>
    </tr>
  </tbody>
</table>
```

MLW Alumni

Name	Destiny	Year
Hal Gibrah	Calculator	2020
Cathy Terr	Waste disposal	2018
Lou Minious	Lightbulb	1956

DWARVES FOUNDATION

Form

```html
<form method="GET">
  <label for="student">Pick a student:</label>
  <select name="student" id="student">
    <option value="hoover">Hoover Sukhdeep</option>
    <option>Blendan Smooth</option>
    <option value="toasty">Toasty McToastface</option>
  </select>
  <input type="submit" value="Submit Form">
</form>
```

Pick a student: Hoover Sukhdeep Submit Form

Image

```
<img src="images/bird.png" alt="Bird">
```

DWARVES FOUNDATION

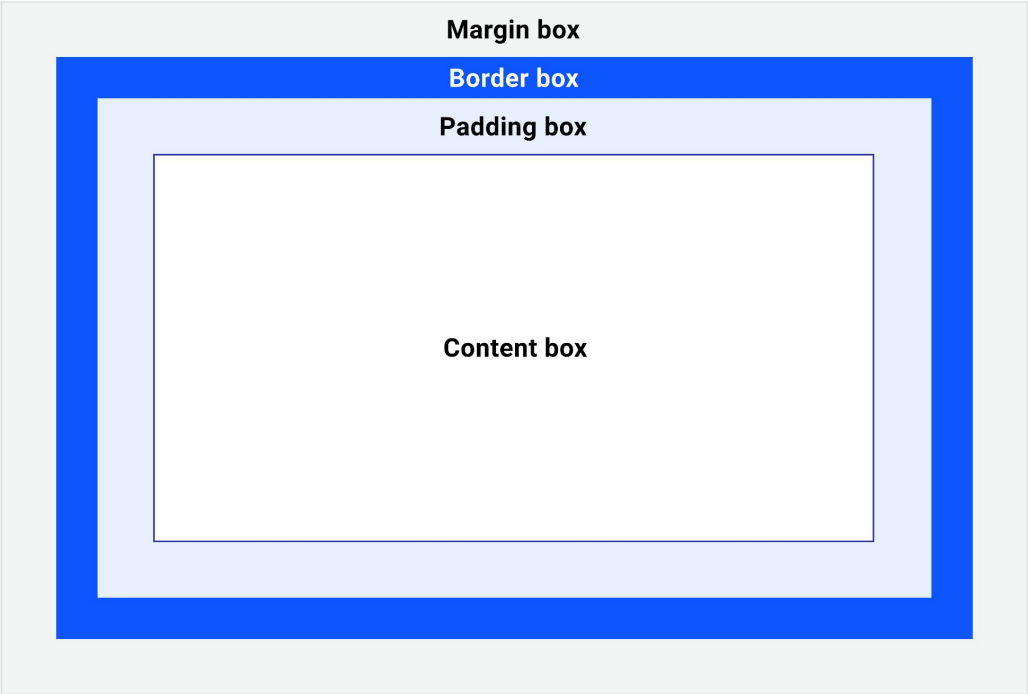
CSS Fundamentals

- CSS stands for Cascading Style Sheets.
- Markup language used in the web document for presentation purpose.
- HTML and CSS work together to produce beautiful and functional Web sites
- HTML -> structure
- CSS -> style

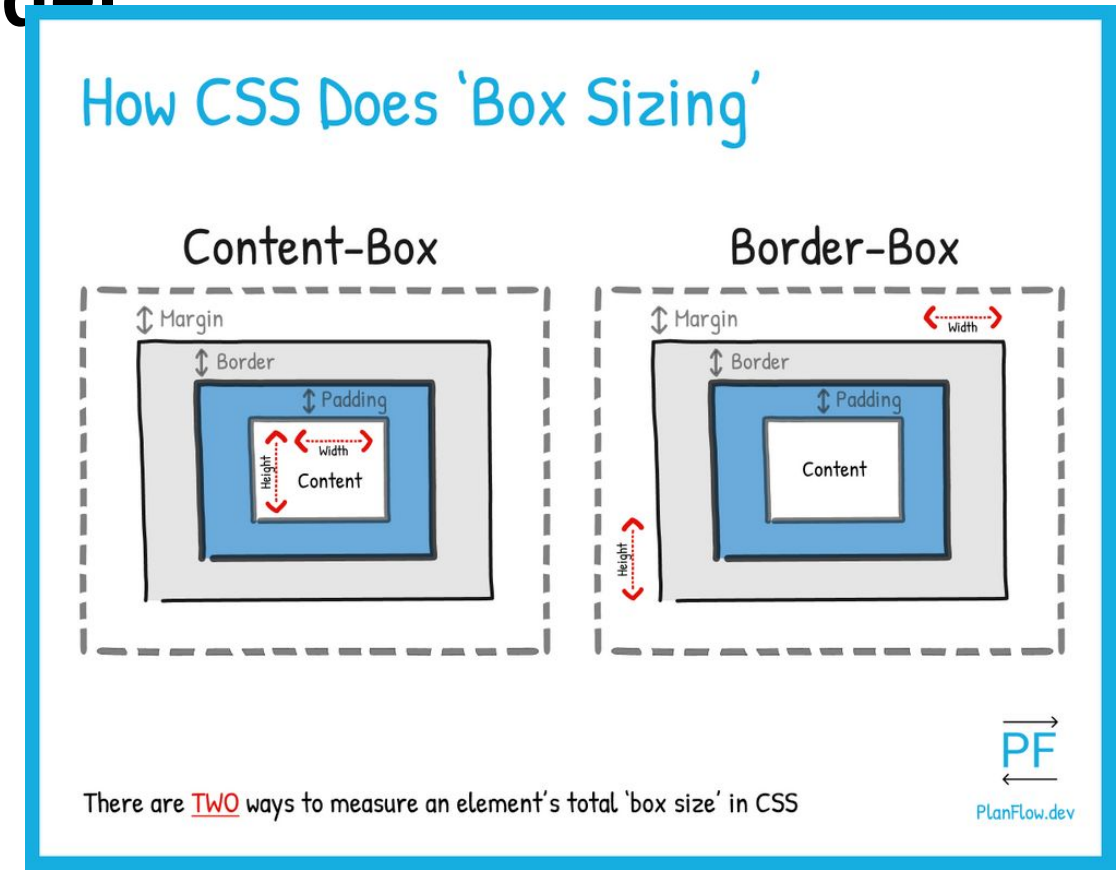
Adding CSS to document

```
// External
<head>
  <link rel="stylesheet" href="styles.css" />
</head>

// internal
<style>
  p {
    color: gray;
    font-size: 14px;
  }
</style>

// inline style sheet
<p style="color: gray; font-size: 14px;">Whereas disregard and contempt for
human rights have resulted</p>
```

Box model



Box model



DWARVES FOUNDATION

Box model

```
*,
*::before,
*::after {
  box-sizing: border-box; // content-box
}
```

Inline vs block

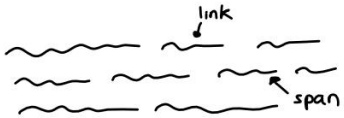
JULIA EVANS
@b0rk

inline vs block

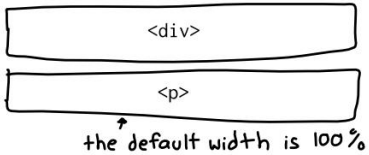
HTML elements default to inline or block

example block elements	example inline elements
<p>	<a>
	<i>
<h1> - <h6>	<button> <input>
<table> <form>	<small> <abbr>
<article> <nav>	<textarea>

inline elements are laid out horizontally



block elements are laid out vertically by default



inline elements ignore width & height

Setting the width is impossible, but you can use line-height to change the height

also, inline elements ignore the vertical padding of other inline elements

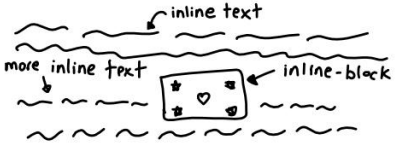
display can force an element to be inline or block

display determines 2 things:

1. whether the element itself is inline, block, inline-block, etc
2. how child elements are laid out (grid, flex, table, default, etc)

display: inline-block;

inline-block makes a block element that's laid out horizontally like an inline element



DWARVES FOUNDATION

Selectors

```
// Universal selector
* {
  color: red;
}


// Type selector
p {
  font-size: 14px;
}
```

```
// class selector
.container {
  max-width: 1200px;
}

// ID selector
#header {
  max-width: 1200px;
}
```

```
// Attribute selector
[data-type='primary'] {
  color: red;
}

// Group selector
strong,
em,
.my-class,
[lang] {
  color: red;
}
```

DWARVES FOUNDATION

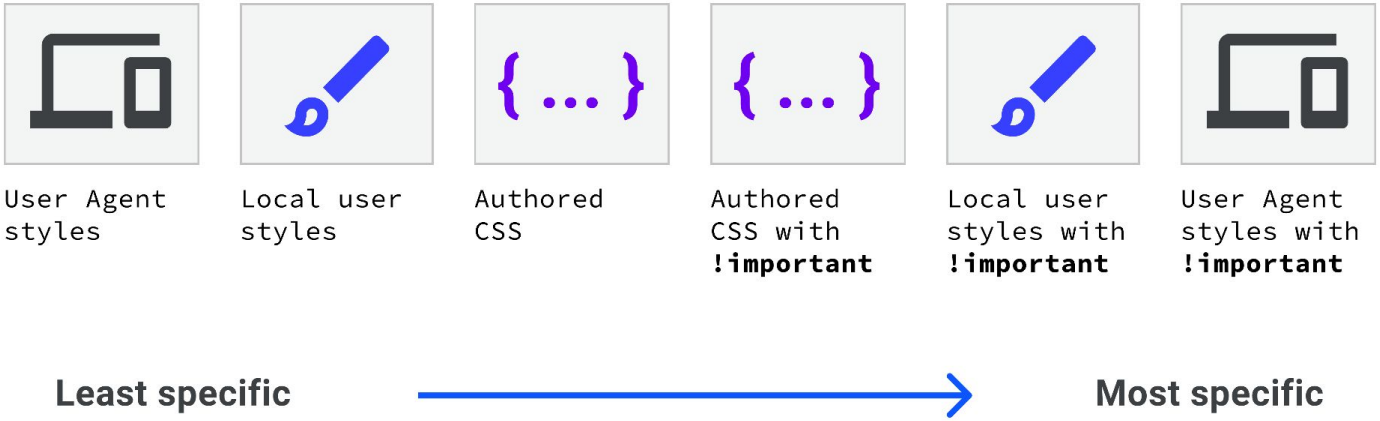
Selectors

```
// Pseudo-classes
a:hover {
  outline: 1px dotted green;
}

// Pseudo-element
.my-element::before {
  content: 'Prefix - ';
}
```

Selectors

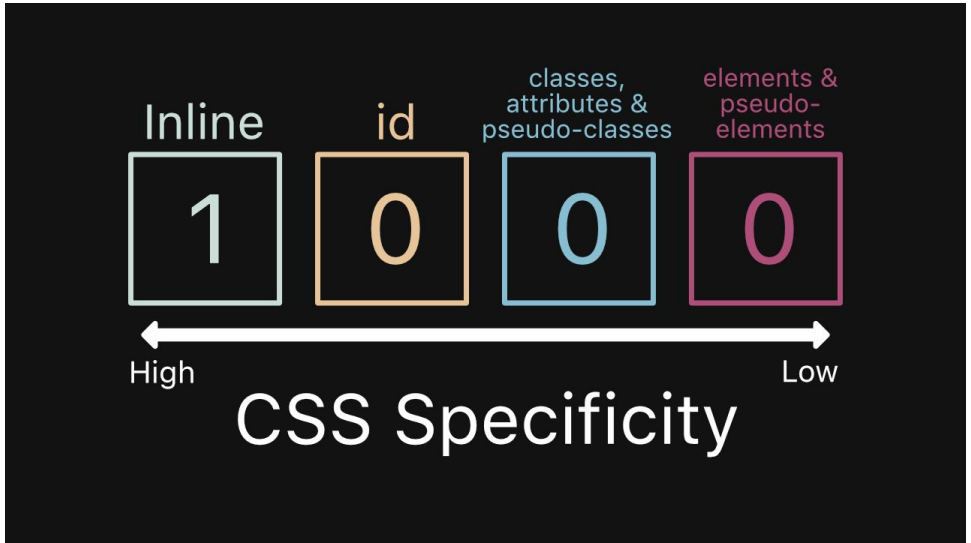
```
// Adjacent sibling combinator
/* Paragraphs that come immediately after any image */
img + p {
  font-weight: bold;
}

// Child combinator
/* List items that are children of the "my-things" list */
ul.my-things > li {
  margin: 2em;
}

// Descendant combinator
/* List items that are descendants of the "my-things" list */
ul.my-things li {
  margin: 2em;
}

// General sibling combinator
/* Paragraphs that are siblings of and  subsequent to any image */
img ~ p {
  color: red;
}
```

The cascade


User Agent styles
Local user styles
Authored CSS
Authored CSS with !important
Local user styles with !important
User Agent styles with !important
Least specific
Most specific

Specificity

- Specificity is an algorithm which determines which CSS selector is the most specific, using a weighting or scoring system to make those calculations.



DWARVES FOUNDATION

Sizing units

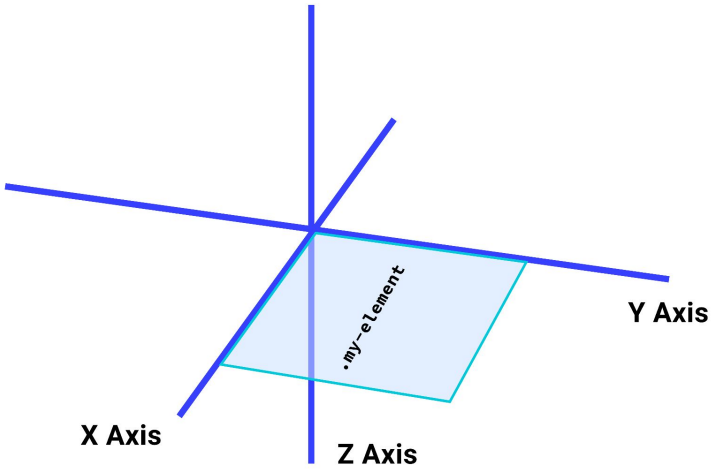
- Numbers
- Percentages
  - If you set margin or padding as a percentage, they will be a portion of the parent element's width, regardless of direction.
- Dimensions and lengths
  - Absolute lengths
  - Relative lengths

Layout

- Flexbox is a layout mechanism for one-dimensional layouts.
- Grid is designed to control multi-axis layouts.
- Flow layout
  - Inline block
  - Floats
  - Multicolumn layout
  - Positioning: `static`, `relative`, `absolute`, `fixed` and `sticky`

z-index

- The z-index property explicitly sets a layer order for HTML based on the 3D space of the browser—the Z axis.



Responsive Design

- Responsive web design (RWD) is a web design approach to make web pages render well on all screen sizes and resolutions while ensuring good usability.
- Designing for mobile first is known as mobile first design.
- Media queries allow you to apply CSS styles depending on a device's general type (such as print vs. screen) or other characteristics such as screen resolution or browser viewport width.

Introduction to JavaScript

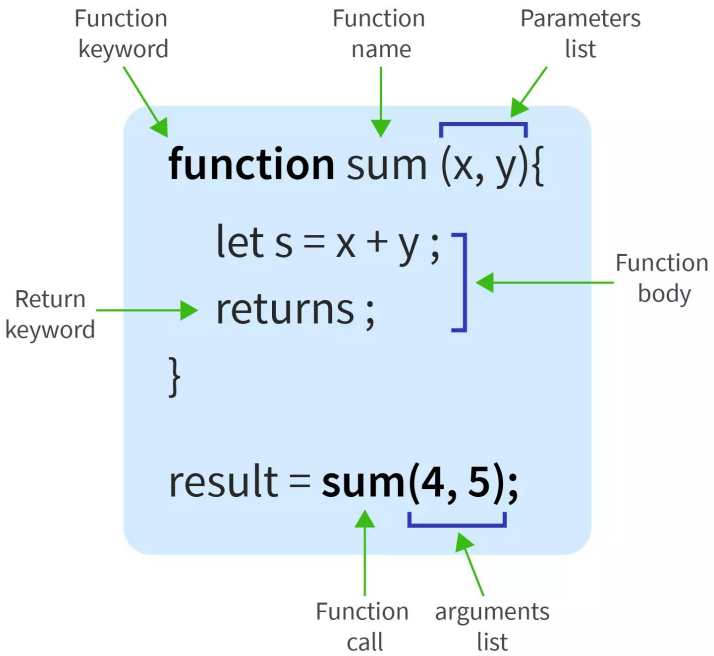
- Javascript is a prototype-based scripting language with dynamic typing and first-class function.
- Javascript is a multi-paradigm language, supporting object-oriented, imperative and functional programming styles.
- Originally designed for the browser but now used literally everywhere.

Data types

- Primitive types: Number, String, Bigint, Boolean, Symbol, Null and Undefined.
- Objects
  - In JavaScript, objects can be seen as a collection of properties.
  - Date.
  - Indexed collections: Arrays and typed Arrays.
  - Keyed collections: Maps, Sets, WeakMaps, WeakSets

Functions

- A function definition (also called a function declaration, or function statement) consists of the function keyword, followed by:
    - The name of the function.
    - A list of parameters to the function, enclosed in parentheses and separated by commas.
    - The JavaScript statements that define the function, enclosed in curly brackets, { /* … */ }.



Function expressions

```
const square = function (number) {
  return number * number;
};

console.log(square(4)); // 16
```

Function hoisting

```
console.log(square(5)); // 25

function square(n) {
  return n * n;
}
```

Function scope

- Variables declared inside a function are limited to that function's scope.
- Functions can access variables from their own scope and any outer scopes, such as parent functions or the global scope.

```
// A nested function example

const name = "Chamakh";

function getScore() {
  const num1 = 2;
  const num2 = 3;

  function add() {
    return `${name} scored ${num1 + num2}`;
  }

  return add();
}

console.log(getScore()); // "Chamakh scored 5"
```

Closure

- A closure is the combination of a function bundled together (enclosed) with references to its surrounding state (the lexical environment).
- A closure gives you access to an outer function's scope from an inner function.
- Closures are created every time a function is created, at function creation time.

Closure

```
// The outer function defines a variable called "name"
const pet = function (name) {

  const getName = function () {
    // The inner function has access to the "name" variable of the
outer function
    return name;
  };

  return getName; // Return the inner function, thereby exposing it to
outer scopes
};
const myPet = pet("Vivie");

console.log(myPet()); // "Vivie"
```

Arrow functions

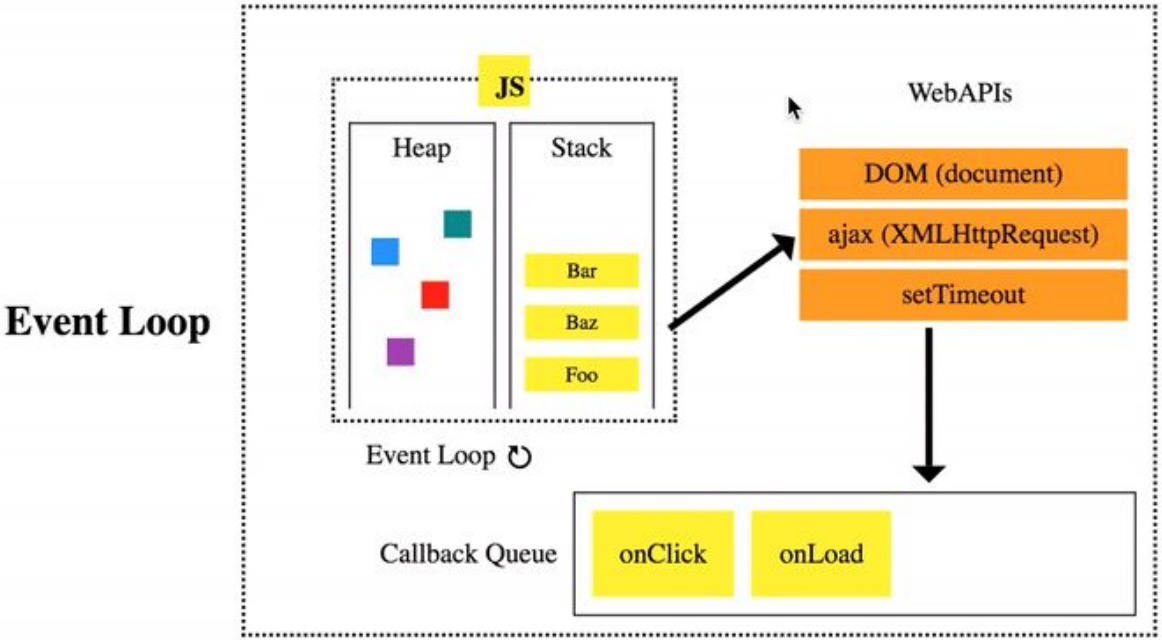
- An arrow function expression is a compact alternative to a traditional function expression, with some semantic differences and deliberate limitations in usage:
    - Don't have their own bindings to this, arguments, or super, and should not be used as methods.
    - Arrow functions cannot be used as constructors. Calling them with new throws a TypeError. They also don't have access to the new.target keyword.
    - Arrow functions cannot use yield within their body and cannot be created as generator functions.

Arrow functions

```
() => expression

param => expression

(param) => expression

(param1, paramN) => expression

() => {
  statements
}

param => {
  statements
}

(param1, paramN) => {
  statements
}
```

DWARVES FOUNDATION

The event loop



DOM

- The Document Object Model (DOM) is the data representation of the objects that comprise the structure and content of a document on the web.
- The DOM represents the document as nodes and objects; that way, programming languages can interact with the page.
- The DOM is not part of the JavaScript language, but is instead a Web API used to build websites.

DOM

```
let allImages = document.querySelectorAll('img');

allImages.forEach((imageInstance) => {
  console.log(imageInstance.alt);
});

// ....

const p = document.getElementById("pid");

p.style.color = "blue";
p.style.fontSize = "18pt";
```

Combining HTML, CSS, and JavaScript

Demo

References

- https://web.dev/learn/html/
- https://web.dev/learn/css/
- https://developer.mozilla.org/en-US/docs/Web/JavaScript
- https://specificity.keegan.st/
- https://twitter.com/b0rk/status/1284528767151611904
- https://planflow.dev/blog/what-is-box-sizing-in-css-how-does-it-work

DWARVES FOUNDATION

Thank You

DWARVES FOUNDATION

Q&A