



# JavaScript & Introduction to the DOM

Web Development  
Lesson 2.3



# JavaScript Objects



# Instructor Demonstration

## Good Arrays



# Instructor Demonstration

Joan of Arc (Bad Arrays)

# Associated Data ==/= Arrays

---

Relating two separate arrays is not fun.

```
var joanOfArcInfoParts = ["Real Name", "Grew Up Where", "Known For", "Scars", "Symbolism"];

var joanOfArcInfoValues = ["Jehanne la Pucelle.", "Domremy, a village in northeastern France.",
    "Peasant girl, daughter of a farmer, who rose to become Commander of the French army.",
    "Took an arrow to the shoulder and a crossbow bolt to the thigh while trying to liberate Paris.",
    "Stands for French unity and nationalism."];
```



# Instructor Demonstration

## Gandalf the Grey Objects

# Gandalf: The Object

---

Gandalf's **properties** and **values** are associated in object form, making it easy to recall specific data.

```
11  var gandalf = {
12      "real name": "Gandalf",
13      "age (est)": 11000,
14      "race": "Maia",
15      "haveRetirementPlan": true,
16      "aliases": [
17          "Greyhame",
18          "Stormcrow",
19          "Mithrandir",
20          "Gandalf the Grey",
21          "Gandalf the White"
22      ]
23  }
24
25  // Object properties can be accessed with "bracket notation"
26  alert("My name is " + gandalf["real name"]);
27
28  // Or with "dot notation" if the property has no spaces
29  if (gandalf.haveRetirementPlan) {
30
31      // Or with a variable that matches the name of the property
32      var ageProperty = "age (est)";
33      var years = gandalf[ageProperty];
34      alert("My 401k has been gathering interest for " + years + " years!");
35  }
```

# Objects Visualized

This is Gandalf. According to code, Gandalf is an **object**.

var gandalf	=	{
-------------	---	---



"real name"	:	"Gandalf"	,
-------------	---	-----------	---

"age (est)"	:	11000	,
-------------	---	-------	---

"race"	:	"Maia"
--------	---	--------

}
---



# Objects Visualized

These are Gandalf's **properties** (like descriptors).

var gandalf	=	{
-------------	---	---



"real name"	:	"Gandalf"	,
-------------	---	-----------	---

"age (est)"	:	11000	,
-------------	---	-------	---

"race"	:	"Maia"
--------	---	--------

}
---

# Objects Visualized

These are the **values** of Gandalf's properties.

var gandalf	=	{
-------------	---	---



"real name"	:	"Gandalf"	,
-------------	---	-----------	---

"age (est)"	:	11000	,
-------------	---	-------	---

"race"	:	"Maia"
--------	---	--------

}
---

# Objects Visualized

Thus: `gandalf["race"] = "Maia"`

`var gandalf`

`=`

`{`



`"real name"`

`:`

`"Gandalf"`

`,`

`"age (est)"`

`:`

`11000`

`,`

`"race"`

`:`

`"Maia"`

`}`



# Instructor Demonstration

## Gandalf: The Grey Objects (Repeat)



## **Group Activity** (2 people): Basic Objects

**Suggested Time:**  
10 minutes



# Group Activity: Basic Objects

---



With a partner, spend the next few moments studying the code just slacked to you.



Then, write code below each comment to log the relevant information about the provided `car` object.



**Bonus:** If you finish early, create a brand new object of your own. Slack out a snippet of the code to the class when you are done. Be Creative!

Suggested Time: 10 minutes





# Instructor Demonstration

Run That Car!



## **Challenge:** Run That Car!

**Suggested Time:**  
15 minutes





# Challenge: Run That Car!

Using the code from the previous activity as a starting point, create a complete application such that:



Users can enter keyboard input (letters).



Each of the car's methods are assigned to a key.



When the user presses a key, it calls the appropriate function.

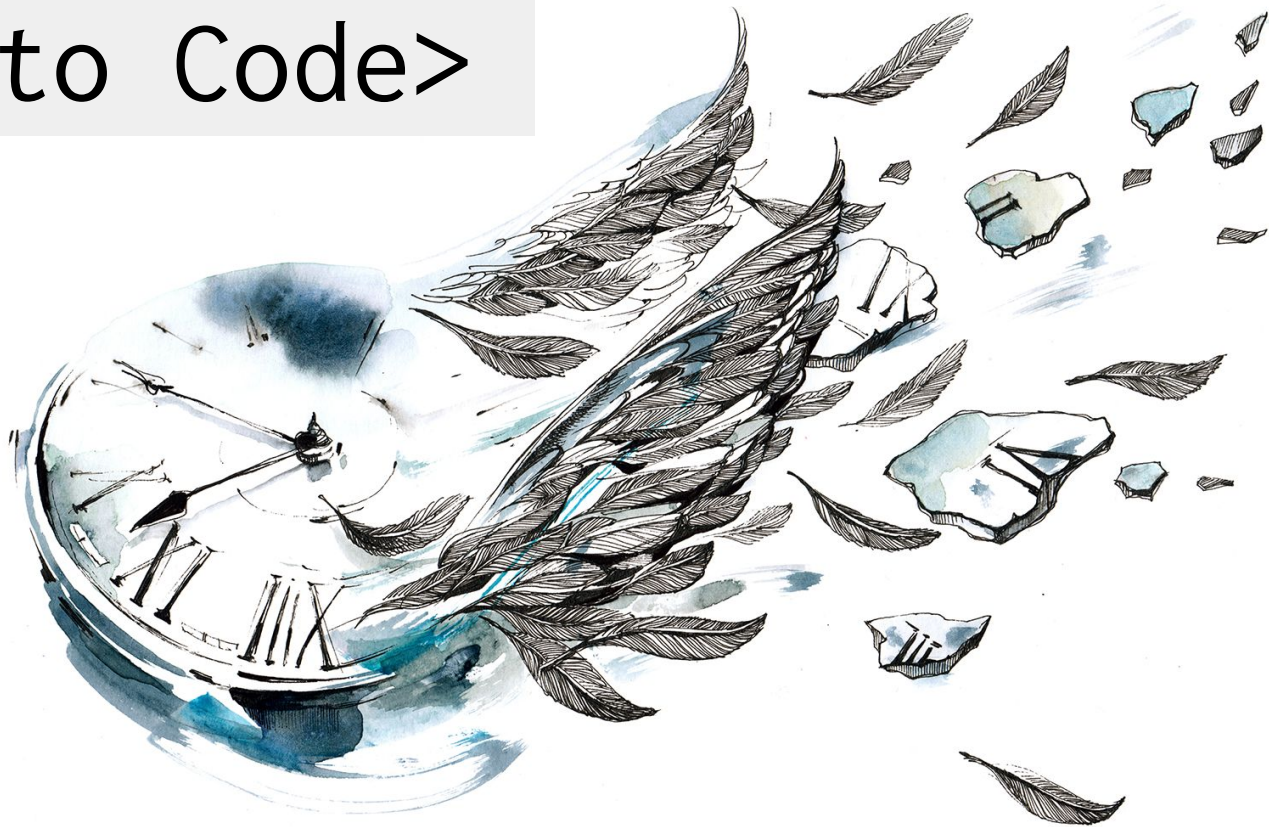


These letters also trigger a global function called `rewriteStats()` that logs the car's make, model, color, mileage, and `isWorking` status to the console.

Suggested Time: 15 minutes



# <Time to Code>

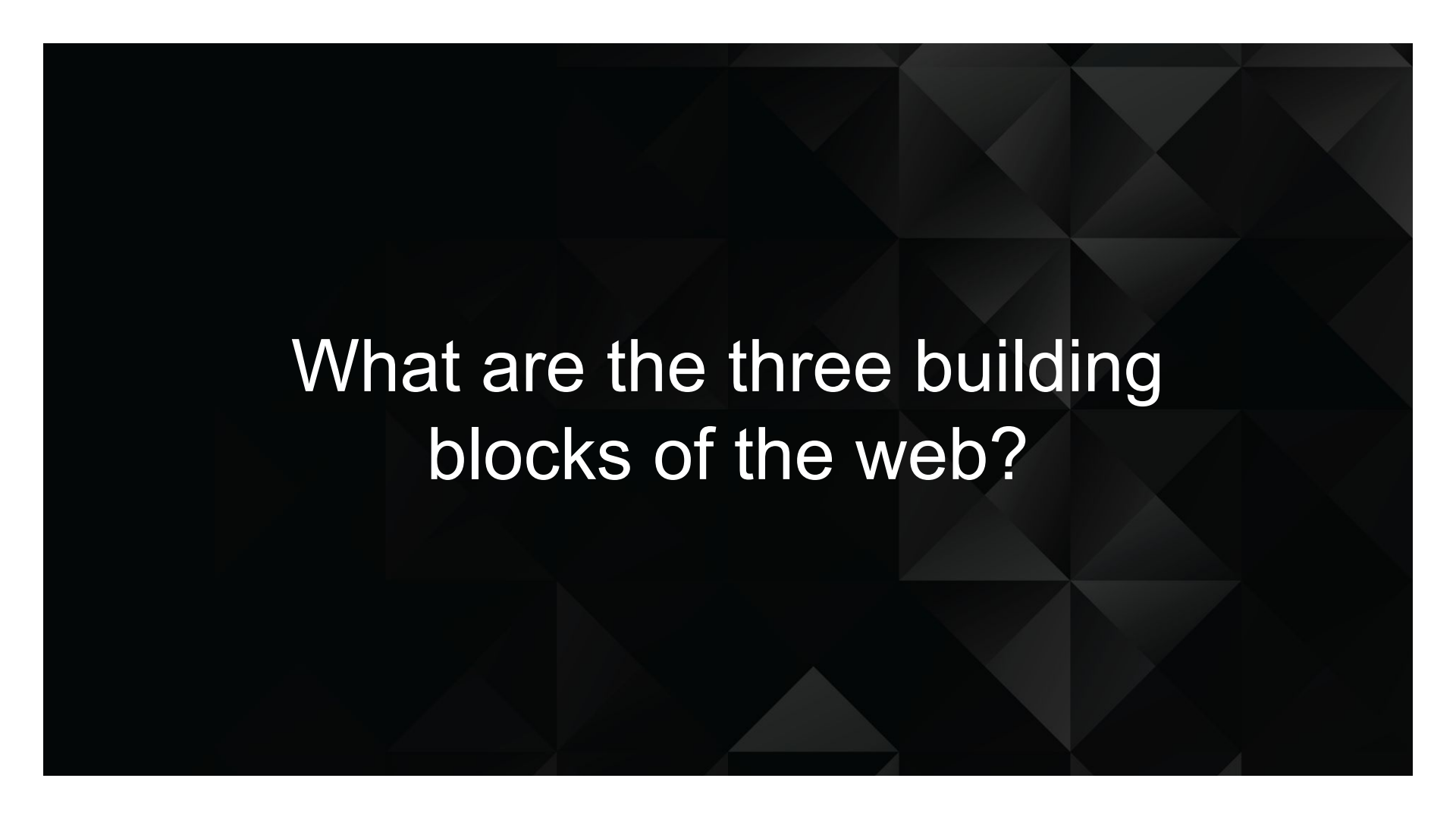


# Take a Break!

---



# Introduction to the DOM




What are the three building  
blocks of the web?

# Building Blocks of the Web

---

HTML	CSS	JavaScript
<p data-bbox="131 354 471 385">Used to write content.</p> <div data-bbox="249 579 469 892"><p data-bbox="280 584 438 623"><b>HTML</b></p></div>	<p data-bbox="710 354 1074 385">Used to format content.</p> <div data-bbox="853 579 1072 887"><p data-bbox="904 584 1023 623"><b>CSS</b></p></div>	<p data-bbox="1290 354 1789 554">Used to create dynamic web applications that take in user inputs, change what's displayed to users, animate elements, and much more.</p> <div data-bbox="1425 584 1644 892"><p data-bbox="1497 584 1574 623"><b>JS</b></p></div>

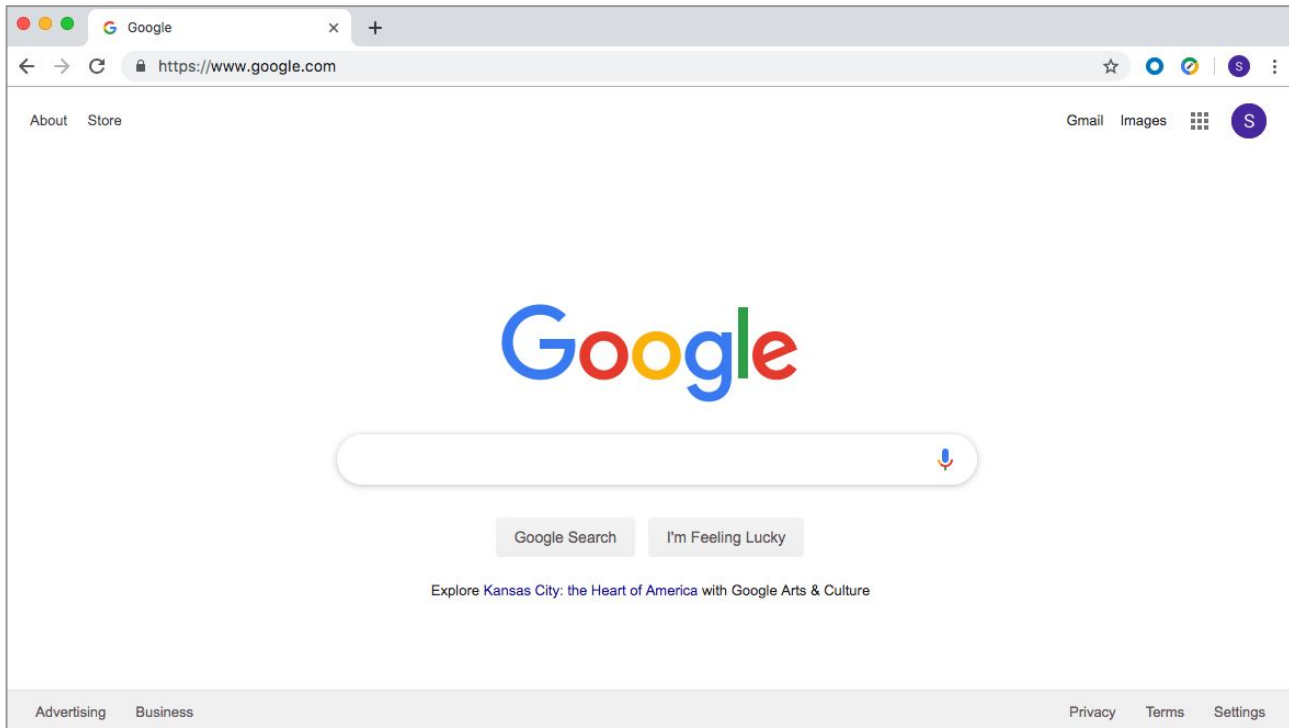


How (or where) do we  
connect all three?

# They Are Connected in the Web Browser

---

The browser brings together HTML, CSS, and JavaScript to create interactive webpages and applications.



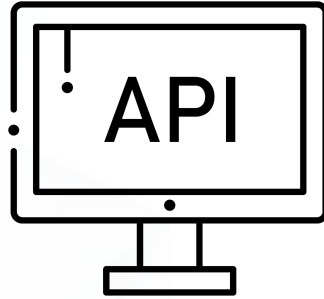




What is a web browser?



A **web browser**, or **browser**, is a program used to access information on the World Wide Web. Every webpage, image, and video on the web can be accessed via a specific Unified Resource Link (URL). This lets browsers retrieve these resources from a web server and display them on a user's device.



**What is an application  
programming interface (API)?**



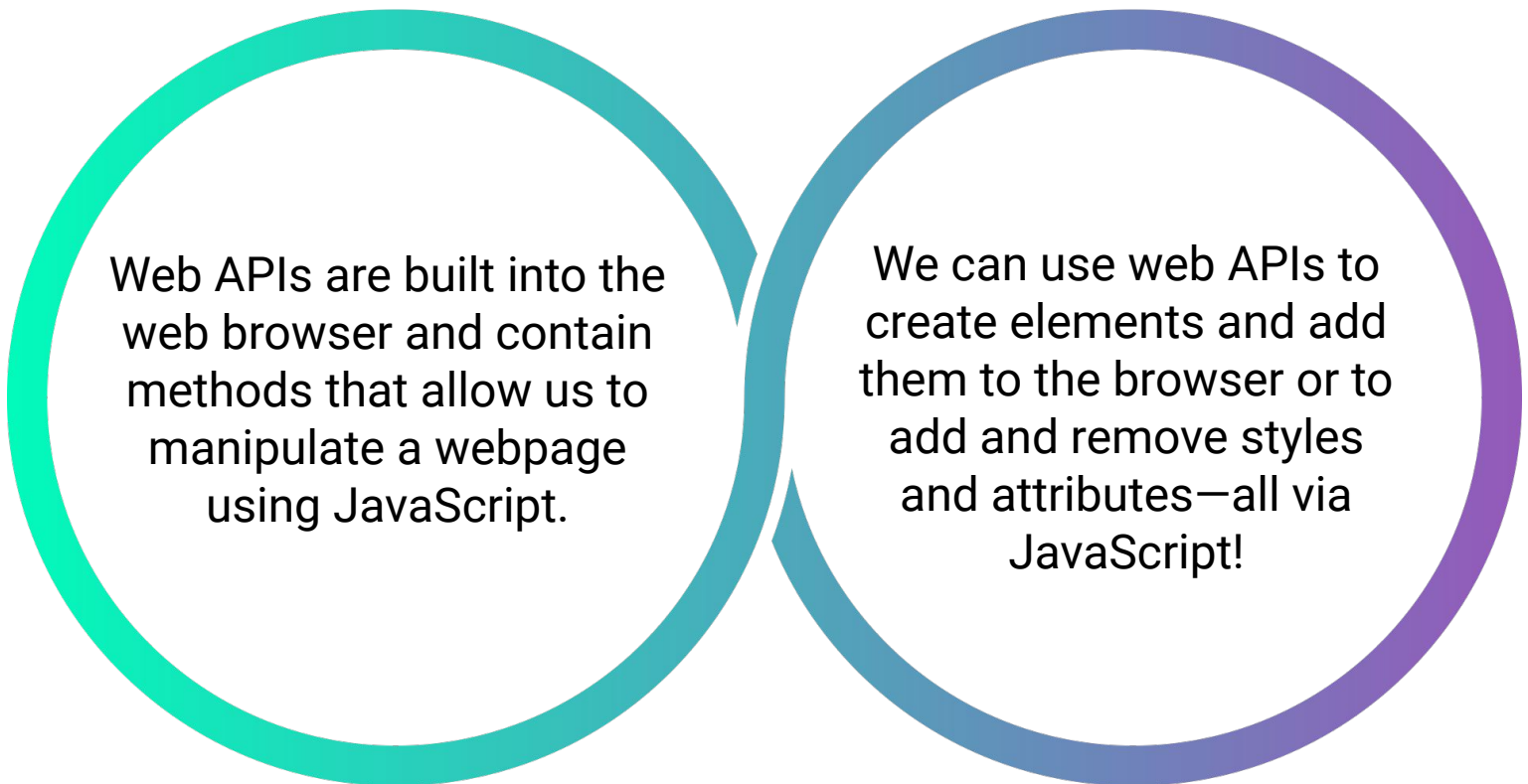
In web development, an **API** is a set of code features (methods, properties, events, and URLs) that developers can use in their apps to interact with components of a user's web browser, data sets, hardware/software on a user's computer, or third-party software and services.



What are web APIs?

# Web APIs

---



Web APIs are built into the web browser and contain methods that allow us to manipulate a webpage using JavaScript.

We can use web APIs to create elements and add them to the browser or to add and remove styles and attributes—all via JavaScript!

---



## **Activity:** This Window

See instructions in `01-Stu_This-Window` in the class repo.

In this activity, you will use `console.log(this)` and dig around inside the returned object, answering some questions along the way.

**Suggested Time:**  
10 minutes



# Activity: This Window

---

## Instructions

- First open the provided `index.html` file in the browser and navigate to the console.

- What is logged?

The `window` object. In this use case, `this` refers to the window. The `window` is an object representation of an open window in a browser.

- Click in the `window` object and begin looking at the numerous properties and methods it contains.
- Make your way down to `document` and click in it.
- Spend some time looking through the properties and methods in `window.document`.







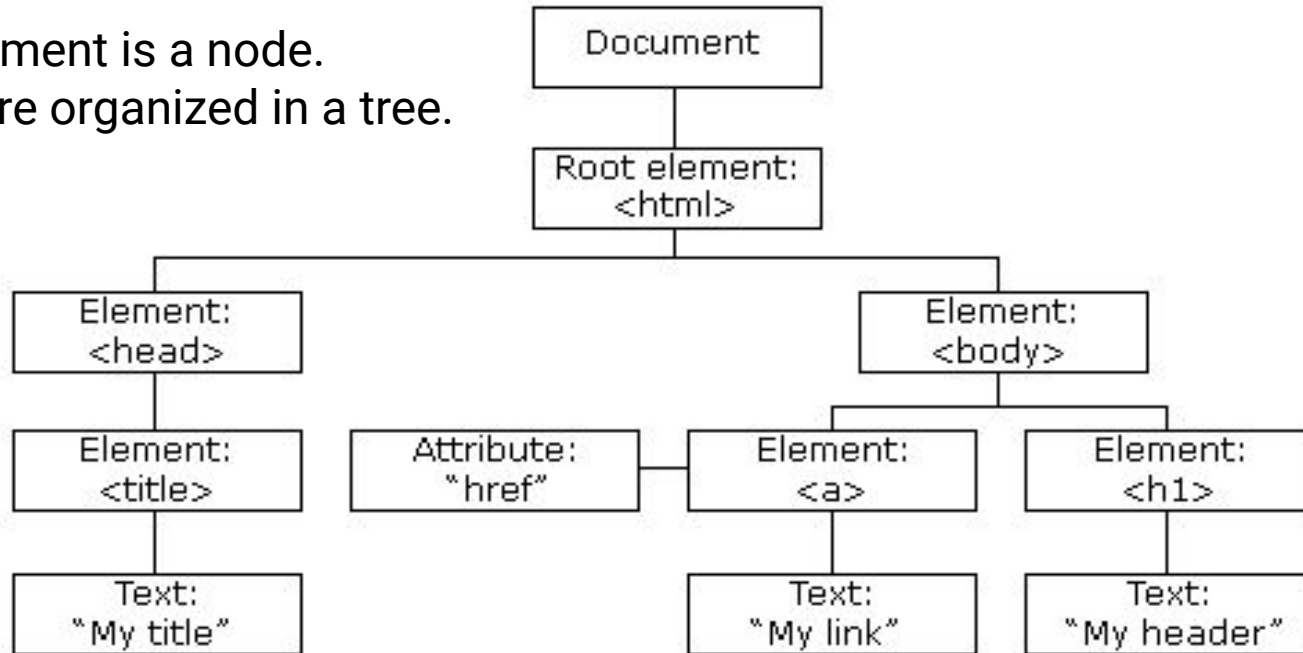
**What is `window.document`?**

# The Document Object Model (DOM)

---

The DOM is an object-oriented representation of HTML (i.e., the HTML document modeled as JavaScript objects).

Each element is a node.  
Nodes are organized in a tree.



# What Is the Node Tree of This HTML?

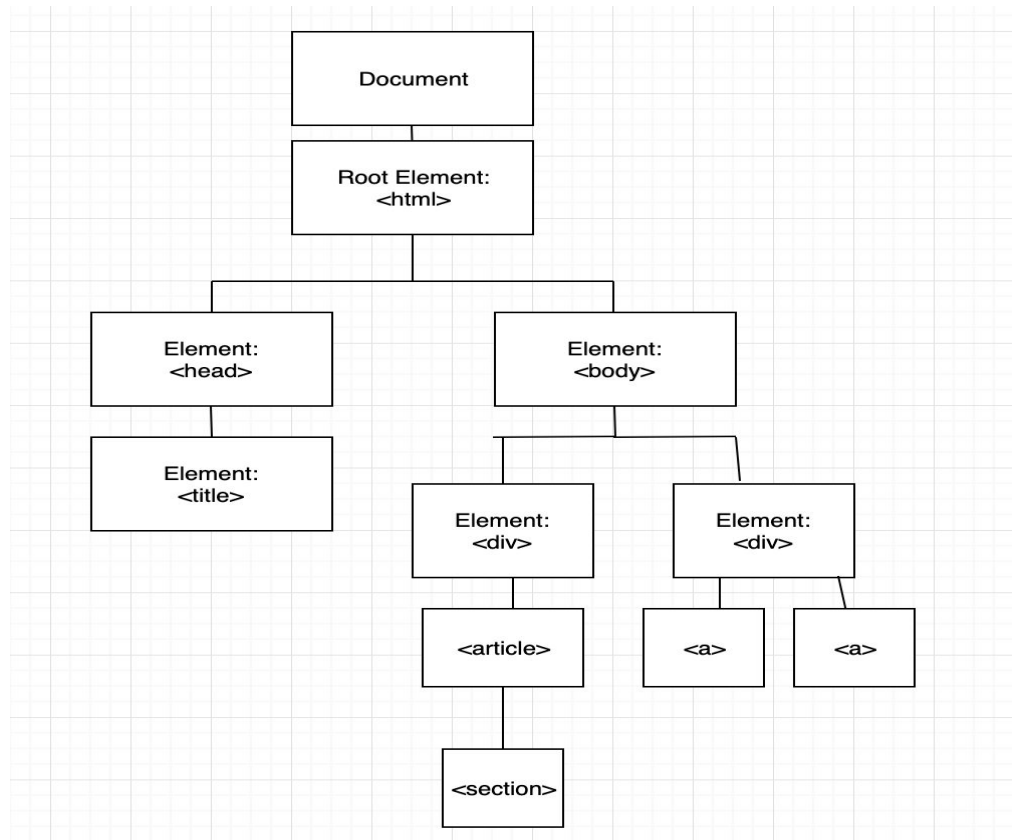
---

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Document</title>
  </head>
  <body>
    <div>Main div
      <article>
        <section>

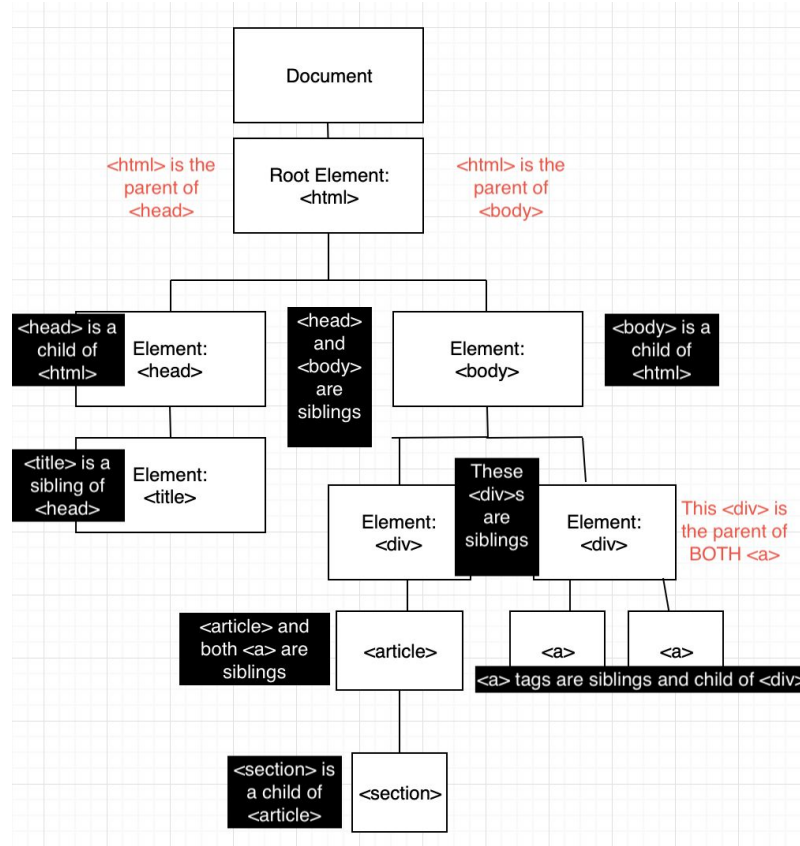
          </section>
        </article>
      </div>
      <div>
        <a href="myImg"></a>
        <a href="secondImg"></a>
      </div>
    </body>
  </html>
```

# What Are the Parent, Child, and Sibling Nodes?

---



# Node Relationships: Parents, Children, and Siblings





**What is traversing the DOM?**

# DOM Traversal

---

Navigate to the [MDN DOM Docs](#). Open the Chrome Dev Tools and enter the following commands one by one.

```
console.log(document.body);  
console.log(document.body.children);  
console.log(document.body.children[3]);  
console.log(document.body.children[3].childNodes[7]);  
console.log(document.body.children[3].childNodes[7].style.fontSize = "20px");
```



When using the `style` method, properties with two words (such as `font-size`) become a single word and camelCased. `Font-size` becomes `fontSize`.

Here is one more example of `.style`:

```
console.log(document.body.children[3].childNodes[7].parentElement.style.color = "red");
```

# Our Goal Today

---

Navigate to the deployed [Speed Reader app](#).

