CHAPTER 10 – JS DOM

Contents of the slides is taken from WD3304 IWEP by HSL, 2019

TOPICS TO BE COVERED

» **Introduction to Document Object Model**

DOM

Event Listener

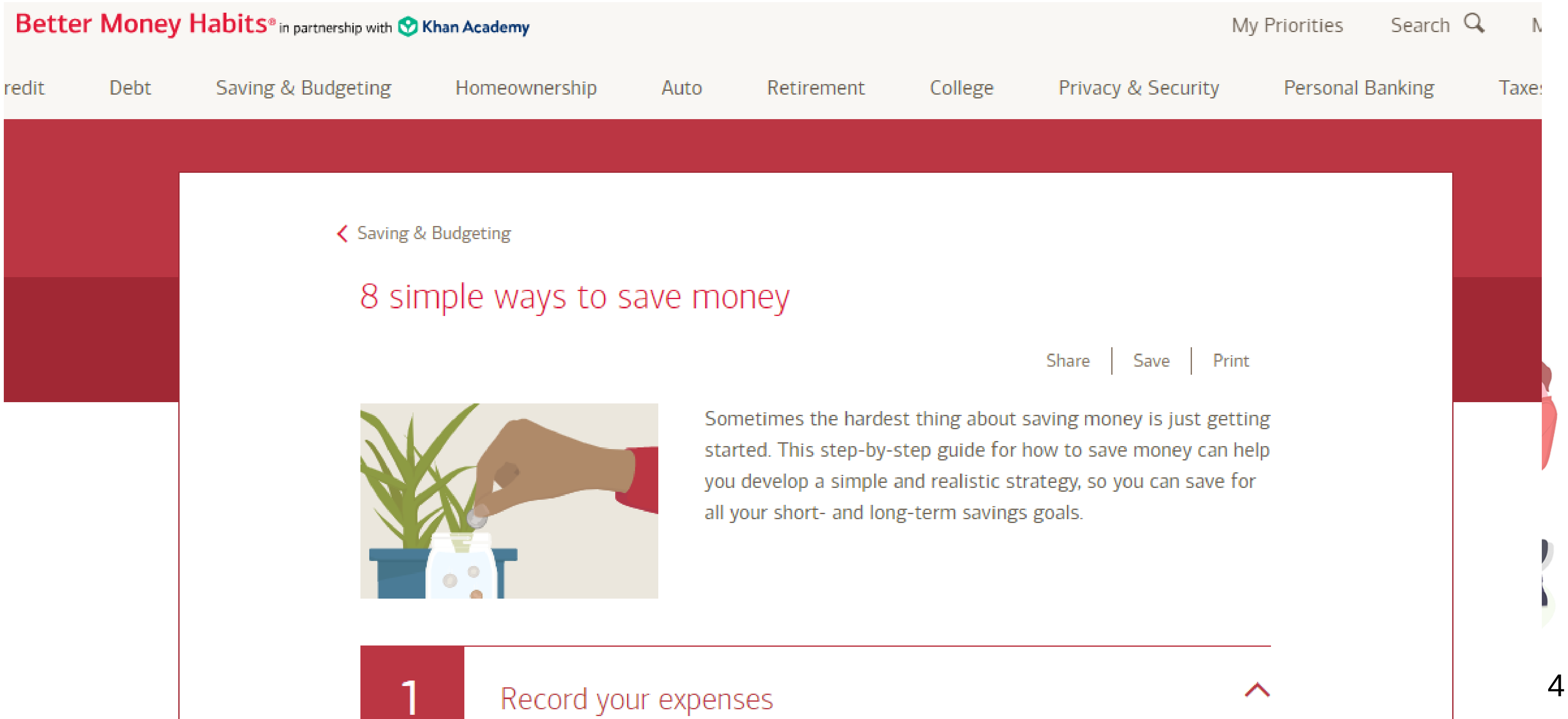
DOM 101

- DOM: Stands for document object model.
- Structured representation in a HTML document;
- The DOM is used to connect webpages to scripts like JavaScript.

- In a document, everything is a node.
- A document is also a node!
- Things like HTML elements, attributes and text elements are nodes!

- **BASICALLY EVERYTHING IS A NODE!!!**

This is a document.



Better Money Habits® in partnership with Khan Academy

My Priorities Search

redit Debt Saving & Budgeting Homeownership Auto Retirement College Privacy & Security Personal Banking Taxe

Saving & Budgeting

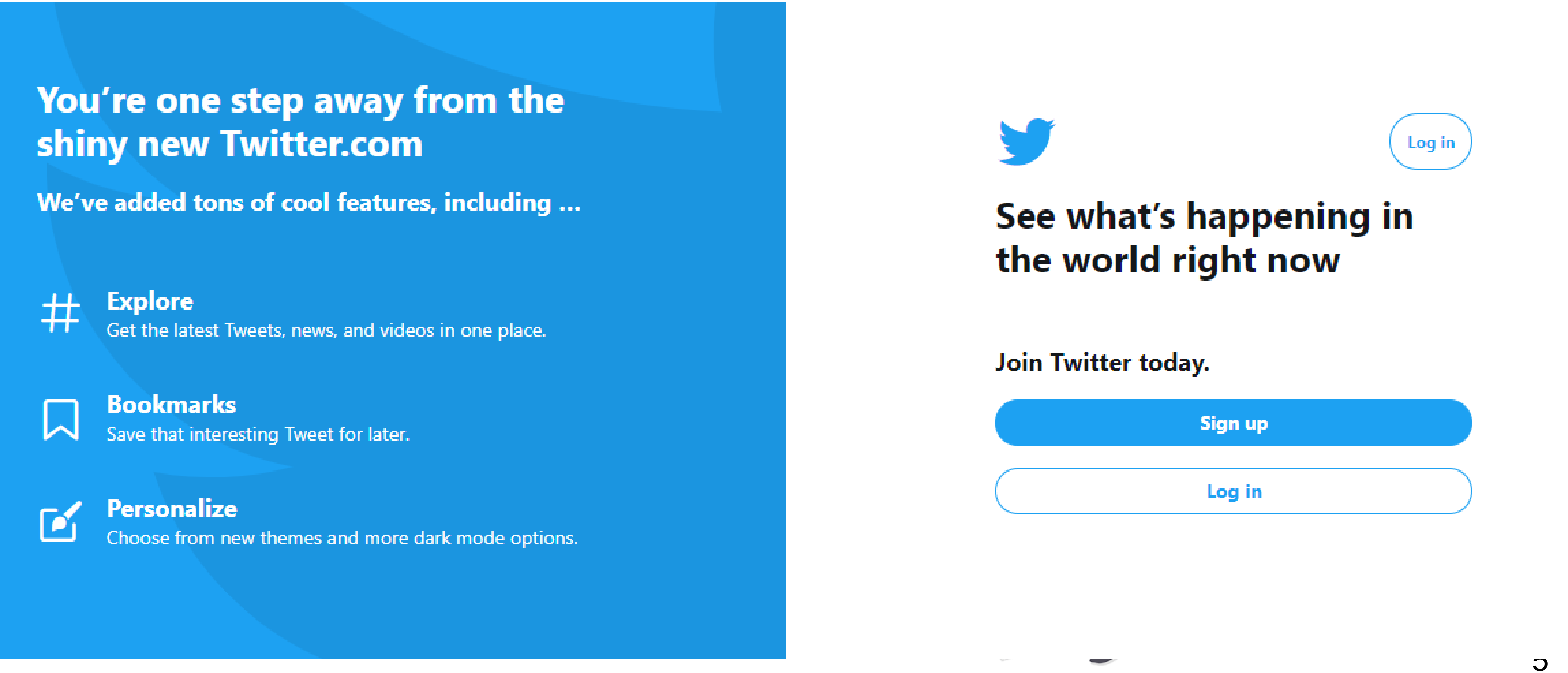
8 simple ways to save money

Share | Save | Print

Sometimes the hardest thing about saving money is just getting started. This step-by-step guide for how to save money can help you develop a simple and realistic strategy, so you can save for all your short- and long-term savings goals.

1 Record your expenses

This is another document.



You're one step away from the shiny new Twitter.com

We've added tons of cool features, including ...

Explore
Get the latest Tweets, news, and videos in one place.

Bookmarks
Save that interesting Tweet for later.

Personalize
Choose from new themes and more dark mode options.

Log in

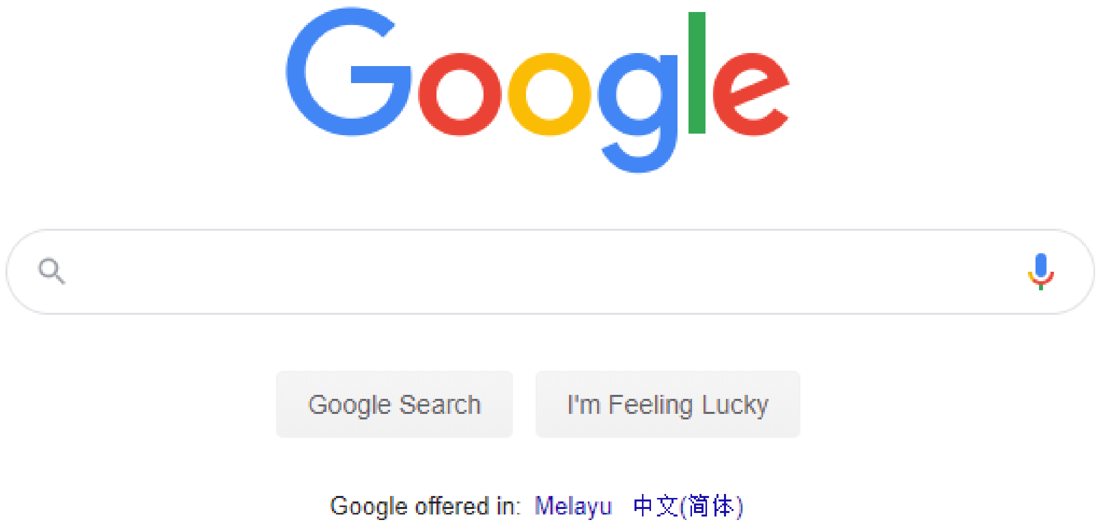
See what's happening in the world right now

Join Twitter today.

Sign up

Log in

Well, you get it right?

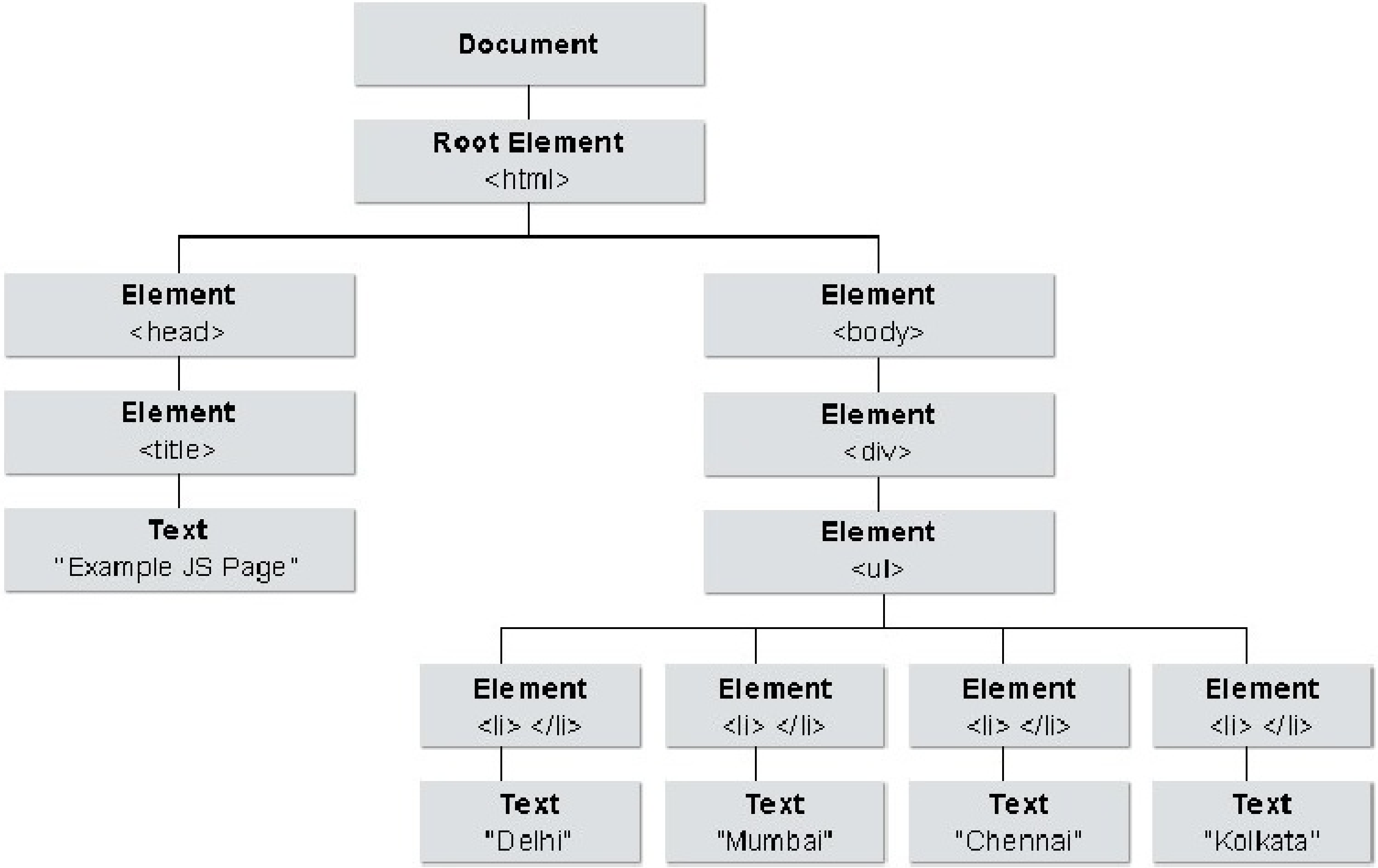


Google offered in: Melayu 中文(简体)

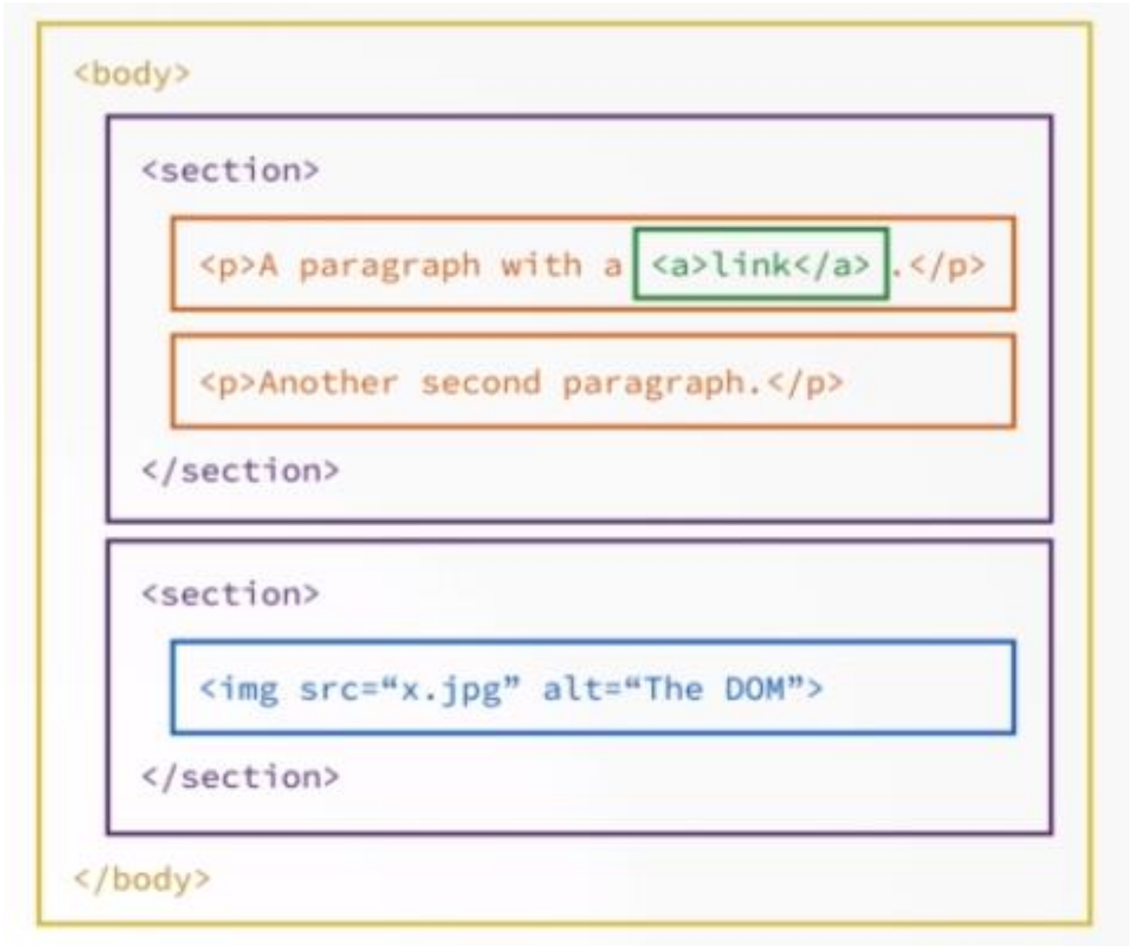
A Simple Webpage....

```
<html>
<head>
  <title>Example JS Page</title>
</head>
<body>
  <div id="Rayyan">
        <ul>
          <li>Delhi</li>
          <li>Mumbai</li>
          <li>Chennai</li>
          <li>Kolkata</li>
        </ul>
  </div>
</body>
</html>
```

Hierarchical! From top to bottom!



Document
Root Element
<html>
Element
<head>
Element
<title>
Text
"Example JS Page"
Element
<body>
Element
<div>
Element
<ul>
Element
<li> </li>
Element
<li> </li>
Element
<li> </li>
Element
<li> </li>
Text
"Delhi"
Text
"Mumbai"
Text
"Chennai"
Text
"Kolkata"



```
<body>
  <section>
    <p>A paragraph with a <a>link</a>.</p>
    <p>Another second paragraph.</p>
  </section>
  <section>
    <img src="x.jpg" alt="The DOM">
  </section>
</body>
```

All these highlighted elements are nodes that we can access and interact with using JS!

A Tree of Nodes

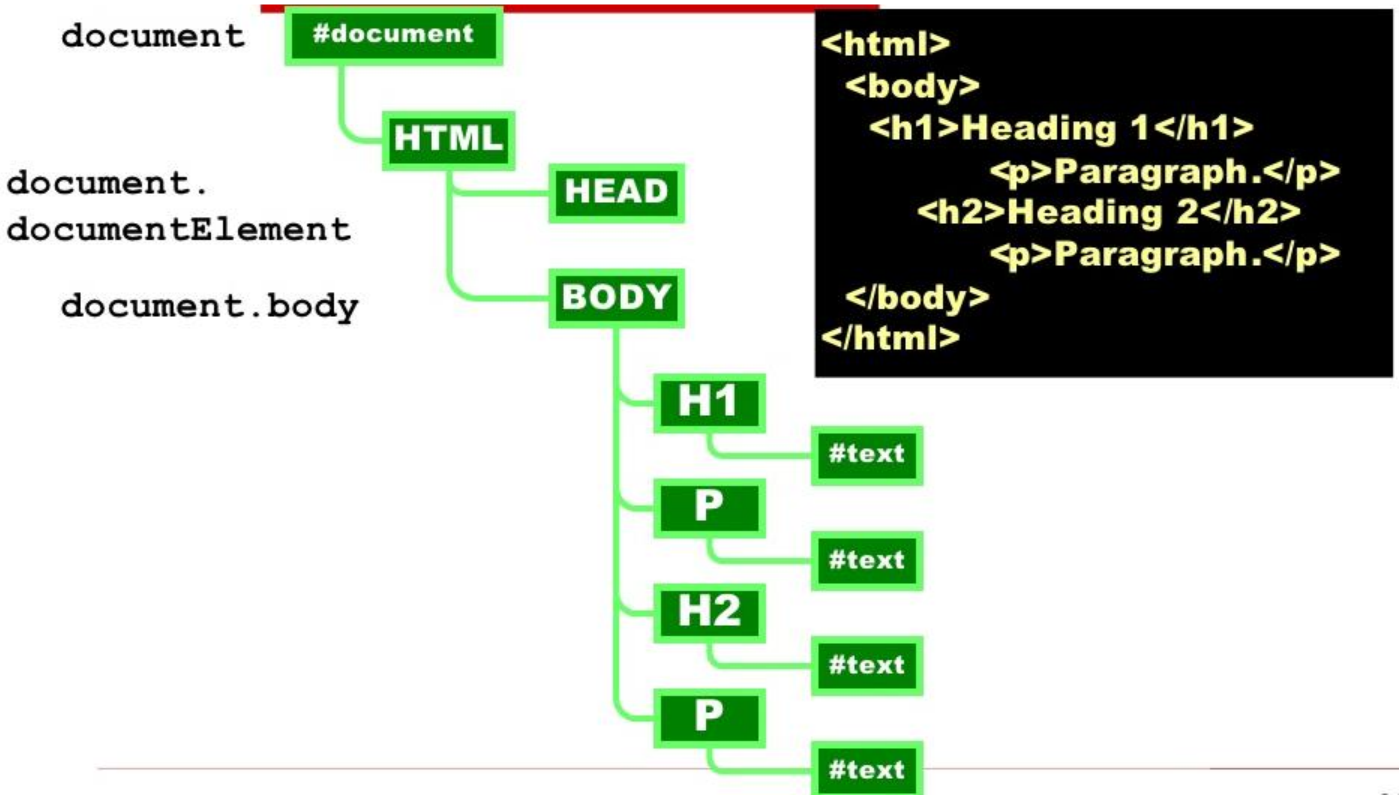
- A document is represented as a tree of nodes of various types.
- Many nodes have child nodes (and any child has a direct parent).
- A child is a node. Any node can have children.
- The ‘sibling’ of a node is a node that is on the same level or line. For example, all the <li> nodes under <ul> are siblings to each other.

A Tree of Nodes Example

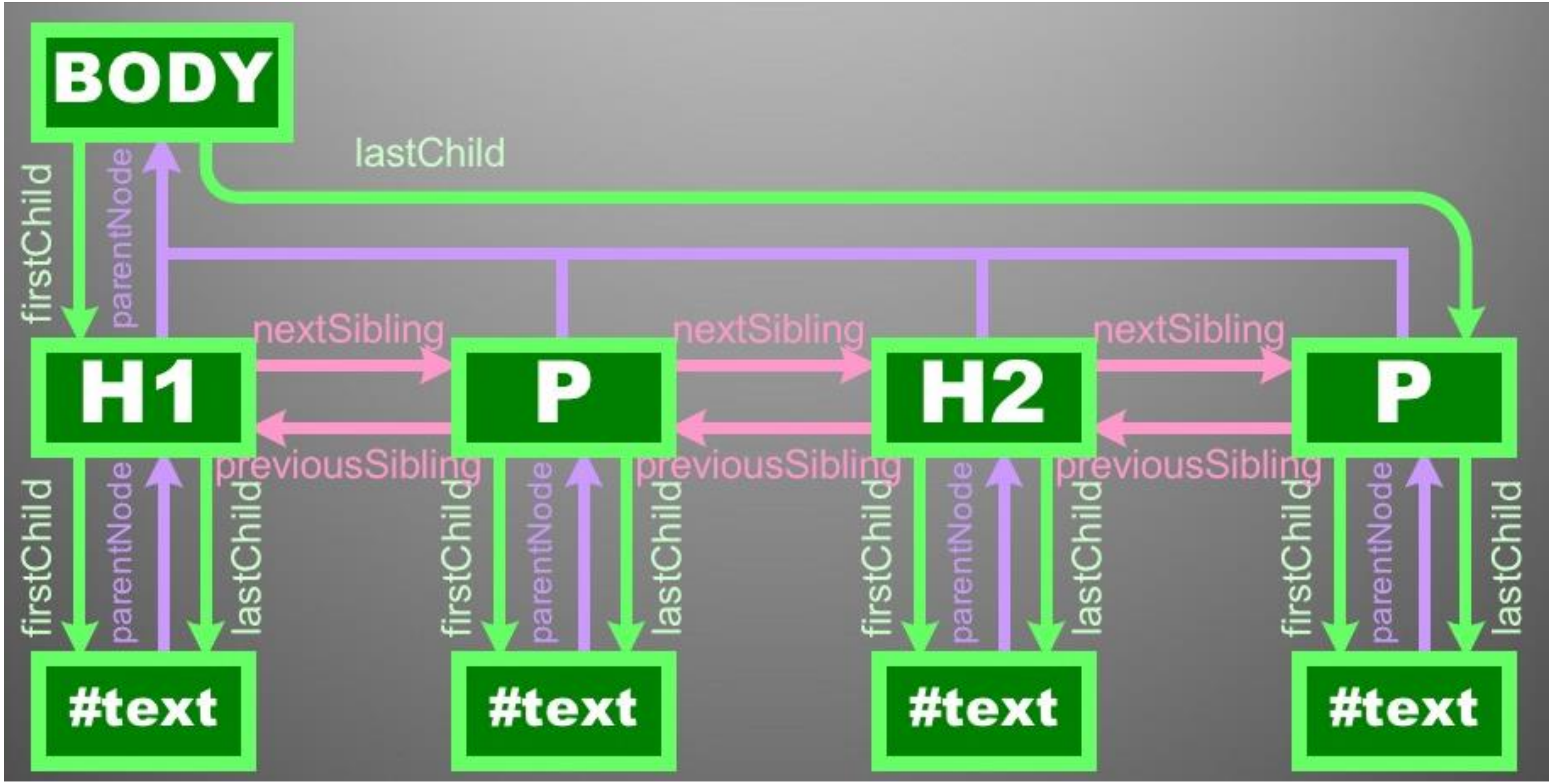
```
<div>
<p>This is a <em>new</em> paragraph
in an HTML document.</p>
</div>
```

- The <p> is a child of the <div>
- The <em> is a child of the <p>

More DOM Tree Examples

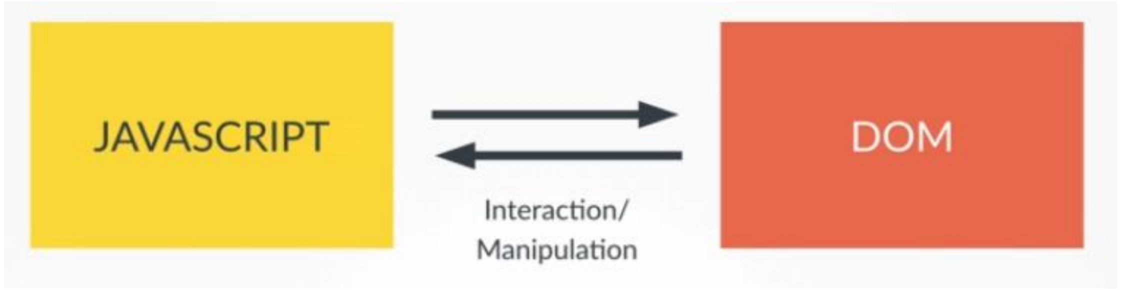


```
<html>
 <body>
  <h1>Heading 1</h1>
         <p>Paragraph.</p>
      <h2>Heading 2</h2>
         <p>Paragraph.</p>
 </body>
</html>
```


BODY
lastChild
firstChild
parentNode
nextSibling
nextSibling
nextSibling
H1
P
H2
P
previousSibling
previousSibling
previousSibling
firstChild
parentNode
lastChild
#text
#text
#text
#text

JavaScript vs DOM

- They are two different things.



- Been using console.log and document.write to print results to our webpage **WITHOUT** interacting with our webpage.

JavaScript vs DOM

- Use of special JavaScript methods to access and manipulate the DOM of a webpage.

- We will be able to change things like the content of a paragraph or a div, or even add more paragraphs or list elements to the webpage, hide and show elements and etc!

- …not to mention change the style of our webpage too!

Commonly Used Selectors

- **Get Element by ID**.
  - *document.getElementById('myID');*
    - It will select all the elments with the id 'myID'

- **Get Element by Tag Name.**
  - *document.getElementByTagName('div');*
    - It will select all the elements which is a div in your page.
  - *document.getElementByTagName('div')[0];*
    - It will select all the first element which is a div in your page.

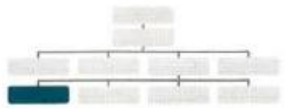
Commonly Used Selectors

- **Get Element by Class Name**
  - *document.getElementByClassName(‘myClass’);*
    - It will select all the elments with the class ‘myClass’.

- **Get Element by selector**
  - *querySelectorAll();*
    - Allows accessing nodes based on CSS-style selector.
    - var sidebarPara = document.querySelectorAll(“.sidebar p”);

ACCESSING THE ELEMENTS

SELECT AN INDIVIDUAL ELEMENT NODE



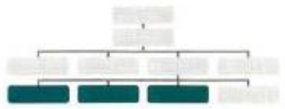
Here are three common ways to select an individual element:

`getElementById()`
Uses the value of an element's `id` attribute (which should be unique within the page).
See p195

`querySelector()`
Uses a CSS selector, and returns the first matching element.
See p202

You can also select individual elements by traversing from one element to another within the DOM tree (see third column).

SELECT MULTIPLE ELEMENTS (NODELISTS)



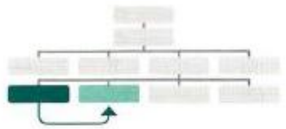
There are three common ways to select multiple elements.

`getElementsByClassName()`
Selects all elements that have a specific value for their `class` attribute.
See p200

`getElementsByTagName()`
Selects all elements that have the specified tag name.
See p201

`querySelectorAll()`
Uses a CSS selector to select all matching elements.
See p202

TRAVERSING BETWEEN ELEMENT NODES



You can move from one element node to a related element node.

`parentNode`
Selects the parent of the current element node (which will return just one element).
See p208

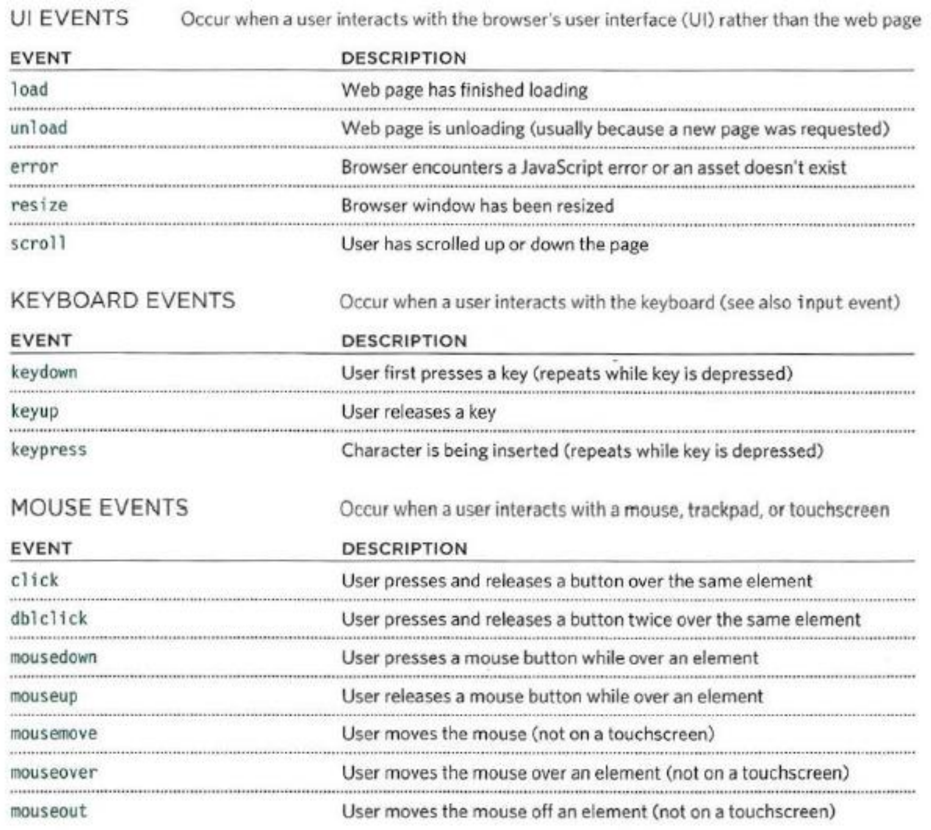
`previousSibling` / `nextSibling`
Selects the previous or next sibling from the DOM tree.
See p210

`firstChild` / `lastChild`
Select the first or last child of the current element.
See p211

Events and Event Handling

- **Events:** Notifications that are sent to notify code that something happened on the webpage.
  - E.g.: Clicking a button, resizing a window, scrolling down or pressing a key, opening a pop up window etc.
- **Event Listener:** A function that performs an action based on a certain event. It waits for a specific event to happen.
  - E.g: What happens after you press a key or click on a button?

Event References



UI EVENTS — Occur when a user interacts with the browser's user interface (UI) rather than the web page

EVENT	DESCRIPTION
`load`	Web page has finished loading
`unload`	Web page is unloading (usually because a new page was requested)
`error`	Browser encounters a JavaScript error or an asset doesn't exist
`resize`	Browser window has been resized
`scroll`	User has scrolled up or down the page

KEYBOARD EVENTS — Occur when a user interacts with the keyboard (see also `input` event)

EVENT	DESCRIPTION
`keydown`	User first presses a key (repeats while key is depressed)
`keyup`	User releases a key
`keypress`	Character is being inserted (repeats while key is depressed)

MOUSE EVENTS — Occur when a user interacts with a mouse, trackpad, or touchscreen

EVENT	DESCRIPTION
`click`	User presses and releases a button over the same element
`dblclick`	User presses and releases a button twice over the same element
`mousedown`	User presses a mouse button while over an element
`mouseup`	User releases a mouse button while over an element
`mousemove`	User moves the mouse (not on a touchscreen)
`mouseover`	User moves the mouse over an element (not on a touchscreen)
`mouseout`	User moves the mouse off an element (not on a touchscreen)

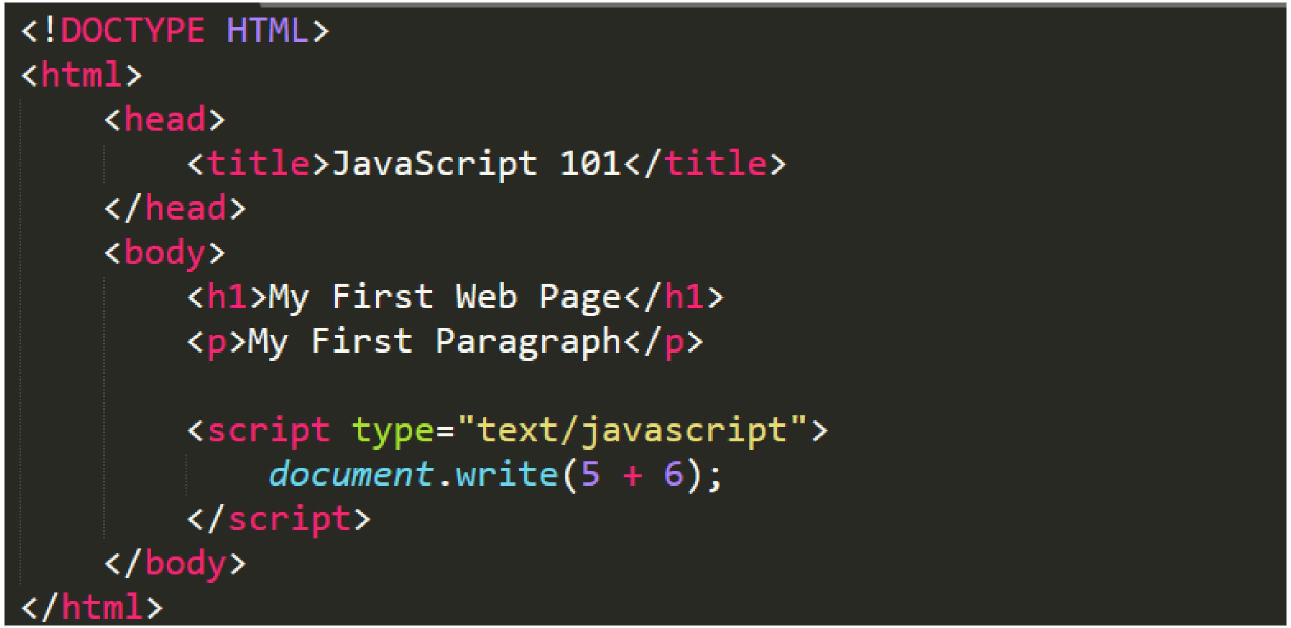
Displaying Data

- JavaScript can display data in “different ways”:
  - Writing into the HTML output using document.write().
  - Writing into an alert box, using window.alert().
  - Writing into an HTML element, using innerHTML.
  - Writing into the browser console, using console.log().

- We will explore more on each of these as we code along the module.

Using document.write()

- For testing purposes only, it is convenient to use document.write():

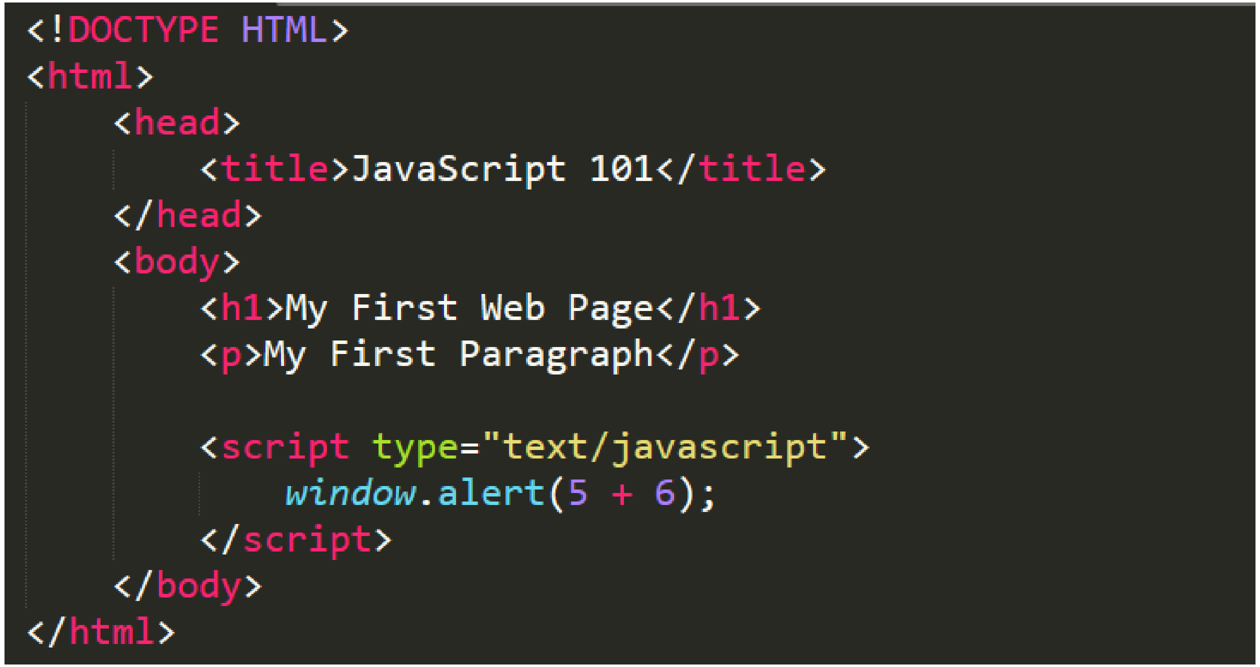


```
<!DOCTYPE HTML>
<html>
    <head>
        <title>JavaScript 101</title>
    </head>
    <body>
        <h1>My First Web Page</h1>
        <p>My First Paragraph</p>

        <script type="text/javascript">
            document.write(5 + 6);
        </script>
    </body>
</html>
```

Using window.alert()

- You can use window.alert() to display data in an alert box:



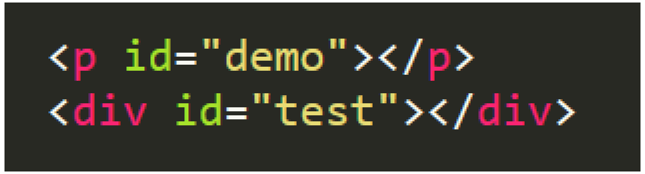
```
<!DOCTYPE HTML>
<html>
    <head>
        <title>JavaScript 101</title>
    </head>
    <body>
        <h1>My First Web Page</h1>
        <p>My First Paragraph</p>

        <script type="text/javascript">
            window.alert(5 + 6);
        </script>
    </body>
</html>
```

Using innerHTML

- To access an HTML element, JavaScript can use the document.getElementById(id) method.

- The id attribute defines the HTML element. The innerHTML property defines the HTML content.

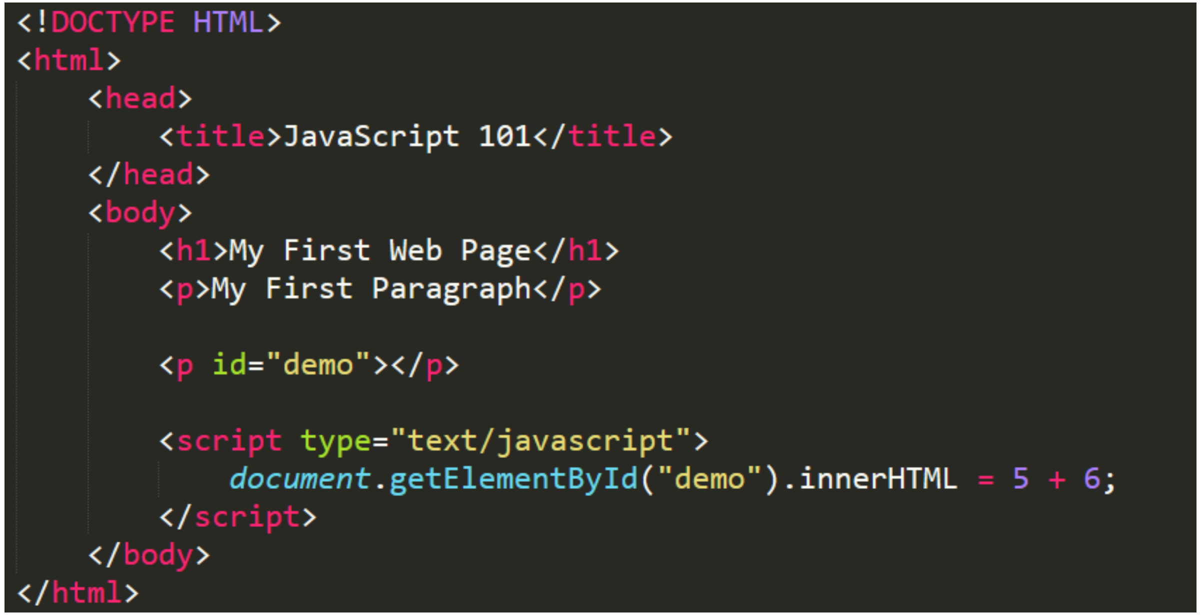


```
<p id="demo"></p>
<div id="test"></div>
```

- We will look more into this when we proceed to DOM manipulation.

Using innerHTML

- Changing the innerHTML property of an HTML element is a common way to display data in HTML using JavaScript.



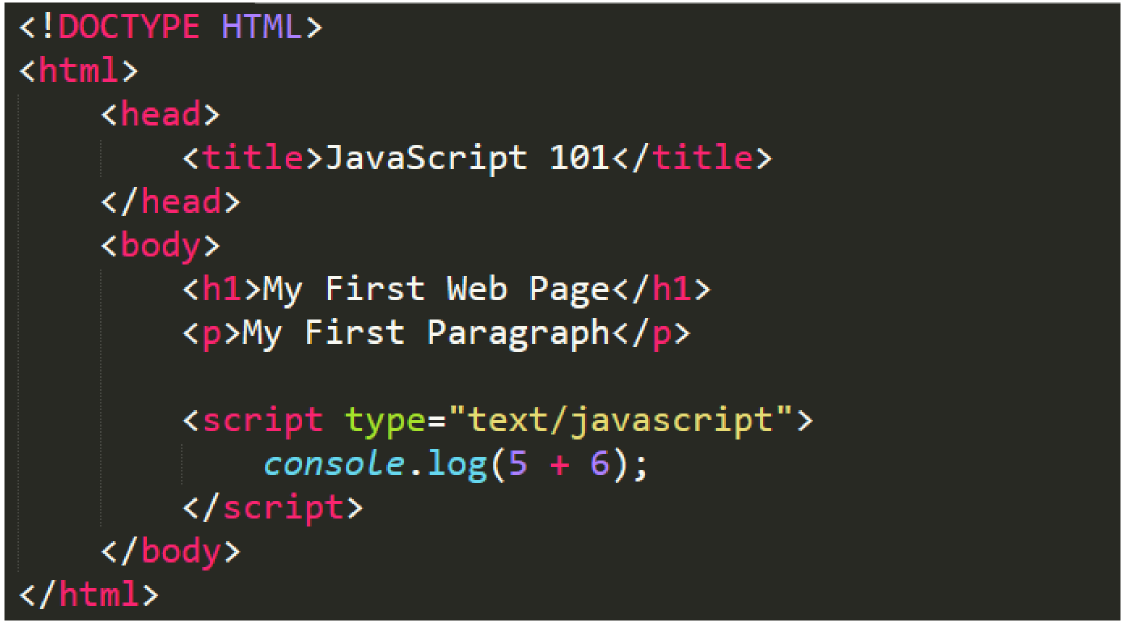
```
<!DOCTYPE HTML>
<html>
    <head>
        <title>JavaScript 101</title>
    </head>
    <body>
        <h1>My First Web Page</h1>
        <p>My First Paragraph</p>

        <p id="demo"></p>

        <script type="text/javascript">
            document.getElementById("demo").innerHTML = 5 + 6;
        </script>
    </body>
</html>
```

Using console.log()

- For debugging purposes, you can use the console.log() method to display data:

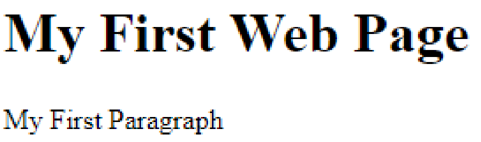


```
<!DOCTYPE HTML>
<html>
    <head>
        <title>JavaScript 101</title>
    </head>
    <body>
        <h1>My First Web Page</h1>
        <p>My First Paragraph</p>

        <script type="text/javascript">
            console.log(5 + 6);
        </script>
    </body>
</html>
```

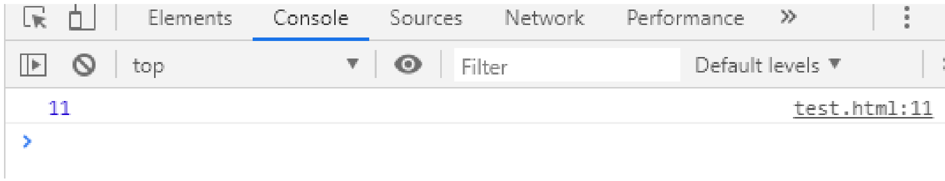
How to check console.log?

- As console.log() suggested, the data printed or displayed by this function will be logged in your browser console.
- Your console can be easily accessed by right-clicking in your browser and click “Inspect”.
- Then, click on **console** to see the value displayed. We will be using this often.



My First Web Page

My First Paragraph



```
11                                                    test.html:11
>
```

CHANGING CSS USING DOM

- To change the style of an HTML element,

  document.getElementById(*id*).style.property = new style;

- For example, changing the paragraph to color *tomato.*

```
<p id="para">Change my color</p>
<script>
    document.getElementById("para").style.color = "tomato";
</script>
```