**JavaScript DOM + Modern JS**

**Window**

* It is a global object.
* Created by browser.
* It represents a browser window.
* It acquires DOM, BOM, JS Core functions.
* It is a top-level entity.
* It can be used to manipulate browser window.
* Example: windows.console.log() and much more.

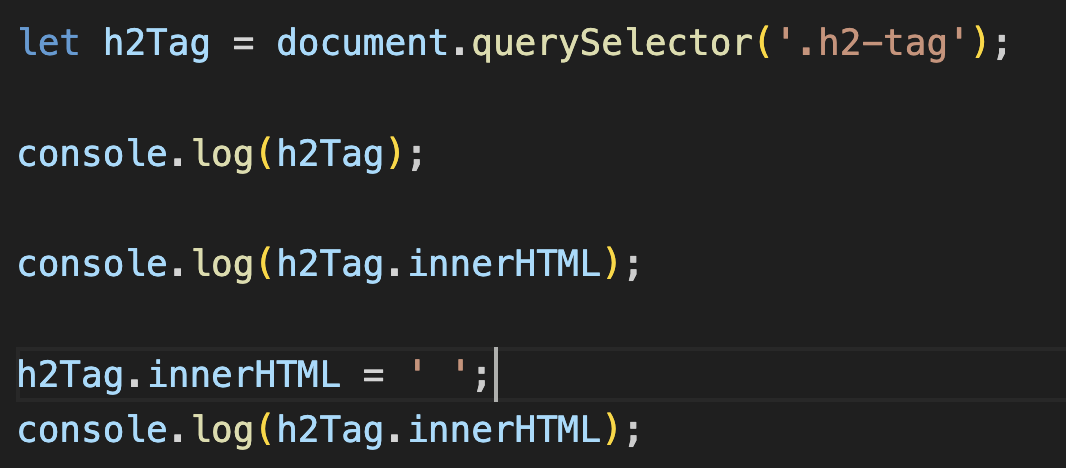
**DOM**

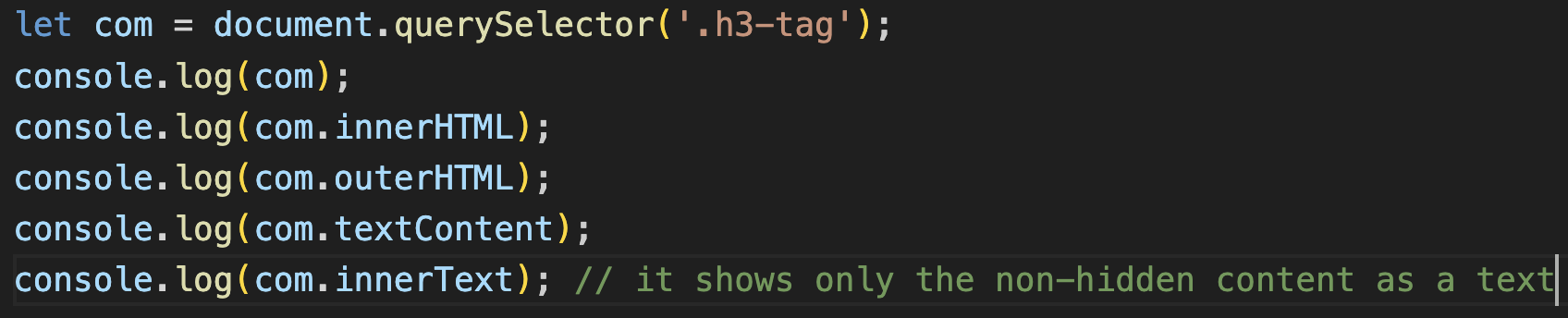
* Document object model.
* Every webpage content is converted into JS Object known as document.
* Example: document.body – it represent html body.

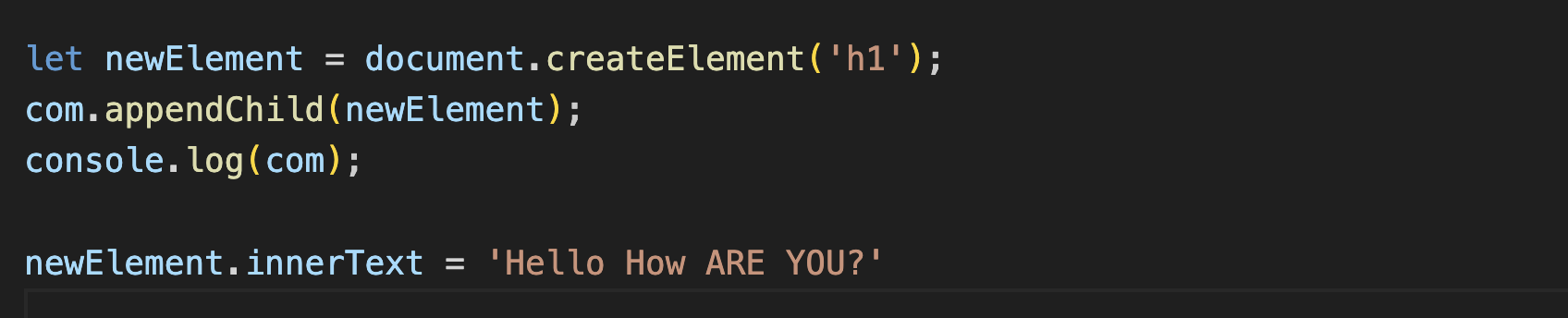
**BOM**

* It allows JS to talk to browser on different thing except page content.
* Like with the help of alert() etc.

**Document Object Model (DOM):**

* Tree-like structure starts with root.
* Root -> html -> head, body
* How it happens.
  + Character -> tags -> token -> nodes -> DOM.
* 
* **Fetching Elements of Document**
  + **getElementByID**(‘<id-given-in-html-code>’)
    - It returns an Element object representing the element whose id property matches the specified string.
    - If not matches with any of the string (id) then it returns null.
    - It is called on document object.
    - It returns only a single object because id is unique.
    - For multiple objects: we use another method.
  + **getElementsByClassName**(‘<class-name-given-in-html-code>’)
    - this interface returns an array like object of all child elements which have all of the given class name(s).
    - when called on the document object, the complete document is searched, including the root node.
    - It is called on document object.
    - Can be iterate via for of loop or any useful loop.
    - It returns HTML Collection.
    - Same as **getElementsByTagName**(<tag>): it returns all the element whose tag is similar to this tag.
    - The list returned is not an array.
  + **querySelector**(‘# or . <name>’):
    - it selects class or any id.
    - We can select element using query selector as above.
    - It returns only single object / element.
    - If multiple section uses same class, then it returns the first one with the matching string.
  + **querySelectorAll**(‘# or . <name>’):
    - it selects same as upper selectors.
    - It returns list of all elements / object. Whose class name or id matches with the specified string.
* **Updating the existent content:**
  + innertHTML:
    - get/set the HTML Content.
    - Get an element or all of its descendent.
    - Set an element’s HTML Content.
    - To only obtain the HTML representation of the contents of an element, or to replace the content of an element, use the innerHTML property instead.
    - 
  + outerHTML:
    - attribute of element dom interface gets the serialized html fragment describing the element including its descendent.
    - It can also be set to replace the element with nodes parsed from the given string.
  + textContent:
    - get / set textual content
    - it renders the tag and show them as it is.
  + innerText
    - get / set textual content.



* Adding new elements using JavaScript:
  + createElement(‘tag-name’):
    - using .appendChild() to append newly created element.
    - 
* Creating a Text-Node:
  + Same as above picture but using document.createTextNode(‘I am the text’) not an easy way.
  + Using textContent.
  + appendChild() – append the element at the last of element.
* How to insert element at the required position
  + insertAdjacentHTML():
    - it has to be called with 2 arguments (location/position, HTML / textContent to be inserted).
    - Location / position
      * beforeBegin: at previous sibling
      * afterBegin: after first sibling
      * beforeEnd: before ending
      * afterEnd: after ending
* How to remove Element at the required position:
  + removeChild():
    - opposite to appendChild()
    - we should know the parent element of the element we are deleting
    - the child element to be removed.
    - parent.removeChild(childElement);
    - drawback is only this we should know the parent.
    - Another method is to find parent using parent = childElement.parent;
* Using JS to change the CSS:
  + .style
    - At a time single property is being changed.
  + .cssText
    - Multiple properties can be changed at a time
  + .setAttribute
    - Separation of concern are being spoiled here.
    - Here style, id, class Name are also being added here.
  + .className
    - It returns the class Name as a string
    - It is not the best method among classList because we need to do trim(), then split() and so on to get all the class names.
  + .classList
    - It returns the list so that we can only iterate to get all the class names.
    - Return array of classes.
    - It has many operations – add(), remove(), toggle(), contains(), etc
  + 