```
                            ┌──────────────┐
                            │    window    │
                            └──────────────┘
            ╱                      │                      ╲
          ↙                        ↓                        ↘
   DOM                      BOM                      JavaScript
   ┌──────────────┐        ┌──────────────┐        ┌──────────────┐
   │   document   │        │   navigator  │        │    Object    │
   └──────────────┘        └──────────────┘        └──────────────┘
         ...               ┌──────────────┐        ┌──────────────┐
                           │    screen    │        │    Array     │
                           └──────────────┘        └──────────────┘
                           ┌──────────────┐        ┌──────────────┐
                           │   location   │        │   Function   │
                           └──────────────┘        └──────────────┘
                           ┌──────────────┐               ...
                           │    frames    │
                           └──────────────┘
                           ┌──────────────┐
                           │   history    │
                           └──────────────┘
                        ┌────────────────────┐
                        │   XMLHttpRequest    │
                        └────────────────────┘
```

* The window object is supported by all browsers. It represents the browser's window.
* All global JavaScript objects, functions, and variables automatically become members of the window object.
* Global variables are properties of the window object.
* Global functions are methods of the window object.

# * Browser Object Model (BOM)

*window.innerHeight - the inner height of the browser window (in pixels)

*window.innerWidth - the inner width of the browser window (in pixels)

*window.open() - open a new window

*window.close() - close the current window

*Window

*object contains information about the user's screen.

* screen.width property returns the width of the visitor's screen in pixels.

* screen.height property returns the height of the visitor's screen in pixels.

* screen.availWidth property returns the width of the visitor's screen, in pixels, minus interface features like the Windows Taskbar.

* screen.availHeight property returns the height of the visitor's screen, in pixels, minus interface features like the Windows Taskbar.

*Window Screen

*window.location.href returns the href (URL) of the current page

*window.location.hostname returns the domain name of the web host

*window.location.pathname returns the path and filename of the current page

*window.location.protocol returns the web protocol used (http: or https:)

*window.location.assign() loads a new document

*Window Location

*history.back() - same as clicking back in the browser

*history.forward() - same as clicking forward in the browser

*Window History

* Every web page resides inside a browser window, which can be considered as an object.

* The way a document content is accessed and modified is called the **Document Object Model**, or **DOM**. The objects are organized in a hierarchy. This hierarchical structure applies to the organization of objects in a web document.

*HTML DOM

* JavaScript can change all the HTML elements in the page
* JavaScript can change all the HTML attributes in the page
* JavaScript can change all the CSS styles in the page
* JavaScript can remove existing HTML elements and attributes
* JavaScript can add new HTML elements and attributes
* JavaScript can react to all existing HTML events in the page
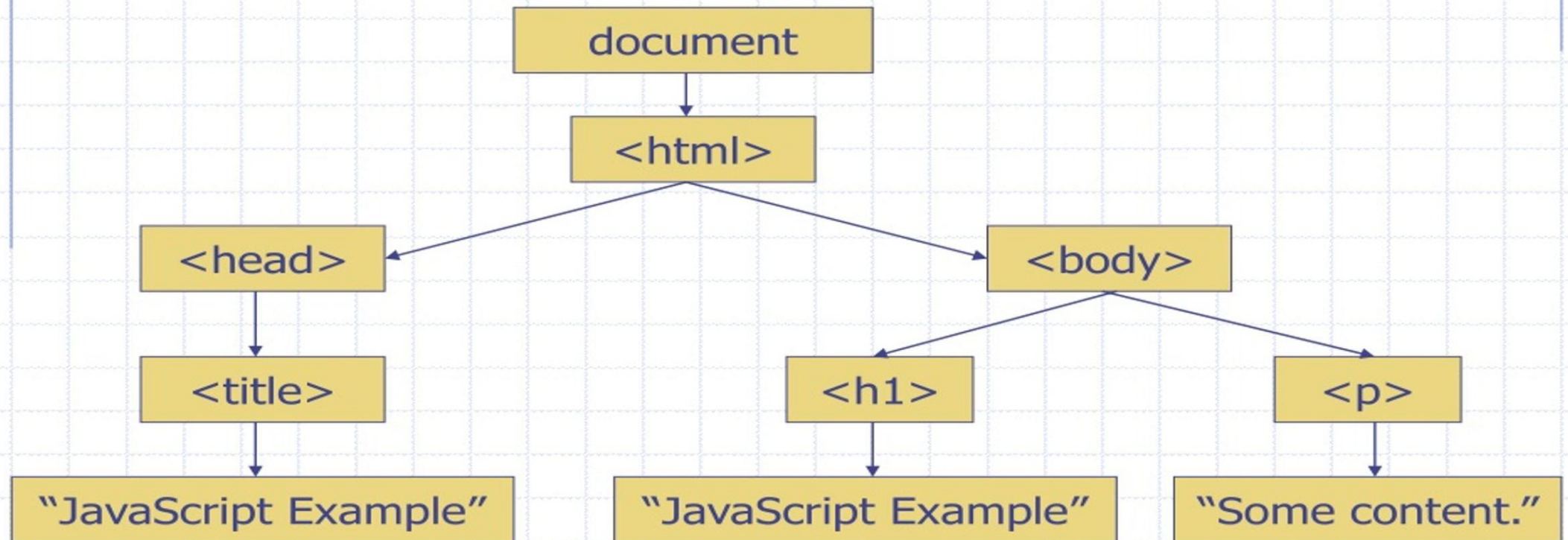* JavaScript can create new HTML events in the page

*DOM

# DOM in Practice

```html
<html>
  <head>
    <title>My Title</title>
  <head>
  <body>
    <section>
      <h1>My Section</h1>
    </section>
  </body>
<html>
```

HTML

HEAD

TITLE

My Title

BODY

SECTION

H1

My Section
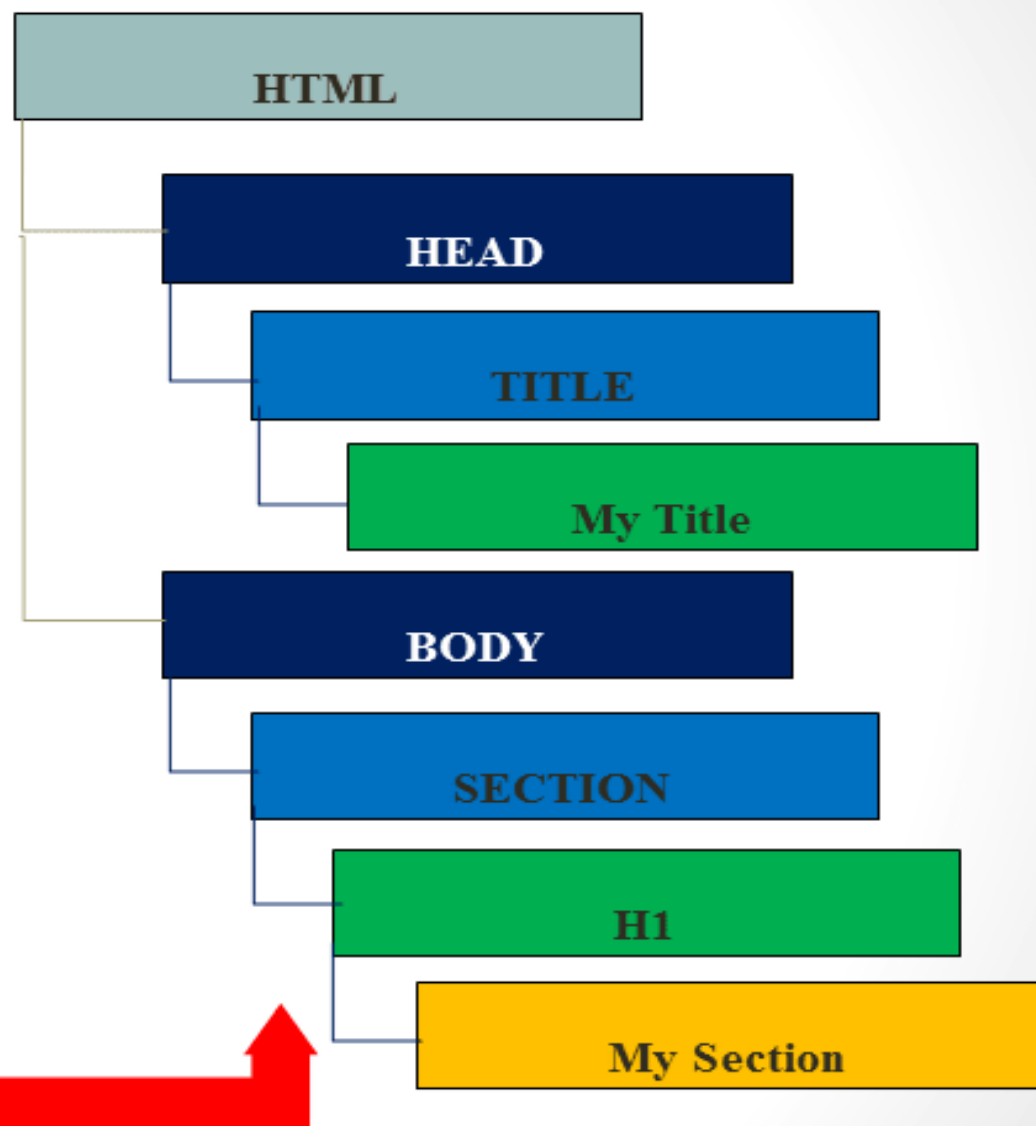
# DOM Representation

* forms[ ] An array of form objects, one for each HTML form that appears in the document.
  * Example : document.forms[0], document.forms[1] and so on
* images[ ] An array of form objects, one for each HTML form that appears in the document with the HTML <img> tag.
  * Example : document.forms[0], document.forms[1] and so on
* Referrer A read-only string that contains the URL of the document, if any, from which the current document was linked.
  * Example : document.referrer
* Title The text contents of the <title> tag.
  * Example : document.title
* URL A read-only string that specifies the URL of the document.
  * Example : document.URL

* Document Properties in DOM

\*document.getElementById(id)Find an element by element id

\*document.getElementsByTagName(name) Find elements by tag name

\*document.getElementsByClassName(name)Find elements by class name

\*document.querySelector(  id or class or type) Find element by name or id or class or type

# \*Finding HTML Elements

*element.innerHTML =  new html content  Change the inner HTML of an element

*element.attribute = new value Change the attribute value of an HTML element

*element.setAttribute(attribute, value) Change the attribute value of an HTML element

*element.style.property = new style Change the style of an HTML element

# *Changing HTML Elements

*document.createElement(element) Create an HTML element

*document.removeChild(element)  Remove an HTML element

*document.appendChild(element)  Add an HTML element

*document.replaceChild(element)  Replace an HTML element

*document.write(text) Write into the HTML output stream

# *Adding and Deleting Elements

* An event is an action or occurrence recognized by the software. It can be triggered by a user or the system. Some common examples of events include a user clicking on a button, loading the web page, clicking on a hyperlink and so on.

*Events

| Event | Description |
| --- | --- |
| onchange | An HTML element has been changed |
| onclick | The user clicks an HTML element |
| onmouseover | The user moves the mouse over an HTML element |
| onmouseout | The user moves the mouse away from an HTML element |
| onkeydown | The user pushes a keyboard key |
| onload | The browser has finished loading the page |

# *Common HTML Events

\* JavaScript provides a way to validate the form's data on the client's computer before sending it to the web server. Form validation generally performs two functions.

  \* **Basic Validation** – First of all, the form must be checked to make sure all the mandatory fields are filled in. It would require just a loop through each field in the form and check for data.

  \* **Data Format Validation** – Secondly, the data that is entered must be checked for correct form and value. Your code must include appropriate logic to test the correctness of data.

\*Validations

*The addEventListener() method attaches an event handler to an element without overwriting existing event handlers.

*can add many event handlers of the same type to one element, i.e two "click" events.

*DOM EventListener