

Web Programming (CSci 130)

Department of Computer Science
College of Science and Mathematics
California State University Fresno
H. Cecotti

Learning outcomes

- In this class, you will learn about the DOM
 - **Document Object Model**
- Web programming
 - Creation of webpages (HTML+CSS+Javascript)
 - Think about the browser
 - The HTML file must be parsed, decoded, and presented on the screen
 - How to decode this file so
 - It stays consistent across browsers
 - → Need of a data structure to contain the webpage downloaded on the server

Introduction

■ Javascript

- Initially for web browsers
- Evolution to other platforms
 - Browser
 - Web server
 - Another hos...
 - → host environment
- Host environment
 - Platform specific objects + functions
 - Web browsers → control web pages
 - Node.JS → server side functions

Introduction

- Window (root object) : access methods, properties of the window object
 - let h=window.innerHeight; // in pixels
 - let w=window.innerWidth; // in pixels
 - **DOM** (Document Object Model)
 - Document → access to the page content
 - document.body.style.background = 'blue';
 - 2 standards: W3C and WhatWC 😊
 - **BOM** (Browser Object Model)
 - Objects provided by the browser
 - Host environment
 - Example of objects: navigator, location
 - Navigator, screen, frames, history,
 - **JS** (Javascript)
 - Object, array, functions, ...

DOM

- Document Object Model (**DOM**)
 - Programming interface for HTML and XML documents
 - It represents the page
 - Programs can change the document structure, style, and content
 - It represents the document as nodes and objects
 - Object Oriented representation of the web page
 - To be modified with a language such as JavaScript (JS)
- Web page document
 - To be displayed in the browser window
 - As the HTML source

DOM

➤ Different versions

- DOM level 1, DOM level 2, DOM level 3, ... → DOM

➤ DOM specification

- Tells the structure of a document
- Gives objects to manipulate the document

➤ DOM: every HTML tag is an object

- Nested tags: children
- Text inside a tag: an object
- → All the objects can be accessed through Javascript

D, O, and M of DOM

- **D: Document**

- What happens when you load a file?
- Conversion of the file that is loaded
 - From an HTML file to an Object

- **O: Objects**

- With properties
- With methods

- **M: Model**

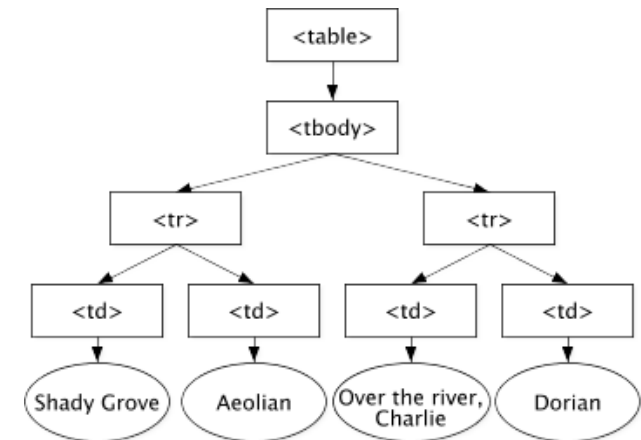
- Definition of what is a document
 - See week 1 (XML)

HTML DOM

- Representation of the document as a tree
 - Notion of node
 - The document is a document: a node
 - All the different HTML elements: nodes
 - All the attributes: attribute nodes
 - Piece of text in the HTML: text nodes
 - Comments: comment nodes

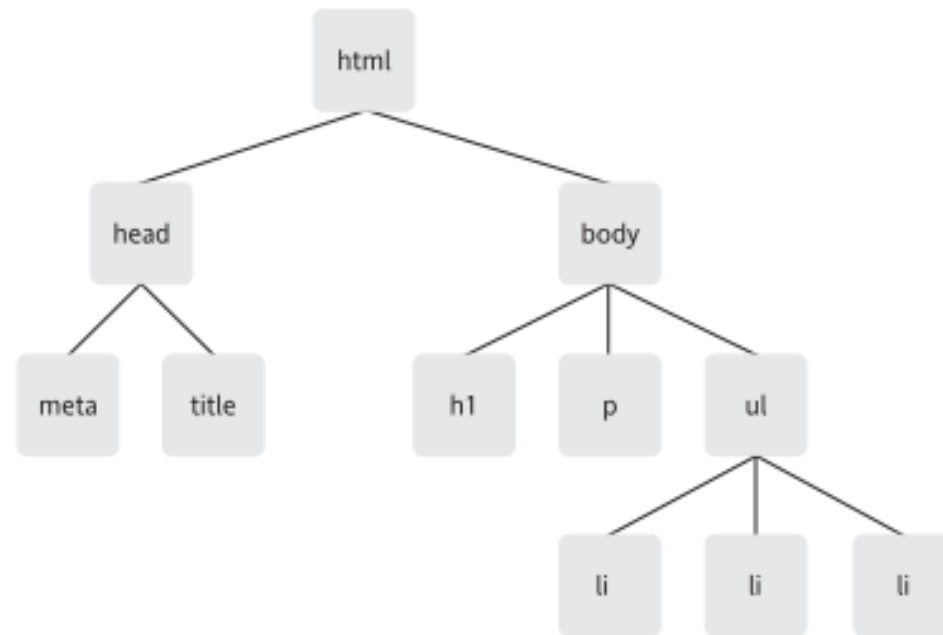
DOM Tree structure

- Tags
 - Element nodes (elements)
- Tree of elements
 - Html = root
 - (always the top tag, even if absent from the HTML file)
 - Head and Body = two children of the root ...
 - Text inside elements
 - Text nodes: #text
 - Comments in HTML
 - #comment → part of the DOM!
 - Tables have always a <tbody>



DOM Tree structure

- Example
 - Elements tree of a basic webpage



DOM

- Once you access the document object
 - Access to MANY properties/methods
 - Use the specifications of the DOM to know what you can access or not
 - Typical functions that you can obtain when you browse in a tree
 - Direct access to the main HTML blocks
 - You can modify
 - Direct update in the HTML page 😊
 - See the list of the methods (Canvas + many websites)
 - Some elements will be seen during the next classes for special functions

Exploring the tree

- DOM

- Allows you to reach the elements in the tree
- Start: document object
- `<html>` = `document.documentElement`
- `<body>` = `document.body`
- `<head>` = `document.head`

- Children

- `childNodes`
- `firstChild`
- `lastChild`

- Example

```
for (let i = 0; i < document.body.childNodes.length; i++) {  
    alert( document.body.childNodes[i] ); // Text, DIV, Text, UL, ..., SCRIPT  
}
```

InnerHTML vs. textContent

■ innerHTML

- The Element property innerHTML gets or sets the HTML or XML markup contained within the element.
- <https://developer.mozilla.org/en-US/docs/Web/API/Element/innerHTML>

■ textContent

- The textContent property of the Node interface represents the text content of the node and its descendants.
- <https://developer.mozilla.org/en-US/docs/Web/API/Node/textContent>

■ Differences

- **innerHTML** it parses content as HTML → it takes longer.
- **nodeValue** it uses straight text, does not parse HTML, and is faster.
- **textContent** it uses straight text, does not parse HTML, and is faster.
- **innerText** it takes styles into consideration. It won't get hidden text for instance.

Examples

- See files on Canvas
 - class_javascript_dom_01.html
 - class_javascript_dom_02.html
 - class_javascript_dom_03.html
 - class_javascript_dom_dynatable.html + js file

Conclusion

■ DOM

- Well ... everything is a node 😊
 - Document, HTML elements, attributes, text, comments
- Allows to get the document as an Object
 - To access its properties
 - To access its methods

■ Programming

- → We can do everything from JS to modify the tree
 - Creation of functions to automatize the creation of HTML pages generations