WEB TECHNOLOGY

1ST SEMESTER 2019

Document Object Model

Asst. Prof. Manop Phankokkruad, Ph.D.
Faculty of Information Technology,
King Mongkut's Institute of Technology Ladkrabang, Thailand

Outline

- ❑ Document Object Model
- ❑ HTML DOM
- ❑ DOM Programming Interface

What is the DOM?

Document Object Model (DOM) is a platform and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure, and style of a document.

The DOM is a W3C standard. The W3C DOM standard is separated into 3 different parts:

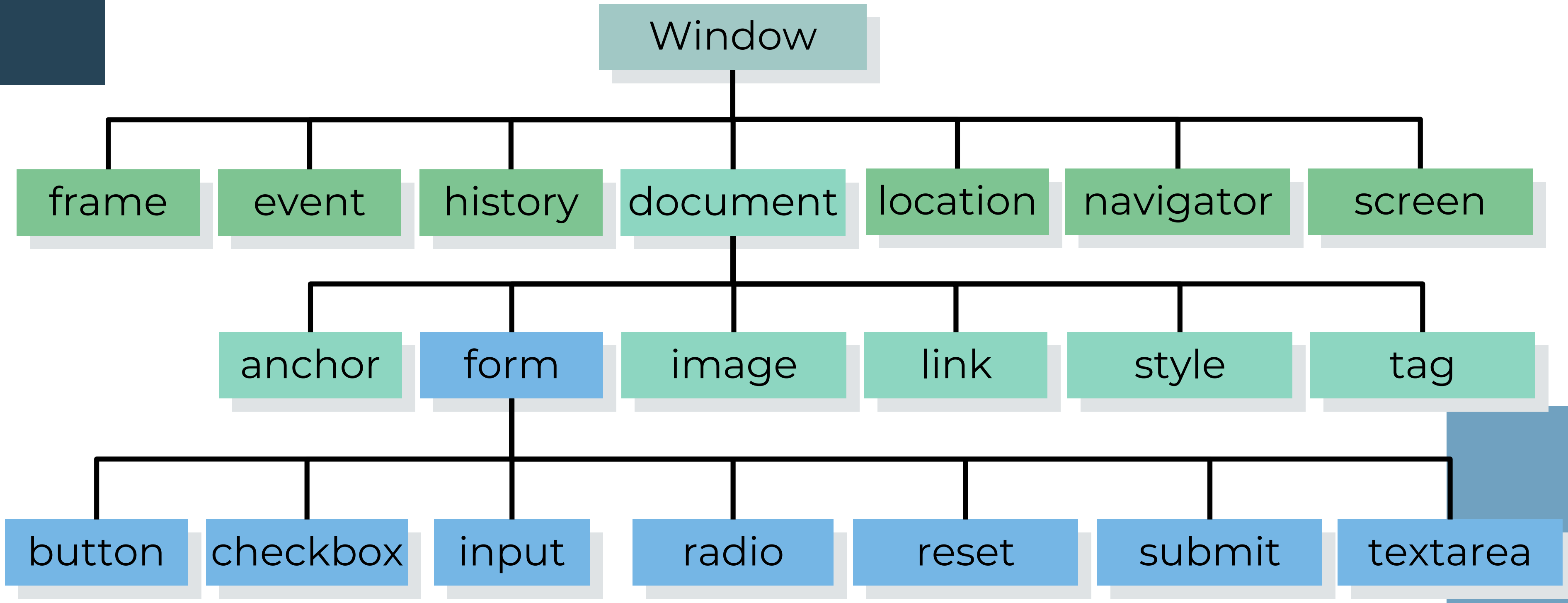
❑ **Core DOM** - standard model for all document types.

❑ **XML DOM** - standard model for XML

❑ **HTML DOM** - standard model for HTML

DOM hierarchy

DOM defines the logical structure of documents and the way a document is accessed and manipulated.



What is the DOM?

DOM gives we access to all the elements on a web page. Using JavaScript, we can create, modify and remove elements in the page dynamically. With the DOM,

❑ HTML elements can be treated as *objects*

❑ Many *attributes of HTML elements* can be treated as *properties* of those objects.

❑ Then, objects can be scripted (through their id attributes) with JavaScript to achieve dynamic effects.

What is the DOM?

- ❑ The web browser builds a *model* of the web page (the *document*) that includes all the *objects* in the page (tags, text, etc).
- ❑ All of the *properties*, *methods*, and *events* available to the web developer for manipulating and creating web pages are organized into objects.
- ❑ Those objects are accessible via scripting languages in modern web browsers.

DOM Levels

DOM Level 1

Interfaces for representing an XML and HTML document.

1. Document
2. Node
3. Attribute
4. Element
5. Text interfaces.

DOM Levels

DOM Level 2

It contains six different specifications:

1. DOM2 Core
2. Views
3. Events
4. Style
5. Traversal and Range
6. DOM2 HTML.

DOM Level 3

It contains five different specifications:

1. DOM3 Core
2. Load and Save
3. Validation
4. Events
5. XPath

HTML DOM

The HTML DOM is a standard object model and programming interface for HTML. It defines:

❑ The HTML elements as objects

❑ The properties of all HTML elements

❑ The methods to access all HTML elements

❑ The events for all HTML elements

HTML DOM Document

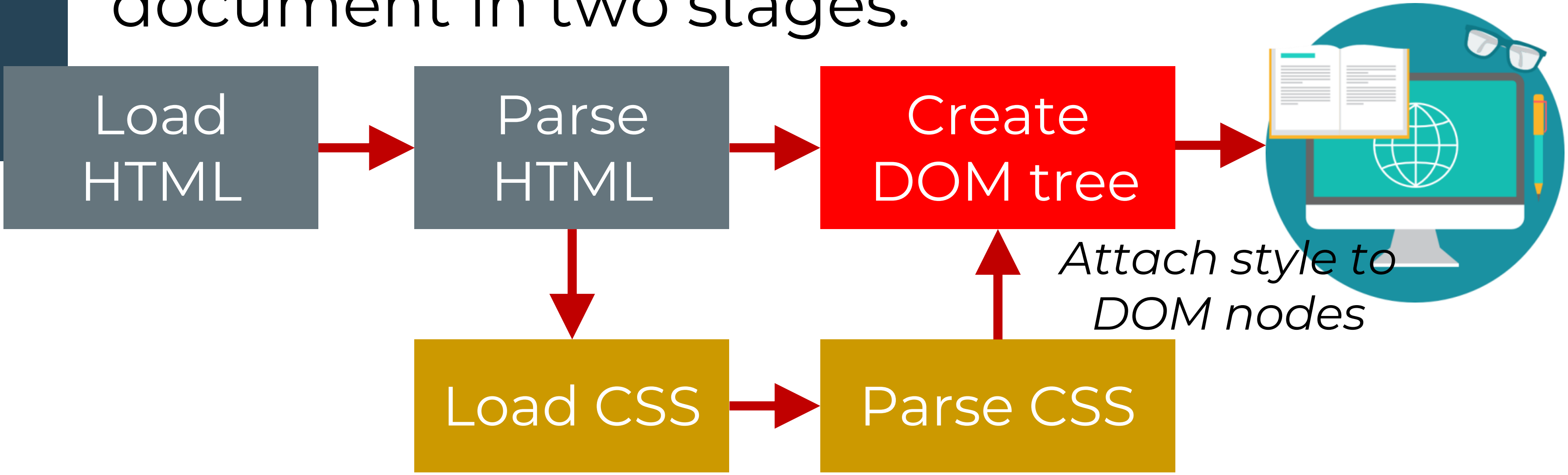
The HTML DOM document object is the owner of all other objects in the web page. So, we can use the document object to access and manipulate HTML document.

- Finding HTML Elements
- Changing HTML Elements
- Adding and Deleting Elements
- Adding Events Handlers
- Finding HTML Objects

❑ The model is made available to scripts running in the browser, not just the browser itself.

Browser and DOM

When a browser displays a document, it must combine the document's content with its style information. It processes the document in two stages.



Load HTML

Parse HTML

Create DOM tree

Load CSS

Parse CSS

Attach style to DOM nodes

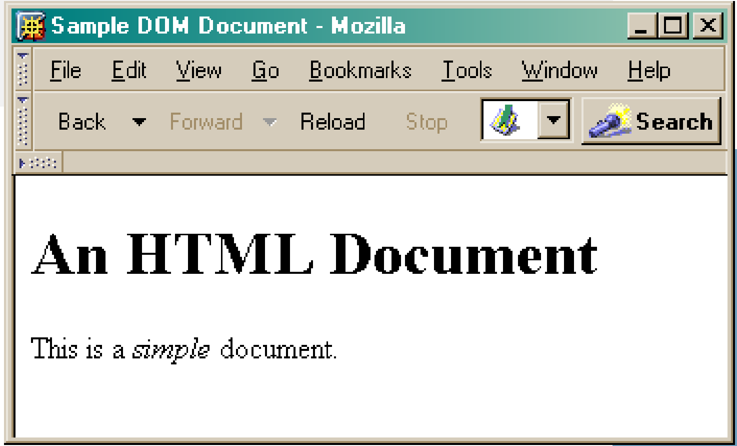
1. The browser converts HTML and CSS into the DOM.
2. The browser displays the contents of the DOM.

Browser and DOM

This is what the browser reads

```
<html>
  <head>
    <title>Sample DOM Document</title>
  </head>
  <body>
    <h1>An HTML Document</h1>
    <p>This is a <i>simple</i> document.</p>
  </body>
</html>
```

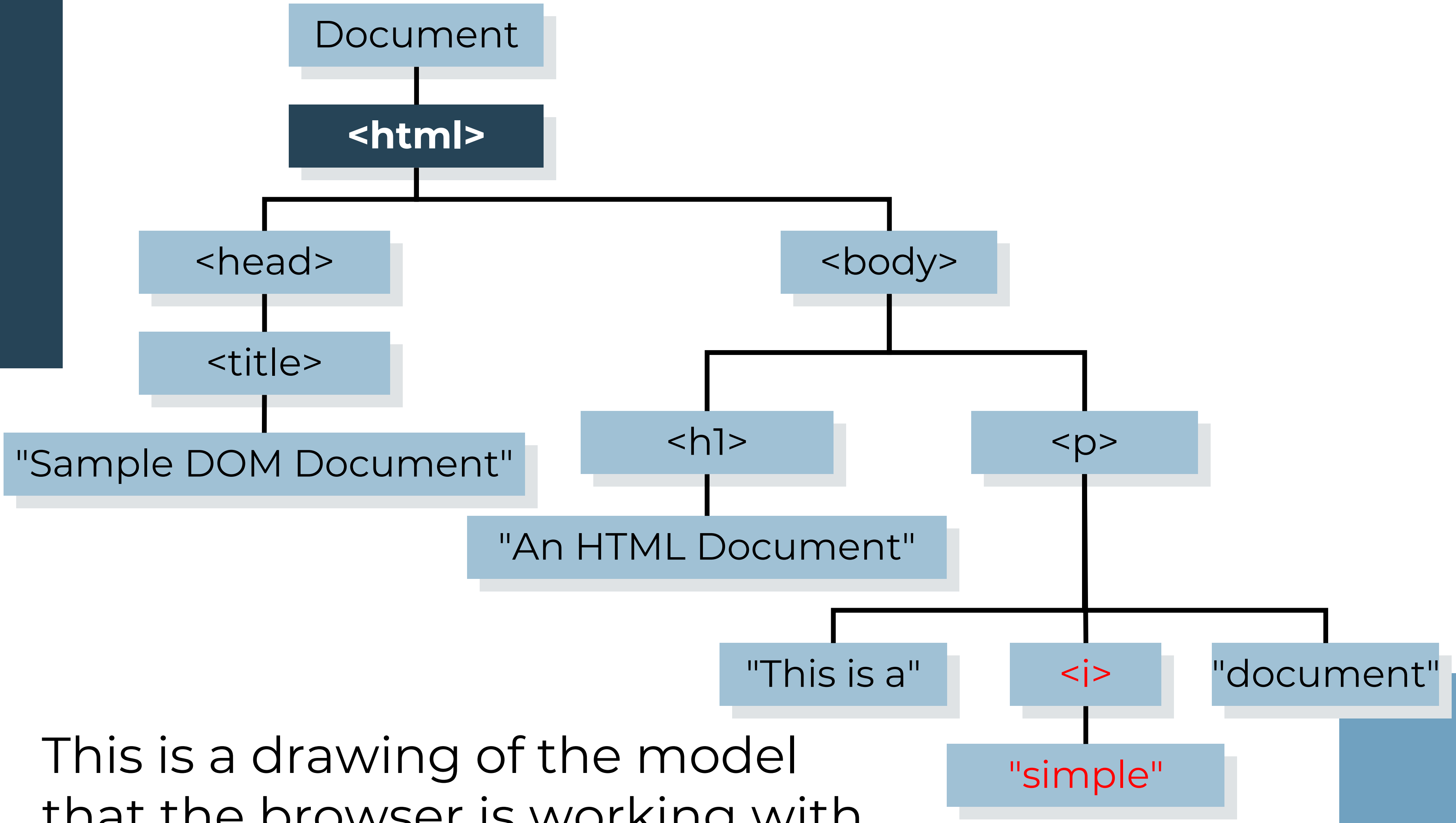
This is what the browser displays on screen.


Sample DOM Document - Mozilla
File Edit View Go Bookmarks Tools Window Help
Back Forward Reload Stop
Search

An HTML Document

This is a *simple* document.

Browser and DOM



This is a drawing of the model that the browser is working with for the page.

Types of DOM nodes

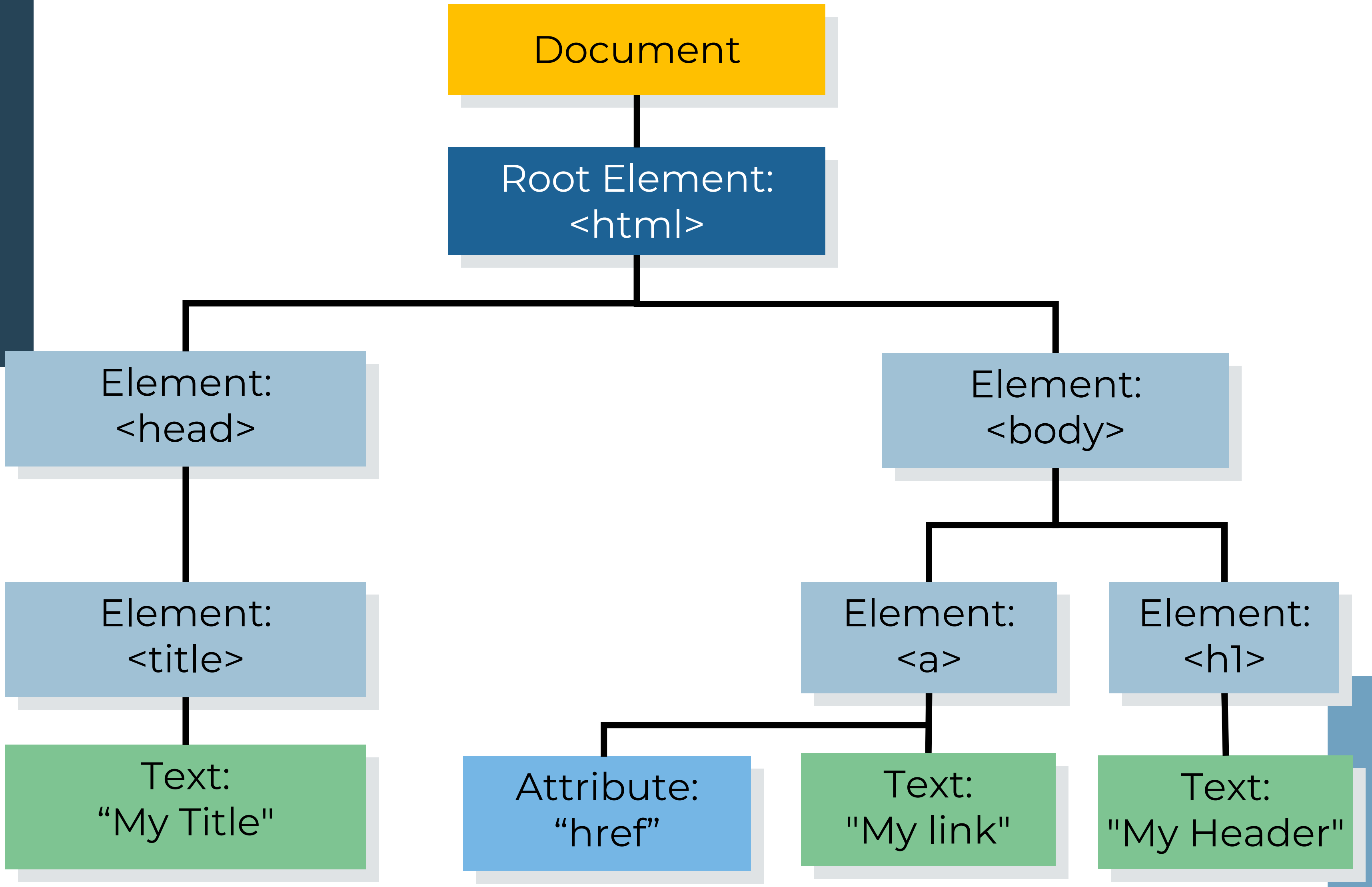
In the HTML DOM, everything is a node. The DOM represents documents as a hierarchy of Node objects. The main DOM node types are:

1. **Document node**
   - the start of the tree
2. **Element Node**
   - contains an HTML tag
   - can have element, text, and attribute child nodes.
3. **Attribute node**
   - Represents attribute of Element node.

Types of DOM nodes

4. **Text Node**
   - contains text / textual content of an element.
   - cannot have child nodes or attributes.
   - contained within Element Nodes.
5. **Comment**
   - an HTML comment
6. **DocumentType**
   - the Doctype declaration

DOM Tree Structure



Relationship among Nodes

- ❑ Every node has exactly one parent node (except root).
- ❑ A **parent node** can have one or more than one child nodes.
- ❑ **Root node** : The topmost node of the tree is the root node. As it is topmost, so there is no parent of this root node.
- ❑ **Leaf** : The leaf nodes are the nodes which have no child node.
- ❑ **Siblings** : The nodes which have same parent are the siblings of each other.

DOM Programming Interface

In the HTML DOM, all HTML elements are defined as objects. The HTML DOM can be accessed with JavaScript and other programming languages.

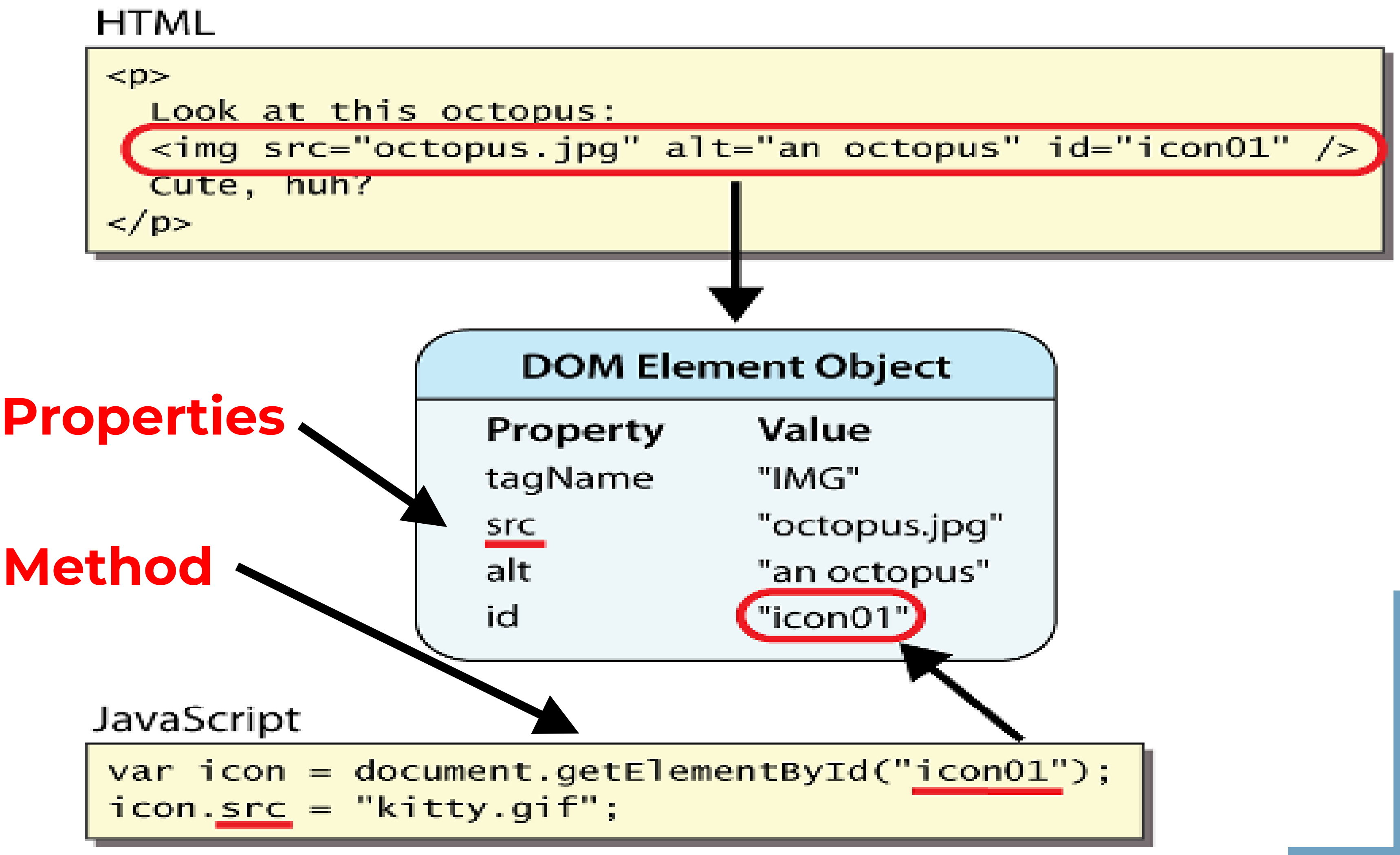
❑ The programming interface is the properties and methods of each object.

❑ A **property** is a value that one can get or set (like changing the content of an HTML element).

❑ A **method** is an action one can do (like adding or deleting an HTML element).

DOM Programming Interface

DOM Element Objects



HTML

```
<p>
  Look at this octopus:
  <img src="octopus.jpg" alt="an octopus" id="icon01" />
  Cute, huh?
</p>
```

DOM Element Object

Property	Value
tagName	"IMG"
src	"octopus.jpg"
alt	"an octopus"
id	"icon01"

Properties

Method

JavaScript

```
var icon = document.getElementById("icon01");
icon.src = "kitty.gif";
```

Programming Interface

Some commonly used HTML DOM Methods:

❑ getElementById(*id*) - get the node with a specified id

❑ appendChild(*node*) - insert a new child node

❑ removeChild(*node*) - remove a child node

The Node object

Properties:

- ❑ **className** - list of CSS classes of element
- ❑ **innerHTML** – text content inside element, including Tags.
- ❑ **parentNode** - the parent node of a node
- ❑ **firstChild** - first child of node
- ❑ **childNodes** - the child nodes of a node
- ❑ **attributes** - the attributes nodes of a node
- ❑ many more, some depending on type of node. These properties can be accessed and changed using JavaScript.

Programming Interface

HTML DOM - The innerHTML Property

```
<body>
<p id="intro">Hello World!</p>


<script>
let
txt=document.getElementById("intro").innerHTML;
document.write(txt);
</script>

</body>
```

Programming Interface

Getting the parent

Every element has just one parent. To get it, you can use Node.parentNode or Node.parentElement.

- ❑ parentNode returns the parent of the specified node in the DOM tree.
- ❑ parentElement returns the DOM node’s parent Element, or null if the node either has no parent, or its parent isn’t a DOM Element.

Programming Interface

Getting the children

❑ To check if a Node has child nodes, use Node.hasChildNodes() which returns a boolean value.

❑ To access all the Children Element Nodes of a node, use Node.childNodes.

❑ The DOM also exposes a Node.children method.

❑ However, it not just includes Element nodes, but also the white space between elements as Text nodes.

Programming Interface

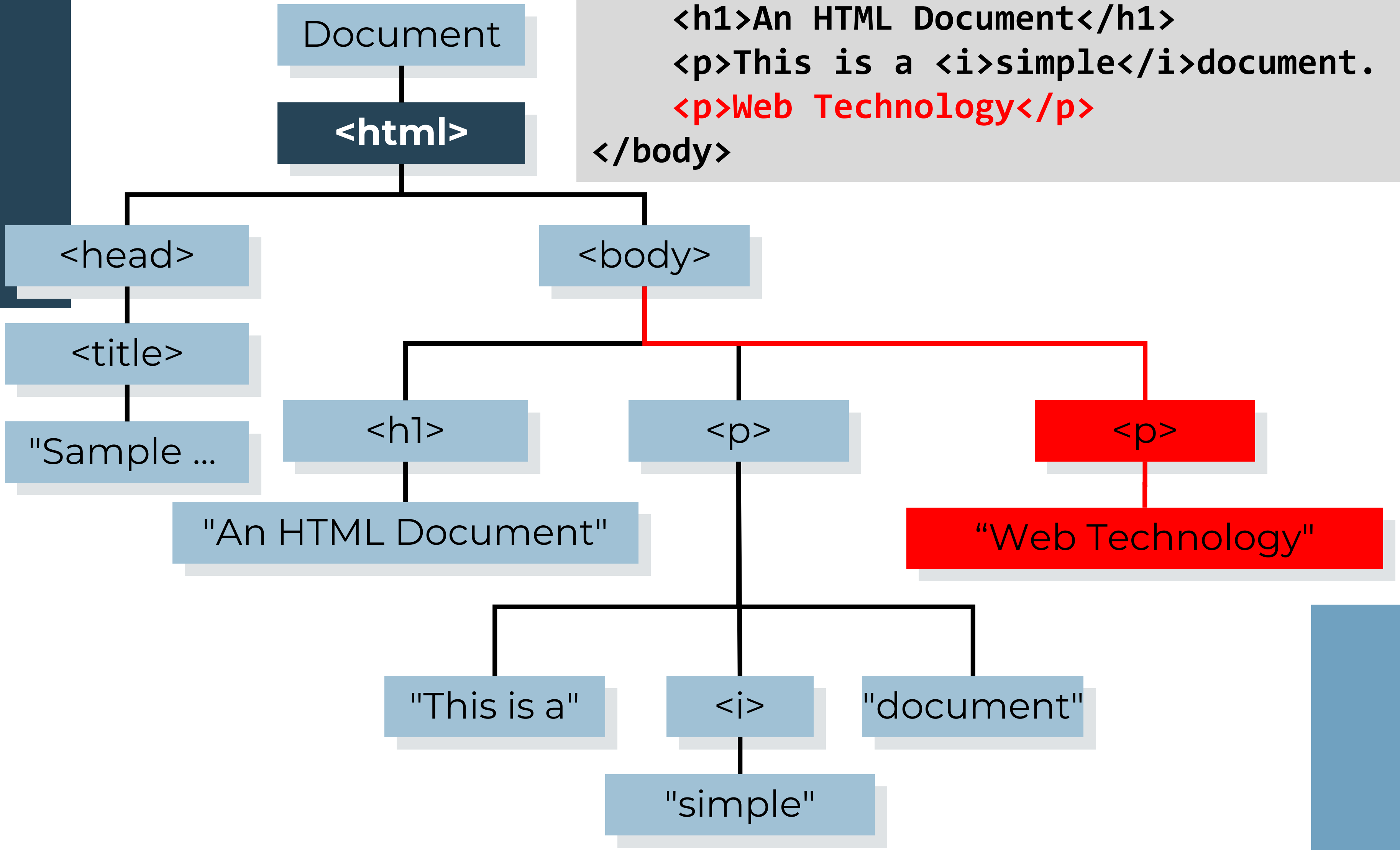
Creating New HTML Elements (Nodes)

To add a new element to the HTML DOM, we must create the element (element node) first, and then append it to an existing element.

```
<html>
  <head>
    <title>Sample DOM Document</title>
  </head>
  <body>
    <h1>An HTML Document</h1>
    <p>This is a <i>simple</i> document.</p>
    <p>Web Technology</p>
  </body>
</html>
```

Programming Interface

```
<body>
    <h1>An HTML Document</h1>
    <p>This is a <i>simple</i>document.
    <p>Web Technology</p>
</body>
```



Programming Interface

Finding HTML Elements

When, you want to manipulate HTML elements. You must find the elements first. There are a couple of ways to find the HTML elements by id, tag name, class name, CSS selectors, HTML object collections.

❑ Finding by **id**

```
let myElement = document.getElementById("intro");
```

❑ Finding by Tag **Name**

```
let x = document.getElementsByTagName("p");
```

Programming Interface

❑ Finding by **Class Name**

```
let x = document.getElementsByClassName("intro");
```

❑ Finding by **CSS**

```
let x = document.querySelectorAll("p.intro");
```

returns a list of all <p> elements with class="intro".

❑ Finding by **HTML Object Collections**

```
let x = document.forms["frm1"];
let text = "";
let i;
for (i = 0; i < x.length; i++) {
    text += x.elements[i].value + "<br>";
}
```

Programming Interface

❑ Modifying HTML content

```
<p id="p1">Hello World!</p>
<script>
document.getElementById("p1").innerHTML = "New
text!";
</script>
```

❑ Changing the Value of an Attribute

```
<img id="myImage" src="smiley.jpg">
<script>
document.getElementById("myImage").src =
"new.jpg";
</script>
```

Programming Interface

❑ Changing HTML Style

```
<p id="p2">Hello World!</p>
<script>
document.getElementById("p2").style.color =
"blue";
</script>
```

❑ Using Events

```
<h1 id="id1">My Heading 1</h1>
<button type="button"
onclick="document.getElementById('id1').style.col
or = 'red’”> Click Me!</button>
```

More Information

❑ JavaScript Tutorial
https://www.w3schools.com/js/default.asp

❑ XML DOM Tutorial
https://www.w3schools.com/xml/dom_intro.asp

❑ XML DOM Tutorial
https://www.tutorialspoint.com/dom/