COMP335
Web Application Development

JavaScript

Introduction

- History
  - Originally developed by Netscape
  - Joint Development with Sun Microsystems in 1995
  - JavaScript was influenced by many languages and was designed to look like Java
  - JavaScript was a trademark of Sun Microsystems, now is a trademark of Oracle.
- HTML to define the **content** of web pages
- CSS to specify the **styles** of web pages
- JavaScript to program the **behavior** of web pages

Where does JavaScript go?

```
<body>

<h1> Hello fellow Web programmers! </h1>
<h1> Don't forget to submit assignment!</h1>
<button type="button" onclick="foo()">Click me</button>


<script >                                   Embedded
  function foo(){
    document.getElementsByTagName("h1")[0].innerHTML="Hello World!";
    window.alert ("Hello World!");
  }
</script>

<script  src="basic.js"></script>          External

</body>
```

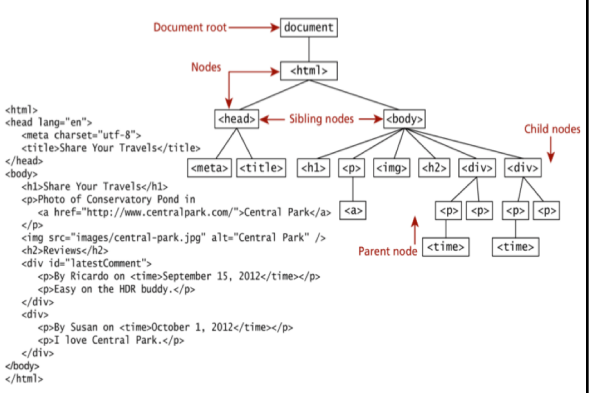
At the end of body tag. Why?

Example: basic.html

Document Object Model (DOM)

- JavaScript is **almost always** used to interact with the HTML document through Document Object Model.
- According to the W3C, the DOM is a:

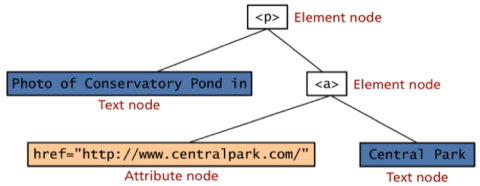
  Platform- and language-neutral interface that will allow programs and scripts to dynamically access and update the content, structure and style of documents.



DOM Nodes

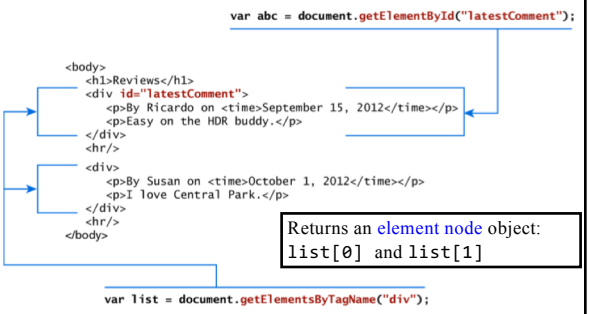
- element nodes
- text nodes
- attribute nodes

```
<p>Photo of Conservatory Pond in
   <a href="http://www.centralpark.com/">Central Park</a>
</p>
```



- The `document` object is the root JavaScript object representing the entire HTML document.

Accessing nodes



Example: node0.html

Modify a DOM element

- Using the DOM document and HTML DOM element objects, we can do exactly that using the `innerHTML` property.
- Although the `innerHTML` technique works well (and is very fast), there is a more verbose technique available to us that builds output using the DOM.
  - DOM functions `createTextNode()`, `removeChild()`, and `appendChild()` allow us to modify an element in a more **rigorous** way

Example: node1.html

Think aloud exercise

- Changing an element's style
  - Open your browser and go to http://www.bridge.edu
  - get all nodes and change its font color to "blue"

Example: BSU homepage

Think aloud exercise

- Changing an element's style
  - Open your browser and go to http://www.bridge.edu
  - get all nodes and change its font color to "blue"

```
> function foo(){
    var tags = document.getElementsByTagName("strong");
    for(each in tags){  //return keys, not values
      if (tags[each].style) { // as long as CSS style exists
         tags[each].style.color = "blue";
      }
    }
  }
<- undefined
> foo()
```

Example: BSU homepage

JavaScript Events

- A JavaScript event is an action that can be detected by JavaScript.
- We say that an event is *triggered* and it can be *caught* by JavaScript functions, which then do something in response
- Examples of HTML events:
  - https://www.w3schools.com/jsref/dom_obj_event.asp

Event Listener

```
element.addEventListener(event, function, useCapture);
```

- The first parameter: the type of the event (e.g., click).
- The second parameter: the function we want to call when the event occurs.
- The third parameter (optional): a Boolean value specifying whether to use event bubbling or event capturing
  - The default value is false, which will use the bubbling propagation.
  - In bubbling the inner most element's event is handled first and then the outer. In capturing the outer most element's event is handled first and then the inner.

```
var greetingBox = document.getElementById('example1');
greetingBox.onclick = alert('Good Morning');
```

The "original" style of registering a listener

In the original JavaScript world, events could be specified right in the HTML markup with *hooks* to the JavaScript code (and still can).

```
var greetingBox = document.getElementById('example1');
greetingBox.addEventListener('click', alert('Good Morning'));
greetingBox.addEventListener('mouseOut', alert('Goodbye'));
```

The "newer" DOM2 approach to registering listeners

When using the `addEventListener()` method, the JavaScript is separated from the HTML markup, for better readability and allows you to add event listeners even when you do not control the HTML markup. You can easily remove an event listener by using the `removeEventListener()` method.

example: listener.html

Validating Forms

- Writing code to pre-validate forms on the client side will reduce the number of incorrect submissions, thereby reducing server load.
- There are a number of common validation activities including email validation, number validation, and data validation.

What does this code do?

```
<form>
  <input type="text" id="name" placeholder="last, first name ">

  <input type="text" id="phone" placeholder="Phone No. (ddd-ddd-
   dddd)">

  <input type = "reset"  id = "reset" >
  <input type = "button" value="submit" id = "submit" >

</form>

<script>
document.getElementById("name").addEventListener('change',chkName);
document.getElementById("phone").addEventListener('change',chkPhone);
</script>
```

The `change` is used when the content of a field changes

15

RegExp

- A regular expression is a sequence of characters that forms a search pattern.
- In JavaScript, regular expressions are often used with method **`search()`**
- The `search()` method uses an expression to search for a match, and returns the position of the match.

- https://www.w3schools.com/js/js_regexp.asp
- https://www.w3schools.com/jsref/jsref_obj_regexp.asp

- Metacharacters are characters with a special meaning (\)
- Brackets are used to find a range of characters: ([ ])
- Quantifiers define quantities (+)

- **`[0-9]`** Find any of the digits between the brackets
- **`n{X}`** Matches any string that contains a sequence of X n's

- **`\d`** Find a digit
- **`n+`** Matches any string that contains at least one n
- **`n$`** Matches any string with n at the end of it
- **`^n`** Matches any string with n at the beginning of it

What does this code do?

```
function chkPhone() {
  var myPhone = document.getElementById("phone");
  // -1 if no match was found
  var pos = myPhone.value.search(/^\d{3}-\d{3}-\d{4}$/);

  if (pos == -1) {
    alert("The phone number you entered (" + myPhone.value +
          ") is not in the correct form. \n" +
          "The correct form is: ddd-ddd-dddd \n" +
          "Please go back and fix your phone number");

    return false;
  }
  else
    return true;
}
```

To keep the form active after the event handler is finished, the handler must return false

18

example: forms.html

What does this code do?

```
function chkName() {
  var myName = document.getElementById("custName");
  // -1 if no match was found
  var pos = myName.value.search(
      /^[A-Z][a-z]+, [A-Z][a-z]+$/);

  if (pos == -1) {
    alert("The name you entered (" + myName.value +
          ") is not in the correct form. \n" +
          "The correct form is: " +
          "Last-name, First-name \n" +
          "Please go back and fix your name");

    return false;
  }
  else
    return true;
}
```

To keep the form active after the event handler is finished, the handler must return false

Validating Forms

example: forms.html

To access contents (or value): `id.value`
for example, `phone.value`
`name.value`

Exercise

- Modify "exercise.html" to validate inputs