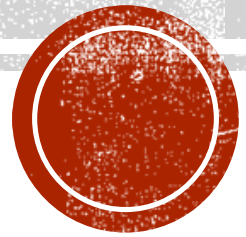
JAVASCRIPT

Andrew Beatty

Andrew.Beatty@gmit.ie

Data Representation



OVERVIEW

- What you will be able to do: Lab
- CSS:
  - Hidden, block and inline display
  - Disabled
- JavaScript (W3Schools)
  - Overview of the language
  - Dom manipulation
  - Attributes
  - Setting values and innerHTML
    - Theory
    - Exercise
    - Demonstration

      (Rinse and Repeat)

CSS (CASCADING STYLE SHEETS)

- Too much for this course
- Style can be defined:
  1. In another file (with selectors)
  2. In the head (with selectors)
  3. In the element itself
- Make an element visible/hidden

```
<div style="display: block">
    text
</div>
```

Can be:
- block (new line)
- Inline (no new line)
- none (not shown)

- Put the keyword disabled in an element to make it disabled

```
<input type="text" value="blah" disabled />
```

User will not be able to enter text into this input

Exercise 2.1:

Write some html that has a hidden element

```
<html>
  <head>
    <title>2.1</title>
  </head>
  <body>
    <div style="display: none">hi mom</div>
    <div style="display: block">div1</div>
    <div style="display: inline">div2</div>
    <div style="display: inline">div3</div>
  </body>
</html>
```

JAVASCRIPT

- Inside <script></script> tags

```
<script>
        console.log("hello World")
</script>
```

Prints to the console, not the page

- Blocks have {} (not indents like in python)

Name of function

```
<script>
        function sayHelloAgain(){
          console.log("hello Again")
        }
        sayHelloAgain()

</script>
```

Define block of function

Calls function

```
<html>
  <head>
    <title>e2.2</title>
    <!-- javascript can go in the head-->
  </head>
  <body>
    hello
    <script>
      console.log("hello world2")

      function sayHelloAgain(message){
        console.log("hello "+message);
      }
      sayHelloAgain('Andrew')
    </script>
  </body>
</html>
```

JAVASCRIPT TO/FROM HTML

- Onclick attribute

```
<button onclick="myFunction('hello')">click me</button>
```

- Document getElementById

```
<div id="messageOut"></div>
```

```
document.getElementById('messageOut').innerText = message
```

Exercise 2.3:

Make a webpage with an input and a button, when the user clicks the button then the contents of the input will display in another div

```
<html>
  <head>
    <title>e2.3</title>

  </head>
  <body>
    <input id="name" type="text" value="enter name"/>
    <br/>
    <button onclick="buttonClicked()">click me</button><br/>
    <div id="output">output will go here</div>

    <script>
      function buttonClicked(){
        var output= document.getElementById('name').value
        document.getElementById('output').innerText = output
      }

    </script>
  </body>
</html>
```

VARIABLES

- Define a variable with the var keyword, eg var age= 21
- Variable types are defined by there value, they can be
  - Integer
  - Float
  - String
  - Boolean
  - Null
  - undefined
  - Object
  - Array
  - function

```
var i = 12
var fl = 3.3
var firstName   = 'andrew'
var lastName    = "O'Neill" //or 'O\'Neill'
var admin = true
var foo = null
var dunno // = undefined
var car = {}
var books = []
var fun = function(){console.log("in fun")}
```

OBJECTS

- Properties can be added to objects

```
car.reg =‘12 mo 1234’
car.make ="ford"
```

- Or defined using JSON

```
var employee = { name: ‘Joe’, role:’Manager’ }
```

- Use JSON.stringify() to output

```
console.log(“car "+ JSON.stringify(car))
```

Exercise 2.4

Create a book object as a variable with title, author and ISBN, and display it in the console

```
<html>
  <head>
    <title>2.4</title>
  </head>
  <body>
    <script>
      var book ={}
      book.title = "some harry potter stuff"
      book.author = "JK"
      book.isbn = "1234"

      console.log("book is "+ JSON.stringify( book) )
    </script>
  </body>
</html>
```

MANIPULATE DISPLAY

- We can make am element visible and hidden

```
document.getElementById("div1").style.display = 'block'
document.getElementById("div2").style.display = 'none'
```

- Or we can disable a button

```
document.getElementById(“button1").disabled = true
```

Exercise 2.5:

Make a webpage that has two , only one should be visible at a time, each will have a button that will make the other div visible (and hide itself)

```
<html>
    <body>
      <div id="div1" style="background-color: royalblue”>
            some text <br/>
            <button onclick="showDiv2()">show Div2</button>
      </div>
      <div id="div2" style="background-color: red; display: none">
            <br/>
            div2 <br/>
            <button onclick="showDiv1()">show Div1</button>
      </div>
      <script>
            function showDiv2(){
               document.getElementById("div1").style.display = "none";
               document.getElementById("div2").style.display = "block";
            }
            function showDiv1(){
               document.getElementById("div1").style.display = "block";
               document.getElementById("div2").style.display = "none";
            }
      </script>
  </body>
</html>
```

CONTROL

- if statement
  - if (condition){
    do stuff
    }else{
    do other stuff
    }

```
if (value < 10){
    outputElement.innerText = 'too low'
}else if (value > 20){
        outputElement.innerText = 'too high'
}else{
    outputElement.innerText = 'good enough'

}
```

Exercise 2.6:

Make a web page that has a text box, when the content of the text area changes the page will display whether the value entered is greater or less than 10, or equal to 10

```
<html>
    <body>
    <input type="number" onchange="checkNumber(this)"/>
    <br/>
    <div id="output">
       output here
    </div>
    <script>
       function checkNumber(inputElement){
          var value = inputElement.value
          if( value <10){
             document.getElementById("output").innerText = "too low"
          }else if (value > 10){
             document.getElementById("output").innerText = "too high"
          }else{
             document.getElementById("output").innerText = "just right"
          }

       }
    </script>
  </body>
</html>
```

FOR LOOP

- Simple count

```
for (var i = 0; i< 10; i++){
  console.log(i);
}
```

- Iterate an array

```
var names = ['joe','Mary', 'Fred']
for (name of names){
   output = output + ' ' + name
}
```

- Iterate an object

```
var book = {title:'harry Potter and something',author:'JKR', isbn:"12345"}
for (propName in book){
   bookoutput += propName+'='+book[propName]+'<br/>'

}
```

Excersise 2.7:

1. Make a web page that outputs 1 to 10 (or N)
2. Make a web page that outputs all the values of an array

```
<body>
    <div id ="output">
        output here
    </div>


    <script>
        for (var i=0; i<10; i++){
            console.log(i)
        }
        var names=['Joe','Mary','Fred']
        var output =''
        for (name of names){
            output += name + '<br/>'
        }
        document.getElementById("output").innerHTML = output

    </script>
  </body>
```

DOM

- Get Element By id, we have seen this already
- QuerySelector: The querySelector function searches all the child nodes of a particular element for nodes that match the query, and returns the first one (it is like CSS selectors and JQuery selectors, see later).
  - Name — name of tag
  - #id — a node with id="id"
  - .classname — A node with class="classname"
  - [atName="atVal"] — A node with the attribute atName="atValue"

```
form.querySelector('#anId')
```

- More data at https://www.w3schools.com/cssref/css_selectors.asp

Exercise 2.8:

Create a webpage with multiple inputs and a button, when the user hits the button read all the inputs, store the values into a class and output the class to the console

```
<body>
    <div id="myForm">
       First name: <input type="text" name='firstName'/><br/>
       Last name: <input type="text" name='lastName'/><br/>
       age: <input type="text" name='age'/><br/>
       <button onclick="doForm()">go</button>
    </div>

    <script>
     function doForm(){
        var employee ={}
        var form = document.getElementById("myForm")
        employee.firstName = form.querySelector('input[name="firstName"').value
        employee.lastName = form.querySelector('input[name="lastName"').value
        employee.age = form.querySelector('input[name="age"').value
        console.log("form data "+JSON.stringify(employee))
     }

    </script>
  </body>
```

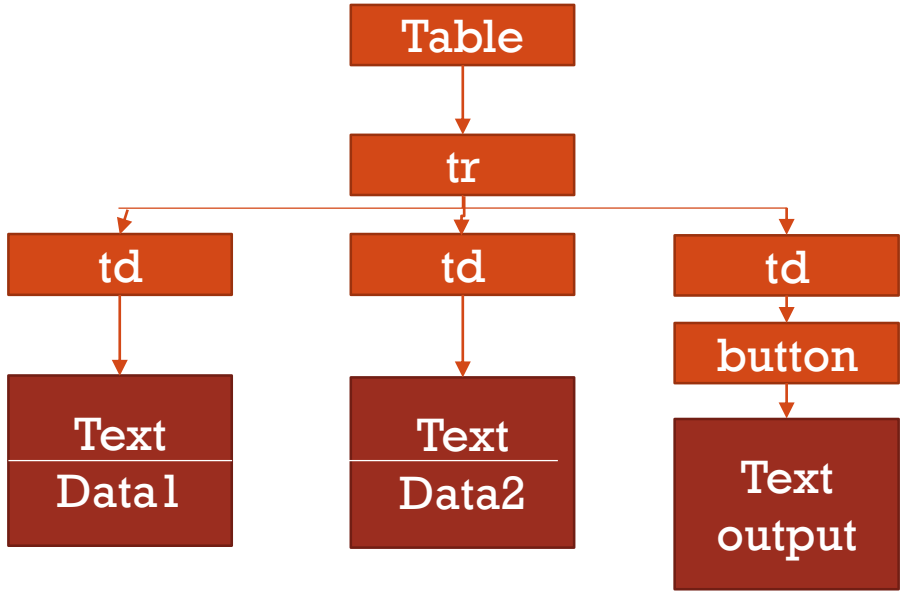
CHILD NODES

- Consider

```
<table>
        <tr>
          <td>data1</td>
          <td>data2</td>
          <td><button onclick="doRow(this)">output</button></td>
        </tr>
 </table>
```

To get the content of the first cell, you need to get the child of that cell, and its text value

```
rowElement.cells[0].firstChild.textContent
```



PARENT NODES

- Consider the previous slide
- What is the relationship between the row element and the button?
- <Td> is the parent and <tr> is the <td>s parent
- So we can access the tr by:

```
var  rowElement  = buttonElement.parentNode.parentNode
//or
Var rowElement = buttonElement.closest('tr')
```

Exercise 2.9

Write the code so that when the user clicks on the button in a row, the contents of the cells in that row are stored in an object and outputted to the console.

```
<body>
    <table>
      <tr>
        <td>how now</td>
        <td>brown cow</td>
        <td><button onclick="doRow(this)">output</button></td>
      </tr>
    </table>

    <script>
      function doRow(buttonElement){
        var rowElement = buttonElement.parentNode.parentNode
        var data={}
        data.cell1 = rowElement.cells[0].firstChild.textContent;
        data.cell2 = rowElement.cells[1].firstChild.textContent;

        console.log(JSON.stringify(data))

      }

    </script>
  </body>
```

ADD TO DOM TREE

- Add element

```
var para = document.createElement("p");
var node = document.createTextNode("This is new.");
para.appendChild(node);
```

- Add row

```
var rowElement = tableElement.insertRow(-1)
```

- Add cell

```
var cell1 = rowElement.insertCell(0);
```

- Add attribute

```
rowElement.setAttribute('id', car.reg)
```

Exercise 2.9.2

Create a button that when it is clicked adds a row to the table (above) with data

```
function createRow(){
        var tableElement = document.getElementById('mainTable')
        var data = {cell1:'hi',cell2:'bye',cell3:'thanks for the fish'}

        var rowElement = tableElement.insertRow(-1)
        var cellElement = rowElement.insertCell(0)
        cellElement.innerHTML = data.cell1;
        cellElement = rowElement.insertCell(1)
        cellElement.innerHTML = data.cell2;
        cellElement = rowElement.insertCell(2)
        cellElement.innerHTML = data.cell3;

    }
```

CONCLUSION