

3

Intro to JavaScript



3

Review of last lecture (HTML+CSS)

- `<head>`
 - `<title>` `<meta>` `<link>` `<script>`
- `<body>`
 - `<p>``
``<hr>````````<h1-6>`
`<div>``````<a>` semantic `<header>``<footer>`
 - `<form>`: `<input>``<select>``<textarea>` ...
- Lots more tags
 - `<pre>` `<code>` `<blockquote>`
 - `<table>`: `<th>` `<tr>` `<td>`
 - Semantic `<aside>` `<nav>` `<article>` `<section>`



4

javadoc

```
/**
 * <h1>Hello, World!</h1>
 * The HelloWorld program implements an application that
 * simply displays "Hello World!" to the standard output.
 * <p>
 * Giving proper comments in your program makes it more
 * user friendly and it is assumed as a high quality code.
 *
 *
 * @author Zara Ali
 * @version 1.0
 * @since 2014-03-31
 */
public class HelloWorld {

    public static void main(String[] args) {
        // Prints Hello, World! on standard output.
        System.out.println("Hello World!");
    }
}
```

For your information



5

```
/**
 * This class contains various methods for manipulating arrays (such as
 * sorting and searching). This class also contains a static factory
 * that allows arrays to be viewed as lists.
 *
 * <p>The methods in this class all throw a {@code NullPointerException},
 * if the specified array reference is null, except where noted.
 *
 * <p>The documentation for the methods contained in this class includes
 * briefs description of the <i>implementations</i>. Such descriptions should
 * be regarded as <i>implementation notes</i>, rather than parts of the
 * <i>specification</i>. Implementors should feel free to substitute other
 * algorithms, so long as the specification itself is adhered to. (For
 * example, the algorithm used by {@code sort(Object[])} does not have to be
 * a MergeSort, but it does have to be <i>stable</i>.)
 *
 * <p>This class is a member of the
 * <a href="{@docRoot}/../technotes/guides/collections/index.html">
 * Java Collections Framework</a>.
 *
 * @author Josh Bloch
 * @author Neal Gafter
 * @author John Rose
 * @since 1.2
 */
public class Arrays {

    /**
     * The minimum array length below
     * which the algorithm will not further partition
     * smaller sizes typically result in
     * tasks that makes parallel processing
     * less effective.
     */
    private static final int MIN_ARR
```

Overview Package **Class** Use Tree Deprecated Index Help

Prev Class Next Class Frames No Frames All Classes

Summary: Nested | Field | Constr | Method Detail: Field | Constr | Method

java.util

Class Arrays

java.lang.Object
java.util.Arrays

public class Arrays
extends Object

This class contains various methods for manipulating arrays (such as sorting and searching). This class also contains a static factory that allows arrays to be viewed as lists. The methods in this class all throw a `NullPointerException`, if the specified array reference is null, except where noted.

The documentation for the methods contained in this class includes briefs description of the *implementations*. Such descriptions should be regarded as *implementation notes*, rather than parts of the *specification*. Implementors should feel free to substitute other algorithms, so long as the specification itself is adhered to. (For example, the algorithm used by `sort(Object[])` does not have to be a MergeSort, but it does have to be *stable*.)

This class is a member of the Java Collections Framework.

Since:

1.2

For your information



6

Sections of a page <div>

```
<div class="shout">
<h2>Coding Horror! Coding Horror!</h2>
<p> See our special deal on Droids!</p>
<p>We'll beat any advertised price!</p>
</div>
```

HTML

Coding Horror! Coding Horror!

See our special deal on Droids!

We'll beat any advertised price!

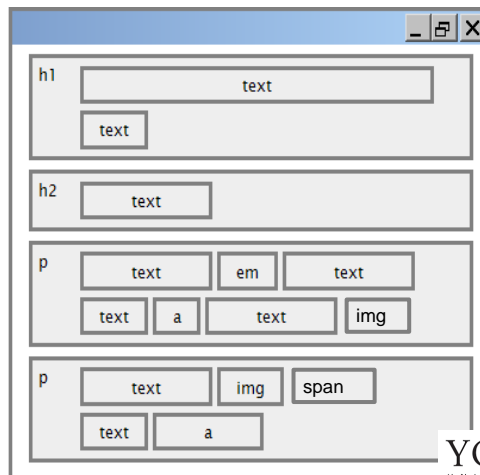
output

- ❑ Tag used to indicate a logical section or area of a page
- ❑ Has **no appearance by default**, but you can apply styles to it
- ❑ HTML5 semantic tags <header> <footer> <section>

7

Document flow - a larger example

- Both block and inline elements can be nested inside block elements;
- Inline element can be nested inside block and inline element;
- Block element *cannot* be nested inside an inline element;



8

Inline CSS

An inline CSS is used to apply a unique style to a single HTML element.
An inline CSS uses the style attribute of an HTML element.
This example sets the text color of the `<h1>` element to blue:

```
<h1 style="color:blue;">This is a Blue Heading</h1>
```

Internal (embedded) CSS

An internal CSS is used to define a style for a single HTML page.
An internal CSS is defined in the `<head>` section of an HTML page, within a `<style>` element:

```
<html>
<head>
<style>
body {background-color: powderblue;}
h1 {color: blue;}
p {color: red;}
</style>
</head>
<body>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
```

External CSS

```
<html>
<head>
<link rel="stylesheet" href="styles.css">
</head>
<body>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
```

9

selectors

```
selector {
  property: value;
}
```

RECALL

□ single

```
p {
  background-color: grey;
}
```

□ group

```
h1, p {
  background-color: ivory;
}
```

□ class

```
.class1 { font-style: italic; color: red;}
```

```
...
<div class="class1">...</div>
...
<div class="class1">...</div>
```

□ id

```
#s123 { font-style: italic; color: red;}
```

```
<p id="s123">
```

□ pseudo classes

```
a:hover {color: pink;}
```

combinators

- descendant selector (space)
selector 1 selector2

10

Pseudo-class selectors

class	description
:active	an activated or selected element
:focus	an element that has the keyboard focus
:hover	an element that has the mouse over it
:link	a link that has not been visited
:visited	a link that has already been visited

```

a:link { color: blue; }      /* unvisited link */
a:visited { color: rgb(255, 255, 0); } /* visited link */
a:hover { color: yellow; }   /* mouse over link */

p:hover {color: green;}

```

CSS

11

CSS context selectors

```

selector1 selector2 {
  properties
}

```

CSS

- applies the given properties to selector2 only if it is **inside** a selector1 on the page

```

<header>    <!-- semantic -->
<h1> Reading Diary </h1>
<h2> The books I've read recently and my (and Wikipedia's) thoughts about them</h2>
</header>

```

Reading Diary

The books I've read recently and my (and Wikipedia's) thoughts about them

```

header h2 {border-style: solid; border-color: orange;}

```

12

CSS context selectors

```
selector1 selector2 {
  properties
}
```

CSS

- applies the given properties to selector2 only if it is **inside** a selector1 on the page

```
<div class="entry">
  <h2> The Handmaid's Tale <span>by Margaret Atwood</span></h2>
  .....
</div>
```

The Handmaid's Tale by Margaret Atwood

After a staged attack that killed the President of the United States and most of Congress, a radical political group call States Constitution was suspended, newspapers were censored, and what was formerly the United States of America

```
div span {border-style: solid; border-color: cyan;}
.entry span {border-style: solid; border-color: cyan;}
```

YORK
UNIVERSITY

13

Context selector example

```
<h2>Eat at <span>Greasy's Burger</span>...</h2>
<p> The <span>greasiest</span> burgers in town! </p>
<h4>Yummy and greasy at the same time!</h4>
</ul>
```

HTML

```
p span { text-decoration: underline; }
```

CSS

Eat at Greasy's Burger...

The greasiest burgers in town!

Yummy and greasy at the same time!

output

p, span



YORK
UNIVERSITY

16

Put these topics together... try it out

```

<body>
  <!-- show id -->
  <p>Coding Horror! Coding Horror!</p>
  <p id="mission">Our mission is to combine programming and <q>human</q> fact
  </p>

  <hr> <!-- show class -->
  <p class="shout">Coding Horror! Coding Horror!</p>
  <h2 class="special">See our special deal on Droids!</h2>
  <p class="special">Today only!</p>

  <hr> <!-- show div (with class)-->
  <div class="shoutDIV">
    <h2>Coding Horror! Coding Horror!</h2>
    <p>See our special deal on Droids!</p>
  </div>
  <p>We'll beat any advertised price!</p>

  <hr> <!-- show span (with class) -->
  <h2>Coding Horror! Coding Horror!</h2>
  <p>See our <span class="special">special</span> deal on Droids!</p>
  <p>We'll beat <span class="shoutSPAN">any advertised</span> price!</p>

  <hr> <!-- show combined selector -->
  <h2>Eat at <span>Greasy's Burger</span>...</h2>
  <p>The <span>greasiest</span> burgers in town! </p>
  <h4>Yummy <span>and greasy</span> at the same time! </h4>

  <hr> <!-- pseudo classes -->
  <a href="http://www.aircanada.com"> a link here, hover me to r
  <h3> some text, hover me to blue</h3>
  </body>

```

Coding Horror! Coding Horror!

Our mission is to combine programming and "human" factors with geekiness!

Coding Horror! Coding Horror!

See our special deal on Droids!

Today only!

Coding Horror! Coding Horror!

See our special deal on Droids!

We'll beat any advertised price!

Coding Horror! Coding Horror!

See our **special** deal on Droids!

We'll beat any advertised price!

17

CSS styles

RECALL

property	description
color	color of the element's text
background-color	color that will appear behind the element

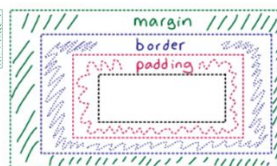


- ☐ (text) color
- ☐ font
- ☐ text
- ☐ background
- ☐ Box model
 - padding, margin, border
- ☐ display, visibility
 - none, hidden
- ☐ Floating
- ☐ width, height, position.....

property	description
font-family	which font will be used
font-size	how large the letters will be drawn
font-style	used to enable/disable italic style
font-weight	used to enable/disable bold style

property	description
text-align	alignment of text within its element
text-decoration	decorations such as underlining
text-indent	indents the first letter of each paragraph

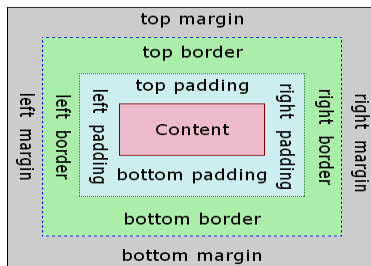
property	description
background-color	color to fill background
background-image	image to place in background
background-repeat	whether/how bg image should be repeated



18

The CSS Box Model (cont.)

- Box width = content width + L/R padding + L/R border + L/R margin
- Box height = content height + T/B padding + T/B border + T/B margin



```
div {
  width: 320px;
  padding: 10px;
  border: 10px solid gray;
  margin: 0;
```

“border-box model”

19

Put these topics together... try it out

```
<link href="box.css" type="text/css" rel="stylesheet" />

</head>
<body>

  <!-- border -->
  <h2 id="bord">text with thin border</h2>
  <h2 >text with thick border</h2>

  <hr> <!-- padding -->
  <h2 id="pad">text with padding 40px (on all 4 directions)</h2>
  <h2 id="pad2">text with padding-left 40</h2>
  <h2 id="pad3">text with adding-left: 100px; padding-top: 30px</h2>

  <hr> <!-- margin -->
  <h2 >text with default margin!</h2>
  <h2 id="marg">text with margin-left 40</h2>
  <h2 id="marg2">text with margin 50px (on all 4 directions)</h2>

</body>
</html>
```

text with thin border

text with thick border

Text with padding 40px (on all 4 directions)

Text with padding-left 40

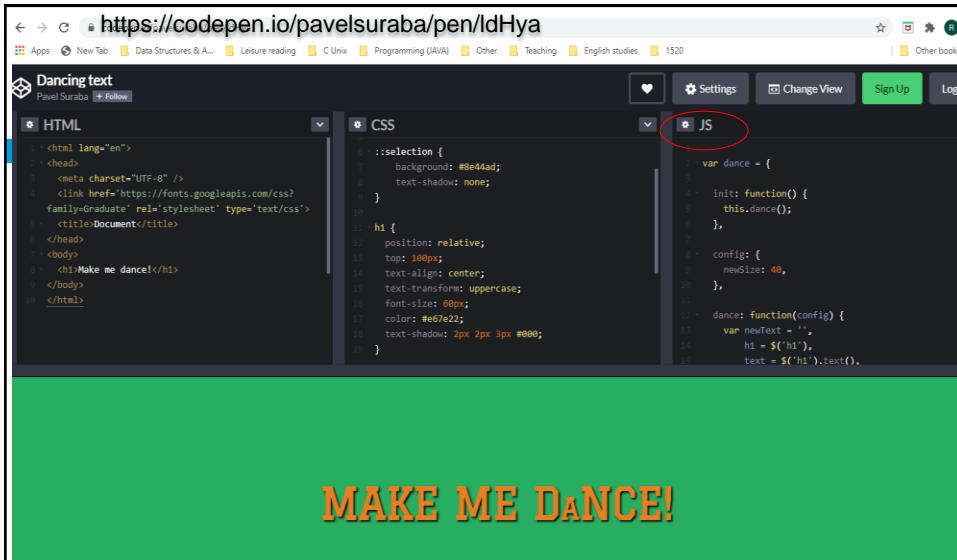
Text with adding-left: 100px; padding-top: 30px

Text with default margin!

Text with margin-left 40

Text with margin 50px (on all 4 directions)

20

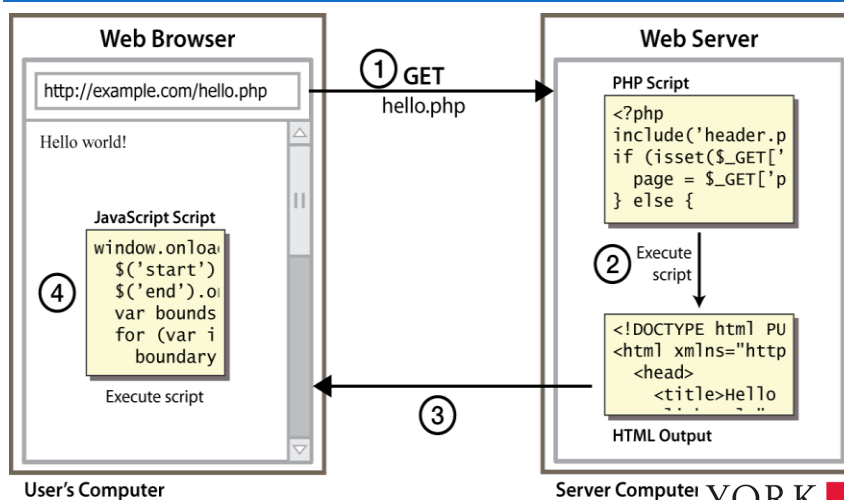


can distribute the work more easily. Faster development

- **writers need not be designers**
- **designers need not be programmers**
- **programmers don't need to understand the text content**

21

Client Side Scripting



22

What is JavaScript?

- a lightweight programming language ("scripting language")
 - used to make web pages interactive
 - E.g., insert/update dynamic text into HTML
 - E.g., update images on HTML
 - E.g., change presentation on HTML (dance me)
 - **react to events** (ex: mouse click, key press **from HTML**)



25

JavaScript vs Java



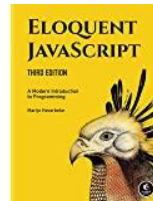
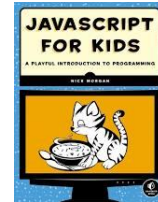
- NOT related to Java other than by name and some syntactic similarities
- interpreted, not compiled
- more relaxed syntax and rules
 - fewer and "looser" data types
 - variables don't need to be declared
 - errors often silent (few exceptions)
- key construct is the function rather than the class
- contained within a web page and integrates with its HTML/CSS content



27

course resources

- **web resources**
 - many web resources (no specific textbook)
 - **W3schools** !!!
 - Mozilla JavaScript
 -
- **recommended (bot not required) resources**
 - **JavaScript for Kids, by Nick Morgan**
 - for beginners
 - **Eloquent JavaScript, 3rd Edition, by Marijn Haverbeke**
 - for those who want to go to more intense levels of JS
 - Ch1-9, 14, 15



28

html+css+js

```
<button type="button"
onclick="document.getElementById('demo').innerHTML = 'Hello' ">
Click me to display Date and Time.</button>
```

```
<script>
function myFunction() {
  var x = document.getElementById("demo");
  x.style.fontSize = "25px";
  x.style.color = "red";
}
</script>

<button onclick="myFunction()">Click Me!</button>
```



30



30

JavaScript Example

```
<!DOCTYPE html>
<html>
<body>

<p id="demo">Click the button to change the layout of this paragraph</p>

<script>
function myFunction() {
    var x = document.getElementById("demo");
    x.style.fontSize = "25px";
    x.style.color = "red";
}
</script>

<button onclick="myFunction()">Click Me!</button>

</body>
</html>
```

Click the button to change the layout of this paragraph

Click Me!



Click the button to change the layout of this paragraph

Click Me!



31

31

html+css+js

Defines contents & structure of the webpage.

HTML File

```
<head>
  <link href="my.css"/>
  <script src="my.js"></script>
</head>

<body>

<button onclick="myFunction">...

</body>
```

CSS file: my.css

```
body{
color: red;
}
```

Defines the style of the webpage.

Javascript file: my.js

```
function
myfunction() {
    alert();
}
```

Defines the behavior of the webpage.

32

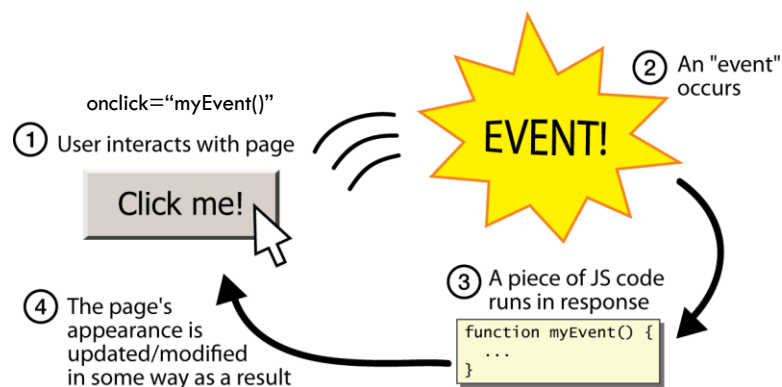
Linking to a JavaScript file: script

```
<script src="filename" type="text/javascript"></script>
```

HTML

- ❑ script tag should be placed in HTML page's head
- ❑ script code is stored in a separate .js file
- ❑ JS code can be placed directly in the HTML file's body or head (like CSS)
 - but this is bad style (should separate content, presentation, and behavior)

Event-driven programming



Event-driven programming

- you are used to programs start with a main method (or implicit main like in PHP)
- JavaScript programs instead wait for user actions called *events* and respond to them
- event-driven programming: writing programs driven by user events
- Let's write a page with a clickable button that pops up a "Hello, World" window...



37

Buttons

```
<button>Click me!</button>
```

HTML

- button's text appears inside tag; can also contain images
- To make a responsive button or other UI control:
 1. choose the control (e.g. button) and event (e.g. mouse click) of interest
 2. write a JavaScript function to run when the event occurs
 3. attach the function to the event on the control



38

JavaScript functions

```
function name() {
  statement ;
  statement ;
  ...
  statement ;
}
```

Can have parameters
Can have branch, loops

JS

```
function myFunction() {
  alert("Hello! How are you?");
}
```

JS

- ❑ the above could be the contents of example.js linked to our HTML page
- ❑ statements placed into functions can be evaluated in response to user events

Event handlers

```
<element attributes onclick="function();">...
```

HTML

```
<button onclick="myFunction();">Click me!</button>
```

HTML

- ❑ JavaScript functions can be set as event handlers
 - when you interact with the element, the function will execute
- ❑ **onclick** is just one of many event HTML attributes we'll use
- ❑ More on next slide...

Events

JavaScript Events



Event	Description
onchange	Script runs when the element changes
onsubmit	Script runs when the form is submitted
onreset	Script runs when the form is reset
onselect	Script runs when the element is selected
onblur	Script runs when the element loses focus
onfocus	Script runs when the element gets focus
onkeydown	Script runs when key is pressed
onkeypress	Script runs when key is pressed and released
onkeyup	Script runs when key is released
onclick	Script runs when a mouse click
ondblclick	Script runs when a mouse double-click
onmousedown	Script runs when mouse button is pressed
onmousemove	Script runs when mouse pointer moves
onmouseout	Script runs when mouse pointer moves out of an element
onmouseover	Script runs when mouse pointer moves over an element
onmouseup	Script runs when mouse button is released



41

JavaScript Events



Object	Event	User Action
Text field	onBlur	Tab away from window, frame or form element. Use onBlur when you tab out of a field.
	onFocus	Tab to a window, frame or form element.
	onSelect	Select text. If a form has multiple choices, you must use onSelect.
	onChange	Change text and click.
Button	onClick	Mouse click.
Checkbox	onClick	Mouse click.
Radio button	onClick	Mouse click.
Link	onMouseOver	Move mouse pointer over.
	onClick	Mouse click.



42

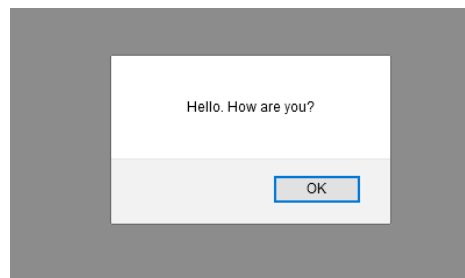
Event handlers

```
<button onclick="myFunction();" >Click me!</button>
```

HTML

```
function myFunction() {  
    alert("Hello! How are you?");  
}
```

JS

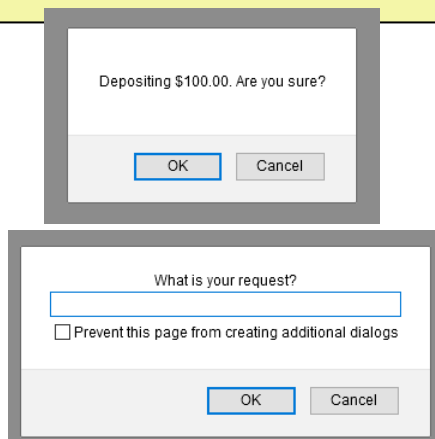


43

More Popup boxes

```
alert("message"); // message  
confirm("message"); // returns true or false  
prompt("message"); // returns user input string
```

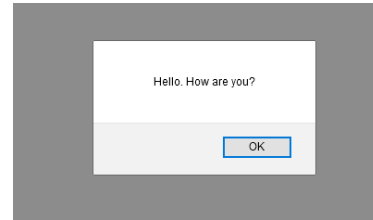
JS



44

Document Object Model (DOM)

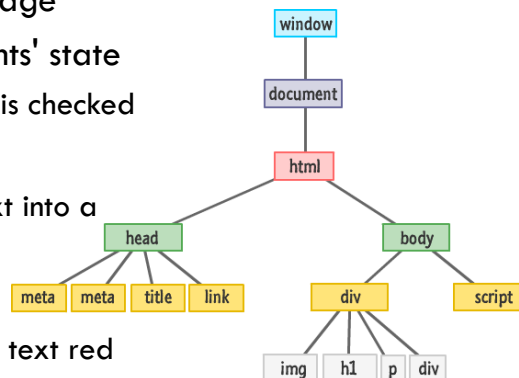
- Event handled!
- But instead of popping up an alert window, a better user experience would be to have the message appear on the page...
- In most situations we want response to update/alter the html page component
- How JavaScript interact with a web page's content.



45

Document Object Model (DOM)

- most JS code manipulates elements on an HTML page
- we can examine elements' state
 - e.g. see whether a box is checked
- we can change state
 - e.g. insert some new text into a div
- we can change styles
 - e.g. make a paragraph text red
-



47

DOM element objects

HTML

```
<p>
  Look at this octopus:
  
  Cute, huh?
</p>
```

DOM Element Object

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

JavaScript

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

- every element on the page has a corresponding DOM object
- act as glue between web content and JS



48

Accessing elements:

`document.getElementById`

- `document.getElementById` returns the DOM object representing an element with a given id
- access/modify the properties of the DOM object with `objectName.propertyName`
- Other “glues”: `document.getElementsByClassName`
`document.getElementsByTagName`
- can change the text inside most elements by setting the `innerHTML` property
- can change the text in form controls by setting the `value` property, `checked` property.
- can change the `styles`



50

Changing HTML element

access/modify the properties of the DOM object with
`objectName.propertyName`

common properties

- innerHTML
- style

corrs to ele attributes

- checked t/f
- disabled t/f
- href
- src alt
- value

Attribute (CSS)	Property or style object
color	color style.color="red";
font-size	fontSize
font-family	fontFamily
text-align	textAlign
text-decoration	textDecoration
background-color	backgroundColor
border-top-width	borderTopWidth
padding	padding
display	display



52

CSS styles

RECALL

property	description
color	color of the element's text
background-color	color that will appear behind the element

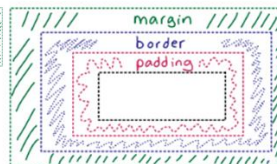


- (text) color
- font
- text
- background
- Box model
 - padding, margin, border
- display, visibility
 - none. hidden
- Floating
- width, height, position.....

property	description
font-family	which font will be used
font-size	how large the letters will be drawn
font-style	used to enable/disable italic style
font-weight	used to enable/disable bold style

property	description
text-align	alignment of text within its element
text-decoration	decorations such as underlining
text-indent	indents the first letter of each paragraph

property	description
background-color	color to fill background
background-image	image to place in background
background-repeat	whether/how bg image should be repeated



53

Accessing elements:

document.getElementById

```
var name = document.getElementById("id");
```

JS

```
<button onclick="changeText();" >Click me!</button>
```

```
<p id="text"> </p>
```

HTML

```
function changeText() {
    document.getElementById("text").innerHTML = "New text";
}
```

JS

Accessing elements:

document.getElementById

```
var name = document.getElementById("id");
```

JS

```
<button onclick="changeText();" >Click me!</button>
```

```
<p id="text"> </p>
```

HTML

```
function changeText() {
    var textB = document.getElementById("text");

    textB.innerHTML = "New text";
}
```

JS

Accessing elements: document.getElementById

```
var name = document.getElementById("id");
```

JS

```
<button onclick="changeText();" >Click me!</button>
<span id="output">replace me</span>
<p id="textbox"> </p>
```

HTML

```
function changeText() {
    var spa = document.getElementById("output");
    var textB = document.getElementById("text");

    spa.style.fontWeight = "bold"
    textB.style.color = "red";
}
```

JS



56

JS

EECS1520: WWW

https://www.w3schools.com/js/tryit.asp?filename=tryjs_intro_lightbulb

57

```

<!DOCTYPE html>
<html>
<body>




<p>Click the light bulb to turn on/off the light.</p>

<script>
function changeImage() {
  var image = document.getElementById('myImage');
  if (image.src.match("bulbon")) {
    image.src = "pic_bulboff.gif";
  } else {
    image.src = "pic_bulbon.gif";
  }
}
</script>
</body>
</html>


```

JS call function

EECS1520: WWW



Click the light bulb to turn on/off the light.



Click the light b


YORK
UNIVERSITY

https://www.w3schools.com/js/tryit.asp?filename=tryjs_lightbulb

58

Lab 1


click button to turn on/off light



ON ON ON ON ON

OFF


light 3 is on



ON ON ON ON ON

OFF


light 5 is on



ON ON ON ON ON

OFF


light is off



ON ON ON ON ON

OFF

light 1 is on



ON ON ON ON ON

OFF

61

checkbox state, attributes

```
<!DOCTYPE html>
<html>
<body>
```

```
<p>Is the checkbox checked?</p>
<input type="checkbox" id="checkbox1" onclick="someFunction()">
<p id="text" style="display:none">Checked!</p>
```

This normally belongs in
a .css file

```
function someFunction() {
  var checkBox = document.getElementById("checkbox1");
  var text = document.getElementById("text");
  if (checkBox.checked == true){
    text.style.display = "block";
  } else {
    text.style.display = "none";
  }
}
```

This normally
belongs in a .js file

Is the checkbox checked?



Checked!

62

Used to validate client side forms

```
<form name="myForm" action="/action_page.php"
onsubmit="return validateForm()" method="post">
```

```
  Name: <input type="text" name="fname">
  <input type="submit" value="Submit">
</form>
```

HTML

```
function validateForm() {
  var x = document.forms["myForm"]["fname"].value;
  if (x == "") {
    alert("Name must be filled out");
    return false;
  }
  return true; // optional?!
}
```

JS

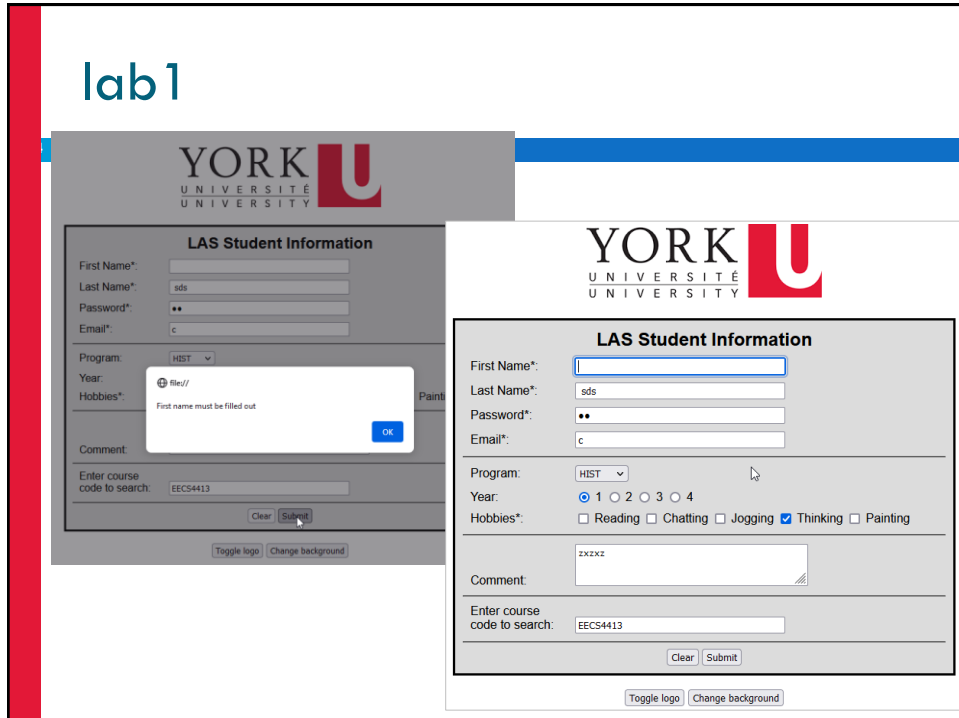
JavaScript Validation

Name:



63

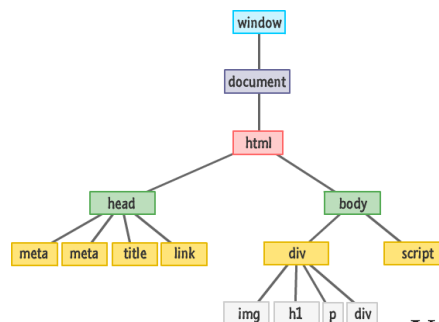
lab1



64

Can generate HTML component dynamically at run time

createElement(), appendChild(), removeChild(), etc.



65