Week 3 Reading

**OOP using “this”**

let user = {

name: "John",

age: 30,

sayHi() {

// "this" is the "current object"

alert(this.name);

}

};

user.sayHi(); // John

‘this’ keyword is used to access the object.

Please note that arrow functions are special: they have no this. When this is accessed inside an arrow function, it is taken from outside.

More info on this: <https://zellwk.com/blog/this/>

**Chapter 5 Objects – Sitepoint**

Objects are copied by reference. Changing a variables value will change that value for all the other variables linked to the address.

Regexp

* . matches any character, except line breaks
* \w matches any word character, and is equivalent to [A-Za-z0-9\_]
* \W matches any non-word character, and is equivalent to [\^A-Za-z0-9\_]
* \d matches any digit character, and is equivalent to [0-9]
* \D matches any non-digit character, and is equivalent to [^0-9]
* \s matches any whitespace character, and is equivalent to [ \t\r\n\f]
* \S matches any non-whitespace character, and is equivalent to [^ \t\r\n\f]

Modifiers can be placed after a token to deal with multiple occurrences of that token:

* ? makes the preceding token in the regular expression optional
* \* matches one or more occurrences of the preceding token
* + matches one or more occurrences of the preceding token
* {n} matches n occurrences of the preceding token
* {n,} matches at least n occurrences of the pattern
* {,m} matches at most m occurrences of the preceding token
* {n,m} matches at least n and at most m occurrences of the preceding token
* ^ marks the position immediately before the first character in the string
* $ marks the position immediately after the last character in the string

**Chapter 6 DOM – Sitepoint**

Node lists are like arrays but aren’t. Items can be accessed through dot / index notation, and they contain a length property, however they don’t work with other array properties such as splice.

Example: listI = document.getElementsByTagName(‘li’) creates an array like list in which items can be targeted as such: listI[0]

Attributes can be assigned by using the .setAttribute method.

Example: wonderWoman.setAttribute(‘id’, ‘amazon’);

In order to create a new element with text you must:

1.) create the element node

2.) create the text node

3.) append the text node to the element node

Either that or by changing the text content as so:

Const flash = document.createElement(‘li’);

flash.textContent = ‘Flash’

To make this process even faster, we can make a function to build elements!

function createElement (tag,text) {

const el = document.createElement(tag);

el.textContent = text;

return el

}

Let’s try it out by creating another new list item element:

const aquaman = createElement('li','Aquaman');

InnerHTML

Innerhtml can be used to output elements as well as text.

**Chapter 6 Events – Sitepoint**

preventDefault() is a method of the event object that can be used inside the callback function to stop the default behavior happening. To see an example, add the following line to the events.html file:

<p>

<a id='broken' href='https://sitepoint.com'>Broken Link</a>

</p>

Then add the following event listener inside the main.js file:

const brokenLink = document.getElementById('broken');

brokenLink.addEventListener('click',(event) => {

event.preventDefault();

console.log('Broken Link!');

});