JavaScript-UI-and-DOM

Document Object Model

Selecting DOM elements

getElementsBy, querySelector

NodeLists

LiveNodeList

StaticNodeList

DOM introduces objects that represent HTML elements and their properties

document.documentElement is <**html**>

document.body is the body of the page, where the content of the page is

Each of the HTML elements has corresponding DOM object type

HTMLLIElement represents <**li**>

HTMLAudioElement represents <**audio**>

Each of these objects has the appropriate properties

HTMLAnchorElement has **href** property

HTMLImageElement has **src** property

The document object is a special object

It represents the ***entry point*** for the DOM API

Different HTML elements have their specific attributes

HTMLImageElement has property **src**

HTMLInputElement has property **value**

HTMLAnchorElement has property **href**

HTML elements have properties for content

innerHTML ->Returns as a string the content of the element, without the element

outerHTML ->Returns as a string the content of the element, with the element

innerText / textContent -> Returns as a string the text content of the element, witout the tags

Select single element

```
var header = document.getElementById('header');
var nav = document.querySelector('#main-nav');
```

Select a collection of elements

```
var inputs = document.getElementsByTagName('li');
var radiosGroup = document.getElementsByName('genders');
var header = document.querySelectorAll('#main-nav li');
```

Using predefined collections of elements

```
var links = document.links;
var forms = document.forms;
```

getElementById(id): -> Returns a **single element** or null

```
var header = document.getElementById('header');
```

getElementsByClassName(className): -> Returns a **collection of elements**

```
var posts = document.getElementsByClassName('post-item');
```

DOM API contains methods for selecting elements based on some characteristic

getElementsByTagName(tagName); -> Returns a **collection of elements**

```
var sidebars = document.getElementsByTagName('sidebar');
```

getElementsByName(name); -> Returns a **collection of elements**

```
var gendersGroup = document.getElementsByName('genders');
```

The DOM API has methods that use CSS-like selectors to find and select HTML elements

querySelector(selector) -> returns the **left most element** that matches the selector

querySelectorAll(selector) -> returns **a collection of all elements** that match the selector

```
//the element with id="header"
var header = document.querySelector('#header');

//li elements contained in element with id=main-nav
var navItems = document.querySelectorAll('#main-nav li');
```

The DOM API can be used to select elements that are inside other elements

```
var wrapper = document.getElementById('wrapper');

// returns all div elements inside the element with id "wrapper"
var divsInWrapper = wrapper.getElementsByTagName('div');
```

A NodeList is a collection returned by the DOM selector methods:

getElementsByTagName(tagName)

getElementsByName(name)

getElementsByClassName(className)

querySelectorAll(selector)

```
var divs = document.getElementsByTagName('div'),

    queryDivs = document.querySelectorAll('div');

for(var i = 0, length = divs.length; i < length; i += 1){

    // do stuff with divs[i]…

}
```

NodeList looks like an array, but is not

It's an object with properties similar to array

Has length and indexer

Traversing an array with for-in loop works unexpected:

```
console.log(Array.isArray(divs)); // false

for (var i in divs) {

   console.log('divs[' + i + '] = ' + divs[i]);

}
```

There are two kinds of NodeLists

getElementsBy methods return a **LiveNodeList**

querySelectorAll returns a **StaticNodeList**