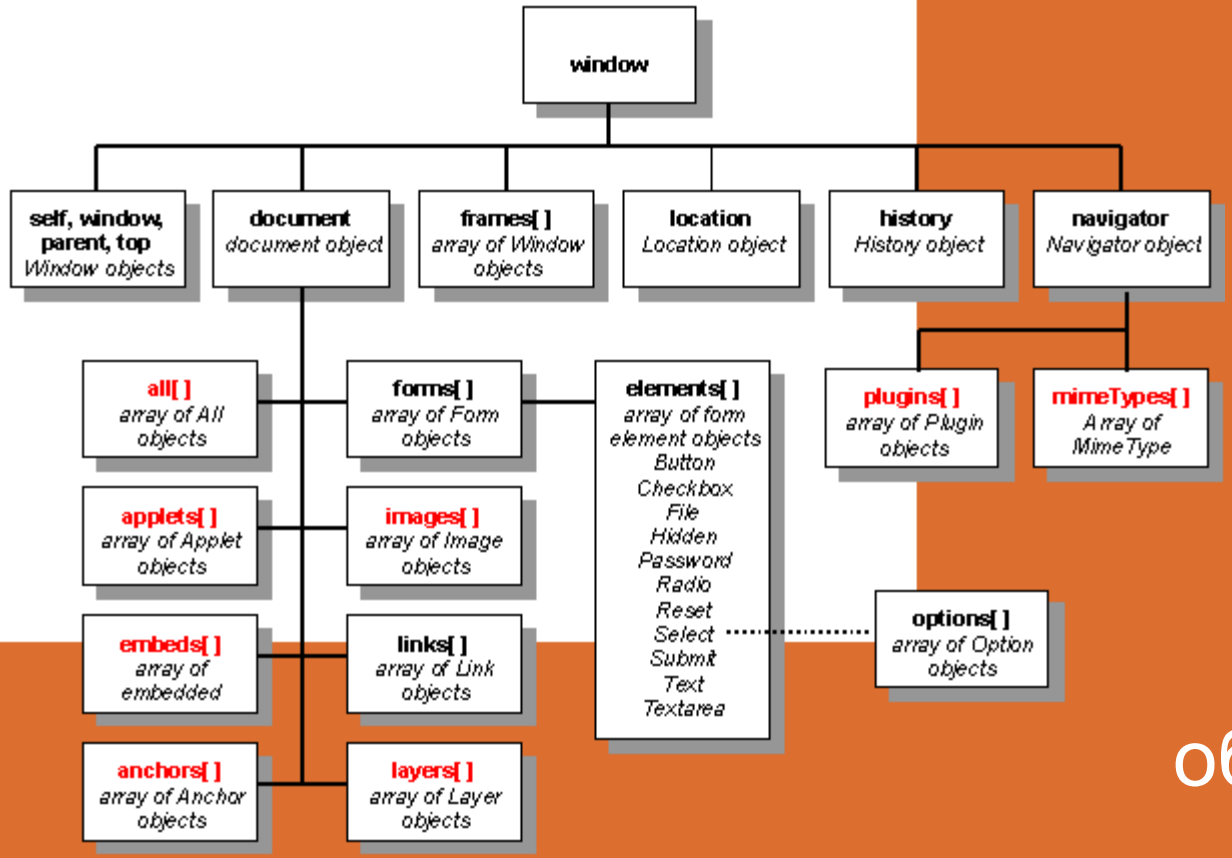
WEB разработка

JavaScript
DOM

Съдържание

- Вградени обекти в браузъра
- DOM -
  - DOM API
    - Селектиране на DOM елементи
    - Node, nodes, nodeLists(Live&Static)



Вградени обекти в браузъра

Вградени обекти в браузъра

Браузърът ни предоставя данни чрез -

window

Елементът, на върха на DOM дървото

Представлява порзореца на браузъра

document

Съдържа информация за текущия документ, зареден в браузъра

screen

Съдържа информация за потребителските настройки за визуализация

navigator

Съдържа информация за браузъра

Demo 1

Вградени обекти в браузъра - demo

document object

Съдържа масиви с данни за текущата заредена в браузъра страница.

```
document.links[0].href = "yahoo.com";
document.write("This is some <b>bold text</b>");
```

Вградени обекти в браузъра - demo

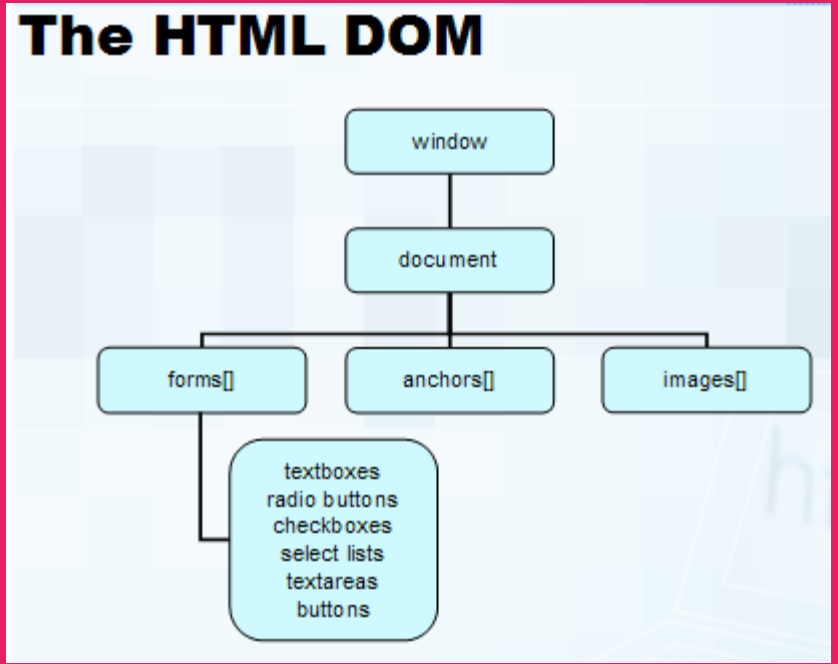
Demo 2

document.location
Текущия URL, зареден в браузъра или за пренасочване на браузъра

```
document.location = "http://www.yahoo.com";
```

DOM

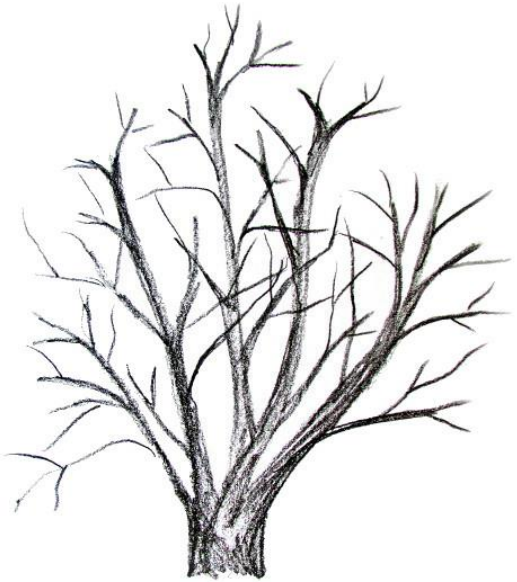
Document Object Model



DOM

DOM - *document object model, MDN*

- DOM - структура на HTML
  - Представяне на HTML документа като ***DOM - дърво***
    - Състоящо се от елементи, които имат деца-елементи,
    - Които на свой ред могат да имат деца елементи и т.н.
- DOM - съдържа API за обхождане/променяне на ***DOM - дървото***
  - Позволява ни да променяме HTML документа динамично -
    - да махаме или добавяме нови елементи
    - Да променяме начина, по който изглеждат елементите
    - Добавяме или премахваме атрибути на HTML елементи



Как взаимодействаме с ***DOM - дървото*** - как го променяме?

- Чрез JavaScript

DOM API - demo

Всеки HTML елемент, чрез DOM API може да бъде достъпен като JS обект.

document.documentElement is the <html> element
document.body is the <body> element of the page

Demo 3

DOM API - *info*

document е специален обект -

Той представлява началото - входната точка, от която

Започва работа DOM API.

document е

корена/root на

DOM - дървото.

DOM API - demo

Атрибутите на HTML елементите съответстват на свойствата/properties на JS обекта.

Атрибути<->Свойства
id, className, style, onclick, и т.н.

Demo 4

DOM API - demo

innerHTML

Съдържа стринг - съдържанието на елемента, без самия елемент

outerHTML

Стринг - съдържанието на елемента и елемента

innerText / textContent

Стринг - текста на елемента, беза таговете

Demo 5

DOM API - *info*

Всеки HTML елемент има съответен тип **DOM object**

Всеки от тези обекти има свои специфични свойства, например -

HTMLAnchorElement има **href** свойство
HTMLImageElement има **src** свойство
HTMLInputElement има **value** свойство

Demo 6

Селектиране на HTML елемент/и

Selecting HTML elements

HTML елементите могат да бъдат достъпени и **запазени в променливи** с помощта на DOM API

- Избиране на един елемент или защо в страницата трябва да имаме само едно id с дадена стойност.

```
var email_2 = document.getElementById('email_address2');

var span = document.querySelector('#email_form span');///първия срещнат такъв елемент
```

Demo 7

Selecting HTML elements

- Селектиране на група елементи

```
var inputs = document.getElementsByTagName('input');
var email = document.getElementsByName('email_address2');
var classGroup = document.getElementsByClassName('className');
var form = document.querySelectorAll('#email_form input');//всички срещнати
```

*Какво съхранява всеки един от примерите?**
Как открива елементите?

Demo 8

Selecting HTML elements

- Използване на дадена група от елементи

```
var links = document.links;
var forms = document.forms;
        var form = document.forms[];
```

Използване на getElementsBy методи

getElementsBy Methods

- DOM API съдържа методи за селектиране на елементи, базирани на някои характеристики
  - **getElementById**(id):
    - Резултатът е един елемент или null

getElementsBy Methods

- **getElementsByClassName(className):**
  - Връща група/масив от елементи

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

- **getElementsByTagName(tagName);**
  - Връща група/масив от елементи

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

getElementsBy Methods

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

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

Задача - Използвайте готовите файлове и приложете четирите метода за селектиране на елементи. Използвайте характерните атрибути на съответните HTML елементи като свойства на DOMобектите и отпечатвайте стойностите им.

Използване на querySelector методи

querySelector methods

- DOM API притежава методи, използващи CSS селектори за откриване и селектиране на HTML елементи.
  - **querySelector(selector)**
    i. Връща **<u>първия елемент</u>**, който селектора открие.
  - **querySelectorAll(selector)**
    i. Връща **<u>колекция/масив от елементи</u>**, които отговарят на селектора.

- И двата метода приемат параметър стринг -
  - CSS селектора на елемента

```
var header = document.querySelector('#header');
```

querySelector methods

- И двата метода приемат параметър стринг -
  - CSS селектора на елемента

```
var header = document.querySelector('#header');

var navItems = document.querySelectorAll('#main-nav li');
```

Задача - Използвайте готовите файлове и приложете методите за селектиране на елементи. Използвайте характерните атрибути на съответните HTML елементи като свойства на DOMобектите и отпечатвайте стойностите им.

Селектиране на вложени елементи

Селектиране на вложени елементи

Избиране на всички DIV-елементи, намиращи се в елемент с id = "wrapper„

```
var wrapper = document.getElementById('wrapper');
var divsInWrapper = wrapper.getElementsByTagName('div');
```

Всички методи могат да бъдат използвани върху HTML елементите, с изключение на **getElementById()**

Задача - селектирайте вътрешни елементи от говите файлове.

Node
Nodes
NodeList

NodeList

- NodeList - колекцията от елементи, резултат от методът със съответния DOM селектор:
    - getElementsByTagName(tagName)
    - getElementsByName(name)
    - getElementsByClassName(className)
    - querySelectorAll(selector)

NodeList

- NodeList прилича на масив, но не е:)
  - Обект - със свойства, подобни на масива
    - length и индекси
  - Ако го обхождаме с for-in може да получим неочаквани резултати:)

Static NodeList

&

Live NodeList

info

Static NodeList and Live NodeList

- Съществуват два вида NodeList-и
  - Методите **getElementsBy** - връщат **LiveNodeList**
  - Методите **querySelectorAll** - връщат **StaticNodeList**

- **LiveNodeList** следят какво се случва със селектираните елементи
  - Дори, когато са направени промени в DOM, след като е създаден NodeList-а, промените се отразяват в него.

Static NodeList and Live NodeList

- **StaticNodeList** пози елементите такива, каквито са били в момента на изпълнение на метода.

- LiveNodeList е по-бавен от обикновен масив.
  - Препоръчително е да се кешира, за да бъде по-бърз
  - По-добре да се конвертира към масив, когато ще бъде обхождан много пъти.

Demo 10