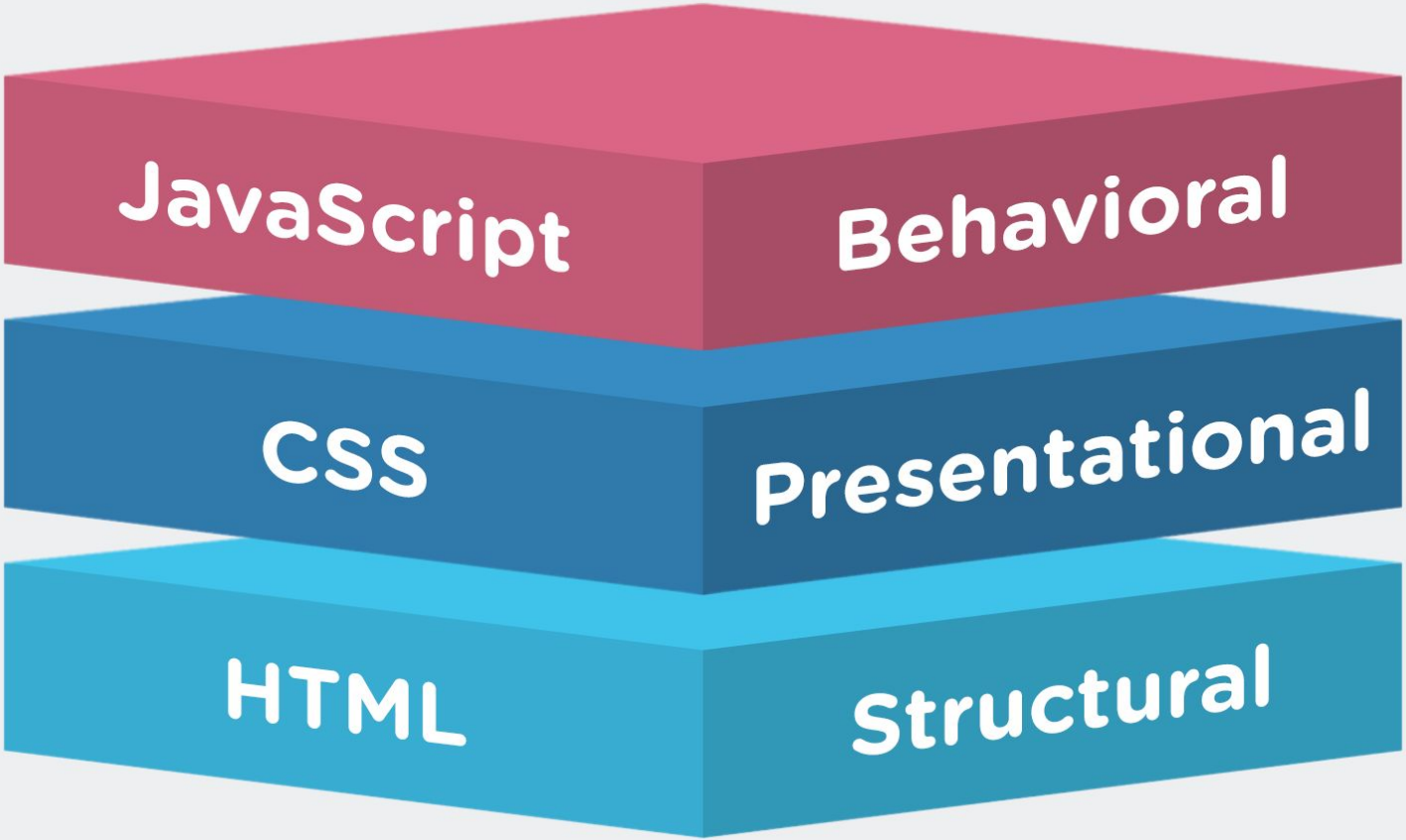
wantsome
The friendly IT Academy

Saptamana 5

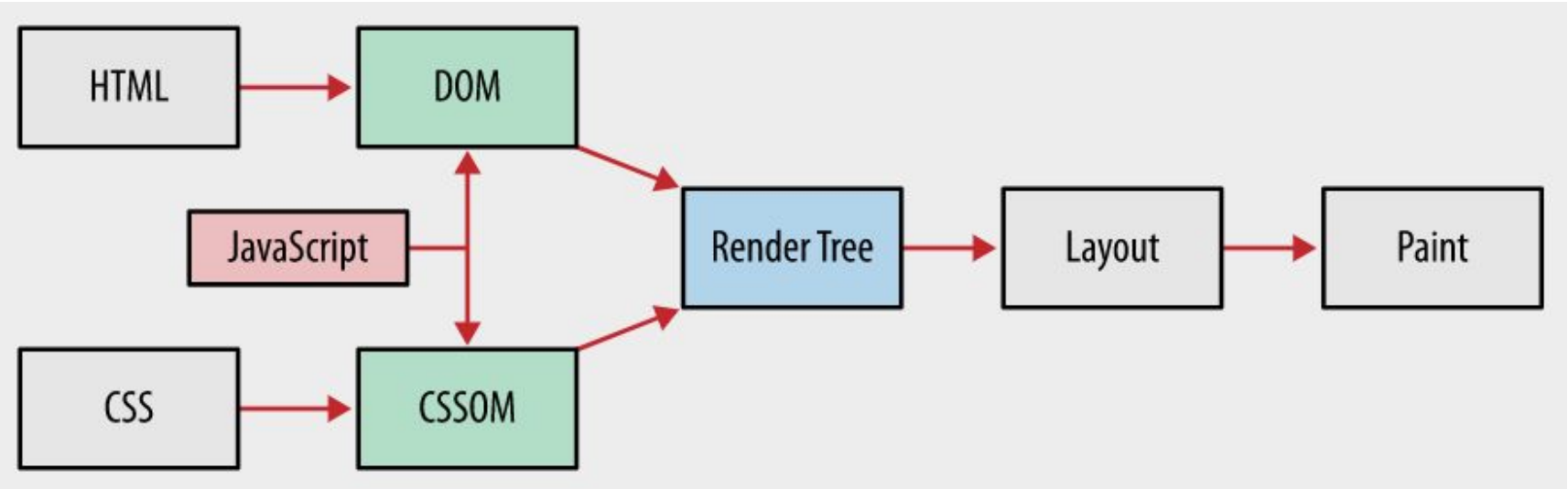
Partea 1

Programare Front-End

III. Interactivity - JavaScript


JavaScript
Behavioral
CSS
Presentational
HTML
Structural

Ciclul de viata al unei pagini web (*lifecycle*)



1. JavaScript Introduction

JavaScript - Ce este si la ce foloseste ?

- **limbaj de programare** al carui scop initial a fost sa “dea viata” paginilor web
- “programele” dezvoltate in acest limbaj se numesc ***script***-uri, iar acestea sunt incluse in cadrul unei pagini HTML spre a fi executate de catre browser in momentul in care pagina este incarcata
- **JavaScript** poate fi executat si pe partea de server (NodeJS)
- **JavaScript** este executat de catre un ***JavaScript Engine****;* De la browser la browser, intalnim diferite *engines*:
    - V8 - Chrome, Opera
    - SpiderMonkey - Firefox
    - Chakra, Trident, ChakraCore - IE, Edge
    - Nitro - Safari
    - ...

JavaScript - Ce este si la ce foloseste ?

- La nivel de browser, **JavaScript** este capabil sa faca urmatoarele lucruri:
    - Sa **manipuleze** elementele HTML din DOM: adaugare, stergere sau modificare - atat din punct de vedere al structurii (atribute HTML) cat si al stilurilor (proprietati CSS)
    - Sa reactioneze la anumite **evenimente / actiuni** produse de catre utilizator: *mouse click, miscarea cursorului, apasarea unei taste*, etc…
    - Sa trimita **request**-uri folosindu-se de retea pentru a descarca si incarca fisiere / resurse - ***AJAX***
    - Sa seteze si sa obtina valoarea unui ***COOKIE***
    - Sa **afiseze alerte**, mesaje utilizatorului (folosind functionalitati puse la dispozitie de browser - OS)
    - Sa **stocheze date** la nivel de client - ***localStorage, sessionStorage*** -
    - Sa faca o multime de alte lucruri grozave pe care le vom descoperi pe parcurs

References: https://javascript.info/intro

JavaScript - Cum folosim un script intr-o pagina HTML ?

- La fel ca si pentru resursele de tip CSS, exista multiple modalitati de a include un **script** intr-o pagina **HTML**, in functie de locatia acestuia:

  - inline

```
<script>
    document.getElementById("demo").innerHTML = "My First JavaScript";
</script>
```

  - external

```
<script type=”text/javascript” src=”path/to/my-script.js”> - path sau URL
```

- *script*-urile pot fi plasate fie in **<head>** sau **<body>** - *felul in care facem plasarea acestora poate afecta performanta !*

wantsome
The friendly IT Academy

JavaScript – Output – Unde sunt afisate “datele” ?

- datele, rezultatul calculelor si al algoritmilor executati in cadrul *script*-urilor pot fi afisate, in functie de situatie, in:
  - **Corpul unui element HTML**
  - **In consola browser-ului** - metoda folosita pentru *debugging* - utilizand *console.log()*, de exemplu
  - **In DOM** - metoda folosita pentru testare
  - **Intr-o fereastra de alerta** - metoda foarte rar utilizata luand in considerare stadiul tehnologiei de astazi

JavaScript - Instructiuni = Statements

- Orice limbaj de programare are o sintaxa specifica prin intermediul careia se definesc anumite **instructiuni** spre a fi executate de catre calculator
- Aceste instructiuni sunt cel mai des intalnite ca fiind asociate cu denumirea de **statements**
- Un program, *script*, scris intr-un limbaj de programare este alcatuit dintr-un set de astfel de **statements**
- Ca si tipuri de **statements**, avem:
  - **Tipuri de valori**
  - **Operatori**
  - **Expresii**
  - **Keywords**
  - **Comentarii**

2. JavaScript - Statements Syntax and Purpose

JavaScript - Valori

- valorile in JavaScript pot fi fixe sau pot varia
- valorile fixe mai sunt numite si **literals**, iar cele care variaza pe masura ce instructiunile sunt execute sunt numite **variables**, sau, in romana: **variabile**
  - **literals**: 39, 39.93, “Wantsome”, “Ce tare e la cursul de FE !”
  - **variables**:

```
var pi = 3.14;
var sum = 3 + 5 + 6;
var courseName = “FE”;
var userName = “Lucian” + “ “ + “Costin”;
```

JavaScript - Variabile

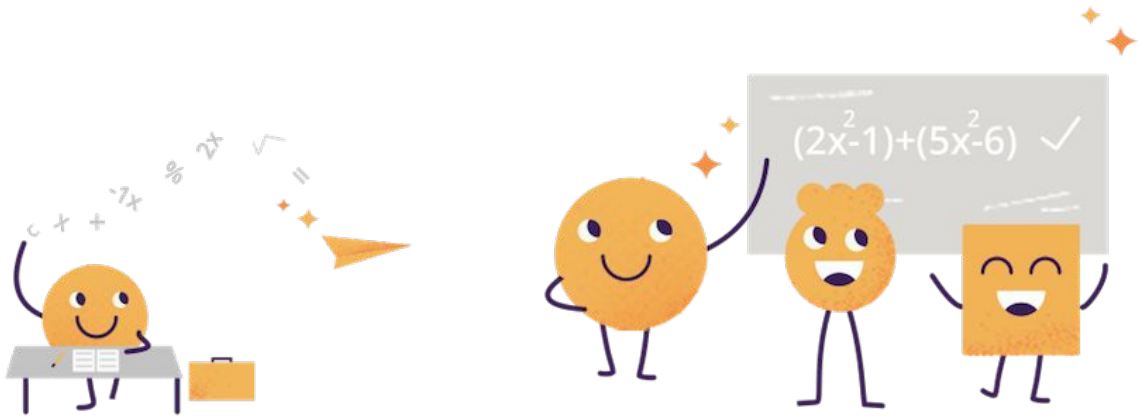
```
var firstValue = 5;
var secondValue = 6;
var sum = firstValue + secondValue;

var firstName = “John”, lastName = “Doe”;
var jobTitle;
jobTitle = “Front-End Developer”;
var nameAndJobTitle = firstName + “ “ + lastName + “, ” + jobTitle;
```

JavaScript - Variabile - *undefined*

- daca incercam sa accesam o variabila care a fost definita dar care nu are valoare, sau daca incercam sa accesam o variabila nedefinita, JavaScript va atribui implicit o valoare ***undefined***

```
var firstValue;          ->    firstValue === undefined
undeclaredVariable       ->    undeclaredVariable === undefined
```

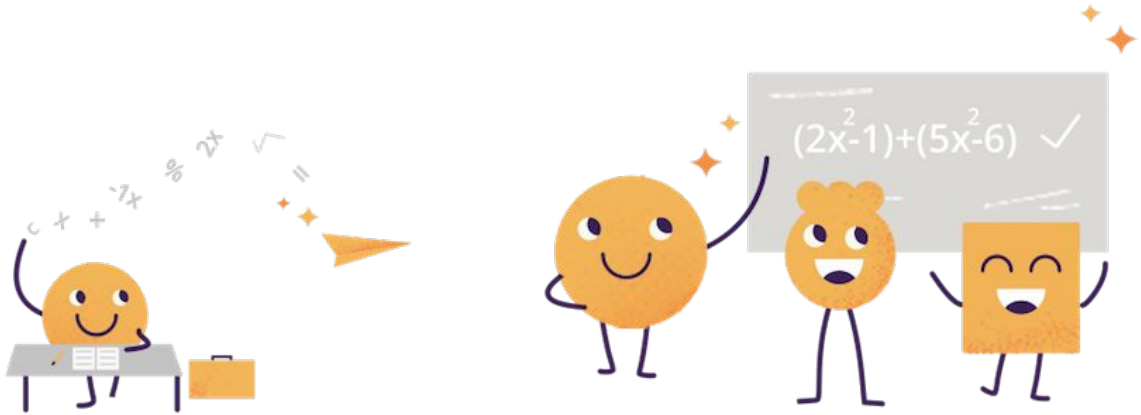
PRACTICE: **JavaScript Values and Variables I**



PRACTICE: **JavaScript Values and Variables**

II

http://bit.do/ex1JS
http://bit.do/ex2JavaScript



wantsome
The friendly IT Academy

JavaScript - Tipuri de date

- JavaScript, la fel ca majoritatea limbajelor de programare lucreaza cu **tipuri de date (numerice, siruri de caractere, …)**

- Tipurile de date definesc tipul valorilor care pot fi asignate unei variabile

- De exemplu, daca vrem sa lucram cu numarul 31 si ulterior sa il adunam cu altceva, cu siguranta va fi un **calcul aritmetic** si avem nevoie tot de un numar, dar daca vrem sa lucram cu “31” ca si sir de caractere, operatia va fi de fapt o **concatenare** si vom avea nevoie tot de un **sir de caractere**

JavaScript - Tipuri de date

- Schimbarea unui tip de date in alt tip (de ex: din 31, facem “31”) poarta denumirea de ***coercion***

- Exista **7 tipuri de date predefinite in JavaScript**: **null, undefined, boolean, number, string, object, *symbol(ES6+)***

- Inafara de **Object**, restul datelor sunt numite **‘primitive’**

- *Pentru a vedea ce tip de date are o anumita valoare, folosim operatorul* ***typeof*** *(de ex:* `typeof 32`*)*

JavaScript - Tipuri de date

- **Number** - `var a = 123;`
- **String** - `var a = “Wantsome”` (putem folosi “” sau ‘ ‘) - Orice valoare insotita de ghilimele va fi *string*
- **Boolean** - avem 2 valori ce reprezinta valoarea de tip bool - **true** (adevarat) si **false** (fals)
  - Folosim *boolean* cand vrem sa facem o comparatie sau sa fim siguri ca ceva este true sau false;
- **null** - nu apartine niciunei valori de mai sus.

  - `var a = null` - Inseamna ca variabila a este declarata goala, nu contine nimic in interior.
- **undefined** - aproximativ ca *null*-ul doar ca inseamna “valoarea n-a fost asignata”

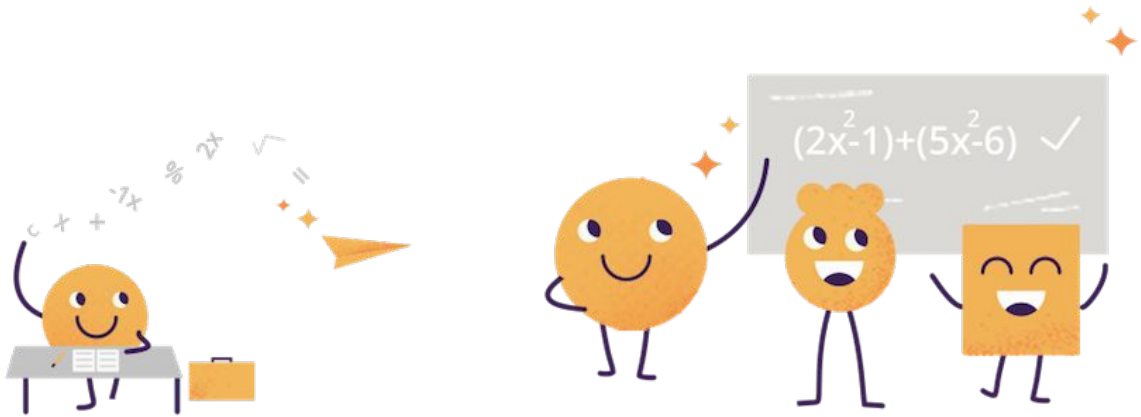
  - `var a;` (daca nu ii asignam o valoare va fi undefined)
- **Object** - este un tip special de date; toate celalte date numite mai sus sunt “**primitive**” pentru ca au asignat un singur lucru (*string, number, null,* etc), **Object** - este o **colectie de date**

```
var myCar = {
  masina: 'tesla',
  roti: 4,
  motor: 1
}
```

wantsome
The friendly IT Academy

PRACTICE: **JavaScript Data types**

http://bit.do/exJS-DataTypes



JavaScript - Operatori

- Exista mai multe tipuri de operatori care definesc diverse operatii

Operatori aritmetici (+ - * / % ++ --)

```
var a = 2 + 3

var b = 2;
var c = 3;
var result = a + b;
```

```
b++
var result2 = b // ce rezultat avem ?
c--
var result3 = c // ce rezultat avem ?
```

Operatori pe stringuri

```
var me = 'Eu sunt'
var jobTitle = 'JavaScript developer'

var result = me + ' ' +  jobTitle;
```

JavaScript - Operatori

Operatori de asignare (=, +=, -=, *=, /=, %=)

```
var a = 5
var b = 20
a = b // ce printeaza ?
```

x += y	x = x + y
x -= y	x = x - y
x *= y	x = x * y
x /= y	x = x / y
x %= y	x = x % y

JavaScript - Operatori

Operatori de comparare (==, ===, !=, !==, > , <, >=, <=)

```
var a = 2;
var b = 3;
var c = '2';
a == b // false
a == c // ?
a === c // ?
```

Operatori logici (&&, ||, !)

```
var a = 2;
var b = 3
(a && b)
(a || b)
!(a ===b)
```

JavaScript - Operatori

Operatorul typeof(typeof)

```
typeof "John"                 // Returns string
typeof 3.14                   // Returns number
typeof NaN                    // Returns number
typeof false                  // Returns boolean
typeof [1, 2, 3, 4]           // Returns object
typeof {name:'John', age:34}  // Returns object
typeof new Date()             // Returns object
typeof function () {}         // Returns function
typeof myCar                  // Returns undefined (if myCar is not declared)
typeof null                   // Returns object
```