Univerzitet u Sarajevu
Elektrotehnički fakultet Sarajevo
Računarstvo i informatika
Predmet: Web tehnologije

Laboratorijska vježba br. 4: JavaScript (DOM)

1. BOM (*Browser Object Model*) i DOM (*Document Object Model*)

1.1. Šta je BOM (*Browser Object Model*)?

BOM (*Browser Object Model*) je srž JavaScript-a na web-u. BOM pruža objekte koji omogućavaju upravljanje funkcionalnostima web pretraživača. JavaScript nudi niz objekata za manipulaciju samim web pretraživačem i okruženjem, kao što su:

- **Window** - nudi metode za upravljanje samim prozorom web pretraživača. Primjeri:

```
window.open("http://etf.ba"); // Otvara link u novom prozoru
window.resize(200,100); // Promjena velicine prozora
```

- **Location** - trenutna lokacija (URL). Primjeri:

```
alert(location.href); // Trenutni URL
location.replace("http://etf.ba"); // Preusmjeravanje
```

- **Navigator** - informacije o web pretraživaču i sistemu. Primjeri:

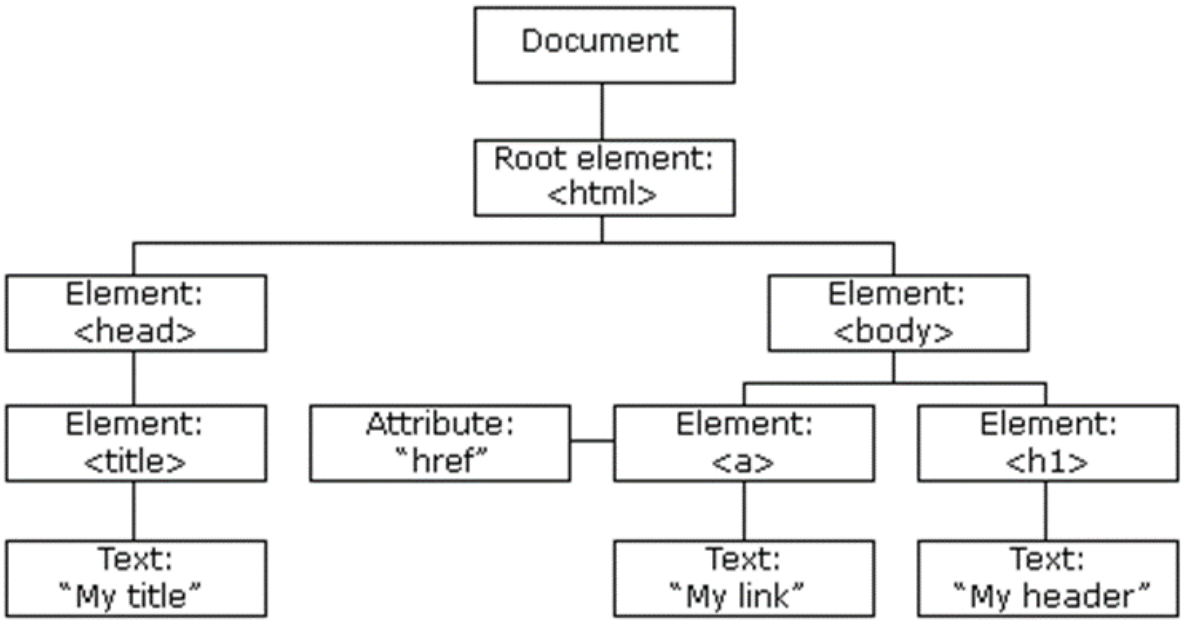
```
alert(navigator.appName); // Naziv web pretrazivaca npr. Firefox
```

- **History** - rad sa historijom preglednika. Primjer:

```
window.history.back(); // Povratak na prethodnu stranicu
```

1.2. Šta je DOM (*Document Object Model*)?

DOM omogućava manipulaciju sadržajem web stranice putem JavaScript-a. Pomoću DOM-a možete: dodavati, mijenjati, brisati sve HTML elemente, njihove atribute, događaje (eng. *events*) i stil (tj. CSS). Svaki tag predstavljen je instancom objekta *HTMLElement* iz koje je izveden niz objekata za rad sa specifičnim tagovima. Npr. *Table* objekat daje metode za pristup redovima i kolonama, dodavanje reda/kolone i slično. Globalni objekat **document** (tipa *Document*) omogućava kretanje kroz stablo tagova (primjer stabla je prikazan na slici 1.1.).



Slika 1.1. Primjer DOM stabla

Specifični HTML element se može pronaći metodama:

- **getElementById** - vraća objekat sa datim atributom *id*. U validnom HTML dokumentu *id* mora biti jedinstven za svaki tag, jer je u suprotnom ponašanje nedefinirano. Primjer:

```
let tabela = document.getElementById("tabela");
let red = tabela.insertRow(0);
let celija = red.insertCell(0);
celija.innerHTML = "Zdravo!";
```

- **getElementsByName** - vraća niz elemenata sa datim atributom *name*. Za razliku od *id*, ovaj atribut je ispravan samo na poljima formi i ne mora biti jedinstven (obratiti pažnju na množinu: elements). Primjer:

```
let input = document.getElementsByName("adresa")[0]; // Prvo polje
naziva "adresa"
```

- **getElementsByTagName** - vraća niz elemenata sa datim imenom taga. Primjer:

```
let prvi_paragraf = document.getElementsByTagName("p")[0];
```

- **getElementsByClassName** - niz elemenata prema vrijednosti atributa *class*.
- **document.forms** - vraća niz *Form* objekata koji odgovaraju formama na stranici. Primjeri:

```
let email = document.forms["forma"]["email"].value;
let email2 = document.forms[0]["email"].value; // Ako je prva
"forma"
let email3 = document.forms[0].email.value;
```

- Kretanje kroz stablo koristeći metode objekta *Node* koje nasljeđuje i objekat *HTMLElement*. Primjer:

```
let element =
document.lastChild.getElementsByTagName("p")[0].childNodes[1].
innerHTML;
```

U prethodnom primjeru, posljednje dijete *document*-a (tj. dijela *body*), prvi paragraf (*p*-tag), njegovo drugo dijete, unutrašnji HTML. Pristup CSS-u se vrši kroz svojstvo (eng. *property*) *style* ali treba obratiti pažnju na tipove podataka i na to da se neki atributi ne zovu isto. Primjer:

```
document.getElementById("celija").style.backgroundColor = "red";
```

HTML tagovi *div* i *span*, koji nemaju nikakav vizuelni efekat, posebno se često koriste kako bi se nekom dijelu stranice pridružio *id* radi manipulacija putem DOM-a. Postoje i druge metode za manipulaciju elementima DOM stabla kao što je:

- **document.querySelector** - metoda koja vraća prvi element unutar HTML dokumenta koji odgovara specificiranom selektoru, ili grupi selektora. Ako se ne pronađe takav element, onda se vraća *null*. Parametar ove metode je CSS selektor *string* pomoću kojeg možete pretraživati dokument po nazivu taga, klasi ili *id*-u a vraća se prvi element koji odgovara upitu. Klasa počinje znakom tačka (.) a *id* znakom *hash* (#). Primjeri:
  `document.querySelector("button")` - vraća prvi HTML element sa tagom `<button>`.
  `document.querySelector(".mojaklasa")` - vraća prvi HTML element čiji je *class* atribut "mojaklasa".
  `document.querySelector("#dugme")` - vraća HTML element čiji je *id* "dugme"
  - Parametri se mogu i kombinirati.
    Npr. `document.querySelector("p.description")` - vraća prvi HTML element sa tagom `<p>` i klasom "description".
  - Znak razmaka omogućava kretanje kroz hijerarhiju tagova. Npr. `document.querySelector("#content  img")` - vraća prvi HTML element čiji je tag `<img>` unutar elementa čiji je *id* "content"
  - Znakovima uglaste zagrade i dvotačke, mogu se definirati očekivane vrijednosti atributa. Primjer:

```
document.querySelector('#formular input[type="radio"]:checked')
```

  - vraća prvi element čiji je tag `<input>`, atribut "type" jednak "radio" a vrijednost mu je "checked", unutar elementa čiji je *id* "formular".
  - `document.querySelectorAll` vraća niz svih elemenata koji zadovoljavaju uslove selektora npr. `document.querySelectorAll("img")` vraća sve slike u dokumentu.

Primjer:

Dat je sljedeći HTML kod:

```
<!DOCTYPE HTML>
<HTML>
<HEAD>
  <META http-equiv="Content-Type" content="text/html;
charset=utf-8">
  <TITLE>WT vjezba, Uvod</TITLE>
  <SCRIPT src="uvod.js"></SCRIPT>
</HEAD>
<BODY>
  <div id="mojelement">Ovo je moj element :))</div>
  <br><br><br>
  <a href="http://www.etf.ba" onclick="return pritisnut()">Pritisni
me</a>
  <br><br><br>
  <input type="button" id="izmjena" onmouseover="return
klikDugme(this)" value="Sakrij">
</BODY>
</HTML>
```

Datoteka *uvod.js* ima sljedeći izgled:

```
window.onload = function () {
  document.getElementById('mojelement').style.fontWeight="bold";
  // postavljamo css stil font-weight na bold
}

function pritisnut() {
  let x = document.getElementById('izmjena');
  x.value="Prikazi";  // pristup atributu elementa
  document.getElementById('mojelement').style.display="none";
  return false; // osigurava da link ne bude izvrsen
}
function klikDugme(referenca) {  // pristup elementu preko this
  if (referenca.value==="Sakrij") {
    // postavi CSS property display na vrijednost "none"
    document.getElementById('mojelement').style.display="none";
    referenca.value="Prikazi";
    return false;
  }

  referenca.value="Sakrij";
```

```
  // Pristup DIV elementu mojelement preko DOM stabla
  let z = document.body.children[0];
  // ili s obzirom da je DOM stablo prvi element je DIV ispod body
taga
  z = document.body.children.mojelement;
  // moze i ovako ili document.body.children['mojelement']
  z = document['body']['children']['mojelement'];
  // moze i ovako pa cak i window['document'] :)
  z.style.display="block"; // prikazi ga ili z['style']['display']
= ...
}
```

Pokrenite prethodni primjer, isprobajte različite opcije i analizirajte koji dio radi šta i zašto.

2. Zadaci

Zadatke označene **zelenom bojom** je potrebno uraditi u toku laboratorijske vježbe. Ukoliko studenti tokom laboratorijske vježbe ne urade te zadatke, onda kući (**najkasnije 8 sati** prije početka naredne laboratorijske vježbe) moraju uraditi i zadatke označene **zelenom bojom** i zadatke označene **narandžastom bojom**. Bez obzira da li se zadaci rade na vježbi ili kod kuće, **obavezno** ih je postaviti na odgovarajući *Bitbucket* repozitorij za vježbe.

Zadatak 1. Korištenjem DHTML-a (JavaScript, CSS, DOM) napraviti interaktivno stablo predmeta. Nakon otvaranja stranice prikazuje se tekst:

+ Prva godina

+ Druga godina

Kada se klikne na znak plus ili tekst "Prva godina" ili "Druga godina" treba se "raširiti" taj dio stabla, a znak plus se treba pretvoriti u minus, tako da izgleda npr. ovako:

- Prva godina

    - IM1

    - IM2

    - OE

    - EES

    - ….

+ Druga godina

Ponovnim klikom na isto mjesto, taj dio stabla se treba ponovo skupiti. Pri tome se stranica ne treba osvježavati!

Savjet: Koristite CSS svojstvo ***display*** kako biste prikazali (*display:block*) odnosno sakrili (*display:none*) sloj (tj. *div*) sa sadržajem podstabla.

Zadatak 2. Napišite JavaScript program koji postavlja boju pozadine teksta paragrafa u crvenu.

Zadatak 3. Napišite JavaScript program koji proračunava zapreminu sfere (kugle). Primjer forme sa koje se učitavaju podaci je prikazan na sljedećoj slici: