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

Laboratorijska vježba br. 3: JavaScript (osnove)

1. Uvod

Sljedeći HTML kôd snimite kao datoteku *digitron.html* i otvorite ga u web pretraživaču.

```
<!DOCTYPE HTML>
<HTML>
<HEAD>
  <META http-equiv="Content-Type" content="text/html;
charset=utf-8">
  <TITLE>Tutorijal 3, Uvod</TITLE>
</HEAD>
<BODY>
  <H1>Digitron</H1>
  <INPUT type="text" id="sabirak1" size="5"> +
  <INPUT type="text" id="sabirak2" size="5"> =
  <INPUT type="text" id="zbir" size="5">
  <INPUT type="button" id="dugme" value=" Izračunaj"
onClick="sabiranje();">
  <SCRIPT type="text/javascript">
  function sabiranje() {
    let a = document.getElementById("sabirak1").value;
    let b = document.getElementById("sabirak2").value;
    let c=a+b;
    document.getElementById("zbir").value = c;
  }
  </SCRIPT>
  <P>Unesite sabirke i kliknite na dugme Izračunaj.</P>
</BODY>
</HTML>
```

Proanalizirajmo prethodni isječak kôda:

- Gdje se nalazi kôd koji vrši sabiranje i u kojem programskom jeziku je napisan? Kako određujemo da će se taj kôd izvršiti klikom na dugme “Izračunaj”?
- Zašto rezultat sabiranja nije tačan? Kako to možemo riješiti?

- Miješanje HTML i JavaScript kôda nije preporučljivo jer je otežano čitanje i održavanje. Modifikujte ovu stranicu tako da je sav JS kôd u zasebnoj datoteci naziva *sabiranje.js*.

Diskusija:

- JavaScript kôd funkcije *sabiranje()* se nalazi unutar `<SCRIPT>...</SCRIPT>` taga. Poziv ove funkcije specificiran je atributom *onClick* na dugmetu `<INPUT type="button">`. U dokumentaciji možete pronaći i mnoge druge korisne događaje (eng. *events*) kojima možete pridružiti JavaScript kôd, npr. *ondblclick* (dvostruki klik), *onmouseover* (prelazak kursorom miša), *onchange* (promjena vrijednosti, npr. za `<INPUT type="text">`) itd.
- Poljima forme pristupili smo preko *Document* objekta, o čemu će više riječi biti na nekoj od narednih vježbi.
- Atribut *value* je tipa *string*. Operator "+" primijenjen na vrijednosti numeričkih tipova vrši sabiranje, a na vrijednosti *string* tipa vrši spajanje stringova (eng. *concatenate*). U JavaScript-u promjenljivima ne moramo davati specifičan tip, čak se tip može i promijeniti za vrijeme postojanja varijable! Npr. ako napišete ovakav kôd:

```
let x = 1+2; // Numericki tip
x = "broj "+x;
alert(x);
```

- Na ekranu će se ispisati tekst: "broj 3". Promjenljiva *x* je prvobitno bila numeričkog tipa, ali kada smo primijenili operator "+" na *string* i broj, rezultat će biti *string* (broj se najprije pretvara u *string*), pa je taj rezultat pridružen varijabli *x* te je i ona postala tipa *string*.
- Za pretvaranje *string* u *int* možemo koristiti funkciju *parseInt*.
- Za izdvajanje JavaScript kôda u zasebnu datoteku koristimo sljedeći HTML tag:

```
<SCRIPT src="sabiranje.js"></SCRIPT>
```

- Ovaj tag treba postaviti u HTML dokument tamo gdje želite da se JavaScript kod izvrši (ako je sav kod u funkcijama, onda je svejedno). No ostaje činjenica da se u *onClick* atributu nalazi JavaScript kôd. Preporučljivo je da se pridruživanje *event*-a dugmetu vršite na ovaj način:

```
// Dugme sa ID-em "dugme"
let dugme = document.getElementById("dugme");
// Funkcija koja se izvršava klikom na dugme
dugme.addEventListener( "click", function( ev ) {
alert( "Zdravo" );
}, false);
```

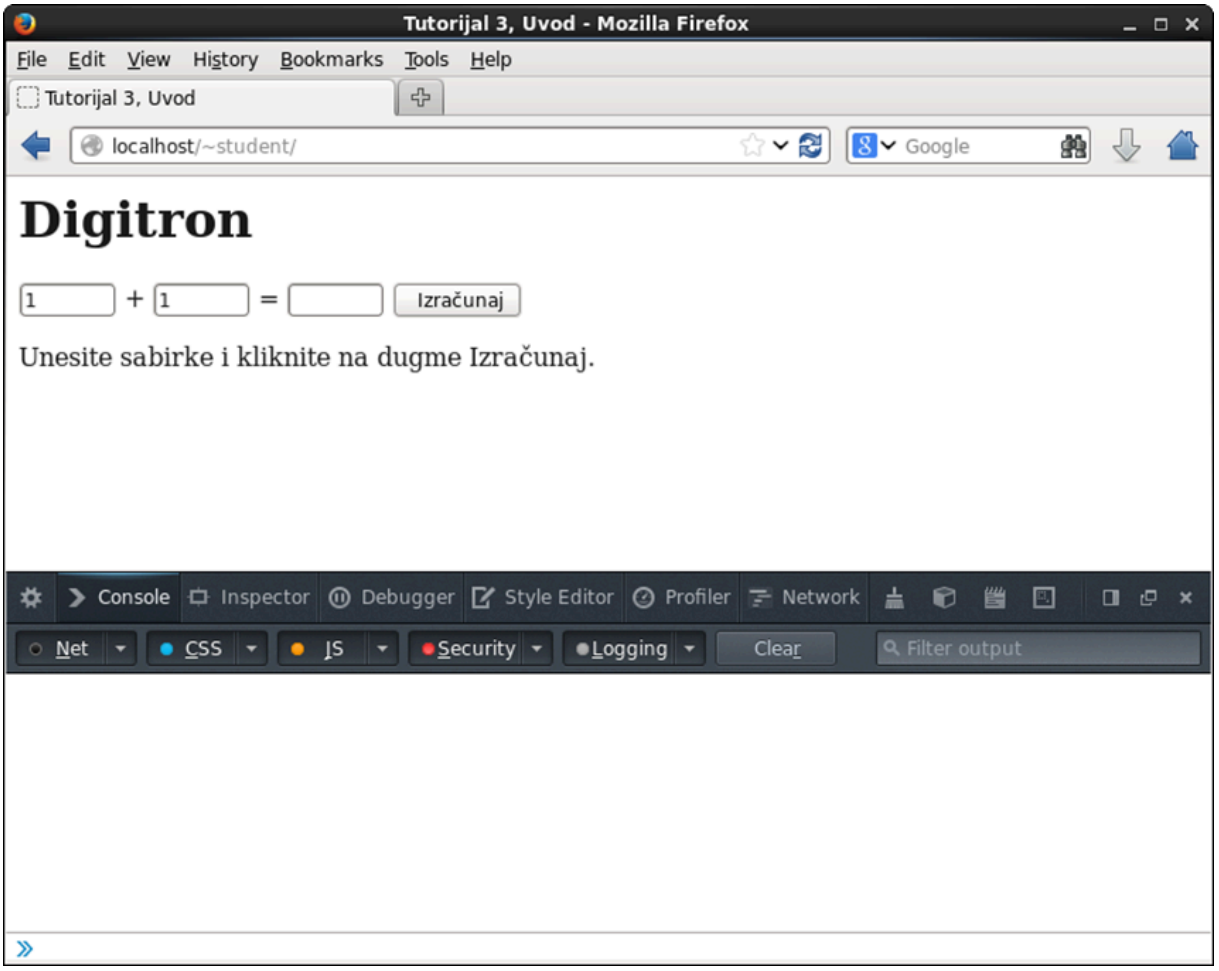
- Na primjeru iznad možemo vidjeti anonimnu funkciju ili funkcijski izraz. Ova funkcija je deklarirana unutar samog poziva *dugme.addEventListener*, nije joj dodijeljeno ime (tako da se ne može pozivati osim klikom na dugme), prima jedan parametar (*ev*) koji nije iskorišten, a tijelo te funkcije je jedna naredba: *alert("Zdravo")*. Anonimne funkcije su vrlo često korištene u JavaScript programima.

Također možemo i pridružiti funkciju varijabli:

```
let nesto = function( ev ) { alert( "Zdravo" ); }
nesto();
```

2. Web developer tools

Pritiskom na dugme *F12* na tastaturi ili putem menija u gornjem desnom dijelu web pretraživača (nakon klika na opciju *Customize and control* koja je obično označena sa tri tačke ili tri ravne crte, birajte *More tools -> Developer tools*), otvara se alat naziva *Web Developer Tools*.



Slika 2.1. Izgled Developer tools

- Kartica *Console* nudi osnovnu JavaScript komandnu ljusku. U ovom prozoru možete otkucati neku komandu (npr. *alert('Zdravo')*), te vidjeti greške i upozorenja koje je proizveo JavaScript interpreter. Inače, u slučaju greške interpreter jednostavno prestane raditi.
- Kada radite *debugging* vaše skripte, često je potrebno da ispišete vrijednosti nekih varijabli. Možete ih upisati u prozor konzole funkcijom *console.log('Proba')*.
- Kartica *Inspector* vam omogućava da kliknete na bilo koji element web stranice, saznate o kojem tagu/tagovima se radi, gdje se nalaze na stranici, te da direktno mijenjate CSS svojstva.
- *Debugger* pruža uobičajene opcije za *debugging* JavaScript kôda kao što su *breakpoints*, *step/trace*, *watches* itd. Kod ovoga treba biti pažljiv jer se zna desiti da se kompletan pretraživač zaglavi.
- *Style Editor* je zgodnija varijanta za uređivanje CSS stilova.
- *Profiler* omogućava da pratite vrijeme izvršenja pojedinih dijelova (funkcija) vašeg JS

kôda kako biste optimizirali program.

- *Network* nudi informaciju o mrežnoj komunikaciji (upućeni zahtjevi prema serveru ili serverima te koliko je koji trajao).

U zavisnosti od web preglednika koji koristite, kartice i pojedine opcije mogu imati različite nazive i mogu biti locirani na različitim mjestima.

Zadatak 1. Pokrenite sljedeći JavaScript kôd. Da li razumijete šta se ovdje dešava i zašto?

```
let odgovor = prompt("Kako se zoves?", "Imenom i
prezimenom(default)");
if (odgovor!=null && odgovor!="")
{
     let r=confirm("Pritisnite OK da prikazete ime u alertboxu a
Cancel za prikaz direktno na stranici");
     if (r==true)  // ili if(r)
           alert(odgovor);
     else
           document.write(odgovor);
}
```

Doradite uneseni kôd na sljedeći način. Postavite korisniku pitanje da unese neki tekst korištenjem *prompt* dijaloga. Nakon unosa, tekst ispisati u *alert box*-u obrnutim redoslijedom. Npr. ako korisnik unese "ovo je test" onda je potrebno ispisati "tset ej ovo".

3. 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 2. Korištenjem *document.write* i petlje napraviti JavaScript funkciju koja u trenutni HTML dokument dodaje tablicu množenja 10x10 koja izgleda kao na slici u nastavku (obratite pažnju da je na slici prikazan samo gornji lijevi ugao tablice):

X	1	2	3	4	...
1	1	2	3	4	5
2	2	4	6	8	10
3	3	6	9	12	15
4	4	8	12	16	20

Za stiliziranje tabele koristiti CSS koji se, kao i JavaScript, treba nalaziti u zasebnoj datoteci. Korištene boje su: *#CCCCCC* (siva) i *#FFF2CC* (žućkasta). Font je *sans-serif*. Zaglavlja redova i kolona su boldirana. Rubovi tabele su širine *1 piksel*.

Savjet: Za ovaj zadatak nemojte koristiti dugme koje iscrtava tabelu na *click*! Naime, *document.write* metoda se može koristiti samo za vrijeme učitavanja HTML dokumenta jer u suprotnom zamjenjuje tekući dokument novim (čime ujedno gubimo CSS koji je tražen zadatkom). Ispravan način izmjene aktuelnog dokumenta je putem DOM manipulacije odnosno atributa *innerHTML* što je tema sljedećeg tutorijala.