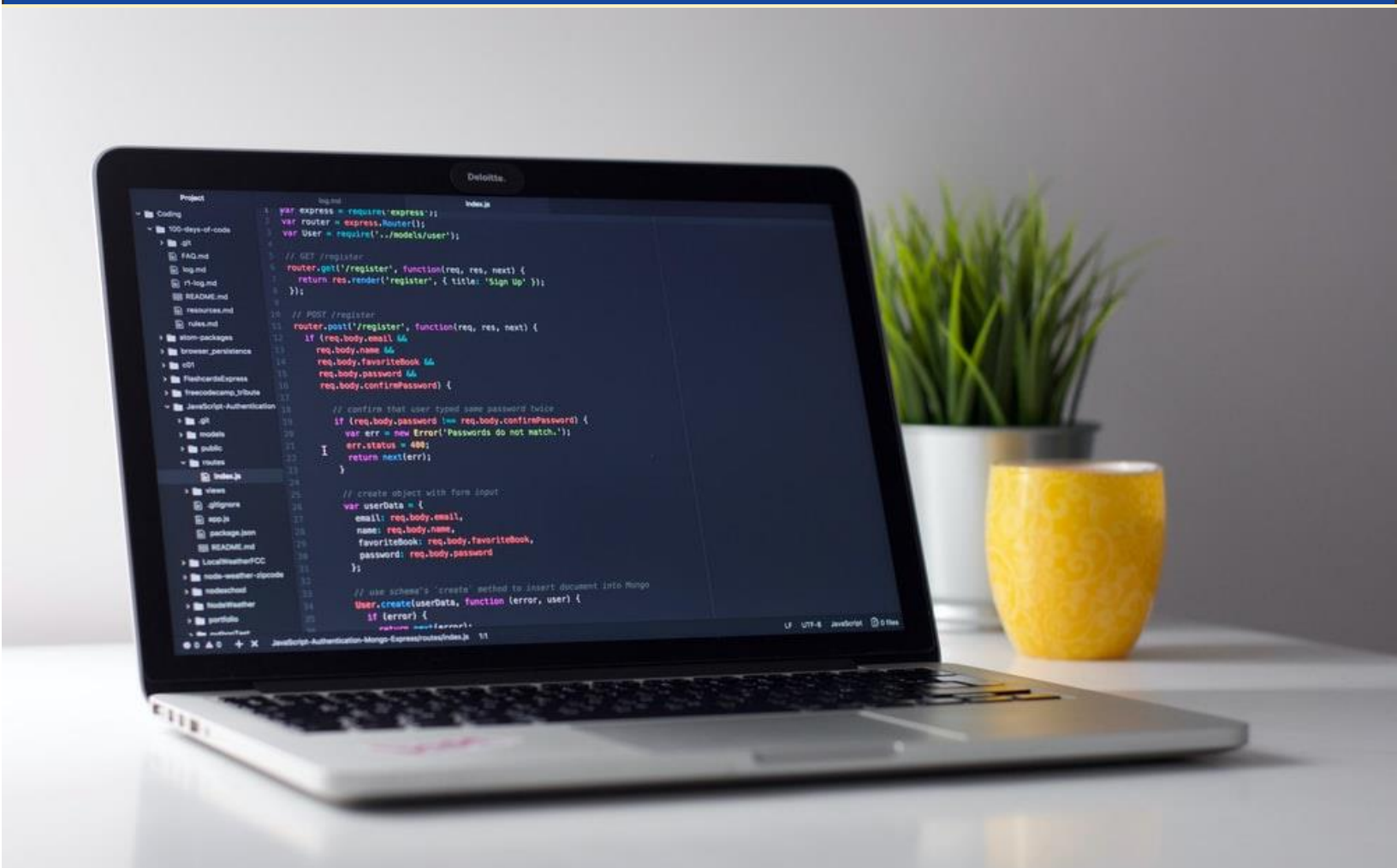
BH Futures Foundation

BUILD WEBSITE FROM SCRATCH WITH HTML, CSS & JAVASCRIPT



SADRŽAJ

O RADIONICI

U zadnjih dvadesetak godina Web stranice su jedan od glavnih načina digitalne komunikacije. Za razliku od standardnih medija, Web nudi mogućnost objave sadržaja i to na jednostavan način. Preporučeno prethodno znanje za ovu radionicu su razumijevanje šta je internet pretraživač, korištenje pretraživača, ponavanje osnova rada na računaru (kreiranje, izmjene i brisanje datoteka).

Radionica je namijenjena svim osobama koje žele da se upoznaju sa kreiranjem statičkih web stranica. Kroz ovu radionicu ćete naučiti korištenje raznih opcija za kreiranje i uređivanje web stranica, kao i njihovih elemenata na moderan način, pored čega ćete imati priliku naučiti pisati uredan, jednostavan kod.

OSNOVNI POJMOVI

Šta je to WEB?

Prije nego što pređemo na samu radionicu , potrebno je izdvojiti nekoliko riječi o ključnoj stvari ovdje bez čega ništa od ovog ne bi bilo moguće – o samom internetu.

Internet je velika računarska mreža i sistem za jednostavnu i efektivnu komunikaciju sa tekstom, slikom i zvukom. Neki od najpopularnijih dijelova interneta su: World Wide Web, E-mail, razmjena dokumenata i sl. Koristimo ga više nego ikad prije, na mjestima gdje možda i ne vidimo odmah, jer je internet više od Web stranica koje posjećujemo unošenjem *URL-a* u *internet pretraživač*; da li je to provjeravanje email-a na tvom telefonu ili objavljivanje slike na Instagramu – ti koristiš internet.

Kako Web stranice funkcioniraju?

Počet ćemo sa korištenjem interneta koje je najviše očito. Posjećuješ *Web stranicu* Google unošenjem *URL-a* : www.google.com.

Web stranica je ustvari jedan web dokument koji je pogodan za *World Wide Web* (puno ime za skraćenicu www u samom URL-u) i internet pretraživač (Google Chrome, Firefox, Safari i slično).

World Wide Web (www) – može se prevesti i kao 'svjetska mreža' - je jedna od najkorištenijih usluga interneta koja omogućava pregled hipertekstualnih dokumenata, tj. dokumenata koji mogu sadržavati tekst, slike i multimedijalne sadržaje, a međusobno su povezani tzv. hiperlinkovima.

Internet pretraživač je program koji korisniku omogućava pregled Web stranica i multimedijalnih sadržaja vezanih uz njih. Najpopularniji pretraživači danas su Firefox, Google Chrome, Safari, Opera itd.

URL je skraćenica za pojam Uniform Resource Locator, odnosno, putanja do određenog sadržaja na Internetu te se obično naziva link, poveznica, a ponekad i mrežna adresa.

Od trenutka kada uneseš adresu u svoj pretraživač i klikneš ENTER do trenutka prikazivanja Web stranice, puno stvari se desi :

1. URL se prevodi

Kod za Web stranicu se ne nalazi na tvom računaru, zato ga je potrebno preuzeti sa drugog računara na kojem je pohranjen. Taj 'drugi računar' se naziva *server* (poslužitelj).

Server je računar ili softver koji šalje i prima podatke od više klijenata (u našem slučaju, klijent je naš Web pretraživač).

Iako smo unijeli www.google.com (koja se može nazvati i domenom), server na kojem je ta Web stranica nju identificira na osnovu *IP* (Internet Protocol) adrese. Web pretraživač šalje 'zahtjev' serveru sa IP adresom koju smo unijeli (odnosno našu domenu).

IP je jedinstveni broj koji je dodjeljuje svakom uređaju na računarskoj mreži koje za komuniciranje koriste internetski protokol. IP adresa uglavnom izgleda kao 172.56.180.5.

Kako se domena www.google.com prevodi u svoju IP adresu?

Postoji posebna vrsta servera na internetu koja se naziva DNS – Domain Name System. Posao DNS-a jeste da prevodi domenu u IP adresu, najlakše ih je zamisliti kao velike rječnike kod kojih se u jednoj koloni nalazi domena, a u drugoj njoj odgovarajuća IP adresa.

Kada smo unijeli domenu, pretraživač prvo dohvata IP adresu od DNS servera.

2. Zahtjev je poslan

Kada se IP adresa prevela, pretraživač nastavlja sa radom i šalje zahtjev serveru sa tom IP adresom. Taj zahtjev sadrži dosta podataka, kao što su tačan URL, koja je vrsta zahtjeva i slično. Podaci se šalju uz pomoć HyperText Transfer Protocol-a (HTTP).

HyperText Transfer Protocol je standardizirani protokol koji definira kako bi zahtjev (i odgovor) trebao izgledati, koji podaci mogu biti uključeni i kako zahtjev treba poslati.

Pošto se koristi HTTP, cijeli URL izgleda : http://www.google.com.

Server nakon što je primio i obradio zahtjev, šalje odgovor. Odgovor je sličan zahtjevu, možemo ga čak nazvati i zahtjev u obrnutom smjeru. Kao i zahtjev, odgovor sadrži različite podatke. Ako pretražujemo stranicu poput Google, odgovor će sadržavati kod koji je potreban da se stranica na ekranu, ali odgovor ne mora sadržavati samo kod, može da sadrži bilo koje podatke, uključujući i dokumente ili slike i sl.

3. Odgovor je učitan

Web pretraživač je dobio odgovor od servera, ali odgovor ne prikazuje ništa na ekranu. Umjesto prikazivanja, sljedeći korak je da pretraživač učita odgovor, kao što je server uradio sa zahtjevom.

Pretraživač provjerava podatke koji se nalazu u sklopu odgovora i na osnovu toga odlučuje šta će dalje raditi. Npr. može se desiti da se PDF dokument otvori u pretraživaču. To je zato što je odgovor informirao pretraživač da su podaci PDF dokument, a ne Web stranica.

Kada je u pitanju naša Web stranica (Google), odgovor bi trebao da sadrži specifični dio meta podataka koji govore pretraživaču da su podaci koji se nalaze u odgovoru ustvari text/html tip.

HTML je Hyper Text Markup Language i on definira strukturu Web stranice. O HTML ćemo detaljnije u nastavku.

Pretraživač sada zna kako da učita HTML kod i prolazi kroz njega te prikazuje stranicu.

4. Stranica je prikazana

Kao što smo spomenuli u prethodnom koraku, pretraživač prolazi kroz HTML kod koji je dobio kao dio odgovora od servera i prikazuje stranicu na osnovu toga.

Ono što je bitno napomenuti jeste da HTML kod ne sadrži instrukcije kako će stranica izgledati, on samo definira strukturu i 'govori' pretraživaču koji dio je naslov, slika, paragraf i slično.

Za izgled su zadužene druge tehnologije kao što je CSS o čemu ćemo detaljnije u nastavku.

U ova četiri koraka smo objasnili šta se dešava 'iza scene' kada unesemo određeni URL u Web pretraživač do trenutka kada je Web stranica prikazana.

HTML – HYPER TEXT MARKUP LANGUAGE

Web stranice se sastoje od različitih tipova sadržaja, poput teksta, slika, formi ili audio i video zapisa. Svaka internet stranica je različita te njen izgled i funkcija ovise o tome kako je pisan kod, ali ipak u jednom segmentu Web stranice imaju nešto zajedničko, a to je **HTML**, tj. Hyper Text Markup Language.

HTML je osnovni gradivni element svake Web stranice i to je jezik za 'označavanje' kojim se određuje struktura, sadržaj i funkcija nekog HTML dokumenta, odnosno Web stranice. Prvi HTML dokument je kreirao Tim Berners-Lee 1990. godine.

Pojam "Hypertext Markup" upućuje na jezik za označavnje, te mogućnost međusobnog povezivanja dokumenata hiper-poveznicama (engl. Hyperlink).

Označavanje se vrši korištenjem tagova kojima se stvaraju, povezuju i strukturiraju elementi HTML dokumenta. Tagovi upućuju Web pretraživač na način kako će prikazati tekst koji slijedi nakon taga. HTML datoteka mora imati ekstenziju .html, te može biti kreirana korištenjem bilo kojeg tekst editora.

CSS – CASCADE STYLE SHEETS

CSS je skraćenica od "Cascading Style Sheets", a služi za definiranje stilova koji određuju izgled HTML elemenata (font, boje, pozadine, razmake i slično). Ti stilovi se nadovezuju u "Style Sheets", eksterne fajlove sa .css ekstenzijom, ili jednostavno se pišu u zaglavlju HTML dokumenta ili čak inline, tj. na samim elementima. Eksterni stilovi su i najbolji jer vam omogućuju uštedu vremena pri radu i pri redizajniranju HTML dokumenta, stoga ćemo i mi u nastavku koristiti eksterni stil.

STRUKTURA HTML DOKUMENTA

Kao i bilo koji drugi dokument, HTML dokument ima određenu strukturu. HTML dokument tako može imati naslov, odlomak, slike, tabele i sl.

U nastavku se nalazi vrlo jednostavan primjer HTML dokumenta koji ima naslov i odlomak (HTML elementi su označeni zelenom bojom):

```
<html>
     <body>
          <h1>  Ovo  je  naša  prva  Web  stranica</h1>
          <p> A ovo je neki tekst o našoj Web stranici</p>
     </body>
</html>
```

DOCTYPE DEKLARACIJA

DOCTYPE deklaracija opisuje koja se verzija HTML-a koristi u dokumentu. Potrebno je navesti na samom početku HTML dokumenta, prije svih drugih elemenata.

DOCTYPE deklaracija nije HTML element, nego instrukcija pretraživaču da bi on znao koju verziju HTML-a treba interpretirati. Dolaskom verzije HTML5, DOCTYPE deklaracija je postala jednostavna. Dakle, dovoljno je na početku dokumenta napisati sljedeće:

```
<<!DOCTYPE html>
```

OSNOVNA STRUKTURA HTML DOKUMENTA

Svi HTML dokumenti, neovisno od kompleksnosti i sadržaja, imaju neke zajedničke dijelove pomoću kojih se definira struktura HTML dokumenta. Elementi koji su zajednički svim HTML dokumentima su: <html> </html>, <head> </head>, <title> </title> i <body> </body>. Dakle, element koji je obavezan na svakoj stranici je html. To je prvi element unutar kojeg se nalazi cijeli sadržaj nekog HTML dokumenta, a služi kako bi web pretraživač „znao“ prepoznati da li je riječ o HTML dokumentu. Budući da je cijeli sadržaj dokumenta unutar ovog elementa, cijeli dokument počinje i završava oznakama <html> i </html>.

Na primjer:

```
<html>
     <head>
          <title> HTML </title>
     </head>
     <body>
     </body>
</html>
```

Na vrhu HTML dokumenta se navodi HTML element ***head***. On sadrži informacije o stranici koje se ne prikazuju kao dio sadržaja stranice. Taj element predstavlja zaglavlje HTML dokumenta. U head elementu se nalazi element ***title*** u kojeg pišemo naslov Web stranice. Naslov stranice se prikazuje na naslovnoj traci pretraživača i ne mora biti identičan

naslovu HTML dokumenta koji se pohranjuje lokalno, na računaru. Cijeli sadržaj, koji se nalazi unutar elementa ***body***, je sadržaj koji će se prikazati u pretraživaču. U prethodnom primjeru će se prikazati prazna stranica, bez sadržaja . Kada se između oznaka *body* upiše tekst, on će postati sadržaj stranice, pa će u pretraživaču biti vidljiv. HTML dokument se čuva, za početak, lokalno na računaru kao datoteka sa ekstenzijom .html: npr. index.html.

STRUKTURA HTML ELEMENTA

HTML element se sastoji od: početne i završne HTML oznake (eng. tag). Većina HTML elemenata se sastoji od početnog tag-a (npr.) i završnog tag-a ( ). Postoje elementi koji nemaju krajnji tag (<img/>).

Sadržaj se unosi između dvije HTML oznake, te se one na neki način ponašaju poput spremnika koji pojašnjava koja vrsta informacije se nalazi u istom. Na primjer, unutar HTML oznaka ** ** i ** ** će se nalaziti jedan odlomak teksta. Kako bi razjasnili strukturu HTML-ovog elementa, potrebno ga je razdvojiti na osnovne dijelove.

Svaki početni tag se sastoji od šiljastih zagrada i identifikatora.

Identifikator određuje o kojem elementu HTML-a se radi.

Završni HTML element se sastoji od: šiljastih zagrada u kojima se nalazi kosa linija (eng. slash) i identifikator. Kosa linija je oznaka koja pokazuje da je to završni element , a ne početni. Na primjeru jednostavnog HTML koda koji se sastoji od jednog odlomka teksta se može vidjeti kako izgleda ispravno napisan odlomak teksta u HTML-u:

```
<p> Lorem ipsum dolor sit amet, consectetur adipiscing elit.
Fusce vel consequat ante. In commodo ultrices neque et interdum.
Morbi fermentum ipsum volutpat sem pulvinar feugiat. Etiam
iaculis vulputate aliquot. </p>
```

HTML ATRIBUTI

Neki HTML elementi mogu imati dodatna svojstva (unutar početne HTML oznake) koja daju više informacija o sadržaju elementa, a najčešće se sastoje od dva dijela: *imena* i *vrijednosti*, odvojeni znakom „=“. Ta svojstva se nazivaju ***HTML atributima***.

Jedan HTML element može imati više atributa. Ime atributa je podatak koji govori o kojoj vrsti informacije je riječ u atributu, dok je svojstvo ili informacija vrijednost atributa. Ime atributa bi se trebalo pisati malim slovima, a vrijednost atributa se uvijek mora nalaziti u dvostrukim navodnicima. Većina atributa se koristi sa tačno određenim HTML elementom, pa nije striktna obaveza. Ipak, postoje HTML elementi koji moraju imati i atribut, jer u suprotnom nije moguća pravilna interpretacija elementa.

Primjer neizostavnog atributa je src (engl. source), a koristi se za označavanje putanje do slike. HTML element se sa skupom atributa može vidjeti na primjeru elementa za umetanje slike <img>. U ovome primjeru oznaka ima tri atributa.

Na primjer:

```
<img src=”img/pic.png” height=”30” width=”30” />
```

Atribut: src=”img/pic.png”

Ime atributa: src

Vrijednost atributa: img/pic.png

HTML ELEMENTI

BLOCK I INLINE ELEMENTI

Prema načinu prikaza, u HTML-u postoje dvije osnovne vrste elemenata. Prva vrsta su block elementi. Oni zauzimaju cijeli red u kojem se nalaze. Moguće im je odrediti visinu i širinu. Čak i ako im se smanji širina, oni će i dalje zauzimati cijeli red, pa će zato naredni element započeti u novom redu.

Primjeri takvih elemenata su div, p i h1.

div
h1, h2, h3, h4, h5, h6
p

Primjeri takvih elemenata su ***div, p i h1***. Element ***div*** služi za grupiranje sadržaja (teksta i HTML elemenata), element ***p*** služi za definisanje paragrafa, a element ***h1*** služi da bi postavili naslov prvog nivoa (***h2*** drugog nivoa, itd). Element ***div***, paragraf i naslov će da zauzmu cijeli red, odnosno cjelokupnu širinu Web pretraživača.

Primjeri korištenja ovih elemenata u našem kodu:

```
<div>
      <h1>Gimnazija "Dr. Mustafa Kamarić"</h1>
      <h2>Gračanica</h2>
</div>
```

Ovdje smo uz pomoć ***div*** elementa grupirali naslov prvog i drugog nivoa kako bismo ih mogli prikazati kao jedan blok element.

```
<p>
    Copyright &copy; 2020 by Bosnia and Herzegovina Futures
Foundation. All rights reserved.
  </p>
```

U ovom isječku koda smo koristili ***p*** element kako bismo mogli prikazati tekst u obliku paragrafa (***p*** uglavnom koristimo kada želimo prikazati neki tekst koji i nije toliko bitno naglasiti na našoj Web stranici).

Druga vrsta su linijski elementi (engl. inline elements). Takvi elementi su prikazani unutar trenutne linije teksta. Oni ne zauzimaju cijeli red nego onoliko prostora koliko zauzima njihov sadržaj. Primjeri takvih elemenata su ***span***, ***em*** i ***strong***.

Element ***span*** služi za označavanje dijela teksta (najčešće kako bi se na njega mogao primijeniti određeni CSS stil), dok elementi ***em*** i ***strong*** služe također za označavanje, odnosno naglašavanje dijela teksta, te se prikazuje sa stilom koji je predefinisan.

Ovi elementi neće uzrokovati prelaženje u novi red, već njihova veličina zavisi od sadržaja (prvenstveno teksta) koji se nalazi unutar njih.

span em strong

U slučaju da je unutar jednog odlomka potrebno započeti tekst u sljedećem redu (bez dodatnih razmaka između reda i ostalog teksta), koristit će se element za prelazak u novi red ***br*** (engl. break row ili breaking rule).

Na primjer:

```
<div>
     <p> Copyright &copy; 2020 by Bosnia and Herzegovina Futures
Foundation.
          <br />All rights reserved.
     </p>
</div>
```

LISTE

Liste su nizovi elemenata koji su međusobno povezani u smislenu grupu. Liste po načinu označavanja i svrsi mogu biti:

1. numerirane (uređene) liste – elementi su označeni rednim brojem;
2. neuređene liste – elementi su označeni simbolom;
3. definicijske liste – elementi popisa sastoje se od pojmova i njihovih definicija;
4. ugniježdene liste – popisi koji imaju više nivoa.

U toku radionice ćemo koristiti samo neuređenu listu.

Neuređene liste su oni popisi čiji su elementi označeni simbolima. U praksi se najčešće koriste kada nije potrebno navesti redoslijed elemenata u popisu, tj. kada su svi elementi popisa jednako važni ili nemaju hijerarhijski odnos. Popis sa grafičkim oznakama se izrađuje pomoću HTML elementa ***ul*** (engl. unordered list), a za definisanje pojedinih elemenata popisa koristi se element ***li*** (engl. list item).


```
<ul>
  <li>O nama</li>
  <li>Zanimljivosti</li>
  <li>Slike</li>
  <li>Kontakt</li>
</ul>
```

LINKOVI

Osim prikazivanja sadržaja, glavna odlika svake Web stranice je mogućnost povezivanja dijelova sadržaja, te prelazak sa jedne stranice na drugu stranicu. Bez ovoga svojstva svake Web stranice, mogućnost pregledanja stranica ili surfanje ne bi uopće postojalo. Način na koji se sa jedne stranice može preći na drugu odvija se pomoću linka ili veze (poveznice).

Postoji nekoliko vrsta linkova:

- linkovi koji vode sa jedne Web stranice na drugu;
- linkovi koji vode sa jedne stranice na drugu, unutar iste Web domene;
- linkovi koji vode sa jednog dijela teksta na drugi dio teksta na istoj stranici;
- linkovi koji se otvaraju u novom prozoru pretraživača;
- linkovi koji pokreću program za slanje e-pošte.

Linkovi se izrađuju korištenjem HTML elementa a. Bilo kakav tekst (ili neki drugi element kao na primjer slika) koji se nalazi između početne oznake <a> i završne oznake , postat će link, što znači da će se na taj tekst ili sliku moći kliknuti mišem. Klikom na link prelazi se na neko drugo mjesto. O kojem je mjestu riječ, definiše se korištenjem atributa ***href***. Atribut mora imati formu oblika ***href*** = ***"URL***".

Na primjer:

```
<a href="https://www.facebook.com/gdmkgracanica/"
     <img src="./assets/facebook.png">
               Gimnazija "Dr. Mustafa Kamarić" Gračanica
</a>
```

Tekst između početne i završne oznake je tekst linka i on će se na web stranici prikazati tako kako je napisan.

SLIKE

Slike se ubacuju u HTML dokument korištenjem tag-a ***img***. Ovaj tag ima mnogo atributa, a mi ćemo spomenuti neke najbitnije.

Obavezni atribut elementa img je ***src***, koji govori pretraživaču putanju do slike. Obično se koristi relativni URL, s obzirom na to da se slika nalazi unutar neke mape na serveru.

Na primjer:

```
<img src="./assets/logo.png" alt="Logo of school">
```

Atribut ***alt*** daje opis slike u slučaju da se slika iz bilo kojeg razloga ne može učitati u pretraživaču.

Neki od atribura za ***img*** tag su: height, width, title, align itd.

SEMANTIČKI ELEMENTI ZA GRUPIRANJE SADRŽAJA ( )

HTML element **div** je blok element koji omogućuje grupisanje više elemenata unutar jednog bloka. Moguće je napraviti jedan div element u kojem ćemo postavljati više elemenata koji logički spadaju na isto mjesto, npr. komentari korisnika, slike,...

Zatim se na tom **div** element može primijeniti tačno određeni CSS koji će omogućiti da svi elementi unutar toga div elementa izgledaju jednako. To bi značilo da će npr. svi komentari će korisniku izgledati kao dio sekcije za komentare ili da će sve slike izgledati kao galerija. Bez implementacije CSS-a, sadržaj div elementa neće biti prezentovan na neki poseban način, osim što će obavezno započeti u novom redu, sa obzirom da je riječ o blok elementu.

Element div se također koristi i za grupisanje drugih semantičkih elemenata na stranici. Na primjer:


```
<div class="row zanimljivosti-kolone">
      <div class="col span-1-of-4 kolona">
        <img src="./assets/blackboard.png" />
        <p>
          11 odjeljenja
        </p>
      </div>
</div>
```

ZAGLAVLJE I PODNOŽJE STRANICE (<header>, <footer>)

U navedenim elementima mogu se grupisati svi ostali elementi koji bi bili dio zaglavlja ili podnožja neke stranice, npr. ime web-stranice, navigacija, informacije o autorskim pravima i slično. U slučaju da se zaglavlje koristi u članku, u njemu može biti naslov članka ili nešto slično. Zaglavlje može izgledati kao u primjeru u nastavku:

```
<header>
     <nav id="navbar">
          <div class="row">
               <img    src="./assets/logo.png"    alt="Logo    of
          school" class="logo">
               <ul class="main-nav js--main-nav">
                    <li><a href="#about">O nama</a></li>
                    <li><a
                    href="#zanimljivosti">Zanimljivosti</a></l
                    i>
                    <li><a href="#slike">Slike</a></li>
                    <li><a href="#kontakt">Kontakt</a></li>
               </ul>
               <a    class="mobile-nav-icon    js--nav-icon"><i
          class="ion-navicon-round"></i></a>
          </div>
     </nav>
     <div class="text-box">
          <h1>Gimnazija "Dr. Mustafa Kamarić"</h1>
          <h2>Gračanica</h2>
     </div>
</header>
```

A podnožje na primjer:

```
<footer>
        <p>    Copyright &copy; 2020 by Bosnia and Herzegovina
    Futures Foundation. All rights reserved.</p>
</footer>
```

NAVIGACIJA (<nav>)

Element **nav** se koristi za „držanje“ navigacije na jednom mjestu, na primjer glavnu navigaciju dobro je smjestiti unutar elementa nav. Na primjer:

```
<nav id="navbar">
    <div class="row">
        <img  src="./assets/logo.png"  alt="Logo  of  school"
        class="logo">
        <ul class="main-nav js--main-nav">
            <li><a href="#about">O nama</a></li>
            <li><a
        href="#zanimljivosti">Zanimljivosti</a></li>
            <li><a href="#slike">Slike</a></li>
            <li><a href="#kontakt">Kontakt</a></li>
        </ul>
        <a      class="mobile-nav-icon      js--nav-icon"><i
        class="ion-navicon-round"></i></a>
        </div>
</nav>
```

SEKCIJA (<section>)

Element **section** omogućava grupiranje povezanog sadržaja u sekcije. Uobičajeno je da svaka sekcija ima vlastito zaglavlje. Na jednoj Web stranici može biti više sekcija. Svaka sekcija može imati nekoliko article elemenata koji su međusobno logički povezani, tj. imaju zajedničku temu ili svrhu. Trebalo bi se izbjegavati smještanje cjelokupnog sadržaja Web stranice unutar jednog section elementa. Za to je bolje koristiti element **div**.

Na primjer:

```
<section class="about">
     <div class="row">
     <h6 >
          Formirana augustovskom odlukom Skupštine opštine Gračanica, nosila naziv Todora Panića (što je u skladu sa tadašnjim političkim tedencijama), počela je raditi 10.09.1959. Veliki udio u njenom formiranju, pored ostalih, imali su: Pašaga Mandžić, Dr Mustafa Kamarić,Asim Dževdetbegović, tadašnji presjednik Opštinske skupštine, Ahmed Ćatić, Refik Hukić,Muhamed Kešetović…
         </h6>
     </div>
</section>
```

CSS-CASCADING STYLE SHEETS

CSS je jezik koji služi za oblikovanje Web stranica. Uz HTML, CSS je osnovna tehnologija na kojoj se temelje današnje Web stranice, a služi za oblikovanje sadržaja. Kroz CSS kod se definira izgled Web stranice i svih elemenata na njoj.

Uključivanje CSS koda u HTML kod

CSS kod se tipično piše odvojeno od HTML koda, tj. u posebnoj datoteci. Da bismo HTML dokument povezali sa CSS datotekom, koristimo HTML element link:

```
<link rel="stylesheet" type="text/css" href="/styles.css" />
```

Kada se taj element koristi za uključivanje CSS-a (što je zapravo njegova glavna namjena), atribut ***rel*** mora imati vrijednost stylesheet, a atribut ***type*** vrijednost text/css. Atribut ***href*** postavljamo na putanju do CSS-datoteke koju želimo uključiti.

Element ***link*** se mora uvijek nalaziti unutar HTML elementa ***head***. Ako se HTML stranica koristi većim brojem CSS datoteka, uključit će se tako da se element link navede više puta.

Mogući su i načini pisanja CSS koda u samoj HTML datoteci. Za to služi HTML element ***style***, u kojem se direktno mogu pisati CSS pravila. Element ***style*** se uvijek mora nalaziti unutar elementa ***head***.

Na primjer:

```
<style>
     p{
     color:green}
</style>
```

Drugi način na koji se CSS kod može direktno „ubaciti“ u HTML je preko atributa style, ovakav način pisanja CSS koda se naziva inline stil.

Na primjer:

```
<p style="color:red;">CSS kod</p>
```

Iako sevponekad može činiti praktičnim, miješanje CSS-a i HTML-a u istoj datoteci se ne preporučuje. Najbolji pristup je CSS kod pisati u posebnoj datoteci. Tako se ostvaruje ponovna iskoristivost koda i sažetost – određeno CSS pravilo je dovoljno napisati samo jednom, a ono će vrijediti u svim HTML dokumentima. Također, kad je potrebno nešto promijeniti u nekom CSS pravilu, dovoljno je to napraviti na jednom mjestu, a ne u svakoj HTML datoteci posebno.

Sintaksa CSS pravila

CSS kod se sastoji od CSS pravila. Primjer jednostavnog CSS pravila:

```
p{
color:red;
}
```

Ovakvo pravilo postavlja svim ***p*** elementima boju teksta na crvenu. Dio pravila koje određuje (odnosno selektuje) elemente na koje se pravilo odnosi zove se ***selektor***. Svako pravilo moramo započeti selektorom, a najjednostavniji selektor je upravo naziv elementa:

```
p{}
```

Nakon selektora dolaze vitičaste zagrade. CSS nije osjetljiv na prazan prostor pa vitičaste zagrade nije obavezno pisati u posebnim redovima, ali se to preporučuje radi čitljivosti. Unutar vitičastih zagrada se prvo navodi svojstvo koje se postavlja. U primjeru se radi o boji teksta (CSS svojstvo color):

```
p{color:}
```

Nakon što se navede svojstvo koje se želi postaviti, dolazi dvotočka, a iza nje vrijednost na koju se postavlja to svojstvo (u ovom primjeru radi se o nazivu boje):

```
p{
color:red;
}
```

Svojstvo i vrijednost zajedno čine ***deklaraciju***. Unutar jednog pravila može biti više deklaracija:

```
p{
color:red;
border: 1px solid red;
}
```

Deklaracije obavezno moramo razdvojiti pomoću tačke sa zarezom, a poželjno je svaku deklaraciju pisati u novom redu. Iza posljednje deklaracije, u CSS-pravilu nije obavezno navesti tačku sa zarezom, ali se to ipak preporučuje (radi eventualnog naknadnog dodavanja novih deklaracija unutar istog pravila).

CSS selektori

Kao što je već rečeno, svako CSS pravilo započinje selektorom, a selektor određuje za koji HTML element vrijedi to pravilo. U selektoru se mogu koristiti nazivi, klase ili identifikatori elemenata. Najjednostavniji selektor je upravo naziv elementa. Ako želimo da se pravilo odnosi na ***p*** elemente, napisat ćemo:

```
p{
color:red;
}
```

Ako želimo da se pravilo odnosi na naslov drugog nivoa; umjesto p ćemo napisati h2, ako želimo da se odnosi na naslov četvrtog nivoa, napisat ćemo h4 i slično.

Često je potrebno da se CSS pravilo primijeni samo na tačno određene HTML elemente. Da bi se to postiglo, u HTML-u te elemente treba označiti klasom određenog imena, upotrebom atributa ***class***. Na primjer:

```
<p class="zeleni-tekst">
    Zeleni tekst
</p>
```

Navedeno pravilo će vrijedit samo za elemente koji imaju postavljenu klasu zeleni-tekst. Kada se u selektoru navodi klasa, obavezno se prije imena klase mora staviti tačka:

```
.zeleni-tekst{
    color:green;
}
```

Kombinaciju koju često možemo sresti su dvije (ili više) klasa u identifikatoru. Naime, na HTML element se može postaviti više klasa, što je ponekad korisno (npr. kad se element želi dodatno stilizovati). Postavljanje više klasa na element u HTML-u izgleda ovako:

```
<p class="zeleni-tekst plavi-tekst">
    Zeleni tekst
</p>
```

Osim pomoću klase, HTML elementi se mogu označiti i identifikatorom, pomoću atributa ***id***.

```
<p id="zeleni-tekst">
    Zeleni tekst
</p>
```

Kada se u selektoru koristi identifikator, prije identifikatora se obavezno mora navesti oznaka:

```
#zeleni-tekst{
    color:green;
}
```

Ovo pravilo će se primijenit samo na element sa identifikatorom zeleni-tekst. Dakle, identifikator treba biti jedinstven– na istoj HTML stranici se ne bi trebalo dva puta pisati isti identifikator. Kod klasa nema tog ograničenja – ista klasa se može upotrijebiti samo jednom, a možemo je upotrijebiti i više puta.

Prioriteti CSS pravila

Ako želimo postaviti pravilo koje svim elementima ***p*** postavlja boju teksta na npr. plavu, napisati ćemo:

```
p{
    color:blue;
}
```

Ako želimo postaviti pravilo koje svim elementima sa klasom .crvena-boja postavlja boju teksta na crvenu, može se napisati:

```
.crvena-boja{
    color:red;
}
```

Na taj HTML element će se primijeniti oba, prethodno navedena CSS pravila:

```
<p class="crvena-boja">
    Tekst tekst
</p>
```

Budući da oba CSS pravila postavljaju svojstvo ***color***, koje od njih će imati prioritet je određeno samim selektorom. CSS pravilo čiji selektor ima veću specifičnost će imati veći prioritet, te će ono odlučiti vrijednost svojstva. Navođenje klase je specifičnije od navođenja naziva elementa, pa će stoga drugo pravilo imati viši prioritet i boja teksta u elementu će biti crvena.

Još specifičnije od klase je navođenje identifikatora, pa pravilo koje u selektoru navodi identifikator ima viši prioritet od oba navedena pravila. Kada se u selektoru nalazi kombinacija naziva elementa, klase i identifikatora, najveći prioritet ima identifikator, zatim klasa i na posljednjem mjestu naziv elementa.

Pseudoklase

Pseudoklase su izrazi u CSS-u koji omogućuju selektovanje elemenata slično kao klase. U CSS pravilima se koriste poput klasa, ali se za razliku od klasa ne postavljaju na elemente u HTMLu.

```
<ul>
    <li class="first">Jagode</li>
    <li>Narandže</li>
    <li>Banane</li>
    <li>Jabuke</li>
</ul>
```

Zatim je potrebno napisati CSS pravilo koje će stavci sa klasom .first postaviti boju na crvenu:

```
.first{
    color:red;
}
```

U ovakvim slučajevima, kada se želi primijeniti neko pravilo na temelju položaja elementa, nije potrebno uvijek napraviti novu klasu, nego je moguće koristiti neku od pseudoklasa vezanih uz položaj elementa. Pseudoklasa :first-child selektuj prvi element unutar roditeljskog elementa. Prilikom korištenja pseudoklase u selektoru treba navesti dvotačku (slično kao što se klasa u selektoru uvijek navodi sa tačkom).

U gornjem primjeru je moguće izbaciti klasu iz HTML koda, a CSS pravilo napisati na slijedeći način:

```
ul:first-child {
    color:red;
}
```

Ako se želi selektovati samo posljednja stavka liste, može se koristiti pseudoklasa .**last-child**. Ta pseudoklasa selektuje element koji se nalazi posljednji unutar roditeljskog elementa. Da bi se u posljednjoj stavci liste boja teksta postavi na zelenu, navest ćemo slijedeću deklaraciju:

```
ul:last-child {
    color:green;
}
```

Pomoću ovih pseudoklasa je moguće različito stilizovati elemente zavisno od korisničkih akcija, te tako dodatno naglasiti informaciju o tome koji je element korisnik odabrao, ili koji je link već posjetio i time mu olakšao navigaciju i interakciju sa Web stranicom. Pseudoklasa :**link** selektuje, bira link (element a) koji korisnik još nije posjetio. Na primjer:

```
a:link{
    color:green;
}
```

Pseudoklasa :**visited** selektuje link koji je korisnik već posjetio.

Pseudoklasa :**active** selektuje element na koji je korisnik upravo kliknuo (npr. link ili dugme).

Pseudoklasa :**hover** selektuje element preko kojeg korisnik trenutno prelazi mišem. To može biti link, ali i drugi element poput elemenata: **div**, **span**, **p** ili **li**.

Primjer:

```
.main-nav li a:hover,
.main-nav li a:active {
  border-bottom: 2px solid #e67e22;
}
```

Nasljeđivanje vrijednosti

Velik broj CSS svojstava se može nasljeđivati. Ako na elementu nije postavljena vrijednost za neko od takvih svojstava, koristit će se vrijednost sa roditeljskog elementa. To omogućava da se osnovna svojstva kao što su boja, veličina i vrsta fonta postave samo na jednom mjestu (najčešće na elementu body), umjesto da ih treba iznova definisati za svaki element.

Ako je na elementu body postavljena boja teksta na plavu, ona će biti plava i na svim drugim elementima, bez da je treba eksplicitno postaviti, jer su svi elementi djeca elementa body. Na elementima za koje je to potrebno, moguće je postaviti različite vrijednosti.

Modifikacija teksta

CSS nudi brojne mogućnosti za oblikovanje teksta – od osnovnih kao što je odabir boje, veličine i vrste fonta, pa do onih rjeđe korištenih svojstava pomoću kojih se može upravljati proredima i razmakom između slova. U novije vrijeme CSS nudi i mogućnosti kao što je postavljanje sjene na slova, te korištenje fontova koje korisnik ne mora imati instalirane na svom računaru, što je značajno unaprijedilo tipografiju na Web-u.

Boja teksta

Boju teksta postavljamo pomoću svojstva **color**, sa kojim smo se već susretali. Ako vrijednost boje nije postavljena, ona će biti naslijeđena od roditeljskog elementa. Vrijednost svojstva color je moguće postaviti na više načina – preko naziva boje, te preko posebnog načina zapisivanja boje – heksadecimalnog koda ili preko zapisa boja u modelima RGB ili HSL.

Najčešći način postavljanja deklaracije boje je navođenjem heksadecimalnog zapisa boje.

Na primjer:

```
h6 {
  color: #414141;
}
```

Vrste fonta

Vrsta fonta se postavlja pomoću svojstva **font-family**. Naziv svojstva **font-family** se odnosi na „familiju" fontova, npr. Arial. Unutar „familije" se nalazi više verzija fonta: Helvetica, Regular, Helvetica Bold, Helvetica Italic,... Kao vrijednost se navodi naziv fonta:

```
font-family:Helvetica;
```

Ako naziv fonta sadrži razmake (ili specijalne znakove), potrebno ga je navesti u navodnicima. Mogu se koristiti jednostruki ili dvostruki navodnici:

```
font-family:"Open Sans";
```

Veličina i debljina fonta

Veličina fonta se postavlja pomoću svojstva **font-size**. Vrijednost tog svojstva se može postaviti na više načina: pomoću ključnih riječi, pomoću apsolutnih jedinica (pikseli i tačke), te pomoću relativnih jedinica (postotci, em ili rem jedinice).

Na primjer:

```
font-size: large;
```

Predefinisane riječi za regulaciju veličine fonta su: xx-small, x-small, small, medium, large, xlarge, xx-large, larger (za jedan nivo veća slova od roditeljskog elementa) i smaller (za jedan nivo manja slova od roditeljskog elementa).

Ako veličina fonta ne postavi, tekst će se prikazivati u predefinisanoj veličini koja je kod većine preglednika 16 piksela.

Veličina se fonta se često definiše pomoću piksela. Ovdje se radi o mjernoj jedinici koja je definisana veličinom pojedine tačke (tj. piksela) na ekranu računara. Jedinica se koristi kada je potrebno precizno definisati veličinu teksta (da bi u potpunosti odgovarala zadanom dizajnu).

Na primjer:

```
font-size: 24px;
```

Nova veličina teksta koja je definisana u pikselima će ipak varirati od pretraživača do pretraživača (jer se svaki preglednik koristi vlastitim algoritmom za renderovanje fonta). Neki mobilni uređaji (npr. iPhone i neki Android uređaji) imaju vrlo visoku rezoluciju, pa se na njima tekst veličine 12 piksela prikazuje kao vrlo malen, dok je inače normalno čitljiv na stolnim računarima.

Osim apsolutnim definisanjem veličine fonta pomoću piksela (ili tačaka), korištenjem postotaka se veličina fonta može definisati u odnosu na roditeljski element. Taj način postavljanja veličine fonta je sličan kao da se koristimo ključnim riječima smaller i larger, s tim da možemo preciznije definisati razliku u veličini. Na primjer, za veličinu fonta koja je za 37.5% manja od roditeljskog elementa:

```
font-size: 62,5%;
```

Prednost relativnog definisanja veličine fonta je u tome što je na jednom mjestu moguće promijeniti veličinu fonta za sve elemente. Često se na elementu body postavi početna veličina fonta, a za sve druge elemente se veličina fonta postavi relativno u odnosu na njihove roditeljske elemente. Tako se promjenom veličine fonta na elementu body proporcionalno mijenja veličina fonta na svim elementima.

„Težina“ (pojačanje ili debljina) fonta se postavlja pomoću svojstva **font-weight**. Kao vrijednost se može navesti ključna riječ:

```
font-weight: bold;
```

Predefinisane riječi za „težinu“ fonta su: **normal** (standardna „težina“ fonta), **bold** (boldirani font), **bolder** („težina“ fonta je veća za jednu vrijednost od roditeljskog elementa – ako je dostupna težina fonta), **lighter** („težina“ fonta je manja za jednu vrijednost od roditeljskog elementa – ako je dostupna težina fonta). Ključna riječ normal će se koristit kada se želi poništiti drugo pravilo koje je postavilo težinu elementa na bold.

Vrijednost ovog svojstva se također nasljeđuje sa roditeljskog elementa. Osim ključnih riječi, koriste se i numeričke vrijednosti:

```
font-weight: 600;
```

Stil fonta se postavlja pomoću svojstva **font-style**. Kao vrijednost se navodi ključna riječ:

```
font-style: italic;
```

Na raspolaganju su slijedeće ključne riječi: **normal**, **italic** i **oblique**.

Italic fontovi su obično namjenski dizajnirani da izgledaju nakošeno, dok su fontovi oblique dobiveni zakošenjem originalnog fonta za određeni broj stepeni. Slično kao i kod „težina“, moguće je koristiti samo stilove koje korisnik ima instalirane na računaru. Većina fontova dolazi zajedno sa verzijom italic, pa se najčešće koristi ta ključna riječ. Ključna riječ normal se koristi kada se želi poništiti drugo pravilo koje je postavilo stil elementa na italic. Vrijednost ovog svojstva se također nasljeđuje sa roditeljskog elementa. Većina fontova dolazi instalirana i sa verzijom Bold i Italic, što čini mogućim da se istom elementu istovremeno postavi težina na bold i stil na italic.

Poravnanje teksta

Za određivanje poravnanja teksta se koristi svojstvo **text-align**.

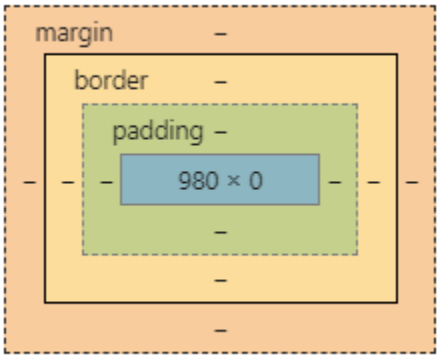
Moguće je postaviti jednu od slijedećih ključnih riječi: **left** (poravnanje uz lijevu ivicu roditelja), **right** (poravnanje uz desnu ivicu roditelja), **center** (centrira sadržaj na sredini roditelja) i **justify** (obostrano poravnanje – zauzima svu moguću dostupnu širinu roditelja). Vrijednost ovog svojstva se nasljeđuje sa roditeljskog elementa.

Na primjer:

```
text-align: justify;
```

BOX model

Svi HTML elementi se mogu smatrati kao kutije (engl. box). U CSS-u, termin „model kutije“ se koristi kada govorimo o dizajnu i izgledu. CSS model kutije je u suštini kutija koja obuhvata svaki HTML element. Sastoji se od margina, okvira, ispune i stvarnog sadržaja. Naredna slika ilustruje model kutije:



Svaki box model se sastoji iz slijedećih dijelova:

1. Sadržaja (engl. content; označen plavom bojom) - prostor u kojem se nalazi neki sadržaj, tekst, fotografije, itd;
2. Padding (označena svijetlo zelenom bojom) – prazan prostor oko sadržaja (unutrašnji dio elementa);
3. Ivice (engl. border; označena žutom bojom) - okvir ili granica koja se nalazi između padding-a i margine, a označava granicu elementa;
4. Margine (engl. margin; označena sa narandžastom bojom) – prazan, transparentan prostor izvan granica elementa

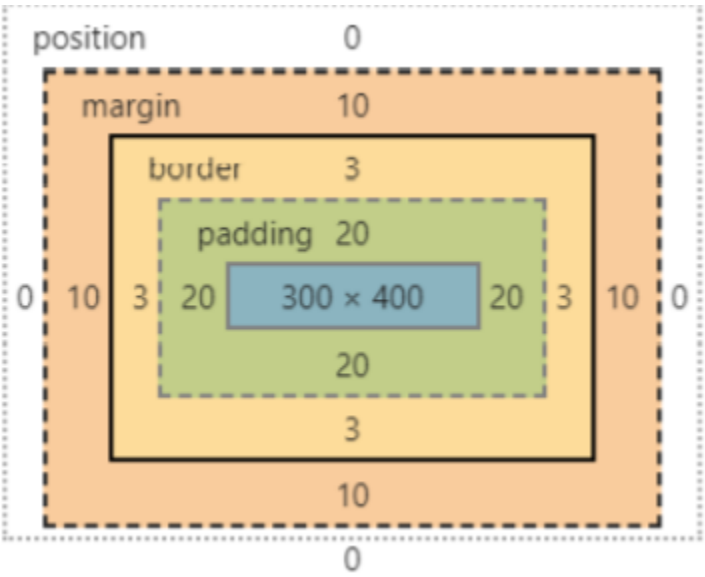
Model kutije nam omogućava dodavanje granice oko elemenata i definisanje prostora između elemenata. Model kutije je najbolje ilustrovati kratkim primjerima.

Da bismo izračunali koliko nam je ukupno prostora potrebno za jedan element, računamo to na sljedeći način:

- Ukupna širina = lijeva margina + lijevi okvir + lijevi padding + širina elementa + desni padding + desni okvir + desna margina
- Ukupna visina = gornja margina + gornji okvir + gornji padding + visina elementa + donji padding + donji okvir + donja margina

Pogledajmo jedan HTML i CSS primjer sa pravilima koja sadrže deklaracije za sva svojstva box modela (širina, visina, ispuna, okvir, margina) definisana na jednoj klasi koju smo asocijativno nazvali .**box-model**:

```
.box-model{
    width: 300px;
    height: 400px;
    padding: 20px;
    border: 3px solid greenyellow;
    margin: 10px;
}
```



Dakle, ukupna širina je:

Širina = 10 + 3 + 20 + 300 + 20 + 3 + 10 = 366 px;

Visina = 10 + 3 + 20 + 400 + 20 + 3 + 10 = 466 px.

Iz ovog kratkog primjera vidimo da naš element zauzima najmanje 366px u širinu i 466px u visinu. Ako na stranici nema toliko mjesta, naš element će biti pomjeren ili će se „preliti“ izvan svog bloka (engl. overflow), čime će se stranica po zadanim postavkama povećati za tu veličinu.

Oblikovanje elemenata

Svi HTML elementi su zapravo pravougli, odnosno zauzimaju pravougaonu površinu unutar ekrana preglednika. U ovom dijelu ćemo objasniti kako se tim pravougaonicima postavljaju boja, veličina, sjena i druga vizualna svojstva.

Boja pozadine

Da bi HTML elementi bili vidljivi, može im se postaviti boja (boja pozadine). Za postavljanje boje pozadine koristi se svojstvo **background-color**. Kao i kod svojstva color, vrijednosti se mogu postaviti na više načina. Može se koristiti naziv boje, heksadecimalni kod ili RGB vrijednosti.

Na primjer:

```
background-color: red;
```

```
background-color: #a0252a;
```

Prikaz elemenata

Prema načinu prikaza u HTML-u postoje dvije osnovne vrste elemenata. Prva vrsta su **blok elementi** (engl. block elements), a druga su **linijski elementi** (engl. inline elements), o čemu smo već detaljnije govorili.

Pomoću CSS svojstva **display** je moguće promijeniti tip prikaza na elementu. Postavi li se blok elementima svojstvo na **inline**, oni će se ponašati kao da su linijski elementi:

```
div, p, h1 {
    display: inline;
}
```

Postavi li se linijskim elementima vrijednost svojstva display na block, oni će se ponašati kao da su blok elementi:

```
span, em, strong {
    display: block;
}
```

Širina i visina elemenata

Ako se širina i visina HTML elementa ne postave eksplicitno, njegova širina i visina će zavisiti ili od sadržaja koji se nalazi unutar njega ili od roditeljskog elementa unutar kojeg se nalaze.

Širina se može podešavati samo kod block elemenata (i elemenata inline-block). Ako im se ne postavi određena širina, blok elementi će zauzeti svu širinu roditeljskog elementa. Sa druge strane, širina linijskog elementa je određena njegovim sadržajem. Postavljanje širine na tim elementima neće imati nikakav učinak. Blok elementi inicijalno zauzimaju svu širinu roditeljskog elementa. Za definiranje širine elementa koristi se svojstvo **width**. Vrijednost tog svojstva postavlja se u jednoj od mjernih jedinica dužine (pikseli, tačke, jedinice em ili rem), a može i u postotcima.

Na primjer:

```
width: 400px;
```

Smanjuje li se prozor preglednika, moguće je primijetiti da se i širina ovako definisanog elementa smanjuje. Često se takvo smanjivanje širine elementa želi ograničiti. Pomoću svojstva **min-width** je moguće definisati minimalnu širinu koju element mora zauzimati. Vrijednost minimalne širine se također može postaviti u mjernim jedinicama za dužinu ili u postotcima. Inicijalna vrijednost za ovo svojstvo je 0.

Na primjer:

```
min-width: 200px;
```

Osim minimalne širine, elementu se može definisati i maksimalna širina koju smije zauzeti. Maksimalnu širinu postavljamo pomoću svojstva **max-width**. Vrijednost maksimalne

širine se također može postaviti u mjernim jedinicama za dužinu ili u postotcima. Inicijalna vrijednost za ovo svojstvo je none, tj. maksimalna širina nije postavljena.

Na primjer:

```
    max-width: 400px;
```

Visinu možemo podešavati samo kod block elemenata (i kod elemenata inline-block). Visina se postavlja pomoću svojstva **height**. Vrijednost za svojstvo se postavlja u mjernim jedinicama za dužinu ili u postotcima. Na primjer:

```
    height: 200px;
```

Kao i kod širine, moguće je postaviti minimalnu visinu pomoću svojstva **min-height** i maksimalnu visinu pomoću **max-height.**

PROJEKAT FONDACIJE BUDUĆNOSTI U BOSNI I HERCEGOVINI

KONSTRUIRALA I OBRADILA

BERINA OMERAŠEVIĆ