

# 1 Дизајн статичких веб-страница (увод у HTML и CSS)

## 1.1 Приступи креирању веб-страница

Постоје два основна приступа креирању текстуалних и мултимедијалних докумената (укључујући и веб-странице). Код првог, названог „шта видиш то и добијеш“ тј. **WYSIWYG** (енгл. *what you see is what you get*), програми су такви да корисник на екрану види коначни изглед документа док га креира. Текст се куца тастатуром, а онда се његови делови форматирају коришћењем ГКИ, миша и пречица тастатуре. Други приступ подразумева да се текст уноси коришћењем обичног едитора текста, а форматира тако што се у самом тексту наводе специјалне инструкције за форматирање дефинисане неким специјализованим **језиком за обележавање** (енгл. *markup language*). На пример, ако се користи језик HTML намењен опису веб-страница, истакнута реч у едитор се уноси у облику `<em>реч</em>`, а ако се користи језик LaTeX, намењен припреми математичког текста, реч се уноси у облику `\emph{реч}`. Да би се добила употребљива форма текста, тј. текст из којег су уклоњена обележја и коме је на основу њих додељен одговарајући изглед, текст се обрађује специјализованим софтвером (прегледачем веба у случају језика HTML или LaTeX преводиоцем, који од LaTeX описа прави PDF документ).

Слика:

Потпис: Документ припремљен помоћу WYSIWYG система и помоћу језика за обележавање.

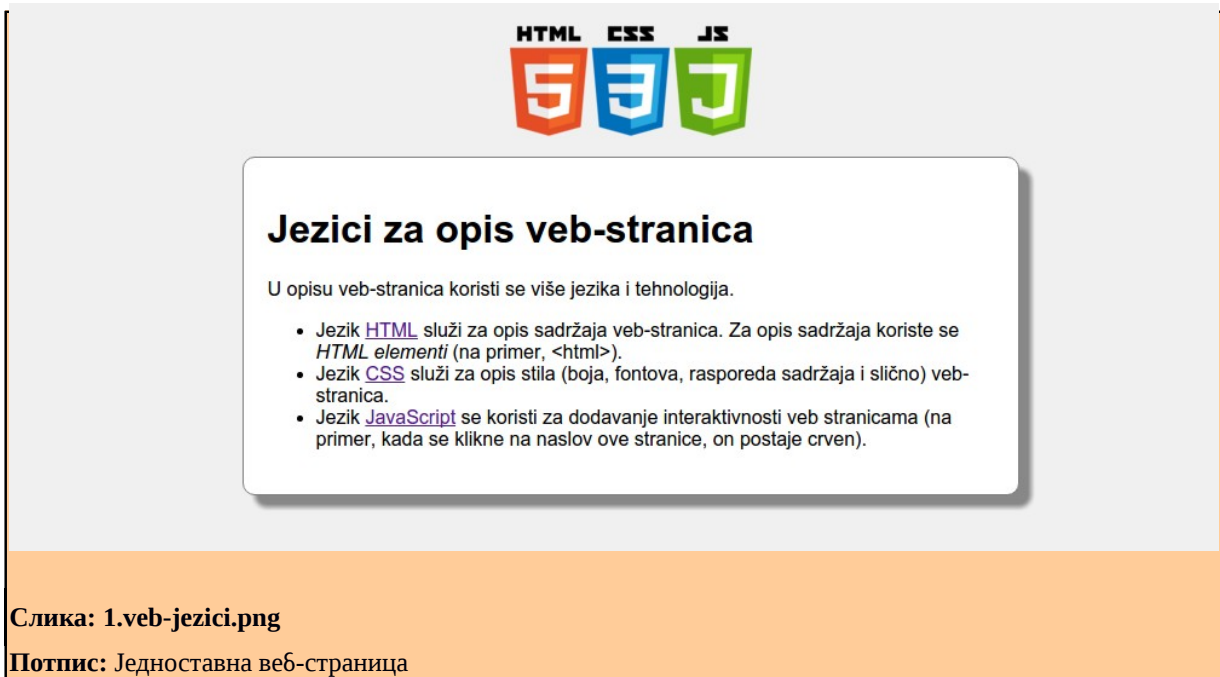
Веб-странице се описују у облику чистог текста (енгл. *plain text*), па је да је за њихово креирање директним коришћењем језика за обележавање потребно коришћење едитора текста, било оних опште намене (на пример, Notepad, Notepad++, gedit, Emacs, Vim), било оних специјализованих за веб-дизајн и веб-програмирање (на пример, sublime, WebMatrix). Постоји и значајан број програма (на пример, Adobe Dreamweaver, Microsoft Visual Studio) у којима је могуће креирање веб-страница WYSIWYG приступом – на основу акција које корисник изводи они генеришу опис у језику HTML. Чак и неки текст процесори могу да сниме документе у HTML формату. Ипак, коришћење оваквих алатки не може да одмени познавање језика за обележавање и предуслов за креирање иоле озбиљнијих сајтова је добро овладавање језицима за обележавање и програмским језицима који се користе на вебу.

Једна од честих грешака почетника је то да код који представља опис веб странице унесу у текст-процесор и сниме у формату текст-процесора (на пример, откуцају га у програму Microsoft Word и сниме под екстензијом `.doc`).

## 1.2 Језици за опис веб-страница

Циљ овог дела је да одговори на следећа питања.

- Који се језици користе за опис садржаја, изгледа и понашања веб-страница?
- Шта су основне карактеристике језика HTML, CSS и JavaScript и за шта се користи сваки од њих?



Размотримо пример једне комплетне једноставне веб-странице чији је изглед приказан на слици. Да би се та страница креирала у обичном едитору текста откуцан је следећи код (он се може видети ако веб-прегледачу издамо команду `view-source`):

```
<!DOCTYPE html>
<html>

<!-- Opis zaglavlja veb-stranice -->
<head>
  <meta charset="UTF-8" />
  <title>Kako se prave veb-stranice?</title>

  <style type="text/css">
    body {                               /* Telo dokumenta: */
      font-family: Arial;                 /* font Arial*/
      background-color: #f0f0f0;         /* sivkasta boja pozadine */
    }
    header {                              /* Zaglavlje: */
      text-align: center;                 /* centrirano poravnavanje */
    }
  </style>
</head>
<body>
  <h1>Kako se prave veb-stranice?</h1>
</body>
</html>
```

```

    main {
        background-color: white; /* Glavni deo strane: */
        width: 800px; /* bela boja pozadine */
        margin: auto; /* širina 800 piksela */
        padding: 20px; /* centriran glavni deo */
        border: 1px solid grey; /* unutrašnja margina 20 piksela */
        box-shadow: 10px 10px 5px #888888; /* okvir: 1px, puna linija, siv */
        border-radius: 10px; /* senka */
    } /* zaobljene ivice */
</style>
</head>

<!-- Opis tela veb-stranice -->
<body>
    <header>
        
    </header>
    <main>
        <h1 id="naslov">Jezici za opis veb-stranica</h1>
        <article>
            <p>U opisu veb-stranica koristi se više jezika i tehnologija.</p>
            <ul>
                <li>Jezik <a href="https://www.w3.org/TR/html5/">HTML</a>
                    služi za opis sadržaja veb-stranica. Za opis sadržaja
                    koriste se <i>HTML elementi</i> (na primer, &lt;html&gt;).</li>
                <li>Jezik <a href="https://www.w3.org/Style/CSS/">CSS</a>
                    služi za opis stila (boja, fontova, rasporeda sadržaja i
                    slično) veb-stranica.</li>
                <li>Jezik
                    <a href="https://en.wikipedia.org/wiki/JavaScript">JavaScript</a>
                    se koristi za dodavanje interaktivnosti veb stranicama (na
                    primer, u njemu ćemo isprogramirati da kada se klikne na
                    naslov, on promeni boju).</li>
            </ul>
        </article>
    </main>

    <script type="text/javascript">
        // kada se klikne na naslov, on postaje crven
        document.getElementById('naslov').onclick =
            function() {this.style.color = 'red'};
    </script>
</body>

</html>

```

У овом тренутку овај кôд нећемо потпуно детаљно анализирати, али ћемо га употребити да вам на примеру њега прикажемо основне појмове и технологије (језике) који се користе у савременом веб-дизајну. У оквиру овог кода можемо јасно идентификовати три значајне целине (које смо, прегледности ради обојили различитим

бојама).

**Пробај ово на рачунару.** Ова веб-страница је доступна у електронском додатку. Покушај да је отвориш и свог прегледача веба, а затим да видиш и њен изворни код помоћу опције `view-source`.

У коду се препознаје текст који се види на приказаној страници (на пример, наслов `Jezici za opis veb-stranica` или реченица `U opisu veb-stranica koristi se više jezika i tehnologija.`), али видимо да је он окружен мноштвом додатних ознака. На пример, наслов странице (који се не види у самом телу странице, али се види у наслову картице у којој је та страница приказана у веб-прегледачу) описан је помоћу кода `<title>Kako se prave veb-stranice?</title>`, реченица коју смо навели је смештена у први пасус који је описан између ознака `<p>` и `</p>`, док је слика која је приказана у врху странице описана помоћу кода `` и окружена ознакама `<header>` и `</header>`, што указује на то да она чини заглавље садржаја странице. Код `<a href="https://www.w3.org/TR/html5/">HTML</a>` означава да се на текст `HTML` постави линк који води ка наведеној веб-адреси. Да би се научило прављење веб-страница, потребно је научити велики број оваквих ознака (научити, на пример, да се пасуси увек означавају са `p`, слике са `img`, а линкови са `a`) и то ће нам бити један од главних задатака, али тек у наредним поглављима.

Даље, приметимо да је опису странице присутан један велики блок кода у коме нема текста, а у коме се помињу разна графичка својства, попут фонта, боја, поравнавања и слично (на пример, `font-family: Arial` означава фонт који треба користити, `background-color: #f0f0f0` означава да боја позадине треба да буде светло сива, док `text-align: center` говори да неки текст треба да буде центриран). Ако би тај блок кода био обрисан, страница би имала потпуно исти садржај, али би изгледала доста једноставније.





## Jezici za opis veb-stranica

U opisu veb-stranica koristi se više jezika i tehnologija.

- Jezik [HTML](#) služi za opis sadržaja veb-stranica. Za opis sadržaja koriste se *HTML elementi* (na primer, `<html>`).
- Jezik [CSS](#) služi za opis stila (boja, fontova, rasporeda sadržaja i slično) veb-stranica.
- Jezik [JavaScript](#) se koristi za dodavanje interaktivnosti veb stranicama (na primer, kada se klikne na naslov ove stranice, on postaje crven).

Слика: 2.veb-jezici-nostyle.png.png

Потпис: Приказ странице без стилских описа

На крају, постоји блок кода `document.getElementById('naslov').onclick = function(){this.style.color = 'red'};` који донекле личи на програме какве сте писали у трећем разреду. Он служи да се дода одређена динамика овој страници – када корисник кликне на наслов, тада се боја тог наслова промени на црвену.

Дакле, можемо јасно идентификовати три основна аспекта веб-странице: **садржај**, **изглед** и **понашање** и за опис сваког од тих аспеката користи се посебан језик.

Садржај странице и њена логичка организација (подела садржаја странице на наслове, пасусе, заглавља, подножја, листе, табеле, уметање слика, линкова и слично) описују се језиком који се назива **HTML** (енгл. *HyperText Markup Language*). Стил странице, односно њен изглед, тј. визуелна презентација (опис фонтова, боја, оквира, маргина, поравнавања текста и слично) описује се језиком **CSS** (енгл. *Cascading Style Sheets*). Понашање странице, тј. додавање некакве динамичности и интерактивности страници дефинише се у најчешће у језику **JavaScript**.

Све ове језике прегледачи-веба директно разумеју, тј. да бисте могли да креирате и прегледате веб-страницу потребно је да у едитору текста откуцате њен изворни код и да је отворите из прегледача веба. С обзиром на то да је данас на сваком рачунару постоји и неки едитор текста и веб-прегледач, не морате ништа додатно да инсталирате. HTML и CSS се не могу убројати у програмске језике (јер не подржавају стандардне програмске конструкције попут, наредби, променљивих, петљи, гранања и слично) већ у **језике за обележавање** (енгл. *markup languages*). Са друге стране, JavaScript је прави програмски језик.

**Занимљивост.** Иако деле део имена, JavaScript и чувени програмски језик Java су сасвим различити програмски језици.

Основу странице увек чини опис њеног садржаја, структуре и логичке организације, а у тај опис се може директно уметнути опис изгледа тј. стила странице (између ознака `<style type="text/css">` и `</style>`) и опис понашања странице тј. реакција на одређене догађаје које корисник проузрокује (између ознака `<script type="text/javascript">` и `</script>`). С обзиром на то да се тада у истом документу мешају синтаксе три различита језика потребно је бити веома обазрив и на сваком месту у коду знати тачно који се од тих језика користи. На пример, приметимо да су изворном коду задати и коментари, попут `<!-- Opis zaglavlja veb-stranice -->`, `/* sivkasta boja pozadine */` и `// kada se klikne na naslov, on postaje crven` и да је синтакса за запис коментара у сваком од ова три примера различита – коментари у језику HTML записују се између `<!--` и `-->`, у језику CSS између `/*` и `*/`, док се у језику JavaScript обично записују иза две косе црте `//`.

Осим директним уметањем кода у HTML опис веб-странице, стилске и програмске описе је могуће задати и у засебним датотекама (обично са екстензијама `.css` и `.js`) које се онда укључују у основни опис (датотеку са екстензијом `.html`) – конкретне примере како се ово ради приказаћемо у наставку.

Пошто се веб-странице описују помоћу језика HTML за њих се некада каже и да су **HTML документи**.

Пошто планом и програмом овог предмета није предвиђено да се изучава JavaScript у наставку овог поглавља ћемо упознати неке основне појмове језика HTML и CSS.

## 1.3 HTML

Језик HTML везан је за саме настанке **веба** (енгл. *World Wide Web, WWW*) који је настао касних 1980-тих година у истраживачком центру CERN у Швајцарској. Развио га је енглески информатичар и физичар Тим Бернерс-Ли у оквиру система који је омогућавао истраживачима у CERN-у једноставну размену докумената. Кључне идеје на којима је Бернерс-Ли искористио су да се документи размењују коришћењем **Интернета** и да се документи представљају као **хипертекст**, тј. текст који садржи везе (упутнице, линкове) ка другим документима и који, уколико се чита уз подршку специјализованог софтвера, омогућава изузетно једноставну навигацију кроз велики број докумената и тиме значајно повећава доступност информација. Основу веба представљао је језик **HTML** који је служио за опис садржаја страница и линкова у њима, као и протокол **HTTP** који је коришћен за размену тако описаних веб-страница између клијената и удаљених сервера на којима се веб-странице чувају. На клијентским рачунарима HTML документи тј. веб-странице се најчешће приказују помоћу специјализованог софтвера који се назива **прегледач веба** или **веб-браузер** (енгл. *web browser*). Ипак, треба имати у виду да се HTML документи обрађују аутоматски и од стране другог софтвера (на пример, претраживачи попут Google-а анализирају садржај веб-странице, слабовидим особама садржај страница читају читачи екрана и слично), па се уместо појма веб-прегледач за софтвер који обрађује HTML документе понекад користи мало општији термин **кориснички агент** (енгл. *user agent*).

**Занимљивост.** Први први прегледач веба који је увео могућност приказивања слика је Mosaic развијен током 1990-их, што вебу доноси широку популарност и велики број корисника.

Језик HTML је током година доживљавао многе промене. У неким тренуцима се развијао прилично стихијски (компаније које производе прегледаче веба су проширивале језик многим елементима који нису били стандардизовани), све док његов развој и стандардизацију није преузела непрофитна организација World Wide Web Consortium (W3C) на чијем је челу Бернерс-Ли, а која окупља стручњаке запослене у водећим софтверским компанијама. Тренутно важећа верзија стандарда је HTML 5, који је 2014. заменио верзију 4.01, актуелну током више од десет година.

**Слика:**

**Потпис: Логотип језика HTML5**

Још једном нагласимо да је HTML језик који је намењен опису садржаја и логичке структуре веб-страница, а не опису њихове визуелне презентације, тј. стила и изгледа. Приликом означавања садржаја веб-страница пре свега се треба водити његовом семантиком тј. значењем и његовом улогом у целини документа, а не тиме како тај садржај треба да изгледа када се прикаже у прегледачу (никада не заборавите то да прегледачи нису једини софтвер који чита веб-странице). На пример, добро означена логичка структура помаже претраживачима да одреде шта је централни садржај ваших страница и самим тим таква страница постаје боље рангирана у резултатима претраге, што је данас веома битно (на пример, претраживачи тексту који је експлицитно означен као наслов дају много већи приоритет него тексту који је приказан крупни словима, али није означен као наслов).

## Општа синтакса језика

Општа синтакса језика HTML је заснована на општијем језику SGML (чији је савремени наследник језик XML). Документ почиње декларацијом DOCTYPE која између осталог указује на верзију језика HTML која ће у документу бити коришћена. Верзија HTML 5 захтева веома једноставу декларацију DOCTYPE:

```
<!DOCTYPE html>
```

Иако није неопходно, пожељно је да се DOCTYPE пише свим великим словима, а html свим малим.

Документ је сачињен од елемената (енгл. *element*). Елементи се означавају **ознакама** тј. **таговима** (енгл. *tag*), које на неки начин одговарају заградама и могу бити **отварајуће ознаке** (енгл. *opening tag*) које су облика облика `<ime-elementa>` и **затварајуће ознаке** (енгл. *closing tag*) које су облика `</ime-elementa>`. Све између отварајуће и затварајуће ознаке сматра се **садржајем** елемента (енгл. *content*). На пример, елемент `<p>U opisu veb-stranica koristi se vise jezika i tehnologija.</p>` представља један пасус у документу, почиње отварајућом ознаком `<p>`, завршава се затварајућом ознаком `</p>`,



док му је садржај текст U opisu veb-stranica koristi se više jezika i tehnologija. Називи елемената се наводе у ознакама и обично се пишу свим малим словима. Неки елементи немају садржај тј. њихов садржај је празан. На пример, пасуси увек садрже неки текст док су слике атомички елементи и оне немају свој садржај. Један начин да се такви елементи означе је да се отварајућа и затварајућа ознака ставе једна до друге. Други начин је уведен у циљу скраћења записа и тада се користе посебне **самозатварајуће ознаке** (енгл. *self-closing tags*) које су облика `<ime-elementa />` (на пример, за прелазак у нови ред користи се елемент `br` који нема садржај и често се записује у облику `<br/>` или `<br />`).

Елементи могу да имају **атрибуте** (енгл. *attribute*) који их додатно описују. На пример, елемент `` има три атрибута. Атрибут `src` има вредност `web-logos.png` и представља име датотеке која чува слику коју прегледач на овом месту треба да прикаже. Атрибут `alt` представља алтернативу слици тј. текст који ће прегледач исписати ако из неког разлога не може да прикаже слику, док атрибут `width` представља шируну слике у пикселима. Вредности атрибутима се задају у оквиру отварајуће ознаке елемента у облику `ime-atributa="vrednost-atributa"`. Уместо двоструких наводника, исправно је користити и једноструке апострофе, тј. атрибуте задавати у облику `ime-atributa='vrednost-atributa'`. Иако прегледачи толеришу и када се изоставе наводници, то се не препоручује. И имена атрибута се обично пишу свим малим словима. Око знака једнакости допуштено је користити размаке који се игноришу.

Елементи могу бити и угнежђени тј. садржај једног елемента, поред текста могу сачињавати и други елементи. На пример, садржај елемента `header` у наредном примеру је елемент `img` који представља слику (и заглавље странице осим те слике не садржи ништа друго).

```
<header>
  
</header>
```

Међусобни однос угнежђених елемената може се представити дрветом.

**Слика: 3.drvo-elemenata.png**

**Потпис: Распоред елемената представљен дрветом**

У неким случајевима допуштено је изостављање затварајућих ознака (па чак и отварајућих). На пример, следећи чланак (елемент `article`) садржи два пасуса (елемента `p`), иако ни један од није експлицитно затворен.

```
<article>
  <p>Ово је први пасус
```



```
<p>Ово је други пасус
</article>
```

Пошто пасуси не могу да буду угнежђени (један пасус у себи не може да садржи други), почетак новог пасуса, тј. његова отварајућа ознака `<p>` указује на то да се на том месту претходни пасус завршава. Слично томе, крај чланка (затварајућа ознака `</article>`) указује да се на том месту мора завршити други пасус (јер он не може да изађе из оквира елемента у којем је садржан). Иако изостављање неких ознака мало скрађује код и потребу за куцањем, оно се не сматра добром праксом и ми то убудуће нећемо радити.

**Занимљивост.** Прегледачи веба су направљени тако да буду веома флексибилни и да се труде да по сваку цену прикажу и код који није исправан, а све у циљу да корисник види неки садржај, чак и у случајевима када је аутор веб-странице направио неке грешке. Иако их то ставља у донекле удобнију позицију од програмера, аутори веб-страница не би требали да се ослањају на то јер је једини начин да буду сигурни да ће се њихове веб-странице исправно приказати на свим прегледачима тај да стриктно поштују правила језика за креирање веб-страница.

## Основна структура HTML странице

Кренимо сада да уводимо основне елементе и атрибуте језика HTML један по један, кренувши од костура сваке веб-странице. Пример који ће се протезати кроз цео уџбеник биће креирање школског сајта гимназије која носи име нашег чувеног просветитеља Доситеја Обрадовића.

```
<!DOCTYPE html>
<html>

  <!-- Opis zaglavlja veb-stranice -->
  <head>
    <meta charset="UTF-8" />
    <title>Gimnazija Dositej Obradović</title>
  </head>

  <!-- Opis tela veb-stranice -->
  <body>
    Dobrodošli na sajt Gimnazije „Dositej Obradović“.
  </body>

</html>
```

**Пробај ово на рачунару.** Укључи свој омиљени едитор текста. На пример, ако користиш оперативни систем Windows, можеш да користиш едитор Notepad, или још боље, да преузмеш са интернета и инсталираш мало напреднији едитор који се назива Notepad++ и слободан је за коришћење, док, ако користиш оперативни систем Ubuntu Linux можеш да користиш едитор gedit. Прекуцај пажљиво зачетак веб-странице који смо претходно приказали. Приликом снимања странице, дај јој име `index.html` и обезбеди да едитор на то име аутоматски не дода екстензију `.txt` (на пример, у едитору Notepad име `"index.html"` откуцај под наводницима или у категорији `Save as type` уместо `Text Documents`

(\* .txt) одабери All Files). Ако користиш оперативни систем Windows, да би био сигуран да је датотека снимљена са исправним екстензијама, искључи подразумевану опцију скривања екстензија (подсети се градива првог разреда гимназије, када се учило како се то ради).

**Слика:**

**Потпис: Снимање датотеке у едитору Notepad**

Такође, потребно је да одабереш кодирање карактера (кодну страну) која ће се користити приликом снимања документа у датотеку. Пошто претпостављамо да ће документ садржати латиничке или ћириличке карактере српског језика, одабери кодирање UTF-8. Одабир кодирања се у неким едиторима (на пример у едитору Notepad или gedit) врши приликом снимања, обично у категорији Encoding, док се у неким едиторима може урадити и током уноса садржаја документа, из посебних менија (на пример, у едитору Notepad++ кодирање се подешава у менију Encoding).

**Слика:**

**Потпис: Одабир кодирања карактера**

Отвори креирану веб-страницу из веб-прегледача (обично је довољно само да двоструко кликнеш на иконицу којом је представљена снимљена страница). Ако је све урађено како треба, страница би требало исправно да буде приказана.

**Слика:**

**Потпис: Приказ странице из веб-прегледача**

Настави да уносиш садржај у ову страницу, пратећи уџбеник. Након сваке измене HTML датотеке, потребно је поново учитати је у прегледач веба (уместо затварања прегледача и његовог поновног отварања, много је једноставније само покренути освежавање веб-странице, обично пречицом F5).

## **Елементи html, body и head**

Након декларације типа документа (`<!DOCTYPE html>`), цела веб страница је представљена једним елементом `html`. Садржај елемента `html` увек треба да чине два елемента: `head` и `body`. Елемент `head` садржи опис заглавља веб-странице (значајних информација о веб-страници који се не виде директно на веб-страници тј. у прозору веб-прегледача), док елемент `body` садржи опис тела веб-странице (садржаја који се приказује директно на веб-страници). Ознаке `<head>` и `</head>` је допуштено изоставити и тада се заглављем странице сматра све оно што је наведено након ознаке `<html>`, а пре ознаке `<body>` (као што смо већ рекли, изостављање ознака се не препоручује и стога ћемо ознаке `<head>` и `</head>` увек наводити).

## **Заглавље веб-странице и мета-подаци**

Поред обавезног елемента `title` који служи за опис наслова странице, заглавље странице (елемент `head`) може да садржи и друге елементе који служе за опис мета-података о страници (под мета подацима сматрају се подаци који нису део садржаја саме странице већ говоре нешто о том садржају). Поменућемо само елементе `meta`, `link`, `style` и `script`.

## Елемент `title`

Заглавље странице (елемент `head`) обавезно мора да садржи елемент `title` у којем се описује наслов веб-странице (овако описан начин се приказује у језичку картице прегледача у којој се приказује та веб страница, али се приказује и као главни наслов у резултатима претраживача веба). Аутори би требало да користе елемент `title` тако да читаоци на основу њега могли да идентификују потенцијални садржај документа. Пошто корисници овај наслов виде и ван контекста странице (на пример, када се страница прикаже у оквиру резултат претраге коришћењем неког претраживача веба), препоручује се да наслов документа буде широк и довољно информативан. Нпр. уместо кратког и неинформативног наслова `Увод`, боље је користити информативнији наслов `Увод у језик HTML`.

**Слика:**

**Потпис:** Наслов странице у језичку картице и у резултатима претраживања

## Element `meta`

Елемент `meta` обично има празан садржај (обично се наводи у облику `<meta ... />`) и служи да се путем његових атрибута задају неке основне мета информације о страници. Једна од најчешћих употреба овог елемента је да се помоћу атрибута `charset` прегледачу саопшти који начин кодирања карактера је коришћен приликом снимања садржаја странице у датотеку (већина едитора текста допушта кориснику да одабере кодирање карактера). У случају да се начин кодирања карактера у едитору текста и начин кодирања карактера у прегледачу веба не поклапају, прегледач ће неке карактере у веб-страници погрешно приказати. Када веб-страница садржи текст на српском језику (без обзира да ли се користи ћирилица или латиница), потребно је користити и карактере који нису подржани таблицом ASCII. Стога је пожељно користити таблицу Unicode, тј. кодирање карактера UTF-8. У том случају, приликом његовог снимања у едитору текста у категорији `Encoding` (или, у неким едиторима, у оквиру подешавања у менију `Encoding`) потребно је одабрати опцију UTF-8, као што смо већ нагласили приликом описа креирања наше прве веб-странице. Уз то, у заглавље веб-странице (у елемент `head`) потребно је додати елемент `<meta charset="UTF-8" />`.

**Ко жели да зна више?** Поред кодирања UTF-8, за кодирање латиничког текста на српском језику користе се кодне схеме `windows-1250` и `iso-8859-2`, а за кодирање ћириличког текста користе се кодне схеме `windows-1251` и `iso-8859-5`. Иако текст записан овим схемама заузима мало мање простора него у случају коришћења UTF-8, у њима није могуће у потпуности исправно комбиновати текст на различитим писама, што UTF-8 допушта.

Осим за навођење кодирања карактера, елемент `meta` се често користи за саопштавање информација које могу претраживачима веба бити корисне приликом разматрања странице. На пример, могуће је записати име аутора странице (`author`),

кључне речи (`keywords`), опис странице (`description`) и слично. Да би се то постигло, вредност атрибута `name` поставља се на `keywords`, `author`, `description` и слично, док се информације уносе у облику вредности атрибута `value`. На пример, у заглављу једног школског сајта, могу се налазити следеће мета информације:

```
<head>
  ...
  <meta name="author" value="Filip Marić" />
  <meta name="keywords" value="gimnazija, škola, obrazovanje" />
  <meta name="description"
        value="sajt gimnazije Dositej Obradović" />
</head>
```

**Ко жели да зна више?** Оптимизација претраживачима веба (енгл. *Search Engine Optimization, SEO*) је скуп техника којима аутори веб-страница побољшавају видљивост својих страница. Сматра се да просечни корисници претраживача обично посете само странице које су приказане на првој, евентуално другој страни резултата претраживања и стога је јако важно да се ваша страница за одређене карактеристичне упите високо рангира. Компаније које се баве претрагом (на пример, Google) откривају да више од 200 различитих фактора утиче на рангирање страница, али је тачан алгоритам рангирања тајна. На ранг странице утиче пре свега њен садржај, затим мета-подаци, али и структура долазних линкова (странице ка којима воде линкови са великих и важних сајтова бивају добро рангиране).

## **Елементи `script`, `style` и `link`**

Елементе `script` и `style` смо већ укратко поменули у уводном поглављу.

Садржај елемента `script` представља програмски кôд, најчешће на језику JavaScript који служи да се страници дода нека интерактивност (обично се њиме програмирају реакције на догађаје корисника попут клика на дугме или неке акције мишем, или нека аутоматска обрада података које корисник уноси у страницу кроз формуларе које страница може да садржи).

Садржај елемента `style` представља стилски опис на језику CSS који описује како одређени елементи на страници треба да изгледају (на пример, како су одређене боје, фонтови, поравнавање текста, маргине, оквири и слично) и како треба да буду распоређени на страници.

Елемент `link` се користи да повеже страницу у којој је наведен са другим ресурсима. Његова најчешћа употреба је да се у страницу увезе стилски опис задат у засебној датотеци. На пример, ако датотека `style.css` садржи опис стила, она се може укључити у веб-страницу елементом, о чему ће бити више речи када будемо говорили о стилским описима и језику CSS:

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

Још неке од могућих употреба су да се зада верзија веб странице у неком другом формату (на пример, PDF) или на неком другом језику (на пример, на енглеском).

```
<link rel="alternate" href="page.pdf" type="application/pdf"
      title="PDF verzija" />
<link rel="alternate" href="page-en.html" type="text/html"
      hreflang="en" title="Verzija na engleskom jeziku" />
```

## Целине у телу HTML документа

Садржај тела HTML документа се обично може поделити на неке мање логичке целине. На пример, на већини веб-страница се може идентификовати део странице који садржи линкове за кретање унутар веб-сајта, може се идентификовати неко заглавље које често садржи логотип веб-сајта, главни наслов, може се идентификовати неки део, обично постављен са стране који садржи споредне информације (на пример, неке рекламе) и слично. Почевши од верзије 5, језик HTML подржава неколико различитих елемената који се користе за опис поделе веб-страница на овакве целине. Нагласимо да су ти елементи логичке (семантичке) природе и да немају никаквог директног утицаја на начин како ће њихов садржај бити приказан, нити на позицију на екрану где ће се тај садржај постављати. Означивање садржаја увек треба да буде по томе шта тај садржај представља (његове семантике), а не по томе како тај садржај треба да изгледа. Распоређивање и стилизовање садржаја ради се накнадно, помоћу језика CSS.

### *Елементи main, nav, aside, header, footer*

Елемент `main` се користи да означи централни садржај тела веб-странице. Очекује се да је садржај овог елемента у директној вези са централном темом веб-странице или централном функционалношћу веб-апликације. Садржај елемента `main` треба да буде јединствен за ту специфичну страницу и елемент `main` не би требало да садржи елементе који се понављају на свим страницама истог веб-сајта (заглавља странице, навигационе линкове и слично). Свака страница може да садржи највише један елемент `main`.

Елемент `header` представља заглавље (енгл. *header*) веб-странице или заглавље одређене мање секције у оквиру странице (о таквим секцијама ће бити речи у наставку). Заглавље странице често садржи уводних елемената попут наслова веб-сајта, логотипа, групе навигационих линкова и слично.

Елемент `footer` представља подножје (енгл. *footer*) веб-странице или подножје одређене мање секције у оквиру странице. Подножје обично садржи податке о томе ко је написао страницу, информације о заштити ауторских права (енгл. *copyright*), линкове ка сродним документима и слично.

Елемент `nav` представља део веб-странице који садржи груписане линкове ка другим страницама (обично у оквиру истог веб-сајта) или линкове ка неком садржају унутар исте те веб-странице. Елемент `nav` је обично део заглавља странице, али то није обавезно.

Имајући ово у виду, можемо да започнемо обележавања тела једног школског сајта. Тело ће се састојати од четири велика дела: заглавља које ће садржати наслов и логотип гимназије, као и навигационе линкове, централног дела у коме ће се налазити неколико

важних информација за посетиоце сајта и ученике школе, споредног дела који ће садржати вести и подножја које ће садржати име аутора сајта и датум последње измене. Садржај ових елемената унећемо касније.

```
<body>
  <header>
    <!-- naslov i logotip stranice -->
    <nav>
      <!-- linkovi za navigaciju unutar sajta -->
    </nav>
  </header>

  <main>
    <!-- centralni deo stranice -->
  </main>

  <aside>
    <!-- vesti i aktuelnosti -->
  </aside>

  <footer>
    Autor: Filip Marić, poslednja izmena: 1. 1. 2016.
  </footer>
</body>
```

## **Елементи *h1*, *h2*, *h3*, *h4*, *h5* и *h6***

Сви структурирани текстуални документи, па и веб-странице садрже хијерархију наслова и поднаслова. Језик HTML пружа шест нивоа наслова (сматра се да је то и више него довољно) и они се означавају елементима *h1*, *h2*, *h3*, *h4*, *h5* и *h6*. Ови наслови су део тела странице и приказују се у оквиру тела, те их не треба мешати са насловом задатим путем елемента *title*. Елемент *h1* означава наслов највишег ранга (обично главни наслов целе веб-странице), следи елемент *h2* (он се обично користи за поднаслове, тј. наслове појединачних секција странице), док елемент *h6* означава наслов најнижег ранга. Наслове би требало увек наводити на почетку дела странице (секције) на коју се односе.

Заглавље нашег школског сајта можемо сада проширити главним насловом:

```
<header>
  <h1>Gimnazija „Dositej Obradović“, Kragujevac</h1>
  ...
</header>
```

## **Елементи *article*, *section***

Ако веб-страница има доста материјала, погодно је тај материјал поделити у засебне секције (енгл. *sections*). Секције би могле да одговарају поглављима неке књиге, а нека

веб-страница би могла да има своју уводну секцију, секцију са вестима и секцију са контакт-подацима. За означавање секција користи се елемент `section`. Свака секција представља неко тематско груписање садржаја и требало би да почне неким насловом који илуструје о чему се у тој секцији говори. Пошто секције могу представљати и крупније целине, оне у себи могу садржати и заглавља, подножја, мање подсекције и слично.

Потпуно, заокружене целине, независне од осталог садржаја називају се чланци (енгл. *articles*). Чланак би могао представљати чланак у неким електронским новинама, али и сваки појединачни пост (објава) на неком форуму или блогу као и сваки појединачни коментар читалаца. Чланци се обележавају се елементом `article`. Чланак може да садржи мање чланке (на пример, објава на блогу је чланак, који у себи може да садржи коментаре читалаца који су такође чланци). Дужи чланци могу бити подељени на секције (и садржати елементе `section`), а секције у себи могу садржати чланке. Чланци могу имати наслове (али не морају).

Секције и чланци имају своју јасну семантику и не би их требало употребљавати вештачки, тј. само да би се груписао неки садржај да би се стилизовао (за то се може користити генерички елемент `div` о коме ће касније бити речи).

Кренимо да уносимо главни садржај на наш школски сајт. Вести које се налазе са стране ћемо поделити у две секције: најновије вести и остале вести. Свака од тих секција садржаће чланке који представљају појединачне вести (њихов текст ћемо унети касније).

```
<aside>
  <section>
    <h2>Najnovije vesti</h2>
    <article>
      <h3>Uspeh naših matematičara</h3>
      <!-- tekst o takmičenju iz matematike -->
    </article>
  </section>
  <section>
    <h2>Ostale vesti</h2>
    <article>
      <h3>Novi raspored časova</h3>
      <!-- tekst o novom rasporedu časova -->
    </article>
    <article>
      <h3>Sekcija iz istorije</h3>
      <!-- tekst o sekciji iz istorije -->
    </article>
  </section>
</aside>
```

Слично томе, централни део сајта ће садржати више појединачних насловљених чланака које ћемо касније попунити садржајем.



```
<main>
  <article>
    <h2>Škola vrednih đaka</h2>
    <!-- ... -->
  </article>

  <article>
    <h2>Broj upisanih učenika po razredima</h2>
    <!-- ... -->
  </article>

  <article>
    <h2>Upis u novu školsku godinu</h2>
    <!-- ... -->
  </article>

  <article>
    <h2>Kontakt</h2>
    <!-- ... -->
  </article>
</main>
```

## **Елемент `div`**

У ранијим верзијама језика HTML елемент `div` је био један од убедљиво најкоришћенијих елемената. То је генерички елемент, без унапред придружене семантике, чији је једини задатак да групише неки садржај (блокове текста, али и друге елементе). Садржај груписан елементом `div` могуће је третирати тј. обрађивати као целину. На пример, могуће је стилизовати га у језику CSS (позиционирати на неко место унутар веб-странице, уоквирити и слично) или му додељивати неку интерактивност помоћу језика JavaScript (на пример, када је део странице груписан у елемент `div` могуће је сакривати га или приказивати на захтев корисника). Елемент `div` се обично користи у комбинацији са неким од глобалних атрибута о којима ће више речи бити касније (најчешће су то `id` путем којег дајемо име елементу и `class` путем којег придружујемо елементу одређену класу којој припада). У верзији 5 у језик HTML уведени су семантички елементи `main`, `header`, `footer`, `aside`, `nav`, `section` и `article`, чиме је значајно смањена потреба за коришћењем генеричког елемента `div`. Основна разлика између елемента `div` и, на пример, елемента `header` је та што елемент `header` представља правоугаони блок садржаја за који знамо да представља заглавље стране, док елемент `div` представља неки правоугаони блок садржаја о коме ништа посебно не знамо. Семантички елементи за груписање садржаја су управо уведени на основу најчешће употребе елемента `div` (на пример, раније се заглавље обично обележавало помоћу елемента `<div class="header">...</div>`, при чему се атрибут `class` користио да мало ближе опише намену елемента `div`, док се данас за то користи много јаснија и једноставнија варијанта `<header>...</header>`). Ипак, елемент `div` је

задржан, али га треба користити само када се процени да ниједан од семантичких елемената није примерен да опише садржај који се групише.

На пример, на дно централне секције нашег сајта можемо да убацимо три сличице, које ћемо груписати и уоквирити. С обзиром на то да није у питању прави чланак (јер не садржи никакав текст) и да је једина сврха груписања тих слика да се оне касније заједно могу позиционирати и уоквирити, употребићемо елемент `div` коме ћемо придружити јединствени идентификатор `gallery` (сврха идентификатора елемената је најчешће то да се приликом стилизовања или додавања интерактивности елементима, елементу може приступити навођењем његовог идентификатора).

```
<main>
  ...
  <div id="gallery">
    <!-- три sličice na dnu centralnog dela strane -->
  </div>
</main>
```

## Пасуси, листе, адресе, цитати, ...

### Елемент `p`

Најмања јединица груписаног текста је пасус (енгл. *paragraph*) и он се обележава елементом `p`. Пасуси су мање структуралне јединице и они могу да садрже означен текст, линкове, слике и слично, али не и крупније елементе попут секција, чланака, али других пасуса и листа и табела о којима ће бити речи касније.

На пример, чланак о успеху ученика на такмичењу из математике природно може да садржи два пасуса која следе иза наслова.

```
<article>
  <h3>Uspeh naših matematičara</h3>
  <p>Na okružnom takmičenju iz matematike održanom u subotu,
  16. januara, naši učenici su ostvarili odličan uspeh. Dvanaest
  učenika se plasiralo na republičko takmičenje, koje će biti
  održano u Novom Sadu u martu mesecu.</p>
  <p>Čestitamo svim učesnicima takmičenja i profesorima koji
  su ih spremali.</p>
</article>
```

### Елементи `ul`, `ol`, `li`, `dl`, `dd`, `dt`

Често се у тексту јављају набрајања, тј. листе ставки. Језик HTML подржава два основна типа листа:

- ненумерисане листе (енгл. *unordered lists*), представљене елементом `ul`,
- нумерисане листе (енгл. *ordered lists*), представљене елементом `ol`,

И нумерисане и ненумерисане листе садрже низ ставки означених елементом `li`.

Чланак о упису у нову школску годину може да садржи листу докумената које треба

понети на упис.

```
<article>
  <h2>Upis u novu školsku godinu</h2>
  <p>Na upis je potrebno doneti sledeće dokumente:</p>
  <ol>
    <li>svedočanstvo iz prethodnog razreda,</li>
    <li>izvod iz matične knjige rođenih,</li>
    <li>đačku knjižicu.</li>
  </ol>
</article>
```

Листа се приказује на следећи начин:

**Слика:**

**Потпис:** Нумерисана листа

Део стране посвећен навигацији (елемент `nav`) садржи појединачне линкове. Иако се они обично стилизују другачије, линкове је семантички природно задати у облику нумерисане листе (и то се у пракси најчешће тако и ради). Пошто још нисмо приказали како се описују линкови, за сада ћемо навести само обичан текст линкова, а линкове ћемо убацити касније.

```
<nav>
  <ul>
    <li>O školi</li>
    <li>Zaposleni i učenici</li>
    <li>Raspored časova</li>
    <li>Kontakt</li>
  </ul>
</nav>
```

**Слика:**

**Потпис:** Ненумерисана листа

Поред ненумерисаних и нумерисаних листа, језик HTML подржава и дефиниционе листе (енгл. *definition lists*), представљене елементом `dl`. Дефиниционе листе садрже низ термина који се дефинишу. Оно што се дефинише означава се елементом `dt`, а оно чиме се дефинише означава се елементом `dd`. Оне се ређе користе и стога их нећемо детаљно описивати.

### **Елементи `br` и `pre`**

Распоред текста у HTML датотеци не утиче на то како се текст приказује у веб-прегледачу. Вишеструке белине, па и преласци у нови ред се потпуно игноришу. Да би се прешло у нови ред, није довољно у едитору откуцати текст у новом реду, већ је

потребно употребити елемент `br`. Елемент `br` нема садржај и обично се задаје у облику `<br />`. Елемент `br` не треба употребљавати да би се тематски раздвојио садржај унутар пасуса (боље је том случају текст организовати у више пасуса).

У неким ретким случајевима желимо да веб-прегледач поштује распоред текста унет у HTML документ (на пример, желимо да веб-страници прикажемо неки програмски кôд, и желимо да он буде назубљен у складу са правилима назубљивања програмског језика). Такав текст називамо преформатирани текст и обележавамо га елементом `pre`. На пример,

```
<pre>
program Hello;
begin
  WriteLn('Zdravo svete')
end.
</pre>
```

**Слика:**

**Потпис:** Преформатирани текст

### ***Елемент address***

Веб-страница некада садржи адресе. Оне се обележавају елементом `address`. На пример, у главном делу наше веб-странице гимназије поставићемо чланак са контакт информацијама гимназије (укључујући и поштанску адресу, телефон и адресу електронске поште). Појединачне линије адресе раздвојићемо преласцима у нови ред (елементима `br`).

```
<article>
  <h2>Kontakt</h2>
  <address>
    Gimnazija „Dositej Obradović“, <br />
    Bulevar oslobođenja 38, <br />
    34000 Kragujevac <br />
    Telefon: 034/123-456, e-mail: sekretarijat@gimnazija.rs
  </address>
</article>
```

**Слика:**

**Потпис:** Подразумевани приказ адресе

### ***Елементи blockquote и cite***

Понекад веб-странице садрже неке цитате. За опис цитата користи се елемент `blockquote`, док се име аутора цитата или референца (линк) ка извору цитата наводе у

оквиру елемената `cite`. На пример, у уводни чланак можемо поставити следећи цитат Доситеја Обрадовића:

```
<article>
  <h2>Škola vrednih đaka</h2>
  <blockquote>
    Nema sramotnijeg zanata od dangube, besposlice i lenjosti.
    <cite>Dositej Obradović</cite>
  </blockquote>
  <p>Dobrodošli na sajt naše gimnazije. Mi se ponosimo svojim
  vrednim đacima.</p>
</article>
```

**Слика:**

**Потпис:** Подразумевани приказ цитата

## Опис текста

Веб-странице обично садрже доста текста (на пример, садржај пасуса, тј. елемената `p` обично чини искључиво текст, уз евентуално неке слике и линкове). Делови тог текста могу бити на неки начин истакнути и посебно обележени.

## Елементи `em`, `strong`, `small`, `i`, `b`, `u`

Текст означен елементом `i` се обично приказује у курсиву (енгл. *italic*), текст означен елементом `b` се обично исписује подебљано (енгл. *bold*), док се текст означен елементом `u` приказује подвучено (енгл. *underline*). Ипак, стандард и овим елементима придружује богатију семантику од једноставног подешавања начина приказа текста па се препоручује да се елемент `i` користи за означавање, на пример, техничких термина, фраза преузетих из другог језика, транслитерација и слично, да се елемент `b` користи, на пример, за истицање кључних речи у апстракту неког документа, имена производа и компанија у неком тексту или почетни пасус неког документа.

**Слика:**

**Потпис:** Приказ елемената `b`, `i` и `u`.

Елемент `em` се користи да се означи да је неки део текста истакнут. Елемент `strong` се користи да се нагласи да нагласи важност, озбиљност, хитност неког дела текста. Елемент `small` се користи да се неки део текста потисне, тј. да се нагласи да он представља неке напомене које су споредне у односу на централни део текста. Иако се елемент `em` се обично приказује курсивним словима, исто као и елемент `i`, док се елемент `strong` обично приказује подебљаним словима, исто као и елемент `b`, њихова семантика је различита (на пример, `strong` служи да истакне важност неког дела текста, док се елементи означавају елементом `b` углавном из разлога техничке природе, без намере да се тај део текста прогласи важним).

**Слика:**

**Потпис:** Подразумевани приказ елемената `em`, `strong` и `small`.

### **Елементи `sub` и `sup`**

Језик HTML не пружа могућност описа математичких формула (за то се могу користити посебне софтверске алатке попут JavaScript библиотеке MathJax или описа MathML). Ипак, постоје елементи `sub` и `sup` којима се могу означити индекси и експоненти. На пример,

```
Hemijska oznaka vode je H<sub>2</sub>O. Polinom x<sup>2</sup> - y<sup>2</sup> naziva se razlika kvadrata.
```

**Слика:**

**Потпис:** Приказ елемената `sub` и `sup`.

### **Елемент `span`**

У неким случајевима потребно је груписати одређени део текста у целину и означити га на неки начин. Елементи попут `em`, `strong`, `small` и слично служе да означе делове текста, али они имају јасну, стандардом прописану семантику (на пример, `em` означава наглашене делове текста). У неким случајевима је потребно означеном делу текста придружити неку посебну семантику, која није дефинисана HTML стандардном и за то се може употребити елемент `span`. Он је генерички елемент и сам по себи не значи ништа, али може бити користан када се користи у комбинацији са глобалним атрибутима (на пример, `id`, `class`, `lang`). На пример, ако желимо да на веб-страници рачунарске термине означимо на неки посебан начин, можемо сваки такав термин означити елементом `span` и сваком таквом елементу придружити класу `termin`. На пример,

```
Основни делови рачунарског система су <span class="termin">процесор</span>, <span class="termin">меморија</span> и <span class="termin">улазно-излазни уређаји</span>.
```

Иако би се без додатних подешавања рачунарски термини у веб-прегледачу приказивали потпуно исто као и остали текст, захваљујући њиховом експлицитном обележавању коришћењем могуће би, на пример, било да се коришћењем посебног програма издвоје све рачунарске термине из веб-странице. Такође, коришћењем језика CSS једноставно се може подешавати приказ свих термина (на пример, може се подесити да се они приказују у курсиву или да се приказују другом бојом).

Слично, коришћењем `span` елемената можемо означити синтактичке елементе у фрагменту програмског кода, да бисмо касније, уз помоћ CSS-а, овај код могли обојити.

```
<pre>  
<span class="keyword">program</span> <span class="id">Hello</span>;  
<span class="keyword">begin</span>  
  <span class="id">WriteLn</span>(<span class="str">'Zdravo svete'</span>)  
<span class="keyword">end</span>.  
</pre>
```

Све кључне речи означили смо класом `keyword`, све идентификаторе класом `id`, а ниске класом `str`. Приметимо да је и даље коришћен елемент `pre` који служи да се у приказу употреби исти формат текста као и у изворном коду документа.

## Линкови

**Везе** тј. **линкови** (енгл. *link*) представљају једну од кључних одлика веба – мреже повезаних докумената. Линк представља везу између два ресурса на вебу, од чега је један тренутни документ, тј. документ који је тренутно приказан у прегледачу и који у свом HTML опису садржи линк. Најчешће коришћена врста линкова су **хиперлинкови**. Хиперлинк је елемент странице који корисник може да активира (најчешће кликом мишем, а ређе помоћу тастатуре) и да тиме проузрокује да прегледач учита и прикаже нека другу веб-страницу уместо тренутне и да се приказ позиционира на одређени део неке веб странице (при чему, то може бити или део неке ново учитане странице или део исте оне која садржи линк). Наме, пошто су странице обично дугачке и није их могуће видети одједном, већ је потребно скроловати и прегледати део по део странице, позиционирање на одређени део странице знатно олакшава прегледање садржаја.

**Занимљивост.** За корисника који чита садржај пратећи хиперлинкове каже се да прегледа веб (енгл. *web browsing*) или да сурфује вебom (енгл. *web surfing*).

Слика:

Потпис: линк на веб-страницу

## Елемент `a`

Хиперлинкови се креирају коришћењем елемента `a`. Садржај тог елемента је обично или неки кратак текст или нека слика и приказ тог садржаја представља активну површину на коју корисник може да кликне да би активирао линк. Атрибут `href` садржи адресу (URL) која описује ресурс који се приказује активирањем линка. Веома једноставан пример линка може бити:

```
<a href="http://www.matf.bg.ac.rs">Математички факултет, Београд</a>
```

Посетилац странице види текст Математички факултет, Београд и када кликне на њега, у прегледачу се отвара главна страна Математичког факултета која се налази на веб-



адреси <http://www.matf.bg.ac.rs>.

На нашем сајту гимназије, у вести о такмичењу из иможемо направити линк ка сајту Друштва математичара Србије (њихов сајт се налази на адреси <http://www.dms.rs>).

```
<p>Na okružnom takmičenju iz matematike u organizaciji  
<a href="http://www.dms.rs">Društva matematičara Srbije</a>  
održanom u subotu, 16. januara, naši učenici su ostvarili  
odličan uspeh.</p>
```

Ако желимо да линк буде постављен на слику, између `<a>` и `</a>` треба просто ставити елемент `img` којим се описује слика, о чему ће бити речи ускоро.

### **Апсолутно и релативно адресирање**

Адресе наведене у оквиру атрибута `href` могу бити апсолутне, какве су биле у претходним примерима, али и релативне. Апсолутне адресе су оне које представљају исправно формиран URL (оне обично почињу ознаком протокола попут `http://...` или `https://...`). Све адресе које нису тог облика се сматрају релативним и на њих се примењује поступак разрешавања адреса. Приликом разрешавања користи се базна адреса која се може навести у оквиру елемента `base` у заглављу сајта (што је ређи случај) или се одређује аутоматски на основу адресе на којој се налази тренутни документ (онај који садржи хиперлинк). На пример, ако документ отворен са адресе <http://www.gimnazija.edu.rs/site/index.html> садржи линк

```
<a href="maturanti.html">Maturanti</a>
```

и ако у њему није присутна елемент `base`, тада се базном адресом сматра <http://www.gimnazija.edu.rs/site/>, и наведени линк ће упућивати на страницу <http://www.gimnazija.edu.rs/site/maturanti.html>.

Добра препорука је да се за линкове који воде ка спољашњим документима, тј. ка веб-страницама ван вашег сајта користе њихове апсолутне адресе, а да се за линкове ка страницама унутар вашег сајта користе искључиво релативне адресе. Тиме се постиже да цео ваш сајт можете лако да пребацујете са сервера на сервер, тј. да мењате адресе на којима се ваш сајт налази, а да сви линкови и даље раде исправно, без потребе за икаквим додатним подешавањима. Изузетно је битно нагласити да линкови никада не би требало да садрже апсолутне адресе вашег локалног система датотека, што се може десити ако само копирате адресе које се приказују у прегледачу (на пример, адреса `file://C:\Documents\site\index.html` по правилу не би требало да буде употребљена), јер ако користите такве линкове можете бити сигурни да ће они престати да раде чим пребаците свој веб-сајт на неки други рачунар (на пример, на сервер), па чак и ако му измените локацију у локалном систему датотека.

**Пробај ово на рачунару.** Направи две једноставне веб-странице и повежи их хиперлинковима (линк

са прве треба да отвара другу, а линк са друге, треба да отвара прву страницу). Употреби релативно адресирање. У прву страницу убади хиперлинк ка свом омиљеном веб-сајту. Употреби апсолутно адресирање.

## Адресирање фрагмената

Да би се помоћу линка адресирао део неке странице потребно је користити идентификаторе фрагмената странице. На пример, адреса

```
http://www.gimnazija.edu.rs/index.html#vesti
```

указује на то да треба да се прикаже страница `index.html` која се налази на сајту `http://www.gimnazija.edu.rs/`, и да се у тој страници прикаже елемент који има идентификатор `vesti` (то може да буде неки елемент `div`, `section`, `article` и слично). Идентификатори фрагмената се могу користити и у комбинацији са релативним адресирањем. Могуће је навести и само идентификатор фрагмента и тада се подразумева да се адресира елемент који се налази у страници у којој је линк. На пример, веб-страница гимназије може садржати чланак који садржи контакт податке гимназије и коме је идентификатор (атрибут `id`) постављен на вредност `kontakt`.

```
<article id="kontakt">
  <h2>Kontakt</h2>
  ...
</article>
```

Линк из саме стране ка овом чланку треба да има атрибут `href` чија је вредност само `#kontakt`:

```
<a href="#kontakt">Kontakt</a>
```

Тај линк можемо поставити, на пример, у навигациони део нашег сајта заједно са линковима ка још неколико страница нашег сајта (како добра пракса налаже, за линкове у оквиру истог веб-сајта користимо релативне адресе).

```
<nav>
  <ul>
    <li><a href="skola.html">O školi</a></li>
    <li><a href="zaposleni.html">Zaposleni i učenici</a></li>
    <li><a href="raspored.html">Raspored časova</a></li>
    <li><a href="#kontakt">Kontakt</a></li>
  </ul>
</nav>
```

**Ко жели да зна више?** Некада се уместо идентификатора за означавање долазног сидра користио елемент `a` са атрибутом `name` (`<a name="...">...</a>`), међутим, по стандарду језика HTML 5 то више није исправно (елемент `a` не може да има атрибут `name`).

## Табеле

И у првом и у другом разреду, када сте проучавали програме за припрему текста и програме за табеларна израчунавања, а и ове године, када сте проучавали базе података, научили сте да се подаци најпрегледније представљају у форми табеле. Језик HTML пружа веома богате могућности за опис табела, од чега ћемо ми описати само неке најосновније.

**Ко жели да зна више?** У ранијим верзијама језика HTML табеле су биле основни механизам распоређивања садржаја на страници (такве табеле су приказиване без оквира и посетиоци сајта обично нису били свесни да на страници уопште и постоји табела). Данас се за распоређивање садржаја по страници користи CSS и табеле су изгубиле ову функцију.

### *Елементи `table`, `tr`, `th`, `td`*

У језику HTML табела се описује коришћењем елемента `table`. Табеле обично садрже низ врста, при чему је свака врста представљена елементом `tr` (назив потиче од енглеског *table row*). Врсте садрже ћелије, при чему ћелије могу бити обичне ћелије са подацима представљене елементом `td` (назив потиче од енглеског *table data*) или насловне ћелије представљене елементом `th` (назив потиче од енглеског *table heading*). Једноставан пример табеле би могла бити табела која представља број дечака и девојчица у неком одељењу:

```
<table>
  <tr><th>Дечака</th><th>Девојчица</th></tr>
  <tr><td>18</td>    <td>19</td>
</table>
```

Подразумевано се ћелије заглавља обично приказују са подебљаним словима и центрираним садржајем, а табела се приказује без оквира. Оквир се може укључити на више начина. Један је да се употреби атрибут `border` чија је вредност дебљина оквира. На пример,

```
<table border="1">
  ...
</table>
```

**Слика:**

**Потпис: Једноставна табела**

Оквир око табеле се касније може стилизовати коришћењем језика CSS. Раније верзије језика HTML су за табеле, врсте табеле и ћелије табеле предвиђале неке атрибуте који су служили за подешавање њихових параметара (на пример, атрибуте `width`, `height`, `cellspacing`, `cellpadding`). Међутим, већина ових параметара се може

подесити и помоћу језика CSS што је и препоручени начин, тако да ћемо се на финије подешавање табела вратити у склопу представљања језика CSS. Истакнимо само атрибуте `rowspan` и `colspan` елемента `td` и `th` који омогућавају спајање ћелија табеле тј. изградњу ћелија које се простиру кроз више суседних врста или кроз више суседних колона табеле. На пример, проширимо претходну табелу ћелијом заглавља која се простира кроз две колоне.

```
<table border="1">
  <tr><th colspan="2">Број ученика</th></tr>
  <tr><th>Дечака</th><th>Девојчица</th></tr>
  <tr><td>18</td>    <td>19</td>
</table>
```

**Слика:**

**Потпис: Ћелија која се простира кроз две колоне**

На веома сличан начин можемо креирати и ћелије која се простиру кроз више врста (једино што уместо атрибута `colspan` наводимо атрибут `rowspan`). Приликом описа наредних врста, ћелије које се протежу кроз њих, а раније су већ описане се не описују поново. Неке ћелије могу да се протежу истовремено кроз више врста и кроз више колона. Да бисмо све ово илустровали, проширимо претходну табелу на следећи начин.

```
<table border="1">
  <tr>
    <th rowspan="2" colspan="2">&nbsp;</th>
    <th colspan="2">Пол</th>
  </tr>
  <tr>
    <th>Мушки</th>
    <th>Женски</th>
  </tr>
  <tr>
    <th rowspan="2">Одељење</th>
    <th>I1</th>
    <td>18</td>
    <td>19</td>
  </tr>
  <tr>
    <th>I2</th>
    <td>20</td>
    <td>17</td>
  </tr>
</table>
```

**Слика:**

## Потпис: Сложенија табела са спојеним врстама и колонама.

Прва врста садржи празну ћелију која се простире кроз две врсте и две колоне, а затим и ћелију заглавља у којој пише `Пол` и која се простире кроз две колоне. Приликом описа друге врсте, њена прва ћелија се прескаче, јер је већ описана приликом описа претходне врсте, тако да се описују само две обичне ћелије заглавља у којима пише `Мушки` и `Женски`. Приликом описа треће врсте наводи се ћелија заглавља у којој пише `Одељење` и која се простире кроз две врсте, затим ћелија заглавља у којој пише `I1` и на крају две обичне ћелије у којима пише `18` и `19`. На крају, у последњој врсти опет прескачемо прву ћелију јер је она већ раније описана, тако да описујемо само ћелију заглавља у којој пише `I2` и на крају две обичне ћелије у којима пише `20` и `17`.

Још један пример табеле може бити табела коју додајемо у чланак у главном делу наше веб-странице школе који приказује број ученика по разредима.

```
<article>
  <h2>Број ученика по разредима</h2>
  <table>
    <tr> <th>&nbsp;</th>      <th colspan="2">Ученици</th>      </tr>
    <tr> <th>&nbsp;</th>      <th>Девојчице</th> <th>Деčаци</th> </tr>
    <tr> <th>1. разред</th> <td>120</td>      <td>125</td>      </tr>
    <tr> <th>2. разред</th> <td>118</td>      <td>123</td>      </tr>
    <tr> <th>3. разред</th> <td>125</td>      <td>119</td>      </tr>
    <tr> <th>4. разред</th> <td>119</td>      <td>120</td>      </tr>
  </table>
</article>
```

## Уметнути садржај

Веб-странице по правилу нису само текстуалне већ садрже различит мултимедијални садржај. Раније су се у странице могле уметати само слике, али језик HTML 5 донео је могућност уметања звука и видео-исечака.

## Елемент `img`

Слике се у веб-странице умећу коришћењем елемента `img`. Елемент нема садржај (па се често користи самозатварајућа ознака) и уз њега је обавезно навести атрибуте `src` и `alt`.

Атрибут `src` је обавезан и представља URL слике која се учитава и приказује. Слично као и код линкова, и код слика је могуће користити и апсолутно и релативно адресирање. Преузимање слика са других сајтова се често сматра лошом праксом (јер на тај начин оптерећујете туђе сервере који морају да доставе слике посетиоцима вашег сајта). Зато се релативно адресирање чешће користи и обично се као вредност атрибута `src` наводи само име датотеке са сликом (на пример, `src="image.jpg"`), под претпоставком да се слика која се укључује налази у истом директоријуму као и сама

HTML датотека. Понекад се слике сместе у посебан директоријум који је поддиректоријум директоријума у којем се налази HTML датотека и тада се слика додатно квалификује именом директоријума у коме се налази (на пример, ако се слика image.jpg налази у директоријуму images, она се укључује навођењем атрибута `src="images/image.jpg"`).

Атрибут `alt` представља алтернативни текст и значајан је ако кориснички агент из неког разлога није у могућности да прикаже слику, јер се у том случају кориснику приказује текст наведен као вредност атрибута `alt`.

Дакле, ако желимо да у нашу страницу уметнемо слику summer.jpg која се налази у директоријуму images, а која представља слику са летовања у природи, то можемо учинити на следећи начин.

```

```

Поред обавезних, уз слике се често наводе и атрибут `title` чија се вредност некада приказује у облачићу који се појави када се показивач миша нађе изнад слике, као и атрибути `width` и `height` који одређују димензије слике (задате најчешће у пикселима или у проценту димензије окружујућег елемента који садржи слику). Ако су наведене вредности у пикселима различите од природне димензије слике кориснички агенти проширују или сакупљају слику. У случају ширења слике обично се губи на њеном квалитету. У случају да се жели приказ слике у димензијама мањим од њених природних димензија, уместо да се то ради навођењем ових атрибута, препоручљиво је смањити слику и изменити њене природне димензије (коришћењем неког програма за обраду слика) како би се уштедело приликом преноса слике кроз мрежу. Дакле, пожељно је да вредности ових атрибута одговарају вредностима природне димензије слике. Иако се у том случају наизглед не добија ништа (приказ остаје идентичан као и у случају да ови атрибути нису наведени), навођење ових атрибута ипак има смисла јер се тиме корисничком агенту даје до знања колика је слика и пре него што се сама слика учита, тако да је могуће правилно распоредити остатак садржаја.

Проширимо сада наш школски сајт тако што ћемо у њен први чланак у главном делу странице уметнути слику на којој се виде вредни ученици који имају подигнуте руке и која ће се простирати целом ширином чланка.

```
<article>
  <h2>Škola vrednih đaka</h2>
  
  ...
</article>
```

**Слика:**

**Потпис:** Део сајта са уметнутом сликом

Слично томе, у елемент `div` који представља галерију сличица на дну стране убацимо три слике, чије ћемо димензије подесити касније, кроз CSS.

```
<div id="gallery">
  
  
  
</div>
```

**Пробај ово на рачунару.** Сними неку слику са интернета у исти директоријум у којем се налази нека једноставна веб-страница и затим је убади у ту страницу. Прилагоди јој величину коришћењем неког програма за обраду слика. Премести је у неки поддиректоријум (на пример, `images`) и промени HTML кôд тако да се слика исправно види.

## Елемент `iframe`

Елемент `iframe` омогућава да се у оквиру једног HTML документа (веб-странице) прикаже цео други HTML документ. Садржај овог елемента је обично празан и његова својства се подешавају атрибутима. Основни атрибути су му `width` и `height` којима се задају дужина односно ширина елемента у коме ће се укључени документ приказати, као и атрибут `src` који садржи веб-адресу (URL) документа који се укључује.

Многи познати веб-сајтови корисницима пружају спремне фрагменте које могу укључити у своје странице. На пример, сајт YouTube сугерише да се њихови видео снимци у веб-страницу укључују коришћењем елемента `iframe`. На пример, један говор Тима Бернерс-Лија, оснивача веба можете у свој сајт укључити у своју страницу на следећи начин:

```
<iframe width="560" height="315"
  src="https://www.youtube.com/embed/rCplocVemjo"
  frameborder="0" />
```

Приликом укључивања туђих докумената у своју веб-страницу треба бити обазрив. Прво, треба водити рачуна о ауторским правима садржаја, а друго, треба водити рачуна о безбедности јер укључена страница може садржати неке злонамерне скриптове.

**Ко жели да зна више?** Елемент `iframe` омогућава да се помоћу његовог атрибута `sandbox` фино подешавају дозволе укљученој веб-страници (на пример да се забрани извршавање скриптова или да се спречи слање података из попуњених формулара).

**Пробај ово на рачунару.** Учитај свој омиљени видео-запис на youtube-у, а затим га убади у неку једноставну веб-страницу.

## Елемент `audio, video, source`

На крају овог кратког прегледа основних елемената језика HTML прикажимо



елементе `audio` и `video` који служе за уметање аудио односно видео записа у веб-странице. Оба елемента садрже низ елемената `source` који указују на датотеке са аудио тј. видео записима. Имена датотека се наводе као вредности атрибута `src` и сва правила која су важила за атрибут `src` елемента `img` остају на снази (препоручује се коришћење релативног адресирања, при чему је пристојно датотеке држати на свом серверу). Поред атрибута `src` наводи се и атрибут `type` који описује тип тј. формат записа. Најчешће вредности овог атрибута су обично `audio/mpeg` (за mp3 датотеке), `audio/ogg` и `audio/wav` тј. `video/mp4` и `video/ogg`. Ако постоји више `source` елемената прегледач ће пустити први наведени запис чији формат препознаје. Поред елемената `source` у садржај елемента `audio` тј. `video` је пожељно уписати и неки текст који ће се приказивати ако прегледач није у могућности да прикаже одговарајући мултимедијални садржај. Уз елемент `video` се обично наводе и атрибути `width` тј. `height` које описују димензију приказа видео-материјала. Атрибут `autoplay` означава да се запис аутоматски пусти када се учита, док атрибут `controls` на екрану приказује контроле којима корисник може да контролише пуштање мултимедијалног материјала. Прикажимо још примере укључивања аудио и видео материјала.

```
<audio controls>
  <source src="sound.mp3" type="audio/mpeg" />
  <source src="sound.ogg" type="audio/ogg" />
  Ваш прегледач није у могућности да пусти аудио снимак.
</audio>
```

```
<video width="320" height="180" controls>
  <source src="movie.mp4" type="video/mpeg" />
  <source src="movie.ogg" type="video/ogg" />
  Ваш прегледач није у могућности да пусти видео снимак.
</video>
```

Слика:

Потпис: Аудио и видео снимци уметнути у веб-страницу

## Формулари

Језик HTML даје могућност креирања формулара помоћу којих посетиоци веб-сајтова могу да уносе податке који се након тога обрађују (било директно на рачунару клијента било тако што се пошаљу на сервер). Формулари донекле подсећају на форме (дијалогске прозоре) апликација какве сте интензивно користили у оквиру изучавања програмирања у трећем разреду и у њих се могу постављати све уобичајене контроле, попут поља за унос текста, натписа, дугмади и слично. С обзиром да наставним планом и програмом није предвиђено изучавање скрипт језика који би се користили за обраду података унетих помоћу формулара, овој теми ћемо се посветити само информативно и веома површно. Најчешће коришћени елементи за креирање формулара су следећи.

- `form` – овај елемент представља формулар и сви други елементи формулара (појединачне контроле) се постављају као садржај елемента `form`.
- `input` – овај елемент се користи за опис већег броја различитих контрола. Врста контроле одређена је атрибутом `type`, чије вредности могу бити следеће.
  - `text` – класично поље за унос текста,
  - `password` – поље за унос лозинке (уместо унетих карактера, приказују се обично тачкице или звездице),
  - `radio` – радио дугме,
  - `checkbox` – поље за штриклирање,
  - `button` – обично (акционо) дугме,
  - `submit` – специјално дугме чијим се активирањем прикупљају сви подаци унети у формулар и шаљу на сервер.

Поред атрибута `type`, обично се задају атрибути `name` којим се задаје име податка и `value` којим се одређује вредност податка. Да би више радио-дугмади било синхронизовано, потребно је да сва имају исту вредност атрибута `name`. Садржај елемента `input` је празан.

- `select` – (падајућа) листа. Појединачне ставке задају се елементом `option` чији садржај представља ставку која се види у листи. Елемент `option` се опет карактерише атрибутима `name` и `value` који су релевантни приликом слања података.
- `textarea` – поље за унос текста у више редова. Садржај овог елемента представља текст који се иницијално уписује у поље (а који корисник може да мења). Атрибут `name` именује податке који се уписују, док се атрибутима `width` и `height` одређују димензије поља.
- `label` – представља натпис чији се садржај види на екрану. Атрибут `for` може да садржи идентификатор неке контроле (обично `input`) и тада прегледачи успостављају одређену везу између контроле и натписа (на пример, када корисник кликне на натпис, контрола се ставља у жижу тј. фокус).

Прикажимо на крају један пример формулара (на пример, формулар за наручивање пице). Поменимо да се додатно стилизовање формулара и распоређивање контрола по формулару обично врши накнадно, помоћу језика CSS.

```
<form>
  <label for="ime">Ime i prezime: </label>
  <input type="text" id="ime" name="ime" />
  <br />

  <label for="ime">Lozinka: </label>
  <input type="password" id="lozinka" name="lozinka" />
  <br />

  <label for="adresa">Adresa: </label>
```

```

<textarea id="adresa" name="adresa"></textarea>
<br />

<label for="vrsta">Vrsta pice: </label>
<select id="vrsta">
  <option value="kapričoza">Kapričoza</option>
  <option value="margarita">Margarita</option>
  <option value="vegetarijana">Vegetarijana</option>
</select>
<br/>

<input type="radio" name="velicina" id="velika" value="velika"
  checked />
<label for="velika">Velika</label> <br/>
<input type="radio" name="velicina" id="srednja" value="srednja" />
<label for="srednja">Srednja</label> <br/>
<input type="radio" name="velicina" id="mala" value="mala" />
<label for="mala">Mala</label> <br/>

<input type="checkbox" name="kecap" id="kecap" />
<label for="kecap">Kečap</label>
<input type="checkbox" name="origano" id="origano" />
<label for="origano">Origano</label>
<br />

<input type="button" value="Poništi"
  onclick="alert('Još nije implementirano')"/>

<input type="submit" value="Naruči" />
</form>

```

Слика:

Потпис: Формулар за наручивање пице

## СТИЛСКИ ЛИСТОВИ – ЈЕЗИК CSS

Визуелна презентација (изглед, стил) се HTML документима подешава најчешће коришћењем стилских листова (енгл. *stylesheets*) описаних на језику CSS (енгл. *Cascading Style Sheets*). CSS је настао као резултат еволуције различитих језика за стилизовање веб-докумената и прва верзија је званично објављена као W3C препорука крајем 1996. године. Тренутно актуелна W3C препорука је верзија језика CSS 3 и неки њени делови ће бити описани у наставку овог поглавља.

Слика:

Потпис: Логотип језика CSS3

## Општа синтакса стилских листова

Основна синтакса језика CSS је прилично једноставна. Подешавање изгледа веб-страница описује се стилским листовима. Стилски листови могу бити задати унутар HTML документа и то у његовом заглављу, у оквиру елемента `style`, али и у посебним CSS документима који се укључују у HTML документ помоћу елемента `link`, о чему ће више речи бити касније. Стилски лист садржи низ **правила**. На пример,

```
p { color: red; }
h1 { font-family: Arial; margin: 20px; }
```

**Пробај ово на рачунару.** Креирај HTML страницу следећег садржаја и покушај да уочиш шта је ефекат наведених CSS декларација.

```
<!DOCTYPE html>
<html>
  <title>CSS</title>
  <style>
    p { color: red; }
    h1 { font-family: Arial; margin: 20px; }
  </style>
  <h1>Naslov</h1>
  <p>Ovo je prvi pasus</p>
</html>
```

Распоред белина у стилском листу није битан, па се, прегледности ради, стилски листови често "назубљују", слично коду у програмским језицима. На пример,

```
p {
  color: red;
}

h1 {
  font-family: Arial;
  margin: 20px;
}
```

Свако правило се састоји од **селектора** који одређују на које се елементе подешавања односе, и затим, од у витичастим заградама наведеног низа декларација, при чему су декларације међусобно раздвојене тачка-запетама (иза последње вредности није неопходно наводити тачка-запету, али често се она и ту наводи). Свака декларација садржи име **својства** које се подешава и након двотачке наведене његове нове **вредности**. На пример, у правилу:

```
p { color: red; margin: 10px; }
```

**p** је селектор и он означава да се врши подешавање свих пасуса у оквиру странице, а затим се кроз две декларације својству *color* се поставља вредност *red* (чиме се боја текста у пасусима поставља на црвену), а својству *margin* се поставља вредност *10px* (чиме се спољне маргине пасуса постављају на 10 пиксела). Селектори могу бити доста комплекснији од простог навођења имена елемента, о чему ће више речи бити касније. Могуће је навести и више селектора, раздвојених запетама. На пример, правило

```
h1, h2, h3 { color: blue; }
```

поставља плаву боју текста у насловима првог, другог и трећег нивоа.

Стилски листови могу да садрже и коментаре који се наводе између симбола */\** и *\*/*. На пример,

```
p { /* podešavamo sve pasuse */  
  color: red;      /* crvena boja teksta */  
  margin: 10px;   /* margina od 10 piksela */  
}
```

## Укључивање стилских листова у HTML документе

Аутори HTML докумената могу уметнути CSS описе у документе на неколико различитих начина.

### Описи на нивоу елемената (*atribući style*)

Један од начина да се појединачном елементу промени стил је да му се наведе атрибут *style* чији је садржај низ CSS својстава и њихових вредности (селектори се не наводе) које одредују изглед тог појединачног елемента. На пример,

```
<p style="color:red; margin-left:10px;">Ovo je pasus</p>
```

С обзиром на то да се на овај начин губе предности које доноси раздвајање описа структуре и визуелне презентације документа (јер описи презентације бивају разасути по целом документу и испреплетани са описом структуре), коришћење овог начина се препоручује само у изузетним случајевима.

### Описи на нивоу документа (*елемент style*)

Наредно место на коме се може навести CSS опис је елемент *style* у оквиру заглавља документа (елемента *head*). Уколико садржи опис на језику CSS елемент *style* би требало да има атрибут *type* са вредношћу *text/css*. CSS опис се наводи као садржај елемента *style*. На пример,

```
<head>
```

```
...
<style type="text/css">
  p { color : blue; }
</style>
...
</head>
```

CSS селектори се користе да би се одредило на које тачно елементе документа се појединачна наведена CSS својства и њихове вредности односе.

### **Спољашњи описи**

Опис наведен у заглављу документа (у оквиру елемента `style`) се односи искључиво на тај документ тј. односи се на појединачну веб-страницу. Са друге стране, често је пожељно на исти начин стилизовати већ број страница које заједно чине неки веб-сајт. Да би се то урадило на најбољи могући начин, користе се спољашњи CSS описи. Они су записани у засебним датотекама које садрже само CSS опис (најчешће са екстензијом `.css`). Иста датотека са CSS описом се затим може укључити у већи број HTML докумената (појединачних веб-страница). Овим се постиже униформан изглед читавог веб-сајта, као и то да се визуелна презентација читавог сајта може променити само изменом једне (или евентуално неколико) `.css` датотека.

Укључивање спољашњег CSS описа у HTML документ (појединачну веб-страницу) се најшешће врши коришћењем елемента `link` у заглављу документа (елементу `head`). У овом случају као атрибути елемента `link` морају се навести `rel` са вредношћу `stylesheet`, `type` са вредношћу `text/css` и атрибут `href` у коме се наводи УРИ спољашњег документа (најчешће само име `.css` датотеке, под претпоставком да се она налази на истом месту као и сам HTML документ). На пример,

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

Језик CSS допушта да се директивом `@import` у један стилски лист компетно увезе неки други стилски лист, тако да је ово још један од могућих начина коришћења спољашњих датотека са стилским описима.

```
<style type="text/css"> @import("stil.css") </style>
```

### **Наслеђивање стилских описа**

Нека својства се наслеђују кроз цело стабло документа, тј. у неким случајевима када се декларација стила придружи неком елементу, исту декларацију аутоматски наслеђују и сви елементи садржани у том елементу. На пример, ако се подеси боја слова тела документа, сви садржани елементи наслеђују наведену боју.

```
body { color : red; }
```

Са друге стране, наслеђено својство се може променити и експлицитно подешена вредност неког својства увек има предност у односу на наслеђену. На пример, ако после подешавање боје текста целог тела документа на црвену подесимо боју текста у свим пасусима на плаву

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

тада ће боја текста у свим пасусима бити плава, док ће боја текста ван пасуса бити црвена.

Нека својства се не наслеђују. На пример, ако се подеси маргина телу документа, то не значи да ће и сви садржани елементи имати исту маргину.

### ***Каскада стилских описа***

Стилски опис за неки елемент може бити истовремено наведен и на неколико различитих места. Аутори документа могу својим документима да придруже стилске описе на три претходно описана начина (у спољашњим стилским листовима, у оквиру елемента `style` и атрибута `style`). Корисник може својим подешавањима прегледача да утиче на стил и постоји стилски опис на нивоу корисника. Уколико ни аутор документа ни корисник не наведу стилски опис за неки елемент, онда се користи подразумевани стил тог елемента задат на нивоу корисничког агента (тј. прегледача веба). Различити описи за неки елемент се „сабирају” тј. комбинују како би се добио коначан стил приказа елемента. У случају да дође до конфликта, предност се даје стилским листовима аутора, затим стилским листовима корисника, док подразумевани стилски листови корисничких агената имају најмањи приоритет. У случају да до конфликта дође у оквиру стилског листа аутора, приоритет имају правила наведена на нивоу елемента (у оквиру атрибута `style`), затим правила наведена у на нивоу документа (у оквиру елемента `style`) и тек на крају правила наведена у спољашњим листовима (укључене елементом `link`). Ова особина се често у пракси користи, да би се променила нека ранија подешавања. На пример, ако се на нивоу целог веб-сајта промене маргине пасуса, то подешавање би било унето на нивоу спољашњег листа који се укључује у све странице сајта. Ако у некој страници желимо да променимо то подешавање, само је потребно да у њеном заглављу променимо подешавање маргине за све пасусе и то ново подешавање ће имати предност у односу на она подешавања учитана из спољашњег листа.

### **Најчешће коришћени селектори**

#### ***Називи елемената***

Најједноставнији облик селектора је селектор који се састоји само од назива елемента. У том случају се подешавања односе на све елементе са тим називом. На



пример, ако се наведе правило:

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

тада се свим пасусима (сви елементима `p`) у документу боја текста мења у црвену.

## Идентификатори

Понекад желимо да се подешавање стила односи само на један конкретан елемент. Једна могућност је да се сва стилска подешавања наведу у оквиру атрибута `style` тог елемента. На пример, плаву боју текста у неком пасусу можемо поставити на следећи начин.

```
<p style="color:blue">У овом пасусу биће описана главна својства ... </p>
```

Ипак, важан недостатак овог приступа је то што су стилска подешавања испреплетана са обележавањем структуре и садржаја документа. Много бољи приступ је тај да се елементу придружи неки идентификатор (навођењем атрибута `id`) и да се касније тај идентификатор искористи као CSS селектор (облика `#id`) да би се стилска подешавања применила на тај конкретан елемент. Идентификатор мора бити јединствен и то на нивоу читаве веб-странице (HTML документа), тј. у читавој страници не сме да постоји други елемент чији би идентификатор био једнак датом. На пример, ако бисмо желели да стилизујемо пасус који садржи неки опис, у телу документа тај пасус би смо обележили на следећи начин.

```
<p id="opis">У овом пасусу биће описана главна својства ... </p>
```

Подешавање боје би се онда урадило или у оквиру елемента `style` у заглављу документа или у засебној датотеци `.css`.

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

Селектор `p#opis` означава да је у питању пасус (елемент `p`) са идентификатором `opis`. Пошто је идентификатор јединствен, име елемента није неопходно наводити.

```
#opis { color : blue; }
```

Ипак, ако се наведе и име елемента читаоцу CSS правила може бити мало јасније шта се заправо стилизује, па ћемо име елемента увек наводити уз идентификатор.

## Класе

Понекад се жели стилизовање неколико конкретних елемената истовремено. Један начин је да се сваком елементу који се жели стилизовати на одабрани начин додели нека

класа (постављањем атрибута `class`), а затим да се та класа употреби као CSS селектор (облика `.class`). На пример, у документу можемо имати пуно пасуса, од којих се у неким од њих резимирају основне ствари наведене у претходним пасусима. Ако желимо да пасусе који представљају резимее уоквиримо и испишемо плавом бојом, то можемо урадити на следећи начин. Прво, у HTML документу пасусима који представљају резимее доделићемо класу `rezime`. Класа се додељује постављањем атрибута `class`.

```
<p class="rezime">Jezik HTML je ...</p>
...
<p class="rezime">Jezik CSS je ...</p>
```

У оквиру стилског описа (било у елементу `style`, било у засебном стилском листићу) извршили бисмо следеће подешавање.

```
p.rezime { color : blue; border: 1px solid black; }
```

Опет је име елемента могуће изоставити.

```
.rezime { color : blue; border: 1px solid black; }
```

Међутим, пошто више различитих елемената могу припадати истој класи (на пример, могуће је истовремено дефинисати и неки елемент `div` класе `rezime`, тј. Елемент облика `<div class="rezime">...</div>`), претходна два CSS правила не морају бити идентична. У првом се врши стилизовање свих пасуса класе `rezime`, а у другом се врши стилизовање свих елемената класе `rezime` (у нашем примеру, били би стилизовани елементи `p`, али и елемент `div`).

**Ко жели да зна више?** Приметимо да класе елемената одговарају стиливима у процесорима текста.

### ***Псеудокласе и псеудоелементи***

Најчешће придруживање стила свим елементима одређеног имена или свим елементима одређене класе задовољава потребе корисника. Медути, понекад се јавља потреба за финијом контролом. На пример, елемент `a` означава везе (линкове) у HTML документима, међутим, често се жели различит приказ за везе које су већ посећене од приказа веза које нису посећене. Како би се пружила подршка за оваква финија подешавања, у језику CSS се уводе псеудокласе и псеудоелементи. Неки од њих су и следећи.

- `:link` – означава везе које нису посећене (обично се наводи селектор `a:link`).
- `:visited` – означава везе које су посећене (обично се наводи селектор `a:visited`).
- `:hover` – означава елементе изнад који се тренутно налази показивач миша, а који још нису активирани (није се кликнуло мишем). На пример, правило

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

узрокује да се текст у пасусу изнад којег је показивач миша исписује црвеном бојом.

- **:active** – означава елементе који су управо активирани (кликнуто је мишем на њих или је притиснут тастер `enter` док су били у фокусу).
- **:focus** – означава елементе који су тренутно у фокусу тастатуре.
- **:first-line** – односи се на прву линију (најчешће пасуса).
- **:first-letter** – односи се на прво слово (најчешће пасуса).
- **:first-child** – односи се на елементе који су наведени први у оквиру неког елемента (прва су деца свог родитељског елемента).
- **:nth-child(n)** – односи се на елементе који су n-та деца својих родитеља. Вредност `n` може бити и `even` или `odd` чиме се одабиру само парна или непарна деца (ово се, на пример, често користи за стилизовање табела да би се свакој парној или непарној врсти доделио посебан стил).

### Селектори атрибута

Некада је потребно променити стил само елементима којима је наведен неки атрибут или којима неки атрибут има тачно одређену вредност. Ово се постиже селекторима облика `selector[attr]` или `selector[attr="value"]`. Тако се, на пример, правилом:

```
input[type="text"] { color : blue; }
```

свим пољима за унос текста (елементима `input` чија је вредност атрибута `type` једнака `text`).

### Угнежђени елементи

Понекад је пожељно променити стил само оних елемената који су садржани у неком ширем елементу (на пример, желимо да стилизујемо пасусе који се налазе у главном делу странице, тј. само пасусе који су садржани у елементу `main`). У таквим ситуацијама користе се селектори угнежђених елемената.

- **selector1 selector2** – Ако се два селектора наведу раздвојена размаком, тада се селекују сви елементи описан селектором `selector2` који се налазе у оквиру неког елемента описаног селектором `selector1`. На пример, наредним правилом се поставља плава боја текста за све пасусе који се налазе у оквиру централног дела странице.

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

Слично, на пример, наредним правилом се ширина свих слика класе `small`, а које се налазе у оквиру елемента `div` са идентификатором `gallery` поставља на 50 пиксела (о својству за постављање ширине, више речи биће касније).

```
div#gallery img.small { width: 50px; }
```

- **selector1 > selector2** – Понекад је потребно само одабрати елементе који су непосредни потомци датог елемента и то се постиже комбиновањем селектора оператором `>`. На пример, део `main` може да садржи неколико елемената `section`, при чему сваки од њих даље може да садржи неколико чланака и сваки чланак у себи може да има нове елементе `section`. Ако, на пример, желимо да уоквиримо само главне секције (али не и подсекције), то можемо урадити на следећи начин (о својствима за подешавање оквира више речи ће бити касније).

```
main > section { border: 1px solid black; }
```

- **selector1 + selector2** – Понекад је потребно одабрати елементе који следе непосредно иза датих елемената и то се постиже оператором `+`. На пример, свим пасусима који следе непосредно након главног наслова, можемо искључити увлачење прве линије (о подешавању увлачења прве линије биће више речи касније).

```
h1 + p { text-indent: 0; }
```

- **\*** – овим селектором се бирају сви елементи.

## Најчешће коришћена својства и њихове вредности

У овом поглављу биће наведена нека најчешће коришћена CSS својства и њихове најчешће коришћене вредности. Потпун преглед својстава и вредности могуће је наћи у званичној CSS спецификацији.

### Словни лик (фонт)

Наредна својства дефинишу изглед словних ликова тј. фонтова (енгл. *fonts*) који се користе приликом исписа текста. Сва ова својства се могу применити на било који елемент и аутоматски се наслеђују на све садржане елементе.

- **font-family** – Ово својство одредује фонт или фамилију фонтова који ће се користити приликом исписа текста. Могуће је навести прецизан назив фонтова, при чему се имена фонтова која имају више речи обично наводе под наводницима (нпр. "Times New Roman", Arial, "Courier New"), име фамилије фонтова (нпр. Times) или име врсте фонтова (нпр. serif, sans-serif, monospace, cursive, fantasy). **Серифни фонтови** (енгл. *serif*), на пример Times

New Roman или Cambria јесу они који на ивицама знакова имају мале детаље (серифе), рецимо на подножјима слова. **Безсерифни фонтови** (енгл. *sans-serif*), на пример Calibri или Arial, јесу они који немају серифе. **Непропорционални фонтови** (енгл. *monospace*), на пример Courier, јесу они који изгледају као да су откуцани на писаћој машини, код којих свако слово има потпуно исту ширину. Могуће је навести и више описа у опадајућем приоритету. Тако, на пример,

```
p { font-family: "New Century Schoolbook", Times, serif }
```

означава да се користи New Century Schoolbook фонт ако постоји на систему. Ако не постоји, користи се било који Times фонт, а ако не постоји, користи се било који серифни фонт.

**Ко жели да зна више?** Велики број фонтова који се могу користити на вебу доступан је на страници <https://www.google.com/fonts>.

- *font-style* – Ово својство одређује искошеност текста. Могуће вредности су *normal*, *italic* и *oblique*.
- *font-variant* – Ово својство даје могућност коришћења „МАЛИХ-ВЕЛИКИХ” слова. Могуће вредности су *normal* и *small-caps*.
- *font-weight* – Ово својство одређује „дебљину” слова. Најчешће вредности су *normal*, *bold* и *lighter*.
- *font-size* – Ово својство одређује величину фонта (то није величина ниједног конкретног карактера, већ је обично величина мало већа од висине која се израчунава као размак између најниже тачке која се може достићи и највише тачке која се може достићи у неком карактеру (не обавезно истом). Најчешће се вредности задају у апсолутним јединицама мере какве су pt или px. Тако, на пример, правило

```
p { font-size: 12pt; }
```

одређује да ће величина текста у свим пасусима бити 12 типографских тачака. Поред овога, могуће је наводити и релативне мере (у процентима) или у јединицама каква је em која представља умножак тренутне (подразумеване или наслеђене) величине фонта. Тако, на пример, правила

```
p { font-size: 120%; }  
p { font-size: 1.2em; }
```

оба одређују да ће величина текста бити 120% тренутне.

**Ко жели да зна више.** Иако би се по имену могло закључити да је величина од 1px величина једног пиксела на екрану, то није тако. У CSS3 стандарду пиксели су апсолутна јединица мере (један пиксел износи 96. део инча), и у зависности од резолуције уређаја за приказ као и нивоа зумираности може да заузме мањи или већи број физичких пиксела. Подсетимо се и да је величина типографске тачке 1pt једнака 72. делу инча (дакле, мало је већа од пиксела).

Препоручује се да се само основна величина фонта задаје апсолутним јединицама, а онда се остале величине фонтова постављају релативно, како би се постигло да односи у величини фонта између разних делова странице остану непромењени.

- *font* – Ово својство обједињује сва остала својства наведена у овој секцији и омогућава да се истовремено постави више различитих карактеристика фонта. На пример,

```
p { font: italic bold 12pt/14pt Times, serif; }
```

означава да у пасусима треба да се користи italic, bold фонт, величине 12 тачака, серије Times или неки други серифни фонт ако фонт серије Times није на располагању.

Слика:

Потпис: Својства фонта

### Стилизовање текста

Одређени број својстава посвећен је подешавању текста – могуће је подешавати увлачење прве линије, поравнавање текста, размак између речи и слова и слично. За подешавање текста користе се наредна CSS својства.

- *text-indent* – Ово својство дефинише увлачење прве линије текста у оквиру елемената. Ово својство се наслеђује. Вредност се најчешће наводи било у облику дужине или у облику процента тј. јединице em. На пример, наредно правило проузрокује да прва линија буде увучена два пута више од величине фонта (користи се јединица em која је, подсетимо се, једнака величини фонта, тако да се променом величине фонта пропорционално мења и увлачење).

```
p { text-indent: 5em; }
```

Слика:

Потпис: Својство `text-indent`

- `text-align` – Ово својство дефинише поравнавање текста у оквиру елемента. Поравнавање целог елемента у оквиру њему окружујућег елемента постиже се на друге начине (на пример, постављањем аутоматских маргина, о чему ће бити речи касније). Својство се наслеђује. Могуће су вредности `left` (поравнавање на леву страну), `right` (поравнавање на десну страну), `center` (центриран текст) или `justify` (текст поравнат на обе стране).

Слика:

Потпис: Својство `text-align`

- `text-decoration` – Овим својством се могу задати различите декорације текста. Могуће вредности су: `none` (текст нема декорација), `underline` (текст је подвучен), `overline` (текст је надвучен), `line-through` (текст је прецртан) и `blink` (текст „трепће“). На пример, линкови су обично подразумевано подвучени и дизајнери често искључују тај ефекат, осим када је миш изнад линка.

```
a { text-decoration: none; }
a:hover { text-decoration: underline; }
```

Слика:

Потпис: Својство `text-decoration`

- `text-transform` – Овим својством се задају различите аутоматске трансформације текста. Вредност `uppercase` проузрокује да се цео текст, без обзира на то како је откуцан, прикаже великим словима (на пример, OVO JE NEKI TEKST), `lowercase` проузрокује да се цео текст прикаже малим словима (на пример, ovo je neki tekst), а `capitalize` да прва слова речи приказују као велика (на пример, Ovo Je Neki Tekst). На пример, ако желимо да сва слова у свим главним насловима на страници буду велика, то можемо постићи на следећи начин.

```
h1 { text-transform: upperacase; }
```

Слика:

Потпис: Својство `text-transform`

- *letter-spacing* – Овим својством се подешава размак између слова (вредност је димензија обично задата у пикселима, тачкама или јединици *em*).
- *word-spacing* – Овим својством се подешава размак између речи (вредност је димензија обично задата у пикселима, тачкама или јединици *em*).

**Слика:**

**Потпис:** Својства *letter-spacing* и *word-spacing*

- *line-height* – Ово својство одређује висину једне линије текста, тј. невидљивог правоугаоног поља (кутијице) које садржи текст и у којем је текст центриран (та висина укључује приказани текст и простор изнад и испод њега). Својство *line-height* се обично користи да би се задао размак између суседних линија текста. Вредност се обично задаје као децимални број или проценат при чему се размак између линија израчунава као задати умножак величине фонта. На пример,

```
p { line-height: 1.5; }
```

поставља такозвани проред један-и-по, тј. одређује ће висина једне линије текста бити 150% величине текста унутар ње.

**Ко жели да зна више?** Вертикално поравнавање садржаја помоћу CSS-а није увек једноставно. Ако се жели поравнавање једне линије текста у оквиру неког ширег блока, једна од техника је да се својством *line-height* постави да висина те линије буде једнака висини окружујућег блока.

## Боје

Стилизовани документи често користе мноштво боја. Обично се поставља боја текста, боја позадине докумената, боја оквира, и слично. На пример, боја текста се одређује помоћу својства *color*.

- *color* – Ово својство одређује боју текста који се налази у елементу. Вредност овог својства се наслеђује (на пример, ако се постави боја текста на нивоу целог тела веб-странице, ту боју ће имати и текст у свим пасусима који се налазе у телу).

Својства којима се подешава боја позадине елемената или боја оквира неког елемента биће приказана касније.

Боје се могу задати на неколико начина. Најкоришћенији начини да се зада боја тј. нијанса боје су следећи:

- **име** – боја се задаје преко имена боје (на пример, *red*, *yellow*, *black*, ...). На пример, наредно правило поставља боју текста у пасусима на црвену:

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



- **хексадекадни кôд облика #rrggbb** – боја се задаје задавањем три двоцифрена хексадекадна броја који одређују редом црвену, зелену и плаву компоненту боје. Вредност 00 је најмања, а вредност ff је највећа. Свеједно је да ли се пишу велика или мала хексадекадна слова. На пример, вредност #00ff00 означава зелену боју (вредност 0 код црвене компоненте и зелене компоненте и максимална могућа вредност код плаве компоненте). Стога, наредно правило поставља сав текст у пасусима на зелену боју:

```
p { color : #00ff00; }
```

Често се користе нијансе сиве боје које се одликују по томе што су им вредности црвене, зелене и плаве боје једнаке (на пример, #000000 је црна боја, #808080 је сива боја, док је #ffffff бела боја).

Слика:

Потпис: Неки хексадекадни кодови боја

- **хексадекадни кôд облика #rgb** – боја се може задати и задавањем три хексадекадне цифре. Запис #rgb представља скраћени облик записа #rrggbb. На пример, запис #27c је скраћени запис за боју #2277cc.
- **Декадна спецификација облика rgb(r, g, b)** – боја се задаје задавањем три декадна броја (између 0 и 255) који одредују редом црвену, зелену и плаву компоненту боје. Овај начин прилично одговара претходном, једино што се вредности задају у декадном, уместо у хексадекадном бројевном систему. На пример, наредним правилом се боја текста у свим пасусима поставља на плаву:

```
p { color : rgb(0, 0, 255); }
```

Слика:

Потпис: Својство color

## Модел куџија

Сви HTML елементи могу да се сматрају правоугаоним површинама — куџијама, (енгл. *boxes*). Свака куџија има свој **садржај** (енгл. *content*) и може да има свој **оквир** (енг. *border*). Оквир је раздвојен од садржаја **унутрашњом маргином** тј. **пуњењем** (енгл. *padding*), а од околних елемената **спољашњом маргином** (енгл. *margin*).

Слика:

Потпис: Модел куџије

Сва својства куџија подешавају се наредним CSS својствима.

- *width*, *height*, *box-sizing* – Својства *width* и *height* служе за постављање ширине и висине елемента (међутим, како ћемо касније видети, нису примењива на све елементе). Пуњење, оквир и маргине подразумевано нису урачунате, али се то може променити својством *box-sizing*. Својство *box-sizing* одређује шта се урачунава у ширину и висину елемента.
  - Подразумевана вредност је *content-box* и тада се рачуна само ширина и висина само садржаја (без пуњења, оквира и маргина).
  - Са друге стране, ако се наведе вредност *border-box* рачунају се ширина и висина садржаја, пуњења и оквира (без спољашњих маргина). Ово може понекад бити zgodno јер дизајнер често зна колико простора на страни неки елемент треба да заузме (од једне до наспрамне ивице оквира) и ако користи *border-box* прерачунавање, тада не мора да од те вредности одузима вредности дебљине оквира и пуњење да би поставио ширину елемента (што би морао да ради ако се користи *content-box*).

Слика:

Потпис: Однос између *border-box* и *content-box*

- *max-width*, *max-height*, *min-width*, *min-height* – Ова својства одређују максималну односно минималну висину и ширину елемента. Шта се у ширину и висину урачунава, зависи од вредности својства *box-sizing* (исто као у случају својстава *width* и *height*). Ова својства могу да имају неке предности у односу на експлицитно постављање ширине и висине (својствима *width* и *height*). На пример, када се екран сузи, кутијице којима је постављена максимална ширина се такође сужавају, за разлику од оних којима је ширина фиксна.
- *margin* – Овим својством подешава се спољашња маргина. Када се као вредност својства *margin* наведе једна димензија (на пример, *margin: 10px*) подразумева се да се она односи на све четири маргине (леву, десну, горњу и доњу). Ако се наведу две димензије (на пример, *margin: 10px 20px*) прва од њих се односи на леву и десну, а друга на горњу и доњу маргину. Ако се наведу четири димензије оне се односе редом на леву, горњу, десну и доњу маргину (наводе се, дакле, у правцу казаљке на сату). Вредност *auto* проузрокује да се маргине аутоматски подједнако распореде што се јако често користи у центрирању елемената. Могуће је засебно подешавати сваку од четири маргине и то атрибутима *margin-top*, *margin-right*, *margin-bottom* и *margin-left*. Маргина је најмањи простор око елемента који ће сигурно бити остављен празан. То значи да ако постоје два елемента тако да је за сваки од њих постављена маргина од 10 пиксела, један испод другог или један поред другог, простор између њих ће бити 10, а не 20 пиксела, како би се можда могло очекивати. Дакле, маргине суседних елемената се не сабирају, већ се за простор између елемената оставља већа од две наведене маргине.

Слика:

Потпис: Маргине суседних елемената

- `padding` – Овим својство подешава се унутрашња маргина (тј. пуњење) и оно се користи аналогно својству маргин. Појединачне димензије могу се подешавати својствима `padding-top`, `padding-right`, `padding-bottom` и `padding-left`.
- `border-width` – Овим својством се подешава дебљина оквира (на пример, `border-width: 3px`). Појединачни оквири (леви, горњи, десни и доњи) се могу подешавати својствима `border-top-width`, `border-right-width`, `border-bottom-width` и `border-left-width`.
- `border-style` – Овим се својством подешава тип линије оквира (вредности су `solid` за оквир исцртан пуном линијом, `dashed` за испрекидане линије, `dotted` за тачкасти оквир итд.). Опет се могу подешавати и појединачни оквири и то својствима `border-top-style`, `border-right-style`, `border-bottom-style` и `border-left-style`.

Слика:

Потпис: врсте оквира

- `border-color` – Овим својством се подешава боја оквира. Опет се могу подешавати и појединачни оквири и то својствима `border-top-color` и `border-right-color`, `border-bottom-color` и `border-left-color`.
- `border` – Ово својство комбинује сва три аспекта оквира (дебљину, врсту линије и боју). На пример, `border: 1px solid black` креира танки, црни оквир, исцртан пуном линијом. Појединачни оквири се могу подешавати својствима `border-top`, `border-right`, `border-bottom` и `border-left`.
- `border-radius` – Овим својством одређује се заобљеност оквира. Вредност се задаје обично у пикселима и одређује полупречник замишљених кругова уписаних у сваки од 4 темена правоугаоног поља. Што је већи полупречник, то је оквир заобљенији.

Слика:

Потпис: заобљеност оквира

## Позадина

Сваком елементу је могуће модификовати позадину коришћењем CSS-а. Могуће је поставити боју позадине, али и у позадину поставити неку слику и додатно подесити многа њена својства. Позадина може да обухвата само садржај елемента, садржај и пуњење или садржај, пуњење и оквир (што је подразумевано).

- `background-color` – Ово својство одређује боју позадине елемента. Својство се наслеђује. Осим спецификације боје, вредност овог својства може да буде и

`transparent` чиме се елемент чини провидним.

Слика:

Потпис: Елемент са бојом позадине

Осим задавања боје, позадина неког елемента може бити и слика. Својства која се користе за постављање и подешавања слика у позадинних елемената су следећа.

- `background-image` – Ово својство одредује слику која ће бити приказана као позадина елемента. Својство се не наслеђује. Могуће је додатно подесити неке параметре приказа слика (нпр. позицију, начин понављања). Могуће вредности су попе када се не користи слика или URI слике у облику `url(...)`. Препоручује се да се приликом коришћења овог својства постави и боја позадине (коришћењем својства `background-color`). Боја се приказује у случају да слика није доступна, и приказује се на оним деловима елемента где слика није постављена или је провидна.

Слика:

Потпис: Елемент са бојом позадине

- `background-repeat` – У случају када је елементу постављена позадинска слика (коришћењем својства `background-image`), овим својством се подешава да ли ће слика бити приказана само једном или ће бити понављана док не испуни целу ширину и/или висину елемента. Могуће вредности су:
  - `repeat` – слика се понавља и хоризонтално и вертикално.
  - `repeat-x` – слика се понавља само хоризонтално.
  - `repeat-y` – слика се понавља само вертикално.
  - `no-repeat` – слика се не понавља (приказује се само једна копија)

Слика:

Потпис: Понављање слике

- `background-attachment` – У случају када је елементу постављена позадинска слика (коришћењем својства `background-image`), овим својством се одредује да ли ће се приликом померања садржаја странице (тзв. скроловања) слика померати заједно са садржајем или ће остати фиксна у односу на прозор и посматрача. У првом случају наводи се вредност `scroll`, а у другом `fixed`.
- `background-position` – У случају када је елементу постављена позадинска слика (коришћењем својства `background-image`), овим својством се одређује

његова почетна позиција. Уколико се наведу две вредности, прва одредује хоризонталну, а друга вертикалну позицију. Уколико се наведе само једна вредност, за другу се подразумева центер. Вредности могу бити наведене на следећи начин:

- проценат – Вредност од x% поравнава тачку која се налази на x% ширине (дужине) слике са тачком која се налази на x% ширине (дужине) елемента. Тако нпр. 0% 0% поравнава горња лева темена, док 100% 100% поравнава доња десна.
- дужина – горње лево теме слике се поставља на тачку померену за наведене вредности у односу на горње лево теме елемента.
- `top`, `bottom` – еквивалентно 0%, односно 100% за вертикалну позицију.
- `left`, `right` – еквивалентно 0%, односно 100% за хоризонталну позицију.
- `center` – еквивалентно 50% за хоризонталну позицију (ако она није наведена) или за вертикалну позицију (ако је хоризонтална позиција наведена).

Слика:

Потпис: Позиција слике у позадини

- `background` – Ово својство омогућава да се истовремено наведе више аспеката позадине. На пример, уместо:

```
body {
  background-color: #aaaaaa;
  background-image: url('background.png')
  background-repeat: repeat-x;
  background-attachment: scroll;
  background-position: right top;
}
```

могуће је користити:

```
body {
  background: #aaaaaa url("background.png") repeat-x scroll right top;
}
```

- `background-clip` – Ово својство одређује шта се тачно сматра позадином елемента. Подразумевана вредност је `border-box` што подразумева да се позадином сматра садржај, пуњење и оквир. Вредност `padding-box` подразумева да се позадином сматрају садржај и пуњење, док вредност `content-box` подразумева да је позадина само позадина садржаја, без пуњења и оквира.

Слика:

Потпис: Својство `background-clip`

## Стилизовање листӣа

И нумерисане и нумерисане листе се могу стилизовати. Најчешћа подешавања односе се на подешавање ознаке ставки листе (тачкица, тј. цртица у нумерисаним и бројки тј. слова у нумерисаним листама). За то се користе наредна CSS својства.

- `list-style-type` – Ово својство дефинише облик „цртице” у нумерисаној листи или броја у нумерисаној листи (најчешће вредности су `disc` (испуњен круг), `circle` (празан кружић), `square` (квадрат), `decimal` (број), `lower-alpha` (мала латиничка слова a, b, c, ...), `lower-roman` (мали римски бројеви i, ii, iii, iv, ...), `upper-alpha` (велика латиничка слова A, B, C, ...), `upper-roman` (велики римски бројеви I, II, III, IV, ...) итд.).
- `list-style-image` – Уместо „цртице” може се приказати и нека слика, која се задаје постављањем овог својства. На пример, ако желимо да се слика `slika.png` снимљена у исти директоријум као и HTML документ прикаже уместо тачкица у нумерисаним листама, то можемо урадити на следећи начин.

```
ul { list-style-image: url("slika.png"); }
```

Слика:

Потпис: слика као ознака листе

- `list-style-position` – Ово својство одређује да ли ће се цртице, бројеви или слике приказивати унутар листе (тј. унутар њеног оквира) или ван ње. Вредности овог својства су `inside` и `outside`.
- `list-style` – Ово својство обједињује претходна својства, тако што се прво наведе вредност својства `list-style-type`, `list-style-position` и `list-style-image`.

## Стилизовање табела

`border-collapse` – Ово веома често коришћено својство уз вредност `collapse` проузрокује да се суседне ћелије „прилепе” тако да између тако да између њих нема простора и тако да се између суседних поставља јединствени оквир (у супротном свака ћелија има свој посебни оквир).

`text-align`, `vertical-align` – ова својства се користе да би се одредило поравнавање садржаја ћелија табеле. Приметимо да је и вертикално поравнавање садржаја прилично једноставно (што није увек случај), коришћењем својства `vertical-align` чије су најчешће коришћене вредности `top`, `middle` и `bottom`.

`border`, `background-color`, `padding`, `margin` – ова својства су веома важна за стилизовање табела, а користе се на потпуно исти начин као и код свих других елемената. Нагласимо да је ове атрибуте могуће постављати на нивоу табеле (елемента `table`), на нивоу појединачних редова (елемента `tr`) и појединачних ћелија (елемента

td).

## Приказ

Сви елементи заузимају одређени простор правоугаоног облика на екрану, али постоје разлике у томе како се ти правоугаоници формирају (на пример, колики простор заузимају) и како се слажу један уз други. Изузетно важно својство, које одређује начин на који ће неки елемент бити приказан је својство `display`. На основу њиховог подразумеваног начина приказа, елементи се углавном деле на **блок-елементе** (енгл. *block elements*) чија је подразумевана декларација `display: block` и на **линијске елементе** (енгл. *inline elements*) чија је подразумевана декларација приказа `display: inline`. Блок елементи су крупнији елементи који у себи могу да садрже текст, линијске елементе, али и друге блок елементе, док су линијски елементи ситнији елементи који у себи могу да садрже само текст и друге линијске елементе (али не и блок елементе). Приликом приказа, блок елементи се простиру целом ширином простора и слажу се један испод другог (сваки ново наведени елемент, заузима посебан ред). Линијски елементи се приказују уже (обично заузимају само онолико простора колики им је садржај) и слажу се један поред другог (не започињу аутоматски следећи ред).

На пример, пасуси су блок-елементи, док су линкови линијски елементи. Зато се у наредном примеру пасуси (уоквирени црном бојом) протежу целом ширином екрана и слажу један испод другог, док су линкови (уоквирени црвеном бојом) смештени у правоуганике довољне само да се прикаже текст линка.

```
<p>
  Nekoliko korisnih linkova: <a href="http://www.w3.org">w3c</a>,
  <a href="http://www.w3schools.com/">w3schools</a>
</p>
<p>Puno sreće u učenju veb-tehnologija.</p>
```

**Слика:**

**Потпис: Однос између блок и линијских елемената**

Елементи `div`, `section`, `article`, `header`, `footer`, `main`, `aside`, `form`, `p`, `ul`, `ol`, `li` и слично су блок-елементи, док су елементи `span`, `a`, `img`, `em`, `strong`, `small`, `i`, `b`, `u`, `sub`, `sup` и слично линијски. Табеле нису ни блок ни линијски елементи.

Својство `display` може имати много различитих вредности, при чему се најчешће користе следеће:

- `none` – ова вредност означава да елемент не треба да се прикаже уопште, већ да се потпуно изостави из приказа (самим тим, не заузима никакав простор на страници и околни елементи се распоређују као да он није ту). Својство `display: none` се обично користи у комбинацији са језиком JavaScript да би се

реагујући на акције корисника неки елементи динамички приказали или сакрили.

**Ко жели да зна више?** Декларација слична декларацији `display: none` је декларација `visibility: hidden`, међутим, у том случају се елемент не приказује, али заузима простор на страници.

- `block` – ова вредност означава да се неки елемент приказује као блок-елемент. Елементу је могуће постављати ширину, висину, оквир и маргине на уобичајени начин.
- `inline` – ова вредност означава да се неки елемент приказује као линијски елемент. Оваком елементу је могуће подешавати оквир на уобичајени начин, али није могуће подешавати ширину, висину је могуће постављати својством `line-height` (а не својством `height`), док се унутрашња и спољашња маргина могу поставити (својствима попут `margin` и `padding`), међутим, само лева и десна маргина померају околни садржај и стога се горња и доња унутрашња и спољашња маргина код линијских елемената обично и не постављају.

Слика:

Потпис: Маргине линијског елемента

- `inline-block` – ова вредност даје приказ у облику линијског блока – елемента који се донекле понаша као линијски елемент, јер се не простире целом ширином и слаже се уз, а не испод и изнад својих суседних елемената, а донекле као блок елемент јер му је могуће постављати ширину, висину и маргине на потпуно исти начин као код блок елемената.

Слика:

Потпис: `display: inline-block`

Једна од могућих употреба овог начина приказа је на пример, да се елементи истих димензија сложе у матрицу:

```
div { border: 1px solid red; }
div.inblock {
  display: inline-block;
  width: 100px; height: 50px;
  margin: 10px;
}
```

```
<div>Ispred</div>
<div class="inblock">1</div>
<div class="inblock">2</div>
<div class="inblock">3</div>
<div class="inblock">4</div>
<div class="inblock">5</div>
<div class="inblock">6</div>
```



```
<div>Iza</div>
```

Слика:

Потпис: Елементи сложени коришћењем приказа `display: inline-block`

Још једна честа употреба овог начина приказа је креирање хоризонталне траке менија (такав пример ће бити приказан касније).

- `table`, `table-row`, `table-cell` – табеле, врсте табела и ћелије табела имају посебне начине приказа, који се могу употребити и код других елемената. На пример, једна веома корисна особина ћелија табеле је та што је могуће веома једноставно постићи вертикално центрирање садржаја (помоћу `vertical-align: middle`). Стога се некада садржај центрира вертикално у неком елементу тако што му се приказ постави на `table-cell`.

## Позиционирање

Позиционирање елемената врши се постављањем њиховог својства `position`. Могуће су четири вредности: `static`, `relative`, `absolute` и `fixed`.

- `static` – **Статичко позиционирање** је подразумевано (елементи којима вредност атрибута `position` није постављена имају статичко позиционирање). То подразумева да се они уклапају у такозвани **нормални ток** (енгл. **normal flow**). Елементи нивоа блока се постављају један испод другог, док се линијски елементи слажу један уз други.
- `relative` – **Релативно позиционирање** омогућава да се елементи помере у односу на своју статичку позицију, тј. да се изместе из нормалног тока, при чему то никако не утиче на позиционирање осталих елемената. Координате се задају својствима `top`, `left`, `bottom` и `right`. На пример,

```
p {  
  position: relative;  
  top: -10px;  
  left: 20px;  
}
```

проузрокује да се сви пасуси помере 10 пиксела горе и 20 пиксела десно у односу на њихову нормалну позицију.

Слика:

Потпис: Релативно позиционирани пасуси

Променом позиционирања елементи могу и преклопити, а који ће бити изнад, а који испод одредује се својством `z-index` – веће вредности одредују елементе који ће бити изнад тј. који ће се видети. Нагласимо и да се релативно

позиционирање често користи у комбинацији са апсолутним и да се декларација `position: relative` без промене координата често користи као део апсолутног позиционирања које следеће описујемо.

- **absolute** – **Апсолутно позиционирање** доводи до тога да се елемент позиционира у оквиру неког ширег елемента и то тако што се задају његове координате у односу на тај шири елемент. По дефиницији, то је најближи елемент који га садржи а позициониран је некако тј. има вредност својства `position` различиту од подразумеване вредности `static`. Ако такав елемент не постоји, апсолутно позиционирање се врши у односу на елемент `body`. Да би се нагласило да ће се неки елемент користити као елемент у коме ће се садржај позиционирати апсолутно, њему се обично само вредност својства `position` постави на `relative`. Апсолутно позиционирани елемент се избацује из нормалног тока и остали елементи се позиционирају као да он не постоји. Координате апсолутно позиционираних елемената се опет задају својствима `top`, `left`, `bottom` и `right`. Размотримо један пример. Нека у HTML коду постоје следећи елементи:

```
<div id="container">
  <div id="content">ABCD</div>
  <div>EFGH</div>
</div>
```

и нека је дат следећи стилски лист.

```
div {
  border: 1px solid black;
}

#container {
  position: relative;
  width: 500px;
  height: 200px;
  top: 50; left: 70;
}

#content {
  position: absolute;
  top: 50px; left: 50px;
}
```

Тада се елемент са идентификатором `content` позиционира тако да му је горњи леви угао 50 пиксела испод и 70 пиксела лево од горњег левог угла елемента `container`. Елемент у коме пише EFGH налази се у врху елемента са идентификатором `container` јер је елемент са идентификатором `content` избачен из нормалног тока.

Слика:

Потпис: Апсолутно позиционирани елемент.

- **fixed** – Фиксно позиционирање је донекле слично апсолутном, али се елемент позиционира у односу на екран прегледача и не помера се приликом скроловања странице. Најчешће се користи за фиксирање наслова или потписа странице. Пошто се и фиксно позиционирани елементи ибацују из нормалног тока, ако се жели обезбедити да не преклапају друге елементе, потребно је тим другим елементима поставити одговарајуће маргине.

Слика:

Потпис: Фиксно позиционирани елемент.

## Плутајући елементи

Неки елементи могу да буду „плутајући” тј. „лебдећи”, тј. да се подесе тако да се померају у леви или десни крај окружујућег елемента, а да се остали садржај „обмотава” око њих (слично као што се текст обмотава око слика у новинама). Ово се постиже својством `float`, чије су вредности `left` и `right`. Када два суседна елемента имају исту вредност својства `float`, они се позиционирају један до другог (на пример, ако је вредност `left` први елемент ће се померити налево, до почетка окружујућег елемента, а други ће одлебдети налево, али само до краја првог лебдећег елемента). Ако се жели да се лебдећи елемент сложи испод, а не поред другог лебдећег елемента, онда је потребно навести му својство `clear` (могуће вредности су `left`, `right` и `both`). На пример, наредне три слике биће позиционирание као на слици:

```
<div>
  
  
  
  Lorem ipsum dolor sit amet, ...
</div>
```

Слика:

Потпис: Плутајући елементи

## Пример стилизовања веб-странице

Покажимо сада пример стилизовања сајта гимназије. Без додатних стилских подешавања страница изгледа као што је то приказано на следећој слици.

# Gimnazija „Dositej Obradović”

- [O školi](#)
- [Zaposleni i učenici](#)
- [Raspored časova](#)
- [Kontakt](#)

## Najnovije vesti

### Uspeh naših matematičara

Na okružnom takmičenju iz matematike u organizaciji [društva matematičara Srbije](#) održanom u subotu, 16. januara, naši učenici su ostvarili odličan uspeh. Dvanaest učenika se plasiralo na republičko takmičenje, koje će biti održano u Novom Sadu u martu mesecu.

Čestitamo svim učesnicima takmičenja i profesorima koji su ih spremali.

## Ostale vesti

### Novi raspored časova

Od 15. oktobra 2016. godine radi se po novom [rasporedu časova](#).

### Sekcija iz istorije

Profesor Miloje Milojević poziva sve zainteresovane učenike da prisustvuju sekciji iz istorije koja će se održavati svakog četvrtka kao prečas u poslepodnevnoj smeni

## Škola vrednih đaka



Nema sramotnijeg zanata od dangube, besposlice i lenjosti. *Dositej Obradović*

Dobrodošli na sajt naše gimnazije. Mi se ponosimo svojim vrednim đacima.

## Broj učenika po razredima

	Učenici	
	Devojčice	Dečaci
1. razred	120	125
2. razred	118	123
3. razred	125	119
4. razred	119	120

## Upis u novu školsku godinu

Na upis je potrebno doneti sledeće dokumente:

1. **Svedočanstvo** iz prethodnog razreda,
2. **Izvod** iz matične knjige rođenih,
3. **Đačku knjižicu**.

## Kontakt

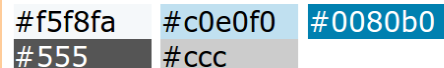
Gimnazija „Dositej Obradović”,  
Bulevar oslobođenja 38,  
34000 Kragujevac  
Telefon: 034/123-456, e-mail: [sekretarijat@gimnazija.rs](mailto:sekretarijat@gimnazija.rs)



Autor: Filip Marić, poslednja izmena: 1. 1. 2016.

Једна од важних ствари које треба имати на уму пре него што се упустимо у детаље

је да желимо да постигнемо уједначеност нијанси боја. Бирамо да основна боја сајта буде плава и да њу комбинујемо са нијансама сиве боје. Одлучујемо се да користимо следећу палету боја која садржи три нијансе плаве боја (једну, веома светлу, за позадину, једну средње тамну за оквири и једну веома тамну за заглавља и наслове).



#f5f8fa	#c0e0f0	#0080b0
#555	#ccc	

Слика:

Потпис: Палета боја

**Занимљивост.** Facebook је цео у нијансама плаве боје. Једно објашњење за то је то што је његов оснивач далтониста и што не разазнаје добро црвене и зелене нијансе.

Пошто не желимо да се странице неограничено шире на широким екранима ограничавамо димензију тела на 900 пиксела, при чему сав садржај центрирамо постављајући леву и десну маргину на вредност auto, а горњу и доњу маргину на вредност 0. Одлучујемо се да користимо безсерифни фонт Verdana у целом документу (и зато га постављамо на нивоу тела документа), а боју позадине тела постављамо на најсветлију нијансу плаве боје из палете за коју смо се одлучили.

```
body { /* telo */
  width: 900px;           /* širina tela */
  margin: 0 auto;        /* centriranje pomoću automatskih margina */
  font-family: Verdana;  /* font */
  background-color: #f5f8fa; /* svetlo-plava pozadina */
}
```

Примећујемо да се линкови на страници са својим подразумеваним бојама (плавом и љубичастом) не уклапају у нашу основну палету боја и стога им мењамо боје (боје непосећених линкова постављамо на тамну плаву боју из наше палете, док боје посећених постављамо тек на мало тамније). Такође, уклањамо ефекат подвучених линкова, осим у случају када је миш изнад њих.

```
a { /* svi linkovi */
  color: #0080b0;          /* tamno-plava boja teksta */
  text-decoration: none;   /* bez podvlačenja */
}

a:hover { /* linkovi dok je miš iznad njih */
  text-decoration: underline; /* podvlačenje */
}

a:visited { /* posećeni linkovi */
```

```
color: #005080; /* malo tamnije plava boja teksta */
}
```

Пређимо сада на позиционирање дела са вестима (елемента `aside`). Желимо да га поставимо у десни део стране и да заузима око трећине ширине тела. Означићемо га као плутајући елемент (`float: right`), а ширину ћемо му поставити на 300 пиксела (елемент подразумевано нема ни оквир, ни унутрашњу ни спољашњу маргину). Да бисмо обезбедили да се централни садржај (елемент `main`) не преклапа са плутајућим вестима на десној страни странице, поставићемо му маргину на 310 пиксела (300 пиксела одговара ширини вести, а 10 пиксела је додатан размак између централног дела и њих). Да бисмо обезбедили да потпис странице буде испод вести и централног дела странице, поставићемо му атрибут `clear` на вредност `right` (спуштамо га испод десног плутајућег елемента).

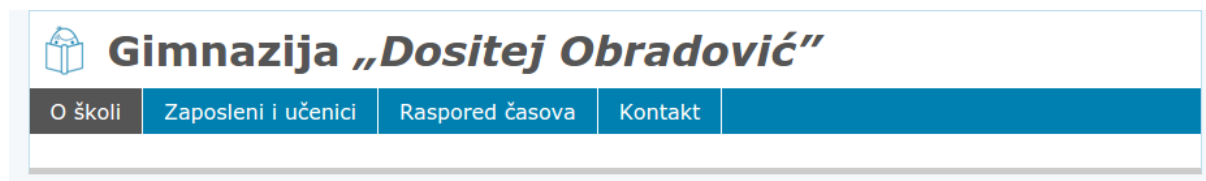
```
aside {
    width: 300px;
    float: right;
}

main {
    margin-right: 310px;
}

footer {
    clear: right;
}
```

У овом тренутку страница изгледа као што је приказано на следећој слици.

Сада можемо прећи на стилизовање заглавља странице. Желимо да након стилизовања оно изгледа као на следећој слици:



Стилизујмо за почетак листу навигационих линкова. За почетак, не желимо да видимо тачкице испред ставки, па својство `list-style-type` постављамо на вредност `none`. Постоји неколико техника да се постигне да ставке листе буду распоређене једна до друге (хоризонтално), уместо једна испод друге (вертикално). Основна идеја ће бити да ставке (елементе `li`) поставимо да буду лебдеће налево (`float: left`). Када се то

уради, у листи (елементу `ul`) не остаје никакав садржај и његова се димензија неће исправно поставити (елементи се по подразумеваном понашању не шире тако да обухвате и лебдеће елементе у њима, већ само обичан, статички позициониран садржај). Зато је листи потребно поставити вредност својства `overflow` на `auto` да би се он аутоматски проширио тако да обухвати и елементе `li` који у њему лебде. Још једно битно подешавање је постављање вредности својства `display` елемената `a` на `block` (чиме им дајемо могућност да добију и горњу и доњу унутрашњу маргину). Остала подешавања су естетског карактера (подешавања боја и маргина). Нагласимо да смо све селекторе писали хијерархијски (на пример, селектор `nav ul li` означава све елементе `li` који се налазе унутар елемента `ul` који се налази унутар елемента `nav`) да бисмо били сигурни да се ова подешавања односе само на листу навигационих линкова, а не и на друге ненумерисане листе које се налазе на странице.

```
nav ul { /* nenumerisane liste unutar navigacije */
  list-style-type: none; /* isključujemo tačkice ispred stavki */
  margin: 0; /* isključujemo spoljašnje margine */
  padding: 0; /* isključujemo unutrašnje margine */
  background-color: #0080b0; /* boja pozadine je tamno plava */
  overflow: auto; /* automatsko širenje da bi se obuhvatile
  i lebdeće stavke */
}

nav ul li { /* stavke nenumerisanih lista unutar navigacije */
  float: left; /* lebdenje nalevo */
  border-right: 1px solid white; /* tanak beli desni okvir - razmak
  u odnosu na sledeću stavku */
}

nav ul li a { /* linkovi u stavkama liste unutar navigacije */
  display: block; /* prikaz u obliku blok elementa */
  color: white; /* tekst bele boje */
  padding: 8px 16px; /* unutrašnje margine */
  text-align: center; /* centralno poravnat tekst */
}

nav ul li a:visited {
  color: white;
}
```

Додајмо још и два интересантна ефекта. Прво, линк који води ка тренутној страни обојимо другачије, да би посетиоци сајта лакше могли да знају на којој се страни сајта тренутно налазе. Тренутну страну обележићемо класом `active`.

```
<nav>
  <ul>
    <li class="active"><a href="skola.html">O školi</a></li>
    <li><a href="zaposleni.html">Zaposleni i učenici</a></li>
```

```
...
</ul>
</nav>
```

```
nav .active { /* elementi klase active unutar navigacije */
  background-color: #555;
}
```

Друго, додајмо ефекат промене боје линкова изнад којих се миш тренутно налази.

```
nav ul li:hover { /* stavke iznad kojih je pokazivač miša */
  background-color: #555; /* tamno siva boja */
}
```

Када смо стилизовали навигационе линкове, можемо прећи на стилизовање целог заглавља. Желимо да поставимо танак оквир око заглавља средње нијансе плаве боје и желимо да ширина целог заглавља буде 900 пиксела, а висина 110 пиксела. Пошто желимо да оквир буде укључен у ове димензије поставићемо `box-sizing: border-box`. Испод заглавља желимо мало дебљу сиву линију, што можемо постићи постављањем дебљег, доњег оквира. На крају, боја позадине целог заглавља треба да буде бела.

```
header { /* zaglavlje */
  box-sizing: border-box; /* dimenzije uključuju i okvire */
  width: 900px; /* širina zaglavlja */
  height: 110px; /* visina zaglavlja */
  border: 1px solid #c0e0f0; /* tanak, plavi okvir */
  border-bottom: 5px solid #ccc; /* donji okvir deblji, sivi */
  background-color: white; /* bela pozadina */
}
```

Стилизујмо и главни наслов који се налази у заглављу. Поред постављања боје текста на тамно сиву и постављања свих спољних маргина на вредност 10 пиксела, најзначајнији ефекат биће постављање малог логотипа уз сам наслов. Иако је то могло бити урађено и у HTML-у коришћењем елемента `img`, одабрали смо да то урадимо помоћу CSS-а, да бисмо вам ту могућност приказали. Сliku `gimnazija.png` ћемо поставити у позадину наслова (`background-image: url("gimnazija.png")`), величину ћемо јој поставити тако да заузме целу висину наслова (`background-size: contain`) и подесићемо да се прикаже само једном, уместо да се понавља више пута иза наслова (`background-repeat: no-repeat`).

```
header h1 {
  margin: 10px;
  padding-left: 50px;
  background-image: url("gimnazija.png");
  background-repeat: no-repeat;
  background-size: contain;
```



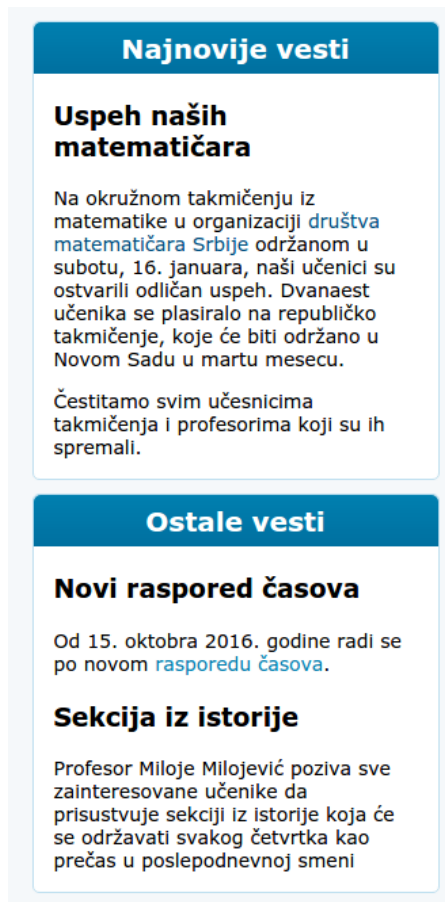
```
color: #555;  
}
```

Један занимљив ефекат који желимо да постигнемо је да заглавље буде стално фиксиран на врху странице. То ћемо постићи постављањем фиксног позиционирање (`position: fixed`) при чему ће горња позиција бити постављена на нулу (`top: 0`), чиме постижемо да оно буде залепљен за врх прозора). Пошто се постављањем фиксног позиционирања за наслов оно измешта из нормалног тока, телу ћемо поставити горњу маргину да се садржај у њему не би преклопио са заглављем.

```
header {  
  position: fixed;  
  top: 0;  
  ...  
}
```

```
body {  
  ...  
  margin-top: 140px;  
}
```

Пређимо сада на стилизовање дела са вестима који смо поставили у десни део сајта.



Свакој секцији постављамо белу боју позадине (`background-color: white`), танак, заобљен оквир (`border: 1px solid #c0e0f0; border-radius: 5px;`) и доњу маргину од 10 пиксела (`margin-bottom: 10px`).

```
aside section {
  border: 1px solid #c0e0f0;
  border-radius: 5px;
  margin-bottom: 10px;
  background-color: white;
}
```

Наслове секција (елементи `h2`) ћемо стилизовати тако да буду центрирани (`text-align: center`), исписани белим словима (`color: white`), величине 20 пиксела (`font-size: 20px`). Демонстрације ради, уместо једнобојне тамно-плаве позадине, користићемо прелаз (градијент) између две нијансе боје (`background: linear-gradient(#0080b0, #0070a0)`). Поред подешавања унутрашње и спољашње маргине (`padding: 7px; margin: 0px;`) потребно је још да регулишемо заобљеност елемента. Желимо да наслови буду заобљени у горњим угловима, а незаобљени у доњим, при

чему заобљеност треба да одговара заобљености самих секција (којима је постављен и оквир). Уместо да понављамо вредност полупречника заобљености, боље је само навести да се његова вредност наслеђује (*border-top-left-radius: inherit; border-top-right-radius: inherit;*).

```
aside section h2 {
  border-top-left-radius: inherit;
  border-top-right-radius: inherit;
  margin: 0px;
  padding: 7px;
  text-align: center;
  font-size: 20px;
  background: linear-gradient(#0080b0, #0070a0);
  color: white;
}
```

Чланцима и пасусима подешавамо само величину слова и маргине.

```
aside article {
  margin: 15px;
}

aside article p {
  font-size: 14px;
}
```

Пређимо сада на стилизовање потписа стране.

```
footer {
  clear: both;
  margin-top: 10px;
  padding: 2px;
  text-align: center;
  font-size: 10px;
  color: white;
  background-color: #555;
}
```

Сада прелазимо на стилизовање централног дела странице. Желимо да узастопне чланке визуелно јасно раздвојимо један од другог и стога сваком од њих постављамо доњу унутрашњу маргину (*padding-bottom: 15px*) и доњи оквир испрекиданом линијом (*1px dashed #bfe0ec*).

```
main article {
  padding-bottom: 15px;
  border-bottom: 1px dashed #bfe0ec;
}
```

Стилизујмо сада табелу која се налази у централном делу странице. Желимо да постигнемо ефекат приказа на слици.

	Učenci	
	Devojčice	Dečaci
1. razred	120	125
2. razred	118	123
3. razred	125	119
4. razred	119	120

Фиксирамо ширину табеле на 400 пиксела (`width: 400px`), центрирамо је постаљањем аутоматских маргина (`margin: auto`), колабирамо све оквире (`border-collapse: collapse`) и постављамо сивкасту сенку иза целе табеле (`box-shadow: 5px 5px 2px #888`).

```
table {
  width: 400px;
  margin: auto;
  border-collapse: collapse;
  box-shadow: 5px 5px 2px #888;
}
```

Ћелијама (и обичним, представљеним елементима `td` и насловним, представљеним елементима `th`) постављамо танак плавкасти оквир (`border: 1px solid #bfe0ec`), леве и десне унутрашње маргине (`padding-left: 20px; padding-right: 20px;`) и постављамо централно поравнавање текста (`text-align: center`).

```
table td, table th {
  border: 1px solid #bfe0ec;
  padding-left: 20px;
  padding-right: 20px;
  text-align: center;
}
```

Насловним ћелијама постављамо тамно-плаву боју позадине (`background-color: #0080B0`) и белу боју текста (`color: white`).

```
table th {
  background-color: #0080B0;
  color: white;
}
```

Да би се подаци у табели могли лакше прочитати, употребићемо тзв. ефекат зебра-пруга и свим парним врстама у табели мало променимо боју позадине (`background-color: #f5f8fa`). Парне врсте можемо селектовати помоћу селектора `tr:nth-`

`child(even).`

```
table tr:nth-child(even) {  
    background-color: #f5f8fa;  
}
```

На крају, опет у циљу лакше читљивости података у табели, поставимо и ефекат промене боје врсте табеле када се миш налази изнад ње.

```
table tr:hover {  
    color: white;  
    background-color: #555;  
}
```

На крају, стилизујмо и галерију сличица на дну стране (елемент `div` коме смо идентификатор `id` поставили на вредност `gallery`). Пошто желимо да слике центрирамо вертикално, желимо да се елемент `div` приказује у облику ћелије табеле (`display: table-cell`), јер тако можемо постављати и хоризонтално и вертикално поравнавање. Постављамо хоризонтално поравнавање (`text-align: center`), а свакој слици посебно постављамо вертикално поравнавање (`vertical-align: middle`). Уз то, свакој слици постављамо ширину од 30% ширине елемента `div` (`width: 30%`).

```
#gallery {  
    text-align: center;  
    display: table-cell;  
}  
  
#gallery img {  
    vertical-align: middle;  
    width: 30%;  
}
```

Након комплетног стилизовања, сајт изгледа као што је то приказно на следећој слици.



## Škola vrednih đaka



Nema sramotnijeg zanata od dangube, besposlice i lenjosti.  
*Dositej Obradović*

DOBRODOŠLI NA SAJT NAŠE GIMNAZIJE. MI SE PONOSIMO SVOJIM VREDNIM ĐACIMA!

## Broj učenika po razredima

	Učenici	
	Devojčice	Dečaci
1. razred	120	125
2. razred	118	123
3. razred	125	119
4. razred	119	120

## Upis u novu školsku godinu

Na upis je potrebno doneti sledeće dokumente:

1. **Svedočanstvo** iz prethodnog razreda,
2. **Izvod** iz matične knjige rođenih,
3. Đačku **knjižicu**.

## Kontakt

Gimnazija „Dositej Obradović“,  
Bulevar oslobođenja 38,  
34000 Kragujevac  
Telefon: 034/123-456, e-mail: sekretarijat@gimnazija.rs



## Najnovije vesti

### Uspeh naših matematičara

Na okružnom takmičenju iz matematike u organizaciji društva matematičara Srbije održanom u subotu, 16. januara, naši učenici su ostvarili odličan uspeh. Dvanaest učenika se plasiralo na republičko takmičenje, koje će biti održano u Novom Sadu u martu mesecu.

ČESTITAMO SVIM UČESNICIMA TAKMIČENJA I PROFESORIMA KOJI SU IH SPREMALI!

## Ostale vesti

### Novi raspored časova

Od 15. oktobra 2016. godine radi se po novom [rasporedu časova](#).

### Sekcija iz istorije

Profesor Miloje Milojević poziva sve zainteresovane učenike da prisustvuje sekciji iz istorije koja će se održavati svakog četvrtka kao prečaz u poslepodnevnoj smeni

***Пишања и заџаи за вежбу***

1.