

HTML5 i CSS3

Prilagodljiv web dizajn

Novi Sad, Novembar 2017.



HTML5 i CSS3

Original Copyright © 2017. Smart d.o.o.

Sva prava zadržana. Ni jedan deo ove knjige ne može biti reprodukovano, snimljen ili emitovan na bilo koji način: elektronski, mehanički, fotokopiranjem, ili drugim vidom, bez pisane dozvole izdavača.

Informacije korišćene u ovoj knjizi nisu pod patentnom zaštitom. U pripremi ove knjige su učinjeni svi naponi da se ne pojave greške. Izdavač i autori ne preuzimaju bilo kakvu odgovornost za eventualne greške i omaške, kao ni za njihove posledice.

Microsoft, Word, Excel, Internet Explorer, Outlook, Windows 8.1 su registrovani zaštitni znaci ili samo zaštitni znaci firme Microsoft Corporation u USA i/ili drugim zemljama.

Izdavač	Smart d.o.o., Trg mladenaca 5, 21000 Novi Sad
	Tel/fax: + 381 21 47 28 200
	www.smart.rs

Sadržaj:

Dan 1	5
Početak rada sa HTML5, CSS3 i prilagodljivim dizajnom	5
Zašto su važni pametni telefoni (a stari IE nije)?	5
Alatke za testiranje.....	6
Prelomna tačka.....	6
Primeri prilagodljivog web dizajna	7
Medija upiti – podržavanje različitih prikaza.....	10
Sintaksa medija upita	10
Isčitavanje upita	12
Viewport	12
Redosled prikazivanja sadržaja u prilagodljivom dizajnu	14
Naš prvi prilagodljiv dizajn	14
DAN 2.....	21
Web tipografija.....	21
Prvi način – standard.....	21
Drugi način - @import.....	21
Treći način - download.....	22
Pseudo klase.....	22
CSS - The :first-child Selector.....	23
CSS - The :last-child Selector.....	25
CSS3 :nth-child() Selector	26
Senčenje teksta pomoću CSS-a 3	29
Senke sa leve i gornje strane.....	30
Onemogućavanje senke u tekstu	30

Senke okvira.....	31
Spajanje parametara CSS-A 3 – dugme THESE SHOULD HAVE WON.....	32
CSS3 prelazi.....	32
Parametri prelaza	33
2D transformacije	35
Šta možemo da transformišemo?	35
Dan 3.....	37
Animacije pomoću CSS-a 3	37
Primer 1	37
Tablični prikaz parametara animacije	39
Skraćeni zapis parametara animacije	39
Forme u HTML 5.....	41

Dan 1

Početak rada sa HTML5, CSS3 i prilagodljivim dizajnom

Donedavno su web sajtovi kreirani sa fiksnom širinom, kao što je 960 piksela i očekivalo se da će krajnji korisnici imati prilično isto iskustvo prilikom pregleda. Ova fiksna širina nije bila dovoljna za ekrane laptopa i korisnici sa monitorima visoke rezolucije imali su veliek margine sa obe strane ekrana.

Međutim, sada postoje pametni telefoni. „Appleov“ iPhone prvi je stvarno ponudio pretraživanje na telonu i mnogi drugi proizvođači sada idu njegovim stopama

Pored toga, osetan je porast korisnika koji za pregled sadržaja radije koriste uređaje sa malim ekranima (na primer. tablete i netbookove) nego velike ekrane. Nespoma je činjenica da sve većom brzinom raste broj ljudi koji koriste male uređaje sa malim ekranom za pregled Interneta. ali je isto tako, sa druge strane, uobičajena i pojava upotrebe monitora od 27 i 30 inča. Sada je razlika između najmanjih i najvećih ekrana za pretraživanje Weba veća nego ikada ranije.

Srećom, postoji rešenje za ovu raznolikost pretraživača i uređaja kojima se prikazi uvek šire. Prilagodljiv web dizajn koji je kreiran pomoću HTML-a 5 i CSS-a 3 omogućava da web sajt funkcioniše na više različitih uređaja i ekrana.

Zašto su važni pametni telefoni (a stari IE nije)?

Iako bi trebalo da se statistika koristi samo kao grubi vodič, interesantno je napomenuti da je za 12 meseci, od jula 2010. do jula 2011. godine, globalna upotreba mobilnih pretraživača porasla sa 2,86 na 7,02 procenata.

Ista statistika pokazuje da je upotreba Internet Explorera smanjena sa 8,79 na 3.42 procenata. Čak je i upotreba Internet Explorera 7 opala za 5.45 procenata do jula 2011. godine. Ako od vas klijenti često traže da napravite sajt da funkcmme u Internet Exploreru 6 i 7, odgovorite im da bi možda trebalo da se koncentrišu na nešto drugo.

Broj ljudi koji koriste uređaje sa malim ekranima za pretraživanje weba je u stalnom porastu, a internet pretraživači ovih uređaja su dizajnirani da rukuju postojećim web sajtovima bez ikakvih

problema. To je omogućeno sužavanjem standardnog web sajta da se prilagodi području prikaza uređaja. Korisnik tada uveličava područje sadržaja za koji je zainteresovan. Međutim, veoma je dosadno i frustrirajuće da konstantno uveličavate i smanjujete područje stranice da biste mogli da pročitate sadržaj. Sigurno je da mi možemo bolje!

Alatke za testiranje

Alatke za testiranje nam omogućavaju da menjamo visinu i širinu aktuelnog prikaza u prozoru pretraživača.

Važno je da razumete da prikaz i veličina ekrana nisu isto. Prikaz se odnosi na područje sadržaja unutar prozora pretraživača, isključujući linije sa alatkama, kartice i tako dalje, a veličina ekrana na fizičko područje prikaza na uređaju.

Alatke za testiranje možete preuzeti u zavisnosti od pretraživača koji koristite na web adresama:

Internet Explorer

<http://www.microsoft.com/download/en/details.aspx?id=18359>

Preuzeti liniju sa alatkama Microsoft Internet Explorer Developer Toolbar.

Mozilla Firefox – preuzeti alatku **Firesizer** na adresi

<http://addons.mozilla.org/en-US/firefox/addon/firesizer>

Google Chrome – preuzeti alatku **Windows Resizer** na adresi

<http://chrome.google.com/webstore>

Prelomna tačka

Uz razumevanje prozora za prikaz, potrebno je da imate dobro razumevanje šta je prelomna tačka. Prelomna tačka u odgovornom dizajnu je širina na kojoj veb sajt menja raspored na osnovu deklaracije medija upita (o njima ćemo tek pričati). Tipično, sajt će biti izgrađen za rad sa najmanje dve, ali obično tri različite kontrolne tačke u cilju posebne vrste uređaja.

Najčešće korišćeni Breakpoints su:

1. Ektra mali uređaji, na primer, telefoni (<768px)
2. Mali uređaji, na primer, tablete (≥768px i <992px)
3. Srednji uređaji, na primer, desktop računari (≥992px and <1200px)
4. Veliki uređaji, na primer, desktop računare (≥1200px)

Osim kontrolne tačke, važno je zapamtiti da medijski upiti reaguju na širini prozora. To je razlog zašto možete jednostavno da promenite veličinu prozora pretraživača i testirate svoje tačke prekida.

Primeri prilagodljivog web dizajna

Otvorimo stranicu <http://2011.dconstruct.org/>. U širokom prikazu (recimo više od 1350 piksela) sajt izgleda kao na sledećoj slici:



Posebno obratite pažnju na grid od devet slika. Dok smanjujete širinu prikaza (na manje od oko 960 piksela), vidite li šta se dešava? Grid od tri reda po tri slike, postaće grid od tri reda sa po dve slike i jednog reda sa tri slike na dnu, kao na sledećoj slici.



Ako još više smanjimo širinu prikaza (na manje od 720 piksela) naići ćemo na još jednu tačku tačku u dizajnu; linkovi u zaglavlju će se promeniti i uključivaće slike koje obezbeđuju bolje ciljno područje za navigaciju na uređajima sa ekranima osetljivim na dodir:



Ako smanjimo prikaz na širinu manju od 480 piksela, grid sa slikama će se ponovo promeniti, prikazujući sada red od po dve slike, zatim tri i četiri slike. Ove slike će nastaviti da se smanjuju dok smanjujemo prikaz na oko 300 piksela.

Na sledećoj slici prikazano je kako stranica izgleda na iPhone-u:



Medija upiti – podržavanje različitih prikaza

Medija upiti predstavljaju ugrađeni modul u CSS3. Bez medija upita ne bismo mogli da prilagodimo izgled našeg sajta na različitim vrstama uređaja preko kojih taj isti sajt gledamo.

Upiti medija mogu da se koriste za proveru mnogih stvari, kao što su:

- širina i visina prozora
- širina i visina uređaja
- orijentacija (landscape ili portrait mod)
- rezolucija

Sintaksa medija upita

Kako izgleda CSS medija upit i kako funkcioniše?

Sintaksa medija upita je sledeća:

```
@media mediatype and (expressions) {  
  CSS-Code;  
}
```

Mediatype može biti:

- screen – koristi se za monitore, tablete, smart telefone
- all – koristi se za sve vrste medija uređaja
- print – koristi se za štampače

Da biste razumeli kako funkcionišu medija upiti, napravićemo jednostavnu HTML stranicu,

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Media Queries in action</title>
    <link href="css/MediaQueryTest.css" rel="stylesheet" type="text/css">
  </head>
  <body>
    <div id="content">
      <h1>Resize the browser to see break points.</h1>
    </div>
  </body>
</html>
```

a potom u CSS dodati kod:

```
body {
  background-color: grey;
}
@media screen and (max-width: 960px) {
  body {
    background-color: red;
  }
}
@media screen and (max-width: 768px) {
  body {
    background-color: orange;
  }
}
@media screen and (max-width: 550px) {
  body {
    background-color: yellow;
  }
}
@media screen and (max-width: 320px) {
  body {
    background-color: green;
  }
}
```

Sada pregledajte fajl u modernom pretraživaču (ako koristite IE, neka bude bar IE 9 verzija) I promenite veličinu prozora pretraživača. Boja pozadine stranice će se menjati, u zavisnosti od veličine aktuelnog prikaza.

Ono o čemu nismo ranije pričali jesu parametri **min-width i max-width**.

Min-width je najmanja širina nekog elementa, dok je max-width najveća širina koju neki element može da ima.

Isčitavanje upita

Na primeru **@media (max-width: 480px) { ... }**, rekli bismo sledeće:
Ako prikaz nije veći od 480px, uradi sledeće....

A **@media (min-width: 481px) and (max-width: 767px) { ... }** bi čitali:
Ako prikaz nije manji od 481px i nije veći od 767px, uradi sledeće...

```
/* Smartphones */  
@media (max-width: 480px) { ... }  
  
/* Smartphones to Tablets */  
@media (min-width: 481px) and (max-width: 767px) { ... }  
  
/* Tablets */  
@media (min-width: 768px) and (max-width: 959px) { ... }  
  
/* Desktop */  
@media (min-width: 960px) and (max-width: 1199px) { ... }  
  
/* Large Display */  
@media (min-width: 1200px) { ... }
```

Viewport

Kada se radi sa medija upitima, potrebno je naznačiti u okviru taga <head> vrstu uređaja koji je primenljiv na stil pomoću media atributa <link> taga.

```
<head>  
    <meta name="viewport" content="width=device-width, initial-scale=1.0">  
</head>
```

- **<meta>** prikaz elementa daje instrukcije pretraživaču o tome kako da kontrolišu dimenzije i proporciju sa stranice.

- **width=device-width** omogućava da širina stranice prati širinu uređaja preko koga se stranica posmatra (koja će se razlikovati u zavisnosti od uređaja).
- **initial-scale=1.0** postavlja početni nivo zumiranja kada se stranica učitava prvi put od strane pretraživača (podešavanje na širinu uređaja znači da naša stranica treba da se renderuje na 100 procenata širine svih podržanih pretraživača).
Vrednost initial-scale bi mogla da bude i 2.0, što bi značilo da sadržaj treba da se skalira na dvostruku veličinu.

Primer – prikaz stranice bez <meta> taga i sa <meta> tagom



Without the viewport meta tag



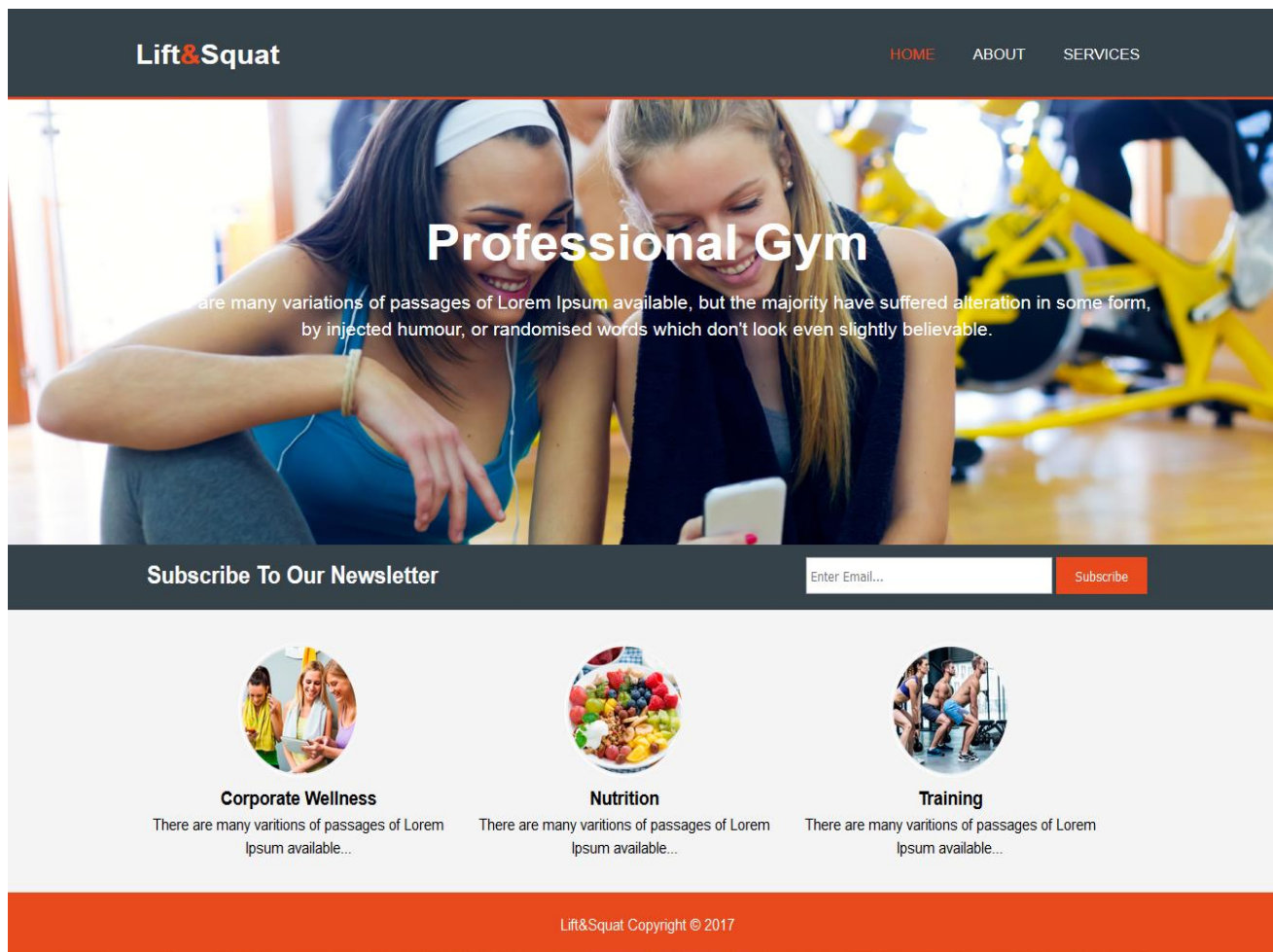
With the viewport meta tag

Redosled prikazivanja sadržaja u prilagodljivom dizajnu

Mi želimo da zadržimo što više funkcija našeg dizajna preko više platformi i prikaza, ali je takođe važno razmotriti redosled u kojem će se elementi pojavljivati. Kada imamo boču traku I glavni sadržaj, moramo podesiti da se glavni sadržaj vidi pre bočne trake, u suprotnom on bi prvo video manje važan sadržaj, a tek onda glavni sadržaj stranice.

Kada je u pitanju navigacija, ona bi trebala na malim uređajima na stoji na vrhu stranice.

Naš prvi prilagodljiv dizajn



HTML:

```
<!DOCTYPE html>
<html>

  <head>
    <meta charset="utf-8">
    <title>Lift&Squat | Wellcome</title>
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" type="text/css" href="css/component.css">
    <link rel="stylesheet" href="stylesheets/style.css" type="text/css">
  </head>

  <body>
    <header>
      <div class="container">
        <div id="logo">
          <h1>Lift<span>&</span>Squat</h1>
        </div>
        <nav>
          <ul>
            <li><a class="active"
href="index.html">Home</a></li>
            <li><a href="about.html">About</a></li>
            <li><a href="services.html">Services</a></li>
          </ul>
        </nav>
      </div>
    </header>

    <section id="showcase">
      <div class="container">
        <h2>Professional Gym</h2>
        <p>There are many variations of passages of Lorem Ipsum
available, but the majority have suffered alteration in some form, by injected humour, or
randomised words which don't look even slightly believable. </p>
      </div>
    </section>

    <section id="newsletter">
      <div class="container">
        <h2>Subscribe To Our Newsletter</h2>
        <form>
          <input type="email" placeholder="Enter Email...">
          <button type="submit"
class="button_1">Subscribe</button>
        </form>
      </div>
    </section>
```

```

        <section id="boxes">
            <div class="container">
                <div class="box">
                    
                    <h3>Corporate Wellness</h3>
                    <p>There are many variations of passages of Lorem Ipsum
available...</p>
                </div>

                <div class="box">
                    
                    <h3>Nutrition</h3>
                    <p>There are many variations of passages of Lorem Ipsum
available...</p>
                </div>

                <div class="box">
                    
                    <h3>Training</h3>
                    <p>There are many variations of passages of Lorem Ipsum
available...</p>
                </div>
            </div>
        </section>

        <footer>
            <p>Lift&Squat Copyright &copy; 2017</p>
        </footer>

    </body>

</html>

```

CSS

```

/*reset*/
* {
    margin: 0;
    padding: 0;
}

body {
    /*font-family: Arial,Helvetica,sans-serif;
font-size: 15px;
line-height: 1.5;*/
font: 15px/1.5 Arial,Helvetica,sans-serif;

```



```

        background-color: #f4f4f4;
    }

    .container {
        width: 80%;
        margin: auto;
        overflow: hidden;
    }

    /*Header*/
    header {
        background-color: #35424a;
        color: #fff;
        padding-top: 30px;
        min-height: 70px;
        border-bottom: #e8491d 3px solid;
    }

    header span {
        color: #e8491d;
    }

    header #logo {
        float: left;
    }

    header nav {
        float: right;
        margin-top: 10px; }

    header ul {
        margin: 0;
        padding: 0;
        list-style-type: none;
    }

    header li {
        float: left;
    }

    header a{
        color: #fff;
        text-decoration: none;
        text-transform: uppercase;
        font-size: 16px;
        padding: 0 20px;
        display: block;
    }

```

```

header li a:hover:not(.active) {
    color: #ccc;
}

header .active {
    color: #e8491d;
}

/*showcase*/
#showcase {
    min-height: 500px;
    background: url(../images/convincer.jpg)no-repeat;
    background-size: cover;
    text-align: center;
    color: #fff;
}

#showcase h2 {
    margin-top: 120px; /*100px kod njega*/
    font-size: 55px;
    margin-bottom: 10px;
}

#showcase p {
    font-size: 20px;
}

/**newsletter***/
#newsletter {
    padding: 15px;
    color: #fff;
    background-color: #35424a;
}

#newsletter h2 {
    float: left;
}

#newsletter form {
    float: right;
    /*margin-top: 15px;*/
}

#newsletter input[type="email"] {
    padding: 4px;
    height: 25px;
    width: 250px;
}

```

```

}

.button_1 {
    height: 35px;
    background-color: #e8491d;
    border: 0;
    padding-left: 20px;
    padding-right: 20px;
    color: #fff;
}

/**boxes**/
#boxes {
    margin-top: 20px;
}

#boxes .box {
    float: left;
    width: 30%;
    padding: 10px;
    text-align: center;
}

#boxes .box img {
    width: 130px;
}

footer {
    padding: 20px;
    margin-top: 20px;
    color: #fff;
    background-color: #e8491d;
    text-align: center;
}

/**Media queries**/
@media screen and (max-width:768px) { /*tablet screen size*/

    header #logo,
    header nav,
    header nav li,
    #newsletter h2,
    #newsletter form,
    #boxes .box {
        float: none;
        text-align: center;
    }
}

```

```

        width: 100%;
    }

    header {
        padding-bottom: 20px;
    }

    #showcase h2 {
        margin-top: 30px;
    }

#newsletter button {
    display: block;
    width: 100%;
}

#newsletter form input[type="email"] {
    width: 100%;
    margin-bottom: 10px;
}
}

@media (max-width:960px){

    #newsletter button {
        display: block;
        width: 100%;
    }
}

```

DAN 2

Web tipografija

Hajde da poboljšamo tipografiju našeg sajta. Postoji veliki broj odličnih izbora za web fontove. Preporučujem vam Google web fontove (<https://fonts.google.com/>), kao i Font Squirrel (www.fontsquirrel.com).

Postoje tri načina za promenu tipografije na našem (bilo kojem) sajtu.

Prvi način – standard

Otvorićemo stranicu <https://fonts.google.com/> i pronaći font Montserrat. Kliknemo na plusić u gornjem desnom uglu, nakon čega se otvara prozor u kome najpre pogledamo opciju CUSTOMIZE. Tu čekiramo vrste slova koje nam trebaju. Mi ćemo čekirati **regular 400** i **bold 700**. Zatim potražimo opciju STANDARD. U <head> naše HTML stranice iskopiramo kod:

```
<link href="https://fonts.googleapis.com/css?family=Montserrat:400,700"
rel="stylesheet">
```

A potom u CSS na mesto gde želimo da se dati font primenjuje iskopiramo kod:

```
font-family: 'Montserrat', sans-serif;
```

Kako da naznačimo da li želimo regular ili bold-ovan font?

To radimo tako što u CSS-u ispod font-family:...; navedemo sledeće:

font-weight: 400; - za regular ili
font-weight: 700; - za bold.

Drugi način - @import

Otvorićemo stranicu <https://fonts.google.com/> i pronaći font Montserrat. Kliknemo na plusić u gornjem desnom uglu, nakon čega se otvara prozor u kome najpre pogledamo opciju CUSTOMIZE. To čekiramo frste slova koje nam trebaju. Mi ćemo čekirati **regular 400** i **bold 700**. Zatim potražimo opciju @IMPORT. U <head> naše HTMLPotom u CSS stranice iskopiramo kod:

```
@import url('https://fonts.googleapis.com/css?family=Montserrat:400,700');
```

A potom (takođe) u CSS na mesto gde želimo da se dati font primenjuje iskopiramo kod:

```
font-family: 'Montserrat', sans-serif;
```

Treći način - download

Otvorićemo stranicu <https://fonts.google.com/> i pronaći font Montserrat. Kliknemo na strelicu u gornjem desnom uglu i tako preuzmемо naš font. Raspakujemo ga u folder fonts koji će biti na istom nivou gde i folder css. Potom ćemo ubaciti font u CSS:

```
@font-face {  
    font-family: 'Montserrat', sans-serif;  
    src: url('../fonts/ Montserrat-Regular.ttf');  
}
```

Ako koristimo i bold font, za njega pišemo novi

```
@font-face {  
    font-family: 'Montserrat', sans-serif;  
    src: url('../fonts/ Montserrat-Bold.ttf');  
}
```

A sa font-weight naglasimo da li je u pitanju Regular ili Bold font.

Ovo ćemo staviti negde na početak CSS-a, a potom ćemo postaviti dati font za navigaciju:

```
nav ul li a {  
    height: 42px;  
    line-height: 42px;  
    margin-right: 2.083333%;           /*20/960*/  
    text-decoration: none;  
    text-transform: uppercase;  
    font-family: 'BebasNeue Regular';  
    font-weight: 400;  
    font-size: 1.6875em;             /*27/16*/  
    color: black;                    }  
}
```

Pseudo klase

Sa pseudo klasama smo se sreli prvi put kada su u pitanju svojstva linkova o kojima smo pričali u prvom delu ovog kursa.

Pa hajde da se podsetimo Anchor pseudo klasa:

```
/* unvisited link */
a:link {
    color: #FF0000;
}

/* visited link */
a:visited {
    color: #00FF00;
}

/* mouse over link */
a:hover {
    color: #FF00FF;
}

/* selected link */
a:active {
    color: #0000FF;
}
```

Ono što možemo da zaključimo, jeste da je sintaksa za pisanje pseudo klase sledeća:

```
selector:pseudo-class {
    property:value;
}
```

Pseudo-klasa se dakle koristi za utvrđivanje specijalnog stanje elementa.

CSS - The :first-child Selector

Selektor :first-child selektuje prvo dete njegovih roditelja. Konkretno na **Primeru 1** to bi izgledalo ovako:

```

<!DOCTYPE html>
<html>
<head>
<style>
p:first-child {
    background-color: yellow;
}
</style>
</head>
<body>

<p>This paragraph is the first child of its parent (body).</p> ←

<h1>Welcome to My Homepage</h1>
<p>This paragraph is not the first child of its parent.</p>

<div>
    <p>This paragraph is the first child of its parent (div).</p> ←
    <p>This paragraph is not the first child of its parent.</p>
</div>

<p><b>Note:</b> For :first-child to work in IE8 and earlier, a DOCTYPE must be
declared.</p>

</body>
</html>

```

This paragraph is the first child of its parent (body).

Welcome to My Homepage

This paragraph is not the first child of its parent.

This paragraph is the first child of its parent (div).

This paragraph is not the first child of its parent.

Note: For :first-child to work in IE8 and earlier, a DOCTYPE must be declared.

U ovom primeru možemo da vidimo da je selektovan prvi paragraf čiji je roditelj <body>, I paragraf čiji je roditelj <div> - označila sam ih strelicama.

Primer 2

Selektuj sve <i> elemente u <p> elementima koji su first-child I promeni im boju u plavo

```
<!DOCTYPE html>
<html>
<head>
<style>
p:first-child i {
    color: blue;
}
</style>
</head>
<body>

<p>I am a <i>strong</i> person. I am a <i>strong</i>
person.</p>
<p>I am a <i>strong</i> person. I am a <i>strong</i>
person.</p>
<p><b>Note:</b> For :first-child to work in IE8 and earlier, a
DOCTYPE must be declared.</p>

</body>
</html>
```

I am a *strong* person. I am a *strong* person.

I am a *strong* person. I am a *strong* person.

Note: For :first-child to work in IE8 and earlier, a DOCTYPE must be declared.

Primer 3 – navigacija za naš sajt

Već smo rekli da želimo da uvedemo neke izmene kada su u pitanju neki od menija koji predstavljaju deo navigacije. Prvi meni WHY? želimo da bude pozicioniran uz levu ivicu, a poslednji meni QUIZ da bude pozicioniran uz desnu ivicu.

Za selekciju prvog menija , koristimo selektor o kome smo pričali, a to je first-child.

Dodaćemo u CSS kod:

```
nav ul li:first-child {
    text-align: left;
}
```

CSS - The :last-child Selector

Last-child selektor za razliku od first-child selektora, selektuje poslednje dete datog roditelja.

Primer 1

Selektuj sve <i> elemente u <p> elementima koji su last-child I promeni im boju u plavo

```
<!DOCTYPE html>
<html>
<head>
<style>
p:last-child i {
    color: blue;
}
</style>
</head>
<body>

<p>I am a <i>strong</i> person. I am a <i>strong</i>
person.</p>
<p>I am a <i>strong</i> person. I am a <i>strong</i>
person.</p>

</body>
</html>
```

I am a *strong* person. I am a *strong* person.

I am a *strong* person. I am a *strong* person.

Primer 2 – Navigacija za naš sajt

Pošto želimo da poslednji meni QUIZ bude pozicioniran uz desnu ivicu, napisaćemo u CSS sledeće:

```
nav ul li:last-child {
    text-align: right;
}
```

CSS3 :nth-child() Selector

Ukoliko bismo želeli da nam treći i peti meni po redu budu npr. crvene boje morali bi na neki način da ih selektujemo i zadamo crvenu boju. To bi mogli da uradimo uz pomoć klasa, ali postoji drugi, mnogo bolji način, a to je uz pomoć :nth-child() Selector-a.

The: :nth-child() selektor selektuje svaki element koji je n-to dete, bez obzira na vrstu, njegovog roditelja.

n može da bude broj, ključna reč ili formula.

Primer 1

Postavite crvenu boju pozadine za svaki <p> elementu koji je drugo dete svog roditelja

```
<!DOCTYPE html>
<html>
<head>
<style>
p:nth-child(2) {
    background: red;
}
</style>
</head>
<body>

<p>The first paragraph.</p>
<p>The second paragraph.</p>
<p>The third paragraph.</p>
<p>The fourth paragraph.</p>

<p><b>Note:</b> Internet Explorer 8 and earlier
versions do not support the :nth-child()
selector.</p>

</body>
</html>
```

The first paragraph.

The second paragraph.

The third paragraph.

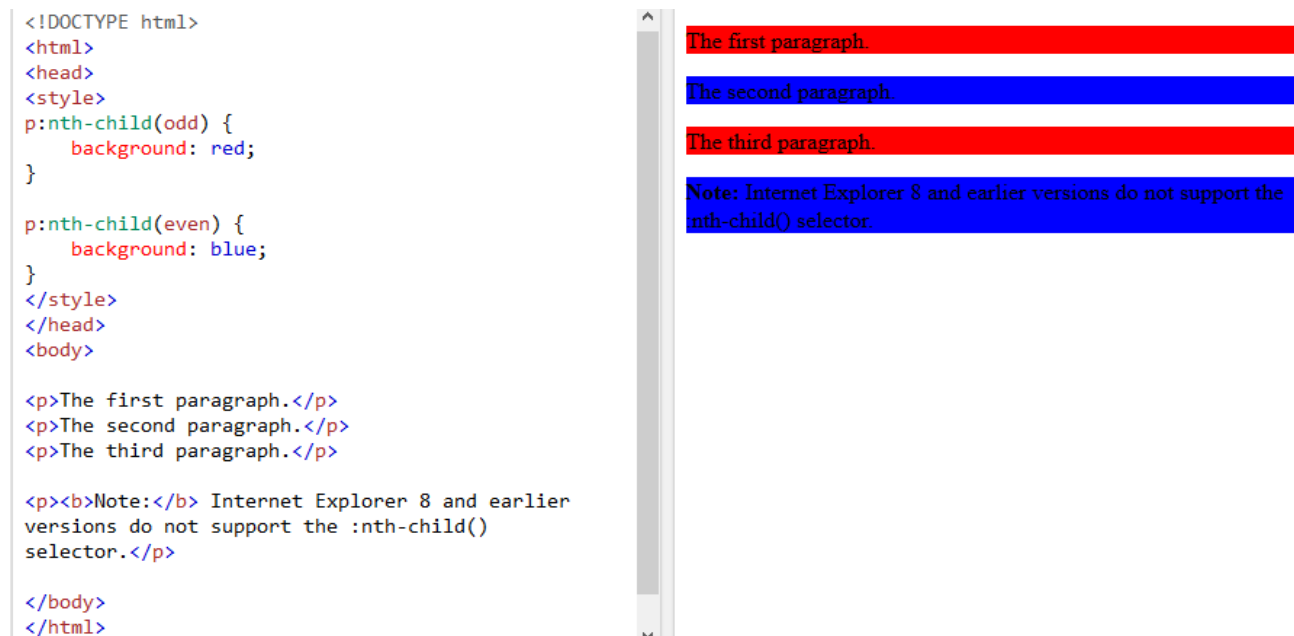
The fourth paragraph.

Note: Internet Explorer 8 and earlier versions do not support the :nth-child() selector.

Primer 2

Parnan (even) i neparan (odd) su ključne reči koje se mogu koristiti da selektujete dete elementa čiji je indeks paran ili neparan (indeks prvog deteta je 1).

Hajde da podesimo dve različite pozadine za neparne i parne p elemente



Hajde sada da sredimo našu navigaciju. Podesićemo da je boja slova neparnih menija (odd) crvene boje

```
nav ul li:nth-child(odd) a{  
  
    color: red;  
  
}
```

Primer 3

Rekli smo da **n** može da bude broj, ključna reč ili formula. Pošto smo prva dva slučaja razmotrili, ostalo nam je još da vidimo kako funkcioniše formula.

:nth-child(3n+1) – započeo bi selektovanjem prve stavke, a zatim bi selektovao svaki treći element

```

<!DOCTYPE html>
<html>
<head>
<style>
p:nth-child(3n+1) {
    background: red;
}
</style>
</head>
<body>

<p>The first paragraph.</p>
<p>The second paragraph.</p>
<p>The third paragraph.</p>
<p>The fourth paragraph.</p>
<p>The fifth paragraph.</p>
<p>The sixth paragraph.</p>
<p>The seventh paragraph.</p>
<p>The eighth paragraph.</p>
<p>The ninth paragraph.</p>

<p><b>Note:</b> Internet Explorer 8 and earlier
versions do not support the :nth-child()
selector.</p>

```

The first paragraph.

The second paragraph.

The third paragraph.

The fourth paragraph.

The fifth paragraph.

The sixth paragraph.

The seventh paragraph.

The eighth paragraph.

The ninth paragraph.

Note: Internet Explorer 8 and earlier versions do not support the :nth-child() selector.

Hajde sada da podesimo uz pomoć formule da svaki drugi naš meni ima plavu boju slova.

nav ul li:nth-child(2n+0) a{

color:blue;

}

Senčenje teksta pomoću CSS-a 3

Senka za tekst se zadaje na ovaj način:

```

.element {
    text-shadow: 1px 1px 1px #ccc;
}

```

Prva vrednost je vrednost senke sa desne strane, **druga** je vrednost senke nadole, **treća** je vrednost zamućenja (udaljenost senke od elementa pre nego što počinje da bleđi), a **poslednje** je boja.

Što se tiče boje, ona može biti zadana kao hex vrednost, HSL(A) ili RGB(A).

Ako boju poseduje alfa kanal, potrebno je da zbog starih pretraživača najpre navedemo hex vrednost boje, a potom rgba, odnosno hsla vrednost.

Primer

text-shadow: 4px 4px 0px #ccc;

text-shadow: 4px 4px 0px hsla(140, 3%, 26%, 0.4);

Hajde da zadamo senku #logo h1, a potom I delu #content h1.

```
#logo h1{
  display: block;
  padding-top: 7.8125%;    /*75/960*/
  color: #0d0c0c;
  text-transform: uppercase;
  font-family: Arial, "Lucida Grande", Verdana, sans-serif;
  font-size: 3em;         /*48/16*/
  padding-left: 1.0416667%; /*dodala kasnije*/
  text-shadow: 2px 2px 2px #000;
}
```

```
#content h1 {
  font-family: Arial, Helvetica, Verdana, sans-serif;
  text-transform: uppercase;
  font-size: 4.3125em;    /*69/16*/
  text-shadow: 4px 4px 4px #000;
}
```

Senke sa leve i gornje strane

Senka sa leve i gornje strane može da se doda korišćenjem negativnih vrednosti.

Primer

text-shadow: -4px -4px 0px #000;

Onemogućavanje senke u tekstu

```
#content h1 {
  font-family: Arial, Helvetica, Verdana, sans-serif;
  text-transform: uppercase;
  font-size: 4.3125em;    /*69/16*/
  text-shadow: none;
}
```

Senke okvira

Sada, kada razumemo kako se zadaju senke za tekst, veoma je jednostavno zadati senku okvira jer je sintaksa ista (horizontalno pomeranje, vertikalno pomeranje, zamućenje i boja).

```
box-shadow: 0px 3px 5px #444;
```

Međutim, senke za okvire nisu dobro podržane u pretraživačima, pa bi bilo dobro kada biste upotrebili prefikse prodavca zbog maksimalne kompatibilnosti – npr:

```
-ms-box-shadow: 0px 3px 5px #444;
```

```
-moz-box-shadow: 0px 3px 5px #444;
```

```
-webkit-box-shadow: 0px 3px 5px #444;
```

```
box-shadow: 0px 3px 5px #444;
```

Spajanje parametara CSS-A 3 – dugme THESE SHOULD HAVE WON

Podesićemo našem dugmetu boju, veličinu slova, dodaćemo senku i okviru i tekstu:

```
#content a {  
    text-decoration: none;  
    font-size: 2.25em;  
    background-color: #b01c20;  
    border-radius: 8px;  
    padding: 30px;  
    color: #fff;  
    float: left;  
    text-transform: uppercase;  
    text-shadow: 0px 1px #000;  
    box-shadow: 5px 5px 5px #000;  
}
```

Potom ćemo **dodati prelazno stanje**:

```
#content a: hover {  
    color: #000;  
    text-shadow: 0px 1px white;  
}
```

Sledeće što ćemo uraditi jeste dodavanje CSS prelaza, o kojima ćemo sad da pričamo.

CSS3 prelazi

Kada stilizujete hiperlinkove, uobičajeno je da kreirate prelazno stanje. To je dobar način da skrenete pažnju korisnicima da je stavka preko koje prelaze zapravo link.

Tradicionalni, kada koristimo samo CSS, prelazna stanja su samo on/off. Postoji jedno stanje koje je standardno i koje se odmah menja na drugačije stanje prilikom prelaska mišem.

Ovo su dva stanja. Prvo je standardno:



A zatim prelazno stanje:



Dodajmo sada malo CSS3 magije u naše prvo pravilo:

```
#content a {  
    ....  
    transitions: all 1s ease 0s;  
}
```

Sada kada prevučemo mišem preko dugmeta, tekst, senka teksta će imati gladak prelaz sa jednog na drugo stanje. Videćete da je prelaz primenjen na originalni element, a ne na prelazno stanje. Dakle, deklaracija prelaza je dodata u element iz kojeg prelazi.

Kako prelazi funkcionišu?

Parametri prelaza

Prelazi mogu da budu deklarirani korišćenjem do četiri parametra ili jedne skraćene deklaracije koja uključuje sva četiri parametra:

- **transition-property:** naziv CSS parametra koji će biti prelazni (kao što su background-color, text-shadow ili all za prelaz svakog mogućeg parametra)
- **transition-duration:** dužina vremena u toku kojeg bi prelaz trebalo da se desi (definišese u sekundama – 3s, 2s...)
- **transition-timing-function:** način kako prelaz menja brzinu u toku trajanja (npr, ease, linear, ease-in, ease-out, ease-in-out ili cubic-bezier).
- **transition-delay:** opcionalna vrednost za određivanje zadržavanja pre početka prelaza.

Mi smo koristili skraćeni zapis, a mogli smo parametre navesti redom:

```
#content a {  
    ....  
    transition-property: all;  
    transition-duration: 1s;  
    transition-timing-function: ease;
```

```
transition-delay: 0s;
```

```
}
```

Ono što se preporučuje kod pisanja prelaza, jeste upotreba prefiksa prodavca:

```
-o-transitions: all 1s ease 0s;  
-ms-transitions: all 1s ease 0s;  
-moz-transitions: all 1s ease 0s;  
-webkit-transitions: all 1s ease 0s;  
transitions: all 1s ease 0s;
```

Kod pisanja kraće verzije, potrebno je pridržavati se redosleda navođenja parametara.

2D transformacije

CSS3 transformacije su potpuno drugačije od CSS prelaza. Razmišljajte o njima ovako:

Prelazi ublažavaju promenu s jednog stanja na drugo, a transformacije transformišu šta će element postati.

Hajde da vidimo na konkretnom primeru jednu transformaciju:

```
#content h1:hover {  
    transform: scale(1.5);  
}
```

Šta možemo da transformišemo?

Postoje sledeće vrste 2D transformacija:

- **scale:** koristi se za skaliranje elemenata (veće ili manje)
- **translate:** pomeranje elemenata na ekranu(gore, dole, levo i desno)
- **rotate:** rotiranje elementa za određenu vrednost (definisano u stepenima)
- **skew:** koristi se za iskošavanje elemenata pomoću X i Y koordinata

Primer za svaku transformaciju:

Scale

```
transform: scale(0.5);
```

Već smo videli ovu transformaciju. Upotrebom vrednosti ispod 1 možemo da skupimo elemente (0.5 će skupiti element na polovinu njegove prvobitne veličine).

Translate

```
transform: translate(40px, 0px);
```

Na ovaj način pomeramo element za vrednost koja je definisana u px ili %. Sintaksa se primenjuje sa leve na desnu stranu (40px), a zatim odozgo nadole (0px).

Pozitivna vrednost pomera element desno ili dole, a negativna vrednost ga pomera levo ili gore.

Rotate

```
transform: rotate(180deg);
```

Transformacija rotate omogućava da rotiramo element. Vrednost u zagradi treba da bude u stepenima.

Skew

Ova transformacija omogućava da element bude iskošen na jednoj ili obe njegove ose.
transform: skew(10deg, 2deg);

Prva navedena vrednost je X osa, a druga vrednost je iskošenje za Y osu.

Dan 3

Animacije pomoću CSS-a 3

Animacija omogućava da element postepeno menja svojstva. Možete promeniti onoliko CSS svojstva koja želite i to koliko god puta poželite.

Da biste koristili CSS3 animacije, morate prvo da navedete neke ključne tačke za animaciju. Ključni kadrovi (keyframes) predstvljaju kakav stil će imati element u određeno vreme.

Primer 1

Ova animacija se svodi na promenu boje pozadine iz crvene u žutu. Animacija će trajati 4 sekunde i za to vreme postepeno će se menjati boja iz crvene u žutu.

```
/* The animation code */
@keyframes example {
  from {background-color: red;}
  to {background-color: yellow;}
}

/* The element to apply the animation to */
div {
  width: 100px;
  height: 100px;
  background-color: red;
  animation-name: example;
  animation-duration: 4s;
}
```

Prvo smo napisali **@keyframes** deklaraciju. Nakon toga, ovoj deklaraciji smo dodali naziv – **example**. Možete da dodelite naziv koji želite. Nakon toga navodimo kada će se stil promeniti pomoću ključnih reči "iz" i "na" (što predstavlja 0% (start) i 100% (kompletan)).

U slučaju da **animation-duration** nije podešeno, animaciju neće imati efekta, jer je default vrednost 0.

Takođe je moguće koristiti procenite. Korišćenjem procenata, možete dodati onoliko promena stila koliko želite.

Primer 2

Sledeći primer će promeniti background-color <div> elementa kada je 25% animacije završeno, 50% završeno i ponovo kada je 100% završeno:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  width: 100px;
  height: 100px;
  background-color: red;
  -webkit-animation-name: example;
  -webkit-animation-duration: 4s;
  animation-name: example;
  animation-duration: 4s;
}

@keyframes example {
  0% {background-color: red;}
  25% {background-color: yellow;}
  50% {background-color: blue;}
  100% {background-color: green;}
}
</style>
</head>
<body>
```

Osim parametara **animation-name**, **animation-duration** imamo i druge važne parametre.

Tablični prikaz parametara animacije

Ime parametra	Objašnjenje / Vrednost koju može da ima
@keyframes	Određuje kod animacije
animation-name	Bilo koje ime
animation-duration	Određuje koliko sekundi ili milisekundi je animaciji potrebno da završi jedan ciklus
animation-timing-function	Promena od jednog skupa CSS pravila do drugog. Moguće vrednosti: linear, easy, easy-in, easy-out, easy-in-out.
animation-iteration-count	Koliko će se puta animacija ponoviti. Može se zadavati brojem (1 je po defaultu) ili ako hoćemo da se beskonačno ponavlja napišemo infinite.
animation-play-state	Određuje da li će animacija da radi ili pauzira. Moguće vrednosti running (default) I paused
animation-delay	Odlaganje početka animacije (u s)

Skraćeni zapis parametara animacije

Parametre jedne animacije bi mogli da napišemo na dva načina. Prvi bi bio npr.

```
animation- name: example;  
animation-duration: 1.5s;  
animation-timing-function: easy-in-out;  
animation-iteration-count: infinite;  
animation-play-state: running;  
animation-delay: 0s;
```

Sve ovo, moglo bi i **skraćeno** da se zapiše:

```
animation: example 1.5s infinite easy-in;
```

Hajde da podesimo da naša navigacija prilikom prelaska mišem ima efekat pulsiranja.

```
nav ul li a:hover {  
    animation: example 1.5s infinite ease-in;  
}
```

```
@keyframes example {  
  from {  
    text-shadow: 0px 0px 4px #000000;  
  }  
  50% {  
    text-shadow: 0 0 40px #000000;  
  }  
  to {  
    text-shadow: 0 0 4px #000000;  
  }  
}
```


Forme u HTML 5

Napravićemo stranicu **quiz.html** I u nju iskopirati kod sa stranice **clips.html**.

Zatim ćemo ponoviti sve o čemu smo pričali a tiče se kreiranja fomulara. Ukratko:

Kreiranje formulara – prosleđivanje podataka

Iako se većina stranica sastoji od teksta, slika i hiperveza, skoro sve Web lokacije imaju bar po jednu stranicu na kojoj se popunjavaju podaci i te informacije se vraćaju operateru WEB lokacije.

Obrazac (eng. form) je skup polja koja se koriste za prikupljanje podataka ili interakciju sa korisnicima. Nakon popunjavanja formulara, posetilac Web stranice vrši potvrdu unetih podataka i ti podaci se zatim šalju serveru na obradu. Forme sadrže različite vrste objekata (polja ili kontrola) kao što su tekst polja, padajuće liste, polja za potvrdu, opciona dugmad, dugmad za slanje podataka, dugmad za brisanje unetih podataka i sl. Kada se serveru proslede podaci iz formulara, vrši se njihova obrada odgovarajućom skriptom ili aplikacijom i kao potvrda uspešnosti popunjenog formulara pojavljuje se naredna strana sa rezultatima obrade.

Prilikom kreiranja stranice sa obrascima, najpre se deifniše koje podatke od korisnika treba prikupiti. Zatim se dodaje forma kojoj pripadaju sva polja i u nju se dodaju pojedinač na polja. Moguće je postaviti pravila unosa podataka u polja (dopuštene vrednosti, format unosa, obavezna polja,...). Za obradu podataka u formama potrebno je da postoji određen skript na serverskoj strani (npr. CGI script –Common Gateway Interface), a na najjednostavniji način obrade je prosleđivanje podataka na određenu e-mail adresu.

Takođe, kontrola popunjenih polja se može izvršiti i na klijentskoj strani, pre slanja na server pomoću JavaScript-a.

Kreiranje obrasca se vrši pomoću elementa form:

```
<form>
```

form elements

```
</form>
```

Dodavanje kontrola u obrazac

Većina kontrola se u obrazac dodaje pomoću taga input. Atributi koji su neophodni da se definišu su:

type – tip polja i

name – jedinstveno ime polja

value – kojim se određuje vrednost podatka

Atribut type uzima jednu od vrednosti: button, checkbox, radio, text, itd.

Na primer:

```
<input type="text" name="ImePolaznika">
```

umeće tekst polje koje ima naziv ImePolaznika. Kada korisnik na Web stranici upiše adresu u ovo tekst polje, i izvrši potvrdu formulara promenljiva ImePolaznika će sadržati tekst koji je korisnik upisao u ovo polje. Takav par atribut – vrednost (ImePolaznika = Marko) se šalje serveru na obradu.

Elementi forme mogu biti:

Input elementi <input>

Input elementi se razlikuju u odnosu na atribut type.

<input type="text">	jednolinijska tekst polja u koja se unose kratki stringovi
<input type="password">	Polje za unos lozinke (umesto unetih karaktera prikazuju se tačkice)
<input type="radio">	Radio dugmad (za odabir jednog od više mogućnosti)
<input type="checkbox">	Polje za štikliranje
<input type="button">	Obično (akciono dugme)
<input type="submit">	Dugme za potvrdu (za slanje sadržaja forme)

Input je oznaka za ulazno polje opšteg tipa, polju se dodeljuje ime atributom name, type je oznaka tipa ulaznog polja, a ostali parametri zavise od tipa ulaznog polja.

Pored atributa **type**, obično se zadaju atributi **name** kojim se zadaje ime podatka i **value** kojim se određuje vrednost podatka. Da bi više **radio**-dugmadi bilo sinhronizovano, potrebno je da sva imaju istu vrednost atributa name. Sadržaj elementa **input** je prazan.

- **select** – (padajuća) lista. Pojedinačne stavke zadaju se elementom option čiji sadržaj predstavlja stavku koja se vidi u listi. Element option se opet karakteriše atributima name i value koji su relevantni prilikom slanja podataka.
- **textarea** – polje za unos teksta u više redova. Sadržaj ovog elementa predstavlja tekst koji se inicijalno upisuje u polje (a koji korisnik može da menja). Atribut name imenuje podatke koji se upisuju, dok se atributima width i height određuju dimenzije polja.
- **label** – predstavlja natpis čiji se sadržaj vidi na ekranu. Atribut for može da sadrži identifikator neke kontrole (obično input) i tada pregledači uspostavljaju određenu vezu između kontrole i natpisa (na primer, kada korisnik klikne na natpis, kontrola se stavlja u žižu tj. fokus).

Text Input – polje za unos teksta

<input type="text">

Primer:

```
<form>
First name:<br>
<input type="text" name="firstname"><br>
Last name:<br>
<input type="text" name="lastname">
</form>
```

First name:

Last name:

Password Input – polje za unos šifre

`<input type="password">`

Primer:

`<form>`

User name:

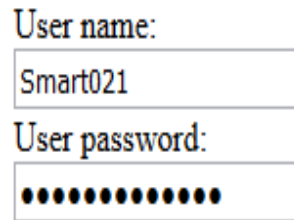
`<input type="text" name="userid">`

`
`

User password:

`<input type="password" name="psw">`

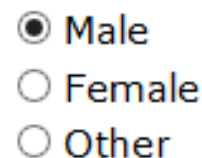
`</form>`



Lozinke type="password" podržavaju isti tip unosa kao i *text* polja s tim što se sadržaj unosa ne vidi na ekranu.

Radio Button Input – radio dugme

`<input type="radio">`



Primer:

`<form>`

`<input type="radio" name="gender" value="male" checked> Male
`

`<input type="radio" name="gender" value="female"> Female
`

`<input type="radio" name="gender" value="other"> Other`

`</form>`

Value – inicijalna vrednost polja (ono što se ispisuje)

Sva radio dugmad u okviru iste grupe treba da imaju isto ime. Samo izabrano radio dugme generiše par *ime/vrednost* koje se prosleđuje. Sa parametrom **CHECKED** se postavlja predefinisana vrednost.

Input Type Checkbox – dugme za štikliranje

`<input type="checkbox">`

Primer:

☐ Ja imam bicikl
☐ Ja imam auto

`<form>`

`<input type="checkbox" name="vehicle1" value="Bike">Ja imam bicikl`

`
`

`<input type="checkbox" name="vehicle2" value="Car">Ja imam auto`

`</form>`

The Submit Button – dugme za slanje sadržaja forme

`<input type="submit">`

Primer:

First name:

Last name:

`<form action="/action_page.php">`

`First name:
`

`<input type="text" name="firstname" value="Mickey">
`

`Last name:
`

`<input type="text" name="lastname" value="Mouse">

`

`<input type="submit" value="Submit">`

`</form>`

Input type reset – dugme za resetovanje na default-ne vrednosti

Primer

First name:

Last name:

`<form action="/action_page.php">`

`First name:
`

```
<input type="text" name="firstname" value="Mickey"><br>
Last name:<br>
<input type="text" name="lastname" value="Mouse"><br><br>
<input type="submit" value="Submit">
<input type="reset">
</form>
```

Ostali Input tipovi

HTML5 je uvela još I tipove:

- email
- color
- date
- datetime-local
- month
- number
- range
- search
- tel
- time
- url
- week

