Sveučilište Jurja Dobrile u Puli
Fakultet za ekonomiju i turizam
"Dr. Mijo Mirković"

Marino Pereša

JavaScript

Završni rad

Pula, 2015.

Sveučilište Jurja Dobrile u Puli
Fakultet za ekonomiju i turizam
"Dr. Mijo Mirković"

Marino Pereša

JavaScript

Završni rad

Matični broj : 0303038155 , redovni student
Smjer: Informatika

Predmet: Elektroničko poslovanje
Mentor: dr.sc.Vanja Bevanda

Pula, kolovoz 2015.

IZJAVA O AKADEMSKOJ ČESTITOSTI

Ja, dolje potpisani _____, kandidat za prvostupnika _____ ovime izjavljujem da je ovaj Završni rad rezultat isključivo mojega vlastitog rada, da se temelji na mojim istraživanjima te da se oslanja na objavljenu literaturu kao što to pokazuju korištene bilješke i bibliografija. Izjavljujem da niti jedan dio Završnog rada nije napisan na nedozvoljen način, odnosno da je prepisan iz kojega necitiranog rada, te da ikoji dio rada krši bilo čija autorska prava. Izjavljujem, također, da nijedan dio rada nije iskorišten za koji drugi rad pri bilo kojoj drugoj visokoškolskoj, znanstvenoj ili radnoj ustanovi.

Student:

U Puli, __.__. 2015. _____

SADRŽAJ

1. Uvod

JavaScript je objektno orijentirani skriptni jezik kojeg je razvio Netscape[1] kako bi omogućio web dizajnerima da dizajniraju interaktivnije web stranice. Uobičajene HTML[2] stranice su pogodne za prikazivanje statičkog sadržaja, poput jednostavnih slika ili teksta. Međutim, danas se stranice sa statičkim sadržajem prilično rijetko viđaju.

Mnoge današnje stranice imaju izbornike, obrasce, slideshow. Budući da JavaScript radi s HTML stranicama, web dizajneri trebaju poznavati HTML kako bi iskoristili puni potencijal tog skriptnog jezika. Iako postoje i drugi jezici koji se mogu koristiti za skriptiranje na webu, u praksi se JavaScript pokazao najpraktičnijim, te je i najrašireniji. Iako se JavaScript uglavnom koristi za interakciju s HTML objektima, može se koristiti i za interakciju s ostalim objektima koji nisu HTML objekti, poput dodataka (eng. *plugins*) u preglednicima, CSS[3] svojstva ili sam preglednik.

JavaScript je jednostavni skriptni jezik posebno napravljen za korištenje u web preglednicima kako bi na web stranicama omogućio dinamični sadržaj. HTML može napraviti statičnu web stranicu, te se nakon što se učita izgled stranice ne mijenja mnogo, sve dok ne kliknemo na link koji će nas odvesti na neku drugu web stranicu. Dodavanjem JavaScript koda omogućava se promjena izgleda dokumenta, od teksta i boje, pa do promjena opcija dostupnih u padajućem izborniku i drugo. JavaScript je *client-side* jezik, što znači da se sve operacije koje se izvode događaju na klijentskom računalu.

Cilj ovog završnog rada je pojasniti skriptne jezike, te ukratko definirati što su HTML i CSS. U radu će se objasniti što je JavaScript i kako je nastao. Također će se prikazati način pisanja i sintaksa u JavaScriptu. Na kraju rada bit će prikazan JavaScript kod na specifičnom slučaju te primjena koda na prethodno napravljenoj stranici.

Ovaj rad se sastoji od deset poglavlja. Nakon uvoda, u prvom poglavlju govori se o skriptnim jezicima.

Drugo poglavlje govori o HTML-u, njegovom razvoju, te razlici HTML5 verzije od prijašnjih verzija. Također se pobliže objašnjava CSS, kao i proces nastanka JavaScripta.

[1] Američka kompanija računalnih usluga
[2] eng. HyperText Markup Language
[3] eng. Cascading Style Sheets

U trećem poglavlju govori se o načinu i pravilima pisanja JavaScript koda.

U četvrtom poglavlju opisuju se varijable i objekti dok su u petom poglavlju prikazani operatori.

Šesto poglavlje prikazuje funkcije, a sedmo poglavlje naredbe za kontrolu tijeka.

U osmom poglavlju primijenjen je JavaScript kod na prije napravljenoj stranici, te je pomoću slika i opisa prikazan način njegovog korištenja.

U devetom poglavlju opisana je jQuery biblioteka, a deseto poglavlje donosi zaključak koji je rezultirao pisanjem ovog rada.

2. Skriptni jezici

Skriptni jezici se u većini slučajeva interpretiraju (iako se mogu i prevoditi), te time omogućuju naizmjenično pisanje i testiranje, liniju po liniju koda, čime se dobiva na dinamičnosti programiranja. No, u usporedbi s Javom, skriptni jezici imaju izrazito malu brzinu izvođenja.

Skriptni jezici dijele se u dvije skupine:

- *„server side“* jezici,
- *„client side“* jezici.

Server side jezici su jezici čiji se kod izvršava na serveru gdje se obrađeni podaci prikazuju na pregledniku koje klijent može vidjeti. Neki od najpoznatijih i najkorištenijih jezika su Python, PHP, ASP, Java, CGI.

Client side jezici su jezici čiji se kod izvršava na računalu klijenta. Ovaj način programiranja je važan dio DHTML[4] koncepta, koji omogućuje stranicama da imaju drugačije i promjenjive sadržaje ovisno o unosu korisnika, uvjetima okoline (kao što je doba dana), ili drugih varijabli. Najpoznatiji jezici su JavaScript, ActionScript, Python i Dart.

Rad sa skriptnim jezicima je izuzetno lagan i fleksibilan. Skripte se ne moraju prevoditi prije izvođenja nego se interpretiraju, a kako postoji ciklička povezanost između pisanja koda i testiranja, povećava se produktivnost programera.

Skriptni jezici imaju sažetiju sintaksu od sintakse tih istih algoritama u objektno-orijentiranim jezicima poput Jave. Proširenjem osnovne funkcionalnosti koda u Javi s kodom napisanim u skripti dobiva se na modularnosti i promjenjivosti koda.

Skriptni jezici su jednostavniji za korištenje od Jave, što omogućuje korisnicima s manjim tehničkim znanjem da prošire osnovnu funkcionalnost nekog kompleksnog koda u Javi koristeći jednostavne skripte koje su sami napisali, te time program prilagođavaju vlastitim potrebama. Skriptni jezici su mnogo sporiji od čistog Java koda, te zbog toga kod kombinacije skriptinih jezika i Jave dolazi do smanjenja brzine izvođenja.

[4] eng. Dynamic HTML

3. HTML, CSS i JavaScript

HTML je zadužen za strukturu web stranica, CSS predstavlja prezentacijski dio, odnosno izgled, a JavaScript je zadužen za interaktivnost na web stranicama. Spomenute tri tehnologije se često nazivaju i tehnologije klijent strane.

3.1. HTML

HTML je engleska kratica za HyperText Markup Language. Koristi se za stvaranje i vizualno predstavljanje, odnosno dizajn web stranice. HTML određuje sadržaj web stranice, ali ne i njegovu funkcionalnost. Oblikovanje sadržaja i stvaranje hiperveza nekog hipertext dokumenta radi se pomoću HTML jezika. Hipertext je posebna vrsta sustava baze podataka u kojem objekti (tekstovi, slike, glazba i slično) mogu biti međusobno povezani. Kada odaberemo neki objekt možemo vidjeti sve ostale objekte koji su s njim povezani. Može se kretati s jednog objekta na drugi iako oni možda imaju vrlo različite forme.

HTML je jednostavan za uporabu i lako se uči, što je jedan od razloga njegove opće prihvaćenosti i popularnosti. Svoju raširenost zahvaljuje jednostavnosti i tome što je od početka bio zamišljen kao besplatan i time dostupan svima. Prikaz hipertekst dokumenta omogućuje web preglednik.[5]

HTML kod osigurava pravilno oblikovanje teksta i slike tako da ih web preglednik može prikazati kako oni trebaju izgledati. HTML jezik opisuje strukturu i semantički sadržaj web dokumenta, gdje je CSS zadužen za promjenu prikaza i oblikovanje tog istog dokumenta. Sadržaj web stranice označen je HTML elementima kao što su <img>, , <title>, <p> i ostali. Cilj HTML-a je napraviti strukturu dokumenta koji će se jednako prikazivati u svim preglednicima.

Html datoteke su zapravo obične tekstualne datoteke, ekstenzija im je .html ili .htm. Povezice unutar HTML dokumenata povezuju dokumente u uređenu hijerarhijsku strukturu i time određuju način na koji posjetitelj doživljava sadržaj stranica.[6]

[5] http://htmlstranica.weebly.com/about.html
[6] ibid

3.1.1. *Razvoj HTML-a*

Prva javno dostupna verzija HTML jezika objavljena je 1993. godine. Ta verzija je bila vrlo ograničena. Razvoj HTML-a nastavljen je tek sljedećom verzijom 2.0, ali ni ona nije postala standard.

Verzija 3.0 objavljuje se u ožujku 1995. godine od strane W3C[7]. HTML 3.0 predlaže mnogo bogatiju verziju HTML-a. Ti nacrti doveli su do usvajanja brojnih novih mogućnosti. Sljedeći korak u razvoju HTML-a značio je prihvaćanje određenih oznaka koje su bile podržane u najpopularnijim web preglednicima. 1996. godine nastao je HTML 3.2.

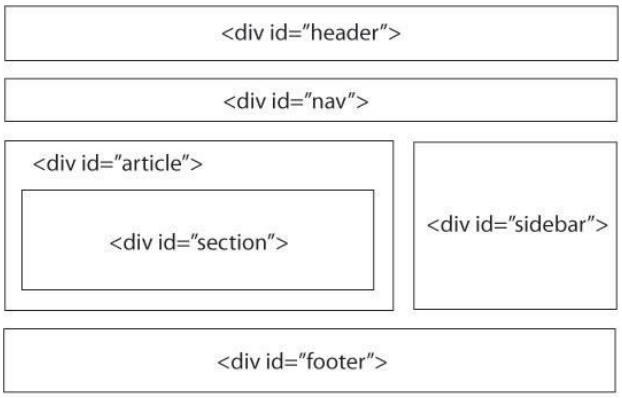
U prosincu 1997. godine predstavljen je HTML4. Konačna verzija ovog jezika je predstavljena u prosincu 1999. godine, gdje su napravili manje promjene u specifikaciji. Ta verzija nazvana je HTML4.01.

Posljednja verzija HTML-a je HTML5 koja se konstantno razvija dodajući brojne nove mogućnosti.

3.1.2. *HTML5*

HTML5 je najnovija verzija HTML standarda. Nudi nove mogućnosti bogatih multimedijskih sadržaja, ali i poboljšanu podršku pri stvaranju web aplikacija koje mogu komunicirati s korisnikom, njegovim lokalnim podacima i poslužiteljem lakše i učinkovitije nego što je prije bilo moguće. U 2011. godini objavljen je HTML5 kojeg su ljudi počeli koristiti i pisati o njemu, no podrška u web preglednicima je bila slaba. Svi glavni web preglednici kao što su Chrome, Safari, Firefox, Opera i drugi u današnje vrijeme nude HTML5 podršku. Stoga se najnovija HTML tehnologija može koristiti u punom potencijalu. HTML5 radi sa CSS3[8] i konstantno se razvija, dodajući sve više i više impresivnih mogućnosti od strane W3C.

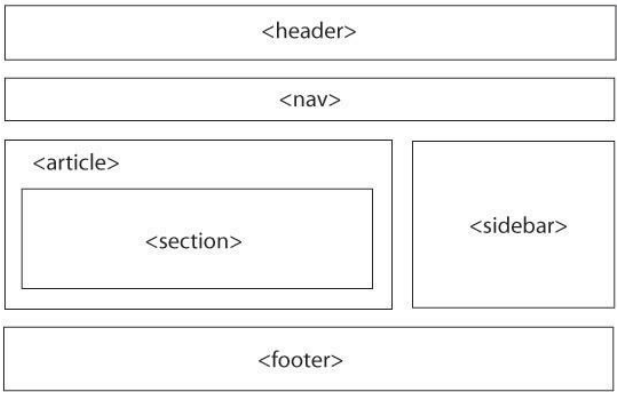
[7] eng. World Wide Web Consortium
[8] nadogradnja CSSa koja donosi novi način dizajniranja



Slika 1. Div struktura

Izvor: http://oshyn.com/general/html5-overview

Na slici 1. i slici 2. prikazana je osnovna razlika u HTML5 strukturi i strukturi prijašnjih HTML standarda. U HTML5 strukturi na slici 2. vidljivo je da su dodani novi elementi. Jedna od razlika HTML5 strukture i Div strukture je u tome što kod HTML5 strukture nema potrebe za korištenjem klasa ili id-a da bi se kasnije definirao stil, već je dovoljno pri definiranju stila koristiti naziv pojedine oznake. S HTML5 strukturom jednostavnije je čitanje i pregledavanje web stranice, gdje se prepoznaje što koja oznaka predstavlja i web dizajnerima pomaže kod izdvajanja dijelova stranice. Još jedna vrlo bitna stvar je što HTML5 struktura daje web tražilicama mogućnost prepoznavanja što koji dio stranice predstavlja pri indeksiranju.



Slika 2. HTML5 struktura

Izvor: http://oshyn.com/general/html5-overview

3.1.2.1. HTML5 elementi

Glavni HTML5 elementi koji se koriste su:

- header
- nav
- hgroup
- article
- section
- sidebar
- footer

HEADER je kao što i samo ime govori prvi element na našoj stranici i u njega obično stavljamo logo i navigaciju. Sam header ne može u sebi imati drugi header niti footer. U starijim verzijama koristio se sljedeći zapis:

```
<div id="header">sadržaj</div>
```

Danas se pak koristi sljedeći zapis:

```
<header>sadržaj</header>
```

NAV element služi za dodavanje navigacije na stranici.

Primjer:

```
<nav id="meni">
    <ul>
    <li><a href="link1">Link1</a></li>
    <li><a href="link2">Link2</a></li>
    <li><a href="link3">Link3</a></li>
    </ul>
</nav>
```

HGROUP spada u tzv. sekcijsku grupu elemenata, a odnosi se na heading elemente h1-h6[9] i njihovo grupiranje.

Primjer:

```
<hgroup>
<h1>Naslov</h1>
<h2>Podnaslov</h2>
</hgroup>
```

SECTION element služi za tematsko grupiranje sadržaja na stranicama. Može sadržavati h1-h6, article, paragrafe i ostalo.

Primjer:

```
<section>
<h1>Naslov</h1>
<h2>Podnaslov</h2>
<article>text</article>
</section>
```

ARTICLE element može biti samostalan ili umjesto oznake section ili unutar njega.Koristise za postavljanje nekog članaka na stranicu.

Primjer:

```
<section>
<article>text</article>
</section>
```

Primjer:

```
<article>text</article>
```

[9] 6 razina naslova u tekstu

ASIDE element je sličan sidebaru[10], ali on to nije. Koristi se za bloquote citate ili za izdvajanje pojedinog sadržaja, a odnosi se na glavni sadržaj. Može sadržavati nav oznaku, citate.

Primjer:

```
<aside>

<nav></nav>

</aside>
```

FIGURE I FIGCAPTION koriste se kod stavljanja slika na stranice. Figcaption se odnosi na tekst ispod slika.

Primjer:

```
<figure>
<img src="myphoto.jpg" />
<figcaption>Photo of me</figcaption>
</figure>
```

FOOTER je samostalan element koji se koristi za dodavanje podnožja stranice gdje se mogu smjestiti kontakt informacije, podaci o zaštiti autorskih prava i slično. Može se nalaziti bilo gdje na stranici, čak i u section oznaci kao njegov footer.

Primjer:

```
<footer>

<small>(c) Copyright www.mojastranica.hr</small>

</footer>
```

Također vrlo korisni i popularni elementi koji se koriste pri dodavanju nekog multimedijskog sadržaja bez potrebe za Flashom[11] su <audio> i <video>.

[10] dio web stranice koji se nalazi uz lijevi i/ili desni rub
[11] multimedijska i softverska platforma za izradu vektorskih grafika, animacija, aplikacija, igra

3.2. CSS

CSS[12] se koristi za oblikovanje izgleda web stranice. Može se koristiti za definiranje stila teksta, veličine slova i druge aspekte web stranice, koji su se nekad mogli definirati samo u HTML-u stranice. Prije CSS-a, gotovo sve prezentacijske osobine HTML dokumenta sadržane su unutar HTML oznaka: sve boje fonta, pozadinski stilovi, poravnanje elemenata, granice i veličine morale su biti izričito opisane, često puta unutar HTML-a. CSS omogućuje dizajnerima da premjeste mnogo tih informacija u drugu datoteku, što rezultira znatno jednostavnijim HTML kodom. Iako se najčešće koristi za promjenu stila web stranica i korisničkih sučelja pisanih u HTML-u i XHTML-u[13], jezik se može primijeniti na bilo koju vrstu XML dokumenta, uključujući i običan XML[14],SVG[15] i XUL[16]. Uz HTML i JavaScript, CSS je temeljna tehnologija koju koristi većina web stranica za stvaranje vizualno zanimljivih web stranica, korisničkih sučelja za web aplikacije, te sučelja za mnoge mobilne aplikacije.

Iako je CSS izvrstan pri stvaranju tekstualnih stilova, koristi se za oblikovanje i drugih aspekata izgleda web stranice. Naprimjer, CSS se može koristiti za definiranje ćelija tablice, stil, debljinu i boju rubova tablice, te oblikovanje slika i drugih objekata. CSS daje web dizajnerima veću kontrolu nad time kako će web stranica izgledati.

CSS pomaže web dizajnerima stvoriti jedinstven izgled na više stranica. Umjesto definiranja stila svake tablice i svakog bloka teksta unutar HTML stranice, obično se koriste stilovi koje je dovoljno definirati samo jednom u CSS dokumentu. Nakon što je stil definiran u CSS dokumentu, može se koristiti za bilo koju stranicu koja ima referencu na CSS datoteku. Osim toga CSS olakšava mijenjanje stila na više stranica odjednom. Naprimjer, ako web dizajner želi povećati veličinu zadanog teksta sa 10pt na 12pt za pedeset stranica. Ako sve stranice imaju referencu na isti CSS dokument, dovoljno je da se promjeni veličinu teksta u tom CSS dokumentu i sve stranice će pokazati povećani tekst.

[12] eng. Cascading Style Sheets
[13] eng. Extensible Hypertext Markup Language
[14] eng. Extensible Markup Language
[15] eng. Scalable Vector Graphics
[16] eng. XML User Interface Language

3.3. JavaScript

JavaScript je programski jezik najčešće korišten za izradu web stranica. Izvorno je razvijen od strane Netscapea kao sredstvo za dodavanje dinamičkih i interaktivnih elemenata na web stranice. Java je utjecala na JavaScript, no sintaksa je sličnija programskom jeziku C. Temelji se na ECMAScriptu, skriptnom jeziku kojeg je razvio Sun Microsystems[17].

JavaScript je client-side skriptni jezik, što znači da se izvorni kod izvršava na računalu web korisnika, tj. posjetitelja dotične web stranice, a ne na web poslužitelju. Odnosno, JavaScript funkcije mogu se izvoditi nakon što se stranica učita, bez komunikacije s poslužiteljem. Naprimjer, funkcija JavaScripta može provjeriti web obrazac prije nego što se pošalje kako bi bila sigurna da su sva tražena polja ispunjena. JavaScript kod može proizvesti poruku o pogrešci prije nego što se bilo kakve informacije proslijede poslužitelju. Kao i server-side skriptni jezici, poput PHP-a, JavaScript kod može biti smješten bilo gdje u HTML web stanici.

JavaScript funkcije se mogu pozvati unutar <script> elemenata ili kada nastane određeni događaj, naprimjer, onClick, on MouseDown, onMouseUp, onKeyDown, onKeyUp, onSubnit i mnogi drugi. Dok se JavaScript standard još uvijek koristi za obavljanje osnovnih client-side funkcija, mnogi web dizajneri preferiraju korištenje JavaScript biblioteka poput jQuery-ja za dodavanje naprednijih i dinamičnijih elemenata na web stranice.

S JavaScriptom se može mnogo toga napraviti. Treba početi s jednostavnijim značajkama poput galerija slika, reagiranja na klikove gumba i mijenjanja izgleda. Nakon što se dobije dovoljno iskustva s jezikom, neki dizajneri će biti u mogućnosti stvarati igre, animirane 2D i 3D grafike, aplikacijske baze podataka i drugo.

Prednosti JavaScripta:

- Brzina – JavaScript je client-side skriptni jezik, koji je veoma brz, zato što se bilo koja ključna funkcija može odmah izvesti, umjesto da se šalje na server i čeka odgovor.
- Jednostavnost – JavaScript se relativno jednostavno uči i primjenjuje.
- Svestranost – može se koristiti u mnogo različitih aplikacija. Za razliku od PHP ili SSI skripta, JavaScript se može umetnuti u bilo koju web stranicu, bez obzira na

[17] Sun Microsystems je bila kompanija koja je prodavala računala, računalne softvere i hardvere, te usluge informacijskih tehnologija.

ekstenziju datoteke. JavaScript se također može koristiti u skriptama napisanim u drugim jezicima kao što su Perl i PHP.

- Dostupnost – JavaScript kod vidljiv je svima.

Nedostaci JavaScripta:

- Sigurnost – budući da se kod izvršava na korisnikovom računalu, u nekim slučajevima to može biti iskorišteno u zlonamjerne svrhe. To je jedan od razloga zbog čega neki ljudi onemoguće JavaScript.

- Oslanjanje na krajnjeg korisnika – JavaScript se ponekad različito tumači u različitim preglednicima. Dok server-side skripte uvijek proizvode isti output, client-side skripte mogu biti ponekad nepredvidive. Ovaj problem se može riješiti na način da se testira skripta u svim glavnim web preglednicima.

3.3.1. *Nastanak i razvoj JavaScripta*

JavaScript jezik je razvio Brendon Eich, suosnivač projekta Mozille, Mozilla Foundation-a i Mozilla Corporation-a. Iako je razvijen pod nazivom Mocha, jezik je službeno nazvan LiveScript kada se prvi put pojavio u beta izdanju Netscape Navigator 2.0 u rujnu 1995. godine. Međutim, preimenovan je u JavaScript nakon što je uključen u Netscape preglednik verzije 2.0B3.

Promjena imena iz LiveScript u JavaScript grubo se podudara s time što je Netscape dodao podršku za Java tehnologiju u svom Netscape Navigator web pregledniku. Konačni izbor imena izazvao je konfuziju, dodajući dojam da je jezik bio dodatak Java programskom jeziku, a izbor naziva bio je okarakteriziran kao marketinški trik kako bi Netscape dao JavaScriptu pečat onoga što je tada bio potpuno novi programski jezik.

Microsoft Windows Script tehnologije uključujući VBScript i JScript objavljene su 1996.godine. JScript, dio Netscape-ove JavaScripta, objavljen je 16. srpnja 1996. godine i bio je uključen u Internet Explorer 3. Internet Explorer 3 također uključuje Microsoftovu prvu podršku za CSS i razne ekstenzije za HTML, ali u svakom slučaju implementacija je bila osjetno drugačija nego što se moglo pronaći u to vrijeme na Netscape Navigatoru. S Internet Explorer 4 izdanjem, Microsoft je predstavio concept Dynamic HTML.

U studenom 1996. godine, Netscape je objavio kako podnosi zahtjev za razmatranje Ecma Internationalu da JavaScript postane industrijski standard, a daljnji rad rezultirao je standardiziranom verzijom pod nazivom ECMAScript.

U lipnju 1997. godine Ecma International objavio je prvo izdanje ECMA-262 specifikacija.

U lipnju 1998. godine, neke promjene su napravljene da se prilagodi ISO/IEC-16262[18] standardu i objavljeno je drugo izdanje. Treće izdanje ECMAScript standarda je objavljeno u prosincu 1999.godine.

Razvoj četvrtog izdanja ECMAScript standarda nikad nije dovršeno. Peto izdanje objavljeno je u prosincu 2009. godine.

Od 2012. godine, svi moderni web preglednici u potpunosti podržavaju ECMAScript 5.1. Stariji preglednici podržavaju barem ECMAScript 3.

17. lipnja 2015. godine objavljeno je šesto veliko izdanje ECMAScripta. Ova verzija službeno je nazvana ECMAScript 2015, ali se obično navodi kao ECMAScript 6 ili ES6.

[18] eng. International Organization for Standardization/International Electrotechnical Commission

4. Sintaksa JavaScripta

Kod pisanja JavaScripta preporučuje se način pisanja prema ASCII[19] standardu, iako se može pisati i prema UNICODE[20] standardu. ASCII za zapis jednog znaka koristi 8 bitova odnosno 1B, dok UNICODE za zapis jednog znaka koristi 16 bitova odnosno 2B. Kod pisanja komentara i nizova znakova nije važno koji se standard koristi.

Prilikom pisanja JavaScript koda treba pripaziti da JavaScript razlikuje velika i mala slova.Veliki broj pogrešaka koje rade početnici upravo su velika i mala slova. To znači da se mora pripaziti u pisanju ključnih riječi, varijabli, funkcija i drugih naziva. Na primjer, ključna riječ se mora pisati *for,* a ne *For* ili *FOR*. Nazivi se pišu malim slovima, u slučaju da naziv sadrži dvije ili više riječi, druga i svaka riječ poslije nje, počinje velikim slovom (npr. getElementById). Isto vrijedi i za varijable. *Broj*, *broj* i *BROJ* tri su različite varijable.

Kraj naredbe u JavaScriptu isti je kao kod programskih jezika C i C++. Nije obavezno staviti točku zarez, ali JavaScript će je ubaciti na kraju svakog retka ako pojedini kod izgleda kao naredba. Na primjer kod:

return

true;

JavaScript će ubaciti znak (;):

return;

true;

4.1. Komentari

Način označavanja komentara u JavaScriptu jednak je kao i u programskim jezicima C i C++. Bilo koji tekst nakon znaka // pa do kraja retka smatra se komentarom. Bilo koji tekst između znakova /* i */ je komentar, te se može protezati u više redaka.

Evo nekoliko ispravnih komentara:

// Običan komentar u jednom retku.

[19] eng. American Standard Code for Information Interchange, sadrži 128 znakova
[20] sadrži 256 znakova

/* Ovo je komentar koji

se proteže na više redaka. */

4.2. Varijable

Pravila kojih se nazivi varijabli i funkcija trebaju pridržavati su sljedeća:

- prvi znak treba biti slovo engleske abecede, znak "$" ili znak podcrtavanja "_",
- mogu sadržavati brojeve i slova engleske abecede,
- velika i mala slova se razlikuju, no uobičajeno je da se pišu malim slovima,
- ključne riječi (for, if, else, class, byte, int…) ne mogu se koristiti u imenu.

Primjeri ispravnih naziva su:

I	g	prva_varijabla	sImeIPrezime
v13	l1	_povlaka	K7

Nazivi varijabli moraju biti različiti od ključnih riječi. U tablici 1. su prikazani nazivi koje treba izbjegavati:

break	Do	If	switch	typeof
case	Else	In	this	var
catch	False	instanceof	throw	void
continue	finally	new	true	while
default	For	null	try	with
delete	function	return		

Tablica1. JavaScript ključne riječi

Izradio autor

4.3. Uključivanje JavaScripta u HTML dokument

JavaScript se može uključiti u HTML dokument na 4 načina:

- pisanjem koda unutar HTML datoteke,
- iz vanjske datoteke uporabom atributa src u oznaci,
- preko obrade događaja navedenog kao HTML atribut (npr. onclick ili onmouseover) ,

- u URL-u koji se koristi posebnim protokolom javascript: .

4.3.1. *Unutar HTML datoteke*

U HTML-u JavaScript kod mora biti umetnut unutar <script> i </script> elemenata. Kada preglednici prikazuju HTML stranicu i pronađu <script> element, oni prelaze u JavaScript način rada i izvršavaju kod unutar elemenata. Nakon što bude gotov s izvršavanjem koda unutar elemenata, preglednik nastavlja prikazivati ostatak HTML-a. JavaScript kod može se smjestiti bilo gdje na stanici, međutim dva najčešća korištenja koda smještaju se:

- u element <head>
- na kraju elementa <body>

HTML dokument može sadržavati više <script> elemenata. Ako postoji više <script> elemenata, oni će se izvršiti redoslijedom kojim su napisani. Ako se neka funkcija ili varijabla definira u jednom <script>elementu, ta ista funkcija ili varijabla može se koristiti i u ostalim elementima bez potrebe za ponovnim definiranjem.

Iako većina preglednika podržava JavaScript, on nije jedini skriptni jezik, te je potrebno definirati vrstu skriptnog jezika u atributu type. Preglednici koji podržavaju definiranu vrstu skriptnog jezika obradit će skriptu, dok će ju ostali zanemariti.

Primjena atributa type:

```
<script type=”text/javascript”>
    // JavaScript program
</script>
```

4.3.2. *Vanjske datoteke s JavaScript kodom*

Skripte se također mogu smjestiti u vanjske datoteke. Vanjske skripte su praktične kada se isti kod koristi na više različitih web stranica ili HTML dokumenata. JavaScript datoteke imaju ekstenziju .js. Za korištenje vanjske datoteke, naziv datoteke se upisuje u src atribut elementa <script>. Naprimjer: <script type=”text/javascript” src=”datoteka.js”></script>

Datoteka s nastavkom .js sadrži samo JavaScript kod, bez <script>elemenata. Vanjsku referencu skripte također se može postaviti u <head>ili<body>elemente. Skripta će se ponašati kao da je smještena upravo tamo gdje su <script> elementi.

Pisanje JavaScripta u vanjskim datotekama ima neke prednosti kao što su:

- odvaja HTML od JavaScript koda
- čini HTML i JavaScript kod preglednijim, te lakšim za čitanje i održavanje
- spremljene JavaScript datoteke mogu ubrzati učitavanje stranice

4.3.3. *Obrada događaja*

JavaScript program izvršava se samo jednom pri učitavanju HTML dokumenta u preglednik.

To ne omogućava interakciju s posjetiteljem web stranice i zbog toga se definira više događaja. Događaji su atributi, kao *onclick, onkeydown* i slično.

Unutar atributa se navodi JavaScript kod kojim se treba izvršiti neki događaj:

<input type=”button” value=”Ispiši” onclick=”windows.alert(‘JavaScript’);” />

U gornjem primjeru nakon što korisnik pritisne dugme *Ispiši* izvršit će se naredba u atributu onclick. Koristi se funkcija *window.alert* koja prikazuje poruku “JavaScript”. Pod vrijednost atributa može se napisati više naredbi, međutim ako ih ima previše preglednije je napisati naredbe u posebnoj datoteci ili u <script> elementima. Nakon toga se poziva navedena funkcija. Na taj način se odvaja HTML sadržaj od JavaScript koda.

4.3.4. *JavaScript u URL-u*

Umjesto da se napiše neki URL, možemo napisati određeni JavaScript kod koji će se aktivirati kada korisnik napravi neku akciju. Razlika je u tome sto URL najčešće započinje sa “http://”, dok se korištenjem JavaScripta započinje s “javascript:”. Naprimjer:

 hyperlink

5. Varijable i objekti

Varijable su mjesta na koja se spremaju vrijednosti podataka. Objekti su varijable koje u sebi mogu sadržavati više vrijednosti podataka.

Tipovi podataka mogu biti:

- brojevi (npr: 16, 3.14, 61*e-7, …),
- logički (Boolean) – (true/false),
- stringovi (npr: “Dobar dan!”),
- null (specijalne ključne riječi s null vrijednošću).

Ako stavimo broj između navodnika, to više nije broj već postaje string.

U stringovima se mogu koristiti specijalni znakovi:

- \f = jedan red dolje (form feed),
- \r = return (carriage return),
- \b = jedno mjesto lijevo (backspace),
- \t = tabulator (tab),
- \n = na početak novog reda (new line character).

5.1. Varijable

Kreiranje varijabli u JavaScriptu nazivamo deklaracijom varijable.

Varijable se deklariraju na način da se napiše ključna riječ „var“, te naziv varijable.

var ime;

Nakon što se varijabla deklarira ona je prazna (nema vrijednosti). Vrijednost se dodaje nakon znaka “=“:

ime=“Marino“;

Također, pri deklaraciji varijable, može joj se dodijeliti vrijednost.

var ime=“Marino“;

Isto tako može se deklarirati više varijabli u jednoj naredbi, gdje se varijable odvajaju zarezom.

var ime="Marino", prezime="Pereša", zanimanje="student";

Varijabla koja nema vrijednost vodi se kao nedefinirana vrijednost. Ako želimo kasnije izračunati ili pomoću forme upisati neku vrijednost, varijabla se pri deklariranju može ostaviti prazna.

5.1.1. *Doseg varijabli*

Doseg djelovanja varijable je područje gdje je varijabla definirana u programu. JavaScript ima dva dosega djelovanja varijable:

- Globalne varijable – imaju globalan doseg, što znači da ju se može koristiti u svakom trenutku, odnosno bilo gdje u JavaScript kodu.
- Lokalne varijable – imaju doseg samo unutar određene funkcije gdje je varijabla definirana. Parametri koji se koriste u funkciji uvijek su lokalnog dosega.

Ako dvije varijable imaju isti naziv, te je jedna globalna, a druga lokalna, veći prioritet ima lokalna varijabla unutar funkcije. Kada se deklarira ime varijable s istim imenom kao što je ime globalne varijable, sadržaj lokalne varijable skriva globalnu, te po izlasku iz funkcije lokalna prestaje djelovati.

5.2. Objekti

U JavaScriptu objekt je samostalan entitet sa svojstvima i tipom. Svojstvo objekta može se objasniti kao varijabla koja je vezana s objektom. Svojstva objekta su u osnovi ista kao i obične JavaScript varijable. Svojstva objekta definiraju karakteristike objekta. Svojstva se u većini slučajeva mogu mijenjati, dodavati, brisati, međutim neka svojstva se mogu samo čitati.

JavaScript ima mnogo predefiniranih objekata. Neki od njih su „Math", „Number", „Date", „String", „Array" i drugi. Osim toga mogu se stvoriti vlastiti objekti. Može se napraviti objekt pomoću "object initializera". "Object initializer" je popis od nula ili više parova naziva i vrijednosti objekata, napisanih unutar vitičastih zagrada ({}).

Pristupanje određenom svojstvu, izvodi se na način da se navede ime objekta, točka i na kraju ime svojstva. Naprimjer, objektu osobe koja ima dva svojstva, ime i zanimanje, pristupa se na sljedeći način:

osoba.ime

osoba.zanimanje

Objekti se stvaraju pozivom konstruktora, odnosno varijable new.

Naprimjer:

```
var objekt = new Object();

var datum = new Date();

var osoba = new Osoba('Marko', 'informatičar');
```

JavaScript metoda je funkcija koja se poziva pomoću objekta:

```
// Funkcija identitet

function identitet(){

        return "Ja sam " + osoba.ime + " i po zanimanju sam " + osoba.zanimanje; }

// Povezivanje s objektom

        osoba.predstaviSe = identitet;
```

6. Operatori

Operatori u JavaScriptu su simboli koji predstavljaju određenu operaciju pomoću koje se spaja jedna ili više varijabli u jedan izraz (aritmetički, logički...). Operatori u JavaScriptu slični su kao i u svim ostalim jezicima, te većina ima potpuno identično značenje. Naprimjer '-' je operator oduzimanja, '==' je operator testiranja jednakosti, '*' je operator množenja i slično.

Izraz je ispravan niz konstanti, varijabli, operatora i izraza koji daje jednu vrijednost, koja može biti broj, niz slova ili logička varijabla.[21]

Postoje dvije vrste izraza:

- izrazi koji sadrže neku vrijednost

Naprimjer a=*16* je izraz kojem se pridjeljuje vrijednost 16 u varijablu a.Takav izraz koristi operator pridruživanja i on sam također ima vrijednost 16.

- izrazi koji sadrže rezultat neke operacije

Naprimjer 2+7 je izraz koji će imati vrijednost 9.

Kod pisanja izraza, te redoslijeda vrijede više-manje uobičajena pravila prednosti operatora.Naprimjer izraz 5 + 3 * 4 rezultira s 17, a ne 32. Kroz sljedeće 4 tablice prikazane su vrste operatora, odnosno na koji način se mogu koristiti.

Operacija	**Objašnjenje**
X + Y	Zbrajanje
X - Y	Oduzimanje
X * Y	Množenje
X / Y	Dijeljenje
X % Y	ostatak dijeljenja X sa Y (modul)
X++	postfiksno povećanje za 1
++X	prefiksno povećanje za 1
X--	postfiksno umanjenje za 1
--X	prefiksno umanjenje za 1

Tablica 2. Aritmetički operatori
Izradio autor

[21]http://www.vitez-studios.com/web-development/hrvatski-tutorijali/javascript-izrazi-i-operatori.html

Operacija	Objašnjenje
X = Y	varijabli X pridružuje se vrijednost Y
X += Y	X = X + Y
X -= Y	X = X – Y
X *= Y	X = X * Y
X /= Y	X = X / Y
X %= Y	X = X % Y

Tablica 3. Operatori pridruživanja

Izradio autor

Operacija	Objašnjenje
X < Y	X manje od Y
X > Y	X veće od Y
X <= Y	X manje ili jednako Y
X >= Y	X veće ili jednako Y
X == Y	X jednako Y
X != Y	X nije jednako Y

Tablica 4. Operatori uspoređivanja

Izradio autor

Operacija	Objašnjenje
X && Y	rezultat je true ako i X i Y imaju istinitu vrijednost (logičko "I")
X || Y	rezultat je true ako ili X ili Y imaju istinitu vrijednost (logičko "ILI")
!X	rezultat je true ako X ima neistinitu vrijednost (logičko "NE")

Tablica 5. Logički operatori

Izradio autor

6.1. Operatori spajanja

Operator + spaja dva niza znakova u jedan novi.

Npr:

a = "2"; b = "2";

c = a + b; // c je 22, a ne 4!!!

Operator spajanja je znak ‘+’ koji se koristi pri zbrajanju, no također se može koristiti pri spajanju stringova ili nekog drugog tipa podataka. Ukoliko je jedan operand broj, a drugi niz

znakova, broj će se također pretvoriti u niz znakova i na taj će se način napraviti spajanje. Međutim, ako koristimo operatore uspoređivanja, situacija je obrnuta. Ako se ponovi prijašnji slučaj gdje postoji jedan broj i drugi niz znakova, niz znakova se pretvara u broj i izvodi se uspoređivanje. Evo nekoliko primjera:

```
3 + 5          // Oba operanda su brojevi, rezultat je 8.

"3" + "5"      // Oba operanda su nizovi znakova, rezultat je "35".

"3" + 5        // Drugi operand se pretvara u niz znakova, rezultat je "35".

16<5           // Oba operanda su brojevi, rezultat je false.

"16" < "5"     // Oba operanda su nizovi znakova, rezultat je true.

"16" <5        // Prvi operand se pretvara u broj, rezultat je false.
```

7. Funkcije

Funkcija je konstrukcija u JavaScriptu koja u sebi sadrži blok naredbi koji se izvršava tek onda kada se u glavnom dijelu programa pozove izvršavanje te funkcije. Funkcije su izrazito korisne kada je potrebno isti blok naredbi izvršiti više puta. Umjesto da se svaki put ponavlja pisanje bloka naredbi, napravi se funkcija i poziva se po potrebi, koliko god puta je potrebno. Pisanjem funkcija skraćuje se pisanje programa, te olakšava čitanje i snalaženje po kodu.

Najčešći način definiranja funkcije je pomoću ključne riječi function:

```
function (, , ...){
// naredbe

}
```

Nije obavezno upisati argumente, ali okrugle zagrade je obavezno napisati. Naredbe koje se izvršavaju pišu se u vitičastim zagradama {...} koje označavaju blok naredbi.

```
// Funkcija koja ništa ne vraća

function ispis(sPoruka) {

    document.write(sPoruka, '');
    }

// Funkcija koja vraća udaljenost između dvije točke

functionfUdaljenost(x1, y1, x2, y2) {

    var fDx, fDy, fRezultat;

    fDx = x2 - x1; fDy = y2 - y1;

    fRezultat = Math.sqrt(fDx * fDx + fDy * fDy);

    return fRezultat;
    }

// Rekurzivna funkcija (poziva samu sebe) koja računa faktorijel

function faktorijel(x){

    if (x <= 1){

        return 1;
```

```
        }
        return x * faktorijel(x-1);
}
// Faktorijel se ovako izračunava: x! = x*(x-1)*(x-2)*...*3*2*1
```

Funkcija se u glavnom kodu poziva na način da se navede njezin naziv, te se u okrugle zagrade upišu svi potrebni argumenti. Ako funkcija nema argumenata, okrugle zagrade se ostave prazne. Ako se pri pozivu funkcije upiše manje argumenata nego što ih sadrži definicija funkcije, svi neupisani argumenti dobivaju vrijednost undefined.

```
// korištenje prijašnje definiranih funkcija za izračunavanje vrijednosti
        ukupno = fUdaljenost(3,2,7,4) + fUdaljenost(1,5,6,2);
        ispis("Rezultat je: " + faktorijel(29)/faktorijel(43));
```

8. Naredbe za kontrolu tijeka

Naredbe za kontrolu tijeka koriste se i pišu na isti način kao u programskim jezicima C i C++.

Naredbe za kontrolu tijeka su:

- if.. [else if].. [else if].. [else]
- switch..case..[case]..[default]
- petlje:
  - for..
  - do..while
  - while..
- break - za prijevremeni izlazak iz petlje ili prekid switch uvjeta
- continue - za prijevremeno završavanje jednog kruga petlje

8.1. Naredbe grananja

Postoje dvije vrste naredbe grananja, a to su *if* naredba i *switch* naredba.

Naredba *if*koristi se naviše načina, ovisno o tome koliko se uvjeta želi upotrijebiti:

- kada se provjerava samo jedan uvjet:

```
var ime = '';
if (ime) {        window.alert('Ime je uneseno'); }
```

Blok naredbi se izvršava ako je uvjet ispunjen.Ovdje je ime prazna varijabla. Samim time uvjet nije ispunjen i blok naredbi se neće izvršiti.

- kada se treba provjeriti više uvjeta ili usporedbi uz alternativu na kraju u slučaju da nije ispunjen niti jedan uvjet:

```
if(n == 1){      // naredbe1      }
else if(n == 2){          // naredbe2      }
else if(n == 3){           // naredbe3     }
else {   // naredbe4      }
```

Ovdje će se provjeravati vrijednost varijable i ovisno o tome koliko iznosi, izvršit će se jedan od blokova naredbi.

Posljednji primjer s višestrukom provjerom prilično je nepregledan pa se umjesto toga koristi *switch* naredba:

```
switch(n){
        case 1: {       // naredbe1
                          break; }
        case 2: {       // naredbe2
                          break; }
        case 3: {       // naredbe3
                          break; }
        default: {      // naredbe4
                          break; }
        }
```

Ovdje se također provjerava vrijednost varijable *n*, i u ovisnosti o vrijednosti varijable izvršit će se jedan od blokova naredbi. Treba pripaziti da se na kraju svakog bloka naredbi napiše naredba *break*. Ako se izostavi naredba *break*, blok naredbi će se normalno izvršiti, međutim na završetku neće izaći već će se izvršiti i sljedeći blok naredbi. Ako ne postoji vrijednost identična izrazu kojeg se provjerava, izvršava se blok naredbi napisan pod default.

8.2. Petlje

Osnovna petlja u JavaScriptu je petlja *while*:

```
var i = 1;
while(i <= 10){
        document.write('Red ' + i + '.');       //naredba će ispisati Red1. ,sljedeću Red2.
        i++;                                    //varijabla se svaki put povećava za 1
}
```

Petlja se izvršava sve dok je uvjet ispunjen. Ako uvjet nije ispunjen na samom početku, petlja se neće niti jednom izvršiti. U ovom primjeru petlja će se prestati izvršavati kada vrijednost varijable postane 10.

Često je potrebno nešto izvršiti barem jednom i u tom slučaju se koristi petlja *do..while.*

```
var i = 1;
do {    document.write('Red ' + i + '.');
        i++;
} while(i <= 10);
```

Petlja *do..while* mora završiti točka-zarezom (;) na kraju zbog toga što završava uvjetom na kraju. U slučaju da uvjet nije ispunjen, petlja će se sigurno barem jednom izvršiti.

Petljafor je najkraći oblik petlje i uključuje tri važna dijela:

- Postavljanje početnih vrijednosti (inicijalizacija).
- Ispitivanje uvjeta – ovaj dio je identičan kao kod while petlje. Uvjet se ispituje svaki put nakon što petlja izvrši zadnju liniju koda.
- Promjena varijable – najčešće se koristi za povećavanje ili smanjivanje varijable za 1.

Ova tri dijela stavljena su u jedan redak i odvojena znakom “;”.

```
for(i = 0; i <= 10; i++) {
    document.write('Red ' + i + '.');
    }
```

9. Primjer JavaScript koda

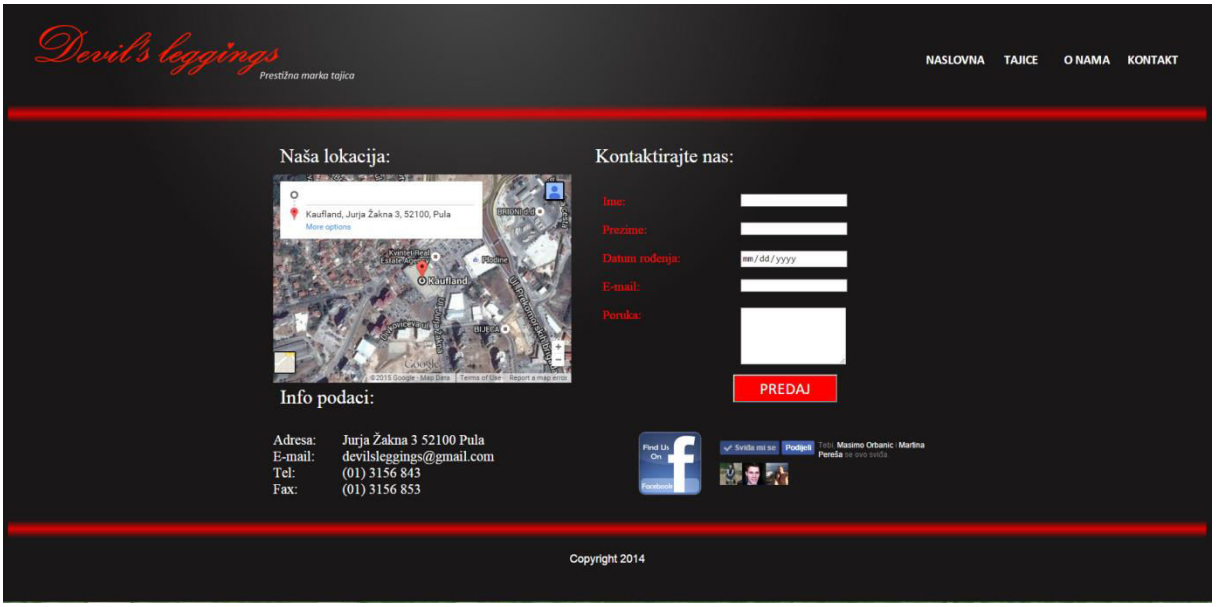
Javascript se najčešće upotrebljava za:

- različite efekte sa slikama (zamjene, rollovere, slide-show,...)
- otvaranje novih prozora Pop-up
- rad s obrascima (automatsko popunjavanje nekih polja, provjera ispravnosti upisanog sadržaja,...)
- promjena svojstava pojedinih elemenata (oznaka) stranice ili prozora
- navigaciju
- sastavni je dio "nove" Ajax tehnologije.

9.1. Primjena JavaScripta na obrascu stranice Devil's leggings

Primijenjen je JavaScript kod na prijašnje napravljenu stranicu za potrebe predmeta Elektroničko poslovanje. Na stranici *Kontakt* upotrijebljen je JavaScript kod kako bi se kroz slike prikazao korišteni kod i što će taj kod napraviti. Specifično je primijenjen JavaScript kodni obrazac za slanje podataka i poruke ili upita ovisno o tome što korisnik želi.

Stranica *Kontakt* izgleda ovako:



Slika 3. Izgled stranice Kontakt

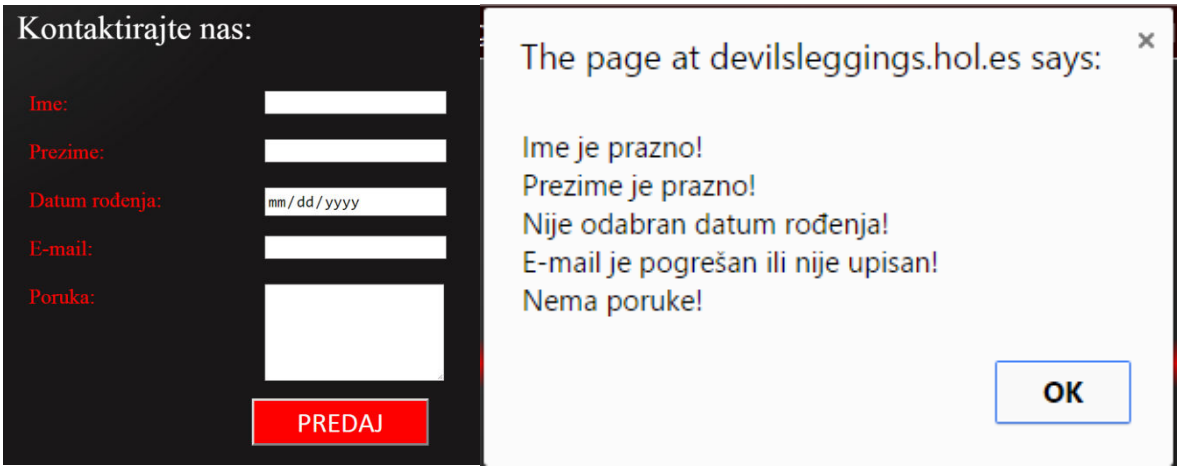
Izradio autor

```
<div class="Informacije">
  <p class="inf">Kontaktirajte nas:</p>
  <div class="nform">
    <form action="mailto:sikirax16@gmail.com" method="GET">

      <label class="ft">Ime:</label>
        <input type="text" name="ime" class="fpod" id="ime" value="" /><br>

      <label class="ft">Prezime:</label>
        <input type="text" name="prezime" class="fpod" id="prezime" value="" /><br>

      <label class="ft">Datum rođenja:</label>
        <input type="date" class="fpod" id="datum" /><br>

      <label class="ft">E-mail:</label>
        <input type="email" class="fpod" id="email" /><br>

      <label class="ft">Poruka:</label>
        <textarea name="poruka" class="fpodatak" id="poruka"></textarea><br>

      <input type="submit" value="PREDAJ" class="batun" onclick="return provjeri();" />
    </form>
</div>
```

Slika 4. Kod za formu (obrazac)

Izradio autor

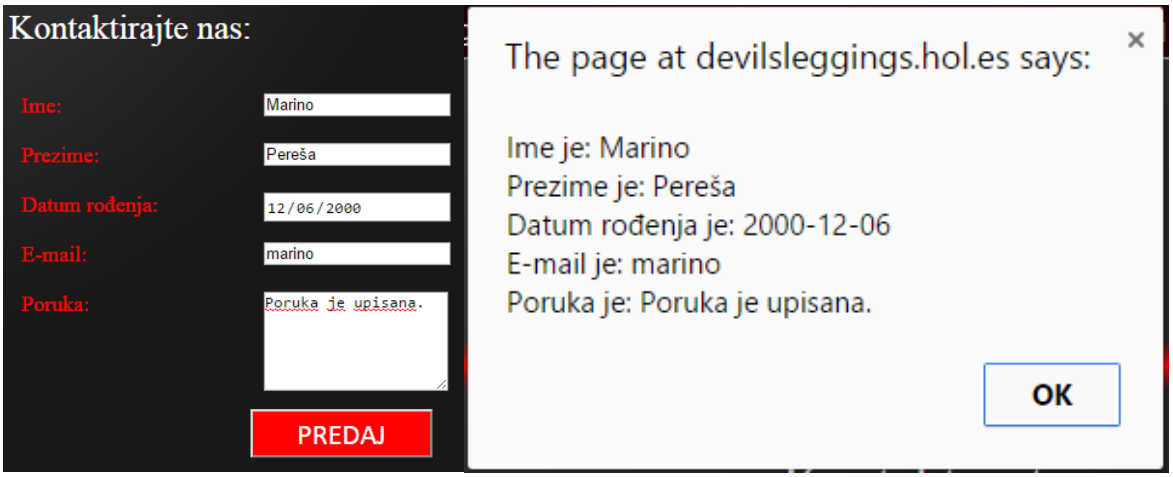
Kroz sljedeće slike bit će prikazano kako JavaScript funkcionira kroz različito upisane podatke.



Slika 5. Prikaz praznog obrasca

Izradio autor

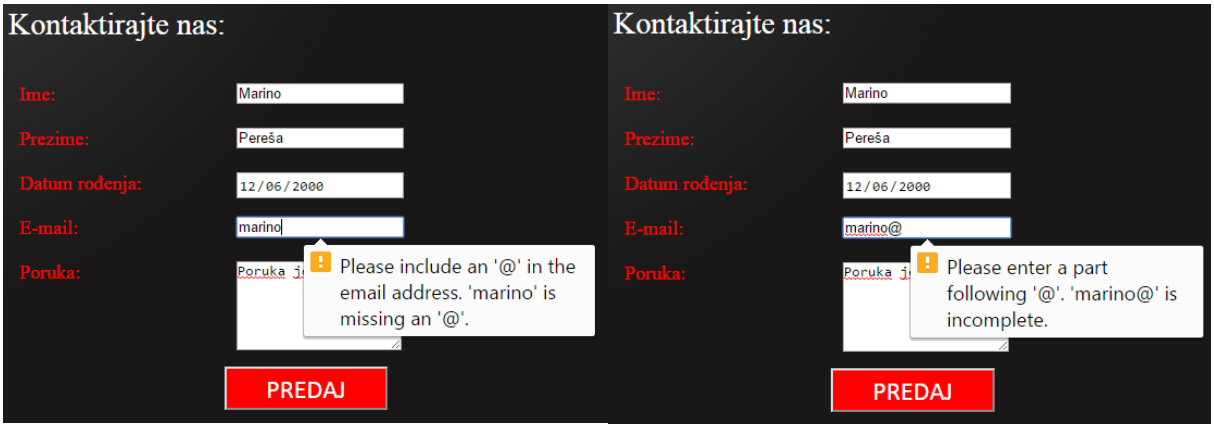
Na slici 5 prikazan je primjer gdje nije upisan niti jedan podatak.



Slika 6. Prikaz s krivo upisanim e-mailom

Izradio autor

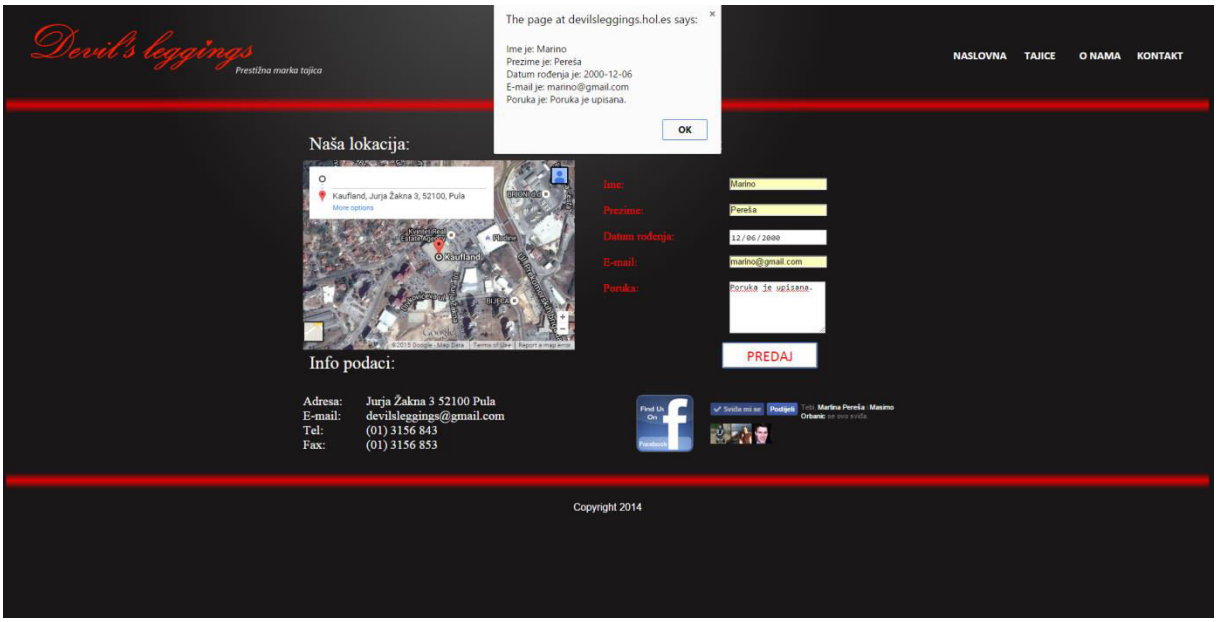
Na sljedećem primjeru sva su polja ispunjena, međutim e-mail nije ispravno upisan što JavaScript kod ne prepoznaje i označava kao da je sve uredu.Međutim, element input tipa e-mail (type=”email”) neće dopustiti krivi unos i upozorit će da e-mail nije ispravno upisan i pokazati što treba dodati pri upisu.



Slika 7. Poruke upozorenja i sintakse

Izradio autor

Na slici 7. prikazane su poruke upozorenja za dva kriva unosa e-maila. Kod lijevog unosa upozorava da je zaboravljen znak @, te nakon što se upiše na drugoj slici upozorava da e-mail nije potpun i da još nešto treba upisati iza toga.



Slika 8. Prikaz stranice Kontakt sa svim podacima i JavaScriptom

Izradio autor

Nakon što se sve ispravno unese, JavaScript kod će prikazati sve ispravne podatke koji su upisani, te nakon što to potvrdi, podaci se prosljeđuju dalje.

```
function provjeri() {
var sIme = '', sPrezime = '', sDatum = '', sEmail = '', sText = '', sPogreska = '', sPoruka = '';
sIme = document.getElementById('ime').value;
sPrezime = document.getElementById('prezime').value;
sDatum = document.getElementById('datum').value;
sEmail = document.getElementById('email').value;
sText = document.getElementById('poruka').value;
if (sIme === '') {
sPogreska += 'Ime je prazno!'; }
else {
sPoruka += 'Ime je: ' + sIme; }
if (sPrezime === '') {
sPogreska += '\nPrezime je prazno!'; }
else {
sPoruka += '\nPrezime je: ' + sPrezime; }
if (!sDatum) {
sPogreska += '\nNije odabran datum rođenja!'; }
else {
sPoruka += '\nDatum rođenja je: ' + sDatum; }
if (sEmail === '') {
sPogreska += '\nE-mail je pogrešan ili nije upisan!'; }
else {
sPoruka += '\nE-mail je: ' + sEmail; }
if (sText === '') {
sPogreska += '\nNema poruke!'; }
else {
sPoruka += '\nPoruka je: ' + sText; }

if (sPogreska === '') {
window.alert(sPoruka);
return true; }
else {
window.alert(sPogreska);
return false; }
}
```

Slika 9. JavaScript kod

Izradio autor

Na slici 9. prikazan je JavaScript kod napisan za izvršavanje na stranici Kontakt web linka devilsleggings.hol.es.

Najprije su u funkciji definirane varijable koje će se koristiti, u ovom slučaju: sIme, sPrezime, sDatum, sEmail, sText, sPogreška koje će sadržavati poruke svih polja koja nisu ispunjena, te sPoruka, koja će se prikazati sa svim podacima nakon što je sve ispravno uneseno.

Nakon što se definiraju varijable, svakoj varijabli se pridodaje vrijednost pomoću funkcije getElementById(‘’) koja preuzima vrijednost preko id-a koji je definiran u formi za pojedino polje.

sIme=document.getElementById(‘ime’).value;

Zatim se svaka pojedina varijabla uspoređuje. Provjerava se je li nešto upisano, te se izvršava if..else uvjet ovisno dali je nešto upisano. Postupak se ponavlja za svaku varijablu.

Na kraju *seif uvjet* provjerava je li nešto upisano u varijablu sPogreška. U slučaju da je sve uredu i da nema ničega u sPogreska, kod će pomoću naredbe *window.alert();* prikazati sve upisane podatke i vratiti vrijednost true.Ako je nešto ipak upisano kroz prijašnje provjere, pomoću naredbe *window.alert()*; prikazat će se što nije upisano pri ispunjavanju obrasca.

10. jQuery

jQuery je open source JavaScript biblioteka. Koriste ju dizajneri da bi si olakšali posao oko dizajniranja interaktivne web stranice. jQuery omogućava manipulaciju HTML-om, animacijom, AJAX-om[22] i slično. Radi u svim modernim browserima kao što su Chrome, Mozilla, Internet explorer...

jQuery je DOM Document Object Model) manipulacijska biblioteka. DOM je stablični prikaz strukture koji sadrži sve elemente web stranice, te jQuery pojednostavljuje sintaksu za traženje, odabir i manipulaciju tim DOM elementima. Na primjer, jQuery se može koristiti za pronalaženje elementa unutar dokumenta sa određenim svojstvom (npr. svi elementi s h1 oznakom), mijenjanje jednog ili više atributa (npr. boja, vidljivost), ili napraviti da reagira na događaj (npr. na klik mišem).

Prednosti jQuery-ja:

- Jednostavnost korištenja - to je glavna prednost jQuery biblioteka, puno je jednostavnija za korištenje u odnosu na JavaScript i druge JavaScript biblioteke. U odnosu na ostale biblioteke osim jednostavne sintakse, ona također zahtjeva puno manje linija koda kako bi se postigla ista značajka.
- Ogromna biblioteka-jQuery pruža veoma više funkcija u odnosu na ostale JavaScript biblioteke.
- Jaka "opensource" zajednica-jQuery ima podršku koja posvećuje svoje vrijeme za razvoj i poboljšavanje jQuery biblioteka. Tako postoje stotine već napisanih dodataka (eng. plugins) dostupnih za preuzimanje, kako bi se ubrzao proces razvoja. Još jedna prednost ovoga je učinkovitost i sigurnost skripte.
- Velika dokumentacija i tutorijali-jQuery web stranice imaju sveobuhvatne dokumentacije i tutorijale, kako bi se i početnicima olakšao rad sa bibliotekama.
- Ajax podrška-jQuery omogućava razvoj Ajax obrasca, Ajax omogućuje sučelje gdje se mogu obaviti radnje na stranicama, bez potrebe da se cijelu stranicu ponovo učita.

[22] eng. Asynchronous JavaScript and XML

Nedostaci jQuery-ja:

- Funkcionalnost je možda ograničena - dok jQuery ima impresivnu biblioteku u smislu količine, ovisno o tome koliko je potrebno prilagoditi web stranicu, funkcionalnost je možda ograničena stoga je ponekad neizbježno korištenje samog JavaScripta.
- jQuery potreba za JavaScript datotekom - jQuery JavaScript datoteka potrebna je za pokretanje jQuery naredba. Iako je veličina takve datoteke relativno mala (25-100KB ovisno o poslužitelju), još uvijek opterećuje klijentsko računalo i čak možda web poslužitelja ako se pohranjuju jQuery skripte na web poslužitelju.

jQuery se može dodati na web stranice na dva načina:

- preuzimanjem jQuery biblioteke sa stranice jQuery.com
- uključivanje jQuery od strane CDN[23], npr. Google

Krajem 2014. godine izašla je stabilna verzija jQuery Mobile. jQuery Mobile je JavaScript biblioteka koju je izradio tim iz jQuerya te se njegov razvoj temelji na razvoju okvira koji je kompatibilan sa raznim tablet računalima i pametnim telefonima. Pomoću jQuery Mobile-a možemo napraviti mobilne aplikacije, na način da se koristi HTML5 i CSS3.

10.1. Osnove jQuery programiranja

U jQuery sintaksi odaberemo HTML elemente i onda izvršavamo neke akcije nad njima. Osnovna sintaksa je $ (selector).action() gdje:

- $ označava da je to jQuery
- (selector) koristimo za pronalaženje i odabir željenih HTML elemenata
- .action() je akcija koja će se izvršiti nad odabranim elementima

jQuery selektori nam omogućavaju da odaberemo i manipuliramo HTML elementima. Svaki jQuery selektor započinje sa znakom $ i zagradama, odnosno $().

Elemente možemo selektirati, tj. odabrati pomoću oznake elementa, id elementa, klase elementa, te razne druge načine. Naprimjer za odabrati element s id=“test“ napišemo $(“#test“).

[23]Content Delivery Network

Događaji koji aktiviraju funkciju ili akciju povezani su sa događajima miša, tipkovnice, forme ili dokumenta, odnosno prozora.

jQuery nudi dizajnerima i programerima da brže i bolje naprave interaktivne web stranice, zbog toga što ne moraju ispočetka pisati svoj vlastiti kod, već iskoriste mogućnosti koje im jQuery pruža. Naprimjer, ako želimo napraviti događaj koji se aktivira kada se klikne na paragraf napišemo $ ("p").click();

U tablici 6 prikazan je popis događaja koji se mogu koristiti.

Miš	**Tipkovnica**	**Forma**	**Dokument/prozor**
Click	keypress	Submit	load
Dblclick	keydown	Change	resize
Mouseenter	keyup	Focus	scroll
Mouseleave		Blur	unload

Tablica 6. Popis događaja

Izradio autor

jQuery nudi višestruk izbor efekata koje možemo primijeniti nad elementima kao što su:

- jQuery Hide/Show
- jQuery Fade
- jQuery Slide
- jQuery Animate
- jQuery stop()
- jQuery Callback
- jQuery Chaining

Također pomoću jQuery-ja možemo čitati, postaviti ili dodati atribute HTML elementima. jQuery nam nudi mogućnost da mijenjamo css svojstva nekog elementa. Naprimjer, ako želimo svim paragrafima u dokumentu promijeniti boju u plavo napišemo $("p").css("color", "blue");

11. Zaključak

Pojavom sve većeg web sadržaja razvila se potreba za interaktivnijim sadržajima koji će korisnicima postati zanimljivi i zabavni, te ih privlačiti da posjete određenu web stranicu. Pojavom skriptnih jezika i JavaScripta ta interaktivnost se ostvarila. Sve više dizajnera web stanica počinje koristiti JavaScript i mnoge bi aktivnosti bez toga postale gotovo neizvedive. Javascript sa svojom interaktivnošću daje novi smisao web sadržajima koji uistinu privlače korisnike. JavaScript je relativno lak za savladavanje, međutim, nužno je imati neko predznanje u HTML-u što čini savladavanje JavaScripta još jednostavnijim.

JavaScript nam najviše koristi pri provjeri je li neki formular pravilno popunjen, navigaciju i ono što je korisnicima interesantno, a to su razni efekti sa slikama kao što je slideshow, zamjena slika pri prelasku mišem i slično.

Međutim, ono što vjerojatno većini smeta, je mogućnost da reklamnim pop-up (iskačućim) prozorima, preusmjere posjetitelja na neku drugu mrežnu lokaciju, protivno volji samog posjetitelja. Mnogi korisnici upravo zbog toga odustanu posjećivati određenu web stranicu.

Nije uvijek dobro osloniti se na JavaScript. Neki korisnici zabrane izvršavanje JavaScripta, te ako je on korišten za nešto bitno, npr. navigaciju po stranici, znači da se korisnici neće moći kretati po stranici. Takve stvari treba predvidjeti i pronaći prikladan nadomjestak.

LITERATURA

1. Gasston, P. *Moderni Web : responzivni Web dizajn uz HTML5, CSS3 i JavaScript*. Zagreb: Dobar plan. 2013.
2. Goodman, D. *JavaScript biblija*. Beograd: Mikro knjiga. 2000.
3. Powers, S. *Naučite JavaScript : (obogatite i oživite Web stranice)*. Zagreb: Dobar plan. 2010.
4. Abrus, L. *Izrada Weba : abeceda za webmastere*. Zagreb: Bug (etc). 2003.
5. Haverbeke, M., 2014, *Eloquent JavaScript.* Dostupno na: http://eloquentjavascript.net/Eloquent_JavaScript.pdf [Pristup: 19.7.2015.]
6. Wagner, G., 2014, *Building Front-End Web Apps with Plain JavaScript*, Dostupno na: https://oxygen.informatik.tu-cottbus.de/webeng/JsFrontendApp/book/ [Pristup: 19.7.2015.]
7. *Building A JavaScript Framework*, Dostupno na: https://s3.amazonaws.com/dailyjs/files/build-a-javascript-framework.pdf [Pristup: 19.7.2015.]
8. Pilgrim, M., *Dive into HTML5*, Dostupno na: http://diveintohtml5.info/index.html[Pristup: 19.7.2015.]
9. Kalafatić, Z., 2012, *SKRIPTNI JEZICI*, Dostupno na: https://www.fer.unizg.hr/_download/repository/Skriptni_jezici_2012.pdf [Pristup: 19.7.2015.]
10. Holjevac, I., 2010, *KLIJENTSKE WEB TEHNOLOGIJE*, Dostupno na: http://www.irenaholjevac.com/dipl/#Povijest-JavaScripta [Pristup: 19.7.2015.]
11. 2003, *HTML – MICROSOFT*, Dostupno na: http://ic.ims.hr/office/frontpage2003/37.html [Pristup: 19.7.2015.]
12. Santiago Valle, 2010, *HTML5 Overview,* Dostupno na: http://oshyn.com/general/html5-overview [Pristup: 19.7.2015.]
13. *Uvod u JavaScript*, Dostupno na: http://dinkosta.tripod.com/sjavascript.html [Pristup: 19.7.2015.]
14. *Web programiranje i primjene JavaScript*, Dostupno na: http://www.mathos.unios.hr/wp/materijali/JS.pdf [Pristup: 19.7.2015.]
15. *Formatiranje JavaScript skripte i njeno uključenje u HTML*, Dostupno na: http://www.znanje.org/knjige/computer/JavaScript/2010/osnove.htm [Pristup: 19.7.2015.]

16. 2013, *Uvod u JavaScript* [Online] Dostupno na: http://www.unidu.hr/datoteke/798izb/03-JavaScript.pdf [Pristup: 19.7.2015.]
17. Dostupno na: http://fly.srk.fer.hr/~keko/javascript/uvod.htm [Pristup: 19.7.2015.]

Popis slika i tablica

Slike:

Tablice:

Sažetak

Cilj ovog rada je navesti i prikazati osnovne pojmove pri izradi web stranica, te detaljnije pojasniti skriptni jezik JavaScript i pokazati njegovu primjenu na specifičnom primjeru.

JavaScript je najčešće korišteni client-side skriptni jezik. To znači da je JavaScript kod napisan unutar HTML stranice. Kada korisnik zatraži HTML stranicu s JavaScriptom, skripta se šalje na preglednik koji će napraviti određenu radnju. JavaScript je razvio Netscape da bi omogućio stvaranje dinamičkog web sadržaja. Iako sadrži mnoge značajke i strukture Java jezika, razvijen je zasebno. JavaScript može komunicirati sa HTML izvornim kodom, omogućavajući web dizajnerima da obogate stranice sa dinamičkim web sadržajem. JavaScript je potvrđen od strane brojnih softverskih kompanija te je otvoren jezik kojeg svatko može koristiti bez kupnje licence.

Ključne riječi: JavaScript, skriptni jezik, HTML, JavaScript jezik, jQuery

Summary

The aim of this work is to go through the basic concepts of designing a website, and a to explain the scripting language JavaScript as well as to present its application to the specific case.

JavaScript is the most commonly used client-side scripting language. This means that JavaScript code is written inside HTML pages. When a user requests an HTML page with JavaScript, the script is sent to the browser, which will do a certain action. JavaScript was developed by Netscape to enable the creation of dynamic web content. Although it contains many of the features and structures of the Java language, it was developed separately. JavaScript can interact with HTML source code, enabling Web designers to enrich the pages with dynamic web content.

JavaScript is confirmed by a number of software companies and it is open language that anyone can use without buying a license.

Keywords: JavaScript, a scripting language, HTML, JavaScript language, jQuery