Programski jezik JavaScript

Seminarski rad u okviru kursa
Metodologija stručnog i naučnog rada
Matematički fakultet

Branislav Jovanović, Svetozar Iković, Marko Vesković
branislavjovanovic052@gmail.com, svetozar.ikovic@gmail.com,
marko.veskovic996@gmail.com

7. maj 2021.

Sažetak

JavaScript je kompaktan i objektno baziran skriptni jezik za razvoj klijent-server veb aplikacija. Programski kod upisuje se direktno u HTML stranicu i omogućuje nam izradu dinamičnih veb stranica. JavaScript je interpretiran jezik - izvršava se bez prethodnog kompajliranja. Ako pratimo trend sve većeg korišćenja veb aplikacija, možemo reći da je JavaScript jezik budućnosti. Tvorac JavaScript-a je Brendan Eich - kasnije jedan od osnivača i direktora Mozilla fondacije.

Sadržaj

1 Uvod

Kroz ovaj rad ćete se upoznati sa jezikom JavaScript, pa se u skladu sa tim u nastavku rada nalaze neke osnove informacije koje će biti od koristi za ljude koji žele da se upoznaju sa istorijom jezika i samim jezikom. Predstavićemo Vam nastanak jezika, namenu, osnovne osobine i specifičnosti, kao i razvojna okruženja koja se koriste i primere kodova.

Cilj nam je da Vam na lak i jednostavan način predstavimo ovaj jezik i da Vas uputimo u opšte informacije kako biste mogli lakše da ga naučite.

2 Nastanak i istorijski razvoj

Nastanak i istorijski razvoj programskog jezika JavaScript usko je vezan za devedesete godine prošlog veka i kompaniju Nestcape Communications (skr. *NCSA*). Naime, 1993. godine nastaje prvi popularni grafički pretraživač Mosaic, objavljen od strane NCSA. Početkom naredne godine većina autora Mosaic-a zapošljava se u tek osnovanoj firmi Mosaic Communications čiji je cilj bio razvoj novog najpopularnijeg pretraživača - Mosaic Netscape. Ideja je uspela i tri četvrtine tržišta je preuzeto. Kako bi izbegli probleme vezane za autorska prava, pretraživač je preimenovan u Netscape Navigator, a kompanija u Netscape Communications. U kompaniji su na vreme shvatili da bi veb trebalo da postane dinamičniji. Kako bi to postigli, pokrenuli su saradnju sa Sun Microsystems kako bi u svoj pretraživač ugradili statički programski jezik Java. Odlučeno je da se kreira programski jezik komplementaran Javi i sa sličnom sintaksom.

Brendan Eich je napisao prototip programskog jezika u maju 1995. godine.

Prilikom prvog objavljivanja beta verzije Netscape Navigator-a 2.0, u septembru 1995. godine, zvaničan naziv programskog jezika bio je **LiveScript** iako je razvijan pod kodnim nazivom Mocha. Svega tri meseca kasnije, u decembru iste godine, jezik je svakako preimenovan u JavaScript. Sam izbor imena izazvao je konfuziju kod korisnika koji su JavaScript povezivali sa Javom, što se ispostavilo kao dobar marketinški trik.

U novembru 1996. godine, Netscape je standardizovao JavaScript, detaljnije o ovome u odeljku Specifičnosti jezika 5.1.

JavaScript je postao jedan od najpopularnijih programskih jezika na vebu, ali ne među profesionalnim programerima. To se značajno menja 2005. godine sa pojavom **AJAX**-a.

AJAX je doprineo razvoju još dinamičnijih veb aplikacija kod kojih se podaci mogu učitavati u pozadini (asinhrono) bez potrebe da se cela stranica ponovo učita. Ono što je bitno - sve je bilo zasnovano na JavaScript-u i time se on vraća u centar pažnje. Rezultat toga je mnoštvo novih okvira i biblioteka kao i povećano korišćenje jezika izvan pretraživača. Drugim rečima, značajno se povećava razvoj JavaScript serverskih platformi [1][7][2].

3 Osnovna namena, svrha i mogućnosti

Na početku je veb bio samo niz informacija u obliku statičkih HTML stranica koje su bile povezane hiperlinkovima. Te informacije su jednostavno prikazivane u okviru pretraživača.

Uvođenjem dinamičnosti stranicama dobijena je mogućnost interakcije između korisnika i tih stranica. Želja tadašnjih administratora veb sajtova bila je da se uštedi prostor na serveru za jednostavne zadatke koji se obavljaju više puta, kao što je recimo validiranje formi (obrazaca).

U to vreme nađena su dva rešenja: JavaScript i Java apleti, s tim što apleti nisu masovno prihvaćeni. Konkretno, od svih skript jezika, JavaScript se vremenom pokazao kao najbolje rešenje.

U današnje vreme, JavaScript više nije toliko usko vezan isključivo za pretraživač. Naime, za pokretanje JavaScript-a uvek se koristi okruženje hosta, a pretraživač je samo jedan od dostupnih hostova. U doba savremenih tehnologija i uređaja, JavaScript može da funkcioniše i na serveru, desktopu, kao i na mobilnim uređajima.

Uz pomoć JavaScript-a možemo napraviti moćne veb aplikacije (aplikacije koje funkcionišu u pretraživaču), mobilne aplikacije, bogate multimedijalne aplikacije kao što su Flash ili Flex i tako dalje.

JavaScript je danas prisutan svuda. Kompanije koje su razvile pretraživače sada se utrkuju u kreiranju najbržih JavaScript virtuelnih mašina čime se otvaraju mogućnosti širenja primene JavaScript-a u oblastima obrade slika, audio i video zapisa, razvoja igara i tako dalje[14].

4 Osnovne osobine i elementi

4.1 Osnovne osobine

4.1.1 Slabo tipiziran jezik

U JavaScript-u možemo reći da promenljive nemaju tipove već da vrednosti imaju tipove. Šta to znači u praksi? To znači da sam jezik razlikuje tipove, ali da programer ne mora unapred zadavati tipove promenljivama, već ih računar automatski prepoznaje.

Takođe, u JavaScript-u je dozvoljeno da ista promenljiva tokom izvršavanja programa dobija vrednosti različitih tipova. Odvajanje memorije za svaku promenljivu se ne vrši pre izvršavanja programa (tj. tokom kompajliranja), već tokom izvršavanja, svaki put kada promenljivoj dodelimo novu vrednost. Suprotno od strogo tipiziranih jezika kod kojih važi da svaka promenljiva unapred mora da ima zadat tip, jezici poput JavaScript-a su slabo tipizirani (eng. *loosely typed*)[8].

4.1.2 Validiranje korisničkog unosa

Kao što smo već spomenuli, JavaScript je veoma koristan kada radimo sa formama. JavaScript ima mogućnosti da proveri korisnički unos u slučaju grešaka, a takođe štedi i vreme [15].

4.1.3 Nezavisan od platforme

Negde se može pronaći i pojam portabilan. Dakle, sav kod koji napišemo možemo koristiti bilo kada i bilo gde odnosno JavaScript aplikacije mogu se pokretati na bilo kom sistemu i pretraživaču bez ikakve promene onoga što je izlazna vrednost aplikacije [15].

4.1.4 Prepoznavanje pregledača i operativnog sistema

Postoje situacije kada bi nam značilo da znamo koji pretraživač i operativni sistem koristi korisnik aplikacije. Recimo da želimo da se na različitim pretraživačima aplikacija ponaša drugačije. JavaScript u tome značajno pomaže jer je sposoban da očita date informacije za programera [15].

4.1.5 Interpretiran jezik

JavaScript je interpretiran jezik. To znači da je JavaScript kod procesuiran liniju po liniju. JavaScript interpreter je ugrađena komponenta u pretraživaču, ali u današnje vreme mnogi JavaScript pokretači (eng. *JavaScript engine*) kao na primer V8 pokretač u Google Chrome pretraživaču koriste *just-in-time*[1] kompilaciju za JavaScript kod [15].

4.2 Osnovni elementi

4.2.1 Tipovi promenljivih

U JavaScript-u, promenljive deklarišemo koristeći neku od tri ključne reči: var, let i const. U početku, postojao je samo var.

Međutim, **ECMAScript2015** je uveo let i const. Ključna reč **const** definiše konstantnu referencu na vrednost, a ne konstantnu vrednost. Ovo znači da ne možemo menjati konstantnu vrednost primitivnog tipa, ali da možemo menjati svojstva unutar objekata.

Razlika između **var** i **let** je u dosegu (eng. *scope*) koji je uveo ECMAScript 2015. Var ima doseg funkcije(eng. *function scope*), dok let ima doseg bloka (eng. *block scope*).

Postoji sedam ugrađenih tipova u JavaScript-u: null, undefined, boolean, number, string, symbol i objekat (eng. *object*) (jedini koji predstavlja vrednosti koje nisu primitivne).

null i **undefined** predstavljaju odsustvo vrednosti: null predstavlja praznu vrednost dok undefined predstavlja vrednost koja nedostaje.

Za logičke vrednosti koristimo **boolean**. Moguće vrednosti su true i false.

Što se **brojeva** tiče, možemo definisati razne vrednosti: celobrojne, razlomljene, negativne, NaN, Infinity.

Niske (eng. *string*) se koriste za reprezentaciju teksta. Mogu se koristiti tri vrste navodnika: jednostruki, dvostruki i nakošeni (tzv. šablon-literali (eng. *template literals*)).

Kao što smo već rekli, **objekat** predstavlja vrednosti koje nisu primitivne. Objekat je jedini mutabilan tip u JavaScript-u. Postoji nekoliko podtipova objekata kao što su nizovi (eng. *Array*), datumi (eng. *Date*) ili funkcije (eng. *Function*) koji se koriste za još specifičnije vrednosti.

Takođe, postoje omotači (eng. *object wrapper*) oko vrednosti koje su tipa boolean, number i string. Za string kao primitivnu vrednost je specifično to da ne postoje nikakve metode za manipulaciju tom vrednošću. Ipak, omotač String ih obezbeđuje [8] [12] [15].

[1] Just-in-time (JIT) je izraz koji se koristi za opisivanje radnje poput kompilacije ili aktiviranja objekta samo u vreme kada to postane potrebno.

4.2.2 Izdizanje promenljivih

Izdizanje promenljivih (eng. *Hoisting*) je JavaScript mehanizam gde se promenljive i deklaracije funkcija premeštaju na vrh svog dosega pre izvršavanja koda.

To znači da bez obzira na to gde su funkcije i promenljive deklarisane, one se premeštaju na vrh svog dosega, bez obzira da li je njihov doseg globalni ili lokalni.

Međutim, napominje se činjenica da mehanizam za izdizanje promenljivih samo izdiže deklaraciju, a dodeljivanje vrednosti ostaje na svom mestu.

Izdizanje promenljivih je veoma retka pojava u programskim jezicima. Pored JavaScript-a, jedan od jezika koji takođe poseduje izdizanje promenljivih je jezik Python [4].

4.2.3 Operatori

Operatori, redosled prednosti jednog u odnosu na drugi i njihova asocijativnost [13].

Operator	Operacija	Redosled	Asocijativnost
++	Inkrement	1	desno-ka-levo
- -	Dekrement	1	desno-ka-levo
-	Negacija	1	desno-ka-levo
!	Ne	1	desno-ka-levo
*,/,%	Množenje, deljenje, deljenje po modulu	2	levo-ka-desno
+,-	Dodavanje, oduzimanje	3	levo-ka-desno
+	Konkatenacija	3	levo-ka-desno
<,<=	Manje, manje-jednako	4	levo-ka-desno
>,>=	Veće, veće-jednako	4	levo-ka-desno
==	Ekvivalentno	5	levo-ka-desno
!=	Različito	5	levo-ka-desno
===	Identitet	5	levo-ka-desno
!==	Ne-identitet	5	levo-ka-desno
&&	Logičko I	6	levo-ka-desno
||	Logičko ILI	6	levo-ka-desno
?:	Ternarni operator	7	desno-ka-levo
=	Dodela	8	desno-ka-levo
+=,-=,*=,/=,%=	Aritmetička dodela	8	desno-ka-levo

5 Specifičnosti jezika

5.1 Standardizacija

U novembru 1996. godine, Netscape je standardizovao JavaScript kako bi ostale kompanije koje razvijaju pretraživače mogle da ga implementiraju u svojim proizvodima. To je dovelo do zvaničnog objavljivanja specifikacije jezika **ECMAScript** koja je objavljena u prvoj verziji **ECMA-262** standarda u junu 1997. godine, čija je najpoznatija implementacija upravo **JavaScript** (pored recimo ActionScript-a i JScripta koji je razvio Microsoft).

Do sada je objavljeno jedanaest izdanja ECMA-262. Najpoznatiji predstavnici su **ECMAScript 5** (skraćeno **ES5**) koji je izbačen 2009. godine i **ECMAScript 6** (skraćeno **ES6**) koji je izbačen 2015 godine, uvode najviše promena u JavaScript jezik [11] [5].

5.1.1 ES5 vs ES6

Pošto je prošlo šest godina između objavljivanja verzija 5 i 6 ECMAScript-e, vidimo dosta velike promene u samom jeziku između ove dve verzije. Neke od najbitnijih su [6]:

- Uvođenje ključnih reči let i const const

```
let x = 13; // globalan opseg
function myFunction() {
    let y = 14; // opseg funkcije
    if(true) {
        let z = 14; // opseg "if" klauze
    }
}
```

Ključna reč let

```
const pi = 3.14; // Menjanje "const" vrednosti vraća greš
ku
pi = 3.141592; // Error: Assignment to constant variable
```

Ključna reč const

- Arrow funkcije

```
const add1 = (a,b) => a+b;
```

Arrow function

- Spread operator (...)

```
function sum(x, y, z) {
    return x + y + z;
}

const numbers = [1, 2, 3];

console.log(sum(...numbers));
// očekivan izlaz 6

console.log(sum.apply(null, numbers));
// očekivan izlaz: 6
```

Spread operator

- Proširena klasa Array

```
let arg = Array.from("ABCDEFGHI"); // ["A", "B", ...]
// ----------------------------------
let languages = ["C#", "JavaScript", "Java","C#"];
let x = languages.indexOf("Java"); // 2
let y = languages.lastIndexOf("C#");  // 3
// ----------------------------------
let numbers = [25, 30, 35, 40, 45];
let rez = numbers.find(n => n % 2 ==0) // 30
```

Nove Array metode

- Podrazumevane vrednosti

```
function calculate(a=30, b=40) {
     console.log(a, b);
}
calculate(10,20); // 10 20
calculate(10); // 10 40
calculate(); // 30 40
```

Podrazumevane vrednosti

5.1.2 ES7 - ES10

Kasnije verzije (7, 8, 9, 10) izdavane su sa razmakom od godinu dana nakon ECMAScript 6, pa samim tim uvode dosta manje značajnih razlika u sam jezik.

- **ES7**
  - Array metod includes()
  - Operator **
- **ES8**
  - Async/await
  - Metode objekta keys(), values(), entries()
  - Objekat Atomics
  - ...
- **ES9**
  - Asinhroni iterator
  - Proširenje regularnih izraza
- **ES10**
  - Array metode flat(), flatMap()
  - String metode trimStart(), trimEnd()
  - Novi primitivni tip BigInt
  - ...

5.2 JSON format

JSON, odnosno JavaScript Object Notation, je tekstualno baziran otvoreni standard dizajniran za razmenu podataka razumljivu ljudima. Ona je izvedena iz JavaScript jezika za predstavljanje jednostavnih struktura podataka.

Napravljen inicijalno da bude reprezentacija JavaScript objekta, dostigao je ogromnu popularnost u serijalizaciji i prenosu strukturiranih podataka preko veb-a i zbog toga postao standard za sebe [9].

```
{
  "firstName": "John",
  "lastName": "Smith",
  "isAlive": true,
  "age": 27,
  "address": {
    "streetAddress": "21 2nd Street",
    "city": "New York",
    "state": "NY",
    "postalCode": "10021-3100"
  },
  "phoneNumbers": [
    {
      "type": "home",
      "number": "212 555-1234"
    },
    {
      "type": "office",
      "number": "646 555-4567"
    }
  ],
  "children": [],
  "spouse": null
}
```

JSON primer

6 Razvojno okruženje

Razvojno okruženje se definiše kao skup procedura i alata za razvoj, testiranje i otklanjanje grešaka u aplikaciji. Integrisano razvojno okruženje (u daljem tekstu IRO, eng. *Integrated development environment, IDE*) je softver za razvoj aplikacija koji kombinuje alate za razvoj aplikacija u jedan grafički korisnički interfejs (eng. *graphical user interface, GUI*). IRO pruža programerima GUI koji služi za pisanje, izgradnju, testiranje i otklanjanje grešaka u aplikaciji. Razvojna okruženja i IRO su namenjeni da olakšaju proces programiranja, dok su većina njihovih karakteristika namenjene da sačuvaju vreme, povećaju efikasnost i organizuju proces rada.

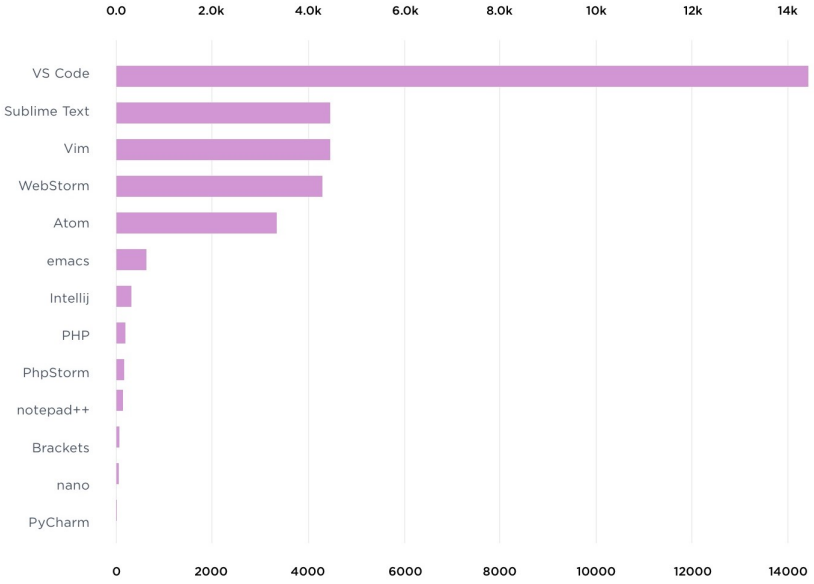
IRO se standardno sastoji od uređivača izvornog koda, alatki za ugrađenu automatizaciju i otklanjivača grešaka.

Uređivač izvornog koda (eng. *Source code editor*) je tekst editor koji pomaže pri pisanju softverskog koda. Njegove glavne karakteristike su isticanje sintakse (eng. *Syntax highlighting*), pametni završetak koda (eng. *auto-completion*) i pronalaženje sintaksičikih ili semantičkih grešaka prilikom izrade koda.

Alatke za ugrađenu automatizaciju (eng. *Local build automation*) su uslužni programi koji jednostavne i ponavljajuće poslove automatizuju, poput kompajliranja izvornog koda u binarni kod i pokretanje automatizovanih testova.

Otklanjivač grešaka (eng. *debugger*) je program koji se koristi za testiranje drugih programa. On ima mogućnost prikazivanja lokacije greške u originalnom kodu kao i postavljanje tački zaustavljanja (eng. *breakpoints*).

U daljem tekstu ćemo navesti neke od najpopularnijih IRO-a, kao i njihove karakteristike. Takođe možemo videti i sliku sa najkorišćenijim IRO-vima 1.



Broj novih korisnika na mesečnom nivou

6.1 Atom

Atom IRO je besplatan i jednostavan uređivač izvornog koda. Napravio ga je GitHub, tako da ima najbolju integraciju sa GitHub-om. Atom ima mogućnosti jednostavnog uređivanja (eng. *editing*) i pretraživanja projekta (eng. *Project browsing*) u jednom prozoru. Glavna namena mu je vođenje velikih komplikovanih projekata koji iskorišćavaju njegove modularne funkcionalnosti. Jedini je IRO koji ima mogućnost zajedničkog rada u realnom vremenu.

Koristi dosta resursa i može da radi sporije čak i na jačim kompjuterima ako paketi nisu uređeni na pogodan način, što ga ne čini pogodnim za sve programere. Pored toga nema podršku za korisnike niti odeljak za pomoć, što ga čini teško pristupačnim.

6.2 Visual Studio / Visual Studio Code

Visual Studio Code je najkorišćeniji JavaScript uređivač. Besplatan je, a iako ga je razvio Microsoft, radi na više platformi. Ima karakteristike poput pametnog završavanja koda sa IntelliSense-om, integraciju sa GitHub-om i mogućnost otklanjanja grešaka (eng. *debug*) direktno iz uređivača koda. Ima dosta mogućnosti za prilagođavanje podešavanja (eng. *customization*). Podržava korišćenje git komandi poput commit, publish, pull, push i rebase zahvaljujući integraciji sa Git-om.

6.3 WebStorm

Jedan od popularnijih IRO-ova za JavaScript je WebStorm. Pruža mogućnosti pametnog završavanja koda, primećivanja greški (eng. *Error detection*) i refaktorisanja (eng. *refactoring*) za jezike JavaScript-a, Node.js, HTML i CSS. Za razliku od drugih integrisanih razvonjih okruženja kompanije JetBrains poput PyCharm-a, WebStorm nema besplatnu verziju (eng. *Community edition*), ali je moguće dobiti edukativnu licencu.

6.4 Sublime

Sublime je moćan uređivač teksta sa odlikama poput istovremenog uređivanja teksta (istovremeno vršenje istih interaktivnih promena na različitim delovima teksta). Iako po definiciji nije IRO često ga nazivaju pseudo IRO. Podržava više platformi i ima veliku mogućnost prilagođavanja podešavanja. Za razliku od Atoma, Sublime nije zahtevan i može se pokrenuti i na slabijim mašinama, ali zbog toga ima manje mogućnosti. Izvorno podržava mnoge programske jezike, a dodatne funkcionalnosti se mogu integrisati pomoću dodatnih komponenti (eng. *plugin*) održavanih pod licencama slobodnog softvera. Neke od dodatnih komponenti koje su bitne za JavaScript razvoj u Sublime-u su DocBlockr, JSFormat, SideBar Enhancement i SublimeLinter.

7 JavaScript svuda

Iako je JavaScript inicijalno nastao za potrebe veb pretraživača, rastom i povećanjem broja korisnika krenuo je da se koristi i šire. JavaScript pokretači su danas ugrađeni u niz različitih softverskih sistema, kako za serverske sisteme i baze podataka, tako i za aplikacije koje nemaju veze sa veb-om. Prva serverska okruženja pisana u JavaScript-u su Netscape LiveWire i Microsoft-ov Internet Information Services, ali ona nisu uzela maha. Tek krajem 2000-ih sa nastankom Node.js-a Javascript je počeo masovno da se koristi za izradu serverskih sistema.

7.1 Node.js

Node.JS (često nazivan samo Node) je softverski sistem otvorenog koda koji je dizajniran za pisanje skalabilnih Internet aplikacija, posebno serverskih veb aplikacija. Node je jedan od glavnih predstavnika paradigme "JavaScript svuda", zajedno sa MongoDB-om i JQuery-jem. Programi Node.js-a su pisani u JavaScript-u, koristeći asinhroni ulaz i izlaz (eng. *asynchronous I/O*) i vođeni događajima (eng. *event driven*), kako bi se smanjili prostorni i vremenski troškovi i povećala skalabilnost. Node se sastoji od Google-ovog V8 JavaScript pokretača, libUV i nekoliko ugrađenih biblioteka. Node je svojom efikasnošću privukao posvećene programere širom sveta da stvore veliku kolekciju dodatnih biblioteka otvorenog tipa (eng. *open soruce*), što ga čini svestranim i privlačnim za razvoj veb aplikacija [3].

7.2 Npm (node package manager)

Rastom popularnosti Node-a i nastankom mnogobrojnih Node biblioteka nastala je potreba da se one prilikom instalacije projekta usaglase. Jedan od sistema za rešenje tog problema je npm. On je pre svega onlajn repozitorijum za skladištenje i objavljivanje Node paketa otvorenog tipa. Pored toga je alatka komandne linije koja služi za interakciju sa gore pomenutim paketima. Njegove glavne funkcionalnosti su instaliranje paketa, kontrola verzija kao i održavanje zavisnosti paketa (eng. *dependency management*)[10].

8 Zaključak

Pored skromnog početka devedesetih godina prošlog veka, Javascript je postao najraspostranjeniji programski jezik. Od 2005. Postao je popularan i među profesionalnim programerima zahvaljujući AJAX-u. Ono što ga čini toliko popularnim u praksi je slaba tipiziranost jezika, nezavisnost od platforme i prepoznavanje veb pregledača i operativnog sistema na kojima se pokreće. Standardizacija jezika i širok izbor kvalitetnih razvojnih okruženja čine da se Javascript aplikacije mogu brzo i jednostavno razviti. Zahvaljujući tome uspeo je da se izdvoji od veb-a i da postane jezik koji se koristi između ostalog i za mobilne aplikacije, multimedijalne aplikacije, obradu slika, audio i video zapisa. Kako stvari stoje danas Javascript-u se u bližoj budućnosti ne vidi pad popularnosti, šta više verovatno će nastaviti da raste i da se razvija.

Literatura

[1] Netscape and Sun announce JavaScript. *PR Newswire*, 5(2), 1995. online at: https://web.archive.org/web/20070916144913/http://wp.netscape.com/newsref/pr/newsrelease67.html.

[2] Douglas Crockford. JavaScript, The World's Most Misunderstood Programming Language, 2001. online at: http://www.crockford.com/javascript/javascript.html.

[3] Jerry Cuomo. Mobile app development, JavaScript everywhere, 2013. online at: https://cse.buffalo.edu/~bina/cse487/threeamigosjs.pdf.

[4] Understanding Hoisting in JavaScript. online at: https://www.digitalocean.com/community/tutorials/understanding-hoisting-in-javascript.

[5] ECMAScript. online at: https://tc39.es/ecma262/.

[6] ES5 vs ES6. online at: https://www.c-sharpcorner.com/article/comparison-between-ecmascript-5-and-ecmascript-6-versions-of-javascript/.

[7] Brendan Eich. New JavaScript Engine Module Owner, 2011. online at: https://brendaneich.com/2011/06/new-javascript-engine-module-owner/.

[8] Marijn Haverbeke. Eloquent JavaScript, 2018. on-line at: https://eloquentjavascript.net/.

[9] JSON. online at: https://www.json.org/json-en.html.

[10] What is NPM. online at: https://nodejs.org/en/knowledge/getting-started/npm/what-is-npm/.

[11] S. Pastore. The technology evolution in web design, development and deployment supporting web scientific applications. *International Journal of Computers and Communications*, 5(2), 2011.

[12] JavaScript Data Types. online at: https://www.w3schools.com/js/js_datatypes.asp.

[13] Kyle Simpson. You Don't Know JS Yet - 2nd Edition, 2020. online at: https://github.com/getify/You-Dont-Know-JS.

[14] Stoyan Stefanov. *Object-Oriented JavaScript - Second Edition*. Packt Publishing Ltd, Birmingham, 2013.

[15] Javascript Features. online at: https://www.studytonight.com/javascript/javascript-features.