Adam Boduch, Jonathan Chaffer, Karl Swedberg

# Naučite jQuery 3

## V idanje

Izgradite interesantne, interaktivne sajtove, koristeći jQuery, automatizovanjem uobičajenih i pojednostavljivanjem komplikovanih zadataka

kompjuter biblioteka

Packt>

**Adam Boduch**
**Jonathan Chaffer**
**Karl Swedberg**

# Naučite jQuery 3

## V izdanje


**Izdavač:**

kompjuter biblioteka

Obalskih radnika 15, Beograd

**Tel: 011/2520272**

**e-mail:** kombib@gmail.com

**internet:** www.kombib.rs

**Urednik:** Mihailo J. Šolajić

**Za izdavača, direktor:**
Mihailo J. Šolajić

**Autori:** Adam Boduch
Jonathan Chaffer
Karl Swedberg

**Prevod:** Slavica Prudkov

**Lektura:** Miloš Jevtović

**Slog:** Zvonko Aleksić

**Znak Kompjuter biblioteke:**
Miloš Milosavljević

**Štampa:** Apollo Plus D.O.O.
Beograd

**Tiraž:** 500

**Godina izdanja:** 2017.

**Broj knjige:** 493

**Izdanje:** Prvo

**ISBN:** 978-86-7310-5xx-x

Learning jQuery 3

Fifth Edition
by Adam Boduch, Jonathan Chaffer and Karl Swedberg

ISBN 978-1-78588-298-2

Fifth edition: May 2017

# O AUTORIMA

**Adam Boduch** je u velikoj meri bio uključen u JavaScript razvoj skoro 10 godina. Pre nego što je prešao na čeoni prikaz, radio je na nekoliko velikih proizvoda cloud računarstva, koristeći Python i Linux. Kompleksnost mu nije strana - on ima praktično iskustvo sa stvarnim softverskim sistemima i sa problemima koje oni izazivaju.

Adam je autor nekoliko knjiga o JavaScriptu, uključujući „React“ i „React Native“, i veoma je zainteresovan za inovativna korisnička iskustva i visoke performanse.

*Zahvaljujem se Johnu Resigu za kreiranje jQueryja i celoj jQuery zajednici što su imali veoma pozitivan uticaj na veb programiranje.*

**Jonathan Chaffer** je član firme za veb razvoj Rapid Development Groupe, koja se nalazi u Grand Rapidsu, u Michiganu. Njegov posao uključuje nadgledanje i implementaciju projekata u širokom spektru tehnologija, sa naglaskom na PHP, MySQL i JavaScript. U zajednici otvorenog koda on je bio veoma aktivan u Drupal CMS projektu, koji je usvojio jQuery kao svoj izborni JavaScript radni okvir. On je tvorac Content Construction Kita koji je sada deo Drupal osnove (ona se koristi za upravljanje strukturiranim sadržajem). Takođe je odgovoran za velike prepravke Drupalovog sistema menija i developer API reference. U slobodno vreme projektuje društvene igre i igre kartama za hobi tržište. Živi u Grand Rapidsu, sa svojom suprugom Jennifer.

**Karl Swedberg** je veb programer u Fusionary Media u Grand Rapidsu, u Michiganu, gde većinu svog vremena provodi u pisanju JavaScripta i na strani klijenta i na strani servera. Kada ne programira, voli da provodi vreme sa svojom porodicom i da vežba u lokalnoj teretani.

# O RECENZENTU

**Andrew Kurz** je UI/UX dizajner i programer sa više od 12 godina iskustva u projektovanju i izgradnji veb sajtova i online aplikacija. Radio je za i mala preduzeća i za velike korporacije. Uživa da uči nove tehnologije i ceni atraktivne aplikacije jednostavne za upotrebu. Živi u Atlanti, sa svojom suprugom i troje dece. Možete da pregledate njegov portfolio i da kontaktirate sa njim na adresi www.kurzstudio.com.

# UVOD

Počeo sam da koristim jQuery 2007. godine i koristim ga još uvek. Doduše, mnogo štošta se desilo u međuvremenu: pojavile su se nove JavaScript biblioteke, više je doslednosti između pretraživača i postignuta su poboljšanja u samom JavaScriptu. Ono što se za ovih 10 godina nije promenilo je ekspresivnost i doslednost jQuerya. Bez obzira na postojanje odličnih naprednih tehnologija, jQuery ostaje omiljena alatka za brzo i efikasno izvršavanje posla.

Ova knjiga ostaje netaknuta u svom petom izdanju. Ona je uspešna zato što je veoma jasna i jednostavna za praćenje. Trudio sam se da sačuvam sve ono što je dobro funkcionisalo u ovoj knjizi. Moj cilj je popularizacija učenja jQuerya za veb programiranje.

## ŠTA OBUHVATA OVA KNJIGA

U Poglavlju 1, „*Početak rada*“, upoznaćete jQuery JavaScript biblioteku. Poglavlje započinje opisom jQuerya i onoga što on može da uradi za vas. Zatim ćete proći kroz korake preuzimanja i podešavanja biblioteke, kao i kroz pisanje prvog skripta.

Poglavlje 2, „*Selektovanje elemenata*“, pomoći će vam da naučite kako da upotrebite izraze selektora jQuerya i metode za kretanje po DOM-u za pronalaženje elemenata na stranici, gde god da se oni nalaze. Upotrebićete jQuery za primenu stilova na različite setove elemenata stranice.

Poglavlje 3, „*Obrada događaja*“, vodi vas kroz jQueryov mehanizam obrade događaja za pokretanje ponašanja kada se desi događaj u pretraživaču. Videćete kako jQuery olakšava nenametljivo povezivanje elemenata, čak i pre nego što se završi učitavanje stranice. Osim toga, opisaćemo detaljnije neke teme, kao što su bubbling događaja, prosleđivanje i imenovanje.

U Poglavlju 4, „*Stilizovanje i animiranje*“, predstavićemo vam tehnike animacije u jQueryu i kako da sakrijete, prikažete i pomerate elemente na stranici pomoću efekata koji su korisni i prijatni za oko.

U Poglavlju 5, „*Manipulisanje DOM-om*“, naučićete kako da promenite stranicu po komandi. Takođe ćete naučiti kako da promenite samu strukturu HTML dokumenta i da u njega dodate sadržaj u toku rada.

Poglavlje 6, „*Slanje podataka pomoću Ajaxa*“, vodi vas kroz mnoge načine na koje jQuery olakšava pristup funkcionalnostima servera bez potrebe da se konstantno „osvežava“ stranica. Kada budete imali osnovne komponente biblioteke „pri ruci“, bićete spremni da istražite kako biblioteka može da se proširi da biste je prilagodili svojim potrebama.

U Poglavlju 7, „*Upotreba dodatnih modula*“, prikazano je kako da pronađete, instalirate i upotrebite dodatne module, uključujući moćne biblioteke dodatnih modula jQuery UI i jQuery Mobile.

U Poglavlju 8, „*Razvijanje dodatnih modula*“, naučićete kako da iskoristite impresivne mogućnosti proširenja jQuerya za razvijanje sopstvenih dodatnih modula od nule. Kreiraćete sopstvene pomoćne funkcije, dodaćete metode jQuery objekta i otkrićete krieranje jQuery UI vidžeta. Zatim ćete pregledati gradivne blokove jQuerya i naučićete više naprednih tehnika.

Poglavlje 9, „*Napredni selektori i traversali*“, upotpuniće vaše znanje o selektorima i traversalima, pa ćete moći da optimizujete selektore za performansu manipulisanjem stekom DOM elemenata i pisanjem dodatnih modula koji proširuju mogućnosti selektovanja i kretanja.

U Poglavlju 10, „*Napredni događaji*“, detaljnije ćete „zaroniti“ u tehnike kao što su prosleđivanje i prigušivanje, što može znatno da poboljša performansu obrade događaja. Takođe ćete kreirati prilagođene i specijalne događaje koji dodaju više mogućnosti u jQuery biblioteku.

U Poglavlju 11, „*Napredni efekti*“, pokazaćemo vam kako da fino podesite vizuelne efekte jQuerya koji mogu da se obezbede kreiranjem prilagođenih funkcija i reagovanjem na svaki korak animacije. Moći ćete da manipulišete animacijama kada se one prikažu i da rasporedite akcije prilagođenim redosledom.

Poglavlje 12, „*Napredna DOM manipulacija*“, obezbeđuje vam praksu modifikovanja DOM-a pomoću tehnika kao što je dodavanje proizvoljnih podataka u elemente. Takođe ćete naučiti kako da proširite način na koji jQuery obrađuje CSS svojstva u elementima.

Poglavlje 13, „*Napredni Ajax*“, pomoći će vam da bolje razumete Ajax transakcije, uključujući jQuery sistem odloženih objekata za rukovanje podacima koji mogu kasnije postati dostupni.

U Dodatku A, „*Testiranje JavaScripta pomoću Qunita*“, predstavićemo vam QUnit biblioteku koja se koristi za testiranje JavaScript programa. Ova biblioteka će biti veliki dodatak alatima za programiranje i održavanje visokosofisticiranih veb aplikacija.

Dodatak B, „*Reference*“, sadrži pregled cele jQuery biblioteke, uključujući svaki od metoda i izraza selektora. Format ovog dodatka je veoma jednostavan za praćenje i savršen je za one momente kada znate šta želite da uradite, ali niste sigurni koji je tačan naziv metoda ili selektora.

# ŠTA VAM JE POTREBNO ZA OVU KNJIGU

Da biste pokrenuli primer koda koji je predstavljen u ovoj knjizi, potreban vam je moderan veb pretraživač, kao što su Google Chrome, Mozilla Firefox, Apple Safari ili Microsoft Edge.

Da biste eksperimentisali sa primerima i da biste radili vežbe na kraju poglavlja, takođe vam je potrebno sledeće:

- osnovni editor teksta
- alatke za razvoj veb aplikacija za pretraživač, kao što su Chrome Developer Tools ili Firebug (kao što je opisano u odeljku „Upotreba alatki za razvoj veb aplikacija“ u Poglavlju 1, „Početak rada“)
- ceo paket koda za svako poglavlje, što uključuje kopiju jQuery biblioteke (koju možete da vidite u odeljku „Preuzimanje primera koda“)

Osim toga, da biste pokrenuli neke Ajax primere u Poglavlju 6, „Slanje podataka pomoću Ajaxa“, i u narednim poglavljima, potreban vam je Node.js.

# ZA KOGA JE OVA KNJIGA

Ova knjiga je idealna za JavaScript programere na strani klijenta. Ne treba da imate prethodno iskustvo sa jQueryem, ali je neophodno osnovno poznavanje JavaScript programiranja.

# KONVENCIJE

U ovoj knjizi pronaći ćete više različitih stilova za tekst koje sam upotrebio za različite vrste informacija. Evo nekih primera ovih stilova i objašnjenja njihovog značenja.

Reči koda u tekstu, nazivi tabela baze podataka, nazivi direktorijuma, nazivi fajlova, ekstenzije fajla, nazivi putanja, kratki URL-ovi, korisnički unos i Twitter identifikatori su prikazani na sledeći način: „Kada damo instrukcije da jQuery pronađe sve elemente klase collapsible i sakrije ih, ne treba kružiti kroz svaki vraćeni element“.

Blok koda je postavljen na sledeći način:

```
body {
  background-color: #fff;
  color: #000;
  font-family: Helvetica, Arial, sans-serif;
}
h1, h2, h3 {
  margin-bottom: .2em;
```

```
}
.poem {
  margin: 0 2em;
}
.highlight {
  background-color: #ccc;
  border: 1px solid #888;
  font-style: italic;
  margin: 0.5em 0;
  padding: 0.5em;
}
```

**Novi termini** i **važne reči** su napisani masnim slovima. Reči koje vidite na ekranu - na primer, u menijima ili okvirima za dijalog, biće prikazane u tekstu na sledeći način: „Kartica **Sources** omogućava da prikažemo sadržaje svih učitanih skriptova na stranici“.


Upozorenja ili važne napomene će biti prikazani u ovakvom okviru.


Saveti i trikovi prikazani su ovako.

# POVRATNE INFORMACIJE OD ČITALACA

Povratne informacije od naših čitalaca su uvek dobrodošle. Obavestite nas šta mislite o ovoj knjizi – šta vam se dopalo ili šta vam se možda nije dopalo. Povratne informacije čitalaca su nam važne da bismo kreirali naslove od kojih ćete dobiti maksimum.

Da biste nam poslali povratne informacije, jednostavno nam pošaljite e-mail na adresu informatori@kombib.rs i u naslovu poruke napišite naslov knjige.

## Preuzimanje primera koda

Možete da preuzmete fajlove sa primerima koda prateći sledeće korake:

1. Posetite veb stranicu knjige Naučite jQuery 3: https://goo.gl/N3Ze3M
2. Kliknite Preuzmite kod: https://goo.gl/4HgcHL

Kada su fajlovi preuzeti, ekstrahujte direktorijum, koristeći najnoviju verziju:

- WinRAR / 7-Zip za Windows
- Zipeg / iZip / UnRarX za Mac
- 7-Zip / PeaZip za Linux

Kod za ovu knjigu možete da pronađete i na GitHubu na adresi https://github.com/PacktPublishing/Learning-jQuery-3

# ŠTAMPARSKE GREŠKE

Iako smo preduzeli sve mere da bismo obezbedili tačnost sadržaja, greške su moguće. Ako pronađete grešku u nekoj od naših knjiga (u tekstu ili u kodu), bili bismo zahvalni ako biste nam to prijavili. Na taj način možete da poštedite čitaoce od frustracije i nama da pomognete da poboljšamo naredne verzije ove knjige. Ako pronađete neku štamparsku grešku, molimo vas da nas obavestite, tako što ćete posetiti stranicu knjige: https://goo.gl/N3Ze3M i ostaviti komentar.

# PIRATERIJA

Piraterija autorskog materijala na Internetu je aktuelan problem na svim medijima. Mi u „Packtu“ zaštitu autorskih prava i licenci shvatamo veoma ozbiljno. Ako pronađete ilegalnu kopiju naših knjiga, u bilo kojoj formi, na Internetu, molimo vas da nas o tome obavestite i pošaljete nam adresu lokacije ili naziv web sajta da bismo mogli da podnesemo tužbu.

Kontaktirajte sa nama na adresi copyright@packtpub.com i informatori@kombib.rs pošaljite nam link ka sumnjivom materijalu.

# 1

# Početak rada

Današnji **World Wide Web (WWW)** je dinamičko okruženje i njegovi korisnici postavljaju visoke zahteve što se tiče stila i funkcije sajtova. Da bi izgradili interesantne i interaktivne sajtove, programeri koriste JavaScript biblioteke, kao što je jQuery, radi automatizovanja uobičajenih i pojednostavljivanja komplikovanih zadataka. Razlog zbog kojeg je jQuery popularan izbor je njegova mogućnost da pomogne u širokom rasponu zadataka.

jQuery izvršava mnogo različitih funkcija. Ipak, u dizajnu biblioteke postoje koherentnost i simetrija; mnogi od njenih koncepata su pozajmljeni iz strukture **HTML**-a i **Cascading Style Sheetsa (CSS-a)**. Dizajn biblioteke omogućava brz početak dizajnerima koji imaju malo programerskog iskustva. U stvari, u ovom početnom poglavlju ćemo napisati funkcionalni jQuery program u samo tri linije koda. Sa druge strane, iskusni programeri će takođe ceniti ovu konceptualnu doslednost.

U ovom poglavlju ćemo opisati:

- osnovne funkcije jQuerya
- podešavanje okruženja jQuery koda
- jednostavan funkcionalni primer jQuery skripta
- razloge za biranje jQuerya, umesto običnog JavaScripta
- uobičajene JavaScript programerske alatke

# ŠTA JQUERY RADI?

Biblioteka jQuery obezbeđuje sloj apstrakcije opšte namene za uobičajeno veb programiranje i zbog toga je korisna u skoro svakoj situaciji u toku programiranja. Njena proširiva priroda označava da nikada ne možemo da obuhvatimo sve moguće načine upotrebe i funkcije u jednoj knjizi, jer se konstantno razvijaju dodatni moduli za dodavanje novih mogućnosti. Međutim, osnovne funkcije nam pomažu da izvršimo sledeće zadatke:

- **pristupanje elementima u dokumentu** - Bez JavaScript biblioteke veb programeri često treba da napišu mnogo linija koda da bi se kretali u Document Object Model (DOM) stablu i locirali specifične delove strukture HTML dokumenta. Zahvaljujući biblioteci jQuery, programeri imaju robustan i efikasan mehanizam selektora na raspolaganju, što olakšava pronalaženje određenog dela dokumenta koji treba da se ispita ili kojim treba da se manipuliše.

```
$('div.content').find('p');
```

- **modifikovanje izgleda veb stranice** - CSS pruža moćan metod za uticaj na način renderovanja dokumenta, ali zaostaje kada ne podržavaju svi pretraživači iste standarde. Pomoću biblioteke jQuery programeri mogu da premoste ovaj jaz, oslanjajući se na podršku istih standarda u svim pretraživačima. Osim toga, jQuery može da promeni klase ili individualna svojstva stila primenjene na deo dokumenta, čak i nakon što je stranica renderovana.

```
$('ul > li:first').addClass('active');
```

- **menjanje sadržaja dokumenta** - Ne ograničavajući se samo na kozmetičke promene, jQuery može da modifikuje i sadržaj samog dokumenta uz svega nekoliko otkucaja. Može da bude promenjen tekst, slike mogu da budu ubačene ili zamenjene, liste mogu da budu preorganizovane ili cela struktura HTML-a može da bude prepisana i proširena - sve to pomoću jednog **Application Programming Interfacea (API)**, koji je veoma jednostavan za upotrebu.

```
$('#container').append('<a href="more.html">more</a>');
```

- **odgovaranje na interakciju korisnika** - Čak ni najsloženija i najmoćnija ponašanja nisu korisna ako ne možemo da kontrolišemo kada se ona dešavaju. Biblioteka jQuery omogućava elegantan način presretanja širokog raspona događaja, kao što je klik korisnika na link, bez potrebe da se sam HTML kod proširuje funkcijama za obradu događaja.

```
$('button.show-details').click(() => {
  $('div.details').show();
});
```

- **animiranje promena koje su izvršene u dokumentu** - Da bi efikasno implementirao ovakvo interaktivno ponašanje, dizajner takođe mora da obezbedi vizuelnu povratnu informaciju korisniku. Biblioteka jQuery to olakšava obezbeđivanjem niza efekata, kao što su zamračivanje i nestajanje, kao i setom alatki za kreiranje novih efekata.

```
$('div.details').slideDown();
```

- **preuzimanje informacija sa servera bez „osvežavanja“ stranice** - Ovaj šablon je poznat kao Ajax, što je originalno predstavljalo skraćenicu za **Asynchronous JavaScript and XML**, ali sada predstavlja mnogo veći set tehnologija za komunikaciju između klijenta i servera. Biblioteka jQuery uklanja složenost specifičnu za pretraživač iz procesa, omogućavajući programerima da se fokusiraju na funkcionalnost na strani servera.

```
$('div.details').load('more.html #content');
```

# ZAŠTO JQUERY FUNKCIONIŠE DOBRO?

Zbog povećanja interesa za dinamički HTML, dolazi do širenja JavaScript radnih okvira. Neki su specijalizovani i fokusiraju se samo na jedan ili dva zadatka (prethodno spomenuta). Neki „pokušavaju“ da katalogiziraju svako moguće ponašanje i animaciju i „posluže“ ih u unapred kreiranim paketima. Da bi održala širok raspon funkcija koje su prethodno opisane i ujedno zadržala relativnu kompaktnost, biblioteka jQuery primenjuje nekoliko strategija:

- **iskorišćavanje znanje CSS-a** - Baziranjem mehanizma za lociranje elemenata stranice na CSS selektorima jQuery nasleđuje sažet, ali čitljiv način izražavanja strukture dokumenta. Biblioteka jQuery postaje ulazna tačka za dizajnere koji žele da dodaju ponašanja u svoje stranice, jer je poznavanje CSS sintakse preduslov za profesionalno veb programiranje.
- **podrška za ekstenzije** - Da bi izbegla „lažne funkcije“, biblioteka jQuery preusmerava specijalne slučajeve upotrebe dodatnih modula. Metod za kreiranje novih dodatnih modula je jednostavan i dobro je dokumentovan, što je podstaklo razvoj raznovrsnih inventivnih i korisnih modula. Čak se većina funkcija u osnovnom jQuery paketu interno realizuje kroz arhitekturu dodatnih modula i one mogu da budu uklonjene ako je potrebno, što dovodi do još manje biblioteke.

- **uklanjanje nepravilnosti pretraživača** - Nesrećna okolnost u veb programiranju je što svaki pretraživač ima sopstveni skup odstupanja od publikovanih standarda. Značajan deo svake veb aplikacije može na drugačiji način da se prosledi funkcijama za obradu za svaku platformu. Iako konstantno razvijanje pretraživača onemogućava savršenu osnovu koda koja je neutralna za pretraživač za neke napredne funkcije, jQuery dodaje apstrakciju sloja koja normalizuje uobičajene zadatke, smanjujući veličinu koda i znatno ga pojednostavljujući.
- **korišćenje skupova** - Kada jQueryu damo instrukcije da pronađe sve elemente klase `collapsible` i da ih sakrije, nema potrebe kružiti kroz svaki vraćeni element. Umesto toga, metodi kao što je `.hide()` namenjeni su da automatski koriste skupove objekata, umesto pojedinačnih objekata. Upotreba ove tehnike, koja se naziva *implicitna iteracija*, znači da mnoge strukture kruženja postaju nepotrebne, što značajno skraćuje kod.
- **omogućavanje višestrukih akcija u jednoj liniji** - Da bi se izbeglo prekomerno korišćenje *privremenih promenljivih* ili napotrebno ponavljanje, jQuery koristi za većinu svojih metoda programski šablon pod nazivom ulančavanje. Ulančavanje označava da je rezultat većine operacija na objektu sam objekat, koji je spreman za sledeću akciju koja će na njega biti primenjena.

Ove strategije zadržavaju malu veličinu fajla jQuery paketa, dok istovremeno obezbeđuju tehnike da prilagođeni kod koji koristi biblioteku ostane kompaktan.

Elegancija biblioteke se ogleda delimično u dizajnu i delimično u razvojnom procesu koji širi zajednica korisnika i koji se širi oko projekta. Korisnici biblioteke jQuery sastaju se i diskutuju ne samo o razvoju dodatnih modula, već i o poboljšanjima u njenoj osnovi. Korisnici i programeri takođe pomažu u kontinualnom poboljšanju zvanične dokumentacije projekta, koju možete da pronađete na adresi http://api.jquery.com.

Bez obzira na sav napor koji je potreban za kreiranje tako fleksibilnog i robusnog sistema, krajnji proizvod je besplatno dostupan svima za korišćenje. Ovaj projekat otvorenog koda je licenciran pod MIT License da bi bila dozvoljena besplatna upotreba biblioteke jQuery na bilo kom sajtu i olakšana upotreba ove biblioteke unutar vlasničkog softvera. Ako projekat zahteva ovu biblioteku, programeri mogu da licenciraju jQuery pod GNU Public License za uključivanje u druge projekte otvorenog koda pod GNU licencom.

# ŠTA JE NOVO U VERZIJI JQUERY 3?

Promene koje su izvršene u verziji jQuery 3 su prilično suptilne u poređenju sa promenama koje su uvedene u verziju jQuery 2. Većina promena izvršena je „ispod haube“. Videćete sada neke od promena i kako će one uticati na postojeće jQuery projekte. Možete da pregledate detalje na adresi https://jquery.com/upgrade-guide/3.0.

# Podrška pretraživača

Najveća promena u podršci pretraživača u verziji jQuery 3 je podrška za Internet Explorer. Potreba za podrškom starijih verzija ovog pretraživača je „noćna mora“ za svakog veb programera. Biblioteka jQuery 3 predstavlja veliki korak napred, jer podržava samo verzije IE9+. Što se tiče podrške za druge pretraživače, podržane su aktuelna i prethodna verzija.


Vreme Internet Explorera je prošlo. „Microsoft“ je promovisao „naslednika“ IE pretraživača, pod nazivom Edge. Ovaj pretraživač je potpuno poseban projekat od IE pretraživača (nije opterećen njegovim problemima). Osim toga, najnovije verzije Microsoft Windowsa forsiraju Edge kao standardni pretraživač; ažuriranja su regularna i predvidiva.

# Deferred objekti

`Deferred` objekti su predstavljeni u verziji jQuery 1.5 kao sredstvo za bolje upravljanje asinhronim ponašanjem. Oni su bili poput ES2015 promisa, ali se dovoljno razlikuju da se ne mogu zameniti. Sada, kada je ES2015 verzija JavaScripta uobičajena u modernim pretraživačima, `Deferrred` objekat je potpuno kompatibilan sa izvornim `Promise` objektima. To znači da se mnogo štošta promenilo u staroj `Deferred` implementaciji.

# Asinhrono ponašanje spremno za dokument

Ideja da je funkcija povratnog poziva spremna za dokument izvršena asinhrono možda na prvi pogled izgleda kontraintuitivno. Postoji nekoliko razloga zbog kojih ova funkcija izgleda kontraintuitivno u jQueryu 3. Prvi razlog je činjenica da izraz `$(() => {})` vraća `Deferred` instancu, a one se onda ponašaju kao izvorni promisi. Drugi razlog je postojanje `jQuery.ready` promisa koji se rešava kada je dokument spreman. Kao što ćete videti kasnije u ovoj knjizi, možete da upotrebite ovaj promis zajedno sa drugim promisima za izvršavanje drugih asinhronih zadataka pre nego što DOM bude spreman za renderovanje.

# Sve ostalo

Postoji i niz drugih promena u API-ju koje su predstavljene u verziji jQuery 3 i koje nećemo ovde opisivati. U vodiču za nadgradnju verzije koji sam ranije spomenuo detaljno su opisane sve ove promene i način kako možemo da ih izvršimo. Međutim, opisaću funkcionalnosti koje su nove ili se razlikuju u verziji jQuery 3.

# KREIRANJE PRVE VEB STRANICE KOJU POKREĆE JQUERY

Sada, kada smo opisali raspon funkcija koje su dostupne u jQueryu, možemo da istražimo kako se biblioteka pokreće. Za početak, potrebno je da preuzmemo kopiju biblioteke jQuery.

## Preuzimanje jQuerya

Nije potrebna instalacija. Da bismo upotrebili jQuery, potrebna nam je samo javno dostupna kopija fajla, bez obzira da li se ona nalazi na eksternom ili na našem sajtu. Pošto je JavaScript interpretirani jezik, ne postoji faza kompilacije ili izgradnje, o kojoj treba da brinemo. Kad god je potrebno da stranica ima dostupan jQuery, jednostavno ćemo uneti lokaciju fajla iz elementa `<script>` u HTML dokumentu.

Zvanični veb sajt jQuerya (http://jquery.com/) uvek ima najažurniju stabilnu verziju biblioteke, koja može da se preuzme direktno sa početne stranice sajta. U određenom momentu može biti dostupno nekoliko verzija jQuerya; ona koja nama, kao veb programerima, najviše odgovara biće najnovija nekompresovana verzija biblioteke. Ta verzija može da se zameni kompresovanom verzijom u proizvodnom okruženju.

Pošto je popularnost jQuerya u porastu, kompanije su fajl učinile besplatno dostupnim kroz njihove Content Delivery Networks (CDNs). Pre svega, „Google“ (https://developers.google.com/speed/libraries/devguide), „Microsoft“ (http://www.asp.net/ajaxlibrary/cdn.ashx) i sam projekat jQuery (http://code.jquery.com) nude fajl na moćnim, niskolatentnim serverima, distribuiranim širom sveta za brzo preuzimanje, bez obzira na lokaciju korisnika. Iako kopija jQuerya hostovana na CDN-u ima pednosti u brzini, nasuprot distribuciji servera i keširanja, u toku programiranja je najbolje da upotrebimo lokalnu kopiju. U ovoj knjizi ćemo upotrebiti kopiju fajla koji se nalazi na našem sistemu, što će nam omogućiti da pokrećemo kod, bez obzira da li imamo internet konekciju ili ne.


Da biste izbegli programske greške, uvek upotrebite specifičnu verziju jQuerya - na primer, 3.1.1. Neki CDN-ovi omogućavaju povezivanje sa najnovijom verzijom biblioteke. Osim toga, ako koristite npm za instaliranje jQuerya, uvek se uverite da package.json zahteva specifičnu verziju.

# Podešavanje jQuerya u HTML dokumentu

Za većinu primera upotrebe jQueryja postoje HTML dokument, CSS fajlovi za stilove i JavaScript fajlovi za delovanje. Za naš prvi primer mi ćemo upotrebiti stranicu sa isečkom knjige koja ima primenjen veliki broj klasa u delu stranice. Ova stranica uključuje referencu ka najnovijoj verziji jQuery biblioteke, koju smo preuzeli, promenili joj naziv na jquery.js i postavili je u lokalni direktorijum projekta:

```
<!DOCTYPE html>

<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Through the Looking-Glass</title>

    <link rel="stylesheet" href="01.css">

    <script src="jquery.js"></script>
    <script src="01.js"></script>
  </head>

  <body>
    <h1>Through the Looking-Glass</h1>
    <div class="author">by Lewis Carroll</div>

    <div class="chapter" id="chapter-1">
      <h2 class="chapter-title">1. Looking-Glass House</h2>
      <p>There was a book lying near Alice on the table, and
        while she sat watching the White King (for she was
        still a little anxious about him, and had the ink
        all ready to throw over him, in case he fainted
        again), she turned over the leaves, to find some
        part that she could read, <span class="spoken">
        "&mdash;for it's all in some language I don't know,"
        </span> she said to herself.</p>
      <p>It was like this.</p>
      <div class="poem">
        <h3 class="poem-title">YKCOWREBBAJ</h3>
        <div class="poem-stanza">
          <div>sevot yhtils eht dna ,gillirb sawT'</div>
          <div>;ebaw eht ni elbmig dna eryg diD</div>
          <div>,sevogorob eht erew ysmim llA</div>
          <div>.ebargtuo shtar emom eht dnA</div>
        </div>
      </div>
      <p>She puzzled over this for some time, but at last
        a bright thought struck her. <span class="spoken">
```

```
        "Why, it's a Looking-glass book, of course! And if
        I hold it up to a glass, the words will all go the
        right way again."</span></p>
      <p>This was the poem that Alice read.</p>
      <div class="poem">
        <h3 class="poem-title">JABBERWOCKY</h3>
        <div class="poem-stanza">
          <div>'Twas brillig, and the slithy toves</div>
          <div>Did gyre and gimble in the wabe;</div>
          <div>All mimsy were the borogoves,</div>
          <div>And the mome raths outgrabe.</div>
        </div>
      </div>
    </div>
  </body>
</html>
```

Odmah nakon normalne HTML preambule, učitava se opis stilova. Za ovaj primer mi ćemo upotrebiti jednostavan opis:

```
body {
  background-color: #fff;
  color: #000;
  font-family: Helvetica, Arial, sans-serif;
}
h1, h2, h3 {
  margin-bottom: .2em;
}
.poem {
  margin: 0 2em;
}
.highlight {
  background-color: #ccc;
  border: 1px solid #888;
  font-style: italic;
  margin: 0.5em 0;
  padding: 0.5em;
}
```


Možete da pristupite primeru koda iz GitHub skladišta https://github.com/PacktPublishing/Learning-jQuery-3.

Nakon što je opis stilova referenciran, uključeni su JavaScript fajlovi. Važno je da oznaka `script` za jQuery biblioteku bude postavljena ispred oznake za prilagođene skriptove; u suprotnom, radni okvir jQuery neće biti dostupan kada kod pokuša da ga referencira.


U ostatku ove knjige biće odštampani samo relevantni delovi HTML i CSS fajlova. Kompletni fajlovi su dostupni u primerima koda za ovu knjigu: https://github.com/PacktPublishing/Learning-jQuery-3.

Sada imamo stranicu koja izgleda ovako:

# Through the Looking-Glass

by Lewis Carroll

## 1. Looking-Glass House

There was a book lying near Alice on the table, and while she sat watching the White King (for she was still a little anxious about him, and had the ink all ready to throw over him, in case he fainted again), she turned over the leaves, to find some part that she could read, "—for it's all in some language I don't know," she said to herself.

It was like this.

**YKCOWREBBAJ**
sevot yhtils eht dna ,gillirb sawT'
;ebaw eht ni elbmig dna eryg diD
,sevogorob eht erew ysmim llA
.ebargtuo shtar emom eht dnA

She puzzled over this for some time, but at last a bright thought struck her. "Why, it's a Looking-glass book, of course! And if I hold it up to a glass, the words will all go the right way again."

This was the poem that Alice read.

**JABBERWOCKY**
'Twas brillig, and the slithy toves
Did gyre and gimble in the wabe;
All mimsy were the borogoves,
And the mome raths outgrabe.

Upotrebićemo jQuery da bismo primenili novi stil za tekst pesme.


U ovom primeru je prikazana jednostavna upotreba jQuerya. U stvarnim situacijama ovaj tip stilizovanja bi mogao da se izvrši samo pomoću CSS-a.

# Dodavanje jQuery koda

Naš prilagođeni kod će biti snimljen u trenutno prazan JavaScript fajl koji smo uključili iz HTML-a pomoću skripta <script src="01.js"></script>. Za ovaj primer potrebne su nam samo tri linije koda:

```
$(() => {
  $('div.poem-stanza').addClass('highlight')
{);
```


Ja ću koristiti noviju ES2015 sintaksu arrow funkcije za većinu funkcija povratnih poziva u ovoj knjizi. Jedini razlog je činjenica da je ova sintaksa mnogo konciznija, nego unošenje ključne reči function u celom kodu. Međutim, ako vama više odgovara sintaksa function() {}, slobodno je upotrebite.

Sada ćemo da pregledajmo ovaj skript, deo po deo, da bismo videli kako on funkcioniše.

# Pronalaženje teksta pesme

Osnovna operacija u jQueryu je selektovanje dela dokumenta pomoću funkcije `$()`. Ova funkcija koristi, obično, znakovni niz kao parametar, koji može da sadrži bilo koji CSS izraz selektora. U ovom slučaju mi želimo da pronađemo sve  elemente u dokumentu koji imaju primenjenu klasu `poem-stanza`, pa je, stoga, selektor veoma jednostavan. Međutim, opisaćemo mnogo sofisticiranije opcije kasnije u ovoj knjizi. Opisaćemo mnoge načine lociranja delova dokumenta u Poglavlju 2, „*Selektovanje elemenata*".

Kada je pozvana, funkcija `$()` vraća novu instancu jQuery objekta, koja je osnovni gradivni blok koji ćemo od sada koristiti. Ovaj objekat kapsulira više DOM elemenata i omogućava interakciju sa njima na mnogo različitih načina. U ovom slučaju mi želimo da modifikujemo izgled ovih delova stranice i to ćemo uraditi menjanjem klasa koje su primenjene u tekstu pesme.

# Injektiranje nove klase

Metod `.addClass()`, kao i većina jQuery metoda, ima opisni naziv; on primenjuje CSS klasu na deo stranice koji smo selektovali. Jedini parametar ovog metoda je naziv klase koja se dodaje. Ovaj metod i metod `.removeClass()` će nam omogućiti da lako pratimo jQuery u akciji i da istražimo različite izraze selektora koji su nam dostupni. Za sada, naš primer jednostavno dodaje klasu `highlight`, koja je u opisu stilova definisana kao iskrivljeni tekst sa sivom pozadinom i okvirom.


Vidite da nije potrebna iteracija za dodavanje klase u sve strofe pesme. Kao što smo već napomenuli, jQuery koristi implicitnu iteraciju unutar metoda, kao što je .addClass(), pa je potreban svega jedan poziv funkcije za promenu selektovanog dela dokumenta.

# Izvršavanje koda

Upotrebljene zajedno, funkcije `$()` i `.addClass()` su nam dovoljne da bismo postigli naš cilj menjanja izgleda teksta pesme. Međutim, ako je ova linija koda ubačena samostalno u zaglavlje dokumenta, neće imati uticaja na tekst. JavaScript kod je pokrenut odmah nakon što je pronađen u pretraživaču; u vreme kada je obrađeno zaglavlje u stilu još nema prisutnog HTML-a. Treba da odložimo izvršenje koda, dok nam ne bude dostupan DOM za upotrebu.

Pomoću strukture `$(() => {})` - prosleđivanjem funkcije, umesto izraza selektora, jQuery omogućava da rasporedimo pozive funkcije za pokretanje kada je DOM učitan, bez potrebe da čekamo da slike budu u potpunosti renderovane. Iako je raspoređivanje događaja moguće bez pomoći jQuerya, struktura `$(() => {})` obezbeđuje posebno elegantno rešenje između pretraživača, koje uključuje sledeće funkcije:

- Koristi izvornu implementaciju spremnu za DOM u pretraživaču kada je dostupna i dodaje funkciju za obradu događaja window.onload kao pomoć.
- Izvršava funkcije koje su prosleđene u `$()`, čak i ako je funkcija pozvana nakon što se već desio događaj pretraživača.
- Obrađuje raspoređivanje događaja asinhrono da bi bilo omogućeno skriptovima da budu odloženi ako je to potrebno.

Parametar funkcije `$()` može da prihvati referencu za već definisanu funkciju, kao što je prikazano u sledećem isečku koda:

```
function addHighlightClass()  {
  $('div.poem-stanza').addClass('highlight');
}

$(addHighlightClass);
```

Programski kod 1.1

Međutim, kao što je prikazano u originalnoj verziji skripta i ponovljeno u programskom kodu 1.2, metod takođe može da prihvati anonimnu funkciju:

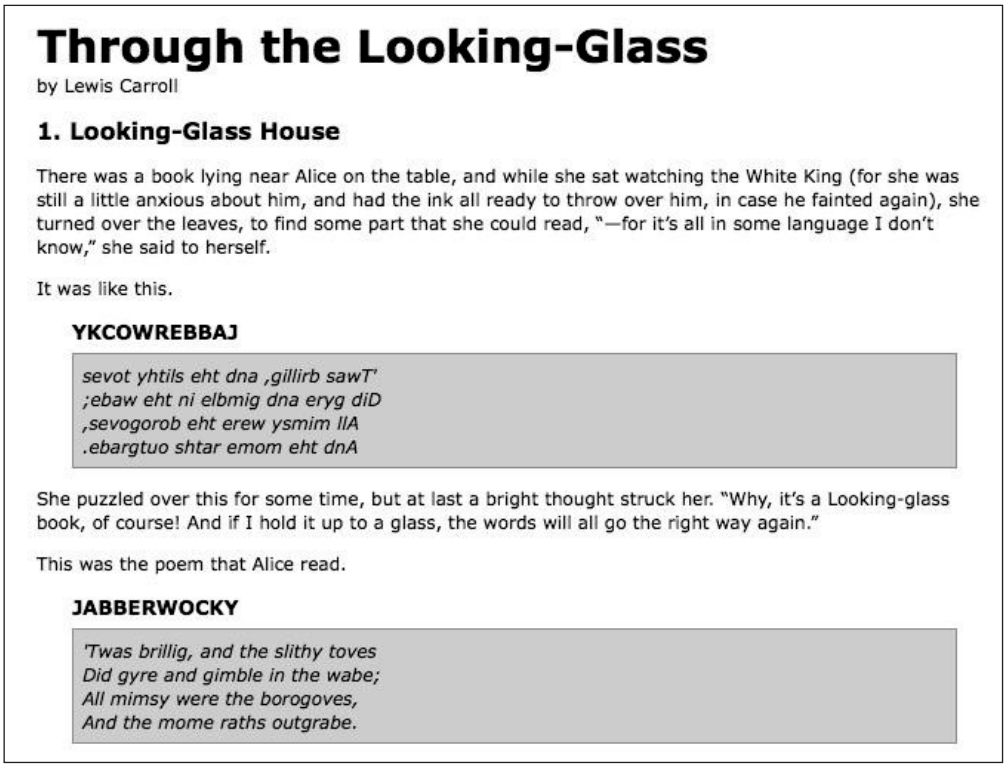
```
$(() =>
  $('div.poem-stanza').addClass('highlight')
);
```

Programski kod 1.2

Idiom ove anonimne funkcije je pogodan za jQuery kod za metode koji koriste funkciju kao argument kada funkcija ne može ponovo da se upotrebi. Štaviše, zatvoreni izraz koji funkcija kreira može da bude napredna i moćna alatka. Ako koristite arrow funkcije, takođe ćete imati leksičku vezu this kao kontekst, pa nećete morati da povezujete funkcije. Međutim, ako niste pažljivi, povezivanje funkcija takođe može pretrpeti neželjene posledice zbog upotrebe memorije.

# Završen proizvod

Sada, kada je postavljen JavaScript, stranica izgleda ovako:




Stihovi pesme su sada ispisani kosim slovima i obuhvaćeni su u okvir, kao što je specifikovano u opisu stila 01.css, zahvaljujući klasi `highlight` koja je dodata u JavaScript kod.

# OBIČAN JAVASCRIPT, NASUPROT JQUERYA

Čak i zadatak koji je jednostavan, kao što je ovaj, može da bude komplikovan bez upotrebe jQuerya. U običnom JavaScriptu možemo da dodamo klasu `highlight` na sledeći način:

```
window.onload =
function() {
  const divs = document.getElementsByTagName('div');
  const hasClass = (elem, cls) =>
    new RegExp(` ${cls} `).test(` ${elem.className} `);

  for (let div of divs) {
    if (hasClass(div, 'poem-stanza') && !hasClass(div,
    'highlight')) {
      div.className += '
      highlight';
    }
  }
};
```

Programski kod 1.3

Bez obzira na dužinu, u ovom rešenju neće biti obrađene mnoge situacije o kojima brine jQuery u programskom kodu 1.2:

- Neophodno je pravilno poštovanje drugih funkcija za obradu događaja window.onload.
- jQuery deluje čim je DOM spreman.
- jQuery optimizuje preuzimanje elementa i druge zadatke pomoću modernih DOM metoda.

Možemo da vidimo da je kod koji pokreće jQuery lakši za pisanje, jednostavniji za čitanje i brži za izvršavanje od običnog JavaScript koda.

# UPOTREBA PROGRAMERSKIH ALATKI

Kao što i poređenje koda prikazuje, jQuery kod je, obično, kraći i jasniji od njegovog osnovnog JavaScript ekvivalentnog koda. Međutim, to ne znači da ćemo uvek pisati kod koji nema programske greške ili da ćemo uvek intuitivno razumeti šta se dešava na stranicama. Programiranje jQuery je mnogo jednostavnije kada se vrši pomoću standardnih programerskih alatki.

U svim modernim pretraživačima dostupne su veoma kvalitetne programerske alatke. Slobodno možemo da upotrebimo okruženje koje nam najviše odgovara. Opcije uključuju sledeće:

- Microsoft Edge (https://developer.microsoft.com/en-us/microsoft-edge/platform/documentation/f12-devtools-guide/)
- Internet Explorer Developer Tools (http://msdn.microsoft.com/en-us/library/dd565628.aspx)
- Safari Web Development Tools (https://developer.apple.com/safari/tools/)
- Chrome Developer Tools (https://developer.chrome.com/devtools)
- Firefox Developer Tools (https://developer.mozilla.org/en-US/docs/Tools)

Svaki od ovih skupova alatki pruža slične programerske funkcije, uključujući:

- istraživanje i modifikovanje aspekata DOM-a
- istraživanje međusobnog odnosa između CSS-a i njegovog efekta na prezentaciju stranice
- pogodno traženje izvršenja skripta kroz specijalne metode
- pauziranje u toku izvršenja pokrenutih skriptova i istraživanje vrednosti promenljivih

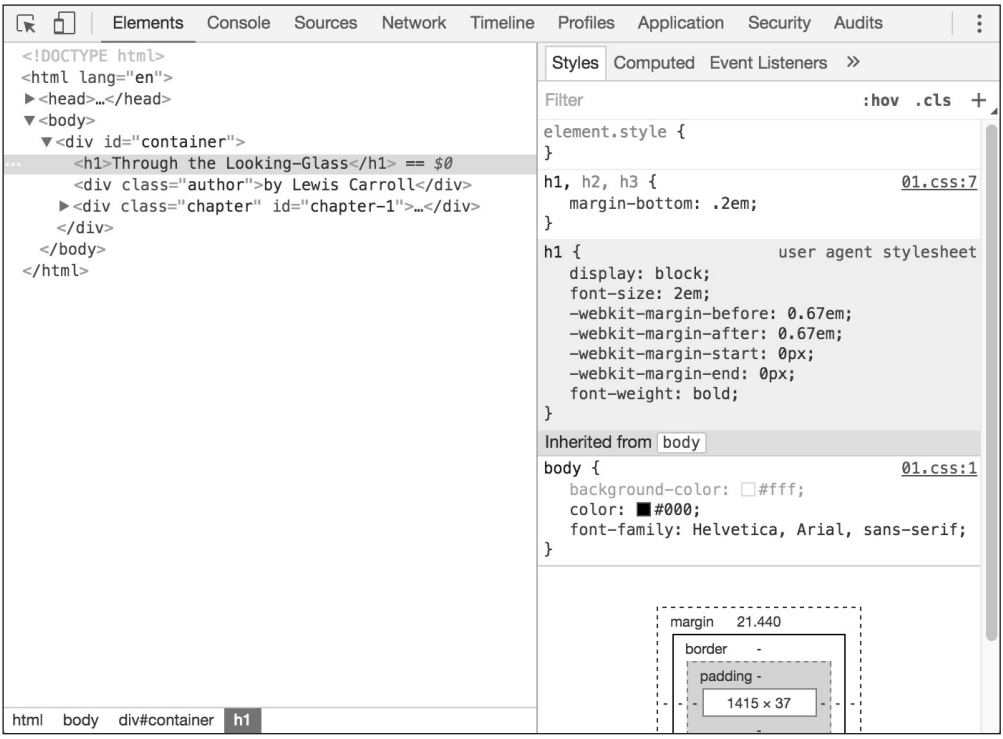
Iako detalji ovih funkcija variraju od jedne alatke do druge, osnovni koncepti ostaju isti. U ovoj knjizi neki primeri će zahtevati upotrebu jedne od tih alatki; mi ćemo upotrebiti Chrome Developer Tools za ove prikaze, mada su i programerske alatke drugih pretraživača dobra alternativa.

# Chrome Developer Tools

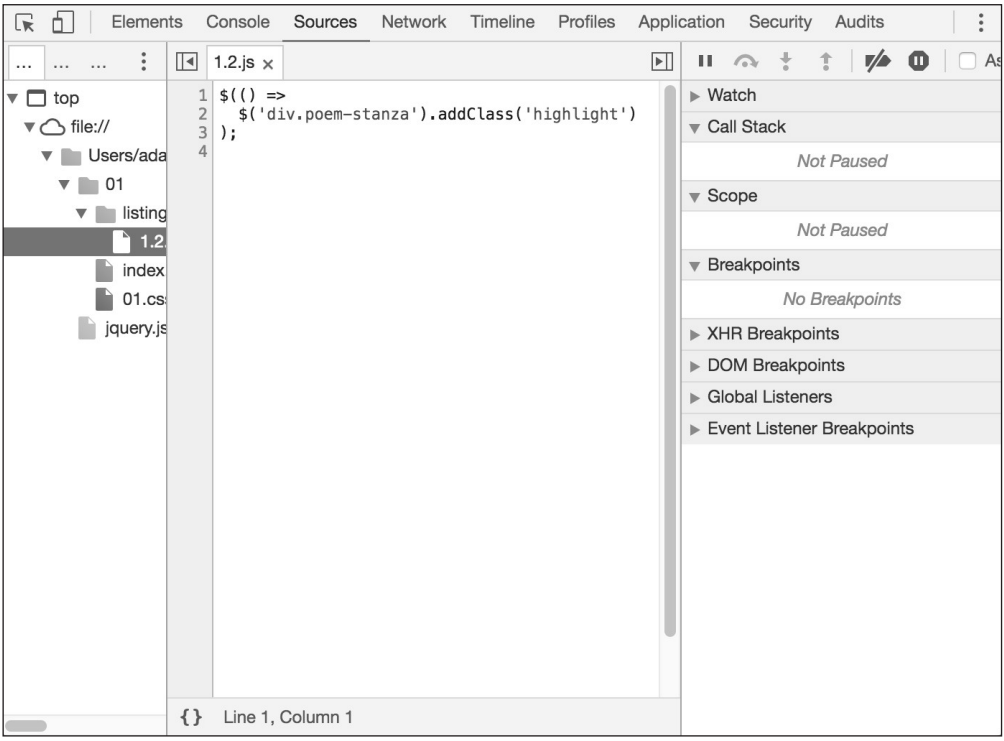
Ažurne instrukcije za pristup i upotrebu alatke Chrome Developer Tools možete pronaći na stranicama dokumentacije projekta na adresi https://developer.chrome.com/devtools. Bilo bi previše da u ovoj knjizi detaljno opisujemo ove alatke, pa ćemo da opišemo neke od najkorisnijih funkcija.


**Razumevanje ovih snimaka ekrana**
Chrome Developer Tools je projekat koji se veoma brzo razvija, tako da se sledeći snimci ekrana možda neće u potpunosti podudarati sa vašim okruženjem.

Kada je aktiviran Chrome Developer Tools, biće prikazan novi panel koji pruža informacije o aktuelnoj stranici. Na standardnoj **Elements** kartici ovog panela možemo da vidimo prikaz strukture stranice sa leve strane i detalje selektovanog elementa (kao što su CSS pravila koja su primenjena) sa desne strane. Ova kartica je posebno korisna za istraživanje strukture stranice i ispravljanje CSS grešaka:

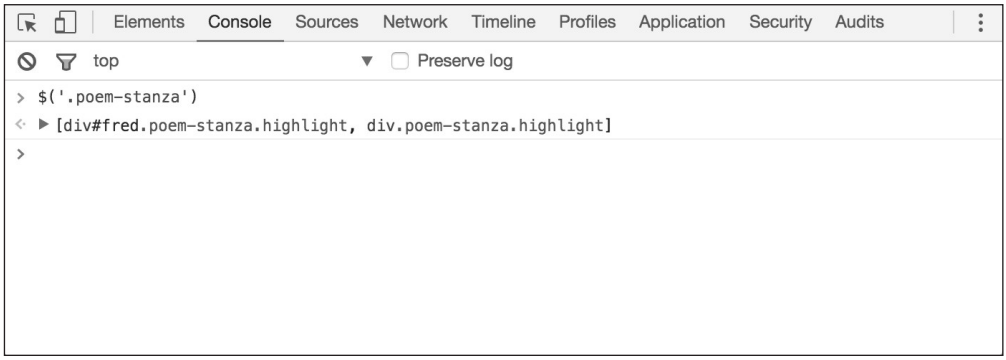


Kartica **Sources** omogućava da pregledamo sadržaje svih učitanih skriptova na stranici. Desnim klikom na broj linije možemo da postavimo tačku prekida, da podesimo uslovnu tačku prekida ili da podesimo da se skript nastavlja u toj liniji nakon što je postignuta još jedna tačka prekida. Tačke prekida su efikasan način za pauziranje pri izvršenju skripta i ispitivanje šta se dešava na način korak-po-korak. Sa desne strane stranice možemo da unesemo listu promenljivih i izraza za koje želimo da znamo vrednosti u bilo koje vreme:




Karticu **Console** ćemo najčešće koristiti dok učimo jQuery. Polje na dnu panela omogućava da unesemo bilo koji JavaScript iskaz, a rezultat iskaza će biti prikazan u panelu.

U ovom primeru izvršićemo isti jQuery kao što je onaj koji je prikazan u programskom kodu 1.2, ali nećemo izvršavati nikakve akcije na selektovanim elementima. Čak i uz takvo ograničenje, iskaz nam daje interesantne informacije - vidimo da je rezultat selektora jQuery objekat koji pokazuje ka dva `.poem-stanza` elementa na stranici. Možemo da upotrebimo ovu funkciju konzole da bismo brzo isprobali jQuery kod u bilo koje vreme direktno unutar pretraživača:

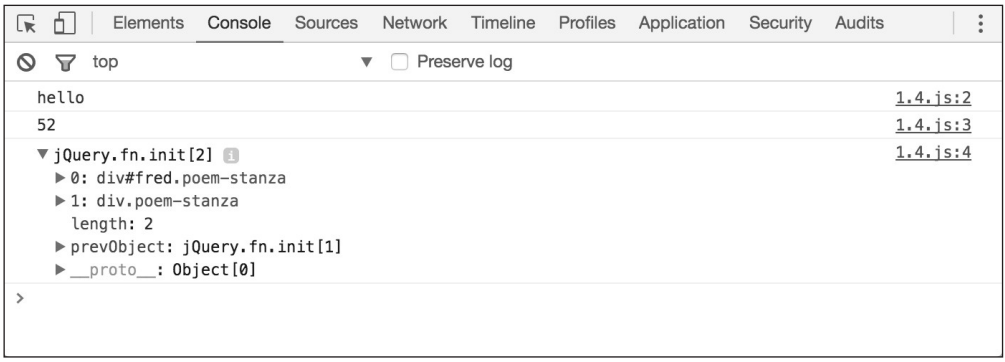



Osim toga, možemo da vršimo interakciju sa ovom konzolom direktno iz koda, koristeći metod `console.log()`:

```
$(() => { console.log('hello');
  console.log(52);
  console.log($('div.poem-stanza'));
});
```

Programski kod 1.4

U ovom kodu ilustrovano je da možemo da prosledimo bilo koju vrstu izraza u metod `console.log()`. Jednostavne vrednosti, kao što su znakovni nizovi i brojevi, odštampane su direktno, a komplikovanije vrednosti, kao što su jQuery objekti, lepo su formatirane za ispitivanje:




Ova funkcija `console.log()`, koja funkcioniše u programerskim alatkama svih pretraživača koje smo ranije pomenuli, predstavlja odgovarajuću alternativu za JavaScript funkciju alert`()` i biće veoma korisna prilikom testiranja jQuery koda.

# REZIME

U ovom poglavlju ste naučili kako da učinite jQuery dostupnim u JavaScript kodu na veb stranici, da koristite funkciju `$()` za lociranje dela stranice koja ima određenu klasu, da pozovete funkciju `.addClass()` da biste primenili stil za ovaj deo stranice i da pozovete funkciju `$(() => {})` da bi bila izvršena prilikom učitavanja stranice. Takođe smo istražili programerske alatke koje se koriste za pisanje, testiranje i ispravljanje grešaka u jQuery kodu.

Sada razumete zašto bi programer radije izabrao da koristi JavaScript radni okvir, umesto da piše ceo kod od nule, čak i za najosnovnije zadatke. Takođe ste videli neke od načina isticanja jQuerya kao radnog okvira, razloge zašto bismo ga izabrali umesto drugih opcija, i generalno, zadatke koje jQuery olakšava.

U jednostavnom primeru koji smo upotrebili prikazano je kako funkcioniše jQuery, ali ovaj primer nije mnogo koristan u stvarnim situacijama. U sledećem poglavlju ćemo proširiti kod istraživanjem sofisticiranog jezika selektora jQuerya, tražeći praktičnu upotrebu za ovu tehniku.