Primljen: 24.10.2023.
Prihvaćen: 08.12.2023.

Stručni rad
UDK: 004.738.5: 004.438

Automatizacija prikupljanja podataka uz pomoć alata Selenium

Data collection automation with the help of Selenium

[1]Denis Jelaš, [2]Petar Olujić, [3]Karlo Leder
[1,2,3]Visoka škola za menadžment i dizajn Aspira, Domovinskog rata 65, 21000 Split
e-mail: [1]denis.jelas@aspira.hr, [2]petar.olujic@aspira.hr, [3]karlo.leder@aspira.hr

Sažetak: *Automatizacija prikupljanja podataka vrlo je relevantna tema jer su podatci najvrjedniji resurs današnjice te je zanimljivo prikazati kako se može automatizirati njihovo prikupljanje s interneta. Selenium je jedan od najčešćih alata koji se danas koristi za automatizaciju web preglednika u svrhu prikupljanja podataka stoga je on odabran da bi se precizirala tema prikupljanja podataka s internetskih stranica (engl. web scraping) i fokus stavio na točno određeno područje.*

U radu se općenito govori o automatizaciji web preglednika za prikupljanje podataka, povijesti, tehnikama, tehnologijama i pravnim aspektima. Na kraju sljedi praktični primjer uporabe programa za prikupljanje podataka uz pomoć alata Selenium s web stranice Hrvatske akademske i istraživačke mreže (CARNet). Program je razvijen kao konzolna aplikacija u programskom alatu .NET koristeći programski jezik C#.

Ključne riječi: *automatizacija prikupljanja podataka, web scraping, Selenium*

Abstract: *Data collection automation is a very relevant topic because data is the most valuable resource today, and it is interesting to demonstrate how data collection from the Internet can be automated. Selenium is one of the most common tools that are used today for automating web browsers for the purpose of data collection. Therefore, it was chosen to specify the topic of data collection from web pages (web scraping) and focus on a specific area.*

The paper provides a general overview of web browser automation for data collection, covering its history, techniques, technologies, and legal aspects. Finally, a practical example of using a program with Selenium for data collection from the website of the Croatian Academic and Research Network (CARNet) is presented. The program is developed as a console application in programming tool .NET using the C# programming language.

Keywords: *data collection automation, web scraping, Selenium*

1. Uvod

Automatizacija prikupljanja podataka odnosi se na prikupljanje ili izvlačenje podataka s web stranice. Podatci se prikupljaju i zatim izvoze u format koji je pogodan za korisnika, bilo da se radi o proračunskoj tablici, datoteci formata JSON, bazi podataka ili aplikacijskom sučelju (API) (engl. *Application Programming Interface*).

Cilj ovoga rada pružiti je prikaz automatskoga prikupljanja podataka koristeći alat za prikupljanje Selenium u kombinaciji s web preglednikom Chrome. U nastavku će se prvo raščlaniti općenito postupak automatskoga prikupljanja i implementacija, korištene tehnike i alati. Nadalje, detaljnije će se putem praktičnoga primjera, koristeći konzolni program u programskom alatu .NET, pokazati prikupljanje podataka s web stranice Hrvatske akademske i istraživačke mreže (CARNet). Na kraju će se dati završna razmatranja i zaključci.

2. Automatizirano prikupljanje podataka

Tehnika parsiranja internetskih stranica (engl. *web scraping*) (Chapagain, 2023.) metoda je izvlačenja podataka s web stranica izravno preko protokola HTTP (engl. *Hypertext Transfer Protocol*) ili preko internet preglednika. Iako se može izvoditi i ručno, ovaj pojam uglavnom se odnosi na automatsko prikupljanje podataka koji se mogu organizirati i spremati unutar baza podataka, tablica ili uz pomoć drugih alata.

Prikupljanje podataka igra važnu ulogu u opskrbi podatcima za modele strojnoga učenja, što dalje potiče napredak tehnologije umjetne inteligencije. (Choudhary et al., 2021.) Na primjer, skupljanje slika s web stranica može opskrbljivati algoritme računalnoga vida, tekstualni podatci mogu se koristiti za modele obrade prirodnoga jezika (Chopra et al., 2020.), a podatci o ponašanju korisnika prilikom posjeta web stranicama mogu poboljšati način i sustav preporuke različitih proizvoda i usluga.

2.1. Povijest

Korijeni parsiranja internetskih stranica počinju od prvih programa za indeksiranje i pretraživanje web stranica (engl. *web crawler*), odnosno, preteča pretraživača. Prvi takvi alati napravljeni 1993. godine nazvani su The Wanderer i Jumpstation. (Britannica, 2023) Sljedeći veći iskorak se javlja 2004. godine kada je implementiran HTML (engl. *Hypertext Markup Language*) parser nazvan BeautifulSoup. (BeautifulSoup, 2023) Alat je napisan koristeći programski jezik Python te je dohvaćao sadržaj unutar HTML elemenata. Od tada, s razvojem novih i efikasnijih programskih jezika, te pogotovo razvojem umjetne inteligencije (engl. AI – *Artificial Intelligence*), mogućnosti parsiranja internetskih stranica se sve više šire. (Web Scraper, 2023)

2.2. Tehnike

Postoje razne tehnike prikupljanja podataka s internetskih stranica. Najpoznatije i najraširenije tehnike su:

- **ručno parsiranje web stranica –** najjednostavnija, ali i najsporija metoda koja se oslanja na poznatu metodu kopiranja i lijepljenja (engl. *copy & paste*). Korisna je za malu količinu informacija, za veće količine problem je monotonost i uzima previše vremena.
- **dohvaćanje teksta i regularni izrazi –** pristup baziran na korištenju UNIX naredbi ili regularnih izraza.
- **pregled koda pomoću stvaranja modela na temelju dokumenata –** struktura web stranice izvlači se s pomoću parsera za DOM (engl. *Document Object Model*). Metoda je jako efikasna za dinamičke stranice jer one sadrže čvorove i spremnike s podatcima koji su potrebni. Obično je potrebno korištenje dodatnih alata, kao što je XPath.
- **semantičko prepoznavanje anotacija –** stranice koje su predviđene za automatsko prikupljanje podataka mogu sadržavati određene metapodatke ili dodatna objašnjenja za

dolazak do korisniku zanimljivih podataka. Ako su anotacije uključene unutar web stranice, moguće je stvaranje semantičkoga sloja uputa i *shema* podataka kojima web scraper može pristupiti prije nego započne sam proces prikupljanja. (Aydin, 2018.)

2.3. Zaštita podataka

Legalnost prikupljanja podataka s internetskih stranica je i dalje u "sivoj zoni" korištenja, odnosno, postojeći zakoni se ne odnose izravno na samu tehniku. Korištenje bilo koje tehnike je legalno ako se radi o javnim podatcima kojima je ionako cilj da budu javno dostupni svim korisnicima. Problem može nastati ako se dođe do određenih podataka koji se smatraju privatnim ili intelektualnim vlasništvom. To se može odnositi na podatke o fizičkim osobama jednako kao i na podatke o organizacijama i tvrtkama. Unutar uvjeta korištenja (engl. *Terms of use*) web stranica je u novije doba sve češće definirano smije li se na toj stranici koristiti tehnika izvlačenja podataka i, ako da, u kolikoj mjeri. Etički ispravno bi bilo koristiti izvlačenje za javno dostupne podatke, podatke koji će se iskoristiti za određeno znanstveno istraživanje ili za stjecanje određenih poslovnih uvida i znanja. (Krotov i Silva, 2018.)

2.4. Metode za sprječavanje izvlačenja podataka

Većina web stranica koje sadrže korisne informacije nema problem da se iskoristi neka tehnika za prikupljanje informacija. Problem nastaje kada u pitanje dođe sigurnost podataka koji ne bi trebali biti dostupni javnosti. Sigurnost je glavni razlog zašto pojedine stranice koriste razne vrste zaštite od izvlačenja podataka kao što su:

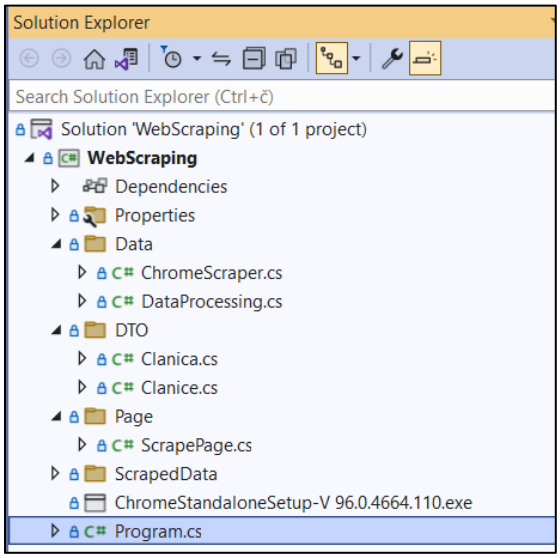
- **ograničenje brzine –** odnosi se na ograničavanje količine zahtjeva koje klijent može u nekom vremenskom intervalu poslati na poslužitelj.
- **prepoznavanje automatiziranoga prometa –** automatizirani promet može se identificirati pronalaženjem neprirodnih (neljudskih) obrazaca u web prometu.
- **prepoznavanje zlonamjernih identifikatora –** postoje mnoge web stranice koje sadrže i održavaju popis poznatih zlonamjernih IP adresa. Zahtjeve s takvih stranica potrebno je prepoznati i odbiti.
- **mamljenje (engl. honey-potting) –** tehnika zaštite u kojoj se na stranicu postavljaju određene poveznice ili gumbovi koji nisu vidljivi običnim korisnicima nego samo *bot*-ovima. Tako se vrlo brzo može utvrditi da se radi o neželjenom ponašanju na stranici. (Miraz et al., 2021).

3. Primjer uporabe programa WebScraping za prikupljanje podataka uz pomoć alata Selenium

U sljedećem primjeru bit će prikazan program pod nazivom *WebScraping* koji služi za prikupljanje podataka s web stranice Hrvatske akademske i istraživačke mreže (CARNet) (*https://www.carnet.hr/popis-ustanova-clanica/)* pri čemu su navedene članice poredane po kategorijama kao na slici 2.

Izvorni kod programa s dodatcima nalazi se na stranici GitHub (GitHub, 2023), a stablo mapa je prikazano na slici 1.

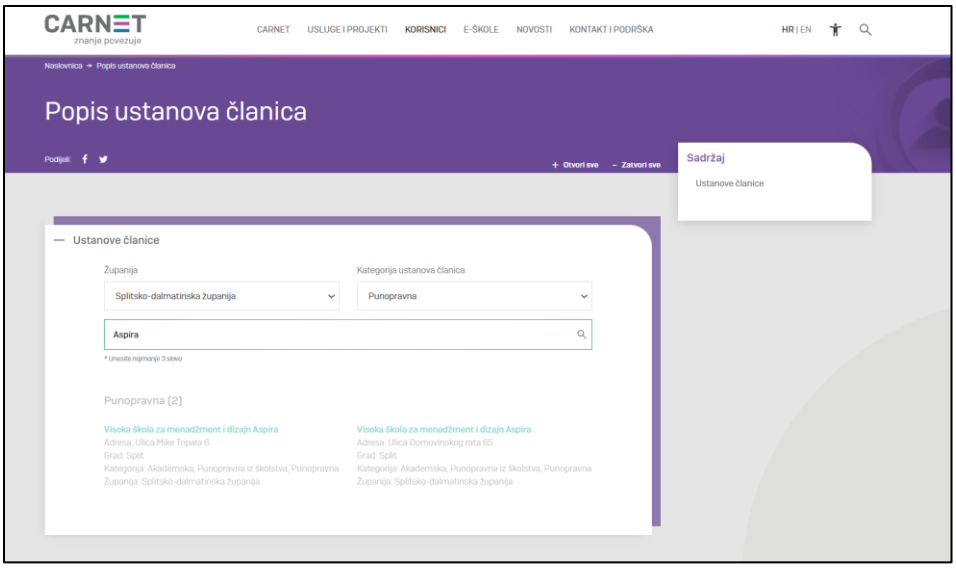
Slika 1. *Visual Studio – Stablo mapa programa WebScraping*



Izvor: autor

Općeniti dio programa se nalazi u mapi *Data* u kojoj se nalaze metode koje omogućuju pokretanje i interakciju s alatom ChromeDriver kao i obradu podataka nakon prikupljanja. Ovaj dio programa mogu koristiti sve web stranice. Mape *DTO* i *Page* namijenjene su za opis podataka i automatizaciju specifične stranice s koje se podatci prikupljaju. Primjeri datoteka s prikupljenim podatcima nalaze se u mapi *ScrapedData*.

Slika 2. *CARNet – popis ustanova članica – primjer „Visoka škola za menadžment i dizajn Aspira“*



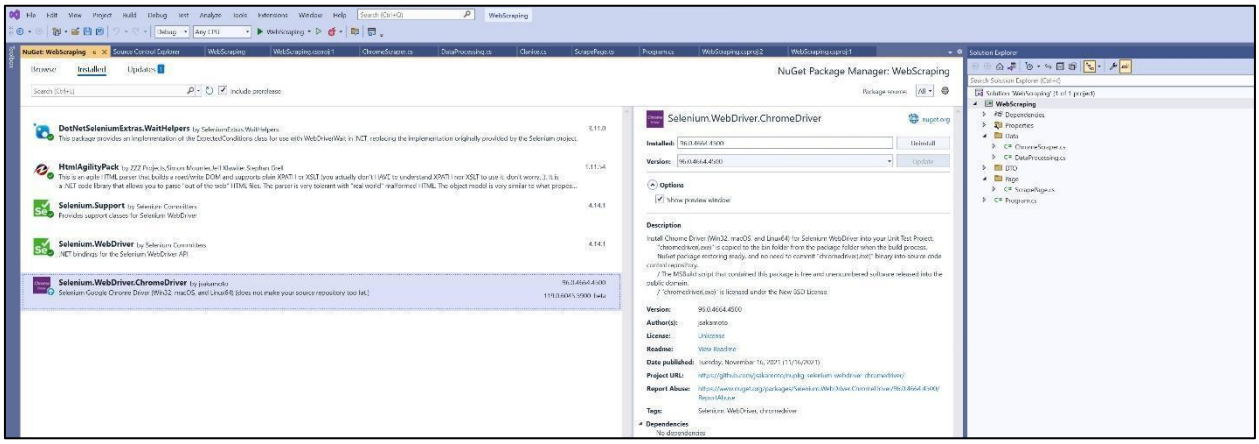
Izvor: https://www.carnet.hr/popis-ustanova-clanica/

Program je napravljen kao konzolna aplikacija za operacijski sustav Windows u programu Visual Studio koristeći programski jezik C#. Podatci se prikupljaju koristeći web preglednik Chrome koji se automatizira uz pomoć alata ChromeDriver i klase WebDriver alata Selenium. Popis ispod pokazuje NuGet pakete prikazane na slici 3 koji su instalirani u programu:

1. **DotNetSeleniumExtras.WaitHelpers** - implementacija klase ExpectedConditions za upotrebu klase WebDriverWait u programskom alatu .NET.
2. **HtmlAgilityPack** - agilan HTML parser koji kreira DOM za čitanje i pisanje te podržava jezike XPATH ili XSLT.
3. **Selenium.Support** - sadrži pomoćne alate i klase za podršku pri upotrebi klase WebDriver alata Selenium.
4. **Selenium.WebDriver** - skup različitih softverskih alata koji sadrže mnoge opcije za pronalaženje i manipulaciju elementima unutar preglednika, a jedna od ključnih značajki je podrška za automatizaciju više platformi preglednika.
5. **Selenium.WebDriver.ChromeDriver** - neovisni poslužitelj koji implementira W3C WebDriver standard za Android, Mac, Linux, Windows i ChromeOS.

Preduvjet korištenja programa je postojanje web preglednika Chrome na računalu u kojem se program izvodi, pa je potrebno da verzija alata ChromeDriver (96.0.4664.4500) bude ista kao i verzija preglednika Chrome stoga je potrebno deinstalirati trenutačnu verziju te instalirati verziju koja se nalazi na stranici GitHub (ChromeStandaloneSetup-V 96.0.4664.110.exe).

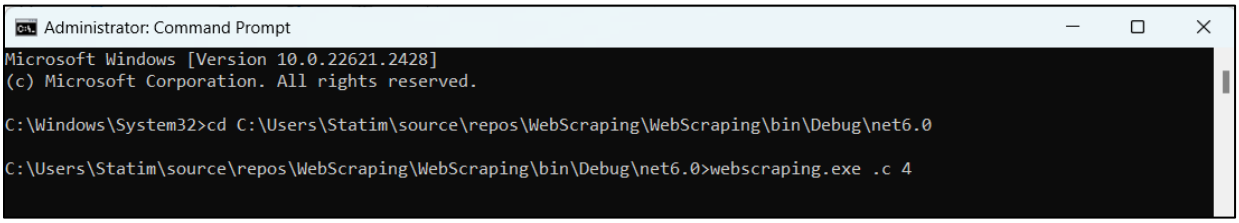
Slika 3. *Instalirani NuGet paketi*



Izvor: autor

Testni program *WebScraping* napravljen je kao konzolna aplikacija za operacijski sustav Windows te se pokreće kao na slici 4 izravno iz programa Visual Studio ili koristeći Command Prompt, poznat i kao cmd.exe ili cmd, zadani interpretator naredbenoga retka (engl. *Command-line*) za Windows operacijske sustave.

Slika 4. *Pokretanje programa WebScraping iz programa Command Prompt*



Izvor: autor

Pri pokretanju programa mogu se odabrati razne kategorije podataka koje će se prikupljati, a definirane su brojem koji dolazi nakon naredbe –c. U prikazanom primjeru program će prikupiti sve podatke o članicama koji imaju kategoriju „Akademska“ jer je zadan broj kategorije 4.

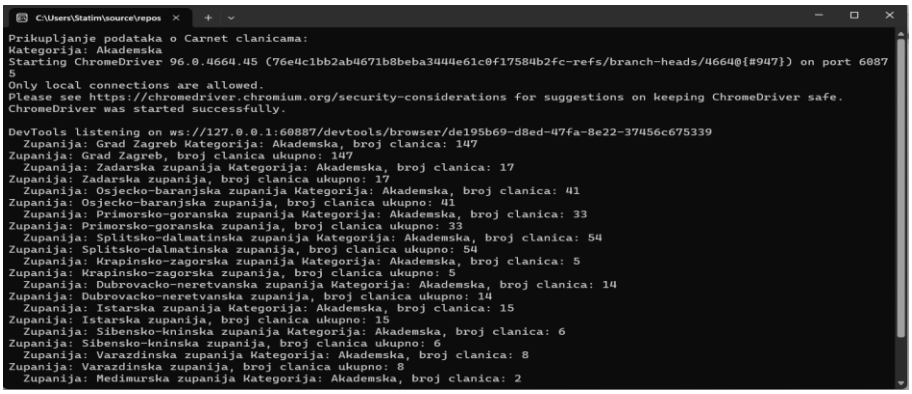
WebScraping.exe –c 4

Podržane su sljedeće kategorije:

1. „Sve"
2. „Punopravna"
3. „Punopravna iz školstva"
4. „Akademska"
5. „Pridružena"
6. „Privremena"

Nakon pokretanja programa vrši se pokretanje web preglednika Chrome i navigacija na određenu stranicu CARNeta te automatizirano navođenje na zadane kategorije prikupljanja za sve županije. Vrijeme izvođenja programa ovisi o odabranoj kategoriji te može trajati od 1 minute za kategoriju „Privremena“ pod rednim brojem 6 (3 članice) do 30 minuta za kategoriju „Sve“ pod rednim brojem 1 (7067 članica). U toku izvođenja se u konzolnom prozoru ispisuju prikupljeni podaci po kategorijama i županijama kao na slici 5, a u konačnici se svi podatci pohranjuju u datoteku JSON formata.

Slika 5: *Ispis podataka u konzolnom prozoru*



Izvor: autor

Ovaj primjer predstavlja jednostavnu uporabu alata Selenium za automatiziranje web preglednika Chrome, a moguće je automatizirati i sve ostale web preglednike. (Vanden Broucke i Baesens, 2018.)

4. Zaključak

Moderne web aplikacije često sadrže velike količine podataka za čiju ručnu obradu i pregled u suvremenom svijetu ima se sve manje vremena. To se posebno odnosi na poslovne web aplikacije kod kojih je veća vjerojatnost da sadrže podatke i informacije koje mogu biti od koristi za donošenje smislenih zaključaka i eventualno stjecanje određenih poslovnih uvida. Automatizirano prikupljanje podataka pomoću alata Selenium jedan je od najboljih načina za implementaciju izvlačenja podataka kako bi do potrebnih podataka došli što brže i efikasnije.

Program *WebScraping* prikazan u radu služi kao primjer prikupljanja podataka sa samo jedne internetske stranice, ali se koristeći istu programsku logiku može napraviti program koji bi prikupljao podatke s neke druge stranice. Pri tome treba prilagoditi program HTML strukturi web stranice i podatcima koji se na njoj nalaze. Moguća je nadogradnja programa u kojem bi se internetske stranice s kojih se žele prikupljati podatci, opisale koristeći XML jezik i na takav način kreirati općenito programsko rješenje koje ne bi bilo posvećeno samo jednoj stranici. Parsiranje internetskih stranica često nije jednostavan zadatak jer one dolaze u različitim oblicima i podložne su promjenama.

Prikazano je prikupljanje podataka izravno iz HTML strukture, ali uz pomoć alata Selenium moguće je izvesti i prikupljanje skidanjem (engl. *download*) datoteka ili izravno iz aplikacijskoga sučelja (API) koji su najstabilniji izvor podataka, naročito ako su službeni i dobro dokumentirani.

Litreratura

1. Chapagain A. (2023). Web scraping with Python. 2. izd. Birmingham, Packt Publishing Ltd.
2. Choudhary A.; Agrawal A. P.; Unhelkar B.; Logeswaran R. (2021). Applications of Artificial Intelligence and Machine Learning. Springer Nature Singapore.
3. Chopra R.; Godbole A. M.; Sadvilkar N.; Shah M. B.; Ghosh S.; Gunning D. (2020). The Natural Language Processing Workshop. Packt Publishing.
4. Britannica, Search Engine, https://www.britannica.com/technology/search-engine#ref1307556 (7.12,2023.)
5. BeautifulSoup, Beautiful Soup Documentation, https://www.crummy.com/software/BeautifulSoup/bs4/doc/ (7.12.2023.)
6. Web Scraper, Brief history of web scraping, https://webscraper.io/blog/brief-history-of-web-scraping (29. 11. 2023.)
7. Aydin O. (2018). R Web Scraping Quick Start Guide. Birmingham, Packt Publishing Ltd.
8. Krotov V.; Silva L. (2018). Legality and Ethics of Web Scraping. New Orleans, Twenty-fourth Americas Conference on Information Systems.
9. Miraz M. H.; Southall G.; Maaruf A.; Ware A.; Soomro S. (2021). Emerging Technologies in Computing. International Conference, iCETiC Virtual Event. Springer.
10. GitHub, Izvorni kod programa s dodacima na repozitoriju GitHub, https://github.com/aspiragithub/WebScrapingProject (29.11.2023.)
11. Vanden Broucke S.; Baesens B. (2018). Practical Web Scraping for Data Science. Apress.