VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INFORMAČNÍCH SYSTÉMŮ
DEPARTMENT OF INFORMATION SYSTEMS

METÓDY EXTRAKCIE DÁT Z WEBOVÝCH STRÁNOK

METHODS OF DATA EXTRACTION FROM THE WEB

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **LUKÁŠ PERINA**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. RADEK BURGET, Ph.D.**
SUPERVISOR

BRNO 2021

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav informačních systémů (UIFS) Akademický rok 2020/2021

Zadání bakalářské práce

Student: **Perina Lukáš**
Program: Informační technologie
Název: **Metody extrakce dat z webových stránek**
Methods of Data Extraction from the Web
Kategorie: Web

Zadání:

1. Seznamte se se současnými serverovými i klientskými technologiemi pro implementaci webových aplikací v JavaScriptu.
2. Prostudujte současné přístupy pro extrakci dat (scraping) z webových dokumentů.
3. Navrhněte architekturu aplikace pro extrakci dat z webových stránek na základě předem definovaných pravidel. Svá rozhodnutí konzultujte s vedoucím.
4. Implementujte navrženou aplikaci pomocí vhodných technologií.
5. Proveďte vyhodnocení funkčnosti vytvořené aplikace na vhodné sadě webových dokumentů.
6. Zhodnoťte dosažené výsledky.

Literatura:

- Alarte, J.; Insa, D.; Silva, J.; et al.: Main Content Extraction from Heterogeneous Webpages. In Web Information Systems Engineering - WISE 2018. Cham: Springer International Publishing. 2018. ISBN 978-3-030-02922-7. pp. 393-407.
- Burget, R.: Model-Based Integration of Unstructured Web Data Sources Using Graph Representation of Document Contents. In: 15th International Conference on Web Information Systems and Technologies. Vienna: SciTePress - Science and Technology Publications, 2019, s. 326-333. ISBN 978-989-758-386-5.

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 až 3

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Burget Radek, Ing., Ph.D.**
Vedoucí ústavu: Kolář Dušan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2020
Datum odevzdání: 12. května 2021
Datum schválení: 22. října 2020

Abstrakt

Cieľom tejto bakalárskej práce je návrh architektúry a následná implementácia aplikácie určenej na extrakciu dát (web scraping) z webových dokumentov. Narozdiel od konvenčných metód sa jedná o extrakciu založenú na definovaní dátových typov a regulárnych výrazov hľadaných prvkov tak, aby nebolo potrebné poznať detailnú štruktúru daného webového dokumentu, a možnosť použitia jednej definície na detekciu hľadaných prvkov na rôznych webových stránkach. Algoritmus dosahuje priemernú presnosť 83.06% a recall 78.15%. Týmto prístupom sa umožní zredukovať čas potrebný na analýzu jednotlivých stránok na minimum, a nebrať štruktúru kódu ako určujúci faktor pri vytváraní požiadavkov na webscraping.

Abstract

The purpose of this bachelor thesis is to design an architecture and subsequent implementation of an application designed for data extraction (web scraping) from web documents. Unlike conventional methods, it is an extraction based on defining data types and regular expressions of requested elements so that it is not necessary to know the detailed structure of given web document, and the possibility of using just one definition to detect requested elements on different web pages. Algorithm is able to achieve overall accuraccy of 83.06% and recall 78.15%. This approach can reduce the time required for analysis significantly, and not to take the structure of the code as a determining factor in creating webscraping requests.

Klíčová slova

Web scraping, Javascript, Node.js, Google Chrome, Json, Extrakcia dát, scraping, web, DOM, CSS, HTML

Keywords

Web scraping, Javascript, Node.js, Google Chrome, Json, data extraction, scraping, web, DOM, CSS, HTML

Citace

PERINA, Lukáš. *Metódy extrakcie dát z webových stránok*. Brno, 2021. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Radek Burget, Ph.D.

Metódy extrakcie dát z webových stránok

Prohlášení

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne pod vedením pána Ing. Radka Burgeta Ph.D. Uviedol som všetky literárne pramene, publikácie a ďalšie zdroje, z ktorých som čerpal.

.....
Lukáš Perina
1. dubna 2021

Poděkování

Moje poďakovanie patrí vedúcemu práce pánovi Ing. Radkovi Burgetovi Ph.D. za dôsledné vedenie bakalárskej práce a konzultácie ktoré mi pomohli pri práci a určili ten správny smer.

Obsah

Kapitola 1

Úvod

Cieľom tejto bakalárskej práce je návrh architektúry a následná implementácia aplikácie určenej na extrakciu dát z webových stránok. V dnešnej dobe je pojem extrakcia dát používaný stále častejšie v spojitosti práve s webovými technológiami a webom samotným práve preto, že sa internet ako taký, a hlavne jeho obsah v podobe webových stránok, neustále rozširuje. Objem dát ktoré sa na webe nachádza sa zväčšuje neúprosnou rýchlosťou, a preto je zber údajov z rôznych webstránok stále zložitejší, a to nielen čo sa zložitosti štruktúry webu týka, ale vzhľadom na veľký počet webov sa zvyšuje aj časová náročnosť analýzy a následnej extrakcie dát.

Práve pre spomínanú expanziu internetu sa tejto téme venuje čoraz viac spoločností so snahou uľahčiť prístup k web scrapingu pre každodenných užívateľov, ale aj rôznych korporácií ktoré by o extrakciu dát z webových stránok mohli mať záujem. Pre neustáli vývoj webu je avšak nutné vyvíjať aj aplikácie, ktoré sú na extrakciu určené.

Pri rozlišovaní takýchto aplikácií sa berie na vedomie hlavne požadovaný vstupný formát a jeho zložitosť, požadovaný výstupný formát a jeho využiteľnosť v praxi, a v neposlednom rade rýchlosť aplikácie z pohľadu extrakcie dát.

Táto práca je zameraná hlavne na časť týkajúcu sa požadovaného vstupu aplikácie, kde je za cieľ požadovaný jednotný vstup vo forme Json súboru pozostávajúci z adries webových stránok, dátových typov, rozloženia objektov na webstránke, a požadované názvy extrahovaných informácií. Aplikácia sa následne pozerá na každú stránku zvlášť a určí výsledné údaje na základe poskytnutých relevantných informácií o štruktúre. Týmto spôsobom je možné definovať jeden vstup raz, a použiť ho na viaceré webové dokumenty bez nutnosti ďalšieho zásahu.

Prvá časť práce sa v kapitole 2 zaoberá aktuálnym prístupom k problematike, ako sa web scraping využíva v súčasnosti, aké sú štandardy a dostupné technológie, a porovnanie týchto prístupov z hľadiska využiteľnosti.

V druhej časti tejto práce sú v kapitole 3 popísané technológie ktoré je treba brať v úvahu pri návrhu, tvorbe a analýze aplikácie určenej na extrakciu dát. Jedná sa hlavne o technológie ktoré sa stali základným kameňom navrhnutej aplikácie, a o technológie na ktorých aplikácia stavia.

Kapitola 4 sa zameriava na analýzu a návrh architektúry aplikácie. Je založená na analýze špecifikácie a požiadavkov aplikácie, a popisuje postup návrhu aplikácie.

V 5. kapitole sú popísané jednotlivé detaily implementácie riešenia, a hlavná logika aplikácie.

6. kapitola sa následne venuje testovaní aplikácie za pomoci niekoľkých datasetov, a následnej analýze a vyhodnotení výsledkov ktoré aplikácia dosiahla.

Kapitola 2

Aktuálny prístup k problematike Web Scrapingu

V tejto kapitole sú uvedené a popísané aktuálne prístupy ktoré sa používajú na extrakciu dát z webových dokumentov. Takáto extrakcia môže prebiehať online alebo offline, v závislosti na tom aké dáta a za akým cieľom chceme extrahovať.

Web scrapingu ako takému sa v súčasnosti venuje stále viac a viac spoločností, a preto nieje prekvapením že sa rozširujú nielen možnosti web scrapingových aplikácií, ale aj prístup takýchto aplikácií k priamej extrakcii dát.

Zároveň sa rozširuje pole pôsobnosti, a možnosti využitia takto extrahovaných dát. Preto je táto kapitola určená hlavne rozboru týchto metód. Poznatky získané z tohto rozboru zároveň pomôžu určiť smer ktorým sa návrh a vývoj aplikácie môžu uberať.

2.1 Web scraping

Web scraping je jednou z foriem získavania údajov z webových stránok. Takéto získavanie údajov je možné buď priamo - World Wide Web za pomoci Hypertext Transfer Protokolu, alebo pomocou webového prehliadača (napr. Google Chrome). Za web scraping sa dá považovať aj manuálny zber dát za využitia ľudskej sily, ale všeobecne sa tým myslí využitie dedikovaného počítača ktorý túto prácu automatizuje. Tu sa môže jednať napríklad o jednoduché kopírovanie dát, alebo zložitejšie generovanie výstupných štruktúr napríklad vo formáte JSON. [12]

Automatizované získavanie informácií z webu sa dostalo do vývoja krátko po zavedení World Wide Web. Kvôli neustálemu vývoju technológií je však vo vývoji doteraz, a jeho možnosti sa neustále rozširujú. Prvé formy automatizovaných web scraperov boli určené primárne pre tvorbu databázy vyhľadávacieho indexu pre vývoj World Wide Web Vyhľadávače. Za jeden z prvých takýchto scraperov je preto považovaný World Wide Web Wanderer, ktorý bol vyvinutý v roku 1993 za účelom zmerania veľkosti celého internetu, a neskôr na indexáciu celého internetu. [12]

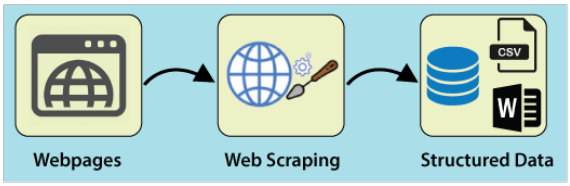
2.2 Súčasné postupy

Existujú rôzne prístupy k danej problematike, kde rozdiely sú jednoznačné hlavne pri spôsobe a type analýzy a extrakcie finálnych dát. Hlavným cieľom pri tvorbe aplikácie na takúto extrakciu dát je užívateľská prívetivosť, a jednoduchosť definovania vstupných požiadavkov.

Postup ktorý aplikácie na extrakciu dát využívajú je prevažne rovnaký, a skladá sa z troch primárnych krokov:

1. Odoslanie GET požiadavku na webový server a následné obdržanie odpovedi
2. Analýza extrahovaného HTML kódu na základe stromovej štruktúry
3. Použitie zvoleného postupu na extrakciu dát a spracovanie hľadaného obsahu

Tieto kroky sú potom pri každej aplikácií mierne prispôsobené určeniu a typu implementácie. [7]



Obrázek 2.1: Všeobecný postup ktorým sa aplikácia typu web scraper riadi. Prevzaté z Javatpoint [4]

Medzi najpopulárnejšie web scrapery patria aplikácie s jednoduchou point-and-click politikou, kde používateľ po inštalácií takejto aplikácie jednoducho zvolí webové dokumenty z ktorých chce extrakciu prevádzať, a následne interaktívne vyznačí dáta ktoré ho zaujímajú. Pre užívateľov ktorý nemajú príliš veľké skúsenosti s informačnými technológiami a nechcú platiť za túto službu nemalé peniaze rôznym korporáciám je to skrátka jediná možnosť.

Spomínaný prístup má samozrejme svoje výhody, medzi ktoré sa radí najmä spomínaná užívateľská prívetivosť a relatívne nízka cena, avšak v prípade potreby automatizácie takejto činnosti existujú na trhu lepšie riešenia. V prípade že je potrebná vysoká automatizácia, alebo sa jedná o veľké množstvo dát a stránok ktoré sú na extrakciu určené, je výhodnejšie použiť aplikáciu ktorá je ladená skôr na tento spôsob. Pri firmách ktoré sa venujú extrahovaniu údajov z webových stránok na profesionálnej úrovni sa počet prenesených dát v súčasnosti pohybuje už v petabajtoch, a počet extrahovaných webových stránok sa pohybuje v miliardách mesačne na jednu takúto firmu.

Pri takto veľkých číslach narážajú spomínané firmy na rôzne problémy, ktoré rádový používateľ web scrapingovej aplikácie riešiť nemusí. Medzi takéto problémy sa radia v neposlednom rade napríklad:

- Náročnosť na hardvérový čas
- Limitovanie počtu možných GET dotazov
- Veľká diverzifikácia architektúry webových stránok
- Objem dát potrebný na prenesenie údajov a následné uloženie extrahovaných údajov

Príklad ako sa takýmto problémom brániť alebo im priam predchádzať je v prípade limitovania počtu možných GET dotazov na jednu webovú stránku meniť IP adresy z ktorých sú GET dotazy odosielané. Ani to však nie vždy môže byť riešením, keďže stránky

v dnešnej dobe môžu využívať rôzne ochrany proti web scrapingu, ako napríklad priamu detekciu človeka od robota alebo tzv. CAPTCHA[1] .[13]

Web scraping má teda svoje výhody, ale aj nevýhody. Tieto sú popísané podrobnejšie v nasledujúcej časti tejto kapitoly.

2.3 Výhody, nevýhody, a využitie web scrapingu

Web scraping má v súčasnosti viac možností a oblastí využitia ako tomu bolo v minulosti. Toto je dané hlavne neustálim rozširovaním sa internetu ako celku. V počiatkoch internetu sa aplikácie typu web scraper nazývali skôr crawler, a využívali prevažne na indexáciu obsahu, a nie na zber, extrakciu a analýzu dát tak ako je tomu teraz.

Web scraper je ale aplikácia ktorá je založená presne na tých fundamentálnych základoch ako spomínaný crawler, s jediným rozdielom a to tým, že jeho úlohou je zber dát. Takýto zber môže prebiehať napríklad kopírovaním dát do databázy. V mnohých prípadoch sú aplikácie typu scraper využívané veľkými korporáciami ako napríklad Microsoft, Amazon alebo Google na cielenie zobrazovaných reklám tak, aby boli pre užívateľa ako jednotlivca relevantné.[7]

Jednotlivé prípady využitia takejto aplikácie sa neustále rozširujú, obecne ale platí že jeho využitie spadá do nasledovných kategórií:

- Monitorovanie ceny produktov
- Získavanie údajov o nových produktoch na trhu
- Analýza konkurencie
- Evidovanie údajov o konkrétnej doméne podľa druhu záujmu
- Monitorovanie ceny leteniek

A mnohé ďalšie prípady. Údaje extrahované touto metódou môžu byť následne využité na širšie pochopenie vývoja udalostí napríklad na trhu, alebo u konkurencie.

Web scraping má teda jasné výhody čo sa kolekcie dát týka. Patria medzi ne už spomínané monitorovanie konkurencie, cien produktov a analýza rôznych dát. V neposlednom rade ale treba zmieniť aj výhody z pohľadu technológie samotnej. Medzi ne patria napríklad:

- Rýchlo sa rozvíjajúce technológie
- Automatizovanie monotónnych činností
- Zber obsiahleho množstva dát, ktorý by bol manuálne nemožný
- V porovnaní s manuálnym zberom dát exponenciálne rýchlejšie výsledky

Každá technológia a jej využitie má však aj svoje nevýhody. V porovnaní s manuálnym zberom dát, ktoré môže byť rozumné ak sa jedná o ojedinelú prípadne unikátnu udalosť, kedy je takýto zber potrebné vykonať jeden krát za dlhé obdobie, sa tvorba alebo priame využitie web scrapera očakáva až v prípade že sa daná akcia musí vykonávať automatizovane a v určitých časových intervaloch. [4]

[1] CAPTCHA - Completely Automated Public Turing test to tell Computers and Humans Apart.

V takom prípade je potreba zvážiť aj rôzne prvky pri nielen prevádzke, ale aj samotnom vytváraní a následnom používaní web scrapera. Medzi najhlavnejšie nevýhody sa preto radí najmä:

- Znalosť kódovania a programovania
- IP Detekcia a CAPTCHA
- Ku každej stránke je potreba pristupovať individuálne

Zároveň treba spomenúť fakt, že dynamické stránky môžu meniť svoju štruktúru, a preto nemôžme zabúdať na údržbu web scrapera nielen z pohľadu úrovne bezpečnosti programu, ale aj z pohľadu údajov ktoré web scraper prijíma. V prípade že webová stránka zmení štruktúru tak, že aktuálna konfigurácia web scrapera nedokáže pokračovať v extrahovaní údajov tak ako predtým, je potrebné v mnohých prípadoch stránku znova analyzovať, a v prípade že je web scraper neschopný efektívne využiť algoritmy ktoré v minulosti fungovali, v niektorých prípadoch aj dodatočne zanalyzovať prístupy web scrapera a upraviť jeho logiku tak, aby mohol v extrahovaní pokračovať.

2.4 Metódy využívané v súčasnosti

V prípade že nás zaujíma len priama extrakcia dát, v dnešnej dobe je na výber z mnohých aplikácií ktoré sú pripravené na použitie, a tak nieje treba programovať aplikácie znova. Každá má však svoje vlastné postupy pri extrakcii, a z toho plynie fakt že nie každá takáto aplikácia dokáže splniť požadované zadanie. Medzi najznámejšie aplikácie v tomto odvetví však patria hlavne:

- Scrapping-bot
- Octoparse
- Import.io
- Dexi.io
- Outwit

V niektorých prípadoch nám stačí jednoduché rozšírenie do prehliadača (napr. Google Chrome) ktoré sú efektívne hlavne pri menšom počte webstránok určených na scraping. [4]

Pokiaľ však chceme vytvoriť vlastný program alebo aplikáciu typu web scraper, je potrebné rozlišovať na úrovni technológií dostupných pri programovaní. V prvom rade je treba vziať do úvahy programovací jazyk. V takom prípade je na výber rovno z niekoľko volne dostupných programovacích jazykov. Medzi dva najpopulárnejšie však patria hlavne Python, a Node.js. Každý má však znova svoje výhody a nevýhody, a preto je treba zvážiť na aké účely bude web scraper používaný. Vo všeobecnosti je populárnejší práve prvý spomínaný Python, vďaka jeho popularite z pohľadu jednoduchosti písania čistého kódu, komunity, a podpory ktorá je mu venovaná. Python ponúka veľké množstvo frameworkov, z ktorých tie najpopulárnejšie zahŕňajú hlavne Selenium a Scrapy, ktoré som samozrejme zohľadnil pri výbere vhodného frameworku na vytvorenie web scrapera.

Na druhej strane Node.js stavia na základoch Javascriptu, ktorý je vyvíjaný priamo pre prácu s web stránkami. Kým Python je lepší čo sa týka podpory a rozšírenosti frameworkov, Javascript má výhodu v lepšej integrácií nielen s webovými stránkami ako takými, ale aj s webovými prehliadačmi samotnými. Python v mnohých prípadoch vyžaduje driver na komunikáciu s prehliadačom, a podpora dynamických stránok je znova jednoznačne lepšia pomocou Javascriptu, práve kvôli jeho spomínanej lepšej integrácií a faktom, že Javascript je jednou z hlavných súčastí dynamických moderných webových stránok. Javascript je jednou z hlavných prvkov modernej webovej stránky, ktorá sa skladá ešte z HTML a CSS. Jeho úlohou je hlavne vytváranie dynamického obsahu vytvárajúceho dojem že webová stránka s užívateľom interaguje. Vo väčšine prípadov beží iba na strane klienta, takže dynamickosť samotná nezaťažuje server. Zároveň je v mnohých prípadoch využívaná asynchrónnosť[2] tohto programovacieho jazyka, na zlepšenie odozvy a interakcie. Práve tento fakt sa stáva rozdielovým faktorom pri extrakcii dát z webových stránok, keďže obsah webovej stránky sa v mnohých prípadoch môže načítavať asynchrónne a nezávisle od ostatných častí webu. Z môjho vlastného rozboru a výskumu som zistil, že dynamickosť a asynchrónnosť stránok je prvok ktorý je lepšie zvládaný za pomoci Javascriptu. [8][6]

Rozhodnutie používať Javascript som učinil zároveň s predpokladom využitia jedného z jeho volne dostupných frameworkov, a zároveň som zohľadnil svoje predošlé skúsenosti práve so spomínaným jazykom. K tomuto rozhodnutiu prispel aj fakt, že v dnešnej dobe je dynamika stránok takmer štandard, a tu hrá úloha Javascriptu veľkú rolu. V procese výberu frameworku som zohľadňoval hlavne integráciu daného frameworku s prehliadačom, výbornú podporu dynamických stránok a asynchrónnosť ktorá je pri takejto aplikácii kľúčová.

Ako prvý framewrok spomeniem určite Apify SDK, za ktorým stojí Česká firma Apify Technologies[3].[13] Práve Apify ma totiž inšpirovalo v rôznych odvetviach vývoja. Medzi najznámejšie technológie ktoré používajú Javascript patria najmä:

- Apify SDK
- Puppeteer
- Cheerio

Z pomedzi týchto technológií ktoré zdieľajú niektoré základné funkcie bola na zostavenie výslednej aplikácie použitá technológia Puppeteer. Puppeteer je zároveň jedna z technológií využívaných práve spomínanou Českou firmou Apify.

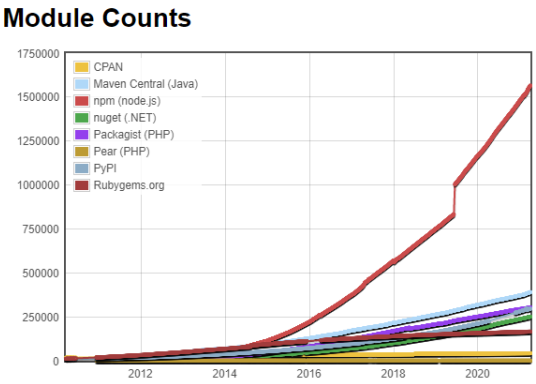
V porovnaní s technológiou Cheerio má Puppeteer svoje výhody. V niektorých faktoroch vyhráva ale aj Cheerio, a preto je treba zvážiť hlavne nasledovné:

- Možnosť scrapingu dynamických stránok
- Funkcionality a možnosti
- Kompatibilita webových technológií
- Rýchlosť analýzy a extrakcie

[2] Asynchrónnosť znamená v tomto prípade neblokovanie prehliadačových prostriedkov a načítavanie obsahu za behu webstránky.

[3] https://apify.com/

V prvom rade treba zvážiť aktuálnu situáciu a technológie ktoré sa na webe vyskytujú. 95% webových stránok v súčasnosti používa Javascript, takže predpoklad že moderné weby budú obsahovať dynamický obsah je prevažne jasný, a preto treba s takýmto obsahom počítať. Node.js ktorý je postavený na Javascripte je zároveň najrýchlejšie sa rozširujúcou technológiou ktorú web využíva. V čase písania má Node.js viac ako 1.5 milióna dostupných rozširujúcich balíčkov, a jeho náskok je v súčasnej dobe jednoznačný.[3]



Obrázek 2.2: Porovnanie najpopulárnejších technológií a vývoj počtu ich modulov. Zdroj [2]

Pri porovnávaní technológií Puppeteer a Cheerio je na prvý pohľad jednoznačné čo majú spoločné, a v čom naopak každá z nich vyniká. Tieto poznatky pomôžu jednoznačne určiť technológiu na základe požiadavkov ktoré vznikli zároveň s výberom témy tejto práce.

Cheerio a Puppeteer sú technológie založené na Node.js, a patria medzi najpopulárnejšie technológie v tomto odvetví. Ich jednoznačné vlastnosti však definujú nielen ich možné prípady použitia, ale zároveň schopnosti a rýchlosť akou sú schopné dosiahnuť očakávané výsledky.

Puppeteer je technológia ktorá je vyvíjaná za účelom automatizácie webového prehliadača, zatiaľčo hlavným určením Cheeria je web scraping samotný. Tým že je Cheerio určené priamo na požadovanú úlohu je v tomto ohľade rýchlejšie a výhodnejšie čo sa nášho požiadavku týka. Cheerio je zároveň multiplatformné, a dokáže pracovať ako s prehliadačom Firefox, tak s prehliadačom Google Chrome. Na druhú stranu Puppeteer je viazaný iba na prehliadač Google Chrome vo forme Chromium. Puppeteer má priamo kontrolu nad prehliadačom, takže je možné ovládať prehliadač ako celok. Tento fakt sa dá brať aj ako výhoda aj ako nevýhoda, keďže je to pravdepodobne jeden z dôvodov prečo je Puppeteer pomalší vo vykonávaní funkcie Web Scrapera. Puppeteer však prináša schopnosť plynulej práce s Javascriptom na webovej stránke, dokáže parsovať aj webové stránky postavené na moderných technológiách ako React alebo Angular, ktoré sú v dnešnej dobe na vzostupe pre ich moderné prvky. Z priamej integrácie s prehliadačom vyplýva aj fakt že Puppeteer dokáže vytvárať snímky obrazovky počas načítavania a pracovania s webovou stránkou. Zároveň podporuje technológiu XML, a nerobí mu problém ako čítanie, tak ani parsovanie a priame spúšťanie Javascriptových funkcií. [9]

Práve podpora dodatočných funkcií a možnosť parsovania dynamických stránok s Javascriptom ma presvedčili o rozširovaní poznatkov technológie Puppeteer, a jej následné využitie pri návrhu architektúry a implementácie výslednej aplikácie.

Kapitola 3

Technológie

Táto kapitola sa zaoberá popisom technológií, modulov a prvkov, na základe ktorých bude následne zostavená aplikácia tak, aby bolo čo najefektívnejšie nielen jej využívanie, ale aj údržba. Zároveň treba brať v úvahu aj prívetivosť a jednoduchosť užívateľského vstupu, a formát, využiteľnosť a prenositeľnosť výstupných dát.

K výberu týchto technológií boli využité poznatky z Kapitoly 2, ktoré ďalej určovali smer návrhu a vývoja aplikácie.

3.1 HTML

Základným stavebným blokom každej webovej stránky je nepochybne HTML. HTML je skratka pre *Hypertext Markup Language*, a už svojím názvom napovedá akú úlohu vo webových technológiách zohráva.

HTML dokument sa skladá z elementov, ktoré sú predstavené v podobe takzvaných *tagov*. Tieto tagy sú reprezentované pomocou znakov <> a dovoľujú určovať typ a rozsah uvedeného obsahu. Tieto elementy môžu obsahovať zároveň rôzne ďalšie atribúty, ako napríklad trieda elementu, alebo identifikátor. Takéto atribúty sú potom využívané v technológiách ktoré s HTML dokumentami spolupracujú, ako napríklad CSS a Javascript.

Takéto dokumenty sú následne reprezentované vo webovom prehliadači, ktorý HTML dokumenty zobrazí zároveň s doplnkovými dokumentami ktoré určujú štylizáciu a správanie sa danej webstránky. HTML dokumenty samotné neobsahujú žiadnu štylizáciu, avšak typ elementu ktorý použijeme pre realizovaní určitého prvku je kritický, pretože udáva základné rozloženie obsahu ktorý element predstavuje.[11]

3.2 DOM

Jazyk HTML nieje určený na programovacie účely, a preto je potrebné pri práci s webovými stránkami zvážiť aj technológiu DOM. DOM je skratka ktorá reprezentuje *Document Object Model*. Predstavuje akési prepojenie medzi HTML a programovacím jazykom ktorý chce s webovou stránkou komunikovať. [5]

Pri komunikácií s programovacím jazykom DOM reprezentuje stránku v dvoch hlavných podobách:

- Uzly
- Objekty

Práve tento fakt nám ďalej dovoľuje s webovou stránkou interagovať a meniť jej obsah. Príkladom takéhoto využitia jazyk Javascript, ktorý bol vyvinutý za účelom vytvorenia dojmu interakcie užívateľa s webovou stránkou.

DOM je zároveň veľmi dôležitý pri web scrapingu, keďže práve komunikácia s webovou stránkou na úrovni programovacieho jazyka je kľúčová pri extrahovaní požadovaných údajov z webových dokumentov. Takáto extrakcia je možná vďaka reprezentácií webovej stránky v objektovo orientovanej podobe.

3.3 CSS

CSS predstavuje istý jazyk, ktorý je používaný za účelom popisu štylizácie HTML dokumentov. CSS znamená *Cascading Style Sheets*, kde Cascading predstavuje jedno z hlavným pravidiel tohto jazyka - štylizačné pravidlá majú určenú prioritu podľa ktorej budú aplikované, a zároveň umožňujú na jednej webovej stránke používať viacero CSS súborov.

Primárne určenie je už spomínaná štylizácia a definovanie rozloženia elementov na webovej stránke. Zároveň môžu definovať vzhľad v závislosti od zariadenia ktoré webovú stránku v danom momente navštívilo.

HTML samotné nebolo vyvinuté za účelom definovania vzhľadu, a preto bolo nutné vyvinúť technológiu ktorá tento nedostatok kompenzovala. CSS pravidlá sú zväčša aplikované na atribút triedy, ktorý bližšie špecifikuje skupinu prvkov v HTML dokumente. V niektorých prípadoch sa však pravdilá aplikujú aj na celé elementy, prípadne identifikátory.[10]

Pri vytváraní CSS pravidiel rozlišujeme 3 hlavné miesta kde sa tieto pravidlá môžu nachádzať:

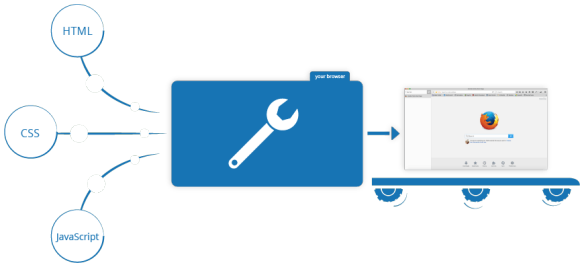
- Riadkové
- Interné
- Externé

Rozlišovanie medzi týmito typmi je dôležité, pretože z názvu *Cascading* vyplýva že každé miesto môže mať inú prioritu finálneho vykonania daného pravidla.

3.4 Javascript

Programovací jazyk Javascript bol vyvinutý za účelom vytvárania dojmu interakcie pri webových stránkach. Javascript je jednou z hlavných súčastí webových stránok, a je používaný na viac ako 95% stránkach celosvetovo.[3] Dá sa preto predpokladať že bude jeho integrácia s webovými prehliadačmi a webovými stránkami na vysokej úrovni.

Práve kombinácia HTML, CSS a Javascriptu totiž dokáže docieliť výsledky ktoré sa pri moderných štandardoch vyžadujú. Či už sa jedná o dynamickú zmenu obsahu alebo animovanie niektorých elementov, je to práve Javascript ktorý túto interakciu docieľuje. [6]



Obrázek 3.1: HTML, CSS a JavaScript sú 3 hlavné zložky ktoré spolu vytvárajú web ako ho poznáme dnes. Zdroj [6]

Javascript je založený na programovacom jazyku Java, ktorý bol ďalej modifikovaný a prispôsobený na prácu priamo vo webovom prehliadači. Javascript dokáže nielen reagovať na vstupy od užívateľa, ale reagovať aj na niektoré udalosti ktoré sa stanú na pozadí, v prehliadači. [6]

Táto technológia ďalej stavia aj na asynchrónnosti, teda prístupu za behu. To dovoľuje stránkam načítať Javascript až po načítaní webového obsahu tak, aby nevznikali zbytočné dlhé čakacie doby na načítanie webstránky. Zároveň sa dá asynchrónnosť využiť pri odosielaní formulárov, kde nieje nutné po odoslaní znova načítavať stránku. Tým sa zároveň zlepšuje responzívnosť, keďže nieje nutné čakať na vykonanie špecifickej akcie.

Jednou z najpopulárnejších vlastností tohto programovacieho jazyka je podpora API[1]. Podpora API umožňuje využitie naprogramovaných rozhraní bez toho aby si tieto rozhrania užívateľ programoval sám. Tento fakt urýchľuje vývoj webových aplikácií a pomáha udržiavať štandard. API pri Javascripte sa primárne rozdelujú do dvoch kategórií:

- Prehliadačové API
- Externé API

Kde prehliadačové API predstavuje napríklad aj spomínaný DOM, ktorý javascriptu poskytuje HTML elementy vo forme objektov, s ktorými javascript dokáže pracovať. Čo sa externých API týka, sem patria najmä API tretích strán, ako napríklad Google Maps alebo Twitter.

V mnohých prípadoch sú to práve API, ktoré môžu naraziť na väčšie množstvo problémov pri načítavaní webovej stránky. Prispieva k tomu aj asynchronita, kde v prípade že sa HTML dokument načíta skôr ako Javascriptové súbory, môže nastať problém kde vykonávanie skriptu zlyhá. Asynchronita zároveň súvisí s postupnosťou vykonávania kódu. V prípade že je využitá asynchrónna funkcia, jej výsledok je reprezentovaný v podobe *Promise*. Výsledok Promise nemusí byť dostupný okamžite, a preto v prípadoch že sa na túto skutočnosť neberie ohľad môže vykonávanie funkcie zlyhať.[6]

[1] Application Programming Interfaces

3.4.1 JSON

Objekty sú v Javascripte popísané pomocou formátu JSON. JSON je skratka pre *JavaScript Object Notation* kde už z názvu vyplýva jeho pôvod. Je to jednoduchá forma popisu objektov tak, že je vo výsledku ľahko čitateľná človekom, a zároveň jednoducho parsovatelná počítačom.

Aj keď je súčasťou názvu *Javascript*, JSON je univerzálny spôsob zápisu objektov a je to formát adaptovaný mnohými programovacími jazykmi súčasnosti. Práve tieto vlastnosti robia z tohto formátu multiplatformný spôsob prenosu informácií.[1]

JSON ponúka dve základné štruktúry:

- Objekt
- Pole

Tieto základné štruktúry sú podporované modernými prehliadačmi a programovacími jazykmi ktoré sa v tomto zmysle používajú.

3.4.2 Node.js

Literatura

[1] *JSON* [https://www.json.org/json-en.html]. (Accessed on 04/01/2021).

[2] *Modulecounts* [http://www.modulecounts.com/]. (Accessed on 03/31/2021).

[3] Elliott, E. *How Popular is JavaScript in 2019? | by Eric Elliott | JavaScript Scene | Medium* [https://medium.com/javascript-scene/how-popular-is-javascript-in-2019-823712f7c4b1]. May 2019. (Accessed on 03/31/2021).

[4] Javatpoint. *Web Scraping Using Python - Javatpoint* [https://www.javatpoint.com/web-scraping-using-python]. (Accessed on 03/31/2021).

[5] mozilla. *Introduction to the DOM - Web APIs | MDN* [https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Introduction]. (Accessed on 04/01/2021).

[6] mozilla. *What is JavaScript? - Learn web development | MDN* [https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript]. (Accessed on 03/31/2021).

[7] Octoparse. *What is Web Scraping and How Does It Work | Octoparse* [https://www.octoparse.com/blog/web-scraping-introduction#]. October 2018. (Accessed on 03/30/2021).

[8] prowebscraper. *The 5 Best Programming Languages for Web Scraping – ProWebScraper* [https://prowebscraper.com/blog/best-programming-language-for-web-scraping/]. (Accessed on 03/31/2021).

[9] tarun007. *Difference between cheerio and puppeteer - GeeksforGeeks* [https://www.geeksforgeeks.org/difference-between-cheerio-and-puppeteer/]. July 2020. (Accessed on 03/31/2021).

[10] w3schools. *CSS Introduction* [https://www.w3schools.com/css/css_intro.asp]. (Accessed on 04/01/2021).

[11] w3schools. *What is HTML* [https://www.w3schools.com/whatis/whatis_html.asp]. (Accessed on 04/01/2021).

[12] Wikipedia. *Web scraping — Wikipedia, The Free Encyclopedia* [http://en.wikipedia.org/w/index.php?title=Web%20scraping&oldid=1014546619]. 2021. [Online; accessed 30-March-2021].

[13] Wolf, K. *Jan Čurn (Apify): Z webu stahujeme už miliardu stránek měsíčně - Lupa.cz* [https://www.lupa.cz/clanky/jan-curn-apify-z-webu-stahujeme-uz-miliardu-stranek-mesicne/]. January 2020. (Accessed on 03/30/2021).