Pannon Egyetem

Műszaki Informatikai Kar

Rendszer- és Számítástudományi Tanszék

programtervező informatikus BSc

SZAKDOLGOZAT

Kotlin web scraping fejlesztése

Radó Bence

Témavezető: Machalik Károly

2020

Alulírott Radó Bence hallgató, kijelentem, hogy a dolgozatot a Pannon Egyetem Rendszer- és Számítástudományi Tanszék tanszékén készítettem a programtervező informatikus végzettség megszerzése érdekében.

Kijelentem, hogy a dolgozatban lévő érdemi rész saját munkám eredménye, az érdemi részen kívül csak a hivatkozott forrásokat (szakirodalom, eszközök, stb.) használtam fel.

Tudomásul veszem, hogy a dolgozatban foglalt eredményeket a Pannon Egyetem, valamint a feladatot kiíró szervezeti egység saját céljaira szabadon felhasználhatja.

Veszprém, 2020. november 1.

aláírás

Alulírott Machalik Károly témavezető kijelentem, hogy a dolgozatot Radó Bence a Pannon Egyetem Rendszer- és Számítástudományi Tanszék tanszékén készítette programtervező informatikus végzettség megszerzése érdekében.

Kijelentem, hogy a dolgozat védésre bocsátását engedélyezem.

Veszprém, 2020 november 1.

aláírás

Köszönetet szeretnék nyilvánítani Machalik Károlynak, aki végig segítette munkámat, illetve szakmai tanácsokkal látott el, a Pannon Egyetem Tanárainak akiktől a megfelelő tudást elsajátíthattam az évek során, továbbá Édesanyámnak és Édesapámnak akik támogattak tanulmányaim során és bíztak bennem.

**TARTALMI ÖSSZEFOGLALÓ**

A dolgozatom témája, amit választottam: Kotlin web scraping fejlesztése. A web scraping egy gyors és könnyű módja különböző adatok kinyerésének különböző weboldalakról. Nagyon sokféle web scraping program létezik, viszont mindegyiknek meg van az előnye, illetve a hátránya is. Témám során a Kotlin nyelvet használtam, és arra törekedtem, hogy a legkönnyebben használható és testreszabható programot készítsek el ami mindenki számára könnyen használható. [TODO]

**Kulcsszavak:** Web, Kotlin, Scraping, Adat

**TARTALOMJEGYZÉK**

[TODO]

**TARTALMI RÉSZ**

**Bevezetés**

A feladat címe, amit választottam a következő: Kotlin web scraping fejlesztése. Ez annyit jelent, hogy Kotlin nyelven kell megvalósítani az adott programot, ami jelen esetben egy web scraping alkalmazás. Először a Kotlin nyelvről szeretnék picit írni. A Kotlin erősen típusos, objektumorientált programozási nyelv. Szentpétervár közelében lévő Kotlin-szigetről nevezték el. 2011-ben hozták nyilvánosságra a nyelv létezését. A Java nyelvvel szokták összehasonlítani, azonban több nyelv is hatással volt rá a kifejlesztése során, mint például a Scala, Groovy, C# illetve még a Gosu is. Több platformra is lefordítható maga a kód, többek között Java Virtual Machine-re (JVM) is, ebből kifolyólag a Java-val teljes mértékben kompatibilis a Kotlin, továbbá könnyebben tanulható, mivel egyszerűbb a szintaxisa.

A web scraping-et tulajdonképpen arra használják, hogy különböző weboldalakról különböző adatokat nyernek/gyűjtenek ki, hogy ezeket rendszerezetten egy helyen tárolják, majd később fel tudják használni különböző tevékenységekhez. Nevezik még web harvesting illetve web data extraction-nek is. Ezt a folyamatot úgy kell elképzelni, mintha egy személy felmenne egy weboldalra és kézzel kimásolgatna különböző adatokat és lementené magának. A web scraping-gel, rengeteg időt és sok fáradalmat lehet megspórolni az automatizáció miatt. Léteznek olyan web scraping programok, amelyekhez szükséges külső interakció, vagyis a felhasználónak különböző paramétereket kell megadnia az alkalmazásnak, például, hogy milyen típusú adatokat keressen a rendszer az adott weboldalon. Azonban léteznek olyanok is, amik teljesen automatizáltan gyűjtik ki az adatokat és a háttérben futnak. Továbbá léteznek ingyenesen használhatóak, illetve olyanok is amelyekért fizetni kell.

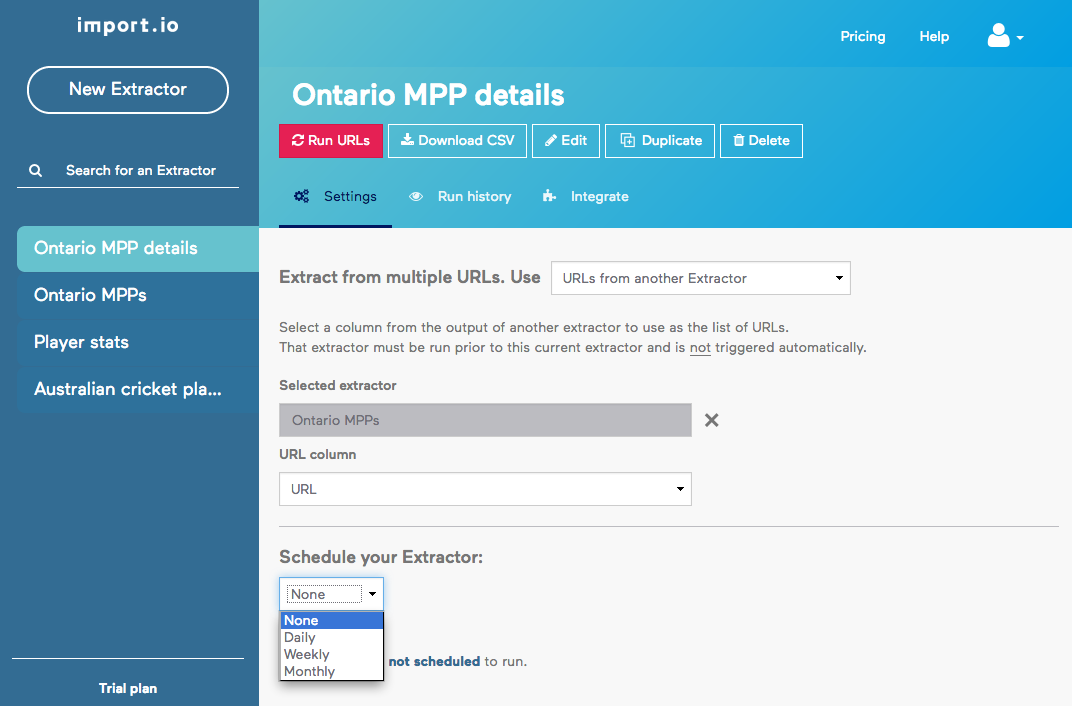
Egy picit arról is írnék, hogy miért is választottam ezt a témát. A mobil programozás mindig is foglalkoztatott, ennek kapcsán elkezdett érdekelni a Java nyelv, mivel az Android alkalmazások jelentős részében ezt használták a fejlesztők. Pár éve a Google a Kotlin nyelv mellé állt, és az elsődlegesen támogatott nyelv az Android operációs rendszerek esetében így már nem a Java. Ez volt a fő oka annak, hogy ezt a témát választottam. A másik pedig, hogy ki szerettem volna próbálni magam egy tőlem elég távol álló területen, mint például a webbel kapcsolatos problémák megoldása. A szakdolgozatom végére azt szeretném elérni, hogy egy olyan programot készítsek, ami ingyenesen elérhető bárki számára és könnyen használható. Egy nagyon letisztult GUI-t képzelek el a program számára, hogy minél könnyebben bele tudjanak jönni a felhasználók a használatába. A legnagyobb probléma, amivel találkoztam a különböző web scraping programok tanulmányozása során, az az volt, hogy nem voltak testreszabhatóak, ezt úgy értem, hogy nagyrészük csak egy adott funkciót tudott. Tovább jelentős probléma, hogy az általam vizsgált rendszerek egy adott ID-vel rendelkező mezőnek az azonosítóját gyűjtötték ki és használták a későbbiekben, azonban egy dinamikusan változó rendszerben ez nem konstans, ezért nem is használható. Az én programomban úgy próbálom kiküszöbölni ezeket a problémákat, hogy testreszabható legyen a program. Ezt úgy értem, hogy meg lehet majd adni, hogy milyen típusú, azonosítójú mezőket, illetve milyen reguláris kifejezésekre illeszkedő elemekre keressen rá a program és azok alapján gyűjtsön adatokat. Olyan funkciót is szeretnék a programba, aminek segítségével exportálni lehet a kigyűjtött adatokat egy Excel fájlba, hogy azokat rendszerezve egy helyen lássa a felhasználó, hogy majd később könnyebben fel lehessen használni azokat.

**Kutatás**

Mindenképpen szeretnék említést tenni a kutatásaimról, amiket szakdolgozatom elkészítése során végeztem. Ebben a részben szeretnék említést tenni 5 különböző web scraping programról, amelyekről picit bővebben is írok. Ennek során megismerhetővé válnak az egyes alkalmazások előnyei, illetve hátrányai, hogy melyik miben jobb, illetve rosszabb a másiknál. Ennek a fejezetnek a végén kifejtem, hogy miben jobb szerintem az én web scraping alkalmazásom. A 4 program amiket be fogok mutatni részletesebben a következők: Import.io, OutWit Hub, Beautiful Soup és végül a Scrapy kerül ismertetésre.

**Import.io:**

Az céget 2012-ben alapította egy három főből álló kis csapat, melynek tagjai: Matthew Painter, Andrew Fogg és David White. Az Import.io egy Software as a Service (SaaS) Web Data Integration (WDI) platform, amely lehetővé teszi a felhasználók számára, hogy strukturáltalan webes adatokat konvertáljanak egy strukturált formátumba, amelyet később a piacon az elemző szektor tud felhasználni. Legtöbbször üzleti, marketing, esetleg értékesítéssel foglalkozó alkalmazások szokták használni az Import.io által nyújtott szolgáltatásokat. Az Import.io egy vizuális felületet biztosít, amely megkönnyíti az adatok kinyerését, illetve átformálását. Erről a felhasználói felületről szeretnék egy képet is megmutatni, hogy könnyebben el lehessen képzelni, hogy valójában hogyan is néz ki egy ilyen GUI.



Ezen a képen szerintem egyből szembetűnő, hogy nem igazán egyszerű, letisztult az alkalmazás. Lehet, hogy funkcionalitást tekintve gazdag, viszont nem átlátható. A saját alkalmazásomban ez egy fontos szempont volt a fejlesztés során.

Szeretném megközelíteni az alkalmazást anyagi szempontból is. Ezeket az adatokat nem a hivatalos honlapon találtam, mivel ahhoz szükséges egy előzetes konzultáció, amely során felmérik a megrendelő igényeit, azaz, hogy milyen adatokat és hány weblapról szeretné az információkat kinyerni. Ennek alapján ajánlanak egy csomagot a leendő felhasználónak. Viszont amit nem hivatalos forrásokból találtam, az a következő: egy kezdő számára a legkedvezőbb „csomag”, az egy próbaverzió. Ez korlátozott adatmennyiség kinyerésére teljesen megfelel, viszont egy nagyvállalati cég számára nem elegendő, ennek megfelelően azonban ingyenes. Ezen felül léteznek további csomagok is. Lehet választani 299$-ért havi előfizetést, illetve éves tagságot is 1999$-ért. Ez jelenlegi árfolyamon 91.000 Ft, illetve 609.000 Ft. Személyes véleményem szerint egy kisebb funkcionalitással bíró alkalmazás több ér ha nincsen használati díja, mint egy magas költségű, ami több funkcióval rendelkezik.

**OutWit Hub:**

2010-ben jelent meg a szoftver első verziója. A jelenleg is frissnek számító verziója a 9-es, 2020 nyarán jelent meg. Az OutWit Hub egy Web Data Extraction szoftver, amely szintén online weboldalakról való adatok kinyerését szolgálja. Ez a program azonban képes linkek, dokumentumok, képek, kontaktok, ismétlődő szavak, RSS hírcsatornák, illetve adott kifejezések gyűjtésére is. Ezeket a kinyert adatokat formázott táblákba tárolja el, majd ezek tovább exportálhatók például Excel táblázatba, vagy akár adatbázisokba is. Ez az alkalmazás két féle formában érhető el bárki számára: létezik egy önálló alkalmazás, ami maga a program, illetve egy Mozilla Firefox bővítmény, amellyel ugyan azok a funkciók érhetők el, mintha letöltöttük volna a programot.

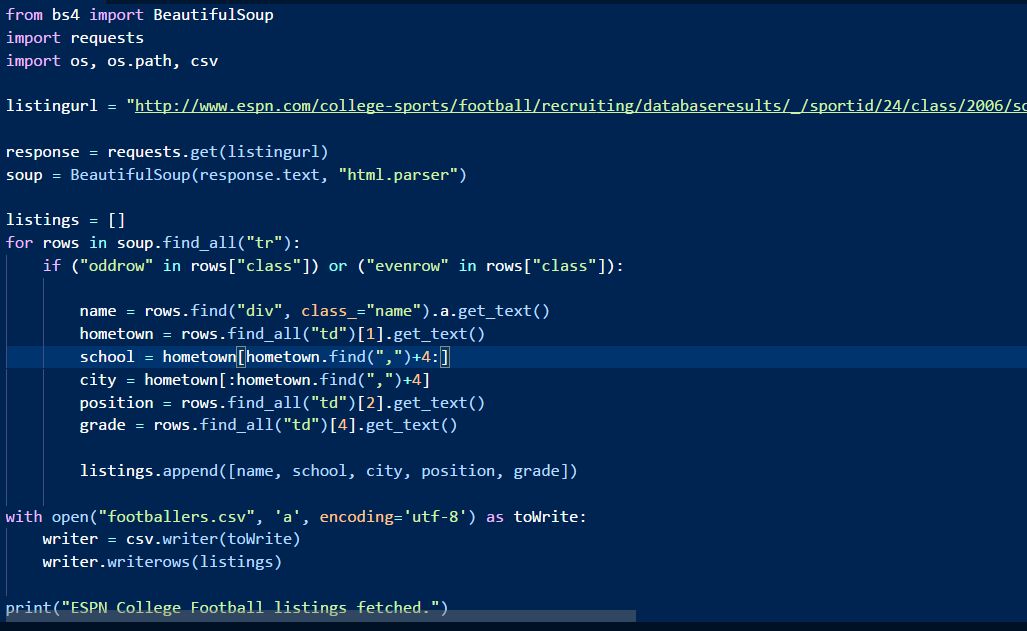
További tulajdonságokról is szeretnék említést tenni a programmal kapcsolatban. Képes email címeket felismerni, automatikus lekérdezések illetve URL címeket generálni, adott időközönként különböző feladatokat végrehajtani, és ami szerintem a legpozitívabb az OutWit Hub-bal kapcsolatban az az, hogy létezik egy „Custom Scrapers” szolgáltatás, ami annyit jelent, hogy mi összegyűjtjük, hogy milyen funkciókra van szükségünk, ezt egy konzultáció során ismertetjük a céggel. Ezek után elkészítik a számunkra optimális „Scraper”-t, majd ezt mi le tudjuk tesztelni, hogy valóban azokat a funkciókat tudja-e, amit mi kértünk. Amennyiben nem vagyunk elégedettek a validáció során, nem vagyunk kötelesek megtéríteni a szolgáltatásuk árát. Ez különben az általunk kért feladat komplexitásának függvényében változik. Kétszáz dollártól indul ez a szolgáltatás és akár többezer dollárba is kerülhet.

Ennél a programnál is elérhető egy ingyenes verzió, ami a tulajdonos honlapjáról letölthető, azonban ennél a verziónál, csak minimális funkciók érhetőek el. Létező továbbá „Pro”, „Expert” és „Enterprise” előfizetés is. Ebben a sorrendben egyre gazdagabb funkcionalitást kapunk, egyre magasabb árakon. A legdrágább előfizetés, amit a program kínál az „Enterprise” csomag, ami nem kevesebb, mint nettó 8900€, ami azt jelenti, hogy ehhez az összeghez még hozzá kell adnunk a különféle adónemeket is. A végösszeg több, mint hárommillió forint. Az én véleményem szerint ez már csak valóban az IT óriásoknak, illetve a piacvezető cégeknek érheti meg.

**Beautiful Soup:**

Először itt is általános információkról szeretnék írni. A Beautiful Soup egy package, amely Python nyelven készült. 2004-ben vált elérhetővé, Leonard Richardson volt az eredeti megalkotó. Azóta számos verzió jelent meg, a jelenleg is stabil a 4.9.1-es verzió, ami 2020. májusában jelent meg. XML, illetve HTML elemzésére használják, ennek a menete a következő: elemző fákat készít elemzett oldalaknak, ezáltal adatokat lehet kinyerni a HTML oldalról.

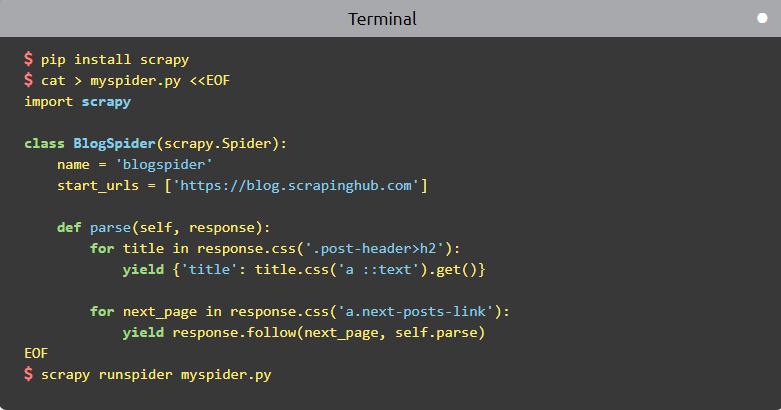
A technológiáról is említenék pár szót. Ahhoz, hogy élvezhessük ennek a csomagnak az előnyeit, legalább minimálisan ismernünk kell a Python nyelvet, vagy képben kell lennünk a programozás világában. Amit egy nagy hátránynak tartok, az az, hogy sajnos nincsen külön grafikus felhasználói felület, amit a laikus felhasználók is könnyen, illetve hamar el tudnának sajátítani. A következő paranccsal telepíthető a Beautiful Soup legújabb verziója: „pip install beautifulsoup4”. Az alább látható is egy példakód a használatára:



Azonban pont a grafikus felhasználói felület hiánya miatt, nem kell fizetünk semmit, a könyvtár használatáért. Ez nagyon fontos szempont lehet, ha egy, akár pár fős csapatról beszélünk felhasználó alatt, akiknek jelentősen kisebb az anyagi háttere, mint a nagyobb cégeknek. Amennyiben valaki nincs jelen az IT világában és szeretne valamilyen adatkinyerő módszert találni és nem riad meg a programozástól, annak jó választás lehet a Beautiful Soup. Aki picit jártasabb a kódolásban, de nem ismeri a Python nyelvet, annak sem kell megijednie, hiszen a Python nyelv elég könnyen elsajátítható.

**Scrapy:**

A Scrapy egy nyílt forráskódú, ingyenes web scraping keretrendszer, ami Python nyelven készült. 2008. június 26-án jelent meg az első kiadása. Jelenleg a Scrapinghub Ltd., ami egy web scraping és szolgáltató cég, tartja karban a Scrapy-t, számos más közreműködő segítségével. Számos jól ismert cég használja ezt a szolgáltatást, mint például a Lyst, Parse.ly, Sayone Technologies, és a Science Po Medialab is. Az következő paranccsal telepíthető: „pip install scrapy”. A használata nagyon sokban hasonlít a Beautiful Soup-hoz – ebben az esetben sem tartozik grafikus felhasználói felület, amely segítené a használatot. Előnyeit, illetve a hátrányait szintén a Beautiful Soup-hoz tudnám hasonlítani, amiket kifejtettem az előző fejezetben. Viszonylag széles körben elterjedt a használata, például GutHub-on ezernyolcszázan figyelik, illetve Twitteren ötezer egyszáz követője van a hivatalos oldalnak. Az alábbi kódrészlet egy használati példát mutat be.



**Tervezés:**

Az alkalmazásom fejlesztése során nagyon sok szempontot figyelembe vettem, illetve számos személytől kértem tanácsokat, mind a funkcionalitással kapcsolatban, mind a grafikus felhasználói felületet tekintve. Voltak közöttük szakmabeliek és laikusak egyaránt. Ezen személyek tanácsai nagymértékben befolyásolták döntéseimet a tervezés során. Kaptam negatívumokat és pozitívumokat is egyaránt. A negatívumokat építő kritikaként fogtam fel, megfogadtam őket és tanultam belőlük, ez egy szélesebb rálátást adott az egész projektre és nagyban megkönnyítette a feladatomat minden egyes fázisában. Végül úgy döntöttem, hogy a főbb szempontok amiket figyelembe fogok venni a fejlesztés során a következők lesznek:

* Letisztult, könnyen kezelhető grafikus felhasználói felület. (I.)
* Díjtalan felhasználás bárki számára. (II.)
* Alacsony funkcionalitás a minimális komplexitásra törekedve. (III.)

Röviden ki szeretném fejteni mind a 3 szempontot, hogy mit is értek alattuk és miért választottam ezeket a legfőbb pilléreknek.

1. Nagyon sok alkalommal találkoztam olyan alkalmazással, aminek nagyon komplex és bonyolult volt a felhasználói felülete, ezáltal több órába, esetleg napokba tellett a program megismerése. Szerintem ez nagyban befolyásolja a felhasználókat abban, hogy hamar megunják a program használatát és keressenek egy olyat aminek a funkcionalitása hasonló, de könnyebben megtanulható annak használata. Az én alkalmazásom egyszerű, jól átlátható grafikus felhasználói felülettel fog rendelkezni, pont azért, hogy ezt a problémát kiküszöbölje.
2. Az én véleményem szerint a legbefolyásolóbb tényező bármilyen alkalmazásnál annak anyagi vonzata. Szerintem rajtam kívül nagyon sokan gondolják így, hogy inkább használnak egy egyszerűbb alkalmazást ingyen, mint egy sokkal magasabb funkcionalitással bíró programot, amiknek rendkívül magas használati díja van. A legtöbb program esetén nem elég az egyszeri megtérítés, sok olyan alkalmazást ismerek amelyekért havonta, vagy esetleg évente használati díjat számolnak fel. Az én programomban ezzel nem lesz probléma, hiszen ingyenesen elérhető lesz bárki számára.
3. Végül a harmadik szempontom az alacsony komplexitás. Ahogy a mondás is tartja: „a kevesebb néha több”, ezt úgy kell érteni ebben az esetben, hogy nem feltétlen a magas funkcionalitású, összetett alkalmazások a jobbak, hiszen ezekben rengetek hibalehetőség merülhet fel. Egy bonyolult rendszert nagyon sok idő hibamentessé tenni, a teszteléséről nem is beszélve. Amennyiben egy nagyon komplex rendszert vizsgálunk, abban az esteben nem is lehetséges a kimerítő tesztelés. Az én programomba inkább minimális funkcionalitást képzelek el, viszont azokat nagyon robusztus módon szeretném implementálni. Szélsőséges esetekben is szeretném tesztelni, ezzel kiküszöbölve az esetlegesen előjövő hibákat a végfelhasználónál.