**Pivot app dokumentáció**

****

**Készítette: Németh Gábor, Vitrai Gábor**

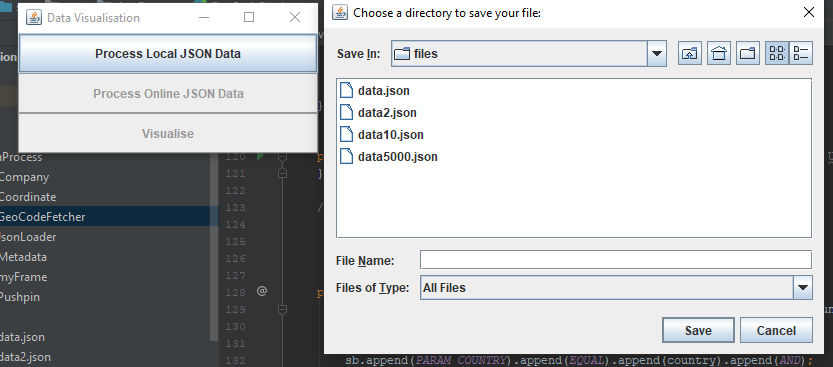
**ELTE IK Szakmai Gyakorlat - Első két hét 2019. Július**

Feladat:

2019. Júliusi ELTE-Soft Szakmai gyakorlatának az első 2 hete:

Az első két hét lényegében betanulás jellegű volt. A lényege, hogy megtanuljuk kezelni azokat az eszközöket, technológiákat amik amik a 3. héttől kezdve a nagyobb, tényleges projektet fogja alkotni.

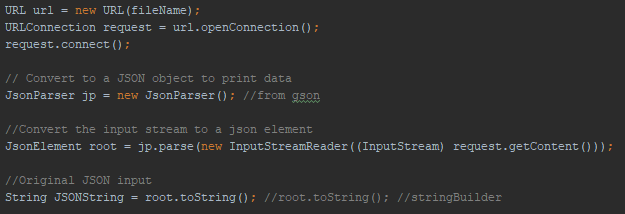
Célok:

1. Adattábla feltöltése az adatbázisba, adatok tisztítása, normalizálás.
2. Az adatbázist elérni ODATA-n keresztül, majd JSON formátumban elérni, feldolgozni. A kliensen tárolt adatokat feldolgozni, és mindegyik címhez lekérdezni a koordinátákat, majd JavaScript segítségével megjeleníteni azokat a térképen.

**Leírás/Dokumántáció:**

1. fázis: *Adatbetöltés, adattisztítás, normalizálás /Németh Gábor/*

* A kapott adattáblában az importálás előtt néhány módosítás került végrehajtásra. A társaságok hosszú és rövid nevét egységesíteni kellett, a cég állapotára vonatkozó rész eltávolításra került, a gazdálkodási forma megjelenése a rövid nevekben kizárólag rövidítésekkel, a hosszú nevekben kizárólag teljes kiírással került bele a nevet tartalmazó oszlopokba, már amelyekben szerepelt.
* Ezen kívül számos apróbb hiba került kijavításra automatizáltan, mint például a szóközök szingulárissá tétele és a sorvégi felesleges karakterek kiiktatása, néhány sor javítása pedig kézzel történt meg. Két sort törölni kellett, az egyikben az adatok nem voltak cellákra bontva, a másikban pedig a néven kívül nem szerepelt semmilyen adat. A címadatokon is elvégezhettünk volna némi egységesítést, de az app további része ezt nem igényelte.
* Ezután az adattáblát importáltuk a HANA rendszerbe, ahol előre elkészítettük ennek a sémáját.
* Töröltük azokat az adatokat, ahol a település cellában lévő érték és a címadat egymással nem voltak kompatibilisek (azaz a két település nem egyezett meg) az adatbázis egyszerűsítése végett, így nagyjából harminc sort vesztettünk el.
* Néhány azonos céghez tartozó hosszú és rövid név még mindig többször jelent meg különböző legépelés miatt, így azokból csak egyet tartottunk meg (ehhez felhasználva az adószámot, mint céget egyértelműen meghatározó értéket), a többit ezzel írtuk felül. Hasonló módon jártunk el a címekkel is, ezért is volt fontos, hogy a településsel nem kompatibilis címekkel rendelkező sorokat eltávolítsuk.
* Ezután a táblát normalizáltuk, így kaptunk tíz kisebb kódtáblát, egy névtáblát (amely tartalmazza a társaság rövid és hosszú nevét, a címét és az adószámát), és a kapcsolótáblát. Ezzel a normalizált adatbázis elkészült, amit a munkafolyamat további résztvevői már használni tudnak.



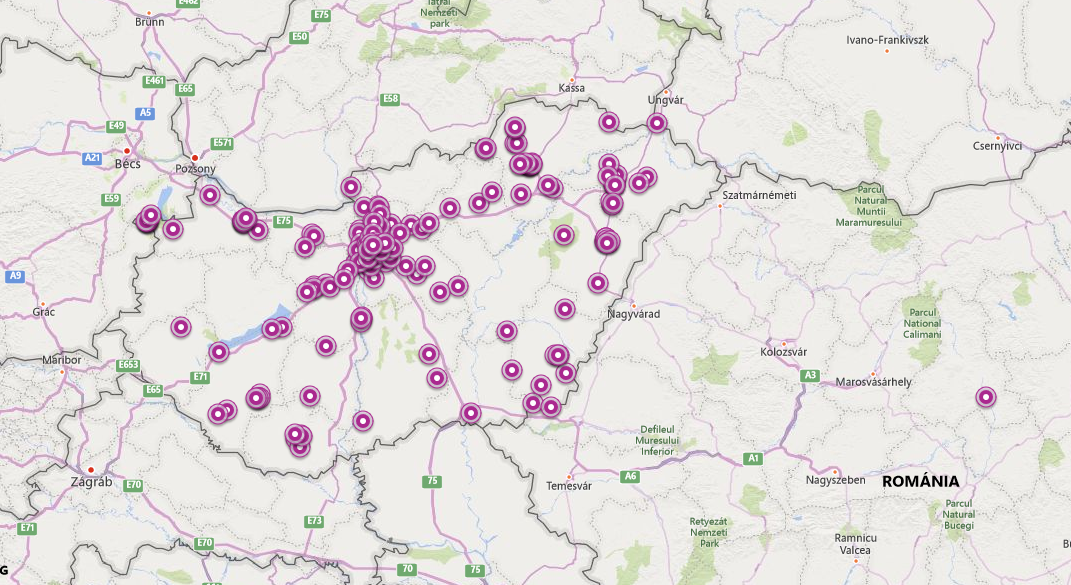
2. fázis: Adatfeldolgozás, back-end, vizualizáció: /Vitrai Gábor/

1. Az első lépés az adatbázis elérése, ez ODATA-n keresztül történik. A Az odata-t lekéri a program JSON formátumban egy String-be majd GSON könyvtárral feldolgozza JAVA oldalon. (Jelenleg jogosultsági problémákba ütköztem, mivel a SAP HANA oldal-t nem hajlandó elérni a kliens mivel nincs tanúsítványa, és a tanúsítvány elfogadtatásához rendszergazdai jogosultsághoz lenne szükségem. Mindazonáltal a kód kész, és le van tesztelve, de jelenleg lokális .json fájlból olvassa be és dolgozza fel az adatokat, amik 100%-ban megegyeznek az URL segítségével lekérhetőkhöz.)
2. A json file parse-olása után az adatok mentésre majd feldolgozásra kerülnek, egy az adattábla sémájával megegyező struktúrájú konténerben. Innen egyszerűen és gyorsan elérhetőek.
3. A későbbiekben a Bing Map API-ját használom az adatbázisból beszerzett címek megjelenítésére. Ezzel az egyetlen probléma hogy egyszerre csak 1 címre lehet manuálisan keresni, ezért úgy döntöttem hogy még Java oldalon minden címhez lekérem a hozzájuk tartozó koordinátákat és azokat csak a lekérdezés után jelenítem meg vizuálisan a térképen. Erre szolgál egy osztály számos metódussal együtt. A getCoordinates() metódus megkapja a címet paraméterben és a Bing API egy szolgáltatása segítségével visszaadja a cím koordinátáit, majd eltárolja azokat.
4. Az idő szűkössége miatt egyszerűbbnek láttam hogy külön böngésző ablakot nyissunk a térképnek. A Font-end azonban JavaScriptet használ a megjelenítéshez, így Java oldalról a program exportálja a koordinátákat (a későbbiekben terv szerint a cég egyéb adatait is) és a böngésző megnyitásakor beolvassa azokat.
5. Ekkor már gyerekjáték megjeleníteni a koordinátákat a térképen.

Tehát a program a következő sémát követi:

**SAP → HANA → ADATBÁZIS → TÁBLA → ODATA → JSON**

**→ JAVA → JS → BING API**



**How to use:**

1. Start App from IntelliJ
2. Select process option (file)
3. Wait until the procedure ends.
4. It will automatically open the browser, with the visualised data
5. (If doesn’t work: In firefox, goto “about:config” find: privacy.file\_unique\_origin and set it to false.

**Használt Linkek:**

Különféle Hana Tutorialok:

<https://developers.sap.com/tutorials/cp-java-reference-app-setup.html>

<https://developers.sap.com/tutorials/cp-node-deploy-cf-cli.html>

<https://developers.sap.com/tutorials/angular-add-javascript.html>

<https://developers.sap.com/tutorials/angular-add-odata.html>

Alapvető technológiákat részletező oldalak:

<https://www.odata.org/>

<https://angular.io/tutorial>

[https://javascript.info/](https://javascript.info/variables)

ODATA:

<https://www.odata.org/documentation/odata-version-2-0/uri-conventions/>

<https://developers.sap.com/tutorials/xsa-xsodata.html>

Bing API:

<https://trac.openstreetmap.org/browser/subversion/applications/viewer/jmapviewer/releases>

<https://docs.microsoft.com/en-us/bingmaps/v8-web-control/map-control-concepts/pushpins/pushpin-events-example>