M2: Projekt bemutató

CodeMetropolis

Összefoglaló

A csapat által elkészített változtatások a következők voltak: két új converter, az egyik a git-inspector converter, a másik a browsing history converter, illetve a meglévő UI kiegészítése. Az új funkciók megvalósítása mellett, fontosnak találtuk ezen fejlesztések megfelelő tesztelését, ezért Unit testeket írtunk saját, illetve meglévő függvényekhez, egyaránt.

A fejlesztés során GitHub-ot használtunk, a fejlesztéshez szükséges volt a CodeMetropolis repository forkolása. Mindhárom feature fejlesztése saját branch-en történt, melyeket később mergeltünk.

1. Git-inspector converter (Zakor Gyula)

1.1. Git-inspector

Statisztikai szoftver gither : https://github.com/ejwa/gitinspector.

Metrikák: insertions, commits, deletions, percentage-in-commits stb. fejlesztőnként.

Hogyan használjuk:

1. git clone
2. Belépünk a könyvtárba, ami az elemezni kívánt branchet tartalmazza.
3. `py pathTo/gitinspector.py --format=xml --since="2018.01.01" --until="2018.01.31" --file-types="java" > pathTo/output.xml`

1.2. A converter működése:

Egy inputként megadott xml fájlt beolvas, feldolgozza a benne lévő git-inspector specifikus adatokat, és egy CdfTree-t épít belőle. Ebből a CdfTree-ből a CodeMetropolis már tud egy olyan xml-t generálni, amit a mapping tool már tud használni.

A mapping tool esetén pedig meg kell adni egy külön mapping.xml fájlt, ami a git-inspector-os metrikákat hozzárendeli az egyes épülettípusok tulajdonságaihoz.

1.3. A converter használata

Ugyanúgy mint a CodeMetropolis-os convertert, elég csak a típust megváltoztatni gitinspector-ra.

```
java -jar converter.jar -t gitinspector -s output.xml
```

Ez elkészít egy converterToMapping.xml fájlt. Amit a mapping tool-al az előbb leírt mapping fájlal fel tud dolgozni.

```
java -jar mapping.jar -i converterToMapping.xml -m mapping.xml
```

Részlet a mapping fájlból:

```
<linking source="floor-metrics" target="floor">
    <binding from="commits" to="height"/>
    <binding from="insertions" to="width"/>
    <binding from="deletions" to="length"/>
    <binding from="POC" to="external_character">
        <conversions>
            <conversion type="quantization">
                <parameter name="level2" value="metal"/>
                <parameter name="level3" value="gold"/>
                <parameter name="level4" value="diamond"/>
            </conversion>
        </conversions>
    </binding>
```

Az eredmény:

2. Browsing History converter (Orosz Mónika)

2.1. Chrome History View program

A böngészési előzmények XML fájlba exportálása a Chrome History View program segítségével valósult meg.

A szoftver letölthető: https://www.nirsoft.net/utils/chrome_history_view.html

A szoftver használata: exportálni kívánt előzmény adatok kijelölése, majd mentés XML fájlként.

Az XML fájlból felhasznált metrikák a következők:

- item (egy adott böngészési előzmény)
- visit_count
- typed_count
- url
- stb...

Ezek vizualizálása volt a cél, a converter implementálásával.

2.2. Browsing History converter felépítése és működése

A converter, a meglévő converterek-hez hasonlóan, egy külön package-ben található:

```
codemetropolis.toolchain.converter.browsinghistory;
```

Ebben a packageben lévő, BrowsingHistoryConverter class tartalmazza a converter megvalósítását.

A createDocumentFromXmlFile() metódus segítségével történik a Chrome History View program által előállított XML fájl beolvasása. Ez után a beolvasott fájl felhasználásával, épít egy CdfTree-t, a createElements() metódus ezzel tér vissza. A CdfTree felépítése a CdfElement-ekből történik. Először bállítja a root elementet, majd ezen element-nek a gyerekeit, melyek az item-ek. Az egyes item-ekhez tartoznak a különböző metrikák (látogatások száma, url, url hossza, begépelések száma, ...). A property-k megadását (XML-ben lévő tagek közti értékek) a megfelelő elementhez az addProperties() metódus végzi, az egyes element-ek beállítását adott tag esetén, a addNameAndTypeOfElement() metódus.

A CdfTree-t tartalmazó output xml fájl használatához, valamint a vizualizációhoz szükséges volt a mapping fájl konfigurálása. Az alábbi mapping file részleten látható *item*

source, az egy böngészési előzmény. Az épültek magassága, például a látogatások száma, így látható a vizualizáció után, hogy mely oldalt látogatta többször egy felhasználó.

Mapping file részlet:

```
<linking source="item" target="floor">
        <binding from="visit_count" to="height"/>
        <binding from="typed_count" to="width"/>
        <binding from="url_length" to="length"/>
            ...
```

2.3. Vizualizáció

A converter használata megegyezik a többi converter használatával, a típus változtatása elegendő:

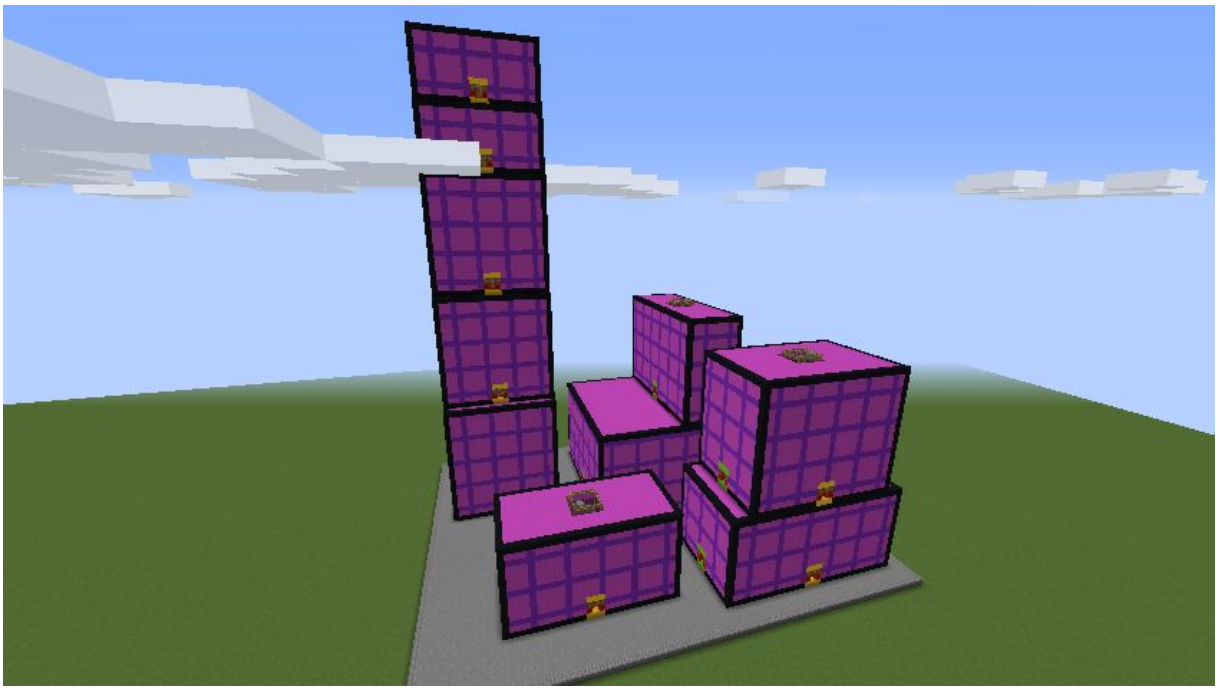
```
java -jar converter.jar -t browsinghistory -i <inputFile>
```

Majd a mapping során, a megfelelő mapping fájl használata szükséges, a megfelelő vizualizációhoz:

```
java -jar mapping.jar -i <inputFile> -o <outputFile> -m
<browsingHistoryMappingFile>
```

A mapping után, a placing és rendering, a megszokott módon működik.

A vizualizáció eredménye:



2.4. További tervek

További tervek közé tartozik, egyszerre több felhasználó böngészési előzményeinek a vizualizálása, úgy, hogy ezek jól látható módon elkülöníthetők legyenek.

3. GUI bővítés - Mapping file editor (Kéri Tamás, Mészáros Viktor)

3.1. Mapping file editor

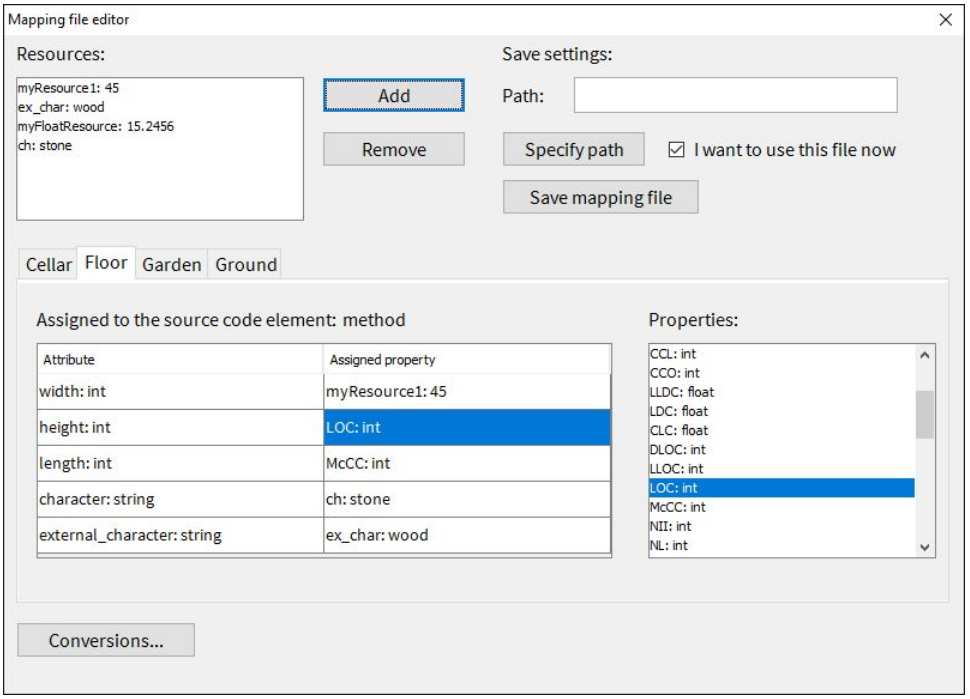
A cél a már meglévő felhasználói felület kibővítése volt egy olyan felület biztosítása révén, amelyen kényelmesen lehet mapping fájlokat készíteni. A kényelmet a drag 'n drop összerendelhetőség biztosítja.

3.2 Felépítés és működés

A GUI megnyitása után a felhasználónak lehetősége van mapping fájlok szerkesztésére is. Először is ki kell tallózni a metrikákat tartalmazó cdf fájlt, ennek a tartalmát olvassa be az editor.

A felhasználónak lehetőség van szabályozni az editorban a különböző buildable (floor, garden, ...) típusokhoz tartozó megjeleníteni kívánt metrikák körét, egy konfigurációs fájl szerkesztésének segítségével (*buildableProperties.cmcfg*). Amennyiben a fájl formátuma hibás (a fájlban szerepelnek az eredeti beállítások is), vagy nem található, akkor az alapértelmezett beállítások lépnek életbe (ez jelen esetben minden attribútum megjelenítését jelenti).

Ha megnyitottuk a felületet, elkezdődhet a szerkesztési folyamat. Lehetőségünk van konstansok felvételére. Ezek helyessége ellenőrizve van. Az általunk definiált



konstansokat ugyanúgy a buildable attribútumokhoz rendelhetünk, mint a metrikákat. Az összerendelési folyamat a következőképp történik - a felhasználó az adott buildable attribútumhoz rendelni kívánt metrikát (vagy konstanst) kijelöli a listában, majd oda húzza az attribútum mellé, amennyiben megfelelőek a típusok (pl. string konstanst nem tudunk az int típusú *height* attribútumhoz húzni). Tehát az összerendelhetőség is ellenőrizve van.

A kvantálási konverzió paramétereit (szintek és a hozzájuk tartozó értékek) a *Conversions…* gomb megnyomására megjelenő dialógusablak adna lehetőséget. A *Specify path…* gomb megnyomására választhatjuk ki a menteni kívánt fájl helyét.