Technische Universität Braunschweig

ifis
Institut für Informationssysteme
Technische Universität Braunschweig

SeaCucumber-Framework

for Neo4j

Bachelor - Teamprojekt
Wintersemester 2017-2018

Instructions for use

Betreuer: Jan-Christoph Kalo, Stephan Mennicke

Name	E-Mail-Adresse
Christian Goldapp	c.goldapp@tu-braunschweig.de
Fabian Kirchner	fabian.kirchner@tu-bs.de
Maximilian Jahn	maximilian.jahn@tu-bs.de
Felix Scholz	felix.scholz@tu-bs.de
Maximilian von Unwerth	maximilian.von-unwerth@tu-bs.de
Tim Witschel	tim.witschel@tu-bs.de

Braunschweig, 28. Januar 2018

1 Instructions

Dear user,
we are delighted that you use our Framework for Neo4j. So that everything runs accordingly you only need Neo4j, Maven and Java for our functions and it runs on every normal operating system. If you have any questions, please do not hesitate to contact us on our „Git Repository“.

1.1 Required-Software

You need for this Framework definitely:

- The Software for Neo4j [1] is very important; it has to be installed otherwise you can't run the framework.
- On your device should be installed the newest version from Java[2] and the needed root-settings are set.
- Also your device need Maven[3], this is important for the „Neo4j - Procedures “.

For a better programming success we recommend to take an IDE. For the instruction we take IntelliJ IDEA[4]. If you really need this instruction, you should download IntelliJ IDEA and you can install the Framework easy step by step (See point 1.3 Step by Step Manual).

[1] `https://neo4j.com/download/other-releases/#releases`
[2] `https://java.com/de/download/`
[3] `https://maven.apache.org/download.cgi`
[4] `https://www.jetbrains.com/idea/download`

1.2 First use guidance for experienced users

- The required Software should be installed on your computer and so you can be sure that everything works right.
- Please start the Neo4j - Software and create a new Database.
- Please create in this Folder (neo4j/default.graphdb) a new Folder with the name „plugins“. You need this for your „Neo4j - Procedures“.
- Now you need our Framework. You can clone our „Git Repository[5] “. If you don't now how you clone the „Git Repository“ please read the Step by Step Manual.
- Furthermore you must download the „Maven Dependencies“ ++++ WIE? +++. If you use an IDE the IDE can do this for you.
- At the package „matcher“ you can create a new Java class where you write your algorithm.
- Where and what needs to be changed at this class, so that „Neo4j - procedures“ can work properly we explain at chapter „start“ coding.

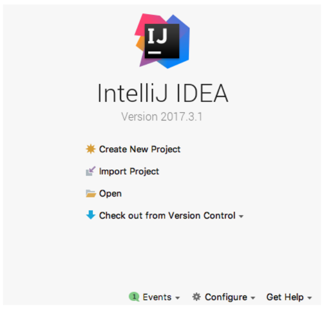
1.3 Step by Step for beginners

This Step by Step is created for using the software „ItelliJ IDEA“.

- The requested Software should be installed on your computer and be sure that everything works right.
- Please start the Neo4j - Software and create a new database. You create it step by step with Neo4j. The only important thing is to know where the storage is. (The Folder is called by default neo4j/default.graphdb).
  (Nice to know for login: by default is user:neo4j, password:neo4j)
- Please create in this Folder (neo4j/default.graphdb) a new Folder with the name „plugins“. You need this for your „Neo4j - Procedures“.

[5] `https://github.com/vonunwerth/neo4j.git`

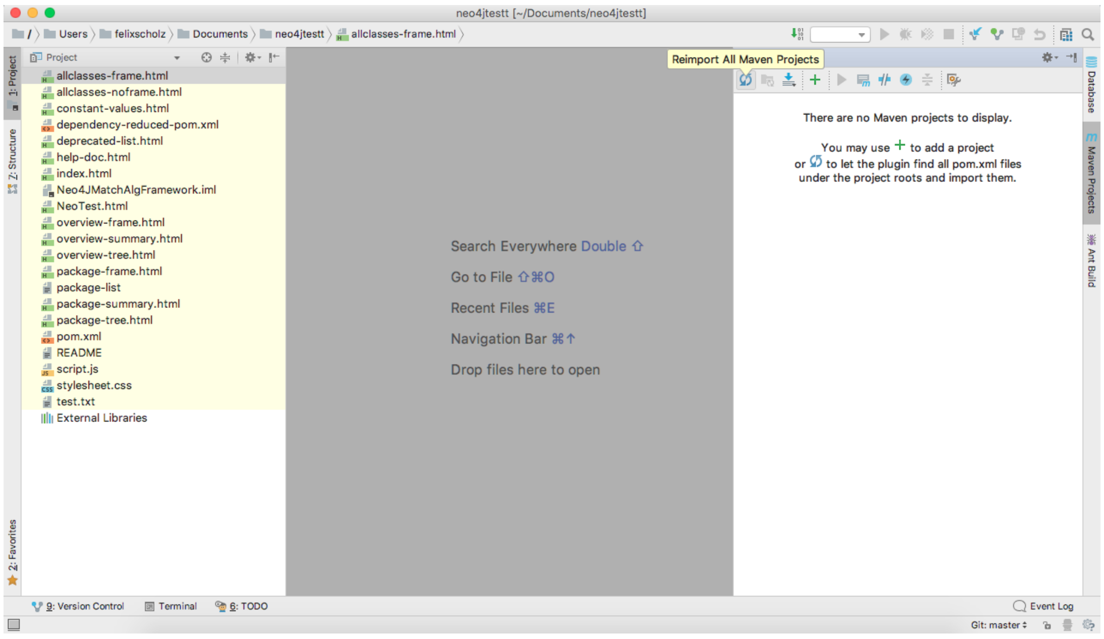
- After downloading the Software „IteliJ IDEA“ opens and ready for use.The following windows opens.



- You click on „Check out from Version Control“ and a window with the name „Clone Repository“ will be open.
  Git Repository URL: https://github.com/vonunwerth/neo4j.git
  Parent Directory: It’s your memory location
  Directory Name: For example Neo4j (You can choose what ever you want.)

  Now you are downloading our Repository to your computer.

- The Folder with our „Git Repository“ is displayed at the „IntelliJ“ window, but you only see „readme“and other data but not the Java files.
  If you want see the Java files, open the Maven Tab on the right side. Click on the tab and on the circle on the left corner. You will be ask, if you want to „import Maven Projects“ - click „yes“. The „Maven Dependencies“ are now downloaded, this needs a while.



There are no Maven projects to display.

You may use + to add a project
or to let the plugin find all pom.xml files
under the project roots and import them.

- All Folder become visible. Open the src/main/java folder.
- At the „package matcher“ you can create a new Java file where you write your algorithm.

1.4 Start coding

On your Computer is the „Git Repository“ and the IDE doesn't show any errors? At this point you can start.

- The new class in the „matcher - package “ must be abstract.
- You have to write a constructor at your class. The constructor (The name must be the same like the classname(the definition for a constructor)). The following structure must be used:

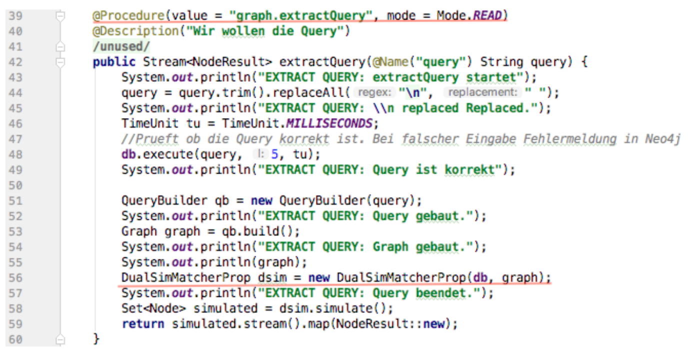
```java
public ???(GraphDatabaseService db, Graph graph) {
        this.db = db;
        this.graph = graph;
}
```

- The next step is to implement the „matchingAlgorithm“. This method is for your algorithm and there you can write what ever you want.

```java
 @Override
public Map<Integer, List<Node>> matchingAlgorithm() {
```

  If you need other help you find a lot helpful methods in the class „matcher “ Check our API.

- After finishing your algorithm, you have to do two more things:
  1.is to change the name of the algorithm object.(GraphProcdures.java Line 56)
  2. Change the procedures name - if you like - it's no must.!(GraphProcdures.java Line 39)
  See the red lines.



```java
39  @Procedure(value = "graph.extractQuery", mode = Mode.READ)
40  @Description("Wir wollen die Query")
41  /unused/
42  public Stream<NodeResult> extractQuery(@Name("query") String query) {
43      System.out.println("EXTRACT QUERY: extractQuery startet");
44      query = query.trim().replaceAll( regex: "\n",  replacement: " ");
45      System.out.println("EXTRACT QUERY: \\n replaced Replaced.");
46      TimeUnit tu = TimeUnit.MILLISECONDS;
47      //Prueft ob die Query korrekt ist. Bei falscher Eingabe Fehlermeldung in Neo4j
48      db.execute(query,  l: 5, tu);
49      System.out.println("EXTRACT QUERY: Query ist korrekt");
50
51      QueryBuilder qb = new QueryBuilder(query);
52      System.out.println("EXTRACT QUERY: Query gebaut.");
53      Graph graph = qb.build();
54      System.out.println("EXTRACT QUERY: Graph gebaut.");
55      System.out.println(graph);
56      DualSimMatcherProp dsim = new DualSimMatcherProp(db, graph);
57      System.out.println("EXTRACT QUERY: Query beendet.");
58      Set<Node> simulated = dsim.simulate();
59      return simulated.stream().map(NodeResult::new);
60  }
```

1.5 After coding

After you write your algorithm in your new class at the Java package matcher, you have to create the „jar " for the database.

- Before you create the jar, you can test the code. Please use the API and the test package. For testing please start the class „ProceduresTest.java".
- Open the Maven tab in „IteliJ " and open the point „Neo4JMatchAlgFramework". The next folder you need is the „Lifecycle" folder, there you click „clean" and after this runs you click on „package".
- The Maven action is finishing? Then you go in your explorer and search the folder where you clone the „Git Repository". There is a new folder named target. Open this „folder" and copy the „original-Neo4JMatchAlgFramework-1.0.jar".
  (The other one is for testing and it is not important. It wouldn't work.)
- Please go to your Neo4j database from the first steps.(see step by step manual) You must paste the „original-Neo4JMatchAlgFramework-1.0.jar" in the created „plugins" Folder.
- After you do this the Neo4j database knew your algorithm. If you are ready open the database with the Neo4j software.

1.6 Work with Neo4j

The very last thing is checking your algorithm is faster or better then everything else. For this you definitely need the following

- You want to use your procedures? Then take the CALL-Statement.
  For example:

  ```
  CALL graph.extractQuery(
  "MATCH (m:Movie)<-[:DIRECTED]-(p:Person)
  RETURN m,p")
  ```

- You want to take your procedures and would like to search with an other query on this result? Then take the CALL statement and the YIELD statement.
  For example:

```
CALL graph.extractQuery(
"MATCH (m:Movie)<-[:DIRECTED]-(p:Person)
RETURN m,p")
 yield node MATCH (m:Movie)<-[:DIRECTED]-(p:Person)
RETURN m,p
```

At this point you know everything we know - So have fun and develop new good algorithms.