Advanced Java Dokumentation

Bennet von Lardon (928809)
Fachhochschule Kiel, MIE, SS2020

1. NEO4J DATENBANK

Version: *4.0.3 Enterprise*
Benötigte Plugins: *Graph Data Science Library 1.2.1*

1.1 Datenbankeinstellungen zum Testen

- Driver: Bolt
- Port: 7688
- Passwort: 4321

1.2 Datenbankeinstellungen für normalen Betrieb

- Driver: Bolt
- Port: 7687
- Passwort: 1234

2. ARCHITEKTUR

2.1 Repository Layer

Hat Zugriff auf die Datenbank.
- Package: *de.fhkiel.advancedjava.repository*

2.2 Service Layer

Hat Zugriff auf das Repository Layer und sich selbst.
- Package: *de.fhkiel.advancedjava.service*
- *DtoConversionLayer* ist den anderen Services übergestellt und konvertiert DTOs in Domain Objekte, andere Services kennen nur Domain Objekte.

2.3 Controller Layer

Hat Zugriff auf das Service Layer.
- Package: *de.fhkiel.advancedjava.controller*

2.4 Task Layer

Hat Zugriff auf das Service Layer.
- Package: *de.fhkiel.advancedjava.tasks*

2.5 Model Layer

Kann von allen Layern zugegriffen werden.
- Package: *de.fhkiel.advancedjava.model*

QueryResult

Modell für Ergebnisse von Cypher queries
- Package: *de.fhkiel.advancedjava.model.queryresult*

Relationship
Modell für RelationshipEntities zwischen Nodes
- Package: *de.fhkiel.advancedjava.model.relationship*

Schedule
Enthält alle NodeEntities für den Graphen
- Package: *de.fhkiel.advancedjava.model.schedule*

Statistics
Enthalten alle Modelle für Statistiken
- Package: *de.fhkiel.advancedjava.model.statistics*

3. REST-API

3.1 /api/connection

Verantwortlich für das Finden von Verbindungen zwischen zwei Stops

GET /api/connection/from/{nameFrom}/to/{nameTo}
- Findet die schnellste Verbindung von der Station mit dem Namen „nameFrom“ zu der Station mit dem Namen „nameTo“ (case sensitive).
- Parameter:
  - nameFrom(String): Name der Station von der gestartet wird. (case sensitive)
  - nameTo(String): Name der Station zu der eine Verbindung gefunden werden soll. (case sensitive)
- Antwort:
  - application/json
  - HTTP status 200
  - Alle in der Verbindung vorkommenden Stationen und die benutzen Linien mit den Segmenten.

GET /api/connection/stops/{nStops}/minutes/{mMinutes}
- Findet eine Sightseeing Verbindung, die mindestens „nStops“ verschiedene Stationen in höchstens „mMinutes“ Minuten besucht.
- Parameter:
  - nStops(Long): Anzahl der Stationen die mindestens besucht werden sollen.
  - mMinutes(Long): Anzahl der Minuten die höchstens verbraucht werden sollen.
- Antwort:
  - application/json
  - HTTP status 200
  - Alle in der Verbindung vorkommenden Stationen und die benutzen Linien mit den Segmenten.

GET /api/connection/cheapest/from/{nameFrom}/to/{nameTo}
- Findet die drei günstigsten Verbindungen von der Station mit dem Namen „nameFrom“ zu der Station mit dem Namen „nameTo“ (case sensitive).
- Parameter:
  - nameFrom(String): Name der Station von der gestartet wird. (case sensitive)

    - nameTo(String): Name der Station zu der eine Verbindung gefunden werden soll. (case sensitive)
- Antwort:
    - application/json
    - HTTP status 200
    - Maximal drei Verbindungen sortiert nach dem günstigsten Preis, die alle in der Verbindung vorkommenden Stationen und die benutzen Linien mit den Teilstrecken enthalten.

GET /api/connection/buytickets/from/{nameFrom}/to/{nameTo}/with/{type}

- Kauft {amount} Tickets von der Station mit dem Namen „nameFrom“ zu der Station mit dem Namen „nameTo“ (case sensitive). Zwischen den beiden Stationen liegt keine weitere Station.
- Parameter:
    - amount(Long): Anzahl der Tickets, die gekauft werden sollen (1-20).
    - nameFrom(String): Name der Station von der gestartet wird. (case sensitive)
    - nameTo(String): Name der Station zu der eine Verbindung gefunden werden soll. (case sensitive)
- Antwort:
    - application/json
    - HTTP status 200
    - Bestätigung des Ticketkaufs mit dem Preis und der Anzahl.

3.2 /api/station

Verantwortlich für das Verwalten von Verkehrsstationen.

POST /api/station/add

- Fügt dem Fahrplan eine neue Station hinzu.
- Parameter:
    - body(application/json)
    - Beispiel:

```
{
    "stopId": 1,
    "types": [
        "SUBURBAN_TRAIN",
        "SUBWAY",
        "BUS"
    ],
    "state": "CLOSED",
    "name": "Hauptbahnhof D2",
    "city": "Kiel",
    "transferTime": 1
}
```

- Antwort:
    - application/json
    - HTTP status 200
    - Die hinzugefügte Station.

PUT /api/station/{name}/set/transfertime/{time}

- Ändert die Transferzeit von der Station mit dem Namen „name“ auf „time“.
- Parameter:

    - name(String): Name von der Station, von der die Transferzeit geändert werden soll.
    - time(Long): Zeit in Minuten, auf die die Transferzeit geändert werden soll.
- Antwort:
    - application/json
    - HTTP status 200
    - Die aktualisierte Station.

PUT /api/station/{name}/set/outoforder

- Setzt den Status der Station mit dem Namen „name“ (case sensitive) auf „außer Betrieb“.
- Antwort:
    - application/json
    - HTTP status 200
    - Die aktualisierte Station.

PUT /api/station/{name}/resolve/outoforder

- Setzt den Status der Station mit dem Namen „name“ (case sensitive) auf „geöffnet“.
- Antwort:
    - application/json
    - HTTP status 200
    - Die aktualisierte Station.

3.3 /api/line

Verantwortlich für das Verwalten von Verkehrslinien.

POST /api/line/add

- Fügt dem Fahrplan eine neue Verkehrslinie hinzu.
- Parameter:
    - body(application/json)
    - Beispiel:

```
{
    "lineId": 55,
    "name": "Test",
    "type": "SUBWAY",
    "sections": [
        {
            "legId": 22,
            "beginStopId": 2,
            "endStopId": 3,
            "durationInMinutes": 8,
            "cost": 2.4,
            "state": "OPENED"
        }
    ]
}
```

- Antwort:
    - application/json
    - HTTP status 200
    - Die hinzugefügte Verkehrslinie.

GET /api/leg/all

- Zeigt alle Teilstrecken im Fahrplan an.
- Antwort:
    - application/json
    - HTTP status 200
    - Alle Teilstrecken im Fahrplan.

PUT /api/leg/{id}/set/outoforder

- Setzt den Status der Teilstrecke mit der ID „id“ auf „außer Betrieb“.
- Antwort:
    - application/json
    - HTTP status 200
    - Die aktualisierte Teilstrecke.

PUT /api/leg/{id}/resolve/outoforder

- Setzt den Status der Teilstrecke mit der ID „id“ auf „geöffnet“.
- Antwort:
    - application/json
    - HTTP status 200
    - Die aktualisierte Teilstrecke.

3.4 /api/schedule

Verantwortlich für das Verwalten von Verkehrslinien.

POST /api/schedule/import

- Importiert einen neuen Fahrplan und löscht den alten, falls vorhanden.
- Parameter:
    - body(application/json)
    - Beispiel:


```
{
    "stations": [
        {
            "stopId": 1,
            "types": [
                "SUBURBAN_TRAIN",
                "SUBWAY",
                "BUS"
            ],
            "state": "CLOSED",
            "name": "Hauptbahnhof D2",
            "city": "Kiel",
            "transferTime": 1
        },
        {
            "stopId": 3,
            "types": [
                "BUS",
                "SUBWAY",
                "SUBURBAN_TRAIN"
            ],
            "state": "OPENED",
            "name": "Hummelwiese",
```

```
            "city": "Kiel",
            "transferTime": 6
        },
        {
            "stopId": 2,
            "types": [
                "SUBURBAN_TRAIN",
                "SUBWAY",
                "BUS"
            ],
            "state": "OPENED",
            "name": "KVG-Btf. Werftstraße C",
            "city": "Kiel",
            "transferTime": 4
        },
        {
            "stopId": 4,
            "types": [
                "SUBURBAN_TRAIN",
                "BUS"
            ],
            "state": "OPENED",
            "name": "Karlstal",
            "city": "Kiel",
            "transferTime": 0
        },
        {
            "stopId": 5,
            "types": [
                "SUBURBAN_TRAIN",
                "BUS"
            ],
            "state": "OPENED",
            "name": "H D W",
            "city": "Kiel",
            "transferTime": 0
        },
        {
            "stopId": 6,
            "types": [
                "SUBWAY",
                "BUS"
            ],
            "state": "OPENED",
            "name": "Fachhochschule",
            "city": "Kiel",
            "transferTime": 0
        }
    ],
    "trafficLines": [
        {
            "lineId": 2,
```

```
        "name": "Subway 1",
        "type": "SUBWAY",
        "sections": [
            {
                "legId": 82,
                "beginStopId": 1,
                "endStopId": 3,
                "durationInMinutes": 5,
                "cost": 0.0,
                "state": "OPENED"
            },
            {
                "legId": 83,
                "beginStopId": 3,
                "endStopId": 6,
                "durationInMinutes": 5,
                "cost": 0.0,
                "state": "OPENED"
            }
        ]
    },
    {
        "lineId": 1,
        "name": "Line 11",
        "type": "BUS",
        "sections": [
            {
                "legId": 4,
                "beginStopId": 5,
                "endStopId": 6,
                "durationInMinutes": 5,
                "cost": 0.0,
                "state": "OPENED"
            },
            {
                "legId": 5,
                "beginStopId": 3,
                "endStopId": 4,
                "durationInMinutes": 2,
                "cost": 0.0,
                "state": "OPENED"
            },
            {
                "legId": 85,
                "beginStopId": 4,
                "endStopId": 5,
                "durationInMinutes": 5,
                "cost": 2.49,
                "state": "CLOSED"
            },
            {
                "legId": 6,
```

```
                    "beginStopId": 1,
                    "endStopId": 2,
                    "durationInMinutes": 5,
                    "cost": 0.0,
                    "state": "OPENED"
                },
                {
                    "legId": 86,
                    "beginStopId": 2,
                    "endStopId": 3,
                    "durationInMinutes": 3,
                    "cost": 0.0,
                    "state": "OPENED"
                }
            ]
        }
    ]
}
```


- Antwort:
    - application/json
    - HTTP status 200
    - Der hinzugefügte Verkehrsplan.

GET /api/schedule/export

- Exportiert den Fahrplan, der in der Datenbank gespeichert ist.
- Antwort:
    - application/json
    - HTTP status 200
    - Der exportierte Verkehrsplan.

3.5 /api/statistics

Verantwortlich für das Verwalten von Statistiken.

GET /api/statistics/get

- Zeigt Statistiken über die Daten in der Datenbank an.
- Antwort:
    - application/json
    - HTTP status 200
    - Statistik über die Daten in der Datenbank.

3.6 /api/vehicle

Verantwortlich für das Verwalten von Statistiken.

GET /api/vehicle/get/all

- Zeigt alle vorhandenen Fahrzeuge in der Datenbank an.
- Antwort:
    - application/json
    - HTTP status 200
    - alle vorhandenen Fahrzeuge.

PUT /api/vehicle/new/{atStationName}/{withLineName}

- Erstellt ein neues Fahrzeug für die Verkehrslinie mit dem Namen „withLineName“ und setzt die aktuelle Position auf die Station mit dem Namen „atStationName“.
- Parameter:
  - atStationName(String): Name der Station, an der das Fahrzeug erstellt werden soll.
  - withLineName(String): Name der Verkehrslinie, für die das Fahrzeug im Einsatz sein soll.
- Antwort:
  - application/json
  - HTTP status 200
  - Das neu erstellte Fahrzeug.

PUT /api/vehicle/update/{id}/{atStationName}

- Setzt den Standort des Fahrzeugs mit der ID „id“ auf die Station mit dem Namen „atStationName“.
- Parameter:
  - id(Long): ID des Fahrzeugs, das geändert werden soll.
  - atStationName(String): Name der Station, an der das Fahrzeug sein soll.
- Antwort:
  - application/json
  - HTTP status 200
  - Das aktualisierte Fahrzeug.