First Run

Beim Ersten Build erscheint immer eine Fehlermeldung. RepoMaven soll angeblich nicht gefunden sein. Diese Fehlermeldung können Sie ignorieren. Diese kommt nur beim ersten Build, das Programm startet aber ohne Probleme.

Beim Ersten Run tritt immer folgende Meldung auf, da ausgegebene Zeilen von bestimmten Maven Imports in der Konsole viel zu lang sind (Bsp. MongoDB, NLP, etc.):

```
Error running 'ParliamentSentimentRadar'

Error running ParliamentSentimentRadar. Command line is too long.

Shorten the command line via JAR manifest or via a classpath file
and rerun.
```

Klicken Sie hierzu auf JAR manifest. Daraufhin wird das Problem nicht mehr auftreten.

Annahmen

Wir nehmen an:

- dass MDB-Stammdaten XML nach unten hin absteigend sortiert ist (von alt nach aktuell)
- dass Abgeordneter = Redner
- dass sich der Aufbau der Bundestag URLs nicht ändert
- dass beim Parsen durch mehrere BundestagBildDatenbank Seiten ein zeitliches Delay von 250ms reicht, ohne vom Server für einen bestimmten Zeitraum blockiert zu werden
- dass ein Redner ohne ID nicht in die DB mitaufgenommen werden kann, da es sein könnte, dass es verschiedene Personen ohne ID bzw. mit der ID = “” gibt und eine eigene zufällige ID-Vergabe nicht möglich wäre, weil das Überprüfen vom Vorhandensein des Redners im Nachhinein dann unmöglich ist
- dass eine Rede ohne Redetext keine Rede ist und somit nicht mit in die DB mitaufgenommen wird
- dass Reden ohne Kommentare NLP-analysiert werden (sonst HeapErrors, SizeErrors, RuntimeErrors, etc., da MongoDB-Dokumente höchstens 16 MB groß sein dürfen und viele Reden mit Kommentaren über diese Grenze hinausgehen)
- dass von 10 Models nur die 2 Models „Status“ und „NamedEntites“ in Swagger-IO nicht darstellbar sind, da man in Swagger nicht mehr als 1 Objekt in ein Array packen kann und somit die Struktur der Modelle nicht modellierbar ist (in Absprache mit unserem Tutor Solaiman)
- dass für die Meta-Daten der Speaker nur alle Infos aus den XMLs und folgende Infos aus der MDB-Stammdatenblatt.xml notwendig sind: Geburtsdatum, Geburtsort, Sterbedatum, Geschlecht und Partei (Fraktion ist ja schon in den XMLs)
- dass ein Redebeitrag da anfängt, wo ein neuer Redner (durch <p klasse=“redner“>
- oder <name>) anfängt und da aufhört, wo daraufhin der nächste Redner (durch <p klasse=“redner“> oder <name>) anfängt. Da <name>-Tags aber nur den Namen als Text enthalten und keine Informationen über den Redner, wie z.B. rednerID, enthalten, weisen wir die Redebeiträge nur den Rednern zu, die sich per <p klasse=“redner“> identifizieren lassen → Die neue SpeechID besteht aus der SpeechID der XML + einem Hashtag + Nummer der Teilrede aus der gleichen Speech

- dass man pro Rede die Tags <a> und <zitat> nicht weiterberücksichtigen muss, da <a> nur etwas mit dem Drucken des Protokolls zu tun hat und kein einziges <zitat>-Tag in allen XMLs vorkam; also berücksichtige ich nur die Haupttags <name>, <p> und <kommentar> für die Reden
- Zur Sicherheit haben wir von den Collections eine Kopie erstellt

Projektverlauf (basierend auf Gant-Diagramm)

Phase 1: (XMLs und Datenbank / NLP)

In der ersten Phase haben wir unser erstes Meeting gehalten. Wir haben unser weiteres Vorgehen geklärt und die Aufgaben aufgeteilt. (Team 1: Lanouar und Nuri) (Team 2: Philipp und Yiran).

Lanouar und Nuri haben sich um den Java Code zum downloaden der XML-Dateien von der Bundestagseite (XMLDownloader.java) gekümmert. Ebenso um die Analyse / Verarbeitung der XML / MDB-Stammdaten zum Ablegen der Daten in geeigneten Datenstrukturen (XMLReader.java) und um die Speaker und Speech Klasse für deren Objektdarstellung (Speaker_File_Impl.java / Speech_File_Impl.java). Währenddessen haben Philipp und Yiran sich um das Grundgerüst für das SBAdmin Dashboard gekümmert. Anschließend wurde die Datenbankanbindung, Verwaltung und NLP-Analyse der Parlamentsdaten von den beiden implementiert.

Phase 2: (API und Dashboard)

In der zweiten Phase hat Lanouar mittels Java Spark und SwaggerIO, die API für die analysierten Daten, implementiert. Parallel dazu haben Philipp und Yiran die SBAdmin-Dashboards mit den Charts und den Reden fertig gestellt.

Phase 3: (Fortschrittsanzeige)

In der dritten und vorletzten Phase haben wir uns um die Fortschrittsanzeige gekümmert. Nuri und Lanouar haben dafür die HTML (index.html) und die JavaScript (loaddata.js) Datei implementiert. Der Datenbankanteil wurde von Yiran und Philipp bereitgestellt (countProgress-Collection).

Phase 4: (Diagramme und Dokumentation)

In der finalen Phase haben wir an den Diagrammen und Dokumentationen unseres Projektes gearbeitet. Das Benutzerhandbuch wurde von Philipp geführt, die

Dokumentation von Lanouar und Nuri, die Diagramme haben Nuri und Philipp erstellt und ebenso haben wir alle unsere Codes auskommentiert (JavaDoc).

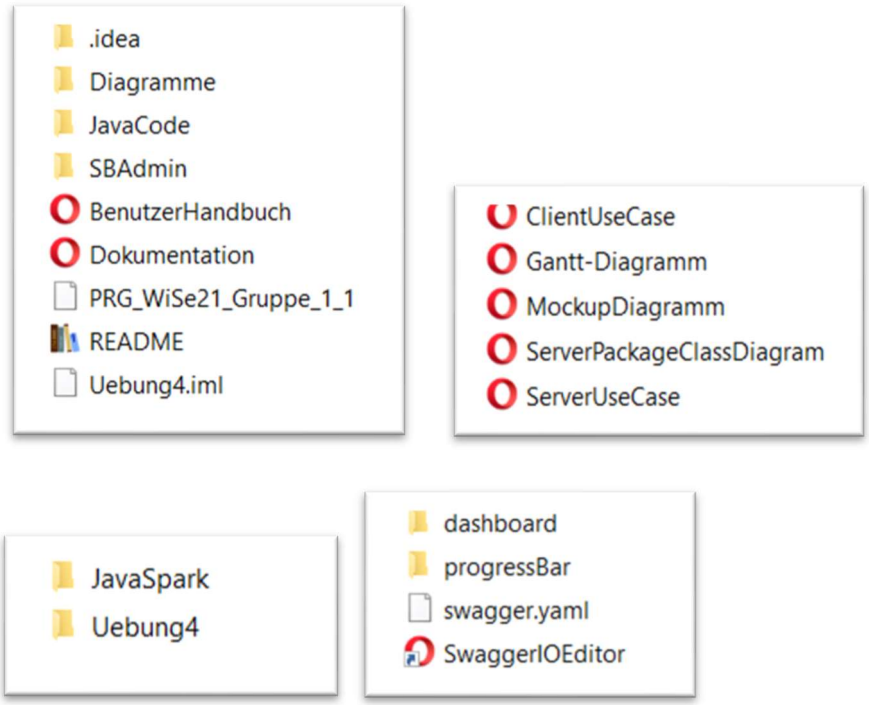
Projektstruktur

Unser Projekt ist folgendermaßen aufgebaut: uebung4 → gruppe_1_1:

- **.idea** → hier befinden sich die GitLab Dateien
- **Diagramme**
    - ServerUseCase
    - ClientUseCase
    - ServerPackageClassDiagram
    - MockupDiagram
    - GantDiagram
- **JavaCode** (hier befindet sich der Java-Teil vom Projekt)
    - Uebung4
        - SourceCodes (Java)
        - ProtokollXMLs (XMLs und MDB-Stammdaten)
    - JavaSpark
        - SourceCodes (JavaSparkMain.java / MongoDBConnectionHandler.java)
- **SBAdmin** (hier befindet sich der HTML/JavaScript-Teil vom Projekt)
    - Dashboard
        - Scripts (alle JavaScripts für die Dashboards + Charts)
        - Charts.html und index.html (SBAdmin Dashboards)
    - ProgressBar
        - Scripts (loaddata.js)
        - Index.html (SBAdmin Dashboard)
    - SwaggerIOEditor (Für die API)
    - Swagger.yaml

- **Benutzerhandbuch** (Anleitung fürs bedienen unseres Programmes)
- **Dokumentation** (Ausführliche Dokumentation unseres Projektes)
- **PRG_WiSe21_Gruppe_1_1** (Datenbank-Login Daten)
- **Readme**
- **Uebung4.iml**

Bilder:



Quellen

- https://www.tutorialspoint.com/how-to-create-a-json-using-jsonobjectbuilder-and-jsonarraybuilder-in-java
- https://scalegrid.io/blog/understanding-mongodb-client-timeout-options/#:~:text=The%20default%20value%20of%20a,you%20risk%20stalling%20your%20application.
- https://stackoverflow.com/questions/37346102/how-to-use-in-operator-in-mongodb-with-two-fields-in-java
- https://www.baeldung.com/spark-framework-rest-api
- https://sparkjava.com/documentation#getting-started