Die Struktur der Dokumentation

1. Anleitung
2. Architektur
3. Methoden

zurück zur Startseite
zurück zur Anleitung
zurück zur Architektur
zurück zur Methoden

Anleitung

Das Datenbankenteam beschäftigt sich damit, die geeignete Struktur für Datenbanken zu erschaffen und, wenn notwendig, zu pflegen. Dafür sind ein paar lebenswichtige für Datenbankenpflege Komponenten zu installieren und einzustellen.

Fürs Projekt sind folgende Komponenten unbedingt zu installieren:
- Apache Maven ab Version 3.5.0 (empfehlenswert ab 3.5.2)
- ElasticSearch (die neueste Version ist zu empfehlen)
- IntelliJ Idea mit allen dafür benötigten Komponenten (wie JDK von Oracle z.B.)

Falls es wichtig ist das Terminal zu benutzen, wobei man nur Windows-Betriebssystem hat, dann kann zusätzlich cygwin installiert werden. Kurze Erklärung, wieso überhaupt das Terminal benutzt werden kann: im Terminal ist es leichter schnell Datenbanken mit Kommando „CURL“ zu überprüfen/verändern. Leider unterstützt die Eingabeaufforderung von Windows BS dieses Kommando nicht.

Wenn man mit der Installation von notwendigen Komponenten fertig ist, dann können die Einstellungen auf der Seite von Datenbankenteam in GitHub benutzt werden.

Am Anfang ist es grundgesetzlich alles, was man zum Arbeiten in Datenbanken von dem Projekt braucht.

zurück zur Startseite
zurück zur Anleitung
zurück zur Architektur
zurück zur Methoden

Architektur

Das Datenbankenschema sieht folgendermaßen aus:

- Index: Articles

- Type: Article
- Type: FullText
- Schema:
  - PMID - Integer
  - Title - String
  - Author - String[]
  - Date - Date
  - Journal - String[]
  - Link - String
  - Keywords - String[]
  - Identifyer - String[] (Synonyme)
  - Annotations - String[]
  - Suggest(Dient zu Autocompletion)
  - Abstract - String[] (Wegen strukturierter Abstracts)
  - Publikationstype - String
  - Substances - String[]
  - Mesh-Terms - String[]
  - Textmining Version - String

zurück zur Startseite
zurück zur Anleitung
zurück zur Architektur
zurück zur Methoden

Pakete, Klassen und Methoden

Die Klasse **ServerApp**:

1. Methode main
2. Methode start

Die Klasse „ServerApp“ dient dem Ausführen von Netty.

Die Klasse **ClientImportService:**

1. Methode parseImport

Die Klasse **ClientQueryService**:

1. Methode parseQuery

Die Klasse **TestClient**:

Hier testet man die Methoden, die von dem Client ausgeführt/aufgerufen werden.

Die Klasse **ImportService**:

1. Methode parseImport
2. createDocument
3. createUtils

Die Klasse **QueryService**:

1. Methode parseQuery
2. Methode getVersion
3. Methode getDocumentByID
4. Methode getUtils
5. Methode getDocumentByTitle
6. Methode deleteDocumentByID
7. Methode deleteDocumentByTitle

Die Klasse **TestService**

Die Klasse **TestApplication**

Die Klasse **App**

zurück zur Startseite
zurück zur Anleitung
zurück zur Architektur
zurück zur Methoden

public static void main(String[] args)

Mit dieser „main“-Methode startet man Netty.
Parameter:

1. String[] args. Der Parameter wird nicht benutzt.

zurück zur Startseite
zurück zur Anleitung
zurück zur Architektur
zurück zur Methoden

public static void start(ResteasyDeployment deployment)

„start“-Methode wird in „main“-Methode der Klasse „ServerApp“ benutzt, um den Dienst von Netty aufrufen.
Parameter:

1. ResteasyDeployment deployment. Der Parameter ist voreingestellt.

zurück zur Startseite
zurück zur Anleitung
zurück zur Architektur
zurück zur Methoden