Министерство науки и высшего образования РФ

Федеральное государственное автономное

образовательное учреждение высшего образования

«**СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ**»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  | | --- | | Институт космических и информационных технологий | | институт | | Программная инженерия | | кафедра | |

**ОТЧЕТ О ПРАКТИЧЕСКОЙ РАБОТЕ**

|  |
| --- |
| Анмаршаллинг данных в формате JSON |
| Тема |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Преподаватель | |  |  |  | К. В. Богданов |
|  | |  | подпись, дата |  | инициалы, фамилия |
| Студент | КИ21-16/2Б, 032161381 |  |  |  | Л. М. Соколов |
|  | номер группы, зачётной книжки |  | подпись, дата |  | инициалы, фамилия |

Красноярск 2024

**СОДЕРЖАНИЕ**

[1 Цель 3](#_Toc180598646)

[2 Задание 3](#_Toc180598647)

[3 Ход выполнения 3](#_Toc180598648)

[3.1 Анализ Json 3](#_Toc180598649)

[3.2 Написание кода 4](#_Toc180598650)

[4 Выводы 7](#_Toc180598651)

[Список использованных источников 8](#_Toc180598652)

# Цель

В ходе выполнения практической работы необходимо изучить способы парсинга (анмаршаллинга) данных в формате JSON.

# Задание

Для выполнения работы необходимо написать утилиту на языке Java, Scala, C++, C# или golnag, скачивающую по указанному URL набор данных в формате JSON, разбирающую эти данные, и помещающую их в таблицу реляционной СУБД (использовать документные СУБД или неатомарные типы данных, например jsonb, запрещено). Утилита должна также вести лог-файл, отражающий события, происходящие в процессе её работы. Можно (и нужно) использовать сторонние библиотеки, такие как GSON, Jackson, fastjson.

# Ход выполнения

## Анализ Json

Для того чтобы понять с чем придется иметь дело, были проанализированы все Json, данные по ссылке (<https://github.com/jdorfman/awesome-json-datasets>). В ходе анализа было выделено 6 типов Json, каждый из которых должен обрабатываться по-особенному, их условный вид предоставлен на рисунке 1. Json же другого вида было принято считать невалидными.

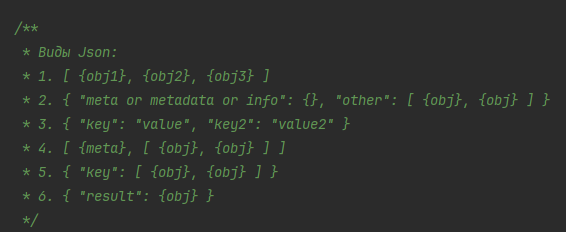


Рисунок 1 – Виды Json

## Написание кода

Для реализации была выбрана Java, в купе с Spring WebClient, который является удобным инструментом для совершения запросов, а также не прекращает работу после ошибок. Кроме того, в Spring «из коробки» поддерживается логирование, которое только необходимо настроить на вывод в файл.

В качестве базы данных был выбран PostgreSQL, как надежный и проверенный инструмент.

Первым делом был сконфигурирован WebClient, для поддержки JSON большого объема, а также для установки connect timeout. Конфигурация предоставлена на листинге 1.

Листинг 1 – конфигурация WebClient

@Configuration  
@ComponentScan("ru.paskal.laba1")  
@Slf4j  
@PropertySource("classpath:application.properties")  
public class WebClientConfiguration {  
  
 private final Integer TIMEOUT;  
  
  
 public WebClientConfiguration(Environment env) {  
 this.TIMEOUT = Integer.*valueOf*(Objects.*requireNonNull*(env.getProperty("client.timeout")));  
 }  
  
 @Bean  
 public WebClient webClientWithTimeout() {  
 HttpClient httpClient = HttpClient.*create*()  
 .option(ChannelOption.*CONNECT\_TIMEOUT\_MILLIS*, TIMEOUT)  
 .doOnConnected(connection -> {  
 connection.addHandlerLast(new ReadTimeoutHandler(TIMEOUT, TimeUnit.*MILLISECONDS*));  
 connection.addHandlerLast(new WriteTimeoutHandler(TIMEOUT, TimeUnit.*MILLISECONDS*));  
 });  
 return WebClient.*builder*()  
 .clientConnector(new ReactorClientHttpConnector(httpClient))  
 .exchangeStrategies(  
 ExchangeStrategies.*builder*()  
 .codecs(configurer -> configurer.defaultCodecs().maxInMemorySize(32 \* 1024 \* 1024))  
 .build()  
 )  
 .build();

Окончание листинга 1

}  
  
 @PostConstruct  
 public void setup() {  
 Hooks.*onErrorDropped*(e -> *log*.error(e.getMessage()));  
 }  
  
}

Затем была продумана и реализована логика программы, она выглядит следующим образом:

Requester – Совершает запрос по заданному url, позволяет обращаться к локальными ресурсам или к web-контенту.

Handler – Обрабатывает полученный JSON, обработка происходит в несколько этапов (о них ниже).

Dao – Сохраняет Json в базу данных, создавая запросы на создание таблицы и сохранения в ней данных.

Обработка JSON происходит следующим образом:

парсинг JSON при помощи Jackson в JsonNode;

распознание типа json;

обработка json распознанного типа (заключается в приведении json к первому типу);

сохранения json в виде пары, где первый элемент Map, вида «Название поля» - «JsonNodeType», второй элемент список из Map, каждый из которых является объектом в изначальном списке.

Каждый этап логируется, это продемонстрировано на рисунке 2.

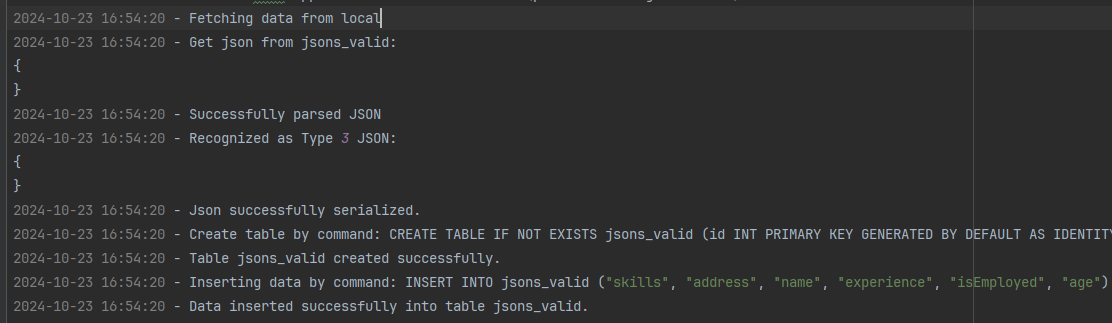


Рисунок 2 – Пример логов для json (сами json удалены для экономии места)

Для сохранения логов в файл был сконфигурирован встроенный в Spring логгер «LogBack». В листинге 2 приведена конфигурация в xml формате.

Листинг 2 – Конфигурация логгера

<configuration>  
 <appender name="FILE" class="ch.qos.logback.core.rolling.RollingFileAppender">  
 <file>logs/app.log</file>  
 <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">  
 <fileNamePattern>logs/app.%d{yyyy-MM-dd}.log</fileNamePattern>  
 <maxHistory>30</maxHistory>  
 </rollingPolicy>  
 <encoder>  
 <pattern>%date{ISO8601} [%thread] %-5level %logger{35} - %msg%n</pattern>  
 </encoder>  
 </appender>  
  
 <appender name="CONSOLE" class="ch.qos.logback.core.ConsoleAppender">  
 <layout class="ch.qos.logback.classic.PatternLayout">  
 <Pattern>  
 %cyan(%d{HH:mm:ss.SSS}) [%level] [%thread] %logger{36} - %yellow(%msg%n)  
 </Pattern>  
 </layout>  
 </appender>  
   
 <root level="INFO">  
 <appender-ref ref="FILE" />  
 <appender-ref ref="CONSOLE" />  
 </root>  
</configuration>

# Выводы

В ходе выполнения практической работы была создана утилита, скачивающая Json по указанному URL, разбирающая этот JSON и сохраняющая данные в реляционной СУБД (PostgreSQL). Разработанная утилита ведет log-файл, отображающий события, происходящие в процессе ее работы. Для реализации парсинга JSON была использована библиотека Jackson.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. СТУ 7.5-07-2021 Система менеджмента качества. Общие требования  
   к построению, изложении и оформлению документов учебной деятельности.  
   Дата введения – 20.12.2021.