Json Parser

Изготвил: Йово Неделчев

От: Катедра СИТ

2Курс 1А Група

Студентски номер:23621691

**1.Задача на проекта :**

JSON е популярен текстов формат за описване на данни. Да се напише програма, която работи с файлове в такъв формат.

След като приложението отвори даден файл, то трябва да може да извършва посочените по-долу операции, в допълнение на общите операции (open, close, save, save as, help и exit):

Validate : Да се направи проверка дали отворения файл е валиден спрямо синтаксиса на JSON. Ако има някакъв проблем, програмата трябва да съобщи максимално ясно какво и къде не е наред, така че потребителят да може да го поправи.

Print : Да се изведе съдържанието на обекта в максимално четим вид.

Search <key> : Да се провери дали в обекта се съдържат данни, записани под този ключ и ако да – да изведе списък от всички такива данни.

Set <path> <string> : При подаване на пълен път <path> към даден елемент, да се замени стойността на посочения елемент с нов обект, получен от низа <string> според синтаксиса на JSON стойност, ако такъв елемент съществува и е единствен и ако символният низ е коректен. В противен случай трябва да обяви каква е грешката.

Create <path> <string> : При подаване на пълен път <path> към даден елемент да се добави такъв елемент, получен от низа <string> според синтаксиса на JSON стойност. Ако такъв елемент съществува, това трябва да се обяви като грешка. В случай, че последните елементи от пътя не съществуват, да се създадат. Ако символният низ или пътят не са коректни, да се съобщи с подходящо описание на грешката.

Delete <path> : При подаване на пълен път <path> до елемент, да се изтрие, ако такъв съществува или да съобщи на потребителя при некоректен път.

Move <from> <to> : Всички елементи, намиращи се на път <from> да бъдат преместени на пътя <to>.

Save [<path>] / Saveas <file> [<path>] : Командите save и saves да работят с произволен път, като записват в текущия или в нов файл обекта на дадения път, ако съществува. Ако <path> не е подаден, да се записва целият обект, който в момента е зареден в паметта.

**2.Съдържание в програмата**

2 Папки за Json файлове : JSON\_Files и FilesFolder2

1 Абстрактен клас - BaseWriter и 5 класа , които го наследяват :

Basic Writer , ContextExtendedWriter , FullWriter , TwoFileWriter и LocationExtendedWriter

1 клас за запазване на временна информация за файловете , докато програмата работи – PathReference

1 клас със Main метод и главната логика на програмата – JsonReader

**3.Код**

**3.1. JsonReader клас**

Импортнати пакети :

import java.io.IOException;  
import java.nio.file.Files;  
import java.nio.file.Path;  
import java.nio.file.Paths;  
import java.util.\*;

java.io.IOException позволява работата с изключения , java.util.\* позволява работата с колекциите като List<> ,ArrayList<> и Map<>,

а java.nio.file.Files/Path/Paths позволяват работата с файлове .

print функция :

public static void print(PathReference path){  
 if(path.getContent()==null || path.getLocation()==null){  
 System.*out*.println("Няма отворен файл все още");  
 }  
 else {  
 try {  
 //System.out.println(path.content);  
 System.*out*.println(Files.*readString*(path.getLocation()).trim());  
 } catch (IOException e) {  
 System.*out*.println("Грешка при изписването на съдържанието на отвореният файл");  
 }  
 }  
}

Взима като аргумент елемент от тип PathReference . Ако съдържаниете или локацията на елемента са празни (те се попълват , когато се извика open функцията) , тогава се изписва в конзолата съобщението „Няма отворен файл все още“. Ако те съществуват се изписва съдържанието на файла (чете се директно от локацията на файла вместо съдържанието на елемента , защото в комбинация с другите методи в програмата може да има разминавания при променяне и изтриване на файловете). При изключение се изпизва в конзолата съобщението „Грешка при изписването на съдържанието на отвореният файл“.

parseJson функция :

public static Map<String, String> parseJson(String json) {  
 Map<String, String> map = new LinkedHashMap<>();  
  
 json = json.trim().replaceAll("[{}\"]", "");  
 String[] pairs = json.split(",");  
  
  
 for (String pair : pairs) {  
 String[] keyValue = pair.split(":");  
 if (keyValue.length == 2) {  
 map.put(keyValue[0].trim(), keyValue[1].trim());  
 }  
 }  
 return map;  
}

Функцията връща Map<String,String> и приема String като аргумент. Първо се създава локална променлива от тип LinkedHashMap<String,String> map . След това чрез json.trim() се махат всички скоби от JSON файла , който беше подаден като аргумент и се създава масив от тип String[] , в който се пазят данните от JSON файла. След това чрез for цикъл двойките от данни се поставят в map-a след като бъдат преместени във още един масив от тип String[] , който ги разделя на 2те части ключ и стойност. На края се връща map-a с новите стойности.

search функция :

public static void search(PathReference path , String keyName) throws IOException {  
 if(path==null || path.getLocation()==null){  
 System.*out*.println("Няма Отворен файл");  
 return;  
 }  
 Boolean printed = false;  
 String content = new String(Files.*readAllBytes*(path.getLocation()));  
 Map<String, String> jsonMap = *parseJson*(content);  
 for (String key : jsonMap.keySet()) {  
 if(keyName.equals(key))  
 {  
 String value = jsonMap.get(keyName);  
 System.*out*.println(key+ ": "+value);  
 printed = true;  
 }  
 }  
 if(!printed){  
 System.*out*.println("Не бешен намерен ключ с такова име");  
 }  
}

Функцията взима за аргументи 2 елемента от тип PathReference и String и може да хвърли IOException изключение. Тя служи за намиране на двойка данни от JSON файл по името на ключа.Първо се проверява дали path елемента е валиден , и ако не е , се изписва на конзолата съобщение „Няма отворен файл“ и се излиза от функцията. Създават се локални променливи : Boolean printed , String conent и Map<String,String> jsonMap . printed е за определяне дали успешно е намерен ключ с даденото име , при което получава стойност true. content взима съдържанието на файла с локация равна на path елемента. JsonMap извиква функцията parseJson и дава като аргумент content. След това с for цикъл се взимат всички ключове от jsonMap , и ако те за равни на keyName , то тогава се създава локална променлива String value , която взима стойноста от двойката ,като извиква функцията jsonMap.get(keyName) и printed получава стойност true. В случай , че не е намерен ключ с такова наименование ,printed ще си остане със стойност false и ще се изпише на конзолата „Не беше намерен ключ с такова име“.