Документация

Постановка задачи

Необходимо конвертировать OpenApiSchema версии 3.1 в версию 3.0.1. Сохранить весь функционал конвертируемой схемы, поддержать возможность расширения конвертера для конвертации в другие версии схемы.

Структура конвертера

Для каждого объекта, описываемого OpenApi схемой существует модельный класс, а также функция, которая конвертирует данную инстанцию класса в JSON объект. В свою очередь, JSON объект описывает исходный объект OpenApiSchema в версии 3.0.1. Данный подход позволяет создавать JSON документ поэлементно, а не сериализовать исходный объект целиком. А учитывая то, что может существовать несколько функций конвертации одного и того же объекта - мы получаем гибкую систему, которая позволяет легко производить конвертацию схемы в любую версию.

Пример функций конвертации

Так как объектов в стандартной схеме очень много, то будут приведены лишь несколько основных примеров функций.

```
public IJsonDocument convertServerVariable(OpenApiServerVairable serverVariable) {
    IJsonDocument result = IJsonDocument.Factory.makeObject();

    if (serverVariable == null)
        return result;

    IJsonDocument doc = serverVariable.document;

    setDocument(doc, result, OpenApiServerVairable.ENUM_FIELD, () -> convertStringArray(serverVariable.enumValues));
    setString(doc, result, OpenApiServerVairable.DEFAULT_FIELD);
    setString(doc, result, OpenApiServerVairable.DESCRIPTION_FIELD);

    return doc;
}
```

Рисунок 1 - Пример конвертации простого объекта

На рисунке 1 мы можем наблюдать, как происходит конвертация простого объекта. Тут, на вход функции подается объект “OpenApi” схемы типа “ServerVariable”. Данный объект содержит в себе два строковых поля и один объект типа “Enum[]”. С помощью функций

“setString” мы переносим из исходного документа (который содержится во входящем объекте), строковые поля в новый JSON объект.
Также, с помощью функции setDocument (рис. 2) и передаваемой в него лямбда-функции устанавливается конвертированный массив “Enum” значений.

```
294 @       public void setDocument(IJsonDocument from, IJsonDocument to,
295                                 String fieldNameFrom, String fieldNameTo,
296                                 ConvertFunction function) {
297             if (from.keys().contains(fieldNameFrom))
298                 to.setDocument(fieldNameTo, function.convert());
299         }
```

Рисунок 2 - Функция установки значения в документ

Рассмотрим пример посложнее. На рисунке 3 происходит конвертация массива ссылок на объект типа “Parameter”.

```
setDocument(doc, result, OpenApiOperation.PARAMETERS_FILED,
        () -> convertObjectArray(operation.parameters,
                obj -> convertObjectReference(obj, this::convertParameter)
        )
);
```

Рисунок 3 - Пример конвертации сложного объекта

Чтобы разобраться, как это работает, рассмотрим функцию конвертации массива объектов (рис.4). На вход поступает список объектов, а также функция, которая будет применяться к каждому элементу исходного списка.
Используем “Java Stream API”. Данная технология является реализацией потоковой обработки данных в Java. С помощью функции “forEach” пройдём по всем элементам исходного списка и запишем результат конвертации каждого элемента в список “result”.

```java
public <T> IJsonDocument convertObjectArray(List<T> array, ConvertObjectFunction<T> function) {
    IJsonDocument result = IJsonDocument.Factory.makeArray();

    if (array == null || array.size() == 0)
        return result;

    array.forEach(o -> result.addDocument(function.convert(o)));

    return result;
}
```

Рисунок 4 - Функция конвертации массива объектов

Так как каждый элемент исходного списка может являться ссылкой, то нам нужно обработать данный случай с помощью функции “convertObjectReference” (рис.5).

```java
public <T> IJsonDocument convertObjectReference(OpenApiReference<T> objRef, ConvertObjectFunction<T> function) {
    if (objRef == null)
        return IJsonDocument.Factory.makeObject();

    if (objRef.isRef())
        return convertReference(objRef);

    return function.convert(objRef.get());
}
```

Рисунок 5 - Функция конвертации ссылки на объект

В данной функции происходит проверка на то, действительно является ли исходный объект ссылкой, и, в зависимости от результата проверки, мы конвертируем либо ссылку, либо с помощью исходной функции конвертации - конвертируем объект, содержащийся в специальном поле объекта “OpenApiReference”.

Во всех вышеописанных примерах утилитарных функций (setDocument, convertObjectArray, convertObjectReference) используются методы функционального программирования в связи с тем, что данные функции делают различные стандартные проверки, которые необходимо производить для всех конвертируемых объектов. Если бы данных функций не было, то исходный код программы очень сильно усложнился как для чтения, так и для дальнейшей поддержки. Также, данный подход объясняется тем, что “OpenApiSchema” является подмножеством JSON схемы, которую удобнее использовать в совокупности с

JavaScript, слабо-типизированным, функциональным языком программирования. По этой причине во многих примерах используются обобщения и интерфейсы, для достижения той самой “слабой типизации” и функциональности.

Используемые приемы функционального программирования

- Лямбда-функции
- Потоковая обработка данных

Описание основных реализованных функций

setString

public void setString(IJsonDocument from, IJsonDocument to, String fieldName)
from - документ, из которого берется строковое поле
to - документ, в который копируется строковое поле
fieldName - имя копируемого поля
Функция копирования строкового поля из одного объекта в другой.

setDocument

public void setDocument(IJsonDocument from, IJsonDocument to,
String fieldName, ConvertFunction function)
from - документ, из которого берется документ
to - документ, в который копируется документ
fieldName - имя копируемого поля
function - функция конвертации объекта
Функция копирования и конвертации документа из одного объекта в другой.

convertObjectArray

public <T> IJsonDocument convertObjectArray(List<T> array, ConvertObjectFunction<T> function)
array - массив объектов
function - функция конвертации объекта
Функция конвертации массива объектов.

convertMap
public <T> IJsonDocument convertMap(Map<String, T> map, ConvertObjectFunction<T> function)
map - массив объектов
function - функция конвертации объекта
Функция конвертации карты объектов.

convert…
public IJsonDocument convert…(OpenApi… objectName)
objectName - коныертируемый объект
Функции конвертации объектов “OpenApi” схемы.

Описание реализованных интерфейсов

ConvertFunction
private interface ConvertFunction
Интерфейс, описывающий функцию конвертации

ConvertObjectFunction<T>
private interface ConvertObjectFunction<T>
T - тип входного параметра
Интерфейс, описывающий функцию конвертации со входным параметром.