Сериализация в JSON

Java Collections
3 уровень, 3 лекция

ОТКРЫТА

— Привет, Амиго!

— Привет, Элли!

— Раз уж ты познакомился с JSON, давай поговорим о нем сегодня подробнее.

— Ок. А где обычно он используется?

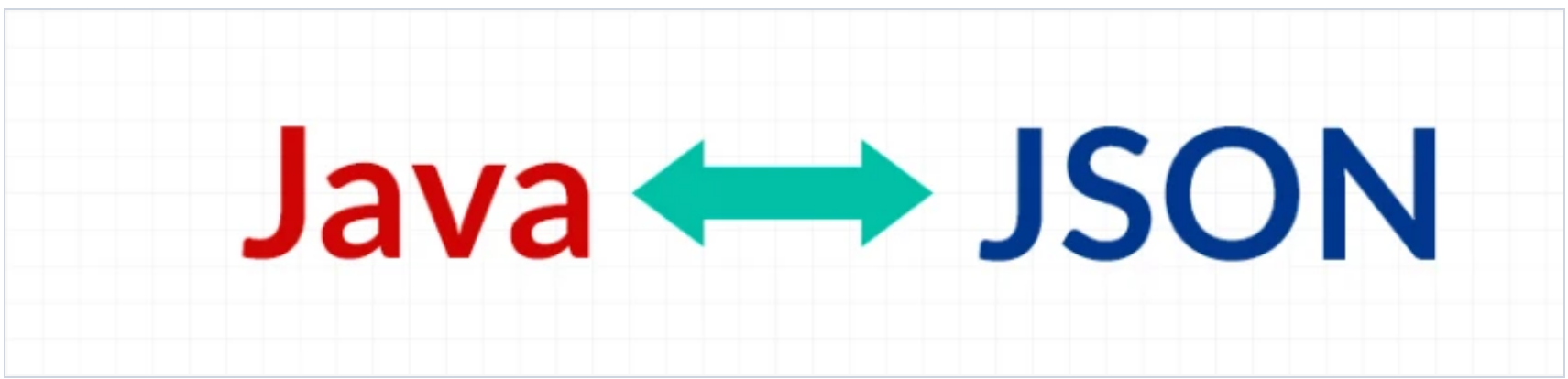
— Обычно дело выглядит так. Кто-то (клиент) запрашивает у Java-программы (сервера) данные. Программа создает Java-объекты и заполняет их информацией из базы данных. Затем преобразовывает их в формат понятный запрашивающему (клиенту), например JSON, и отсылает их обратно.

Давай я тебе расскажу, как работать с ним из Java. Собственно, нам понадобятся только две вещи – сериализовать Java-объекты в JSON-формат и десериализовать Java-объекты из формата JSON.

Т.е. JSON – это стандарт транспортировки сообщений/данных от одной программы к другой. Таких стандартов довольно много. Но если программа написана на JavaScript, она обычно старается работать с JSON.

— Ок. Я готов.

— Отлично. Тогда начнем.



Как ты уже знаешь, в Java есть встроенные стандартные средства сериализации. Но JSON к ним не относится. Поэтому если тебе надо использовать сериализацию объекта в JSON, ты можешь использовать один из популярных фреймворков(библиотек), которые это умеют.

— А чем отличаются различные фреймворки?

— Обычно они отличаются степенью сложности: есть фреймворки, которые умеют делать только самое необходимое, но они очень маленькие и простые. А есть и большие сложные фреймворки, которые могут делать гораздо больше.

Одним из популярных фреймворков считается Jackson. Мы рассмотрим работу с JSON на его примере.

Для начала тебе надо скачать этот фреймворк и добавить его себе в проект. Делать это надо в Intellij IDEA само собой. Загрузить фреймворк можно по ссылке.

— Готово.

— Отлично. Тогда продолжим.

Сконвертировать Java-объект в JSON примерно так же просто, как и сериализовать его. Для этого есть специальный класс ObjectMapper (com.fasterxml.jackson.databind.ObjectMapper)

Конвертация объекта в JSON

```
1  public static void main(String[] args) throws IOException
2  {
3   //создание объекта для сериализации в JSON
4   Cat cat = new Cat();
5   cat.name = "Murka";
6   cat.age = 5;
7   cat.weight = 4;
8
9   //писать результат сериализации будем во Writer(StringWriter)
10  StringWriter writer = new StringWriter();
11
12  //это объект Jackson, который выполняет сериализацию
13  ObjectMapper mapper = new ObjectMapper();
14
15  // сама сериализация: 1-куда, 2-что
16  mapper.writeValue(writer, cat);
17
18  //преобразовываем все записанное во StringWriter в строку
19  String result = writer.toString();
20  System.out.println(result);
21 }
```

Класс Cat, объект которого конвертирует в JSON

```
1  @JsonAutoDetect
2  class Cat
3  {
4   public String name;
5   public int age;
6   public int weight;
7  Cat(){}
8  }
```

Результат сериализации и вывода на экран:

```
1  {"name":"Murka", "age":5, "weight":4}
```

Вот как все было:

В строках 4-7 мы создаем объект класса Cat и заполняем его данными.

Строка 10 – создаем объект Writer, куда будем писать строку — JSON представление объекта.

Строка 13 – создаем объект ObjectMapper, который и выполняет всю сериализацию.

Строка 16 – пишем JSON-представление объекта cat в writer.

Строки 19-20 – выводим результат на экран.

Все выглядит довольно просто. Не сложнее родной сериализации в Java.

— А как будет выглядеть десериализация?

— Да почти так же, только короче:

```
public static void main(String[] args) throws IOException
{
 String jsonString = "{ \"name\":\"Murka\", \"age\":5, \"weight\":4}";
 StringReader reader = new StringReader(jsonString);

 ObjectMapper mapper = new ObjectMapper();

 Cat cat = mapper.readValue(reader, Cat.class);
}
```

Класс, объект которого десериализуется из JSON-формата

```
@JsonAutoDetect
class Cat
{
 public String name;
 public int age;
 public int weight;

 Cat() { }
}
```

Тут еще проще. Берем ObjectMapper и передаем в него строку с JSON или StringReader, а также класс объекта, который надо десериализовать. Вызываем метод **readValue**, и на выходе получаем готовый Java-объект со всеми данными.

— Ну, точно, как десериализация в Java.

— Почти. К объектам, которые сериализуются/десериализуются в JSON есть несколько требований:

1) поля должны быть видимые: или public или иметь getter’ы и setter’ы;

2) должен быть конструктор по умолчанию (без параметров).

— Ясно. Ожидаемо, в общем. Хотя Java отлично сериализовала и private поля.

— Так то — Java. У нее есть доступ к скрытым данным. От себя не утаишь.

Тут есть еще третий аспект. Надеюсь, ты обратил внимание на аннотацию @JsonAutoDetect в классе Cat?

— Ага. Как раз хотел спросить – что это такое.

— Это аннотации – служебная информация для фреймворка Jackson. Можно очень гибко управлять результатом сериализации в JSON формат, расставляя правильные аннотации.

— Круто! А что за аннотации есть?

— Вот тебе несколько:

Аннотация	**Описание**
@JsonAutoDetect	Ставится перед классом. Помечает класс как готовый к сериализациив JSON.
@JsonIgnore	Ставится перед свойством. Свойство игнорируется при сериализации.
@JsonProperty	Ставится перед свойством или getter’ом или setter’ом. Позволяет задать другое имя поля при сериализации.

	Позволяет задать порядок полей для сериализации.

— Как интересно. А есть еще?

— Есть много. Но не сейчас. Сейчас давай немного переделаем наш первый пример:

Конвертация объекта в JSON

```
1  public static void main(String[] args) throws IOException
2  {
3   Cat cat = new Cat();
4   cat.name = "Murka";
5   cat.age = 5;
6   cat.weight = 4;
7
8   StringWriter writer = new StringWriter();
9
10  ObjectMapper mapper = new ObjectMapper();
11
12   mapper.writeValue(writer, cat);
13
14   String result = writer.toString();
15   System.out.println(result);
16  }
```

Класс, объект которого конвертирует в JSON

```
1  @JsonAutoDetect
2  class Cat
3  {
4   @JsonProperty("alias")
5   public String name;
6   public int age;
7   @JsonIgnore
8   public int weight;
9
10  Cat() {
11  }
12 }
```

Результат сериализации и вывода на экран:

```
1  {"age":5, "alias":"Murka"}
```

Код остался тот же, но я поменяла аннотации: указала другое имя полю name — имя alias. А также отметила поле weight как Ignore, в результате JSON объекта поменялся.

— Хорошо, что можно так всего настраивать – думаю, мне это обязательно пригодится.

А десериализация поймет, как с этим работать? При десериализации из JSON в Java-объект, значение поля alias будет занесено в name объекта Cat?

— Да, десериализация будет работать как надо. Она умная.

— Что не может не радовать.

+138

Комментарии (135) популярные новые старые

JavaCoder

Введите текст комментария

Александр Горохов Уровень 25, Дятьково, Россия 10 июля, 13:33

Дольше разбирался с установкой библиотеки, чем с примером:)

Ответить 0

HotTab Уровень 34, Москва 29 апреля, 04:25

Пытаюсь воспроизвести пример с сериализацией, пишет databindExeption.Когда добавил databindExeption пишет что он не throwable. Помогите пожалуйста((

Ответить 0

Anonymous #2941322 Уровень 51 5 мая, 10:57

Что-то делаешь не так. Этот код не генерирует таких исключений.
В классе Кот соблюдены условия (публичные поля, конструктор по умолчанию)?

Ответить 0

Руслан Уровень 42 22 мая, 16:24

У меня была такая же проблема. Помог комментарий Петра от 23.04.19:
Project structure/libraries/+/from maven/ вводим com.fasterxml.jackson.core:jackson-databind, жамкаем лупу и ждем, пока выполнится поиск по репам выбираем стрелочкой нужный источник, ставим галку скачать в проект, по желанию скачать джавадоки, аннотации, сырцы и прочее. выбраем область применения "4. Collections" (потом более вложенным плюсом (+) можно перенести повыше, указав весь проект JavaRush)

Ответить 0

Борис Уровень 30 31 марта, 08:27

Подскажите, при конвертации объекта из JSON (первый пример), подчёркивает (reader, Cat.class) строка № 8 и сообщает: не разрешённый метод readValue, ссылается на StringReader. Спасибо

Ответить 0

Anonymous #2941322 Уровень 51 5 мая, 11:26

Нужно было пример кода загружать, потому что так это даже не на кофейной гуще гадание..
Может быть вызывал readValue не у ObjectMapper?
У меня работает как со стринг-ридером, так и непосредственно со стрингой.

Ответить 0

max Уровень 42, Краснодар, Россия 23 марта, 22:55

После того как произвел сериализацию объекта в JSON файл, где я могу этот файл?

Ответить 0

Anonymous #2941322 Уровень 51 5 мая, 11:19

Чтобы произвести сериализацию в файл, нужно было явно объявить его, произвести в него запись. Судя по поставленному вопросу, ты этого не делал, значит и файла такого нигде нет.

Полученную в примере строку можем записать в файл таким образом:

```
try (FileWriter fileWriter = new FileWriter("C:\\Downloads\\123\\2.txt")) {
    fileWriter.write(result);
}
```

Результат на картинке.

Десериализация обратна: читаем строку, подсовываем ее ObjectMapper-у - достаем из ~~шляпы~~ файла кота.

Ответить 0

fFamous Уровень 51, Санкт-Петербург 6 декабря 2021, 21:11

Интересно, при конвертации в/из выдает такой результат:

```
{"age":15,"name":"AWD","rightHanded":true} // результат конвертации в json

Car{age=15, name='AWD', isRightHanded=true} // результат конвертации из json в объ
```

В json исчезло is и Right стало right. А потом обратно преобразовалось в объект нормально) Видимо там какие-то сокращения есть.

Ответить 0

Роман Кончалов Уровень 28, Россия EXPERT 26 января, 11:45

Может есть метод с соответсвующим названием? Откуда пример?

Ответить 0

fFamous Уровень 51, Санкт-Петербург 26 января, 15:55

Ну, вот:

```
import com.fasterxml.jackson.annotation.JsonAutoDetect;
import com.fasterxml.jackson.databind.ObjectMapper;
import lombok.Data;

import java.io.IOException;
import java.io.StringWriter;

public class JacksonTest {
    public static void main(String[] args) throws IOException {
        Car car = new Car();
        car.setAge(15);
        car.setName("BMW");
        car.setRightHanded(true);

        ObjectMapper mapper = new ObjectMapper();
        StringWriter writer = new StringWriter();

        mapper.writeValue(writer, car);

        System.out.println(writer);
    }
}

@Data
@JsonAutoDetect
class Car {
    private int age;
    private String name;
    private boolean isRightHanded;
}
```

По всей видимости на каком-то этапе "is" удаляется, как в случае с теми же setter'ами в Java, но там это IDE генерит, из того же isRightHanded сгенерит setRightHanded. Видимо, да, где-то какой-то метод удаляет, я чекал аннотации, но слишком глубого было лень лезть :)

Ответить 0

Allari Уровень 35, Санкт-Петербург, Russian Federation 5 апреля, 09:19

так переменная isRightHanded приватная, о ней ничего неизвестно, а сеттер в коде называется setRightHanded без "Is" вот наверное он и преобразует так...

Можно сразу в строку запихнуть и считать:

```
1  String result = objectMapper.writeValueAsString(Ivan);
2  Human John = objectMapper.readValue(result, Human.class);
```

Ответить +1

PaiMei in J# Grand Master в **Eagles' Claw** 21 октября 2021, 11:49

Господа, если у кого-нибудь есть информация, то поделитесь пожалуйста, для чего при десериализации нам нужен default-конструктор?

Ответить 0

Ars Уровень 41 19 ноября 2021, 12:35

Я так полагаю, для того чтобы создать объект, прежде чем заполнять данными его поля.

Ответить +1

Роман Кончалов Уровень 28, Россия EXPERT 26 января, 11:42

Потому что фреймворк не может знать, какие значения передавать в конструктор с параметрами. Стоит помнить при этом, что когда нет вообще никаких конструкторов, то неявно существует конструктор без параметров, и у вас всё будет работать. Но если вы захотите сделать конструктор с параметрами, но дополнительно не сделаете без параметров, тогда неявный конструктор без параметров бесследно исчезнет и программа не будет работать до тех пор, пока вы не напишете его явно, либо не удалите все конструкторы.

Ответить +1

Kes Чайник в **Банк** 21 мая, 05:56

Спокойно десериализовал без конструктора.

Ответить 0

PaiMei in J# Grand Master в **Eagles' Claw** 20 октября 2021, 18:03

Насколько я понимаю можно ввести в поиске Google "репозиторий Maven" и далее перейдя по ссылке мы попадаем на замечательный ресурс, откуда можно вытянуть практически любую библиотеку, плюс когда мы выбираем интересующий нас фреймворк, то можно посмотреть кол-во скачиваний той или иной его версии

Ответить 0

Лиза Воренувкина Уровень 43, Кривой Рог, Ukraine 24 сентября 2021, 18:46

Ни чего не поняла , а зашла по ссылке и что там делать .

Ответить +6

Shamil Уровень 41, Россия 28 августа 2021, 14:48

Кто-нибудь руками пытался подключить? Я просто иногда в связке sublime+cmd делаю мелкие примеры. Я скачал три .jar'a, закинул их в папку lib моего проекта, подключил, разумеется, в файле. Вот так компилирую.
javac Example.java -classpath ./lib/;./lib/jackson-core-2.12.5.jar;./lib/jackson-databind-2.12.5.jar;./lib/jackson-annotations-2.12.5.jar
после запуска (java Example) даёт вот какой ответ:
Exception in thread "main" java.lang.NoClassDefFoundError: com/fasterxml/jackson/databind/ObjectMapper
at Example.convertToJSON(Example.java:55)
at Example.main(Example.java:49)
Caused by: java.lang.ClassNotFoundException: com.fasterxml.jackson.databind.ObjectMapper
at java.base/jdk.internal.loader.BuiltinClassLoader.loadClass(BuiltinClassLoader.java:636)
at java.base/jdk.internal.loader.ClassLoaders$AppClassLoader.loadClass(ClassLoaders.java:182)
at java.base/java.lang.ClassLoader.loadClass(ClassLoader.java:519)
... 2 more

Ответить 0

PaiMei in J# Grand Master в **Eagles' Claw** 20 октября 2021, 17:55

Мб @JsonAutoDetect забыли поставить перед сериализуемым классом?

Ответить 0

Показать еще комментарии

RUSH

JavaRush — это интерактивный онлайн-курс по изучению Java-программирования с нуля. Он содержит 1200 практических задач с проверкой решения в один клик, необходимый минимум теории по основам Java и мотивирующие фишки, которые помогут пройти курс до конца: игры, опросы, интересные проекты и статьи об эффективном обучении и карьере Java-девелопера.

ПОДПИСЫВАЙТЕСЬ

ЯЗЫК ИНТЕРФЕЙСА

Русский

СКАЧИВАЙТЕ НАШИ ПРИЛОЖЕНИЯ

ДОСТУПНО В Google Play

Загрузите в App Store