[**Java读取/写入Yaml配置文件**](https://blog.csdn.net/u013410747/article/details/51647535)

标签： [java](http://so.csdn.net/so/search/s.do?q=java&t=blog) [yaml结构](http://so.csdn.net/so/search/s.do?q=yaml%E7%BB%93%E6%9E%84&t=blog" \t "_blank) [读取yaml](http://so.csdn.net/so/search/s.do?q=%E8%AF%BB%E5%8F%96yaml&t=blog) [解析yaml](http://so.csdn.net/so/search/s.do?q=%E8%A7%A3%E6%9E%90yaml&t=blog) [jyaml解析](http://so.csdn.net/so/search/s.do?q=jyaml%E8%A7%A3%E6%9E%90&t=blog)

2016年06月12日 17:00:5818444人阅读 [评论](javascript:void(0);)(0) [收藏](javascript:void(0);) [举报](javascript:void(0);)

https://csdnimg.cn/release/phoenix/images/category_icon.jpg 分类：

Java（27） https://csdnimg.cn/release/phoenix/images/arrow_triangle%20_down.jpg

版权声明：本文为博主张相逢原创文章，未经本人允许不得转载。违者必究！！！ https://blog.csdn.net/u013410747/article/details/51647535

目录[(?)](https://blog.csdn.net/u013410747/article/details/51647535" \o "系统根据文章中H1到H6标签自动生成文章目录)[[+]](https://blog.csdn.net/u013410747/article/details/51647535)

**JYaml文件流读取/写入Yaml配置文件**

yaml配置文件格式规范：- 表示sequence(list列表结构)，: 表示map键值对

#以下是示例yaml结构

age: 23

children:

-

age: 8

name: mary1

sex: man

-

age: 9

name: simon2

sex: fatel

name: simon.zhang

sex: man

1.准备工作，创建Persion实体类

package com.yaml.entity;

import java.util.List;

/\*\*

\* Created by simon on 16-6-12.

\*/

public class Person {

private String name;

private int age;

private String Sex;

private List<Person> children;

//省略 getter and setter method.....

}

2.实现读取/写入yaml配置文件

package com.yaml;

import com.yaml.entity.Person;

import org.ho.yaml.Yaml;

import org.junit.Test;

import java.io.File;

import java.io.FileNotFoundException;

import java.util.ArrayList;

import java.util.List;

/\*\*

\* Hello world!

\*/

public class App {

//写入yaml配置文件

@Test

public void write() {

/\* Initialize data. \*/

Person father = new Person();

father.setName("simon.zhang");

father.setAge(23);

father.setSex("man");

List<Person> children=new ArrayList<Person>();

for (int i = 8; i < 10; i++) {

Person child = new Person();

if (i % 2 == 0) {

child.setSex("man");

child.setName("mary" + (i - 7));

} else {

child.setSex("fatel");

child.setName("simon" + (i - 7));

}

child.setAge(i);

children.add(child);

}

father.setChildren(children);

/\* Export data to a YAML file. \*/

File dumpFile = new File(System.getProperty("user.dir") + "/test/src/main/conf/testYaml.yaml");

try {

Yaml.dump(father, dumpFile);

} catch (FileNotFoundException e) {

e.printStackTrace();

}

}

//读取yaml配置文件

@Test

public void read() throws FileNotFoundException {

File dumpFile=new File(System.getProperty("user.dir") + "/test/src/main/conf/testYaml.yaml");

Person father = (Person) Yaml.loadType(dumpFile, Person.class);

StringBuilder stringBuilder=new StringBuilder();

stringBuilder.append(father.getName())

.append("\t")

.append(father.getSex())

.append("\t")

.append(father.getAge())

.append("\t")

.append(father.getChildren().size());

System.out.println(stringBuilder.toString());

}

}

Map结构的yaml文件

JDBC.driver: com.mysql.jdbc.Driver

JDBC.url: "jdbc:mysql://127.0.0.1:3306/shipping\_local?autoReconnect=true&autoReconnectForPools=true&useUnicode=true&characterEncoding=utf8"

JDBC.username: shipping

JDBC.password: shipping

JDBC.autoCommit: false

读取Map结构

//读取yaml配置文件Map结构

@Test

public void read2() throws FileNotFoundException {

File dumpFile=new File(System.getProperty("user.dir") + "/test/src/main/conf/testYaml.yaml");

Map father =Yaml.loadType(dumpFile, HashMap.class);

for(Object key:father.keySet()){

System.out.println(key+":\t"+father.get(key).toString());

}

}

打印结果如下：

提供 Maven pox.xml 作为参考

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.yaml</groupId>

<artifactId>test</artifactId>

<version>1.0-SNAPSHOT</version>

<packaging>jar</packaging>

<name>test</name>

<url>http://maven.apache.org</url>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

</properties>

<dependencies>

<dependency>

<groupId>org.jyaml</groupId>

<artifactId>jyaml</artifactId>

<version>1.3</version>

</dependency>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.11</version>

<scope>provided</scope>

</dependency>

</dependencies>

</project>

* 上一篇 [Mysql读写分离,同步复制的探索实现](https://blog.csdn.net/u013410747/article/details/51590744)