

1.序列化简介

1.1 定义

序列化是将对象状态转换为可保持或传输的格式的过程。与序列化相对的是反序列化，它将流转换为对象。这两个过程结合起来，可以轻松地存储和传输数据

1.2 序列化的目的

通过序列化以字节流的形式使对象在网络中进行传递和接收

永久性保存对象，保存对象的字节序列到本地文件或者数据库中

通过序列化在进程间传递对象

深拷贝对象

跨语言传递数据

1.3 序列化常见的方式

是否跨语言：

生成的中间文件二进制流不同：

2.常见的序列化方式特点

2.1 JDK

参考脑图

2.2 JSON

Jackson

相比json-lib框架，Jackson所依赖的jar包较少，简单易用并且性能也要相对高些。
而且Jackson社区相对比较活跃，更新速度也比较快。
Jackson对于复杂类型的json转换bean会出现问题，一些集合Map，List的转换出现问题。
Jackson对于复杂类型的bean转换Json，转换的json格式不是标准的Json格式

Gson

Gson是目前功能最全的Json解析神器，Gson当初是为因应Google公司内部需求而由Google自行研发而来，但自从在2008年五月公开发布第一版后已被许多公司或用户应用。
Gson的应用主要为toJson与fromJson两个转换函数，无依赖，不需要例外额外的jar，能够直接跑在JDK上。
而在使用这种对象转换之前需先创建好对象的类型以及其成员才能成功的将JSON字符串成功转换成相对应的对象。
类里面只要有get和set方法，Gson完全可以将复杂类型的json到bean或bean到json的转换，是JSON解析的神器。
Gson在功能上面无可挑剔，但是性能上面比FastJson有所差距。

FastJson

Fastjson是一个Java语言编写的高性能的JSON处理器,由阿里巴巴公司开发。

无依赖,不需要例外额外的jar,能够直接跑在JDK上。

FastJson在复杂类型的Bean转换Json上会出现一些问题,可能会出现引用的类型,导致Json转换出错,需要制定引用。

FastJson采用独创的算法,将parse的速度提升到极致,超过所有json库。

2.3 kryo

- 1.速度快,序列化后体积小
- 2.跨语言支持较复杂

2.4 Protobuf

- 1.结构化数据存储格式(xml,json等)
- 2.高性能编解码技术
- 3.语言和平台无关,扩展性好
- 4.支持java,C++,Python三种语言。

2.5 hessian

- 1.默认支持跨语言
- 2.较慢

2.6 Thrift

- 1.Thrift支持多种语言(C++,C#,Cocoa,Erlang,Haskell,java,OCaml,Perl,PHP,Python,Ruby,和SmallTalk)
- 2.Thrift适用了组建大型数据交换及存储工具,对于大型系统中的内部数据传输,相对于Json和xml在性能上和传输大小上都有明显的优势。
- 3.Thrift支持三种比较典型的编码方式。(通用二进制编码,压缩二进制编码,优化的可选字段压缩编解码)
- 4.支持RPC

2.7 fst

- 1.fst是完全兼容JDK序列化协议的序列化框架,序列化速度大概是JDK的4-10倍,大小是JDK大小的1/3左右。

2.8 MsgPack

官网: <https://msgpack.org>

redis中使用此标准

- 1.速度快
- 2.序列化后的数据占用空间小
- 3.中间转换的是字符数组,不方便存储到文件之中
- 4.性能不够稳定(要求字段顺序一致,删减字段都需要注意)

2.9 Avro

hadoop研发序列化

支持RPC

3 各大框架序列化方式

3.1 dobbo

通讯协议: dobbo协议, rmi协议, hessian协议, http协议, webservice协议
序列化方式: hessian, jdk, json, soap, kryo, fst, Protobuf(新版本支持), kryo

3.2 springcloud

通讯协议: http
序列化方式: json

3 性能测试

3.1 性能衡量指标

不同的序列化方式, 本身设计就是满足不同的应用场景, 可序列化的数据类型, 比如是否支持StringBuffer和StringBuilder等, 所以单独对比速度快慢, 也不太合理, 但大体上衡量标准可以参考如下

1. 是否支持跨语言, 支持语种是否丰富
2. 编码后的码流
3. 编解码的性能
4. 类库是否小巧, API使用是否方便
5. 使用者开发的工作量和难度。

性能参考:

https://blog.csdn.net/Brady74/article/details/90983288?utm_medium=distribute.pc_relevant_t0.none-task-blog-BlogCommendFromMachineLearnPai2-1.nonecase&depth_1-utm_source=distribute.pc_relevant_t0.none-task-blog-BlogCommendFromMachineLearnPai2-1.nonecase

<https://github.com/eishay/jvm-serializers>

其中: JSON类库, 更适用于业务场景的开发, 类的拷贝复制, 不适合网络传输, 性能低于其他类别的序列化方式

其他的类库, 更倾向于中间件的设计, 网络传输