Gson

# 代码

## TypeAdapter

定义转换java对象的格式,Gson默认转换class文件为json时使用的是内置的adapters,如果gson的默认的json转换格式不适合,可以继承这个类来自定义转换格式,例如:

public class PointAdapter extends TypeAdapter<Point> {

public Point read(JsonReader reader) throws IOException {

if (reader.peek() == JsonToken.NULL) {

reader.nextNull();

return null;

}

String xy = reader.nextString();

String[] parts = xy.split(",");

int x = Integer.parseInt(parts[0]);

int y = Integer.parseInt(parts[1]);

return new Point(x, y);

}

public void write(JsonWriter writer, Point value) throws IOException {

if (value == null) {

writer.nullValue();

return;

}

String xy = value.getX() + "," + value.getY();

writer.value(xy);

}

}

使用这个adapter,Gson会将Points转换为类似”5,8”,而不是”{“x”:5,”y”:8}”的json.通过type adapter将一个复杂的java类转换为一个任一格式的json.

Adapter中的read()和write()方法处理的必须是统一格式的值,对于原生类型,这意味着readers和writers必须是一致的使用nextBoolean(),nextInt()…,对于数组类型,type adapters应该是调用beginArray()方法开始,转换所有elements，以endArray()结尾.对于Objects类型,应该以beginObject()开始,转换对象以endObject()结尾.转换失败或者转换的值太多了会导致程序崩溃.

Type adapter应该做好从流中读取null和将null写入流的准备.另外,应该在使用gson 的时候注册nullsafe()方法,如果gson被以GsonBuilder.serializeNulls(),这些null将会写入最终的json中,否则他们将自动被忽略,无论怎样type adapter一定得处理null.

要使用一个自定义的适配器,可以在GsonBuilder中注册,例如:

GsonBuilder builder = new GsonBuilder();

builder.registerTypeAdapter(Point.class,new PointAdapter());

//如果PointAdapter在read/write方法中没有检查nulls,需要使用builder.registerTypeAdapter(Point.class,new PointAdapter().nullSafe());

Gson gson = builder.create();