<http://blog.csdn.net/wangjun5159/article/details/51328899>

**memchache 将对象序列化后保存**

memcahce将值序列化成字节数组，然后存储到缓存中。

如下例，我们将user对象序列化到文件a.txt中，同时将user保存到缓存中，通过比较文件和缓存中的值   
发现，两者是一样的。

public class User implements Serializable{

private String name;

private String address;

public User(String name, String address) {

super();

this.name = name;

this.address = address;

}

}

public static void main(String[] args) throws InterruptedException, ExecutionException, FileNotFoundException, IOException {

MemcachedClient mcc = null;

try{

// 本地连接 Memcached 服务

mcc = new MemcachedClient(new InetSocketAddress("127.0.0.1", 11211));

System.out.println("Connection to server sucessful.");

}catch(Exception ex){

System.out.println( ex.getMessage() );

}

User u = new User("junwang","qingdao city");

ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream(new File("a.txt")) );

oos.writeObject(u);

Future fo = mcc.set("myuser", 5\*60\*1000, u);

// 查看存储状态

System.out.println("set status:" + fo.get());

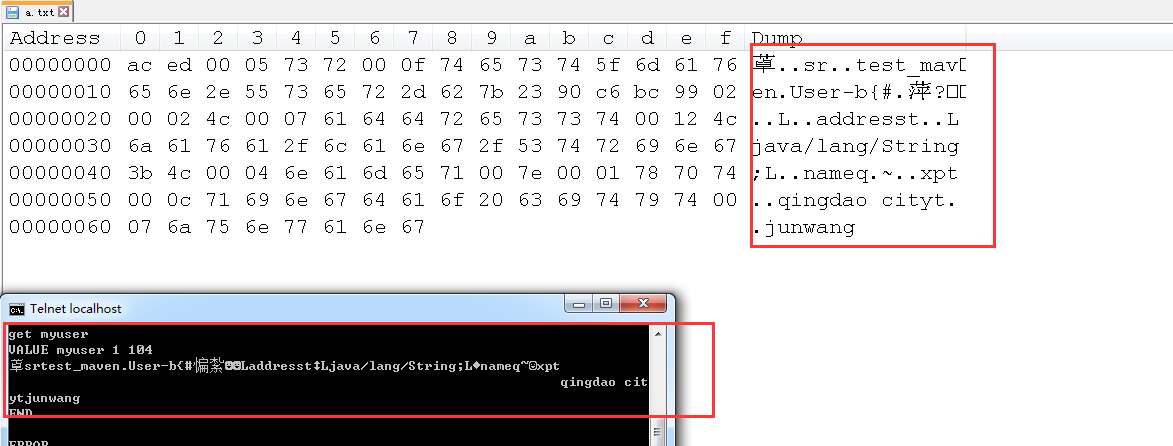
// 输出值

System.out.println("myuser value in cache - " + mcc.get("myuser"));

// 关闭连接

mcc.shutdown();

}



**查看源代码**

查看源代码，可以发现正是将对象序列化，然后保存。证明了我们上述的猜想。

BaseSerializingTranscoder.java

protected byte[] serialize(Object o) {

if (o == null) {

throw new NullPointerException("Can't serialize null");

}

byte[] rv=null;

ByteArrayOutputStream bos = null;

ObjectOutputStream os = null;

try {

bos = new ByteArrayOutputStream();

os = new ObjectOutputStream(bos);

os.writeObject(o);

os.close();

bos.close();

rv = bos.toByteArray();

} catch (IOException e) {

throw new IllegalArgumentException("Non-serializable object", e);

} finally {

CloseUtil.close(os);

CloseUtil.close(bos);

}

return rv;

}

值的存储，都是序列化成字节数组，然后保存

版权声明：本文为博主原创文章，未经博主允许不得转载。