shuaji root业务中使用到了 DigestUtils.md5Hex()方法

在缓存中使用了DigestUtils.md5Hex(String) ---- 这个适用于数据量小

在运营后台使用了DigestUtils.md5Hex(InputStream)

首先理解这个方法的字面含义：

md5Hex：无论是加密的输入 还是解密的输出 还是数字摘要的输出 原始的格式都是byte[]

但是byte[]不能直接输出保存 要变化成字符串进行保存

我们shuaji项目中 有的使用的是Base64的编码 有的使用的本地GBK 有的使用的是UTF-8编码

在这里面 这个使用的二进制的十六进制形式来进行编码

\*\*\*基本的思路2就是 定义一个ByteArrayOutputStream

不断的从InputStream中读出来数据 然后使用digest的update方法把数据不断地更新进去

数据更新完成之后 直接调用digest()方法即可

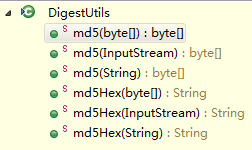
\*\*\*\*分析思路\*\*\*\*\*

Apache的DigestUtils的md5Hex的几种重载形式

对于有固定大小和没有固定大小的数据 表现为byte[], String和InputStream

输出的格式有byte[]和String

所以 输入输出进行排列组合 就有6种重载形式



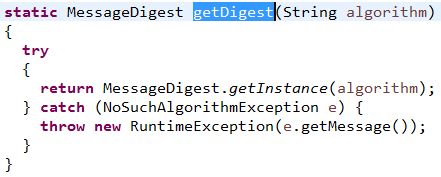
按照输出的类型进行分类 分成两类 byte[]输出类型 和 String输出类型（String类型按照Hex编码）

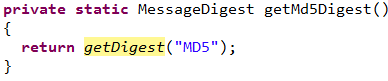
byte[]输出类型有三种重载形式 方法名就是md5

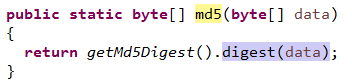
String输出类型也有三种重载形式 方法名就是md5Hex

分析3个基础方法

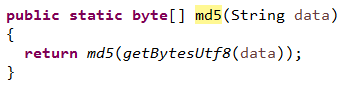
\*\*md5(byte[]) ---- 适合小的数据量的输入

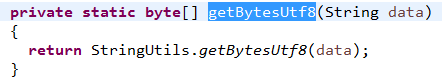


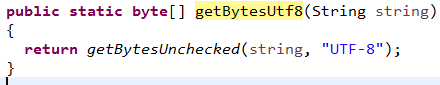


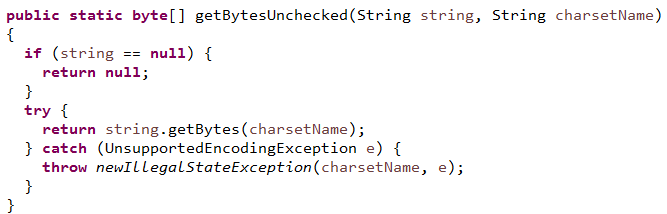


\*\*md5(String) ---- 适合小的数据量的输入

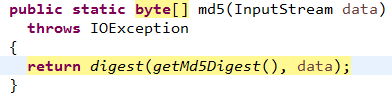


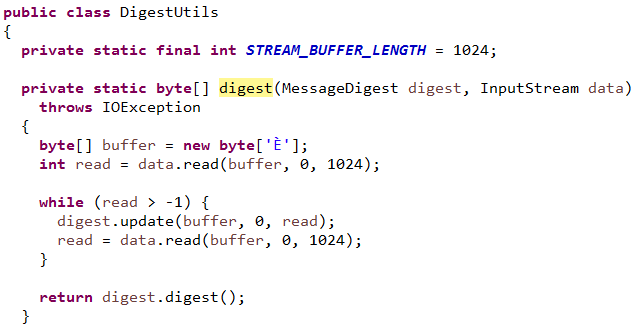






\*\*md5(inputStream)---- 输入大的数据量的输入



 ---- 上面的思路2

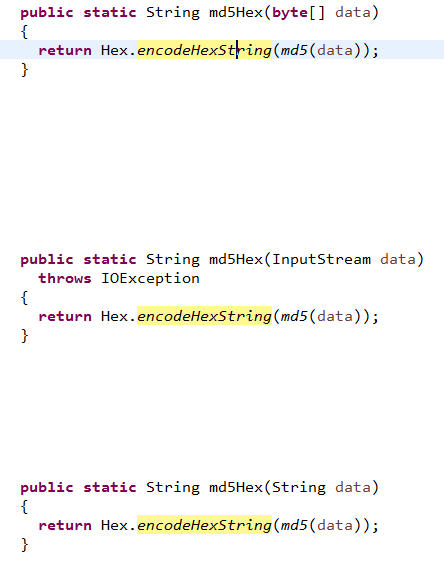
再分析另外3个方法

\*\*md5Hex(byte[])---- 输入大的数据量的输入

\*\*md5Hex(String)---- 输入大的数据量的输入

\*\*md5Hex(InputStream)---- 输入大的数据量的输入

这三个其实都是在最后输出的byte[]的基础上 进行十六进制的编码



这三个方法都先调用各自对应的md5()静态方法 调用完成之后 使用Hex.encodeHexString()进行输出结果的编码即可

