


在本地方法实现代码中创建 java 对象，其中还主要涉及到 jni 和 java 之间中文字符串的的乱码问题。


1.创建 java 对象

首先在 java 端自定义一个 Person 类如下

Java 代码 

```
1. package com.example;
2.
3. public class Person {
4.
5.     public String name;
6.
7.     public int age;
8.
9.     public Person()
10.    {
11.
12.    }
13.     public Person(String name,int age)
14.    {
15.         this.name=name;
16.         this.age=age;
17.    }
18.
19.
20.     public void Desc()
21.    {
22.         System.out.println("姓名: "+this.name+" 年龄: "+this.age);
23.    }
24.
25.
26. }
```

定义本地方法 sayHello 方法，在 sayHello 方法里实现创建 Person 对象，并调用创建对象的 Desc()方法

Java 代码 

```
1. package com.example;
2.
3.
4. public class jni_test {
5.
```


```

6.      //在本地方法 sayHello 里里创建 Person 对象
7.      public native void sayHello();
8.
9.
10.     static{
11.         System.loadLibrary("NativeCode");
12.     }
13.
14.     public static void main(String[] args) {
15.
16.         jni_test temp=new jni_test();
17.
18.         temp.sayHello();
19.     }
20.
21. }

```

jni 中创建 java 对象需要使用 NewObject 方法，jni 里定义构造函数的名称为<init>,返回值为 Void

1.调用默认构造函数创建 Person 对象

Cpp 代码 

```

1.  JNIEXPORT void JNICALL Java_com_example_jni_1test_sayHello(JNIEnv * evn, jobject obj)
2.  {
3.      //获取 java 的 Class
4.      jclass my_class=evn->FindClass("com/example/Person");
5.      //获取 java 的 Person 构造方法 id---构造函数的函数名为<init>, 返回值为 void
6.      jmethodID init_id=evn->GetMethodID(my_class,"<init>","()V");// (类,属性名.签名)
7.      //创建 Person 对象--使用 NewObject 方法
8.      jobject person=evn->NewObject(my_class,init_id);
9.      //为 person 赋值
10.     jfieldID name_id=evn->GetFieldID(my_class,"name","Ljava/lang/String;");
11.     evn->SetObjectField(person,name_id,(evn->NewStringUTF("mike"));
12.
13.     jfieldID age_id=evn->GetFieldID(my_class,"age","I");
14.     evn->SetIntField(person,age_id,20);
15.
16.     //获取 Person 的 Desc 方法 id
17.     jmethodID desc_id=evn->GetMethodID(my_class,"Desc","()V");

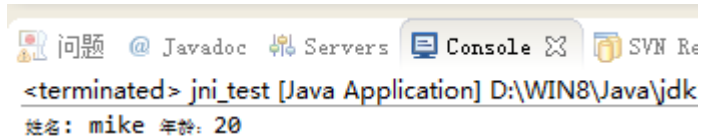
```

```

18.
19.     //调用创建的 person 里的 desc 方法
20.     evn->CallVoidMethod(person,desc_id);
21.
22. }

```

运行结果:




```

<terminated> jni_test [Java Application] D:\WIN8\Java\jdk
姓名: mike 年龄: 20

```

2.调用有参构造方法，构造对象时完成对象属性赋值

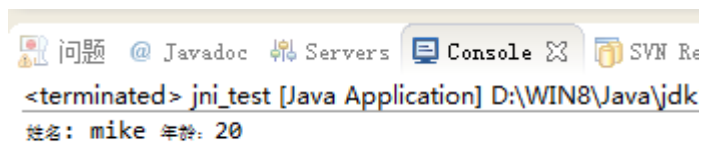
Cpp 代码 

```

1. JNIEXPORT void JNICALL Java_com_example_jni_1test_sayHello(JNIEnv * evn, job
   object obj)
2. {
3.     //获取 java 的 Class
4.     jclass my_class=evn->FindClass("com/example/Person");
5.     //获取 java 的 Person 构造方法 id---构造函数的函数名为<init>, 返回值为 void
6.     jmethodID init_id=evn->GetMethodID(my_class,"<init>","(Ljava/lang/Strin
   g;I)V");// (类,属性名.签名)
7.
8.     //创建 Person 对象--使用 NewObject 方法
9.     jobject person=evn->NewObject(my_class,init_id, (evn->NewStringUTF("mik
   e"),20);
10.
11.    //获取 Person 的 Desc 方法 id
12.    jmethodID desc_id=evn->GetMethodID(my_class,"Desc","()V");
13.
14.    //调用创建的 person 里的 desc 方法
15.    evn->CallVoidMethod(person,desc_id);
16.
17. }

```

运行结果:



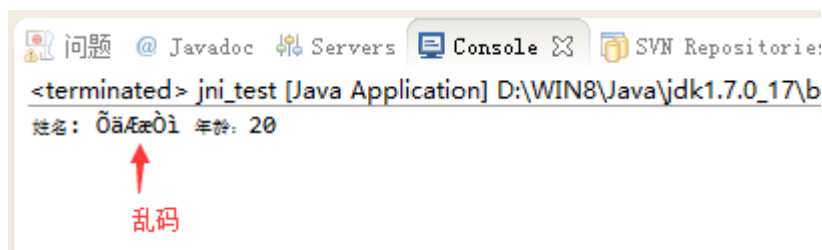
```
<terminated> jni_test [Java Application] D:\WIN8\Java\jdk
姓名: mike 年龄: 20
```

2.jni 和 java 中文乱码

如果把创建 Person 对象改为

```
object person=env->NewObject(my_class,init_id, (env)->NewStringUTF("珍奇异"),20)
```

此时运行结果产生了中文乱码:



```
<terminated> jni_test [Java Application] D:\WIN8\Java\jdk1.7.0_17\b
姓名: 乱码 年龄: 20
      ↑
    乱码
```

乱码产生原因

java 内部是使用 16bit 的 unicode 编码 (UTF-16) 来表示字符串的, 无论中文英文都是 2 字节; jni 内部是使用 UTF-8 编码来表示字符串的, UTF-8 是变长编码的 unicode, 一般 ascii 字符是 1 字节, 中文是 3 字节;

1、java --> c/c++

这种情况中, java 调用的时候使用的是 UTF-16 编码的字符串, jvm 把这个字符串传给 jni, c/c++得到的输入是 jstring, 这个时候, 可以利用 jni 提供的两种函数, 一个是 GetStringUTFChars, 这个函数将得到一个 UTF-8 编码的字符串; 另一个是 GetStringChars 这个将得到 UTF-16 编码的字符串。


2、c/c++ --> java

jni 返回给 java 的字符串，c/c++ 首先应该负责把这个字符串变成 UTF-8 或者 UTF-16 格式，然后通过 NewStringUTF 或者 NewString 来把它封装成 jstring，返回给 java 就可以了。

解决方法

方法一：使用#include <windows.h>头文件函数

导入头文件#include <windows.h> ，此时有利用两个函数来处理字符串

Cpp 代码 

```
1. //将 jstring 类型转换成 windows 类型
2. char* jstringToWindows( JNIEnv *env, jstring jstr )
3. {
4.     int length = (env)->GetStringLength(jstr );
5.     const jchar* jcstr = (env)->GetStringChars(jstr, 0 );
6.     char* rtn = (char*)malloc( length*2+1 );
7.     int size = 0;
8.     size = WideCharToMultiByte( CP_ACP, 0, (LPCWSTR)jcstr, length, rtn,(length*2+1), NULL, NULL );
9.     if( size <= 0 )
10.         return NULL;
11.     (env)->ReleaseStringChars(jstr, jcstr );
12.     rtn[size] = 0;
13.     return rtn;
14. }
15. //将 windows 类型转换成 jstring 类型
16. jstring WindowsToJstring( JNIEnv* env, char* str )
17. {
18.     jstring rtn = 0;
19.     int slen = strlen(str);
20.     unsigned short * buffer = 0;
21.     if( slen == 0 )
22.         rtn = (env)->NewStringUTF(str );
23.     else
24.     {
25.         int length = MultiByteToWideChar( CP_ACP, 0, (LPCSTR)str, slen, NULL, 0 );
26.         buffer = (unsigned short *)malloc( length*2 + 1 );
27.         if( MultiByteToWideChar( CP_ACP, 0, (LPCSTR)str, slen, (LPWSTR)buffer, length ) >0 )
28.             rtn = (env)->NewString( (jchar*)buffer, length );
29.     }
30.     if( buffer )
```

```
31.         free( buffer );
32.     return rtn;
33. }
```

此时把本地完整代码为:

Cpp 代码 ☆

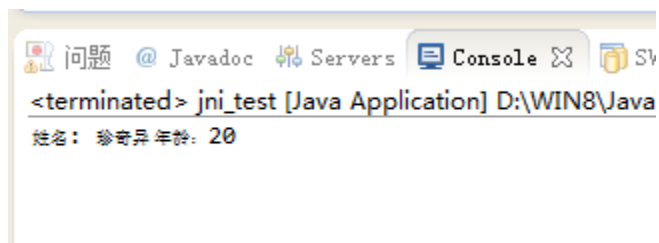
```
1. #include"com_example_jni_test.h"
2. #include<iostream>
3. #include <windows.h>
4. using namespace std;
5.
6. //将 windows 类型转换成 jstring 类型
7. jstring WindowsTojstring( JNIEnv* env, char* str )
8. {
9.     jstring rtn = 0;
10.    int slen = strlen(str);
11.    unsigned short * buffer = 0;
12.    if( slen == 0 )
13.        rtn = (env)->NewStringUTF(str );
14.    else
15.    {
16.        int length = MultiByteToWideChar( CP_ACP, 0, (LPCSTR)str, slen, NUL
        L, 0 );
17.        buffer = (unsigned short *)malloc( length*2 + 1 );
18.        if( MultiByteToWideChar( CP_ACP, 0, (LPCSTR)str, slen, (LPWSTR)buffer, length ) >0 )
19.            rtn = (env)->NewString( (jchar*)buffer, length );
20.    }
21.    if( buffer )
22.        free( buffer );
23.    return rtn;
24. }
25.
26.
27. JNIEXPORT void JNICALL Java_com_example_jni_1test_sayHello(JNIEnv * env, jobject obj)
28. {
29.    //获取 java 的 Class
30.    jclass my_class=env->FindClass("com/example/Person");
31.    //获取 java 的 Person 构造方法 id---构造函数的函数名为<init>, 返回值为 void
32.
33.    jmethodID init_id=env->GetMethodID(my_class,"<init>","(Ljava/lang/String;I)V");// (类,属性名.签名)
```

```

34.     //创建 Person 对象--使用 NewObject 方法
35.     jobject person=evn->NewObject(my_class,init_id, WindowsToJstring(evn,"珍
    奇异"),20);
36.
37.
38.     //获取 Person 的 Desc 方法 id
39.     jmethodID desc_id=evn->GetMethodID(my_class,"Desc","()V");
40.
41.     //调用创建的 person 里的 desc 方法
42.     evn->CallVoidMethod(person,desc_id);
43.
44. }

```

运行结果:



方法二:

完整 C++代码:

Java 代码 ☆

```

1. #include"com_example_jni_test.h"
2. #include<iostream>
3. using namespace std;
4.
5. //检查是否含有中文
6. int isASCII(const char * chp)
7. {
8.     char ch;
9.     jboolean flag= 1;
10.    while(ch = *chp++){
11.        if(ch & 0x80){
12.            flag = 0;
13.            break;
14.        }
15.    }
16.    return flag;
17. }
18.

```

```

19. //jstring to char*
20. char* JstringToPchar(JNIEnv* env, jstring jstr, const char * encoding, jmethodID gmidStringGetBytes)
21. {
22.     char* rtn = NULL;
23.     jstring jencoding;
24.     jencoding=(env)->NewStringUTF(encoding);
25.     jbyteArray barr= (jbyteArray)(env)->CallObjectMethod(jstr, gmidStringGetBytes, jencoding);
26.     jsize alen = (env)->GetArrayLength(barr);
27.     jbyte* ba = (env)->GetByteArrayElements(barr, JNI_FALSE);
28.     if (alen > 0)
29.     {
30.         rtn = (char*)malloc( alen + 1);
31.         memcpy(rtn, ba, alen);
32.         rtn[alen] = 0;
33.     }
34.     (env)->ReleaseByteArrayElements(barr, ba, 0);
35.     return rtn;
36. }
37.
38. //char* to jstring
39. jstring PcharToJstring(JNIEnv* env, const char* pchar, const char * encoding, jclass gStringClass, jmethodID gmidStringInit)
40. {
41.     jstring jencoding;
42.     jbyteArray bytes = (env)->NewByteArray(strlen(pchar));
43.     env->SetByteArrayRegion(bytes, 0, strlen(pchar), (jbyte*)pchar);
44.     jencoding = env->NewStringUTF(encoding);
45.     return (jstring)(env)->NewObject(gStringClass, gmidStringInit, bytes, jencoding);
46.
47. }
48.
49. JNIEXPORT void JNICALL Java_com_example_jni_1test_sayHello(JNIEnv * env, jobject obj)
50. {
51.     //获取 java 的 Class
52.     jclass my_class=env->FindClass("com/example/Person");
53.
54.     //获取 java 的 String 相关方法
55.     jclass str_class=env->FindClass("java/lang/String");
56.     jmethodID Byte_id=env->GetMethodID(str_class, "getBytes", "(Ljava/lang/String;)[B");

```

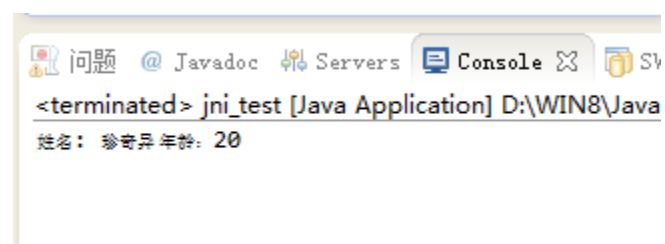


```

57.     jmethodID Strinit_id=evn->GetMethodID(str_class,"<init>","([Ljava/lang/
    String;)V");
58.
59.     //生成防止乱码字符串--结果赋给 result
60.     char *myTest = "珍奇异";
61.     jstring result;
62.     //没有中文的情况
63.     if(isASCII(myTest)) {
64.         result = evn->NewStringUTF(myTest);
65.     }
66.     else{
67.         result = PcharToJstring(evn,myTest,"gbk",str_class,Strinit_id);
68.         char *outbuf;
69.         outbuf = JstringToPchar(evn,result,"utf-8",Byte_id);
70.         result = (evn->NewStringUTF(outbuf);
71.         if(outbuf){
72.             free(outbuf);
73.         }
74.     }
75.     //获取 java 的 Person 构造方法 id--构造函数的函数名为<init>, 返回值为 void
76.     jmethodID init_id=evn->GetMethodID(my_class,"<init>","(Ljava/lang/Strin
    g;I)V");// (类,属性名.签名)
77.
78.     //创建 Person 对象--使用 NewObject 方法
79.     jobject person=evn->NewObject(my_class,init_id, result,20);
80.
81.     //获取 Person 的 Desc 方法 id
82.     jmethodID desc_id=evn->GetMethodID(my_class,"Desc","()V");
83.
84.     //调用创建的 person 里的 desc 方法
85.     evn->CallVoidMethod(person,desc_id);
86.
87. }

```

运行结果：



- [查看图片附件](#)