通过 JNI 从 /vendor 目录访问 .bmp 文件是一种底层方法，可以提高文件读取的效率。以下步骤介绍如何通过 JNI 从特定路径读取 .bmp 文件并将其加载到 Android 应用中。

1. 创建 JNI 代码以读取文件

在 JNI 中，你可以使用 C++ 标准库打开文件，读取 .bmp 内容，并将数据返回给 Java 层。假设你已经配置好了 Android 项目中的 NDK 支持。

JNI 方法

#include <jni.h>

#include

#include

#include <android/bitmap.h>

#include

extern "C"

JNIEXPORT jbyteArray JNICALL

Java_com_example_yourapp_NativeLib_readBmpFile(JNIEnv *env, jobject / this */, jstring path) {
// 将 jstring 转换为 C++ 字符串
const char *cPath = env->GetStringUTFChars(path, nullptr);

// 打开 .bmp 文件
std ifstream file(cPath, std ios::binary);
env->ReleaseStringUTFChars(path, cPath);

// 检查文件是否打开
if (!file.is_open()) {
return nullptr;
}

// 读取文件内容到缓冲区 std vector buffer((std istreambuf_iterator(file)),
std::istreambuf_iterator()); file.close();

// 创建 Java 层的 byte[] 数组并填充数据
jbyteArray result = env->NewByteArray(buffer.size());
env->SetByteArrayRegion(result, 0, buffer.size(), reinterpret_cast<jbyte *>(buffer.data()));

return result;

}

该代码在指定路径打开 .bmp 文件并将其内容读取到一个 byte 数组中。然后返回一个 jbyteArray 给 Java 层。

2. 在 Java 层调用 JNI 方法并加载 Bitmap

在 Java 层，可以调用 JNI 方法来获取 .bmp 文件的数据，然后使用 BitmapFactory.decodeByteArray() 将其转换为 Bitmap。

Java 层代码

```
class NativeLib {
external fun readBmpFile(path: String): ByteArray?

companion object {
init {
System.loadLibrary("your_native_lib")  // 加载 JNI 库
}
}

}

fun loadBitmapFromBmpFile(path: String): Bitmap? {
val nativeLib = NativeLib()
val bmpData = nativeLib.readBmpFile(path) ?: return null

// 将 ByteArray 数据转换为 Bitmap
return BitmapFactory.decodeByteArray(bmpData, 0, bmpData.size)

}
```

3. 调用 JNI 方法并显示 Bitmap

可以在 Activity 中调用该方法从 /vendor 目录加载 .bmp 文件，并将其显示在 ImageView 上：

```
val bmpFilePath = "/vendor/your_image.bmp"

val bitmap = loadBitmapFromBmpFile(bmpFilePath)

imageView.setImageBitmap(bitmap)
```

4. Android.mk / CMakeLists.txt 配置

确保你的 Android.mk 或 CMakeLists.txt 配置正确，以便编译 JNI 代码。这里是一个简单的 CMakeLists.txt 示例：

```
cmake_minimum_required(VERSION 3.4.1)

add_library(your_native_lib SHARED src/main/cpp/native_lib.cpp)

find_library(log-lib log)

target_link_libraries(your_native_lib ${log-lib})
```

注意事项

- 使用 JNI 访问文件时确保文件路径正确，且设备已授予 root 权限。
- 由于直接操作文件，这种方法需确保 .bmp 文件路径的安全性和设备的兼容性。

通过这种方法，你可以使用 JNI 底层访问 /vendor 目录中的 .bmp 文件并将其加载为位图。