1. Xposed

   开源，官网地址 http://repo.xposed.info/

2. Xposed_Demo

   步骤如下：

   (1)导入 lib 文件 XposedBridgeApi-54.jar，在 build.gradle 文件中设置为 provided 模式，使其不参与编译到最终文件中

   (2)AndroidManifest.xml 文件修改

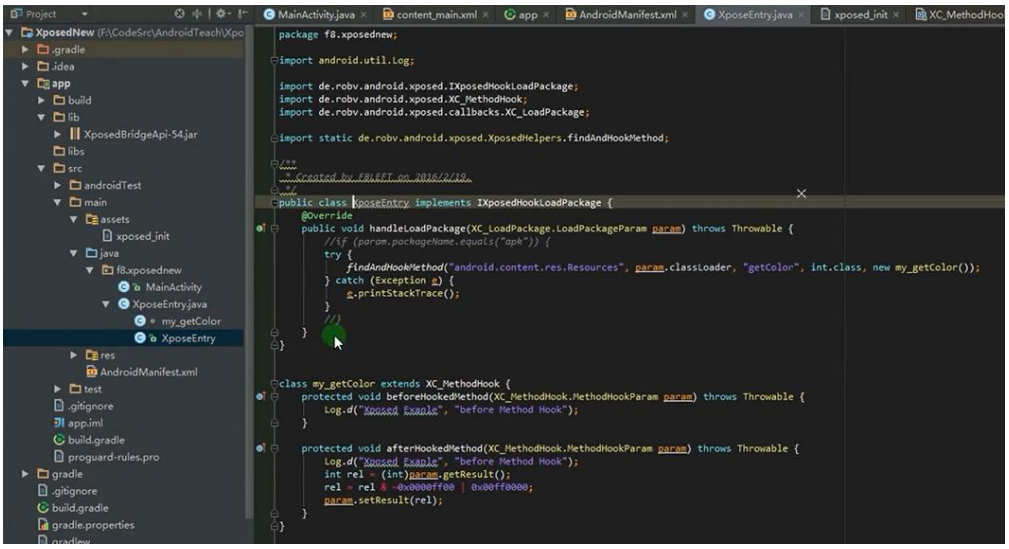
```
<application>
    <meta-data android:name="xposedmodule" android:value="true"/>
    <meta-data android:name="xposeddescription" android:value="模块说明"/>
    <meta-data android:name="xposedminversion" android:value="30"/>
</application>
```

   (3)入口类编写

   新建一个类文件

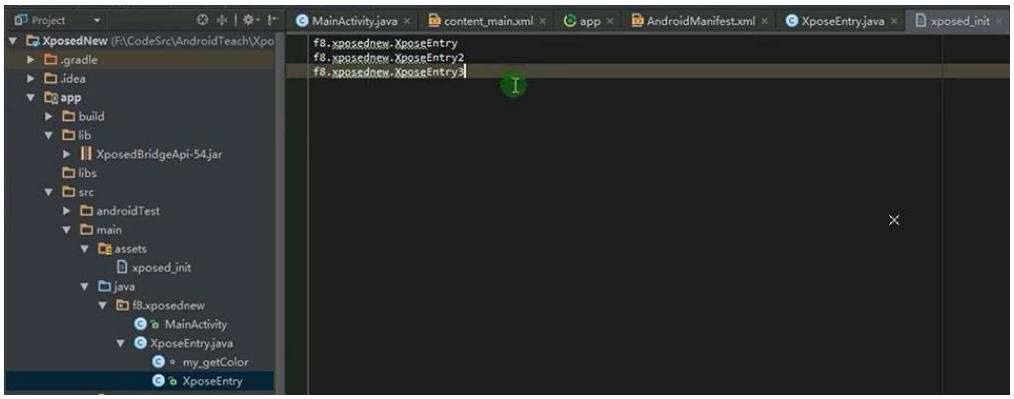
```
public class XposedMain implements IXposedHookLoadPackage{
    public void handleLoadPackage(final LoadPackageParam lpparam) throws Throwabl
{
@Override
public void handleLoadPackage(XC_LoadPackage.LoadPackageParam param) throws
Throwable {
    try {
        findAndHookMethod("android.content.res.Resources", param.classLoader,
"getColor", int.class, new mygetMethod());
    } catch (Exception e) {
        XposedBridge.log(e);
    }
}
}
}
```

```
class mygetMethod extends XC_MethodHook {
    protected void afterHookedMethod(XC_MethodHook.MethodHookParam args) {
        int rel = (int)args.getResult();
        rel = rel & ~0x0000ff00 | 0x00ff0000;
        args.setResult(rel);
    }
    protected void beforeHookedMethod(XC_MethodHook.MethodHookParam args) {
```



(4)设置启动入口

在 assets 文件夹中，新建一个 xposed_init 文件，然后写进入口类的信息，例如 example.xposed.XposedMain



(5)安装激活插件，重启后可看到插件效果

Xposed 是在程序启动的时候同时加载，因此它的函数钩子是区分进程的。如果想对特定进程下钩子的话可以很方便地使用 Xposed。在挂钩函数方面没有 Cydia 那样容易写，但是有一个优点是，对于程序的类和参数，可以通过名字进行下钩子。例如，对应 String 类，类名是 java.lang.String，那么传递的参数可以是”java.lang.String”，与传递 String.class 的效果是一样的。

Xposed 的 Hook 分为函数执行前和函数执行后两个位置，可以分别进行参数修改和结果修改。如果不想进行调用的话，可以在执行前使用 setResult(NULL)函数。

3. classLoader

与 java 上类似，classLoader 就是一个类装载器。与 java 不同的是，classLoader 所加载的是 dex 文件本身。所以通过程序的 classLoader，可以获取程序 dex 文件中定义的所有类和其成员函数。同理，如果一个程序有多个 dex，那么会对应多个 classLoader，特别是使用动态加载的 dex,则需要传递想要的 classLoader 才可以进行数据获取,这点需要注意。