

[SCA] 代码覆盖率使用文档

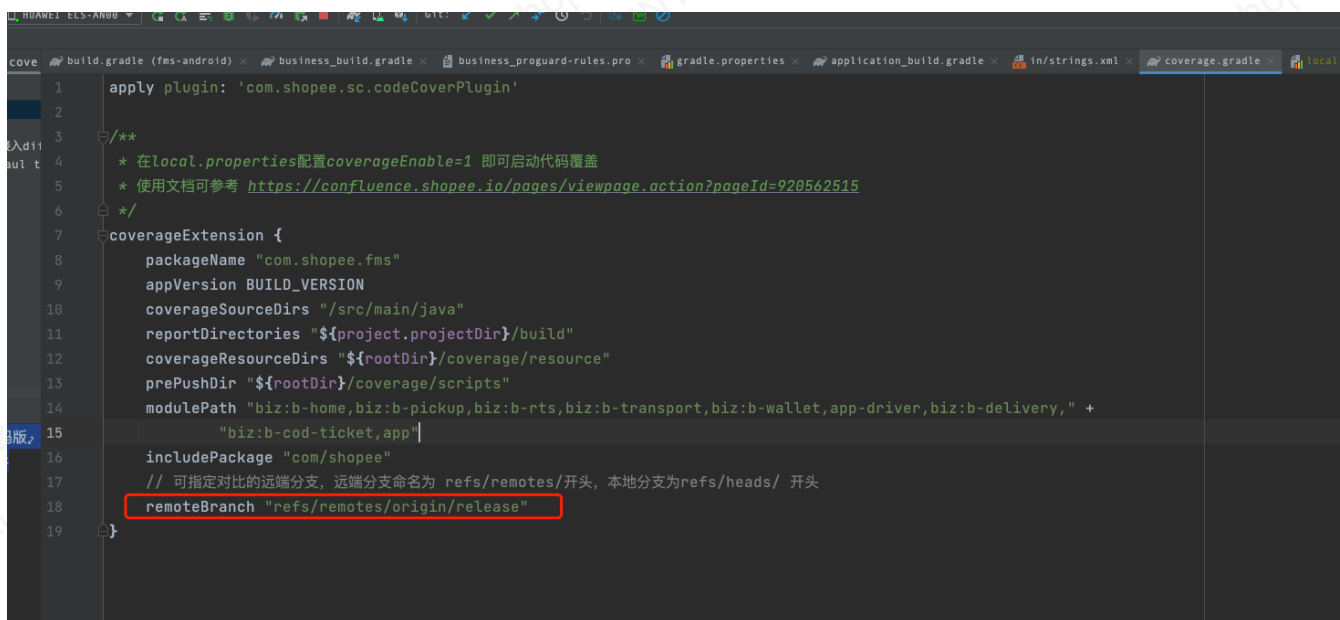
一、代码覆盖率功能介绍：

代码覆盖率工具主要用于在日常开发中，检测源代码和被测试的比例和程度，实现原理简单的来说，就是在源代码的每个节点里面插桩，在运行阶段查看是否运行到代码的每个节点，并不能主动检测bug，关于代码覆盖率的介绍，可[参考文章](#)

目前开发工具支持不同分支的代码覆盖

1. 对比当前分支和特定远端分支代码差异（目前默认设置为release分支）

如需要对比特定远端分支代码，需要在`coverage.gradle`文件中指定分支名，如对比远端分支为release



```
1  apply plugin: 'com.shopee.sc.codeCoverPlugin'
2
3  /**
4   * 在local.properties配置coverageEnable=1 即可启动代码覆盖
5   * 使用文档可参考 https://confluence.shopee.io/pages/viewpage.action?pageId=920562515
6   */
7  coverageExtension {
8      packageName "com.shopee.fms"
9      appVersion BUILD_VERSION
10     coverageSourceDirs "/src/main/java"
11     reportDirectories "${project.projectDir}/build"
12     coverageResourceDirs "${rootDir}/coverage/resource"
13     prePushDir "${rootDir}/coverage/scripts"
14     modulePath "biz:b-home,biz:b-pickup,biz:b-rtts,biz:b-transport,biz:b-wallet,app-driver,biz:b-delivery," +
15             "biz:b-cod-ticket,app"
16     includePackage "com/shopee"
17     // 可指定对比的远端分支，远端分支命名为 refs/remotes/开头，本地分支为refs/heads/ 开头
18     remoteBranch "refs/remotes/origin/release"
19 }
```

二、具体使用步骤

1. 开启覆盖率开关

开启覆盖率开关，开关在`local.properties`文件中（该文件不会上传到仓库，并且在切换分支时不会改变变量），增加`coverageEnable = 1`即为开启代码覆盖率，在生成apk时，会插桩插入覆盖率文件。

```
build.gradle (fms-android) x business_build.gradle x business_proguard-rules.pro x gradle.properties x applica

## This file must *NOT* be checked into Version Control Systems,
# as it contains information specific to your local configuration.
#
# Location of the SDK. This is only used by Gradle.
# For customization when using a Version Control System, please read the
# header note.
#Wed Mar 02 11:49:47 CST 2022
sdk.dir=/Users/jingwei.xie/Library/Android/sdk
ndk.dir=/Users/jingwei.xie/Library/Android/sdk/ndk/21.4.7075529
transifyToken=Token 8187bdf814c8ea180cbd792d51ed821987863813
coverageEnable=1
```

2. 运行apk

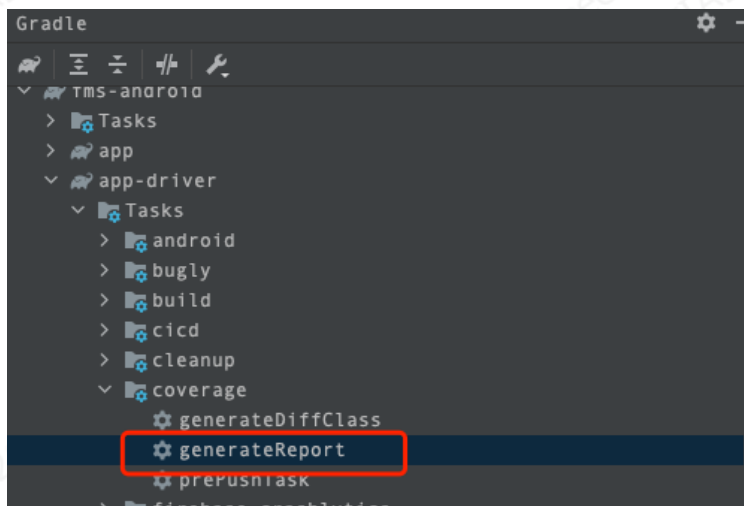
安装apk后，运行应用到差异代码处，当activity被destroy时，会主动生成代码覆盖率文件，同时会输出log

```
2022-01-24 17:52:56.960 16260-19667/com.shopee.fms I/CodeCoverageManager: generateCoverageFile write success /storage/emulated/0/Android/data/com.shopee.fms/cache/JACOCO/5.2
.7/1643017757207_jacoco.exec
```

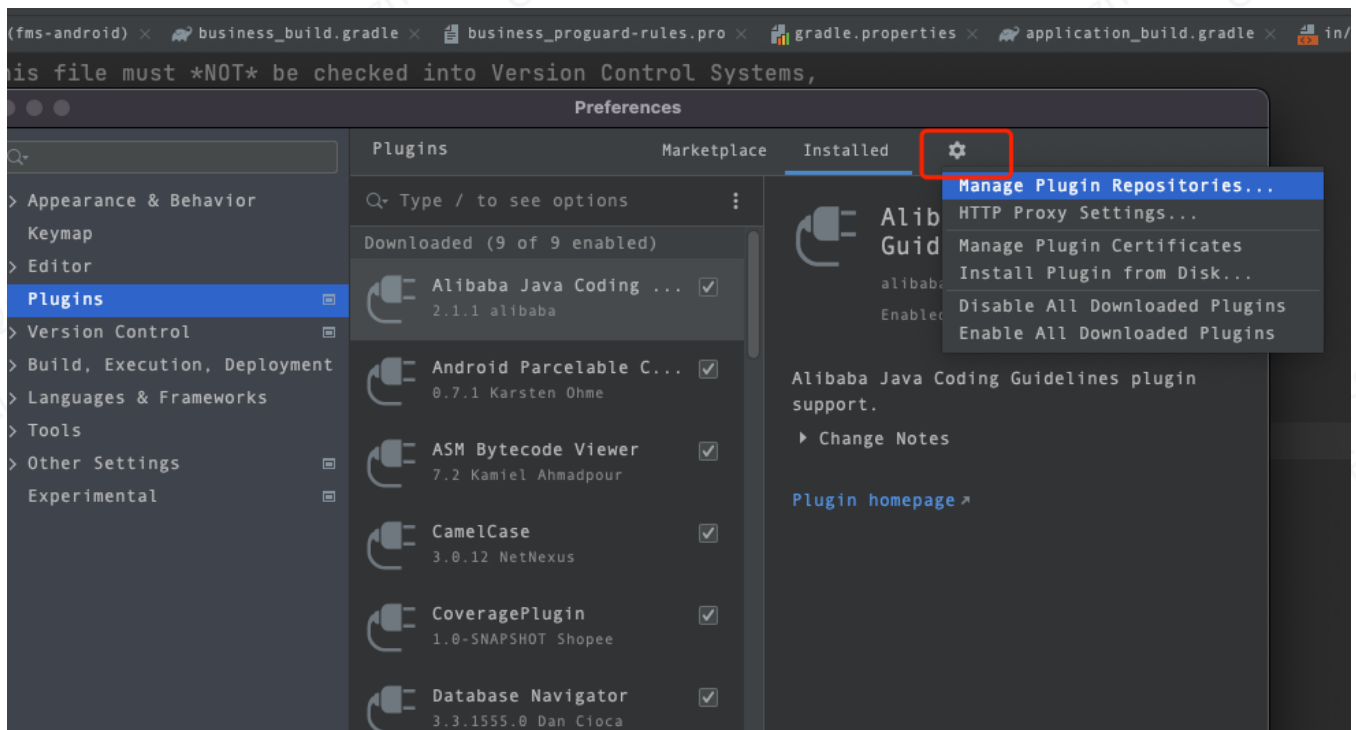
3. 生成覆盖率报告

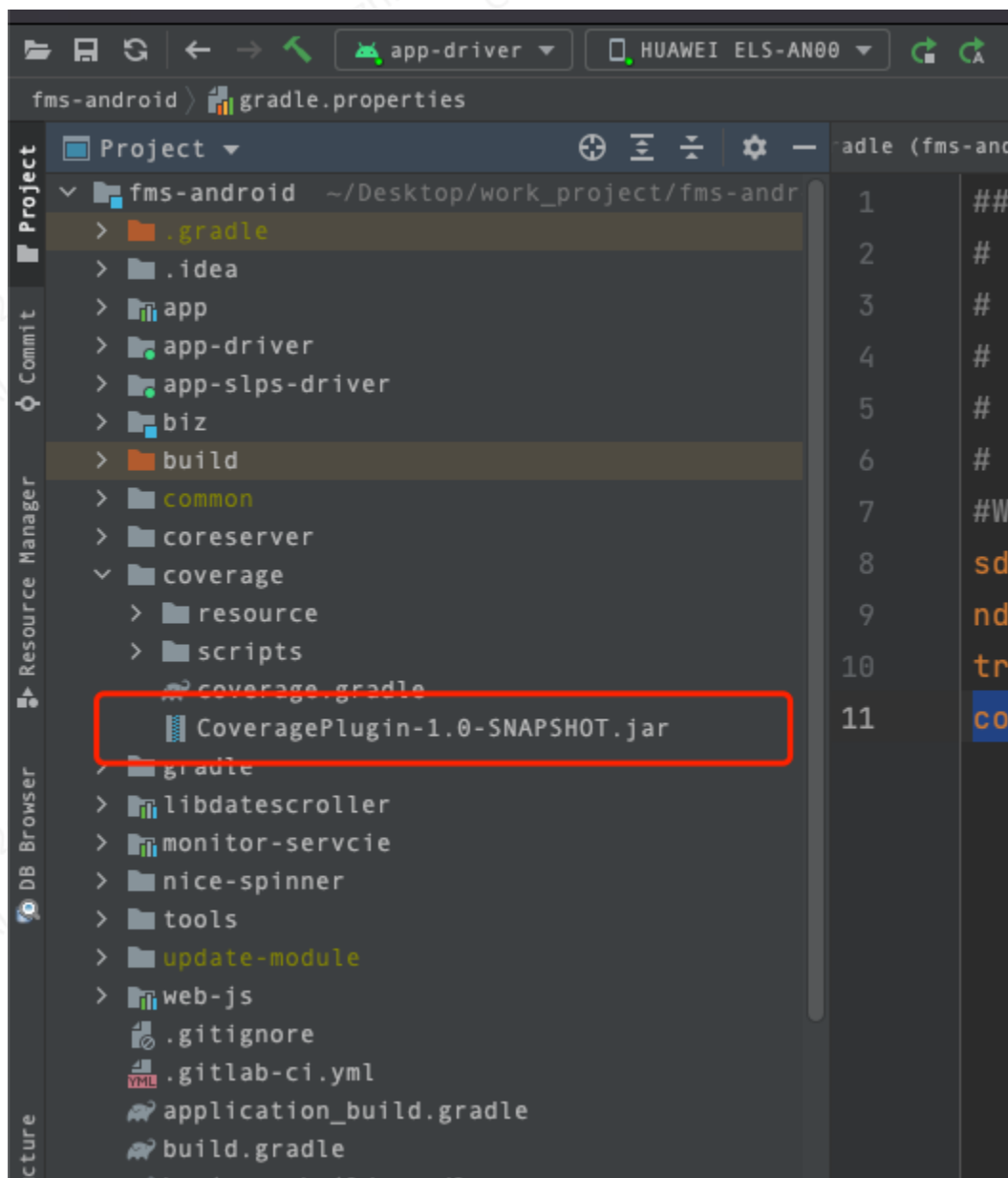
生成覆盖率方式有两种：

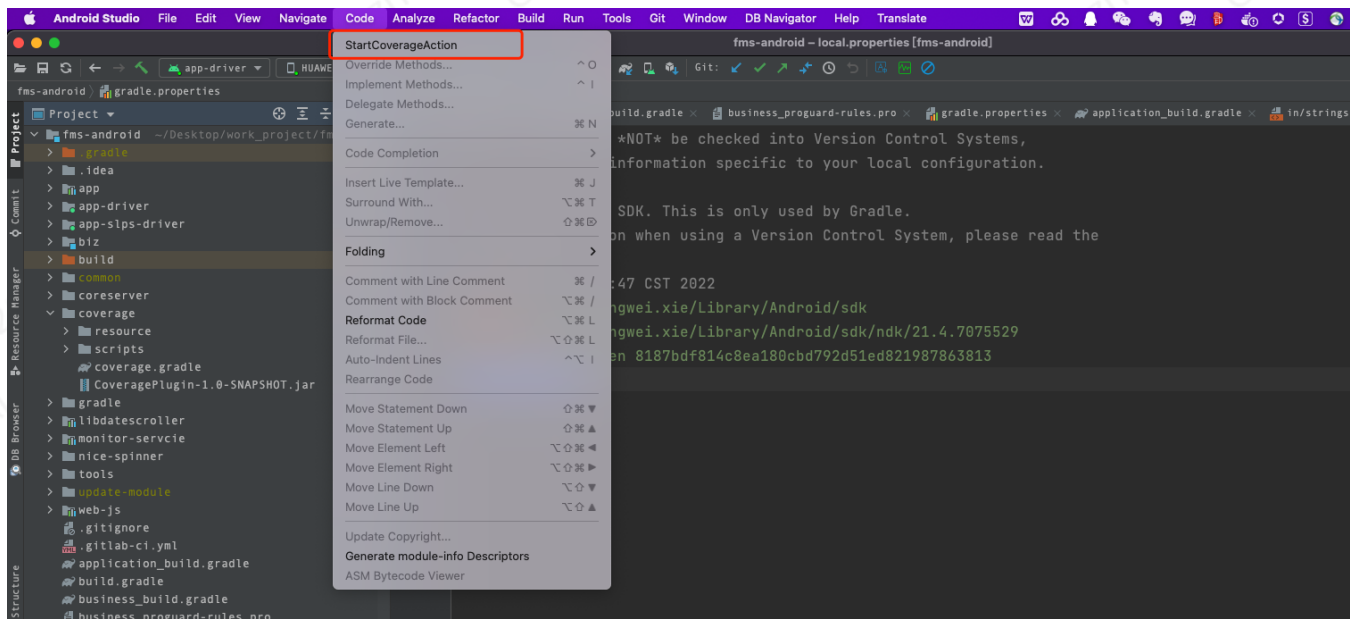
1.启动主项目（app drive主项目为app-driver）的tasks-coverage-generateReport任务，在执行任务过程需要调用adb pull命令，注意此时只能连接一台手机（包括虚拟机）。



2.使用Android studio插件生成报告，本地安装Android studio插件，选择Install plugin from disk，插件在coverage目录下，安装完成后，重新启动Android studio，点击code Menu的start coverage即可启动代码覆盖，由于目前没有区分module，当前生成报告时，会搜索项目中所有的generateTask任务，生成报告时注意查看不同module下是生成报告







在使用时请注意，如果**分支切换或者build目录清空**，直接点击任务，可能会导致出错，需要重新生成apk进行测试

如在执行过程提示adb pull失败，需要查看是否当前手机不支持adb pull命令，run窗口会弹出提示，直接在Terminal窗口运行以下错误命令提示，查看错误

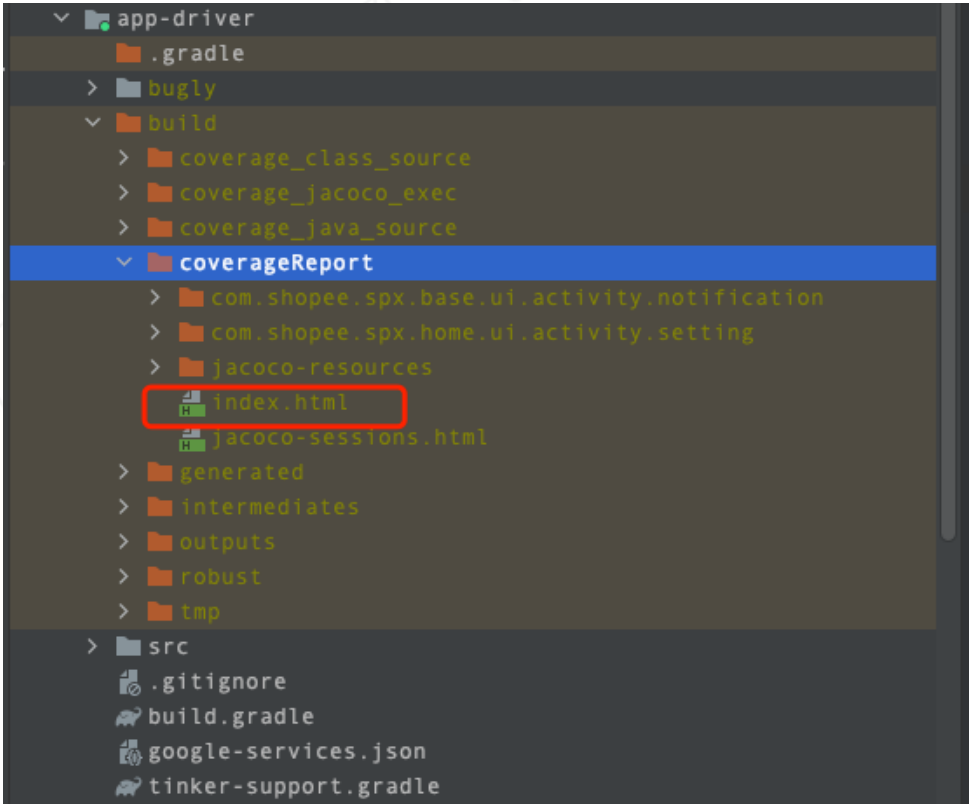
```
CodeCoverage:: Failed to call shell command: adb pull /sdcard/Android/data/com.shopee.fms/cache/JACOC0/5.2.7 /Users/jingwei.xie/Desktop/work_project/fms-android2/fms-android/app-driver/build/coverage_jacoco_exec
```

如执行命令错误为can not create file/direcotry，麻烦查看是否build目录下没有生成apk和class文件，

```
adb: error: cannot create file/directory '/Users/zhangkaiwen/Desktop/SPX-Android/fms-android/app-driver/build/coverage_jacoco_exec': No such file or directory
```

最终生成文件会在主项目的build目录下，可点击跳转查看，路径为coverageReport/index.html，

```
CodeCoverage:: generate jacoco report success /Users/jingwei.xie/Desktop/work_project/fms-android2/fms-android/app-driver/build
```



查看代码覆盖率文件，打开html文件后可以看到代码的覆盖率，其中绿色代码块表示代码已经运行，红色代码块则表示代码没有运行，具体可[参考文章](#)

app-driver											
app-driver											
Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed Cxty	Missed Lines	Missed Methods	Missed Classes			
com.shopee.spx.home.ui.activity.setting	<div><div></div></div>	0%		n/a	1 1	11 11	1 1	1 1			
com.shopee.spx.base.ui.activity.notification	<div><div></div></div>	100%		n/a	0 1	0 6	0 1	0 1			
Total	45 of 61	26%	0 of 0	n/a	1 2	11 17	1 2	1 2			

```
24. import butterknife.BindView;
25. import butterknife.ButterKnife;
26. import butterknife.Unbinder;
27.
28. /**
29.  * 通知列表页
30.  * <p>
31.  * Created by honggang.xiong on 2019-12-30.
32.  */
33. public class NotificationListActivity extends BaseActivity {
34.
35.     @BindView(R2.id.tab)
36.     SlidingTabLayout mSlideTab;
37.     @BindView(R2.id.view_pager)
38.     ViewPager mViewPager;
39.
40.     private Unbinder mUnbinder;
41.     private String[] mTabTitles;
42.
43.     public static void navigate(Context context) {
44.         Intent intent = new Intent(context, NotificationListActivity.class);
45.         context.startActivity(intent);
46.     }
47.
48.     @Override
49.     protected void onCreate(Bundle savedInstanceState) {
50.         super.onCreate(savedInstanceState);
51.         setContentView(R.layout.activity_notification_list);
52.         mUnbinder = ButterKnife.bind(this);
53.         initView();
54.         Logger.d("sadox", "afdoad");
55.     }
56.
```

```

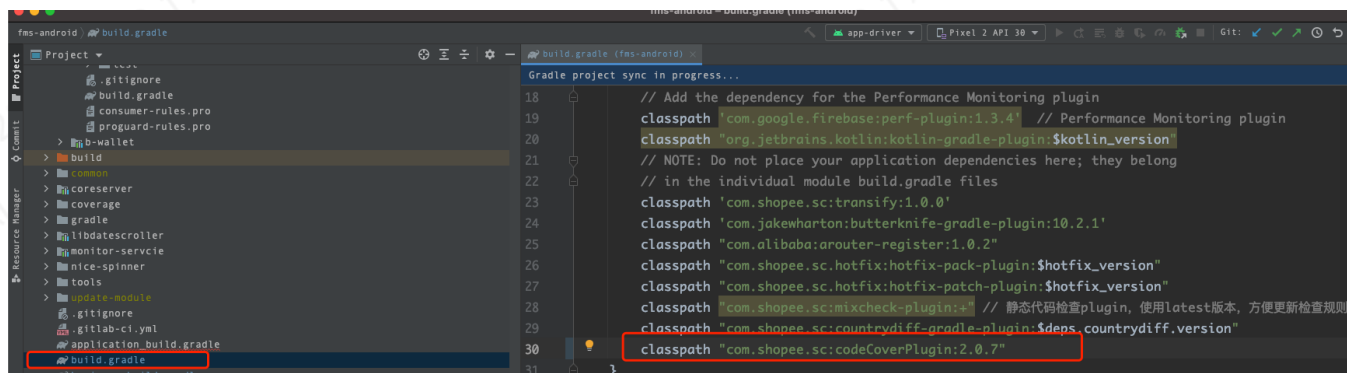
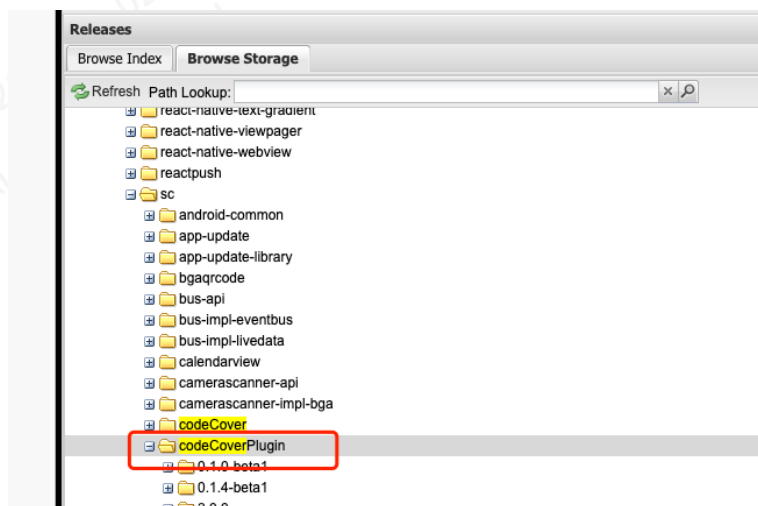
29. /**
30.  * Created by zheng.zeng on 2020/11/10
31.  */
32. public class SettingActivity extends BaseActivity implements ConfigurableNewMenuContract.View {
33.
34.     private static final String TAG = SettingActivity.class.getSimpleName();
35.     private RecyclerView mRecyclerView;
36.     private MenuItemAdapter mMenuAdapter = new MenuItemAdapter(true);
37.     private ConfigurableNewMenuPresenter mPresenter = new ConfigurableNewMenuPresenter(this);
38.     private SettingViewModule mViewModule;
39.
40.     @Override
41.     protected void onCreate(Bundle savedInstanceState) {
42.         super.onCreate(savedInstanceState);
43.         setContentView(R.layout.activity_setting);
44.         mViewModule = new ViewModelProvider(this).get(SettingViewModule.class);
45.         ScCommonTitle title = findViewById(R.id.common_title);
46.         title.setElevation(0);
47.         mRecyclerView = findViewById(R.id.recycler_view);
48.         mRecyclerView.setAdapter(mMenuAdapter);
49.         mPresenter.subscribe(this);
50.         mPresenter.getSettingMenuList();
51.         LogUtils.i("asdfjk", "soso");
52.     }
53.
54.     @Override
55.     protected void onDestroy() {
56.         super.onDestroy();

```

三、常见问题：

1. 插件没有更新到最新版本

如出现覆盖率插件没有更新的问题，需要手动指定覆盖率插件版本，修改文件位置在build.gradle中，指定版本为最新版本，最新版本可在<https://maven.garenanow.com/nexus/#view-repositories/releases-browserstorage> 查阅，插件路径为release-com-shopee-sc-codeCoverPlugin目前最新版本为**2.0.10**



2. 提示错误Caused by: java.lang.ArrayIndexOutOfBoundsException，这个原因可能是生成apk的代码和生成报告检测的代码不一致，需要卸载apk后重新安装

```

at java.base/java.util.Optional.orElseGet(Optional.java:267)
Caused by: java.lang.ArrayIndexOutOfBoundsException: Index 113 out of bounds for length 113
at com.shopee.sc.coverage.jacoco.core.internal.analysis.InstructionsBuilder.addProbe(InstructionsBuilder.java:150)
at com.shopee.sc.coverage.jacoco.core.internal.analysis.MethodAnalyzer.visitProbe(MethodAnalyzer.java:156)
at com.shopee.sc.coverage.jacoco.core.internal.flow.MethodProbesAdapter.visitLabel(MethodProbesAdapter.java:93)
at org.objectweb.asm.tree.LabelNode.accept(LabelNode.java:68)
at com.shopee.sc.coverage.jacoco.core.internal.analysis.MethodAnalyzer.accept(MethodAnalyzer.java:53)
at com.shopee.sc.coverage.jacoco.core.internal.analysis.ClassAnalyzer$1.accept(ClassAnalyzer.java:109)
at com.shopee.sc.coverage.jacoco.core.internal.flow.ClassProbesAdapter$2.visitEnd(ClassProbesAdapter.java:101)
at org.objectweb.asm.ClassReader.readMethod(ClassReader.java:1485)
at org.objectweb.asm.ClassReader.accept(ClassReader.java:711)
at org.objectweb.asm.ClassReader.accept(ClassReader.java:394)
at com.shopee.sc.coverage.jacoco.core.analysis.Analyzer.analyzeClass(Analyzer.java:149)
at com.shopee.sc.coverage.jacoco.core.analysis.Analyzer.analyzeClass(Analyzer.java:165)
... 125 more

```

3.项目构建时提示compiled by a more version of the java runtime class file version，原因在于corelib在构建时使用的是JAVA11的版本，与当前android studio版本不一致，解决方法需要升级Android studio版本到最新，设置为Android 的JDK版本即可（如下图所示）

