Bugly Android 热更新常见问题

Q：之前使用Tinker怎么切换过来使用Bugly？

A：Bugly使用源码集成Tinker，如果之前集成过Tinker，你需要注释掉以下配置：

```
//    compile("com.tencent.tinker:tinker-android-lib:${TINKER_VERSION}")
 { changing = true }
//    provided("com.tencent.tinker:tinker-android-
anno:${TINKER_VERSION}") { changing = true }
```

插件配置不需要更改，只需要加上我们Bugly额外的tinker-support插件即可：

```
      // tinkersupport插件（1.0.3版本无需配置tinker插件）
      classpath "com.tencent.bugly:tinker-support:latest.release"
```

Q： 基线版本表示什么意思?

A：表示你需要修复apk的版本，比如你已经上线了某个版本的apk，你需要用一个唯一的tinkerId来标识这个版本，而补丁包也是基于这个版本打的。

Q：打一个补丁包需要改哪些东西？

A：

> 1. 修复bug的类、修改资源
> 2. 修改oldApk配置
> 3. 修改tinkerId

Q：tinkerId该怎么填？

A：在运行过程中，我们需要验证基准apk包的tinkerId是否等于补丁包的tinkerId。这个是决定补丁包能运行在哪些基准包上面，一般来说我们可以使用git版本号、versionName等等。

Q：两次传入的tinkerId是否一样？

A：不一样的，编译补丁包时，tinker会自动读取基准包AndroidManifest的tinkerId作为package_meta.txt中的TINKER_ID。将本次编译传入的tinkerId，作为package_meta.txt的NEW_TINKER_ID。

Q. 为什么我上传补丁提示我“未匹配到可用补丁的App版本”？

A：如果你的基线版本没有上报过联网，基于这个版本生成的补丁包就无法匹配到，请检查你的基线版本配置是否正确。

Q. 我该上传哪个补丁？patch目录跟tinkerPatch目录下的补丁有什么区别吗？

A：你必须要上传build/outputs/patch目录下的补丁包，

Q：我以前的是Bugly SDK，现在集成升级SDK会有什么影响？

A：不会有影响的，升级SDK内置Bugly功能模块，你只需要将初始化方法改为统一的`Bugly.init(getApplicationContext(), "注册时申请的APPID", false);`即可。

Q：你们是怎么定义开发设备的？

A：我们会提供接口`Bugly.setIsDevelopmentDevice(getApplicationContext(), true);`，我们后台就会将你当前设备识别为开发设备，如果设置为false则非开发设备，我们会根据这个配置进行策略控制。

Q：如果我配置了升级策略，又配置了补丁策略，会是怎样的效果？

A：升级策略优先级会高于补丁策略，后台会优先下发升级策略。毕竟你都要升级了，热更新只是帮助你修复bug而已。

Q：我只想使用热更新，不想使用升级？

A：热更新是包含在升级SDK里面的，你可以不配置任何升级策略，只需按照热更新文档集成即可。

Q：是否支持加固模式？

A：**tinker 1.7.5版本以下**是支持加固模式的，但需要你回退到Qzone方案，将`usePreGeneratedPatchDex`设置为true。

```
// dex相关配置项
dex {
    dexMode = "jar" // 可选，默认为jar
    usePreGeneratedPatchDex = true // 可选，默认为false
    pattern = ["classes*.dex",
               "assets/secondary-dex-?.jar"]
    // 必选
    loader = ["com.tencent.tinker.loader.*",
              "com.tencent.bugly.hotfix.SampleApplication",
        ]
}
```

但要注意Tinker官方的提示：

> 是否提前生成dex，而非合成的方式。这套方案即回退成Qzone的方案，对于需要使用**加固或者多flavor打包(建议使用其他方式生成渠道包)的用户可使用**。但是这套方案需要插桩，会造成**Dalvik下性能损耗以及Art补丁包可能过大的问题，务必谨慎使用**。另外一方面，这种方案在Android N之后可能会产生问题，建议过滤N之后的用户。

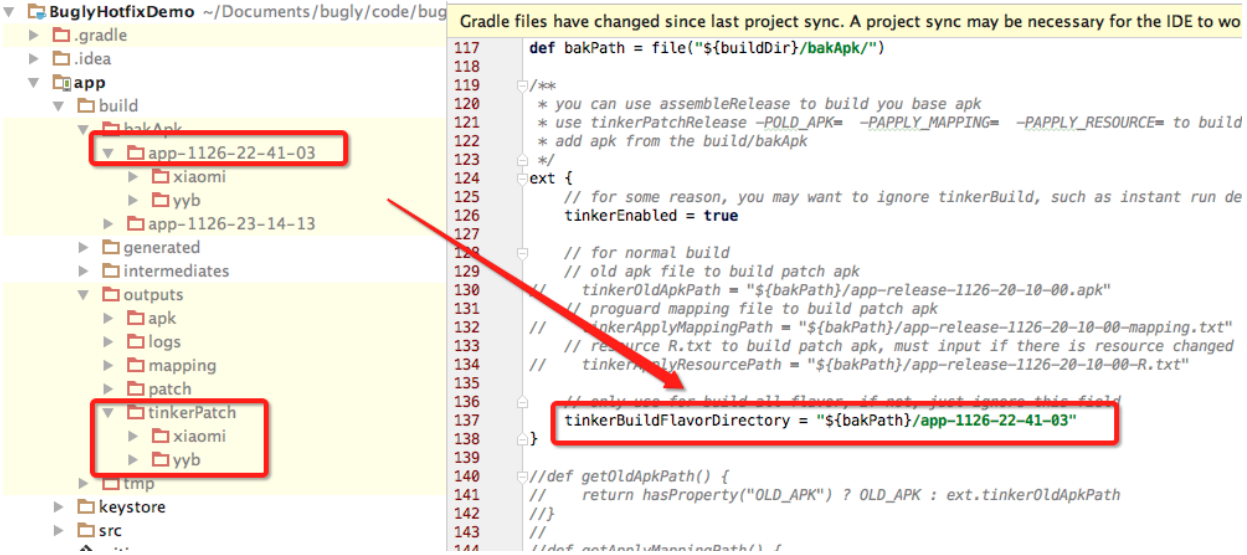
Q：是否支持打多Flavor的patch包

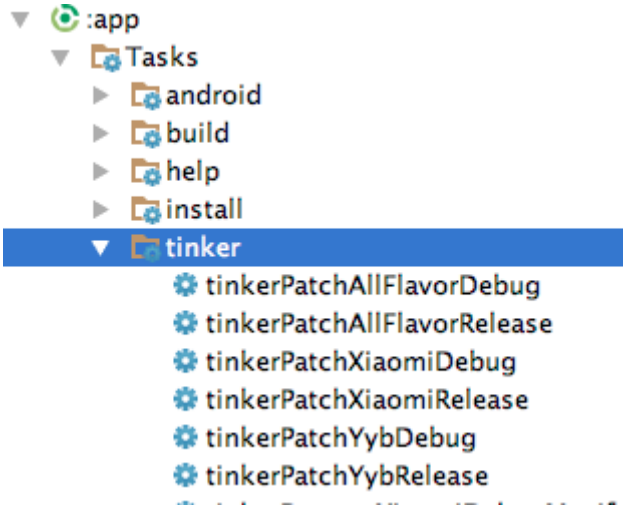
A：支持的。你需要配置productFlavor（示例）：

```
    productFlavors {
        xiaomi {
            applicationId 'com.tencent.bugly.hotfix.xiaomi'
        }

        yyb {
            applicationId 'com.tencent.bugly.hotfix.yyb'
        }
    }
```

打flavor包，只需要配置构建flavor的目录，其他字段不需要填写（执行tinkerPatchAllFalvorRelease就可以得到所有flavor的包）：





Q： 热补丁安全性，以及SDK针对应用补丁失败？

A：Bugly采用HTTPS来保证信息传输的安全性，Bugly会针对下载的补丁包进行签名计算来保证跟下发策略补丁包签名的一致性，如果补丁应用失败，我们会将补丁包直接清除，下次启动你还是可以拿到补丁策略信息。

Q： 是不是每次发版都要保留基准包、混淆配置文件、资源Id文件？

A：当然啦，你不保存基准包，我们打补丁怎么知道要基于哪个版本打补丁？所以建议大家每次发版注意保存基准apk包，还有对应编译生成的mapping文件和R.txt文件。

Q：完整的测试流程是怎样的？

A：

- 打基准包安装并上报联网（注：填写唯一的tinkerId）
- 对基准包的bug修复（可以是Java代码变更，资源的变更）
- 修改基准包路径、填写补丁包tinkerId、mapping文件路径、resId文件路径
- 执行tinkerPatchRelease打Release版本补丁包
- 选择app/build/outputs/patch目录下的补丁包并上传（注：不要选择tinkerPatch目录下的补丁包，不然上传会有问题）
- 编辑下发补丁规则，点击立即下发
- 重启基准包，请求补丁策略（SDK会自动下载补丁并合成）
- 再次重启基准包，检验补丁应用结果

Q: 集成热更新之后出现以下错误

> Error:A problem occurred configuring project ':app'.
> Failed to notify project evaluation listener.
> Tinker does not support instant run mode, please trigger build by assembleDebug or disable instant run in 'File->Settings…'.
> can't find tinkerProcessDebugManifest, you must init tinker plugin first!

A： tinker不支持instant run模式，你需要找到File->Settings->Build,Execution,Deployment->instant run并关闭，日常调试可以tinker关闭来使用instant run。

Q: 日常调试需要使用instant run，怎么关闭tinker

A：这里分两种情况:

使用反射Application方式接入：可以直接在build.gradle中将apply from: 'tinker-support.gradle'注释掉。

改造Application方式接入：先将tinkerSupport中overrideTinkerPatchConfiguration设置为false，然后将tinkerPatch中的tinkerEnable设置为false。

Q：如果我打了多个补丁包，比如先上传了补丁A已经下发到了设备中并且修复，如果我又打了一个补丁，这种情况会怎样？

A：如果你的基于基线版本打了多个补丁包，并且上传了多个，我们会以你最后上传的补丁为准，就是说后面的补丁会覆盖前面的补丁。