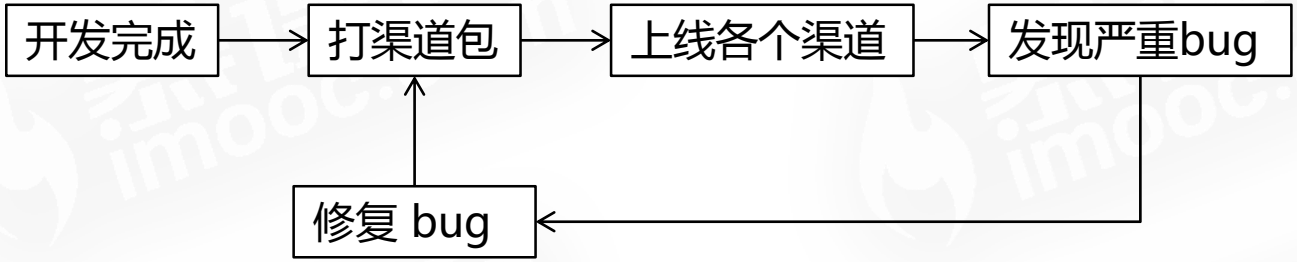
传统开发流程存在的问题



客观因素

- 发版渠道多
- 发版周期长
- 需要用户更新

→

- 修复 bug 成本极高
- 研发心理压力大

如何解决？

我们的诉求

动态修复已发布的app的bug

不需要重新发版

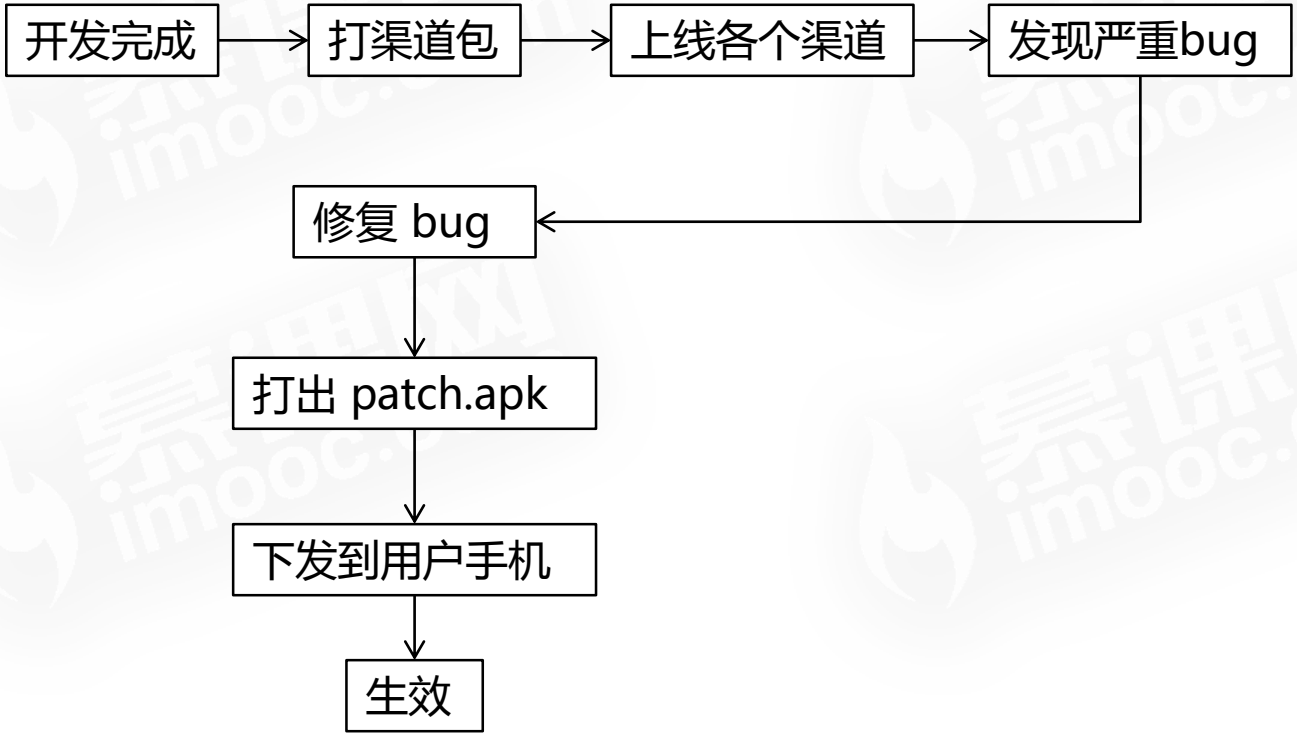
用户无感知

热修复！

什么是热修复

- 用户无感知
- 对线上任何一个版本进行下发代码，修改原有的执行流程
- 可用于修复 bug或者新增功能

热修复执行流程



目前有哪些可选方案

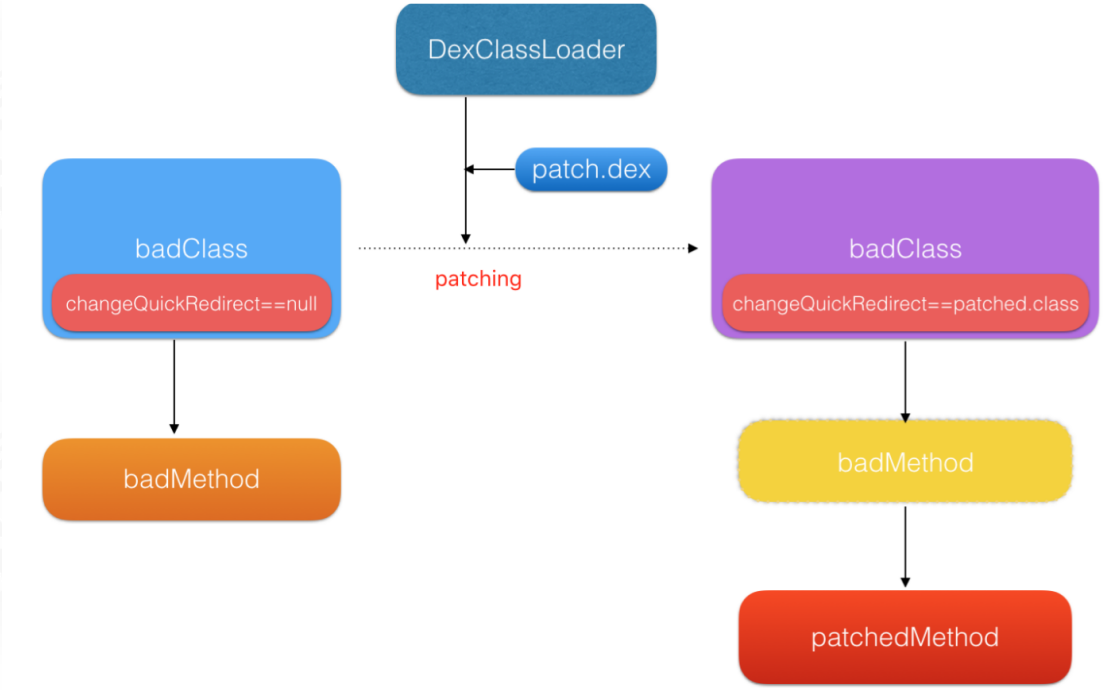
- AndFix (早期方案，阿里，基于 JNI，已开源，不建议使用)
- QQ 超级补丁技术（腾讯，有开发者按照原理开发开源，不建议使用）
- Sophix (阿里，商业方案，可第三方接入)

目前有哪些可选方案

- Robust（美团，已开源）
- Amigo（饿了么，已开源）
- Tinker（微信，已开源）
- Bugly（基于 Tinker，商业方案）

热修复的原理(轻量级)

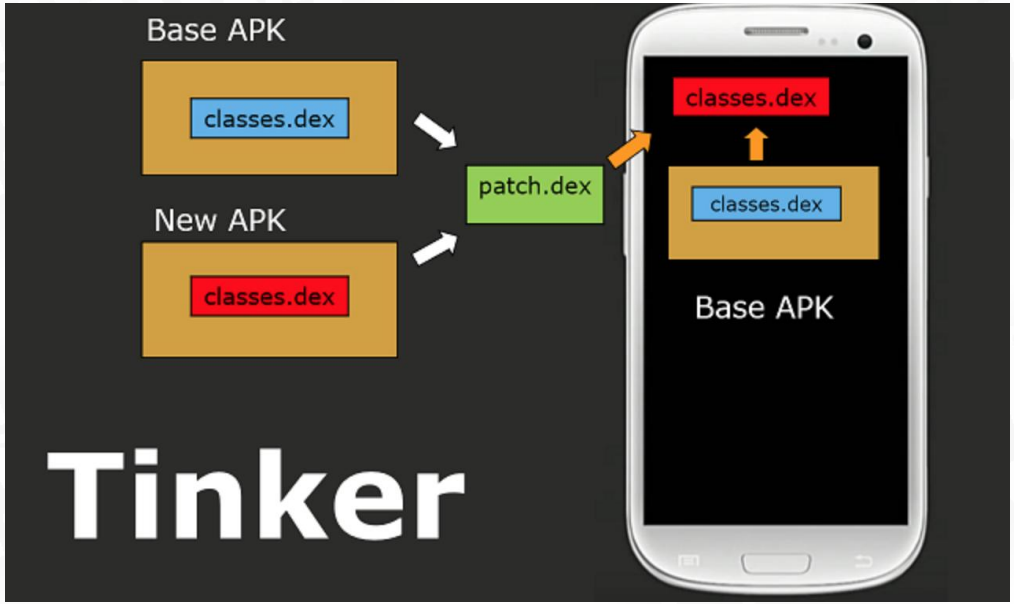
以美团Robust 为代表



来源：https://tech.meituan.com/android_robust.html

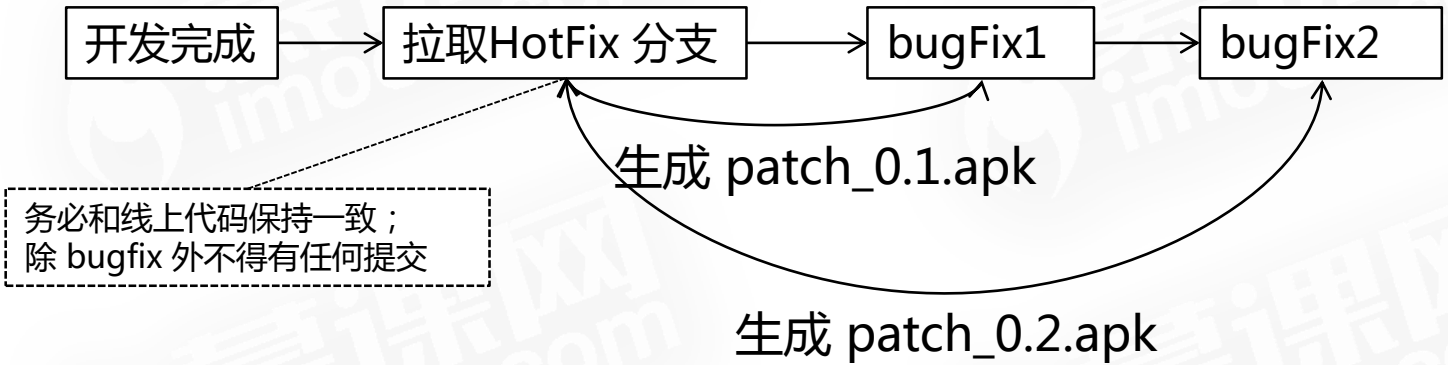
热修复的原理(全量合成)

以Tinker,sophix为代表

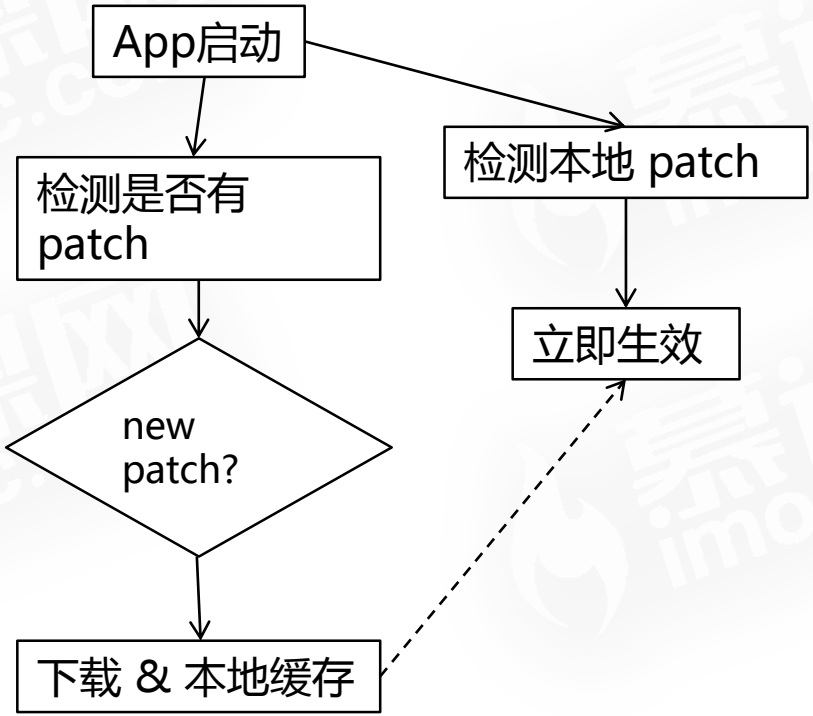


来源：https://github.com/Tencent/tinker

热修复对开发分支的影响



下发 patch.apk的执行流程



案例1：快速接入热修复（bugly）

- 打开bugly 官网，注册项目
- 编写项目
- 为项目计入 bugly 热修复能力
- 修改代码、资源，生成 patch.apk，上线patch.apk
- 测试热修复生效

案例2：通过开源项目使用热修复（Tinker）

- 编写项目
- 接入 Tinker
- 修改代码、资源，生成 patch.apk，上线 patch.apk
- 测试热修复生效（本地测试）

热修复存在的问题

- 自研成本高
- 兼容性问题多（兼容多个 Android 版本，ROM 修改）
- 性能影响
- Android 9.0 ?

Let’s code