Flutter编译原理探讨

WeGene移动开发团队—邓积艺

Framework
Dart

Material
Cupertino
Widgets
Rendering
Animation
Painting
Gestures
Foundation

Engine
C/C++

Service Protocol
Composition
Platform Channels
Dart Isolate Setup
Rendering
System Events
Dart Runtime Mgmt
Frame Scheduling
Asset Resolution
Frame Pipelining
Text Layout

Embedder
Platform-specific

Render Surface Setup
Native Plugins
App Packaging
Thread Setup
Event Loop Interop

- Flutter 架构采用分层设计，从下到上分为三层，依次为：Embedder、Engine和Framework。

- Embedder是操作系统适配层，实现了渲染Surface设置，线程设置，以及平台插件等平台相关特性的适配。

- Engine层主要包含Skia、Dart和Text，实现了Flutter的渲染引擎、文字排版、事件处理和Dart运行时等功能。Skia和Text为上层接口提供了调用底层渲染和排版的能力，Dart则为Flutter提供了运行时调用Dart和渲染引擎的能力。而Engine层的作用，则是将他们组合起来，从他们生成的数据中实现视图渲染。

- Framework层则是一个用Dart实现的UI SDK，包含了动画、图形绘制和手势识别等功能。

几个疑问

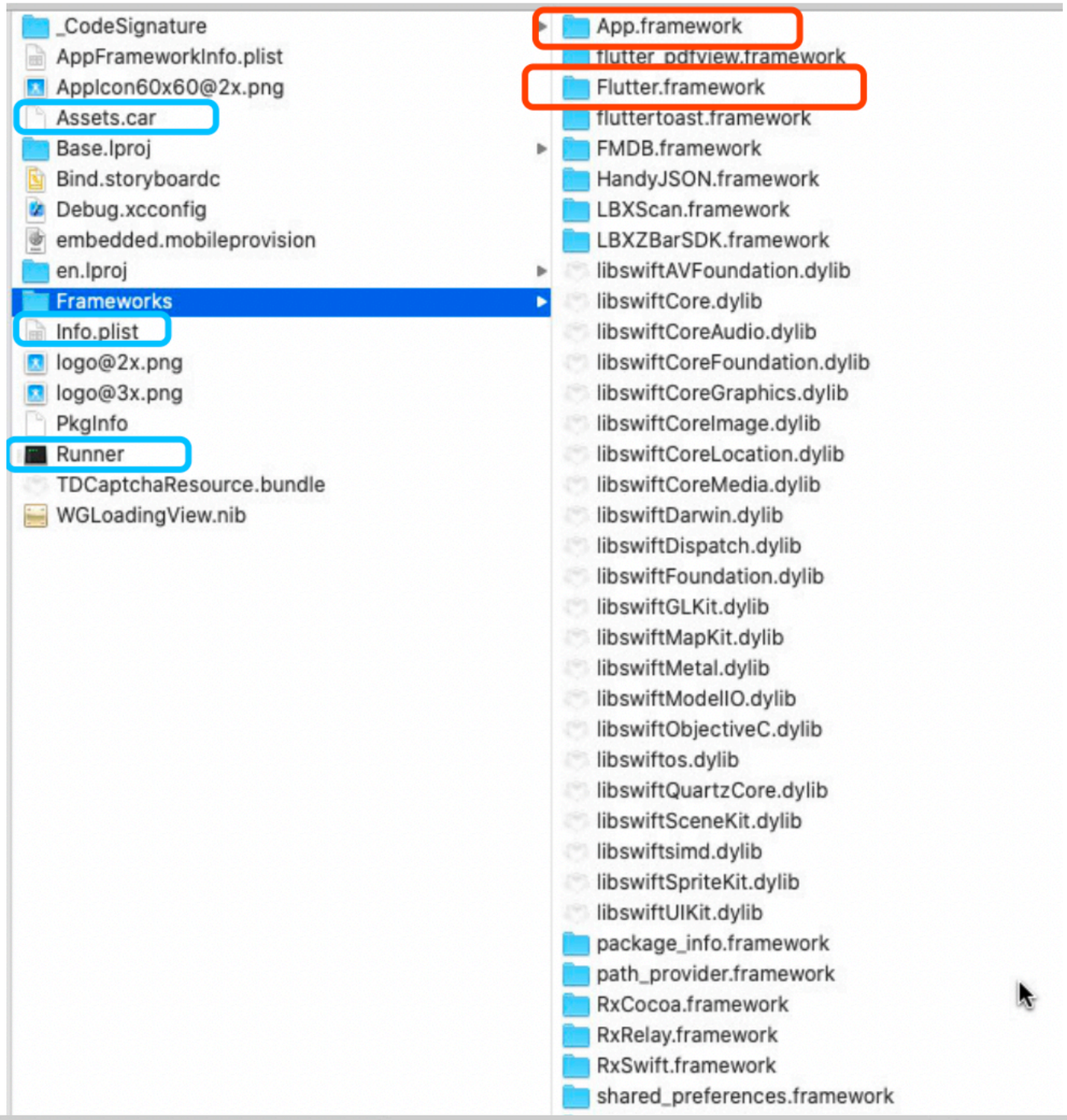
- flutter APP编译后的产物是什么（跟原生的有何不同）？
- debug和release模式有什么不同（hot reload是如何做到的）？
- 如何调试和优化flutter APP?

iOS端 release模式

跟原生APP没有太大区别

Frameworks文件夹主要多了App.framework和Flutter.framework

App.framework里flutter_assets文件夹存放了flutter里引用到的资源文件



安卓端 release模式

跟原生APP基本一致

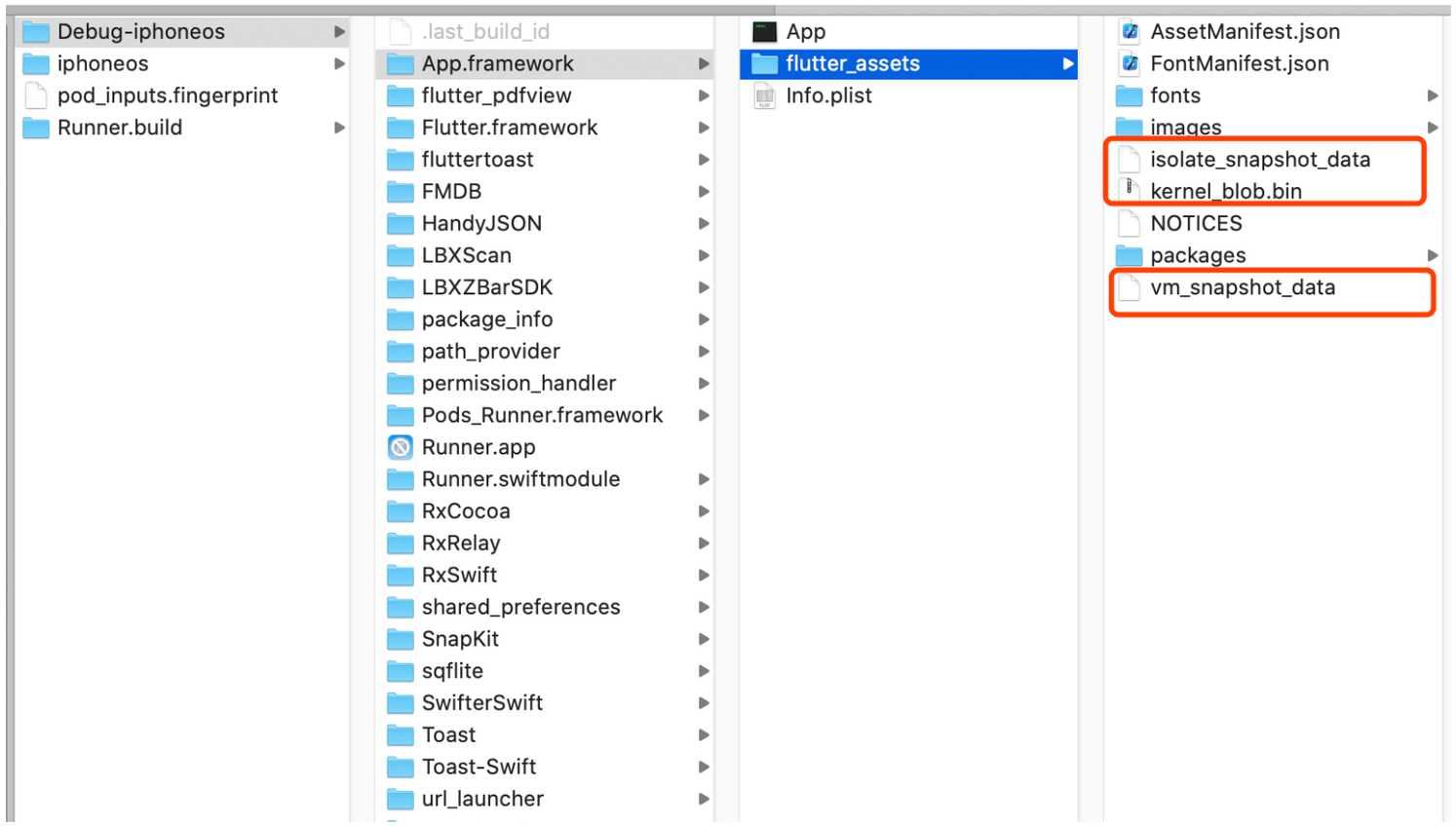
不同的是assets里多个flutter_assets.里面存放的是flutter引用的资源文件

lib里多了libapp.so和libflutter.so

iOS debug模式

App.framework文件夹里多了isolate_snapshot_data, kernel_blob.bin, vm_snapshot_data

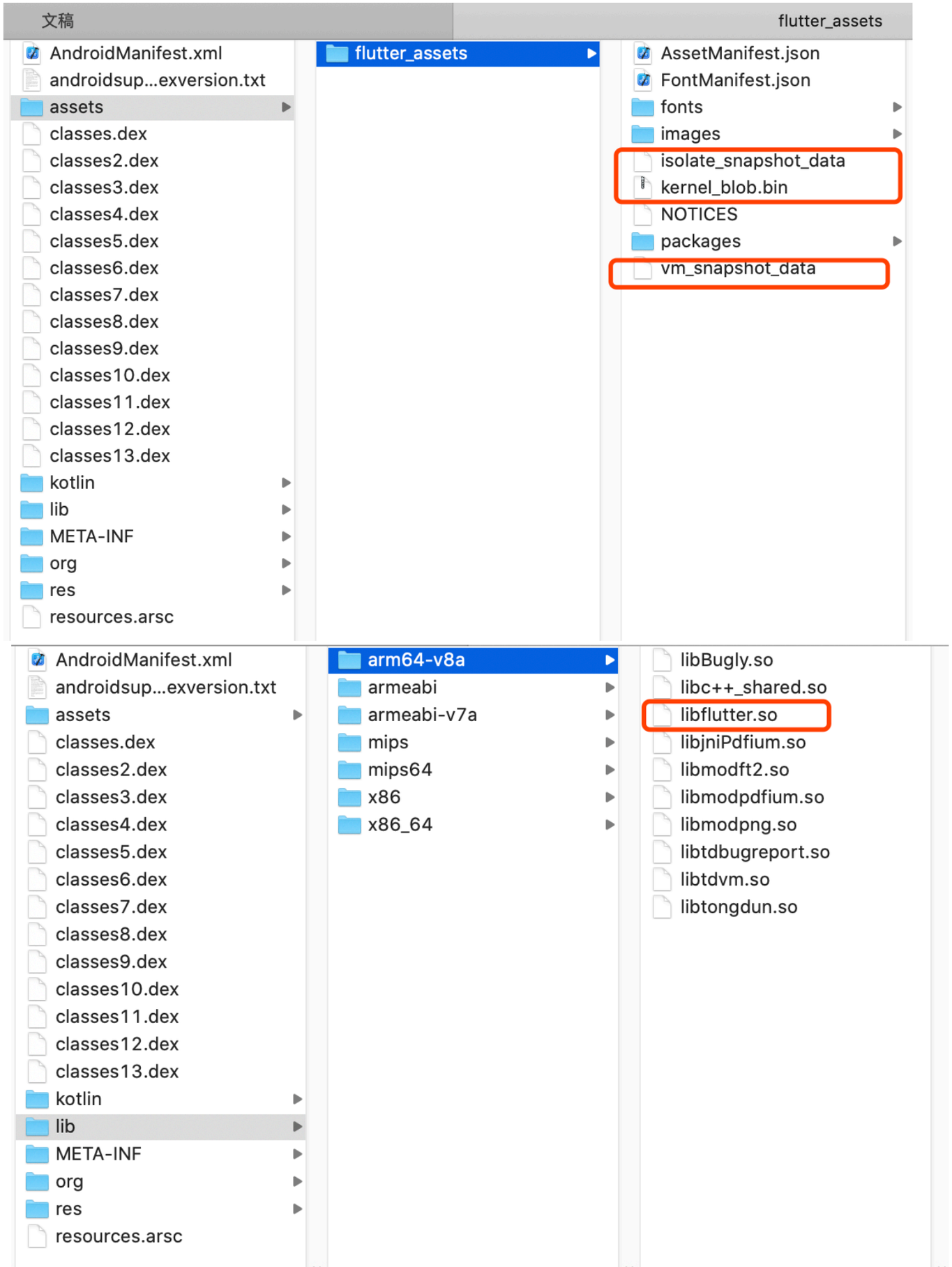
同时App这个二进制文件只有33KB，但是release模式下有8.5MB



安卓 debug模式

1.flutter_assets下多了isolate_snapshot_data, kernel_blob.bin, vm_snapshot_data

2.lib下少了libapp.so



总结

- flutter APP最终会包含两个库，一个是dart代码编译而来的app库，一个是引擎代码编译来的库flutter

- 为了实现hot reload，debug模式下，dart代码会生成至少3个文件，这三个文件就是每次编译动态生成，运行的时候可以替换以便实现hot reload。而release模式下这3个文件会并入app库。flutter库在debug和release模式下大小也是不一样的

- isolate_snapshot_data：用于加速 isolate 启动，业务无关代码

- kernel_blob.bin：业务代码产物

- vm_snapshot_data：用于加速 Dart VM 启动的产物，业务无关代码

- 原生的资源文件和flutter的资源文件是隔离的

- 以上编译产物是基于flutter 2.0.1, Dart 2.12

编译模式

- JIT: 全称 Just In Time (即时编译），典型的例子就是 v8, 它可以即时编译并运行 JavaScript

- AOT: 全称为 Ahead Of Time(事前编译), 典型的例子就是像 C/C++ 需要被编译成特殊的二进制，才可以通过进程加载和运行。

Dart编译模式

- Script：最普通的 JIT 模式，在 PC 命令行调用 Dart VM 执行 Dart 源

模式/比较项	编译模式	区分架构	打包大小	动态化
Script	JIT	否	小	是
Script Snapshot	JIT	否	很小	是
Application Snapshot	JIT	是	比较大	是（注意架构）
AOT	AOT	是	比较大	否

成对应架构的代码。

flutter编译模式（debug）

项目/平台	Android	iOS
代码环境	debug	debug
编译模式	Kernel Snapshot	Kernel Snapshot
打包工具	dart vm (2.0)	dart vm (2.0)
Flutter命令	flutter build bundle	flutter build bundle
打包产物	flutter_assets/*	flutter_assets/*

- vm_snapshot_data: 加速 Dart VM 启动的数据，非业务无关

- kernel_blob.bin: 业务代码产物

flutter编译模式（release）

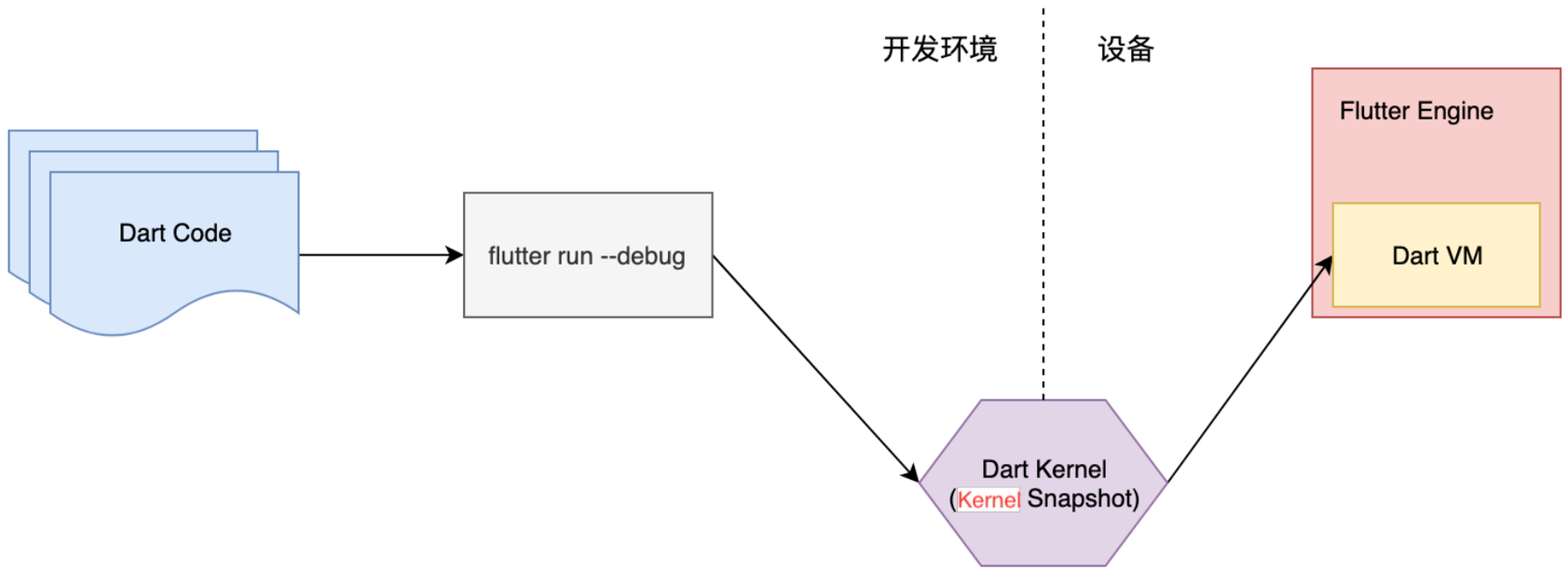
- 在
- A
- 与

项目/平台	Android	iOS	Android(--build-shared-library)
代码环境	release	release	release
编译模式	Core JIT	AOT Assembly	AOT Assembly
打包工具	gen_snapshot	gen_snapshot	gen_snapshot
Flutter命令	flutter build aot	flutter build aot --ios	flutter build aot --build-shared-library
打包产物	flutter_assets/*	App.framework	app.so

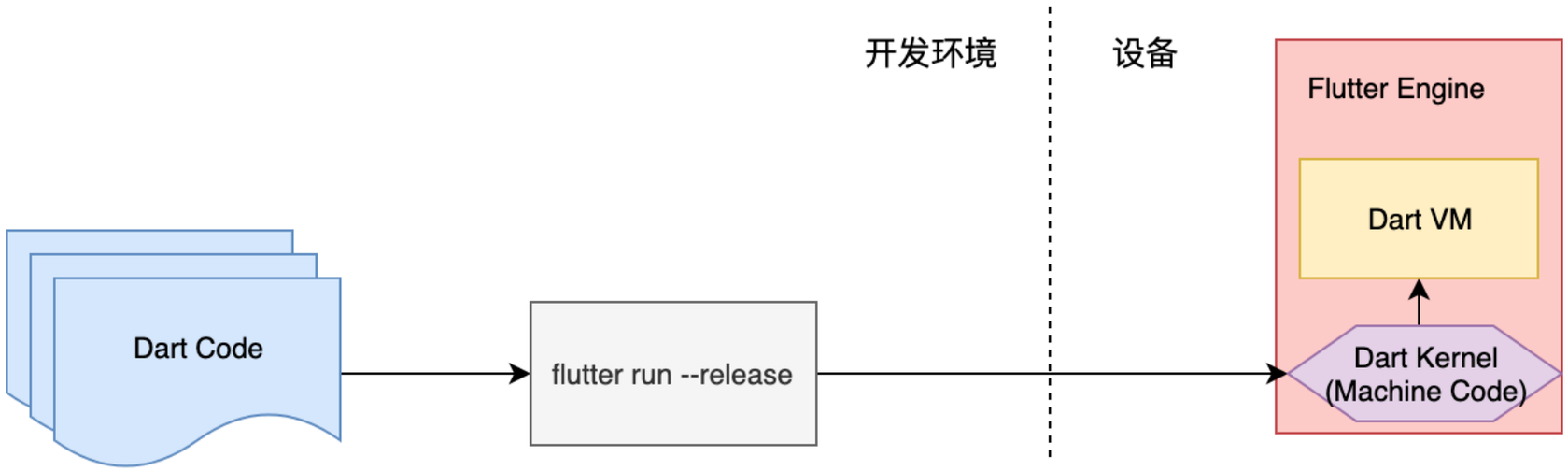
Flutter Engine 编译模式

项目/平台	iOS	Android
Script	不支持	不支持
ScriptSnapshot	理论支持	理论支持
Kernel Snapshot	支持，runmode = dynamic	支持，runmode = dynamic
CoreJIT	不支持	支持
AOT Assembly	支持	支持

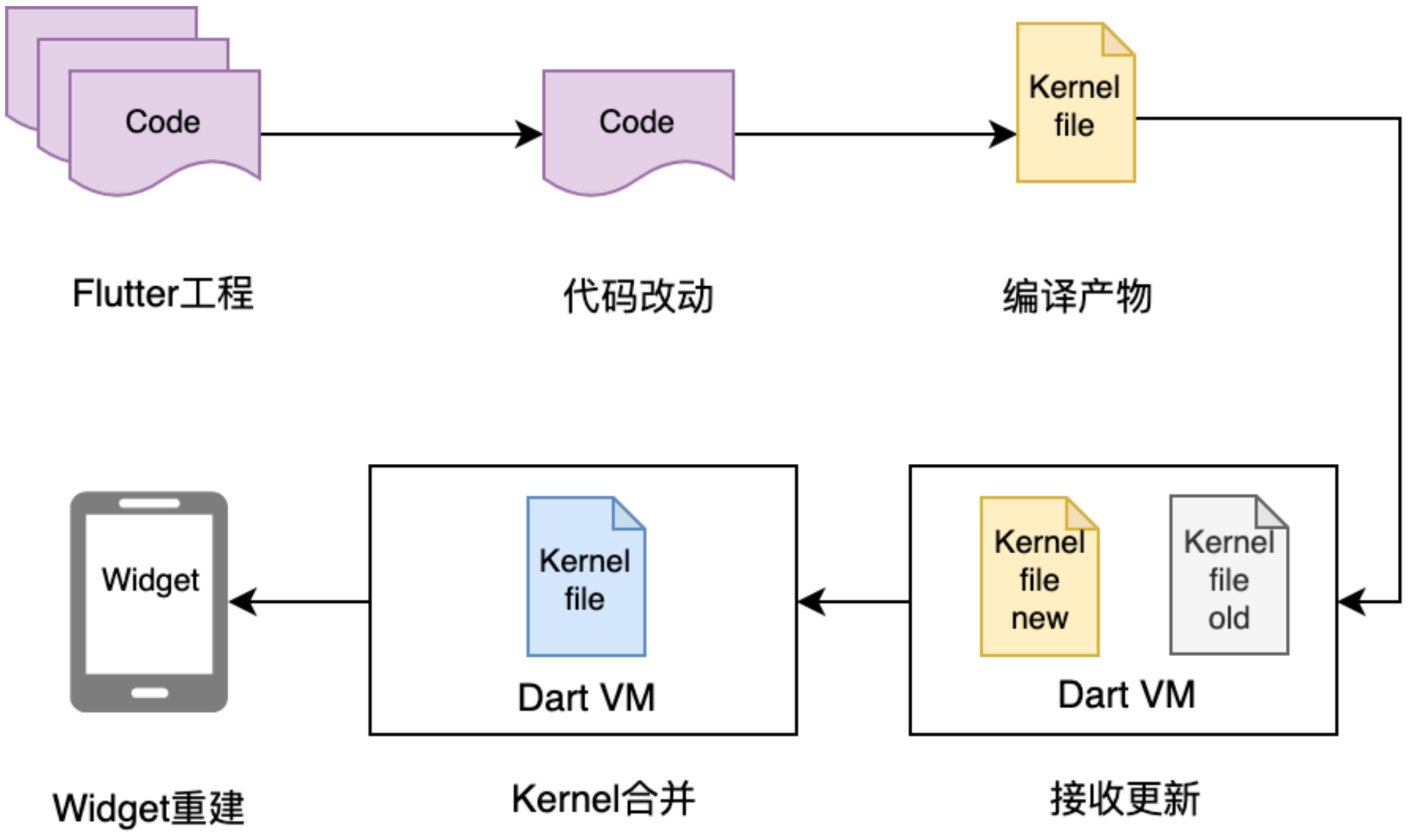
JIT 编译模式示意图



AOT 编译模式示意图

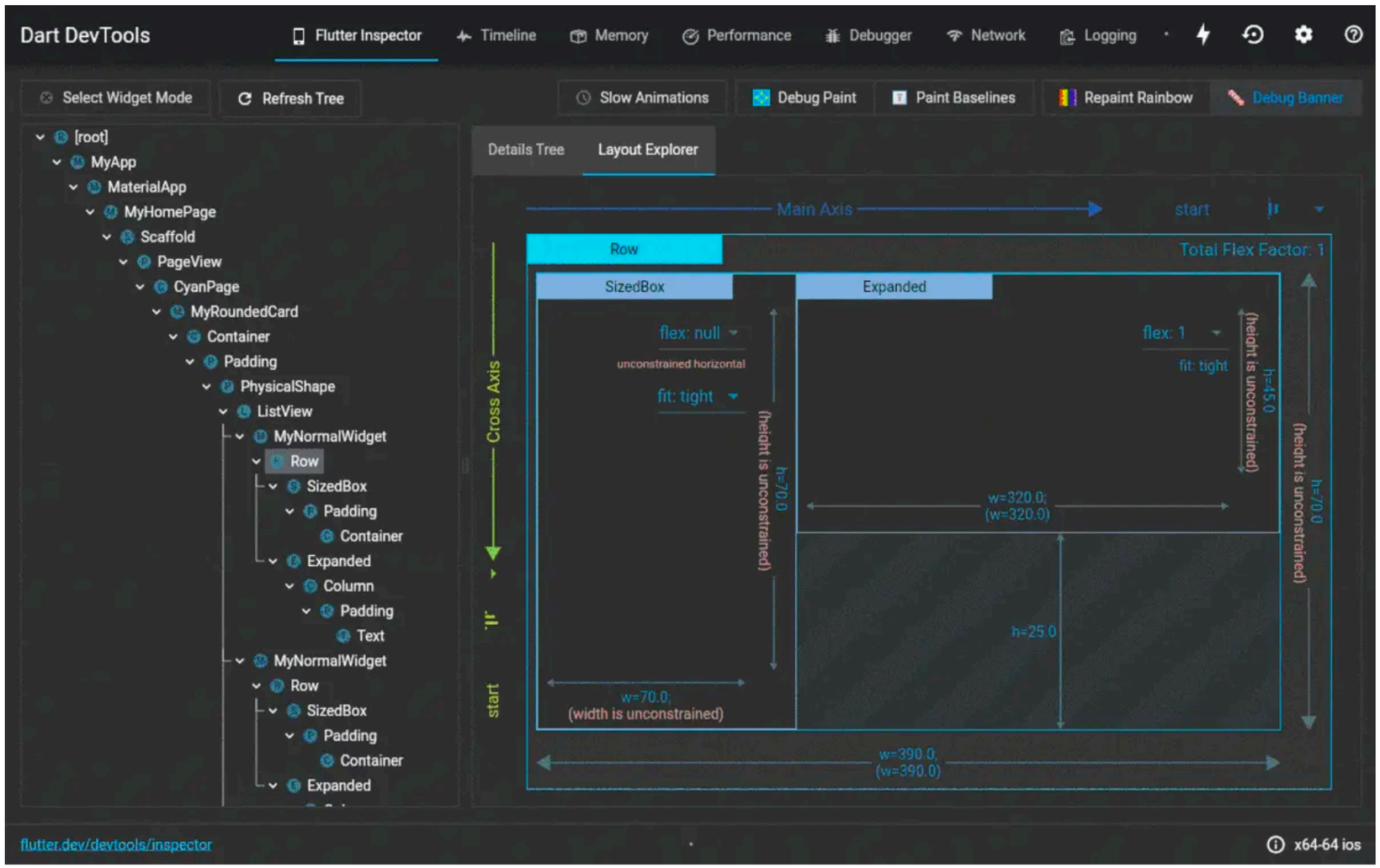


Hot Reload实现方式



DevTools介绍

- Flutter inspector
- Timeline
- Memory
- Performance
- Debugger
- Network
- Logging



引用文章

Flutter 的编译模式：https://zhuanlan.zhihu.com/p/61903658

深入理解 Flutter 的编译原理与优化：https://www.sohu.com/a/239579799_629652

flutter源码解读：http://gityuan.com/archive/

Hot Reload是怎么做到的？https://time.geekbang.org/column/article/136886

Flutter应用如何调试--DevTools介绍(上）https://www.jianshu.com/p/7d8e5e0679f7

flutter engine https://github.com/flutter/engine