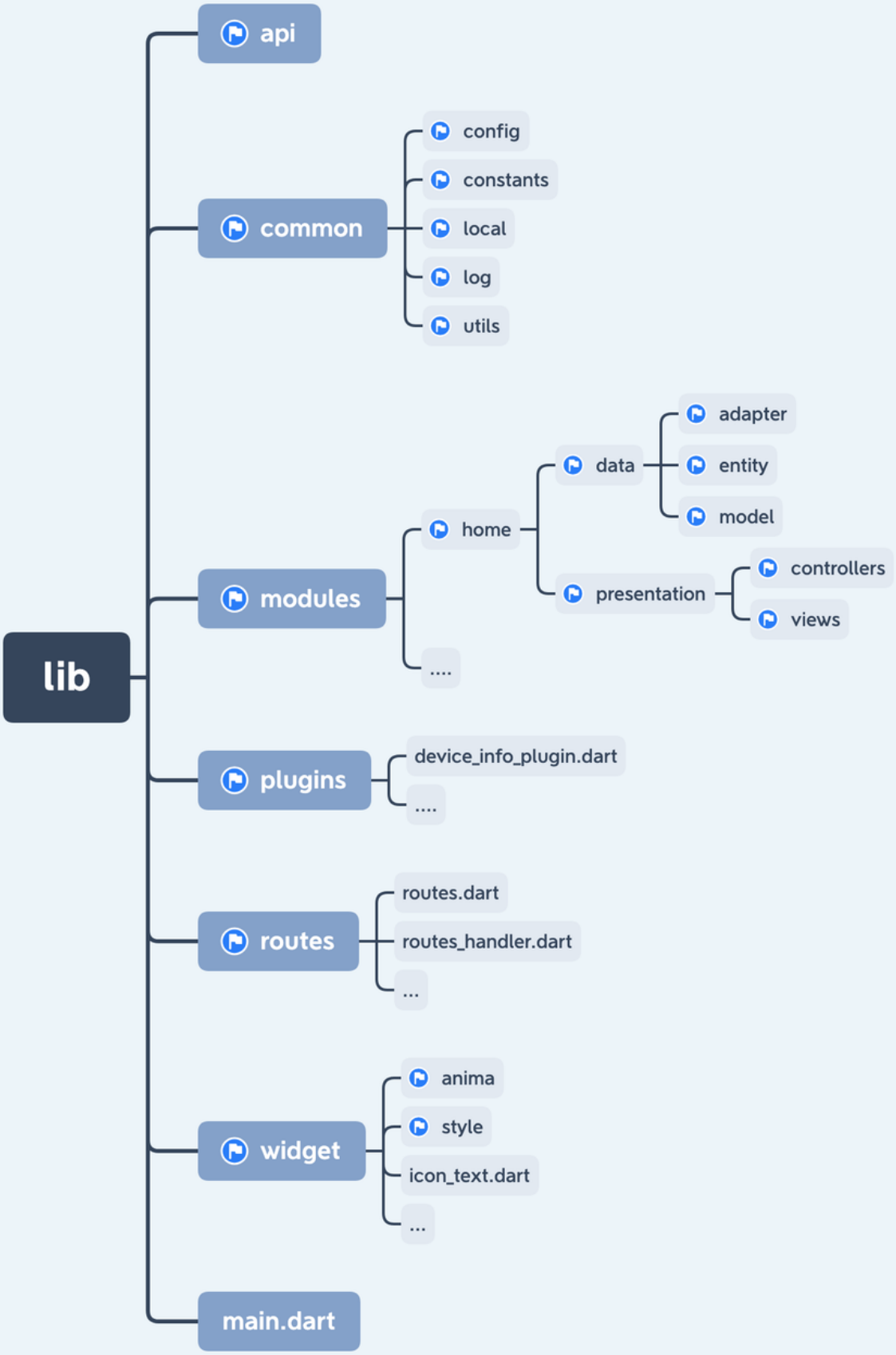
分包结构及边界定义


lib
api
common
config
constants
local
log
utils
modules
home
data
adapter
entity
model
presentation
controllers
views
....
plugins
device_info_plugin.dart
....
routes
routes.dart
routes_handler.dart
...
widget
anima
style
icon_text.dart
...
main.dart

一. api 目录

主要封装全局统一网络请求功能

- 各种拦截器的实现: header, host, log等
- 代理功能，方便测试进行抓包
- 基础参数配置，baseUrl, timeout等

二. common 目录

用于存放一些多个业务模块经常使用的一些变量，方法等

切记不能掺杂业务逻辑的东西在该目录，该目录是可以直接放到别的项目

- config 目录： 填写一些配置相关的参数，例如Firebase配置，wechat配置
- constants 目录: 填写一些全局变量
- local目录：本地持久化的功能类
- localization目录: 多语言适配功能 1. 语言包的懒加载 2. 支持%s, %d字符串的格式化 3. 支持单复数
- log目录：配置全局的网络策略，LRU时间，大小以及日志上传后台时机
- utils目录：工具类

三. modules 目录

业务内容模块，一个大型项目一定是由许多业务子模块组成，这里存放业务逻辑.

data目录

- adapter目录：数据适配器，桥接entity和Model，entity不可能满足业务的任何显示需求
- entity目录： 数据实体类，数据库中字段一一对应或者服务端返回的实体，不能进行任何数据转换等串改
- model目录：模型类型，填充各种数据，可以基于entity进行一些数据拼接，修改，提供给UI界面进行显示
- xxx_api.dart: 该模块的服务端接口
- xxx_repo_provider.dart: 该模块数据库数据接口

presentation目录

- controllers目录: Controller层： 连接data层和views。 这一层可以使用一些状态管理框架，1. 实现Views剧本刷新 2.对外暴露，别的模块可以通过该controller访问或者更新模块数据，实现联动刷新 3. 当模块偏于复杂的时候，对controller进行拆分
- views目录：UI视图层

四. plugins目录

全局plugins地方，当需要底层不同能力的时候，这里定义所有的plugin

五. widget目录

全局通用的一些UI 类， style, theme, color, 子widget等

六. routes目录

定义配置全局路由的地方

具体代码可以参考：GitHub - chejdj/flutter_project_template: Template for Flutter project