链滴

思源源码解析（一）：整体架构

作者：zuoez02

原文链接：https://ld246.com/article/1675744499637

来源网站：链滴

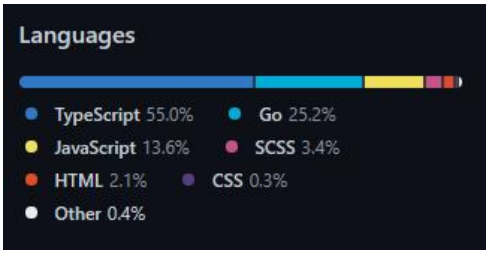
0x00 序

作为一个效率成瘾者和数据本地存储强迫症，自从发现了思源这款笔记软件就十分感兴趣，得益于其全开源，作为程序员的我也热衷于阅读这款程序的代码，并且在集市中贡献自己的挂件。阅读过程中渐发现思源的源码的优点和自认为可以改进的地方，故写下此思源源码系列文章，一边分享阅读源码见解，一边也能从整体架构上分析总结未来的改进方向。

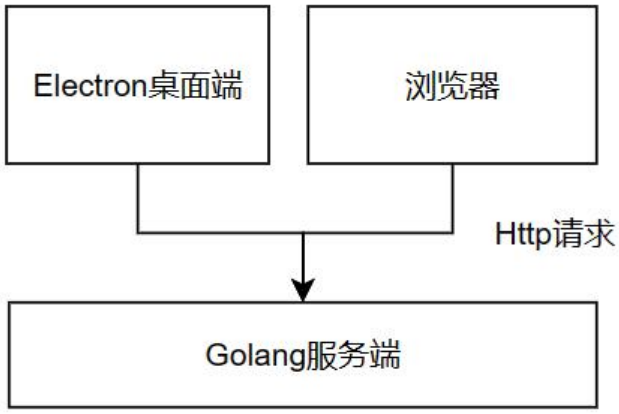
本系列文章基于的版本为2.7.2，并主要分析桌面端程序。

0x01 整体架构

访问思源在Github上的工程主页(https://github.com/siyuan-note/siyuan)，右下角就显示了其语结构。



根据源码结构和本地实际桌面端的运行状况可以了解，思源的桌面端为基于electron实现的应用程序并且使用了Golang编写的后端程序提供http接口实现文件操作、索引等各种功能。这与目前常见的后端分离架构类似，只不过前端放到了electron中。得益于此架构，思源同时可以提供docker容器为的服务端部署以及局域网网络伺服功能。移动端目前还尚未研究，不过根据代码贡献指导文档，同样需要将此go程序编译为移动端程序用于调用。



源码目录结构

```
siyuan
├── .github       // Github相关内容，包括Issue模板、PR模板等内容
├── app           // 前端部分源码(Typescript)
├── kernel        // 后端部分代码(Golang)
├── screenshots       // 功能截图
├── scripts       // 相关自动化脚本
├── API.md        // API文档
├── API_zh_CN.md    // API中文文档
├── CHANGELOG.md    // 版本修改历史
```

```
├── Dockerfile       // Docker镜像构建文档
├── .gitattributes
├── .gitignore
├── .gitmodules
├── LICENSE
├── README.md
└── README_zh_CN.md
```

根据目录结构，我们称呼后端为“Kernel”，前端为“app”。

0x02 APP

app的目录结构为

```
app
├── appearance              // 外观，包含emoji、多语言、引导界面、主题等内容
├── appx                // Appx相关静态资源，应该是移动端用的
├── electron              // electron的主线程源码
├── guide               // 帮助的块数据，我们在思源中自动生成的“帮助”就来自于此
├── pandoc              // 强大的文档转换工具的安装包，包含三大桌面操作系统
├── src                 // APP的主源码
├── stage               // 部分静态资源源码
├── electron-builder-darwin-arm64.yml    // electron macos(arm)构建配置文件
├── electron-builder-darwin.yml        // electron macos(intel)构建配置文件
├── electron-builder-linux.yml        // electron linux构建配置文件
├── electron-builder.yml        // electron windows构建配置文件
├── .eslintignore           // eslint忽略配置
├── .eslintrc.js            // eslint配置文件
├── installer.nsh
├── .npmrc
├── package.json
├── pnpm-lock.yaml            // pnpm锁文件
├── tsconfig.json           // typescript配置文件
├── webpack.api.js
├── webpack.config.js
├── webpack.desktop.js
├── webpack.export.js
└── webpack.mobile.js
```

在这部分的文件名，我们可以猜出很多技术细节：

- 桌面端基于electron，并且支持linux、macos、windows桌面操作系统
- 使用eslint作为代码风格检查工具
- pnpm作为依赖包管理器
- 基于Typescript源码
- 使用webpack打包，同时支持了桌面端、移动端等内容，并且对配置文件进行了多文件拆分
- 使用了pandoc作为桌面端的文档转换工具

从webpack的文件内容和package.json中，我们可以了解到，思源并没有采用React、Vue、Svelte主流前端框架或者库，使用SCSS作为CSS预处理器处理样式。入口文件为app/src/index.ts。后续我在App的详解中可以继续解析这部分内容。

0x03 Kernel

Kernel使用Golang，一门Google开发的跨系统语言，支持编译为各种操作系统和架构的程序。在Readme最下面，我们可以看到思源也是基于各种开源库。入口为kernel/main.go。

```
kernel
├── api           // 各类api的注册及model调用
├── bazaar        // 集市模块
├── cache         // 资源缓存
├── cmd
├── conf        // 配置与类型声明
├── filesys
├── job           // cron任务调度
├── mobile
├── model         // 各个模块的功能实现
├── resource
├── search
├── server        // server服务创建、端口管理
├── sql
├── task
├── treenode
├── util
├── go.mod
├── go.sum
├── main.go
└── versioninfo.json
```

Golang部分代码尚未阅读很多，目前简单列出来，后续在实际了解业务时也可以详细解析。

0x04 小结

以上为基于Github源码的粗略分析，在了解整体结构的情况下，下一篇我们将从系统启动入口开始，析桌面端是如何启动引导的。

首发b3log社区，转载请事先沟通。