链滴

[思源笔记第三方插件系统] v0.3.5 发布，第一次面向社区技术宣传

作者：zuoez02

原文链接：https://ld246.com/article/1677315363862

来源网站：链滴

0x00 写在前面

插件系统事实上已经在脑海里存在很久了，其原因主要可以归结为以下几点：

1. 挂件能力不足以满足开发者的需求

开发者的思路可以说是异想天开的，而挂件由于其iframe的性质，仅能在iframe内激活，也就是依赖用户的笔记展示的时候才能激活。并且由于iframe与主题实际是隔离的，开发中出现诸多不便。

2. 代码片段难以维护和分享

不得不承认代码片段虽然已经给我们提供了注入代码的能力，但是其功能藏的深，并且一堆代码的复粘贴无论在分享方式上，还是本地管理模式上很差。毕竟代码片段就是放在外观配置里的，与原设计不相符。

3. 目前扩展能力很多都是由主题带来的，例如RemCraft、Savor主题的左上角日历功能等，都由每个题作者自行实现。诚然主题作者之间相互熟络可以分享，但是对开发者能力的要求和无法根据用户个想法自行使用的问题依然很大。

思源跟Obsidian很像，都是可以本地存储的双链笔记软件，而OB也有及其丰富的插件生态，那么我也可以参考它的设计，引入一个插件系统来为开发者赋能，为用户个性化赋能。

作为一个挂件开发者（SuperDraw），也算有一些开发经验，因此也思考了如何实现插件系统。最初方案是通过PR到思源本体，通过重构，实现前端API和引入插件管理方案，增加到集市中。但是跟核开发者交流之后，还是决定采取一个中间层的方案，基于稳定性考虑，思源的代码不改，靠我们实现个第三方的插件系统来实现插件控制。

最近几天一直在个人开发插件系统，也算小有结果，正巧社区内也有人讨论插件系统需求的问题，那我觉得也算有同行人在思考类似的问题。因此决定把目前正处于内部alpha版本的第三方插件系统公出来，跟社区里更多志同道合的朋友们交流和推广使用。

本篇文章主要面相开发者群体，可以理解为一个alpha版本

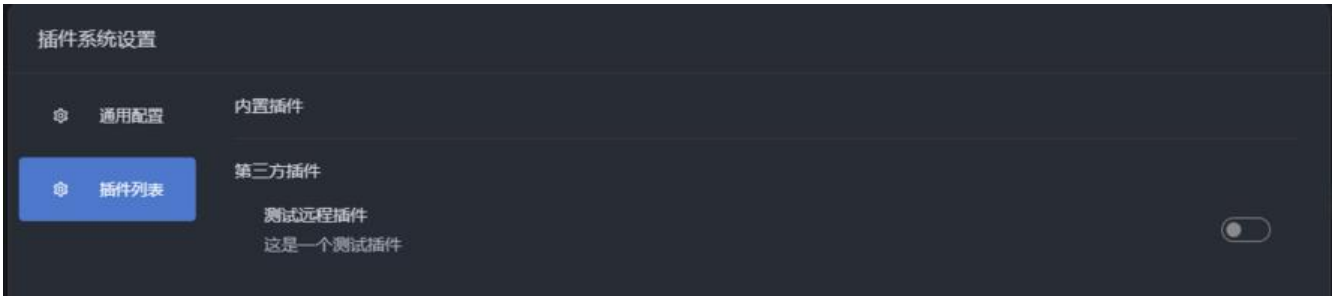
0x01 已有实现

首先通过代码片段的方式，引入第三方插件系统的脚本

然后添加我们的JS脚本，引入插件系统

```
(function(){const path=require('path');const getCrossPlatformAppDataFolder=()=>{let configFilePath;if(process.platform==="darwin"){configFilePath=path.join(process.env.HOME,"/Library Application Support")}else if(process.platform==="win32"){configFilePath=process.env.APPDATA}else if(process.platform==="linux"){configFilePath=process.env.HOME}return configFilePath};try{const data=require('fs').readFileSync(path.join(getCrossPlatformAppDataFolder(),'.siyuan','plugin.js'));const script=data.toString('utf8');console.log('local plugin system found, loading...');eval(script)}catch(e){console.log('local plugin system not found, load online');return fetch('https://gitee.com/zuoez02/siyuan-plugin-system/raw/main/main.js',{cache:'no-cache'}).then((res)=>res.text()).then((sc)=>{window.siyuanPluginScript=sc;eval(sc)})}})();
```

重新启动思源，Ctrl+Shift+i，打开控制台，即可看到插件系统加载的日志输出。



0x02 设计参考

既然是对标了Obsidian的插件系统，那么就做了逆向工程，了解了一下它的插件系统设计，写了一篇源源码解析外传：Obsidian 源码插件系统逆向分析（一）。

我们可以借鉴的方案包括：

1. 插件包含了描述文件、main.js和main.css三个部分组成
2. 插件的模块化方式为commonjs
3. 插件的API通过 require('siyuan')来实现。可以提供类型系统便于支持，更加贴合现代化开发
4. 插件需要暴露一个继承siyuan.Plugin的类，作为插件模块的default export。
5. 插件需要提供生命周期控制
6. 插件的安全性控制很重要
7. 插件的开关、安装、存储流程

0x03 实现方案

目前插件系统是开源的，地址为 https://gitee.com/zuoez02/siyuan-plugin-system和https://github.com/zuoez02/siyuan-plugin-system。

插件系统包括以下几个部分：

1. 插件系统核心PluginSystem
2. 插件系统本地文件及更新管理PluginSystemLocalManager
3. 插件对接存储管理StorageManager
4. 插件加载器PluginLoader
5. 插件API生成器apiGenerate

这几个部件构建了插件系统的基础，而后续功能，则利用插件系统的能力，为插件系统进行拓展，例界面按钮添加，配置菜单，插件系统界面配置等功能。

常言道我们不是巨人，只是站在巨人的肩膀上。得益于社区主题与其他资源的代码共享，我们可以借过来整合为良好的API，并反过来继续为主题作者们赋能，共享API能力，避免单打独斗。

开发用到的技术主要包括：

- Nodejs 18
- Typescript：全栈TS化，提高效率，避免bug
- Vite：一体化的构造器，效率极高
- Svelte：无Runtime的前端框架，降低插件系统对思源的运行压力
- Inversify：ioc容器管理，便于各种对象的管理
- Pnpm：包管理器

需要注意的一些点：

1. 由于我们主要提供脚本能力，未引入CSS，因此尽可能使用官方的类和样式

2. 很多能力还是借用Siyuan本体的能力，例如Menu、Dialog等，后面扩展时依然优先考虑内部已有现，便于后续发展。

插件demo

plugins.zip

将此目录放在工作空间/data下并解压，可以得到工作空间/data/plugins/test1目录，test1中包括man.js入口文件，manifest.json描述文件，readme.md说明文件。

第三方插件默认不启动，需要重载前端并在插件配置中手动开启。

0x04 未来计划

连续肝了几天，内容仍然非常有限，特别是没有任何一个可以赋能用户的插件，这块还需要更多开发同学的支持，整体方案依然在思考推进，欢迎加入“思源笔记折腾群”来探讨有什么想做的。

Roadmap

- 引入更多API支持插件更新
- 跨平台计划，替换API
- 插件加载管理
- 插件生命周期管理，创建与销毁
- 插件包分发方案实现
- 考虑版本兼容性问题，更改自动更新和下载策略

0x05 鸣谢

感谢“思源笔记爱好者折腾群”中的多位热心开发者提供支持，入群方式见https://ld246.com/article/1640266171309。群内 @zuoez02 SuperDraw开发者

感谢@88250 @Vanessa 插件系统目前为社区方式运营，未来是否融合入思源本体，我们可以多多讨smile 。