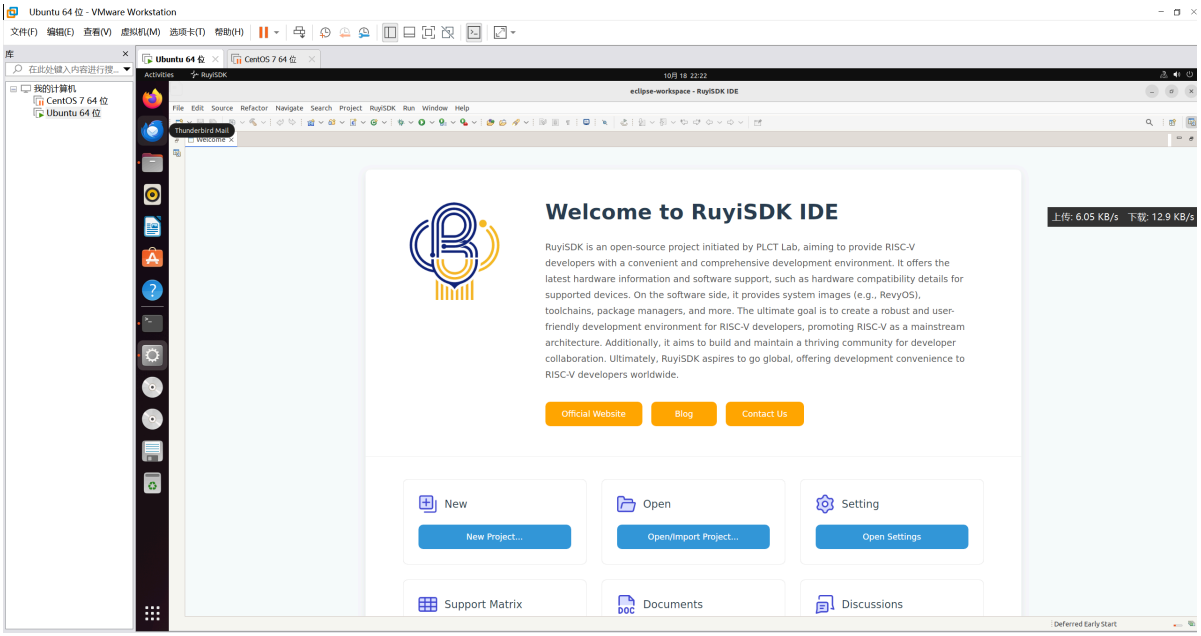
RuyiSDK 使用体验与功能分析

一、RuyiSDK 使用截图和感受

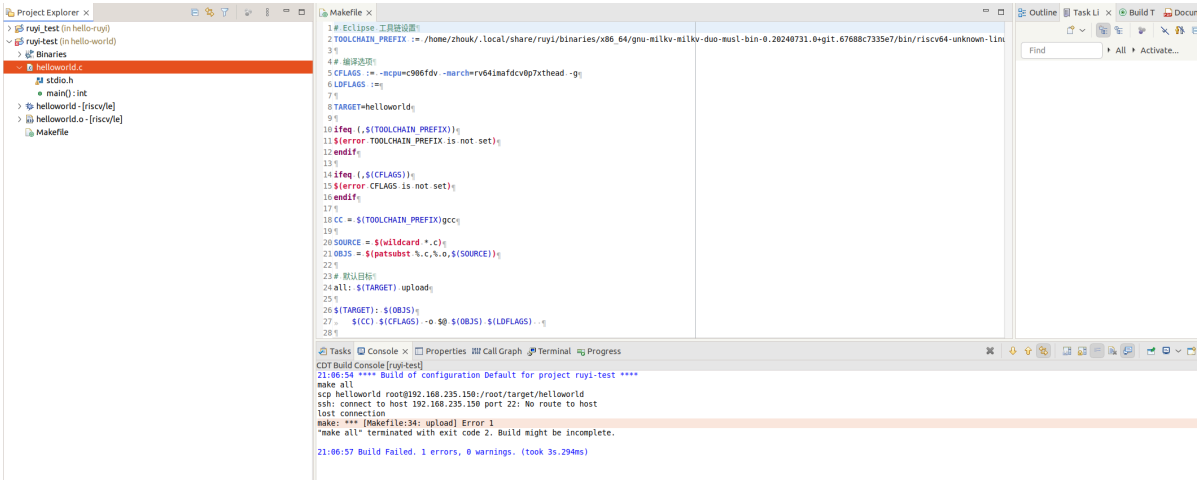
初次使用体验--Ubuntu Linux x86_64

初次打开 RuyiSDK IDE 界面：



为 Milk-V Duo 编译 HelloWorld 程序

由于无设备，报错仅是upload有问题：



编译运行示例

按照官网指导，尝试编译 coremark 并用 qemu 运行：

```
«Ruyi venv» zhouk@zhouk-virtual-machine:~/Desktop/coremark$ ruyi-qemu coremark.exe
2K performance run parameters for coremark.
CoreMark Size    : 666
Total ticks      : 16582
Total time (secs): 16.582000
Iterations/Sec   : 3618.381377
Iterations       : 60000
Compiler version : GCCRuyiSDK Clang 21.1.1 (https://github.com/llvm/llvm-project 5a86dc996c26299de63effc927075dcbfb924167 RuyiSDK 20250915)
Compiler flags   : -O2   -lrt
Memory location  : Please put data memory location here
                        (e.g. code in flash, data on heap etc)
seedcrc          : 0xe9f5
[0]crclist       : 0xe714
[0]crcmatrix     : 0x1fd7
[0]crcstate      : 0x8e3a
[0]crcfinal      : 0xbd59
Correct operation validated. See readme.txt for run and reporting rules.
CoreMark 1.0 : 3618.381377 / GCCRuyiSDK Clang 21.1.1 (https://github.com/llvm/llvm-project 5a86dc996c26299de63effc927075dcbfb924167 RuyiSDK 202509
15) -O2   -lrt / Heap
«Ruyi venv» zhouk@zhouk-virtual-machine:~/Desktop/coremark$
```

使用感受：对于新手用户来说，当前的使用流程不够友好，学习成本较高，介绍资源太过稀少。

二、Ruyi 包管理器使用

Maven 支持功能

当前使用Eclipse官方的tycho去支持 Maven 的初步构想（此前已用于完成 p2 仓库构建），但只能本地压缩包手动更新，对新手用户不够友好，对已经使用的资深用户不够便捷：

拿安装 `tycho` 插件为例(tycho插件是可以将在eclipse中进行插件开发时传统项目转为maven 次项目的一种插件，从而利用maven中的pom文件解决包依赖问题,并且支持更加便捷的构建编译plug in 项目)。

三、Eclipse 插件功能分析

当前问题

了解到 Eclipse 插件功能很丰富，但从新手角度来看，存在以下问题：

- 缺少对插件的详细介绍和直观体验或者说区别
- 对于新用户，不清楚 RuyiSDK 插件的具体功能和应用场景
- 没有说明让用户明显感受到插件带来的便捷性

插件说明

*

ruyisdk-ide-docs

- 项目目标
- 项目组成总览
- 基础研发模块

IDE

Common

- 面向RISC-V开发的主要开发需求总览
- IDE插件需求
- RuyiSDK包管理器介绍
  - 仓库：https://github.com/ruyisdk/ruyi
  - 技术说明：https://github.com/ruyisdk/ruyi/blob/main/docs
  - 接口：https://github.com/ruyisdk/ruyi/blob/main/docs/programmatic-usage.md
  - 使用文档：https://ruyisdk.org/docs/category/ruyi-%E5%8C%85%E7%AE%A1%E7%90%86%E5%99%A8
  - 补充说明：ruyi 包管理器本地文件

Eclipse 插件

- 资源简介： 基于eclipse embeded cdt 定制IDE（ruyisdk ide）包括两个部分：一个是打包工程和一个插件仓库：
  - 打包工程仓库：https://github.com/ruyisdk/ruyisdk-eclipse-packages/
  - 插件仓库：https://github.com/ruyisdk/ruyisdk-eclipse-plugins/
  - 代码规范：https://github.com/ruyisdk/ruyisdk-eclipse-plugins/blob/main/docs/developer/CONTRIBUTING.md
- 设计文档：
  - eclipse-ruyi-plugins 功能构想
- 发布方式：
  - 方式一：将插件打包到IDE中，提供一体化的IDE工具。（适合新用户初装）
  - 方式二：单独提供插件安装和更新（适合插件单独安装/更新场景）
    - 在 Eclipse marketplace 发布（推荐）
    - 在 ruyisdk 镜像站 和 github仓库release 提供jar或者zip包

四、具体问题反馈

SSH 连接问题

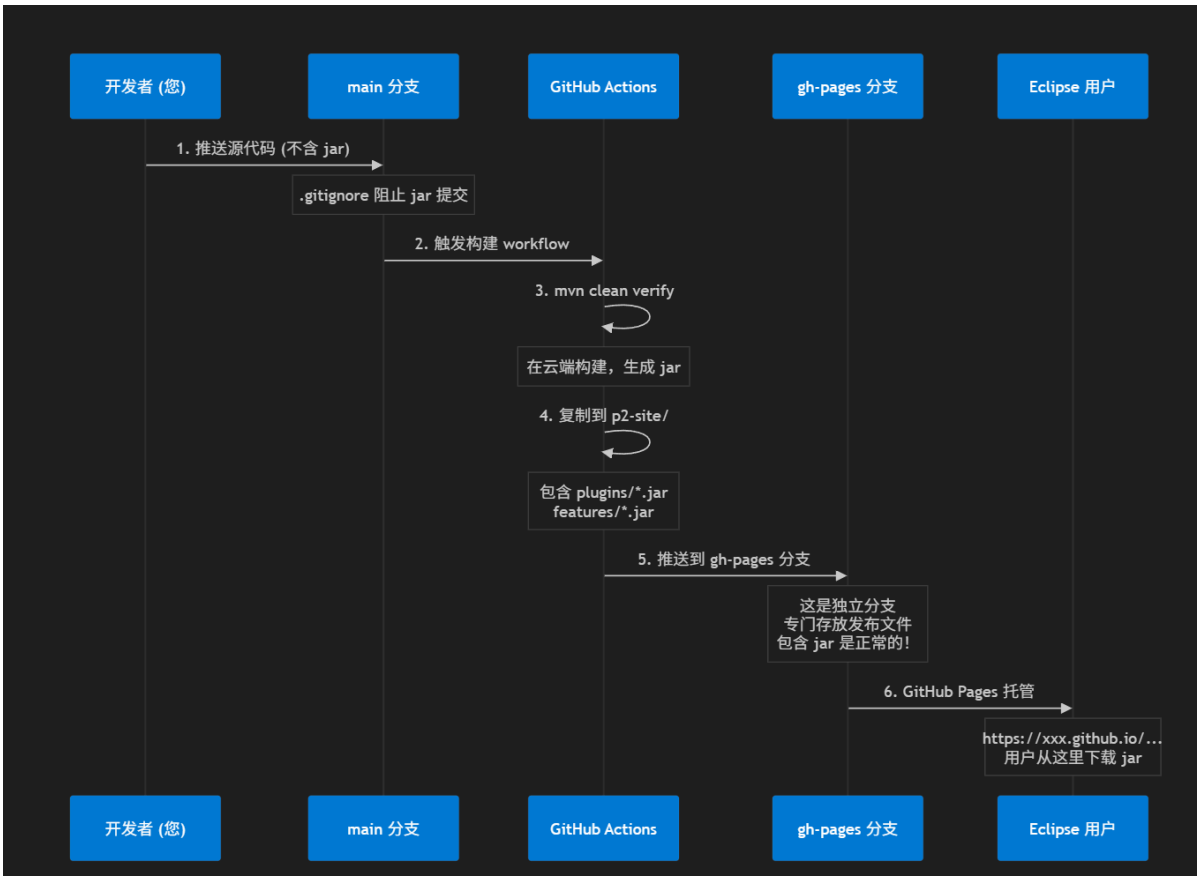
样例 SSH 连接不通：

Terminal: Open a new Terminal View/Open a Terminal > SSH Terminal > 对照下图输入Host、User、Password（milkvduo的root密码是：milkv）

五、CI/CD 集成参考

GitHub Actions 集成--正在做ing

GitAction CI/CD 介绍参考：
actions/starter-workflows: Accelerating new GitHub Actions workflows



1. Eclipse 官方插件：

- 主分支：只有源代码
- P2 站点：包含 jar 包

1. VSCode 扩展：

- 主分支：只有源代码
- Releases：包含 .vsix 文件（类似 jar）

1. Maven Central：

- 源代码仓库：不含 jar
- Maven 仓库：全是 jar

六、插件更新与分发机制研究

核心问题

1. 如何提供插件的更新自动通知?
2. 如何发布插件和扩展，以便让更多人知晓?

参考方案分析

参考一：CSDN上的Mozila更新机制

一、如何提供插件的更新自动通知?

关于Install manifest（即install.rdf）文件内容，Mozilla有篇很详细的文章：https://developer.mozilla.org/en/Install.rdf，可以进行参考。

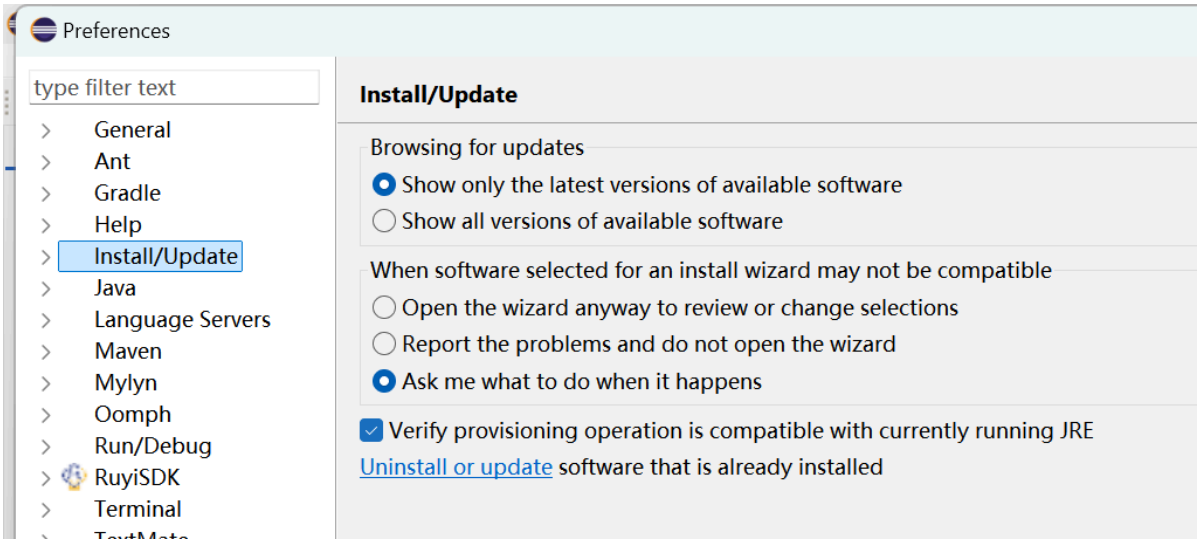
其中有个节点名称是：updateURL。用这个标签包含一个可查询更新信息的URL连接，今后当FF在查询插件是否有更新时，会到你指定的这个URL上去查询。

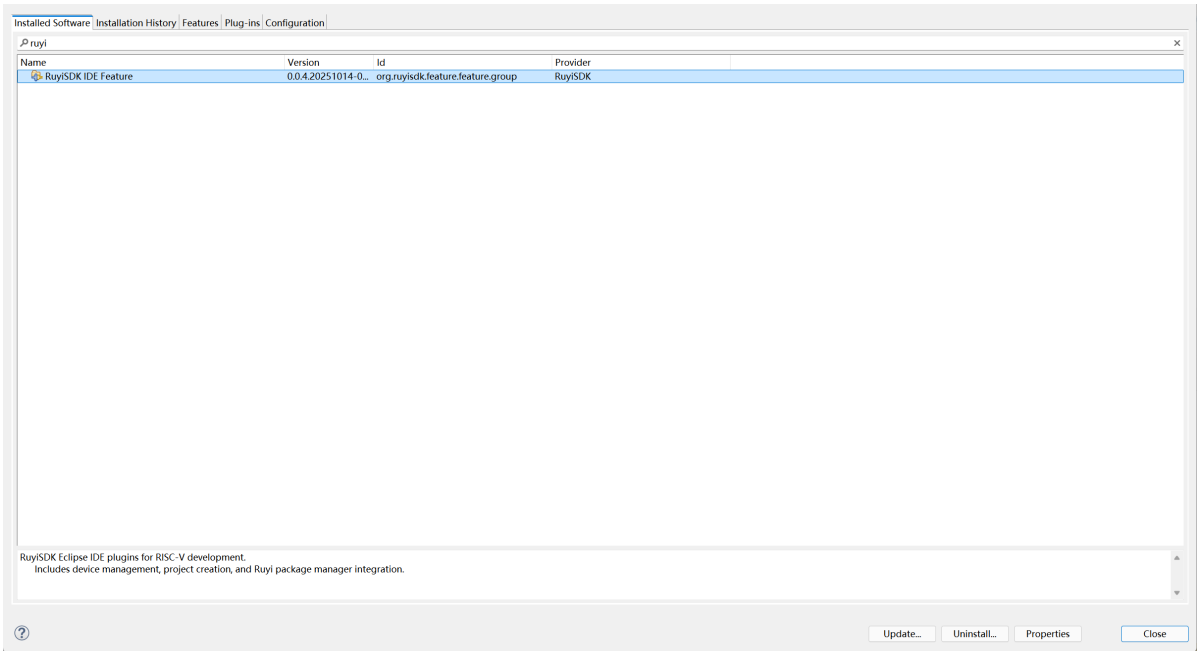
这个URL可以是一个cgi程序、servlet、asp(x)，也可以是一个固定的rdf文件链接。另外文章中也提到了另外一种办法：如果你没有提供updateURL，FF的附加组件管理器会向addons.mozilla.org发送请求，当你上传了一个新版本的插件或者通过作者接口（author interface）更改了兼容性参数，一个更新的manifest文件将会被自动产生。另外，Mozilla建议updateURL使用https而非http，否则你应该提供updateKey（可以用McCoy生成）。

那么，负责更新的rdf怎么编写呢？你可以参考这篇文章：https://developer.mozilla.org/en/Extension_Versioning%2c_Update_and_Compatibility#Update_RDF_Format。由于文章中写的很清晰，并且提供了详细的例子代码，这里我就不废话了。

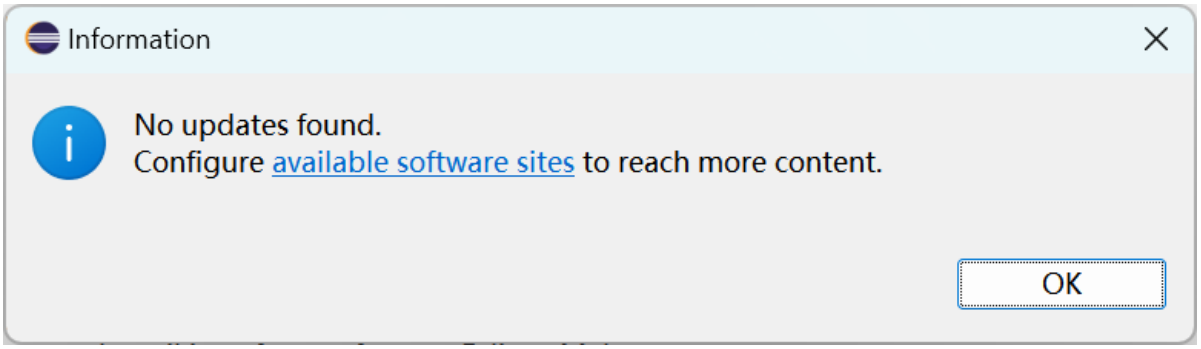
总结:

- 通过 URL 更新，发送请求
- 要么内置，要么官方提供
- Eclipse 提供标准更新路径：Window > Preferences





现状分析：目前没有提供，这个原理我还要再学习一些知识搜索一些资料，才能知道原理，目前猜测是就是利用远程仓库地址之类的



参考二：IDEA 方案

简介

idea启动过程中，并不会自动去插件市场检查插件是否有更新。发布插件后，需要用户重启 IDE，或者点击help -> check for updates，体验不太好。

如何自动检查更新

可以通过以下步骤来实现插件更新：

1. 实现版本管理机制，让插件能够查询到当前最新可用的版本。
2. 获取当前用户安装的插件版本，如果低于最新版本，则打开插件市场，并调用更新API。

这些操作依赖一些IntelliJ的open API：

总结：基于内部 API 或自建远程仓库实现更新机制。

参考三：VS Code 插件机制

注意

只有重启窗口的VS Code实例插件才生效

VSCode 提供命令行工具进行插件的安装， 例如：code --install-extension youplugin.vsix。 但是不要使用 child_process 中的 exec 执行这个命令。VSCode不会执行此命令，无法进行插件的安装

插件更新的流程

vscode插件更新流程 访问密码：L5eA

更新流程

1. 触发更新
   - 方式一：VSCode启动插件初始化检查更新
   - 方式二：创建命令，通过执行命令，主动触发更新
2. 向服务器获取 version.json 得到相关信息，如插件版本信息
3. 判断是否需要更新
4. 需要更新则根据指定URL获取插件包
5. 通过VSCode提供的API（`vscode.commands.executeCommand`）进行插件的安装
6. 重启VSCode使插件生效

总结：

- VS Code 插件使用内部 API
- 支持命令行指令（Eclipse 可能也有类似功能）
- 支持 URL 指定更新，与前述方案类似

参考四：通义灵码参考更新方案

如何禁用VSCode通义灵码自动更新?

青少年编程

如何禁用VSCode通义灵码自动更新？ 在使用VSCode过程中，通义灵码插件的自动更新功能可能会对网络环境或版本稳定性造成影响。用户常希望禁用其自动更新机制，以保持插件版本不变。通常，VSCode插件的更新由编辑器本身控制，无法直接通过插件设置关闭。但可通过以下方法实现禁用：一是通过修改VSCode设置，关闭自动更新功能；二是手动下载插件包并离线安装，避免联网更新；三是利用系统防火墙或hosts文件屏蔽插件更新服务器。具体操作需结合用户环境和权限进行调整。

总结：

- 离不开 IDE 内置的检测接口
- 支持离线安装
- 通过远程 API 发送更新请求，使用 URL 进行更新

七、待解决的问题

1. **插件功能明确化**：需要明确 RuyiSDK 插件的具体功能和价值点，以及暴露给用户的思考
2. **新手友好度提升**：降低使用门槛，提供更清晰的使用指引
3. **SSH 连接问题**：解决样例中的连接问题
4. **更新机制建设**：建立完善的插件更新和分发机制
5. **文档完善**：补充详细的插件介绍和使用文档，添加类似搜索功能、常见问题
6. **我的问题**：由于确实是第一次了解到并投入插件开发和Eclipse其他版本的使用，所以目前还在搜索资料学习，也发现网上的资料有些过于古老