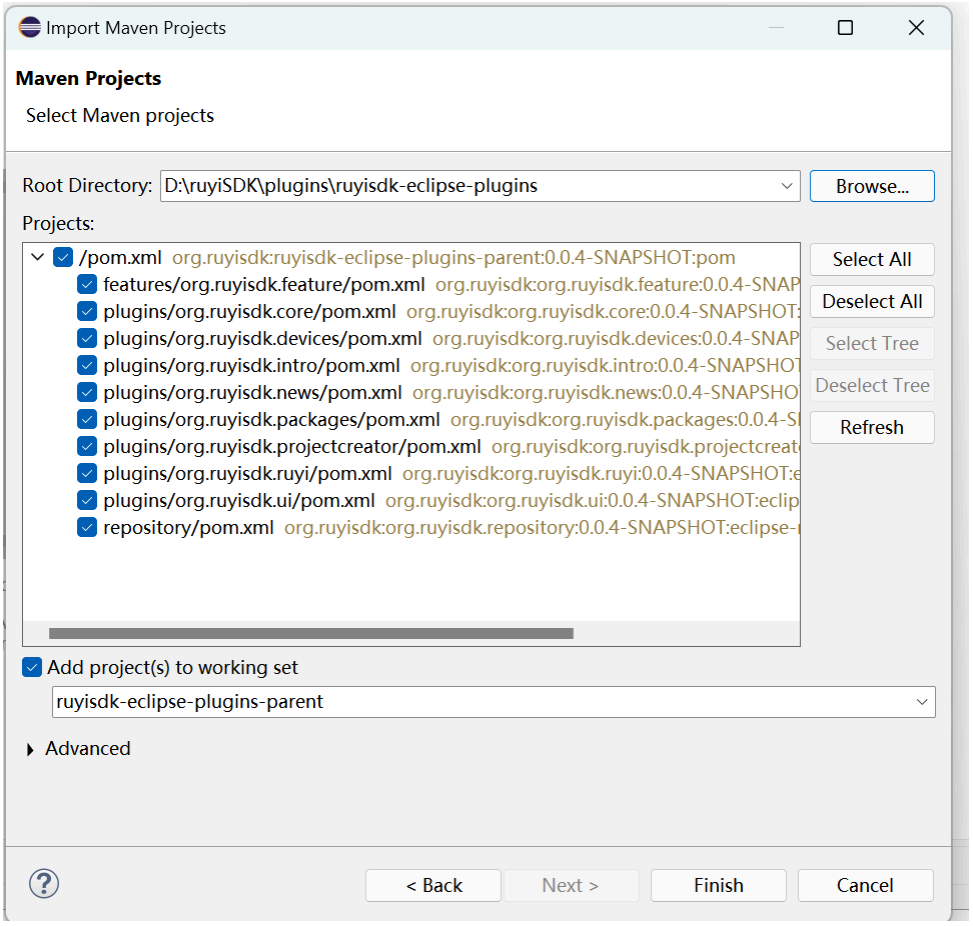
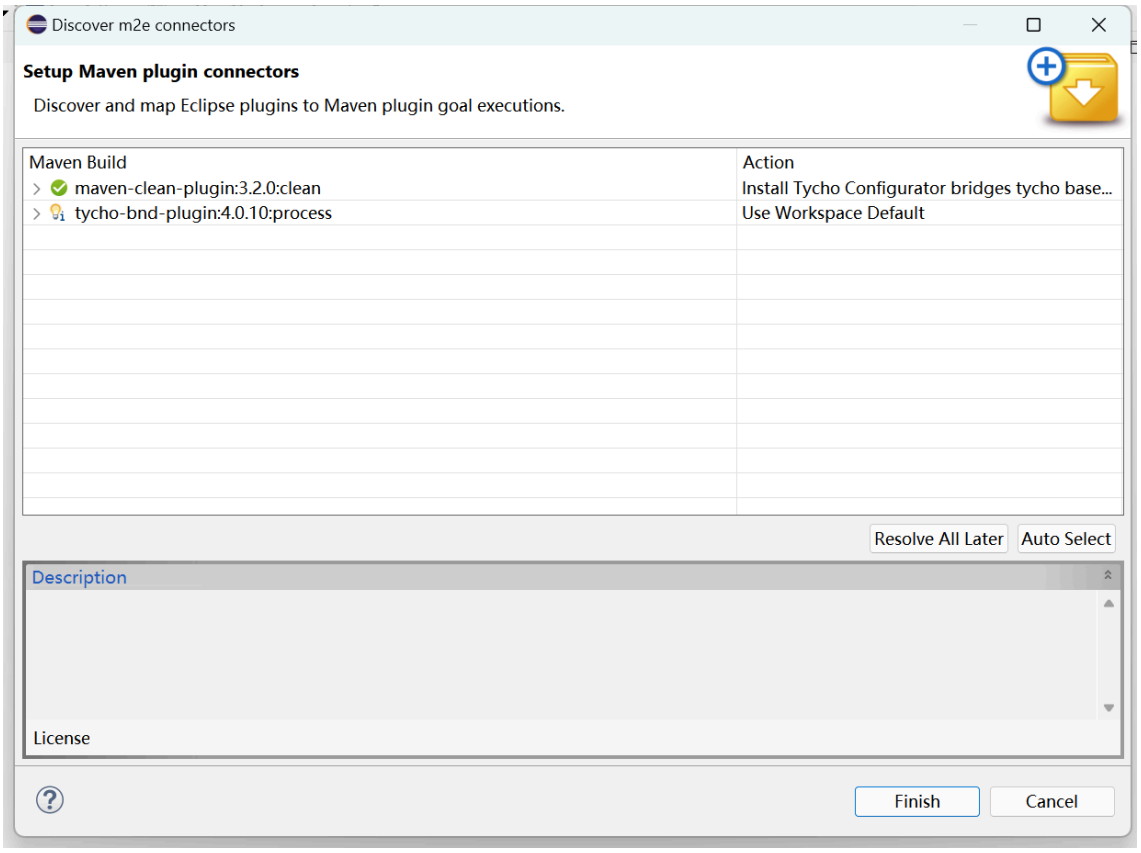
在 Eclipse 中调试 Maven/Tycho 插件的完整步骤

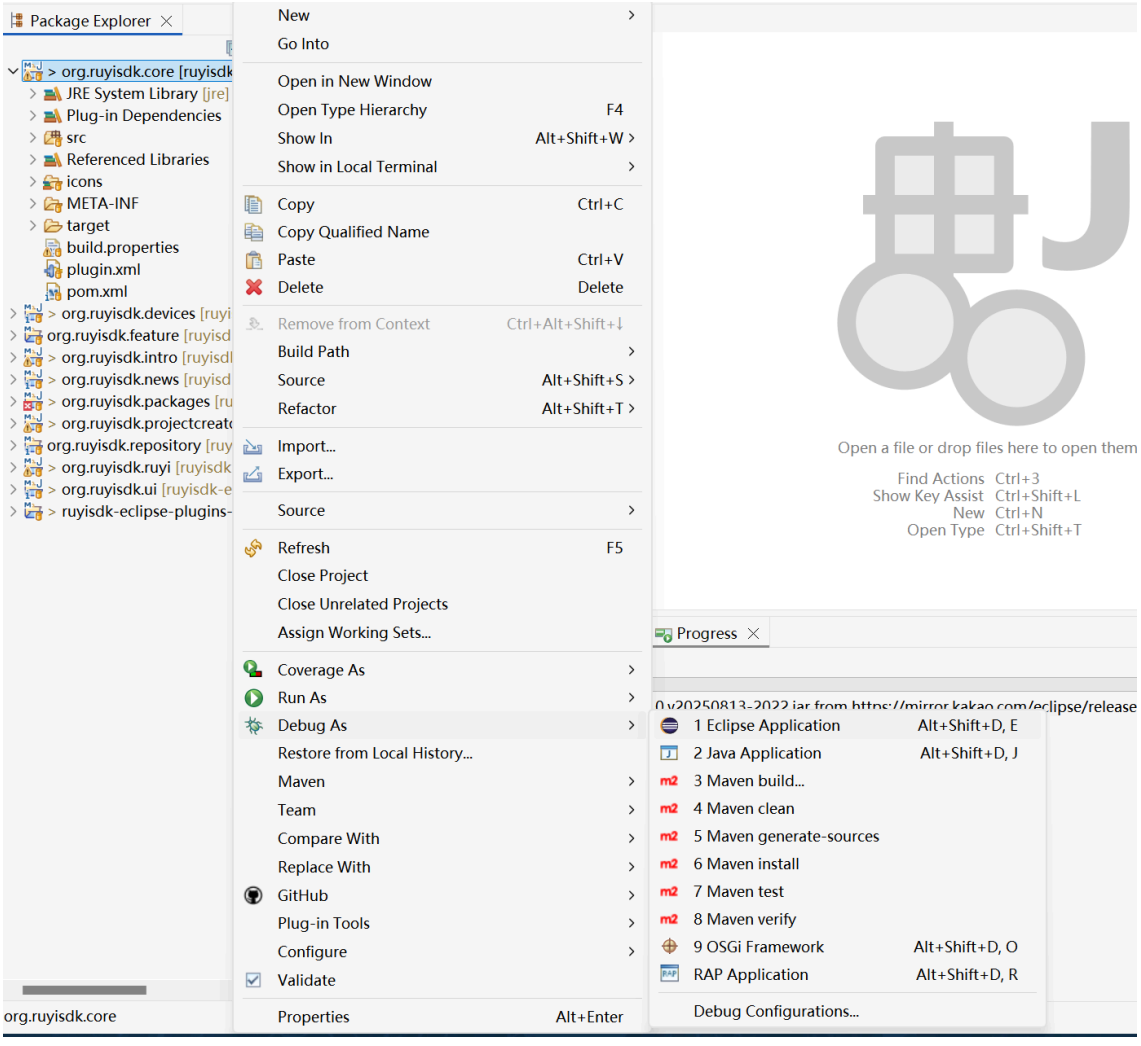
方法一：使用 Eclipse Application 启动配置（推荐）

1. **导入项目到 Eclipse**
   - `File` → `Import` → `Maven` → `Existing Maven Projects`
   - 选择 `ruyisdk-eclipse-plugins` 目录
   - 确保所有子模块都被导入





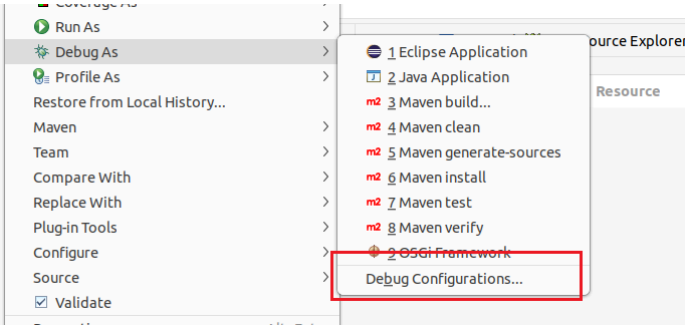
- 右键点击任意一个插件项目（如 `org.ruyisdk.core`）
- 选择 `Debug As` → `Eclipse Application`



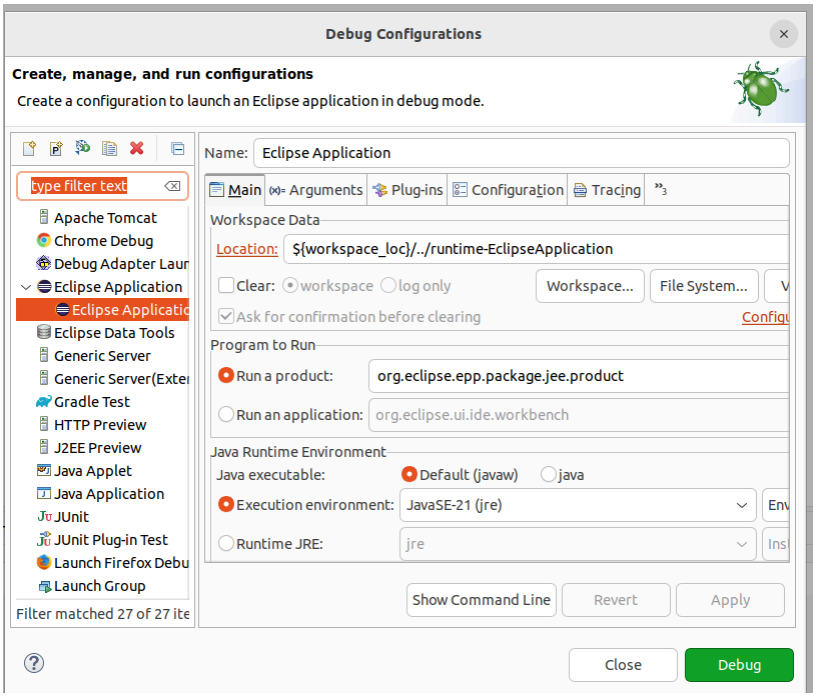
- 这会启动一个新的 Eclipse 运行时实例，其中包含您的插件（有报错见问题四，有待讨论）

方法二：自定义启动配置（还未尝试）

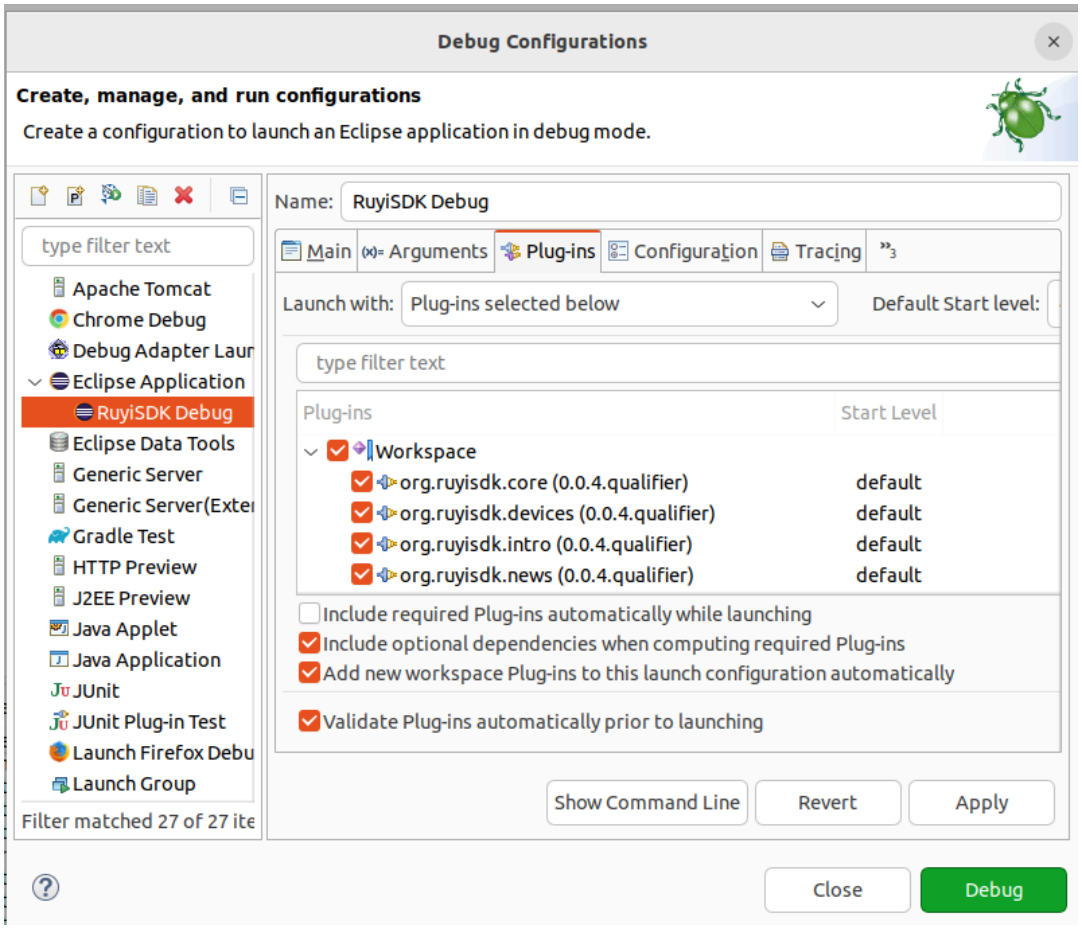
1. **打开运行配置**
   - `Run` → `Debug Configurations...`



   - 双击 `Eclipse Application` 创建新配置



2. **配置参数-参照上图**
   - **Name**: 给配置起个名字，如 "RuyiSDK Debug"
   - **Main** 标签页:
     - Program to Run: `Run an application: org.eclipse.ui.ide.workbench`
   - **Plug-ins** 标签页:



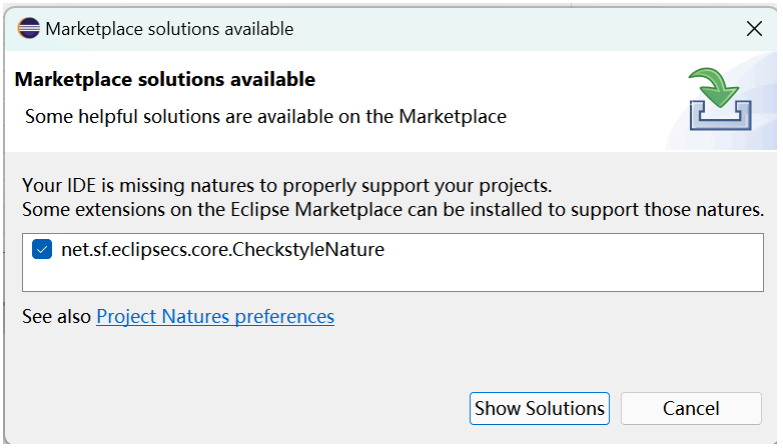
- 选择 `Launch with: plug-ins selected below`
- 点击（可选） `Include Required Plug-ins ...` 自动添加依赖
- 确保您的所有 `org.ruyisdk.*` 插件都被选中

3. **启动调试**
   - 点击 `Debug` 按钮
   - 新的 Eclipse 实例将以调试模式启动

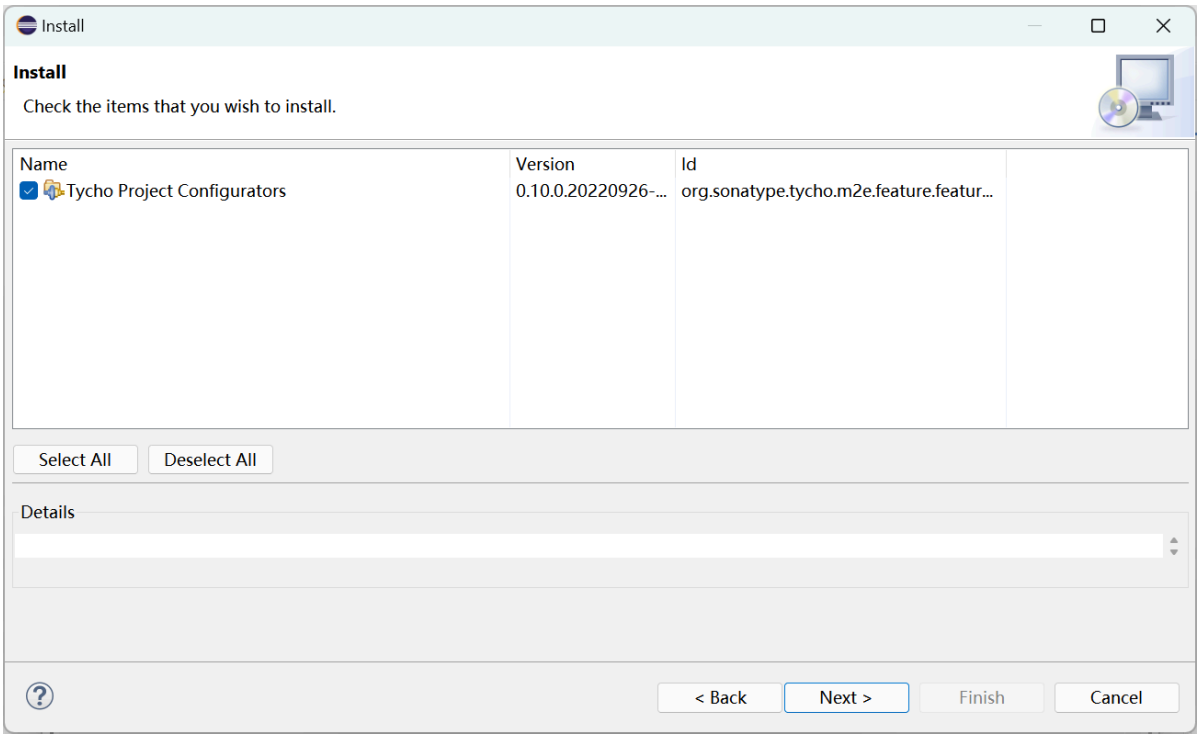
常见问题排查

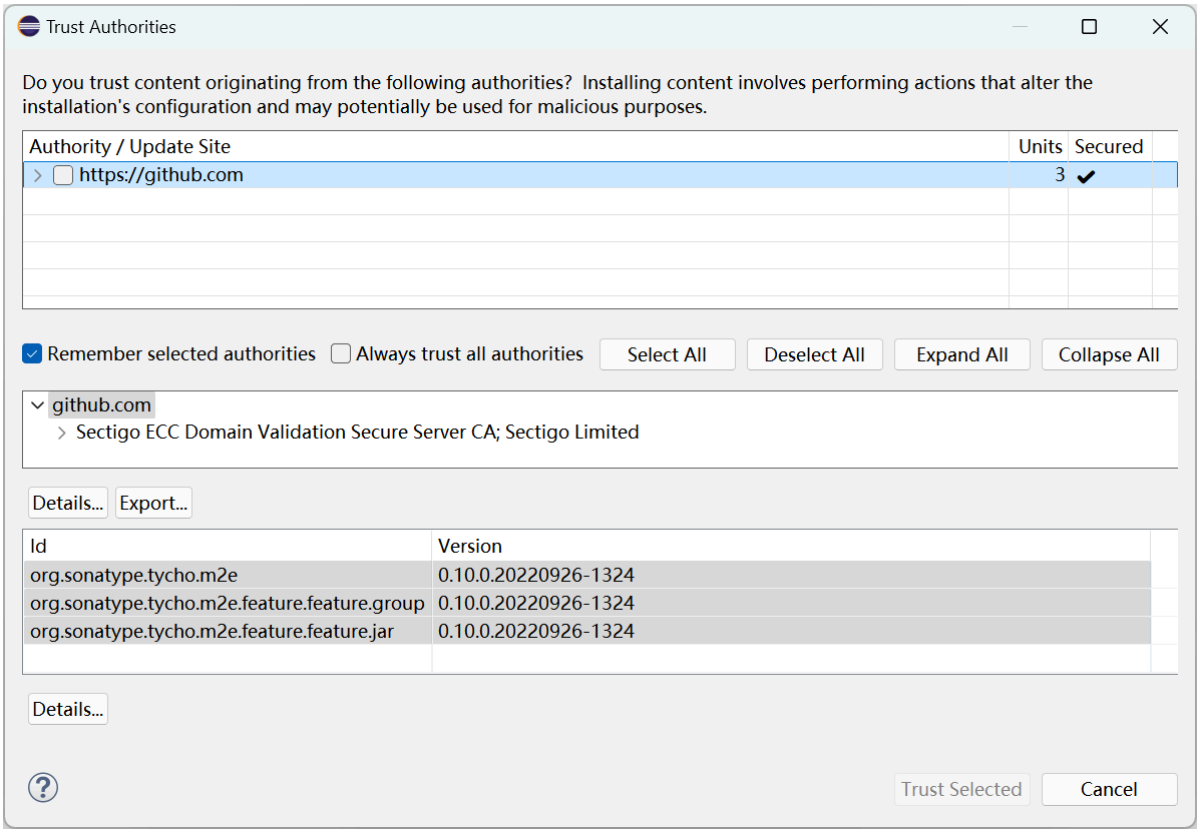
问题一：

提示缺少插件，验证后可以安装也可以不安装：



后续一路点击finish即可，给出必要的权限



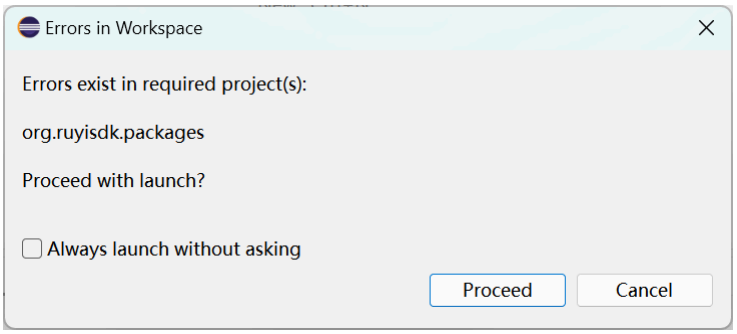


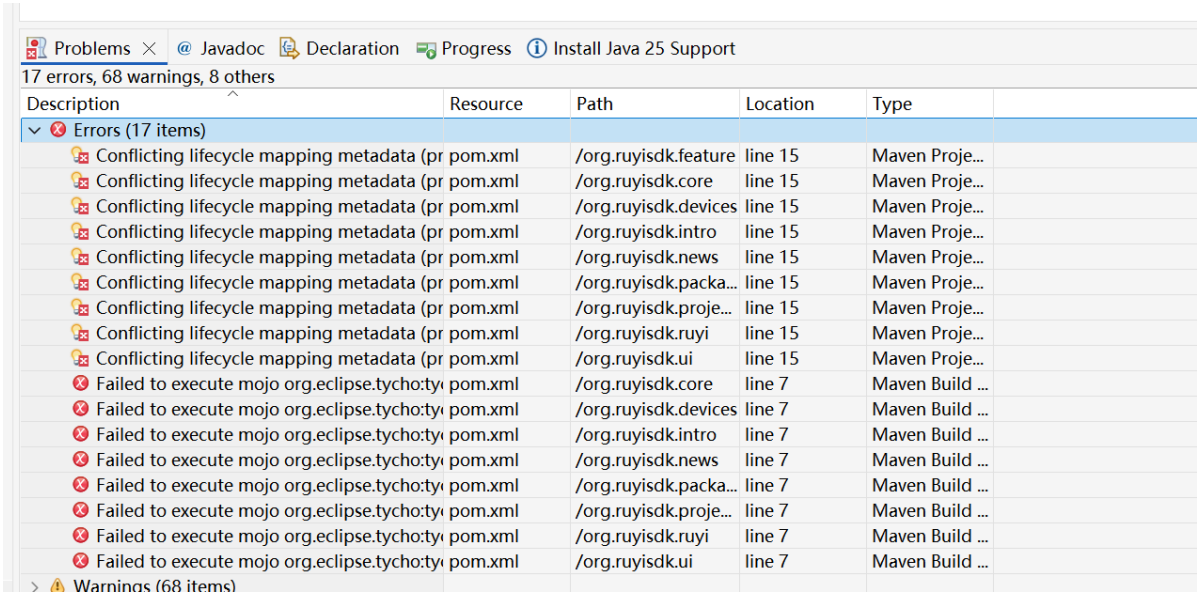
验证linux平台不点击第一张图的提示也可以安装，后续报错应该提示安装的无关

问题二：

知识参考：

https://github.com/eclipse-platform/eclipse.platform/blob/master/docs/FAQ/FAQ_How_do_I_create_an_external_tool_builder.md
https://help.eclipse.org/latest/index.jsp?topic=%2Forg.eclipse.platform.doc.user%2Fconcepts%2Fconcepts-exttools.htm





External Tool Builder 配置缺失

- .project 文件中配置了一个 API Analysis Builder
- 但对应的启动配置文件 .externalToolBuilders/org.eclipse.pde.api.tools.apiAnalysisBuilder.launch 不存在
- 这可能是从其他项目复制配置时遗留的

org.ruyisdk.**projectcreator**/.project 文件中引用了一个不存在的外部工具构建器启动配置。

```
<buildCommand>
      <name>org.eclipse.pde.SchemaBuilder</name>
      <arguments>
      </arguments>
    </buildCommand>
    <buildCommand>
      <name>org.eclipse.ui.externaltools.ExternalToolBuilder</name>
      <triggers>full,incremental,</triggers>
      <arguments>
        <dictionary>
          <key>LaunchConfigHandle</key>

<value>&lt;project&gt;/.externalToolBuilders/org.eclipse.pde.api.tools.apiAnalysi
sBuilder.launch</value>
        </dictionary>
      </arguments>
    </buildCommand>
    <buildCommand>
      <name>org.eclipse.m2e.core.maven2Builder</name>
      <arguments>
```

```
      </arguments>
    </buildCommand>


    <natures>
        <nature>org.eclipse.pde.api.tools.apiAnalysisNature</nature>
    </natures>
```

```
<!-- 删除了这个引用不存在配置文件的构建器 -->
<buildCommand>
  <name>org.eclipse.ui.externaltools.ExternalToolBuilder</name>
  ...
</buildCommand>
```

问题三：

Maven 生命周期映射冲突

- Eclipse m2e 不知道如何处理 Tycho 插件的某些构建步骤
- 添加 lifecycle-mapping 配置后，m2e 会忽略这些 Tycho 特定的 goal
- 实际构建仍由 Tycho 完成，只是 Eclipse 不会报错

Tycho 插件与 Eclipse m2e 的生命周期映射不兼容

这是典型的 Tycho/m2e 集成问题。问题有两类：

问题分析

1. 生命周期映射冲突您安装了两个冲突的 m2e Tycho 连接器：

```
org.eclipse.m2e.pde.connector (Eclipse 官方，较新)

org.sonatype.tycho.m2e (旧版本)问题 2：
```

"does not have an expanded version"这是因为 Eclipse 内的 m2e 尝试执行 Tycho 构建，但 Tycho 需要完整的 Maven 反应堆环境，在 IDE 内增量构建时会失败。🔧

解决方案方案：

禁用 Eclipse 内的 Maven 构建（推荐）

Tycho 项目最佳实践是：在命令行构建，在 Eclipse 内调试。

```
 <build>
    <pluginManagement>
      <plugins>
        <!-- m2e lifecycle mapping to avoid errors in Eclipse -->
        <plugin>
          <groupId>org.eclipse.m2e</groupId>
          <artifactId>lifecycle-mapping</artifactId>
          <version>1.0.0</version>
          <configuration>
            <lifecycleMappingMetadata>
              <pluginExecutions>
                <!-- Ignore all Tycho plugin executions in Eclipse -->
```

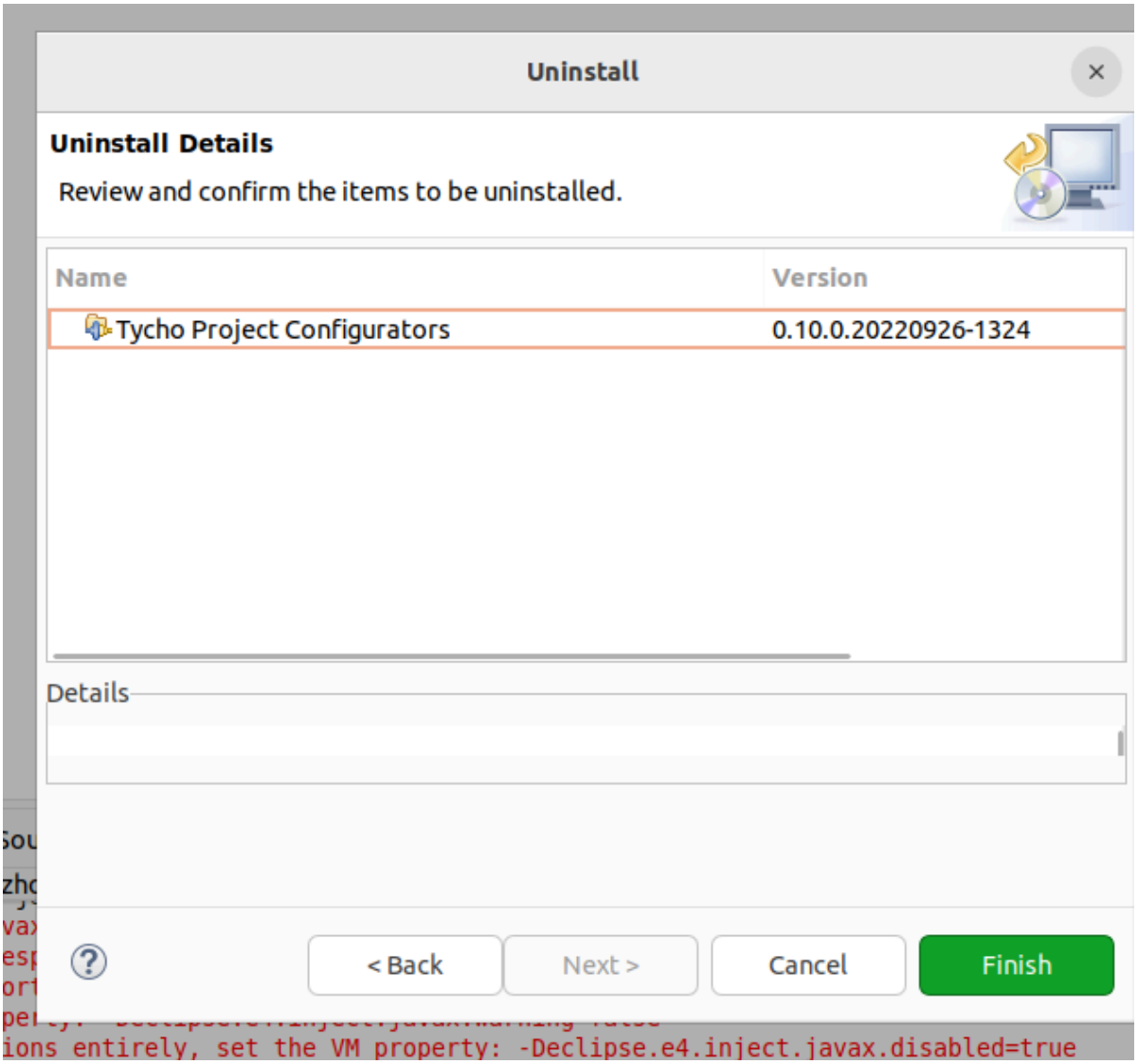
```
                <pluginExecution>
                  <pluginExecutionFilter>
                    <groupId>org.eclipse.tycho</groupId>
                    <artifactId>tycho-compiler-plugin</artifactId>
                    <versionRange>[0.0.1,)</versionRange>
                    <goals>
                      <goal>compile</goal>
                    </goals>
                  </pluginExecutionFilter>
                  <action>
                    <ignore />
                  </action>
                </pluginExecution>
                <pluginExecution>
                  <pluginExecutionFilter>
                    <groupId>org.eclipse.tycho</groupId>
                    <artifactId>tycho-packaging-plugin</artifactId>
                    <versionRange>[0.0.1,)</versionRange>
                    <goals>
                      <goal>build-qualifier</goal>
                      <goal>build-qualifier-aggregator</goal>
                      <goal>validate-id</goal>
                      <goal>validate-version</goal>
                      <goal>package-plugin</goal>
                      <goal>package-feature</goal>
                    </goals>
                  </pluginExecutionFilter>
                  <action>
                    <ignore />
                  </action>
                </pluginExecution>
                <pluginExecution>
                  <pluginExecutionFilter>
                    <groupId>org.eclipse.tycho</groupId>
                    <artifactId>tycho-source-plugin</artifactId>
                    <versionRange>[0.0.1,)</versionRange>
                    <goals>
                      <goal>plugin-source</goal>
                    </goals>
                  </pluginExecutionFilter>
                  <action>
                    <ignore />
                  </action>
                </pluginExecution>
                <pluginExecution>
                  <pluginExecutionFilter>
                    <groupId>org.eclipse.tycho</groupId>
                    <artifactId>target-platform-configuration</artifactId>
                    <versionRange>[0.0.1,)</versionRange>
                    <goals>
                      <goal>target-platform</goal>
                    </goals>
                  </pluginExecutionFilter>
                  <action>
                    <ignore />
                  </action>
```

```
            </pluginExecution>
          </pluginExecutions>
        </lifecycleMappingMetadata>
      </configuration>
    </plugin>
  </plugins>
</pluginManagement>
```

现在需要**卸载**冲突的旧版 m2e 连接器：

操作步骤（我这个方案必须操作）

第 1 步：卸载冲突的 m2e 连接器（就是因为安装了提示中的插件）



1. 在 Eclipse 中打开：Help → About Eclipse IDE → Installation Details
2. 在 Installed Software 标签页中找到：

- Tycho Project Configurators (或 org.sonatype.tycho.m2e)

1. 选中它，点击 Uninstall...
2. 重启 Eclipse

保留：m2e-pde 或 PDE Integration for m2e (这是 Eclipse 官方的)

> **第 2 步：禁用 Maven 自动构建（可选）**

在 Eclipse 中：

Window → Preferences → Maven

取消勾选 ☐ Automatically update Maven projects configuration

或者禁用自动构建：

Project → 取消勾选 "Build Automatically"

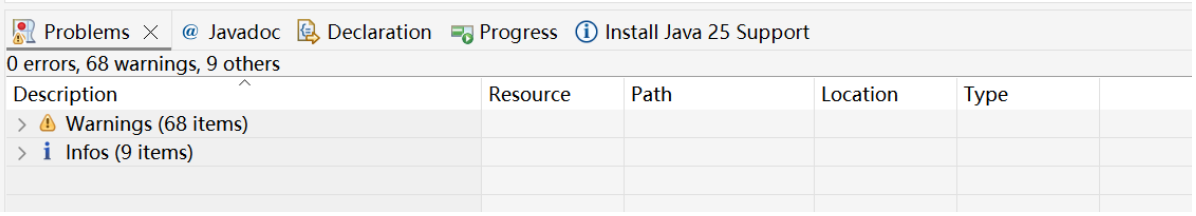
第 3 步：更新 Maven 项目配置

1. 右键点击 "ruyisdk-eclipse-plugins-parent" 项目
2. Maven → Update Project...
3. 勾选：（默认即可）可勾选强制-force更新
4. 选择所有项目
5. 点击 OK

> **第 4 步：使用命令行构建项目（可选）**

```
mvn clean verify -DskipTests
```

解决：



问题四 （linux不受影响）：

（是因为我用的windows平台的缘故，所以我撤销了修改，保留了报错，供参考）

启动core插件控制台**报错**：

```
This warning can be switched off on the Team > Git > Confirmations and Warnings
preference page.
java.io.IOException: Cannot run program "D:\ruyiSDK\test/ruyi": CreateProcess
error=193, %1 不是有效的 Win32 应用程序。
Ruyi is installed ? false
    at java.base/java.lang.ProcessBuilder.start(ProcessBuilder.java:1170)
    at java.base/java.lang.ProcessBuilder.start(ProcessBuilder.java:1089)
    at org.ruyisdk.ruyi.services.RuyiManager.isRuyiInstalled(RuyiManager.java:21)
    at org.ruyisdk.ruyi.jobs.CheckRuyiJob.isInstalled(CheckRuyiJob.java:61)
    at org.ruyisdk.ruyi.jobs.CheckRuyiJob.runCheck(CheckRuyiJob.java:25)
    at org.ruyisdk.ruyi.core.RuyiCore.lambda$0(RuyiCore.java:32)
    at org.eclipse.core.runtime.jobs.Job$1.run(Job.java:166)
    at org.eclipse.core.internal.jobs.Worker.run(Worker.java:63)
Caused by: java.io.IOException: CreateProcess error=193, %1 不是有效的 Win32 应用程
序。
    at java.base/java.lang.ProcessImpl.create(Native Method)
    at java.base/java.lang.ProcessImpl.<init>(ProcessImpl.java:506)
    at java.base/java.lang.ProcessImpl.start(ProcessImpl.java:159)
    at java.base/java.lang.ProcessBuilder.start(ProcessBuilder.java:1126)
    ... 7 more
```

```
java.io.IOException: Cannot run program "D:\ruyiSDK\test/ruyi": CreateProcess
error=193, %1 不是有效的 Win32 应用程序。
    at java.base/java.lang.ProcessBuilder.start(ProcessBuilder.java:1170)
    at java.base/java.lang.ProcessBuilder.start(ProcessBuilder.java:1089)
    at org.ruyisdk.ruyi.services.RuyiManager.isRuyiInstalled(RuyiManager.java:21)
    at org.ruyisdk.ruyi.jobs.CheckRuyiJob.isInstalled(CheckRuyiJob.java:62)
    at org.ruyisdk.ruyi.jobs.CheckRuyiJob.runCheck(CheckRuyiJob.java:25)
    at org.ruyisdk.ruyi.core.RuyiCore.lambda$0(RuyiCore.java:32)
    at org.eclipse.core.runtime.jobs.Job$1.run(Job.java:166)
    at org.eclipse.core.internal.jobs.Worker.run(Worker.java:63)
Caused by: java.io.IOException: CreateProcess error=193, %1 不是有效的 Win32 应用程
序。
    at java.base/java.lang.ProcessImpl.create(Native Method)
    at java.base/java.lang.ProcessImpl.<init>(ProcessImpl.java:506)
    at java.base/java.lang.ProcessImpl.start(ProcessImpl.java:159)
    at java.base/java.lang.ProcessBuilder.start(ProcessBuilder.java:1126)
    ... 7 more
```

为什么会出现 "不是有效的 Win32 应用程序" 错误?

1. 路径混用：D:\ruyiSDK\test/ruyi 混用了 ` 和 /
2. **缺少 .exe**：Windows 上尝试执行 ruyi（没有扩展名）
3. **可能的情况**：
    - 文件不存在 - 文件是 Linux 二进制文件（不能在 Windows 上运行）
    - 文件不是可执行文件 现在代码已经修复，能够：

      ✅ 在 Windows 上查找 ruyi.exe - ✅ 在 Linux/Mac 上查找 ruyi - ✅ 使用正确的路径分隔符

```
修改的（未采用）：
  /**
     * 获取 Ruyi 可执行文件的完整路径（跨平台）
     */
    private static String getRuyiExecutablePath() {
        String installPath = RuyiFileUtils.getInstallPath();
        String ruyiExeName = isWindows() ? "ruyi.exe" : "ruyi";
        return installPath + File.separator + ruyiExeName;
    }

    /**
     * 判断是否为 Windows 系统
     */
    private static boolean isWindows() {
        String os = System.getProperty("os.name").toLowerCase();
        return os.contains("win");
    }

    public static boolean isRuyiInstalled() {
        try {
            Process process = new ProcessBuilder(getRuyiExecutablePath(), "-
V").start();
            return process.waitFor() == 0;
```

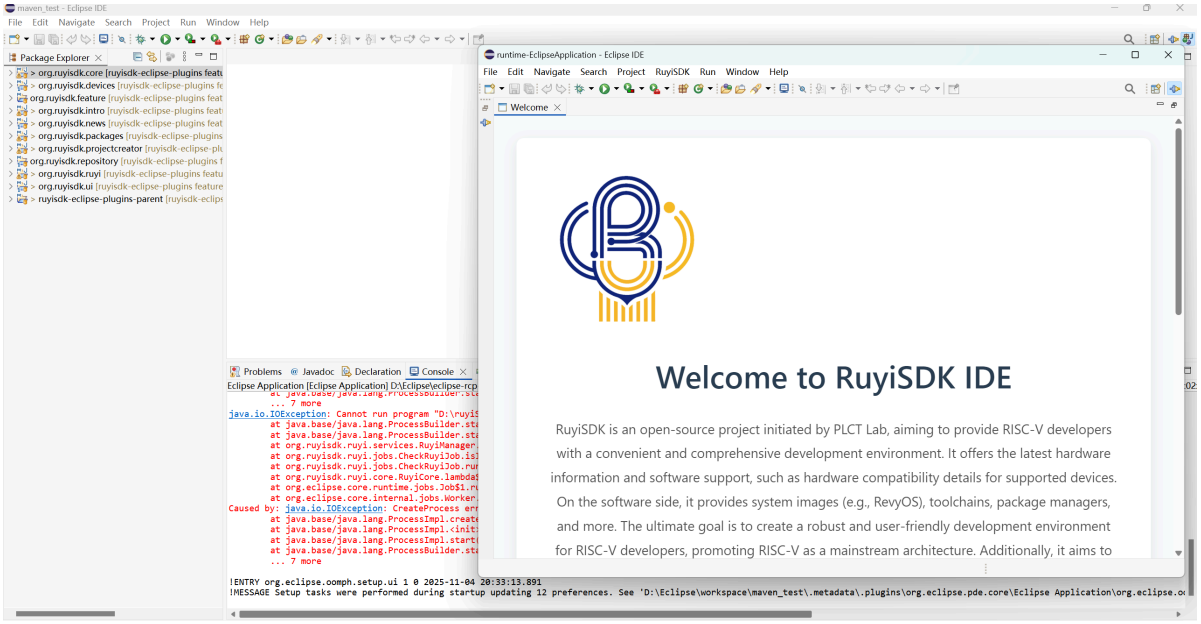
```
        } catch (IOException | InterruptedException e) {
            e.printStackTrace();
            return false;
        }
    }

    public static RuyiVersion getInstalledVersion() {
        try {
            Process process = new ProcessBuilder(getRuyiExecutablePath(), "-
V").start();

            try (BufferedReader reader = new BufferedReader(
                    new InputStreamReader(process.getInputStream()))) {

                // 仅读取首行内容
                String firstLine = reader.readLine();
                if (firstLine == null) {
                    return null;
                }

                // 精准截取版本号部分
                String prefix = "Ruyi ";
                if (firstLine.startsWith(prefix)) {
                    // 截断字符串并清理首尾空格
                    String versionStr = firstLine
                        .substring(prefix.length())
                        .trim();

                    // 正则校验版本号格式（如0.31.0）
                    if (versionStr.matches("^\\d+\\.\\d+\\.\\d+$")) {
                        return RuyiVersion.parse(versionStr);
                    }
                }
            }
        } catch (IOException e) {
            // 可在此处添加日志输出
        }
        return null;
    }

    public static RuyiVersion getLatestVersion(){
        String archSuffix = SystemInfo.detectArchitecture().getSuffix();
        RuyiVersion version  = null;
        try {
            RuyiReleaseInfo info = RuyiAPI.getLatestRelease(archSuffix);
            version  = info.getVersion();
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        return version;
    }
```

未修改：

```
public static boolean isRuyiInstalled() {
```

```
        try {
            Process process = new
ProcessBuilder(RuyiFileUtils.getInstallPath()+"/ruyi", "-V").start();
//           Process process =
Runtime.getRuntime().exec(RuyiFileUtils.getInstallPath()+"/ruyi -V");
            return process.waitFor() == 0;
        } catch (IOException | InterruptedException e) {
            e.printStackTrace();
            return false;
        }
    }

    public static RuyiVersion getInstalledVersion() {
        try {
            Process process = new
ProcessBuilder(RuyiFileUtils.getInstallPath()+"/ruyi", "-V").start();

            try (BufferedReader reader = new BufferedReader(
                    new InputStreamReader(process.getInputStream()))) {

                // 仅读取首行内容
                String firstLine = reader.readLine();
                if (firstLine == null) {
                    return null;
                }

                // 精准截取版本号部分
                String prefix = "Ruyi ";
                if (firstLine.startsWith(prefix)) {
                    // 截断字符串并清理首尾空格
                    String versionStr = firstLine
                        .substring(prefix.length())
                        .trim();

                    // 正则校验版本号格式（如0.31.0）
                    if (versionStr.matches("^\\d+\\.\\d+\\.\\d+$")) {
                        return RuyiVersion.parse(versionStr);
                    }
                }
            }
        } catch (IOException e) {
            // 可在此处添加日志输出
        }
        return null;
    }

    public static RuyiVersion getLatestVersion(){
        String archSuffix = SystemInfo.detectArchitecture().getSuffix();
        RuyiVersion version  = null;
        try {
            RuyiReleaseInfo info = RuyiAPI.getLatestRelease(archSuffix);
            version  = info.getVersion();
        } catch (Exception e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
```

```
        return version;
    }
```



调试技巧

1. **查看控制台输出**
   - 两个 Eclipse 实例都有各自的控制台
   - 原 Eclipse：显示启动信息
   - 运行时 Eclipse：显示您的插件输出
2. **使用 Error Log 视图**
   - 在运行时 Eclipse 中：`Window` → `Show View` → `Error Log`
   - 可以看到插件运行时的错误和警告
3. **添加日志输出**

与传统插件项目的区别

你提到的 Medium 文章中的插件是传统的 Eclipse 插件项目，而我用的是 **Maven/Tycho** 项目（其实一样的）：

特性	传统插件项目	Maven/Tycho 项目
构建工具	Eclipse PDE	Maven + Tycho
依赖管理	MANIFEST.MF	MANIFEST.MF + pom.xml
调试方式	完全相同 ✓	完全相同 ✓
命令行构建	不支持	支持 (mvn clean verify)

调试方式完全相同！Maven/Tycho 只是改变了构建方式，不影响 Eclipse 内的调试体验。

然后按 `Debug As` → `Eclipse Application` 启动，看断点是否能够命中。

> 文档更新时间：2025/11/05
>
> 参考Eclipse版本：2025.9