

CodeArts IDE

常见问题

文档版本 01
发布日期 2024-04-12



版权所有 © 华为技术有限公司 2024。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

商标声明



HUAWEI和其他华为商标均为华为技术有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

安全声明

漏洞处理流程

华为公司对产品漏洞管理的规定以“漏洞处理流程”为准，该流程的详细内容请参见如下网址：

<https://www.huawei.com/cn/psirt/vul-response-process>

如企业客户须获取漏洞信息，请参见如下网址：

<https://securitybulletin.huawei.com/enterprise/cn/security-advisory>


目录

1 登录常见问题	1
1.1 如何登录华为云	1
1.2 登录华为云失败	1
2 CodeArts IDE for C/C++	3
2.1 Huawei C/C++组件激活失败常见问题	3
2.1.1 输出窗口可获得更多日志信息	3
2.1.2 Error: Unable to open check cache file for write	3
2.1.3 Error: command 'clangd.applyFix' already exists	3
2.1.4 Error: wecode-db port file has no content. Make sure there is space available for the folder "/tmp"	4
2.1.5 WeCode-DB is unable to watch for file changes in this large workspace	4
2.2 外部 terminal 配置方法	4
2.3 CMake Build Tool 插件运行调试时中文乱码的问题	6
3 CodeArts IDE for RemoteShell	15
3.1 proxy 配置常见问题	15
3.1.1 登录远程主机，proxy 该如何配置	15
3.1.2 proxy 配置修改后不生效	16
3.1.3 proxy 配置检查连接失败	16
3.2 新建主机连接常见问题	17
3.2.1 新建主机连接失败，可根据以下可能原因排查重试	17
3.2.2 遇到如下连接失败的场景，该如何解决	17
3.3 增加用户连接常见问题	17
3.3.1 用户尝试打开较多终端时，受到限制	17
4 CodeArts IDE for Java	18
4.1 等待语言服务初始化完成	18
4.2 从 IDE 直接跳到实际文件存放路径的功能	18
4.3 文件树上有些修改了会更改颜色，有些不会变或者文件中有 Error/Warning/Info 类型的错误，但是文件树上没有高亮的提示	19
4.4 终端输出中文乱码问题	20
4.5 默认的 Maven 配置文件 settings.xml 配置参考	22
4.6 怎么设置代码片段和代码模板	25
4.7 xml 文件中的标签，写入结束标签</>的时候，需要自动补全标签对	29

4.8 识别并展示 TODO 列表.....	30
------------------------	----

1 登录常见问题

1.1 如何登录华为云

步骤1 单击 CodeArts IDE 右上角，打开登录窗口。

步骤2 输入账号密码，单击登录。

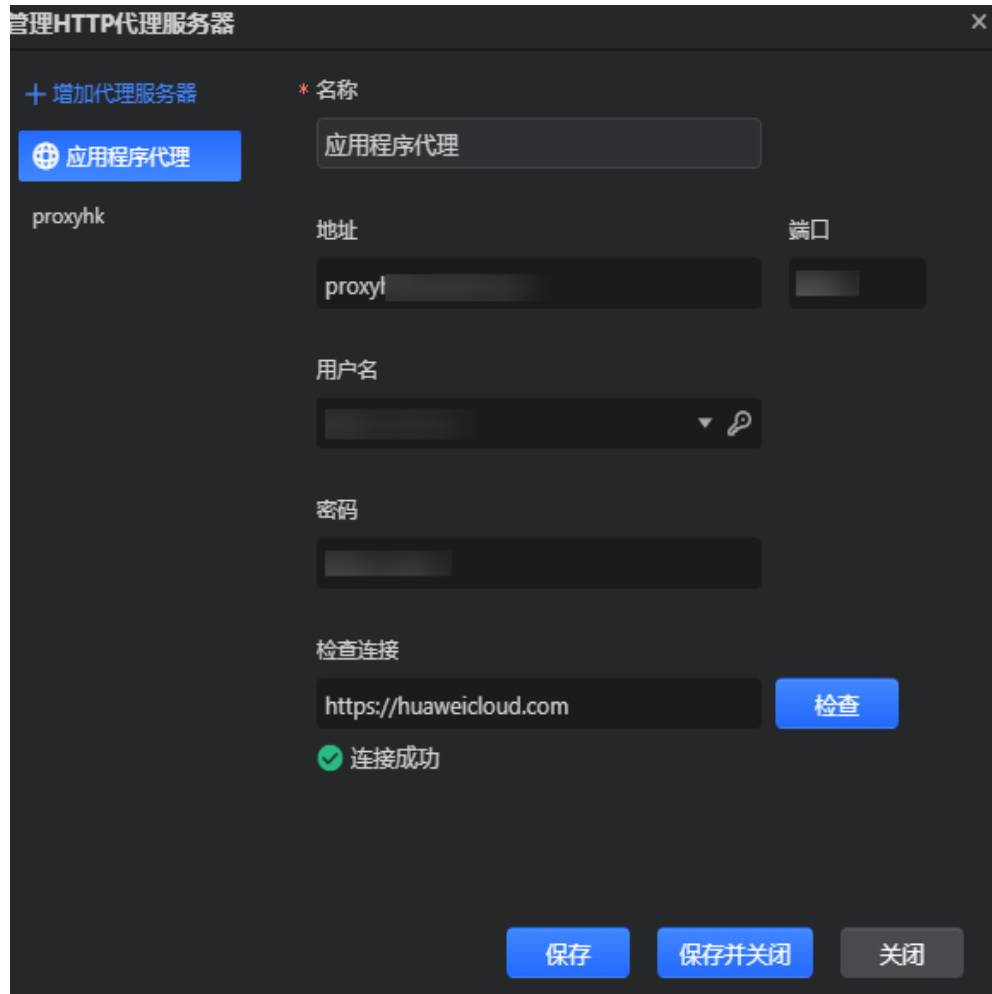
步骤3 登录成功后，右上角会显示用户名。

----结束

1.2 登录华为云失败

解决办法：检查本机是否需要配置proxy登录，如已配置请检查proxy是否有效。在"管理代理服务器"弹窗中配置全局代理，可以成功登录RemoteShell。





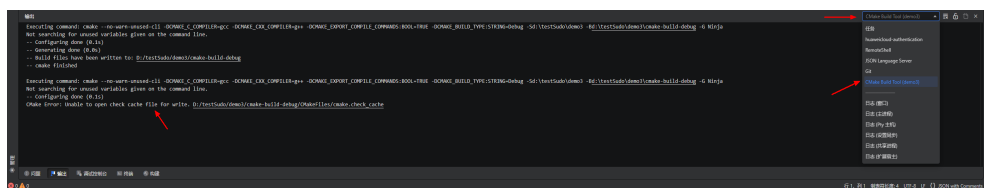
2 CodeArts IDE for C/C++

2.1 Huawei C/C++组件激活失败常见问题

2.1.1 输出窗口可获得更多日志信息



该错误表明CMake项目配置失败，并将错误的信息日志打印输出到输出窗口，单击“是”，以切换到输出窗口获取更多详细信息。此外，可以通过下拉菜单进行手动切换。



2.1.2 Error: Unable to open check cache file for write

该错误表明CodeArts IDE在此工作区文件夹无写入权限，需用户自行确认此文件夹的读写权限。

2.1.3 Error: command 'clangd.applyFix' already exists

该错误表明命令 clangd.applyFix 被重复注册了，原因是插件冲突，目前已知的冲突插件包括clangd, 5G-clangd, VSCode C/C++ Plugin, Nextcode Reference，出现此类错误时，请在插件列表中卸载以上插件。

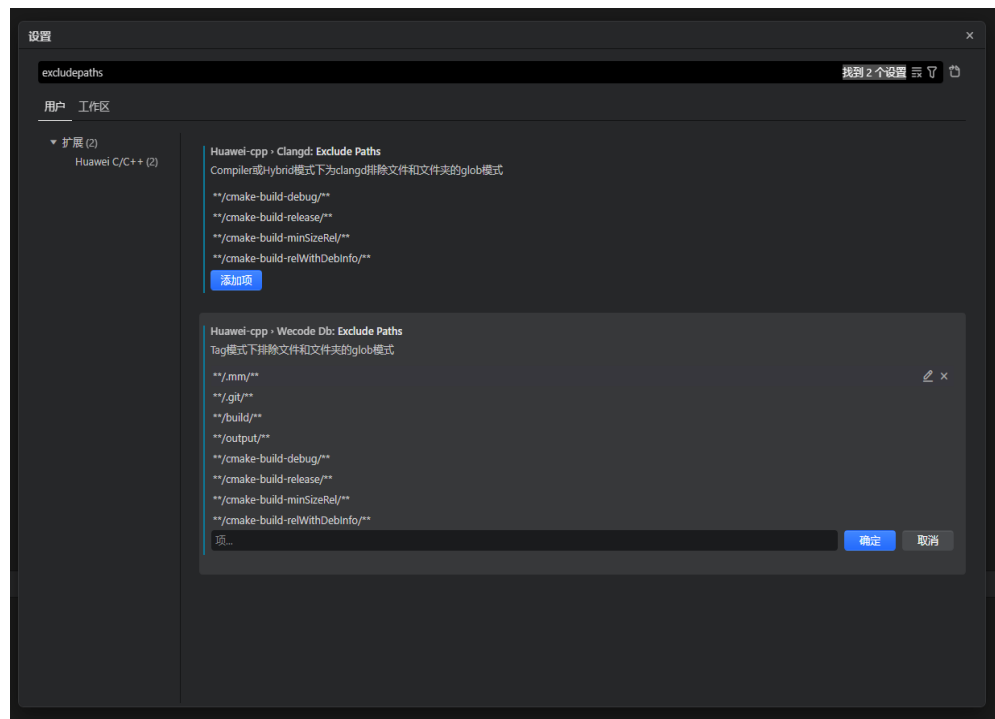
2.1.4 Error: wecode-db port file has no content. Make sure there is space available for the folder “/tmp”

该错误表明WeCode-DB无法创建端口文件，可检查下当前的磁盘使用情况，确认磁盘空间占用是否已满，清理部分空间后重新尝试。

2.1.5 WeCode-DB is unable to watch for file changes in this large workspace

该警告表明 WeCode-DB 的文件监视程序已经耗尽文件句柄，文件句柄通常由操作系统决定，可通过`cat /proc/sys/fs/inotify/max_user_watches`查看，这种情况通常发生在工程非常大，包含的目录非常多的情况下，有以下两种方法解决：

- 在CodeArts IDE的设置中搜索excludepaths，尽可能的将较大的文件夹排除，如output,build目录，例如：

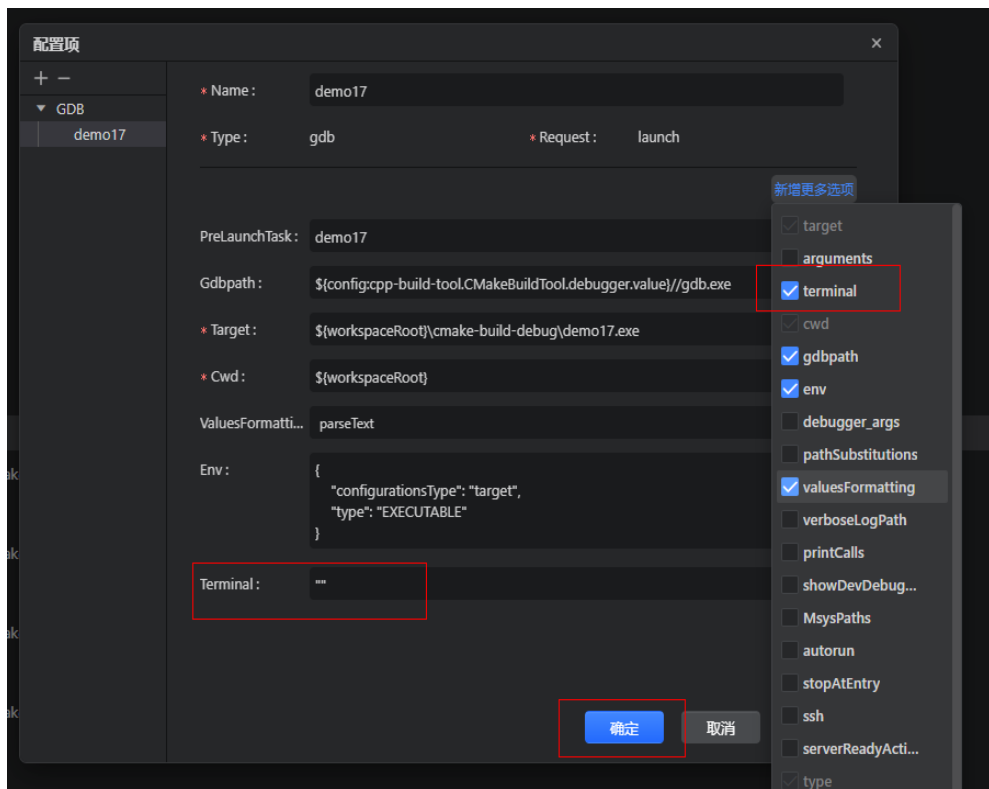
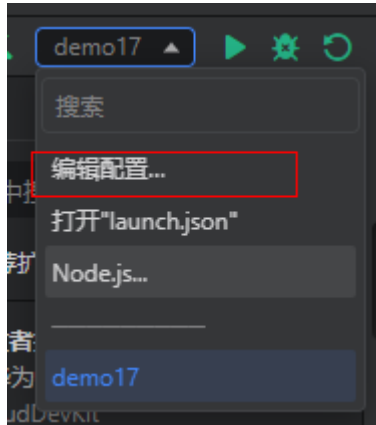


- 增加操作系统的可监控文件句柄数量，执行以下两条命令，将数量增加至524288,该操作最高将会多消耗约 540MB 的内存：

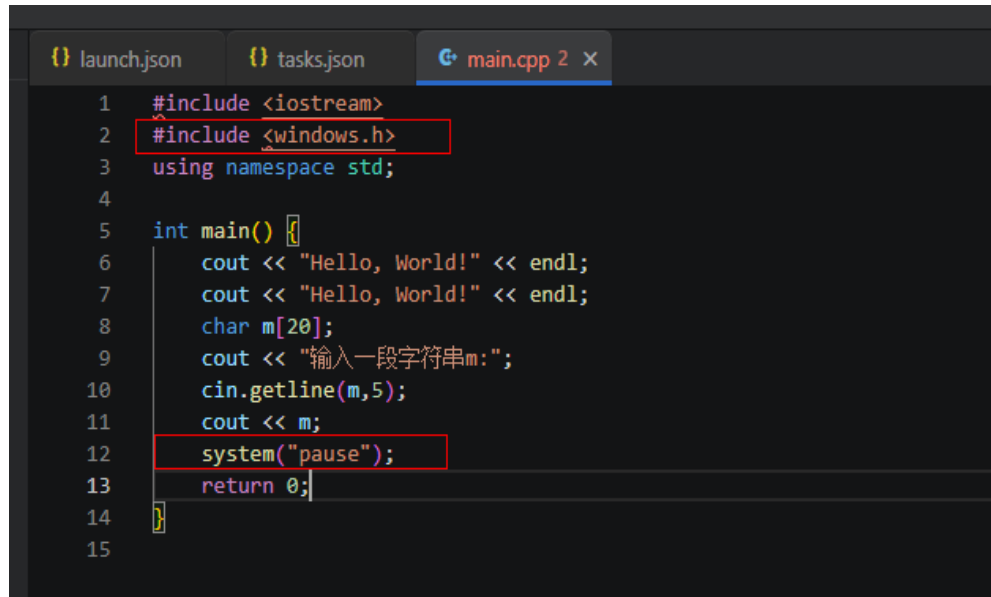
```
sudo sh -c "echo fs.inotify.max_user_watches=524288 >> /etc/sysctl.conf"  
sudo sysctl -p
```

2.2 外部 terminal 配置方法

首先编辑调试配置项。



配置后，运行时，在外部terminal输入字符串，回车后，terminal自动关闭。
如果需要在外部terminal里查看输出，在代码里面添加以下代码：



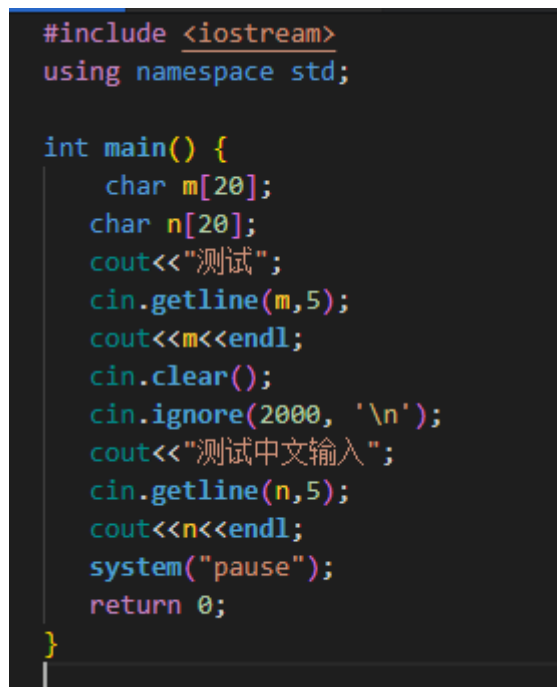
```
1 #include <iostream>
2 #include <windows.h>
3 using namespace std;
4
5 int main() {
6     cout << "Hello, World!" << endl;
7     cout << "Hello, World!" << endl;
8     char m[20];
9     cout << "输入一段字符串m:";
10    cin.getline(m,5);
11    cout << m;
12    system("pause");
13    return 0;
14 }
15
```

2.3 CMake Build Tool 插件运行调试时中文乱码的问题

现象:

1. 文件运行在内部终端的乱码

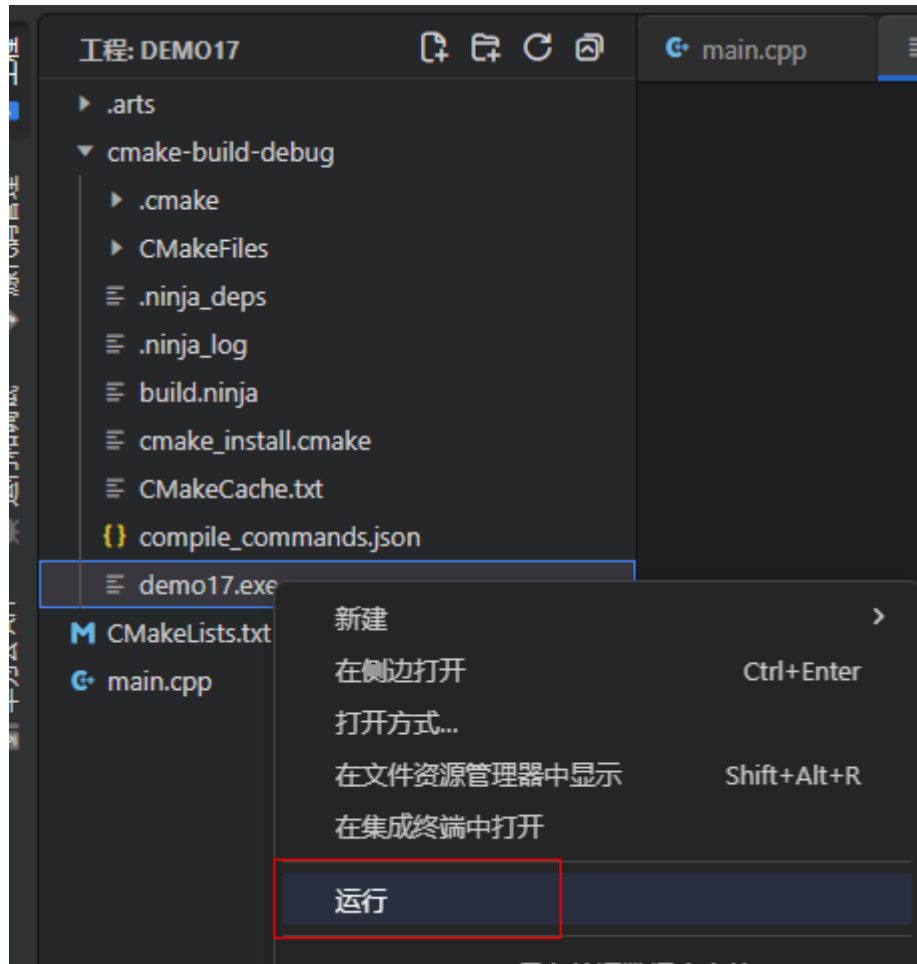
main.cpp文件中有中文文字



```
#include <iostream>
using namespace std;

int main() {
    char m[20];
    char n[20];
    cout<<"测试";
    cin.getline(m,5);
    cout<<m<<endl;
    cin.clear();
    cin.ignore(2000, '\n');
    cout<<"测试中文输入";
    cin.getline(n,5);
    cout<<n<<endl;
    system("pause");
    return 0;
}
```

构建生成可执行文件，右键单击运行

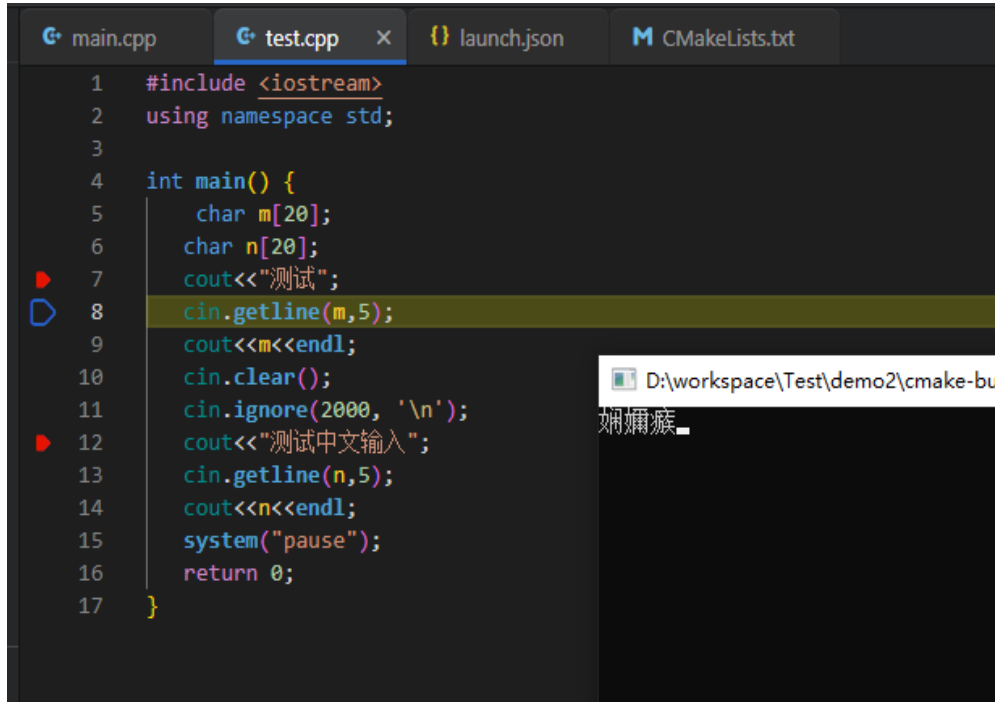


运行结果出现中文乱码：

```
PS D:\workspace\Test\demo17> d:\workspace\Test\demo17\cmake-build-debug\demo17.exe  
乱码
```

2.运行调试使用外部 终端出现乱码

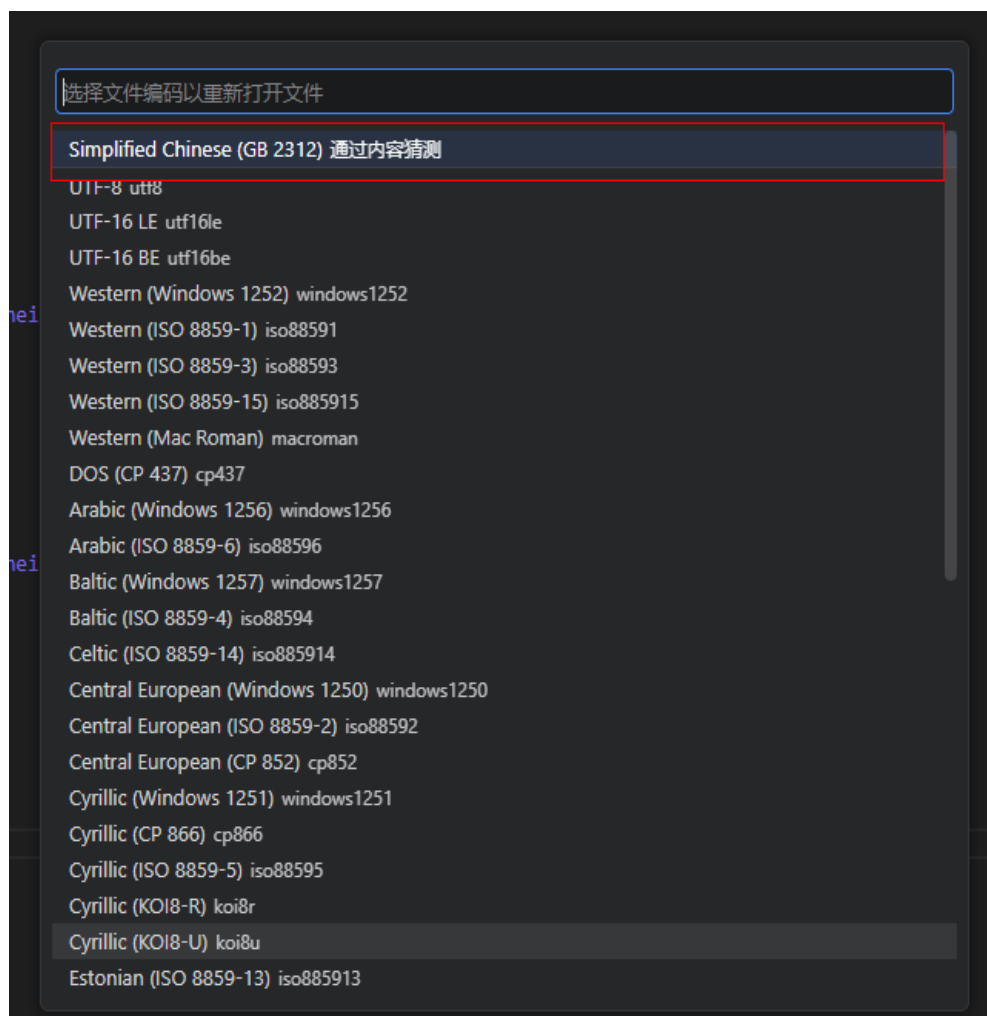
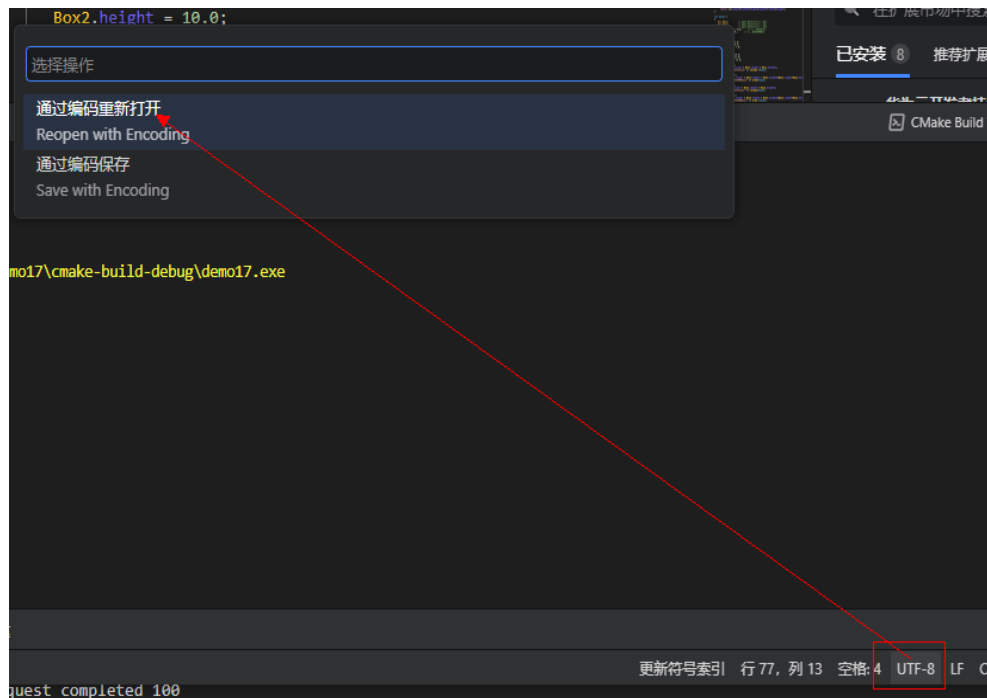
文件里面有输入输出，配置外部terminal, 在terminal中打印出来的中文乱码



解决办法:

- 方法一

单击编码类型, 选择 Reopen with Encoding, 选择 GB 2312编码类型 (或选择 gbk 、 GB 18030类型编码)



选择之后，文件里面的中文乱码

```
#include <iostream>
using namespace std;

int main() {
    char m[20];
    char n[20];
    cout<<"嫵嫵嫵";
    cin.getline(m,5);
    cout<<m<<endl;
    cin.clear();
    cin.ignore(2000, '\n');
    cout<<"嫵嫵嫵涓嶅緇涓嶇緇";
    cin.getline(n,5);
    cout<<n<<endl;
    system("pause");
    return 0;
}
```

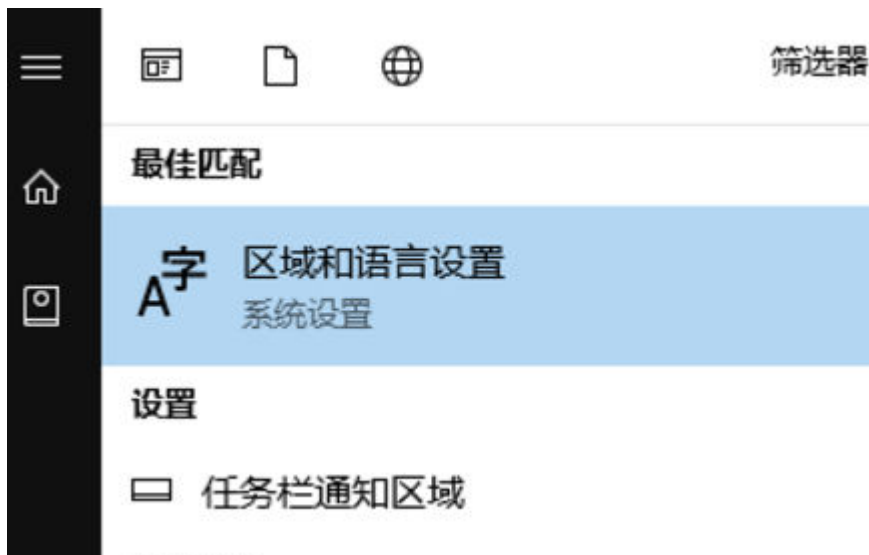
将乱码文字改为简体中文，重新构建，生成可执行文件，运行后

```
PS D:\workspace\Test\demo17> d:\workspace\Test\demo17\cmake-build-debug\demo17.exe
测试
```

- 方法二

修改cwd的默认编码格式，将其改为utf-8，修改方法如下：

1、在开始菜单中搜索“区域与语言设置”



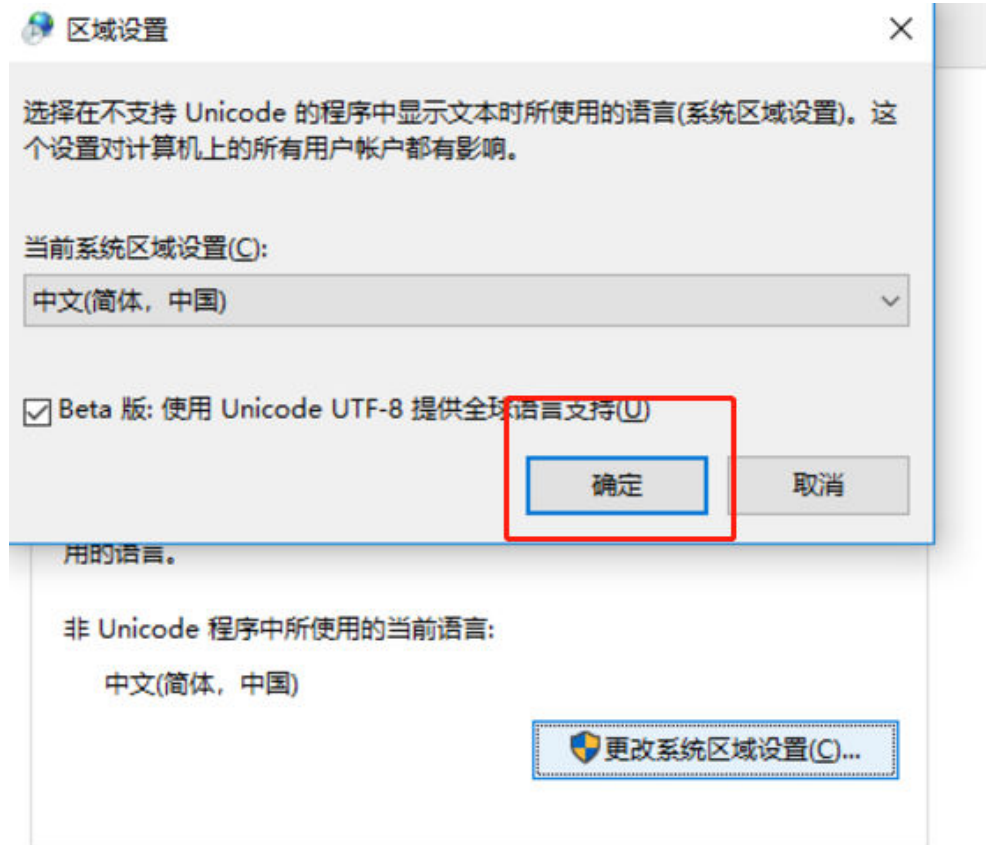
2.选择管理语言设置



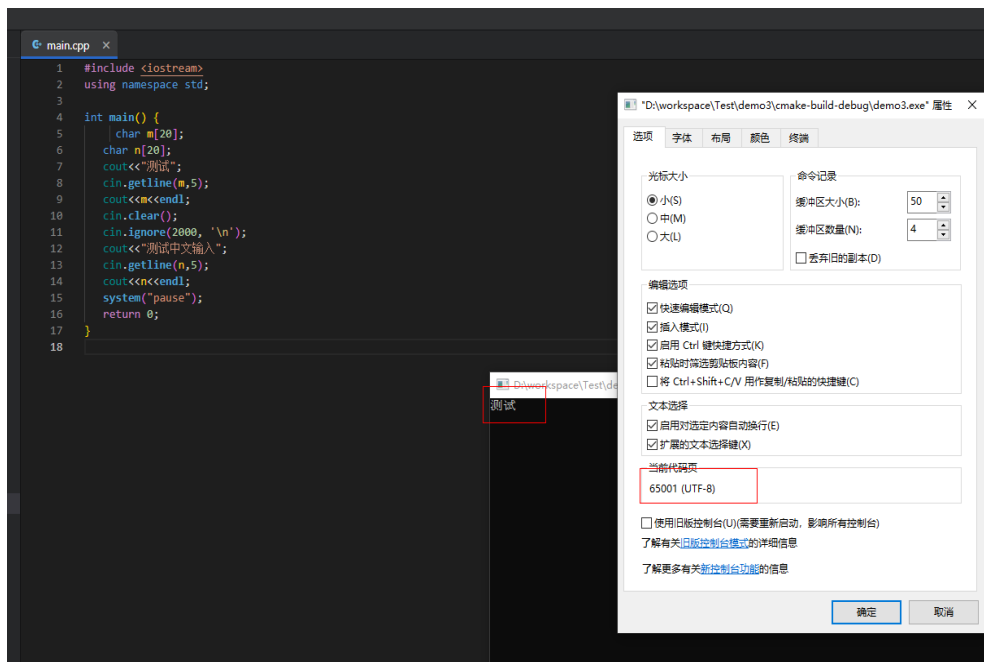
3.在弹出的对话框中的管理选项卡中单击“更改系统区域设置”



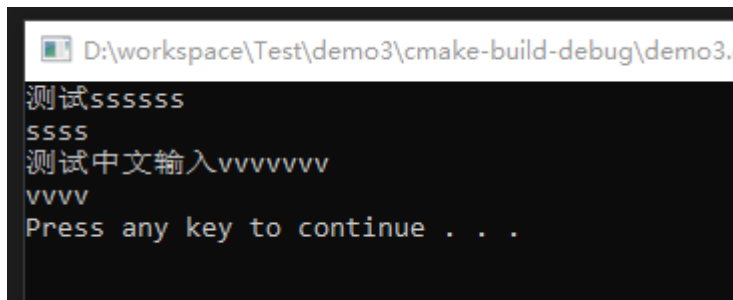
4.勾选Unicode UTF_8 并单击确定和应用



重启系统，并验证：



运行结果

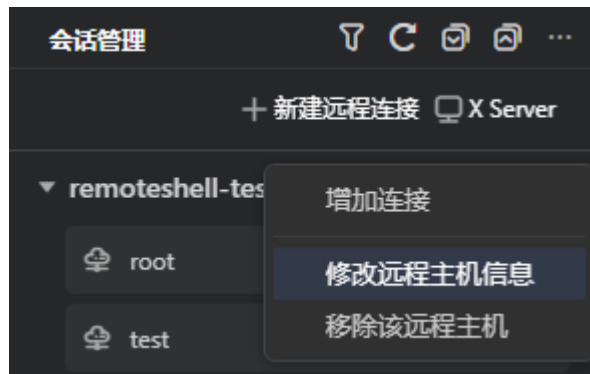


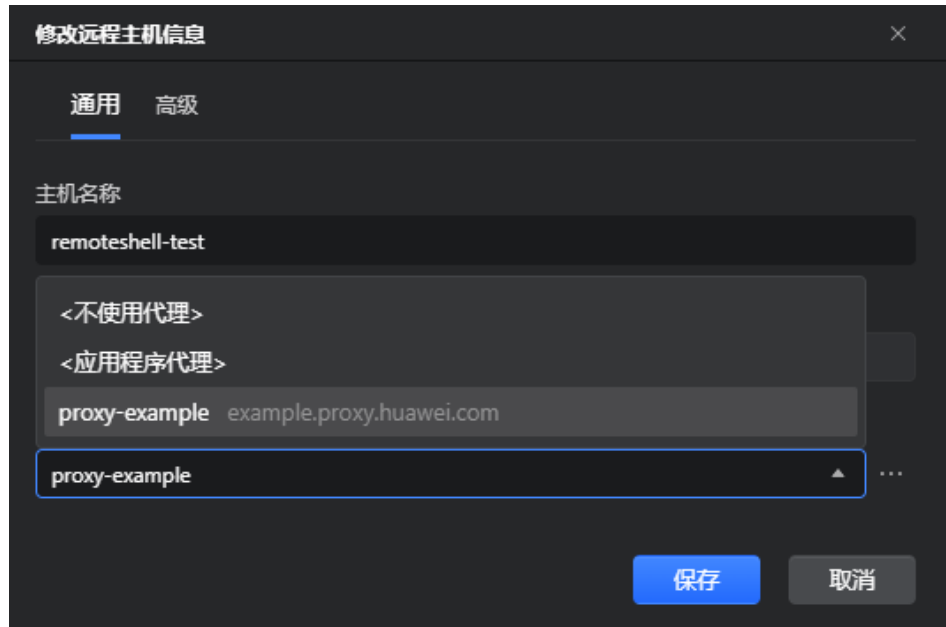
3 CodeArts IDE for RemoteShell

3.1 proxy 配置常见问题

3.1.1 登录远程主机，proxy 该如何配置

可在“管理代理”中新增、删除或修改代理配置。远程主机如需配置proxy或者修改当前使用的proxy，可单击“修改远程主机信息”进行修改。



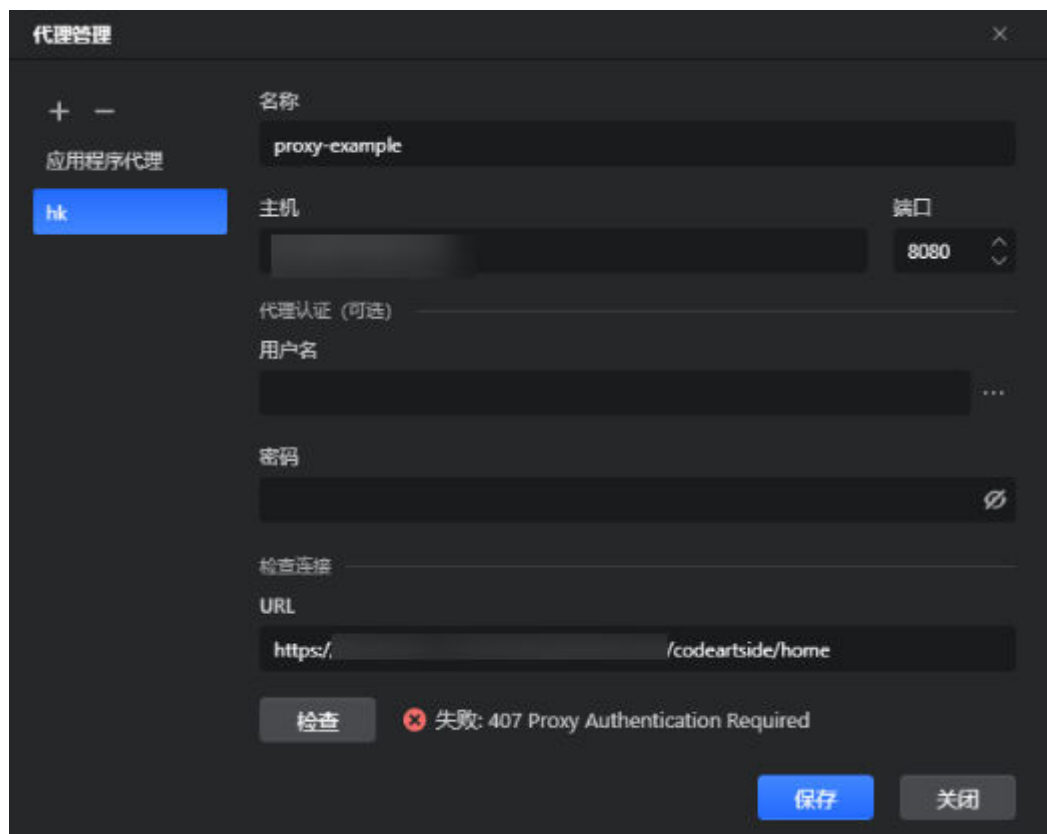


3.1.2 proxy 配置修改后不生效

解决办法：可以尝试单击“关闭该主机的所有连接”，再重新操作连接。

3.1.3 proxy 配置检查连接失败

解决办法：检查proxy的地址和端口是否有效，用户名或密码是否输入错误，修改后重新尝试。



3.2 新建主机连接常见问题

3.2.1 新建主机连接失败，可根据以下可能原因排查重试

- 检查用户名、密码是否输入正确。
- 确保SSH服务已开启（默认端口为22），如果修改SSH默认的22端口，请确认要连接的远程端口已配置且生效，并添加对应的防火墙规则。
- 确认当前使用proxy代理配置出口IP的全部地址的远程端口（SSH默认端口为22）已经允许。
- 检查弹性云服务器的防火墙是否允许当前该主机使用的proxy代理配置的出口IP作为入口IP。
- 请确认网络ACL没有禁用当前该主机使用的proxy代理配置的出口IP。
- 请检查与远程主机之间的网络连通，如需要可绑定弹性公网IP(EIP)。

3.2.2 遇到如下连接失败的场景，该如何解决

- 在连接远程主机时，遇到"407 Proxy Authentication Required"报错时，即为代理认证信息错误，该代理需要用户认证。
解决办法：需在“管理代理”弹窗中，找到所使用的代理，并为它输入用户名和密码。
- 在连接远程主机时，遇到"connect ETIMEDOUT"报错时，即为网络超时引起。
解决办法：确认proxy是否有效。
- 在连接远程主机时，遇到"All configured authentication methods failed"报错时，即为错误的密码或主机IP引起。
解决办法：修改为正确的密码后，尝试重新连接。或尝试移除该远程主机后，使用正确的主机地址进行新建远程连接。

3.3 增加用户连接常见问题

3.3.1 用户尝试打开较多终端时，受到限制

单个用户目前仅支持同时创建7个终端。

4 CodeArts IDE for Java

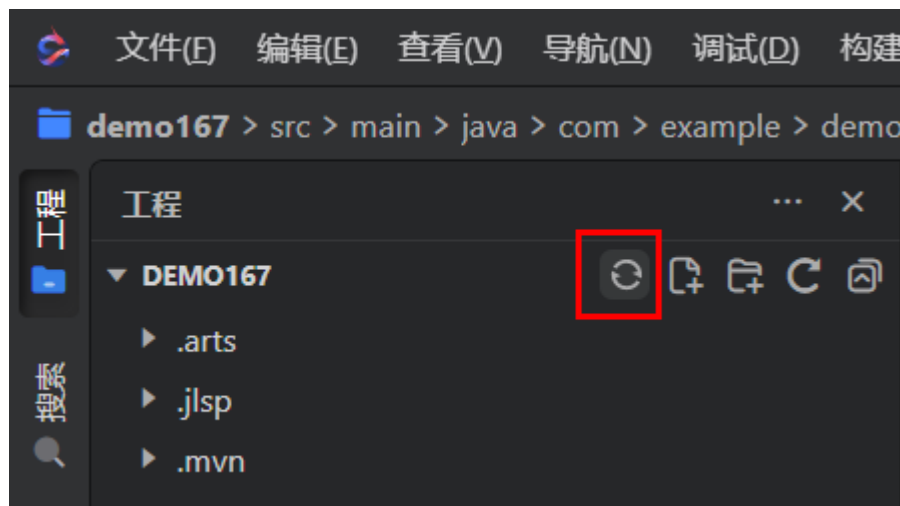
4.1 等待语言服务初始化完成

依赖项视图长时间出现“等待语言服务初始化完成”的提示信息，Maven/Gradle视图也一直显示“等待语言服务初始化完成”。

请检查系统环境是否是虚拟机环境，出现这种问题一般是虚拟机环境的系统缺失相关证书导致，如果是虚拟机环境，请按照如下步骤修复此问题。

步骤1 单击CodeArts IDE for Java左下角的“管理->设置”菜单，在设置窗口中搜索代理关键字，在搜索出的结果中，找到Http: Proxy Strict SSL设置项，取消勾选该设置项。

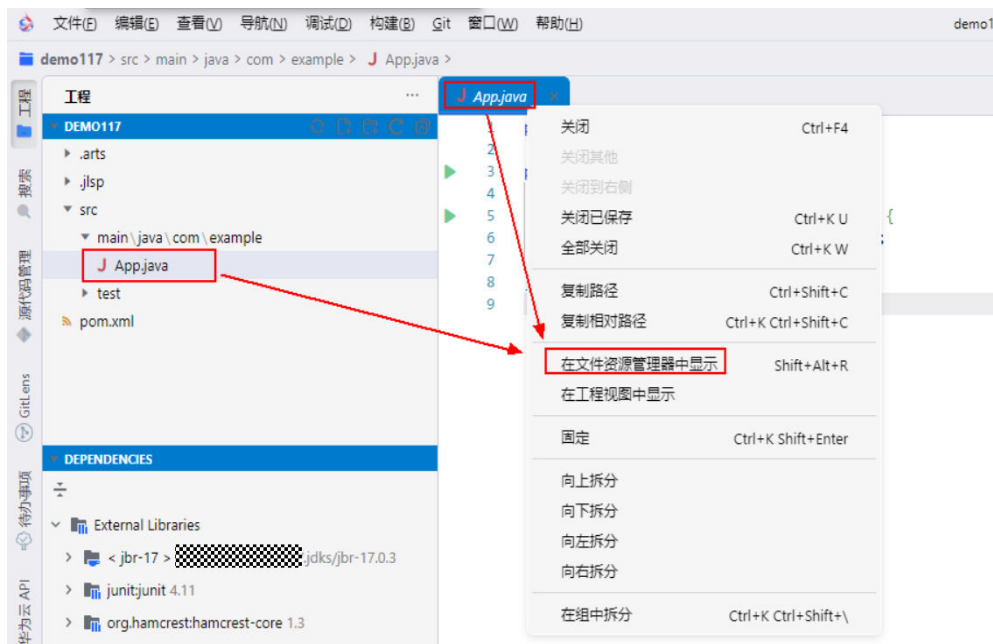
步骤2 然后再单击工程视图的“重新加载工程”按钮，重新加载工程即可。



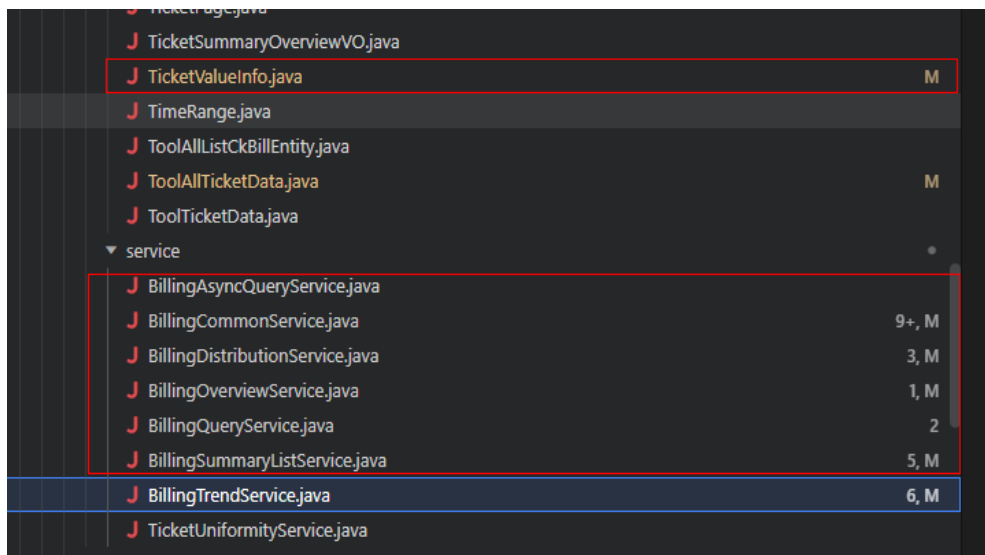
----结束

4.2 从 IDE 直接跳到实际文件存放路径的功能

在当前文件树或者已经打开的文件标题区域右键，选择“在文件资源管理器中显示”即可直接跳转到文件的实际存放路径。



4.3 文件树上有些修改了会更改颜色，有些不会变或者文件中有 Error/Warning/Info 类型的错误，但是文件树上没有高亮的提示



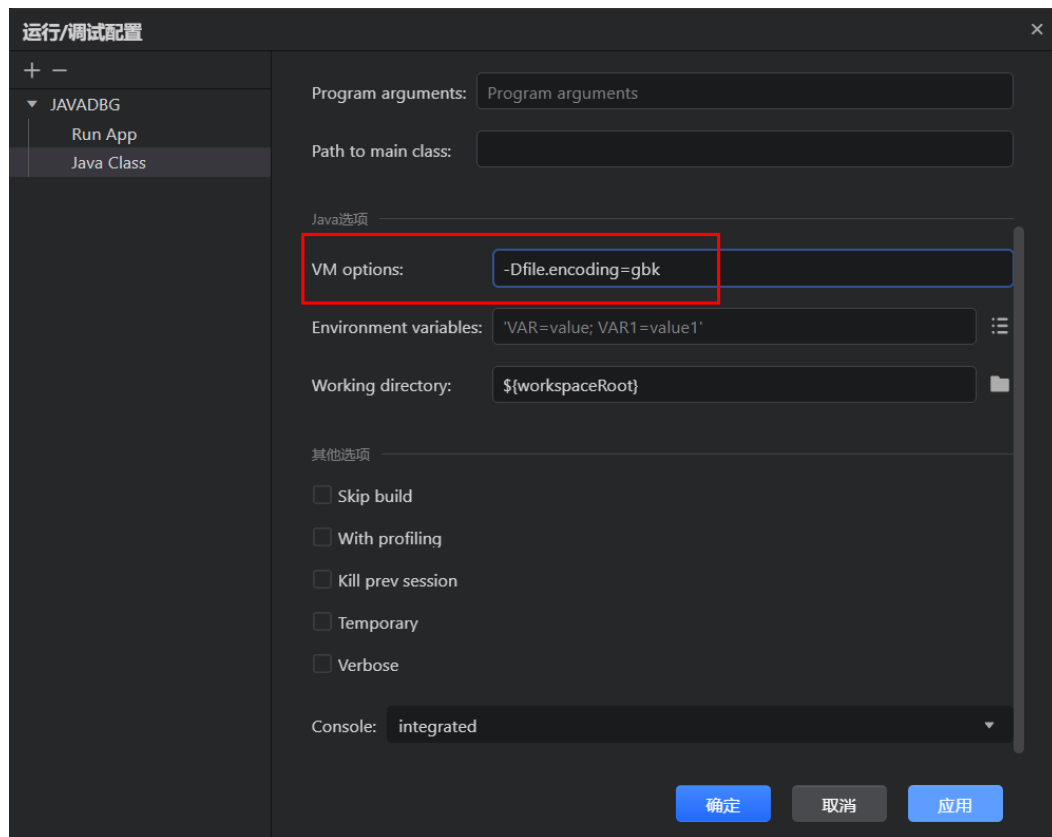
这个是默认的主题色的一个问题，在修复中，可以先尝试切换其他主题色：单击右下角“管理”->“颜色主题”即可切换不同主题：



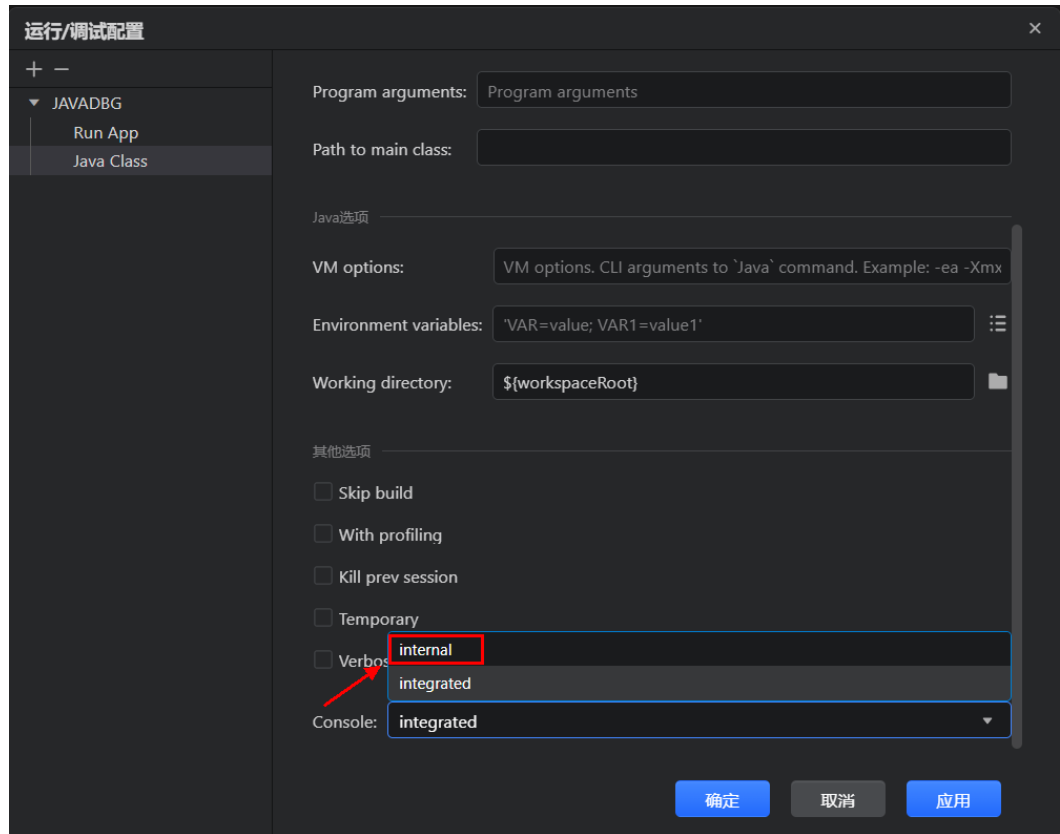
4.4 终端输出中文乱码问题

针对终端输出乱码问题，解决方式有如下两种：

一种方法是在右上角“运行/调试配置”入口中修改对应的运行/调试配置，在启动参数中的VM option参数加入相关值: `-Dfile.encoding=gbk`;



但是这种方法不能解决所有的乱码问题，考虑到实际项目中可能用到Scanner的交互式输入的场景比较少，另外一种方式就是将输出重定向到调试控制台里面：运行配置中 Console设置项中的integrated改成internal值。



4.5 默认的 Maven 配置文件 settings.xml 配置参考

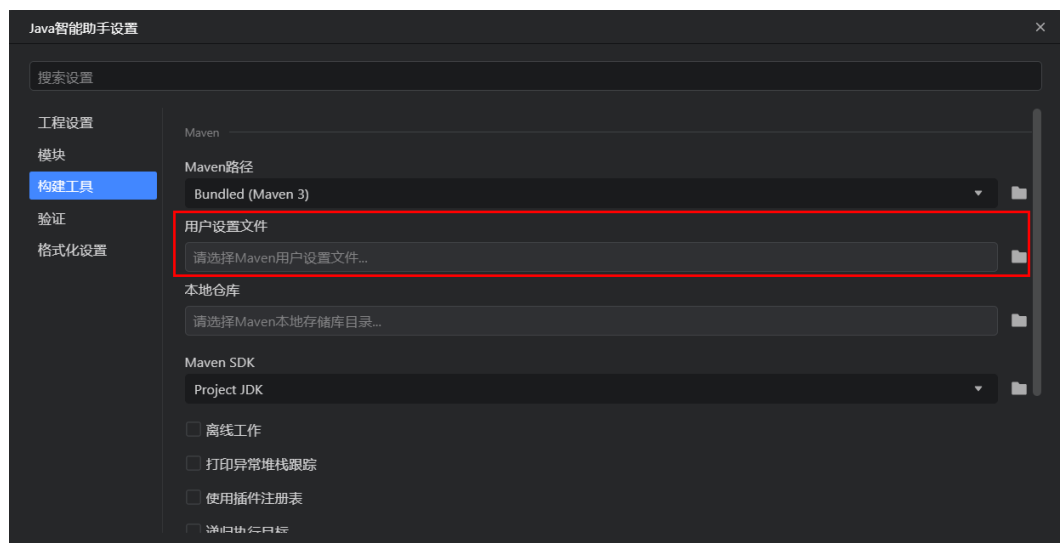
请参考此章节说明中的settings.xml文件配置参考中的配置，若要使用参考配置，可以在“用户路径/.m2”文件夹下，新建或替换settings.xml文件，Java语言服务会默认读取此路径下的maven配置文件。

也可以自定义此settings.xml文件存放路径，然后在工程中指定具体的settings.xml文件的路径，参考如下步骤：

步骤1 单击左侧活动栏下方“管理->Java智能助手设置”：



步骤2 在Java智能助手设置中，单击“构建工具”，找到Maven相关设置，替换用户设置文件即可。



----结束

settings.xml文件配置参考：

```
<?xml version="1.0" encoding="UTF-8"?>
<settings xmlns="http://maven.apache.org/SETTINGS/1.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.0.0 http://maven.apache.org/xsd/
settings-1.0.0.xsd">
  <!-- 默认的值是${user.home}/.m2/repository -->
  <!--<localRepository></localRepository>-->
  <!-- 如果Maven要试图与用户交互来得到输入就设置为true，否则就设置为false，默认为true。 -->
  <!--
  <interactiveMode>true</interactiveMode>
  -->
```

```
<!-- 如果Maven使用${user.home}/.m2/plugin-registry.xml来管理plugin的版本，就设置为true，默认为false。 -->
<!--
<usePluginRegistry>>false</usePluginRegistry>
-->
<!-- 如果构建系统要在离线模式下工作，设置为true，默认为false。如果构建服务器因为网络故障或者安全问题不能与远程仓库相连，那么这个设置是非常有用的。 -->
<!--
<offline>>false</offline>
-->
<servers>
  <!-- server
  | Specifies the authentication information to use when connecting to a particular server,
  | identified by
  | a unique name within the system (referred to by the 'id' attribute below).
  | NOTE: You should either specify username/password OR privateKey/passphrase, since these pairings
  | are
  | used together.
  |
  -->
  <!-- server标签的作用,如下 -->
  <!-- 使用mvn install时，会把项目打的包安装到本地maven仓库 -->
  <!-- 使用mvn deploy时，会把项目打的包部署到远程maven仓库，这样有权限访问远程仓库的人都可以访问你的jar包 -->
  <!-- 通过在pom.xml中使用 distributionManagement 标签，来告知maven 部署的远程仓库地址， -->
</servers>
<mirrors>
  <mirror>
    <id>huaweiyun</id>
    <mirrorOf>*</mirrorOf><!--*代表所有的jar包都到华为云下载-->
    <!--<mirrorOf>central</mirrorOf>--><!--central代表只有中央仓库的jar包才到华为云下载-->
    <!-- maven 会有默认的为“central”的中央仓库-->
    <name>huaweiyun-maven</name>
    <url>https://mirrors.huaweicloud.com/repository/maven/</url>
  </mirror>
</mirrors>
<!-- settings.xml中的profile是pom.xml中的profile的简洁形式。
它包含了激活(activation)，仓库(repositories)，插件仓库(pluginRepositories)和属性(properties)元素。
profile元素仅包含这四个元素是因为他们涉及到整个的构建系统，而不是个别的POM配置。
如果settings中的profile被激活，那么它的值将重载POM或者profiles.xml中的任何相等ID的profiles。 -->
<!-- 如果setting中配置了 repository，则等于项目的pom中配置了 -->
<profiles>
  <profile>
    <!-- 指定该 profile的id -->
    <id>dev</id>
    <!-- 远程仓库-->
    <repositories>
      <!-- 华为云远程仓库-->
      <repository>
        <id>huaweicloud</id>
        <name>huaweicloud maven Repository</name>
        <url>https://mirrors.huaweicloud.com/repository/maven/</url>
        <!-- 只从该仓库下载 release版本 -->
        <releases>
          <enabled>>true</enabled>
        </releases>
        <snapshots>
          <enabled>>false</enabled>
        </snapshots>
      </repository>
      <repository>
        <id>spring-milestone</id>
        <name>Spring Milestone Repository</name>
        <url>https://repo.spring.io/milestone</url>
        <releases>
          <enabled>>true</enabled>
        </releases>
        <snapshots>
```

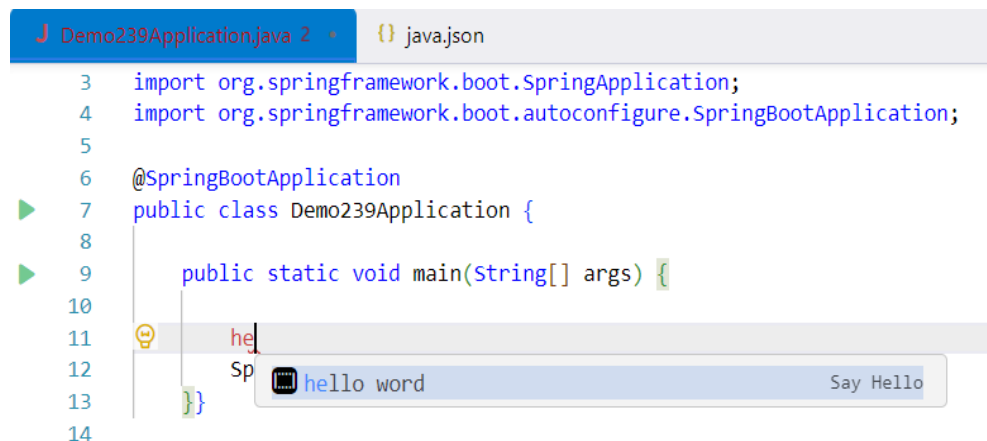
```
<enabled>false</enabled>
</snapshots>
<layout>default</layout>
</repository>
<repository>
  <id>spring-snapshot</id>
  <name>Spring Snapshot Repository</name>
  <url>https://repo.spring.io/snapshot</url>
  <releases>
    <enabled>false</enabled>
  </releases>
  <snapshots>
    <enabled>true</enabled>
  </snapshots>
  <layout>default</layout>
</repository>
</repositories>
<pluginRepositories>
  <!-- 插件仓库。插件从这些仓库下载 -->
  <pluginRepository>
    <id>huaweicloud</id>
    <url>https://mirrors.huaweicloud.com/repository/maven/</url>
    <releases>
      <enabled>true</enabled>
    </releases>
    <snapshots>
      <enabled>false</enabled>
    </snapshots>
  </pluginRepository>
</pluginRepositories>
</profile>
</profiles>
<!-- activations是profile的关键，就像POM中的profiles，profile的能力在于它在特定情况下可以修改一些值。
而这些情况是通过activation来指定的。 -->
<!-- <activeProfiles/> -->
<activeProfiles>
  <activeProfile>dev</activeProfile>
</activeProfiles>
</settings>
```

而这种情况是通过activation来指定的。 -->

4.6 怎么设置代码片段和代码模板

1、设置自定义代码片段：CodeArts IDE支持自定义代码片段的能力，可以通过代码补全的方式插入自定义的代码片段。

效果如下：



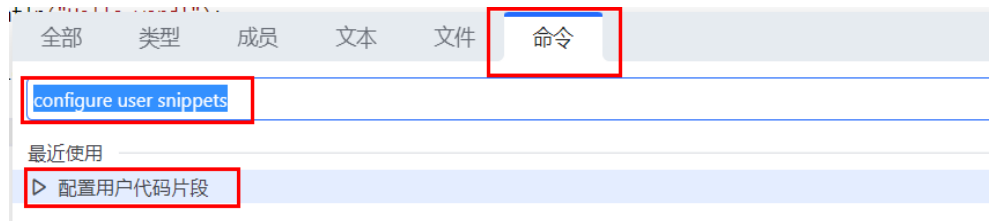
The screenshot shows the CodeArts IDE interface with a Java file named 'Demo239Application.java'. The code is as follows:

```
3 import org.springframework.boot.SpringApplication;
4 import org.springframework.boot.autoconfigure.SpringBootApplication;
5
6 @SpringBootApplication
7 public class Demo239Application {
8
9     public static void main(String[] args) {
10
11         he
12         Sp
13     }
14
```

A code completion popup is visible, showing the text 'hello word' and 'Say Hello'.

```
Demo239Application.java x {} java.json
3 import org.springframework.boot.SpringApplication;
4 import org.springframework.boot.autoconfigure.SpringBootApplication;
5
6 @SpringBootApplication
7 public class Demo239Application {
8
9     public static void main(String[] args) {
10
11         System.out.println("Hello word!");
12
13         SpringApplication.run(Demo239Application.class, args);
14     }
15 }
```

相关设置如下，ctrl+shift+p打开命令面板，输入configure user snippets:



可以选择全局的，也可以根据特定语言文件设置。





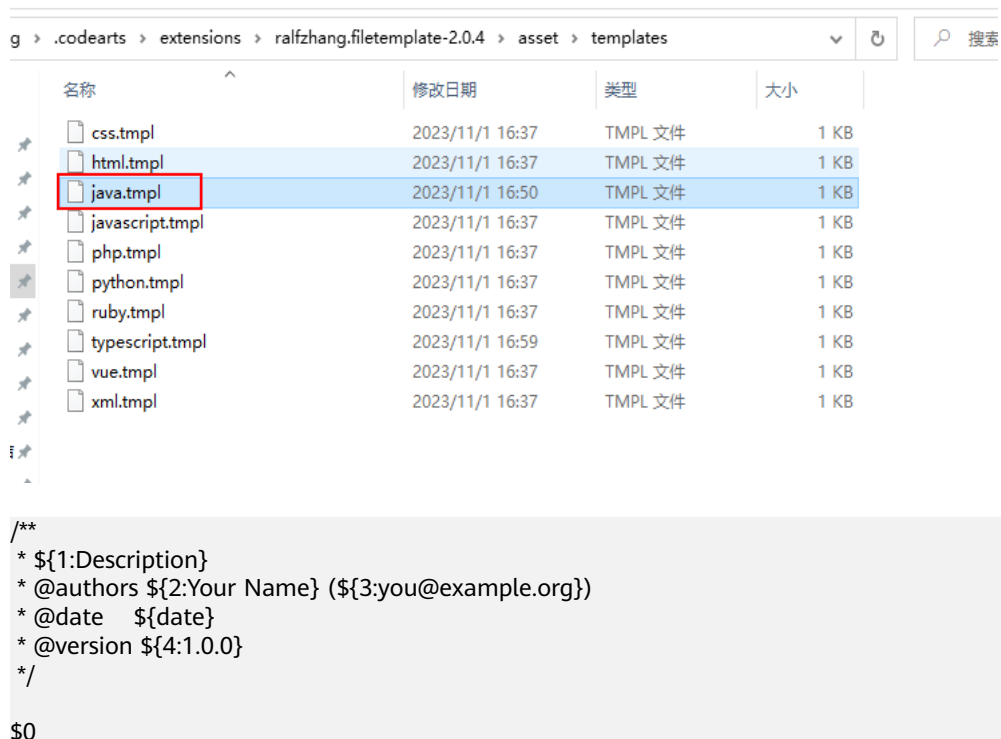
```
1 {
2   // Place your snippets for java here. Each snippet is defined under a snippet name and
3   // description. The prefix is what is used to trigger the snippet and the body will be
4   // $1, $2 for tab stops, $0 for the final cursor position, and ${1:label}, ${2:another}
5   // same ids are connected.
6   // Example:
7   "Say Hello": {
8     "prefix": "hello word",
9     "body": [
10      "System.out.println(\"Hello world!\");",
11      "$2"
12    ],
13    "description": "Print hello world to output"
14  }
15 }
```

2、设置代码模板，支持动态设置日期

(1) 下载最新的File Template插件，然后安装到CodeArts IDE中，附件中提供了此插件，名称是RalfZhang.filetemplate-2.0.4.rar，可直接下载（需解压）。

（插件市场此插件链接：<https://marketplace.visualstudio.com/items?itemName=RalfZhang.filetemplate>）

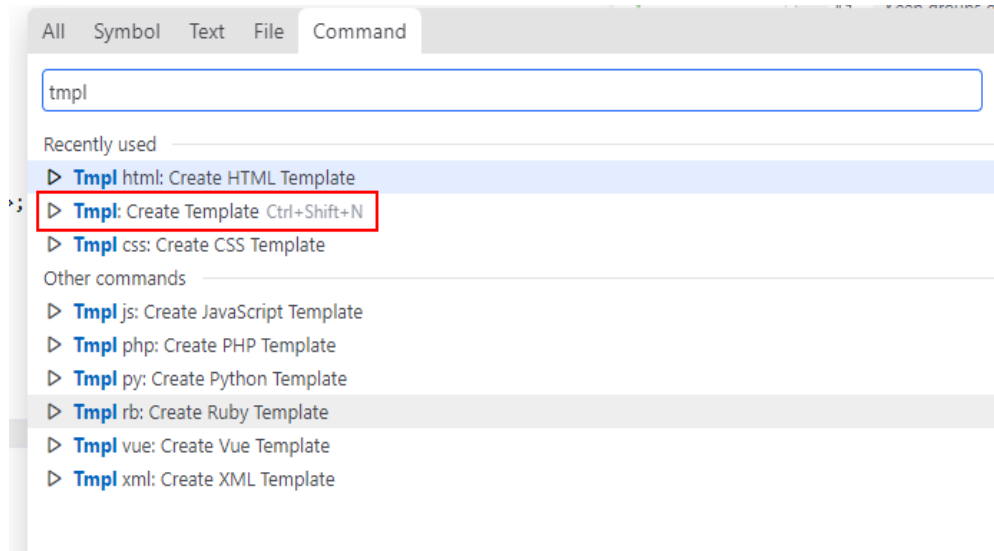
(2) 具体使用方法是，在插件安装路径：“用户路径\codearts\extensions\ralfzhang.filetemplate-2.0.4\asset\templates”新增一个java.templ的文件（默认没有此文件），里面可以自定义Java语言相关的代码片段。



名称	修改日期	类型	大小
css.templ	2023/11/1 16:37	TMPL 文件	1 KB
html.templ	2023/11/1 16:37	TMPL 文件	1 KB
java.templ	2023/11/1 16:50	TMPL 文件	1 KB
javascript.templ	2023/11/1 16:37	TMPL 文件	1 KB
php.templ	2023/11/1 16:37	TMPL 文件	1 KB
python.templ	2023/11/1 16:37	TMPL 文件	1 KB
ruby.templ	2023/11/1 16:37	TMPL 文件	1 KB
typescript.templ	2023/11/1 16:59	TMPL 文件	1 KB
vue.templ	2023/11/1 16:37	TMPL 文件	1 KB
xml.templ	2023/11/1 16:37	TMPL 文件	1 KB

```
/**
 * ${1:Description}
 * @authors ${2:Your Name} (${3:you@example.org})
 * @date   ${date}
 * @version ${4:1.0.0}
 */
$0
```

(3) 然后使用关键字“templ”搜索相关模板命令，使用“Tmpl: Create Template”命令就能根据该文件所对应的语言生成对应的代码片段（注意：没有Java相关的templ命令，只能在打开的Java文件中使用“Tmpl: Create Template”命令）：

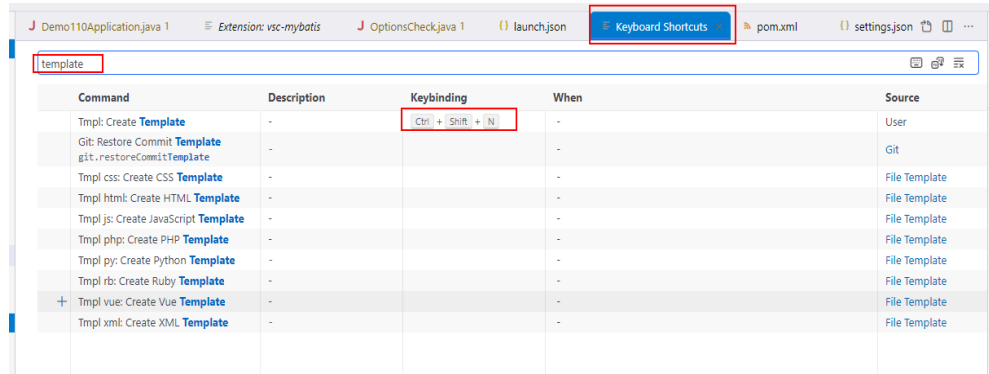


效果如下:



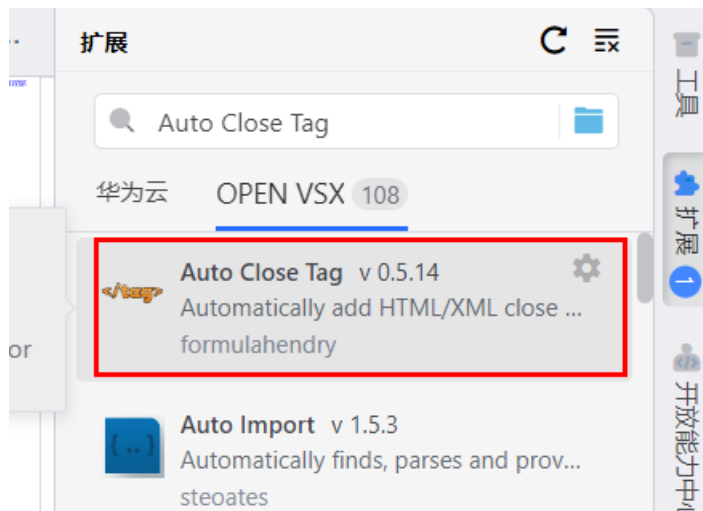
(4) 默认的Create Template命令是能够根据当前打开的文件所对应的语言（如Java），来插入对应语言的模板文件中定义的代码片段。

(5) 也可以给这些命令绑定如下类似的快捷键：

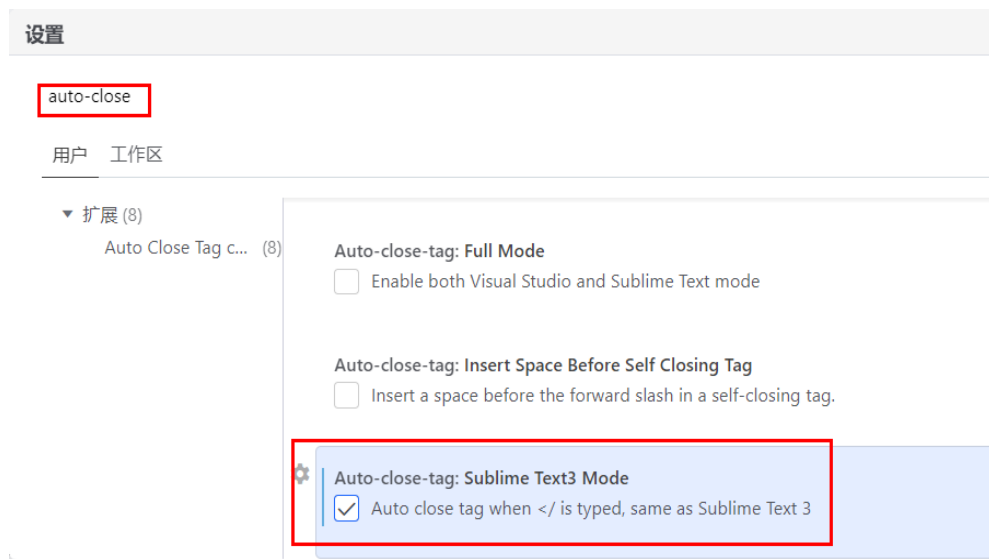


4.7 xml 文件中的标签，写入结束标签</的时候，需要自动补全标签对

可以安装如下“Auto Close Tag”插件：



然后启用下面的配置项，就可以在输入</ 的时候自动补全标签对：



4.8 识别并展示 TODO 列表

可以安装如下“Todo Tree”插件，IDE左侧会生成“待办事项”视图，自动识别工程中的TODO标签。

