

# Knowledge Querying (I)

# 一、RDF4J配置

示例代码：<https://github.com/eclipse/rdf4j/tree/main/examples>

# 注意须知：

1. 考虑到同学们电脑上有**JAVA 8**版本，无需删除，跟着环境配置教程进行即可。
2. 若电脑上已有**java 11**版本的同学，可跳过“安装JDK”与“环境变量配置步骤”。
3. 本教程提供了**Windows 64**位操作系统和**Mac**操作系统的相关安装包，采用其它操作系统的同学请自行下载安装包，具体网址为：

Java JDK 11.0.13: <https://www.oracle.com/java/technologies/downloads/#java11>

Eclipse 2021-12 : <https://www.eclipse.org/downloads/packages/>

# 环境配置—安装JDK【涉及到的安装包均已提供】

1. 双击安装包（Windows系统选择.exe后缀，Mac系统选择.tar.gz后缀，本示例将以Windows系统为例），点击“下一步”开始安装。



# 环境配置—安装JDK

2. 选择安装路径（请记录该安装路径，后续会用到），本示例采用的路径为 G:\java\JDK\jdk-11.0.13\
3. 点击下一步 继续安装



# 环境配置—安装JDK

4. 等待JDK安装，点击“关闭”完成安装。

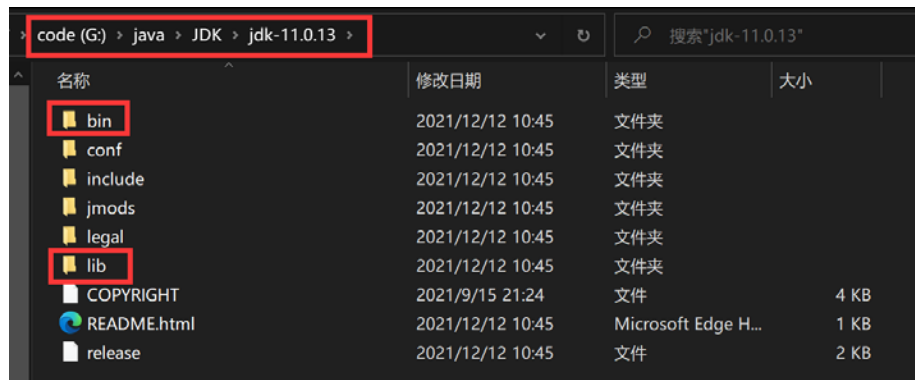


# 环境配置—环境变量配置

1. 找到对应路径。

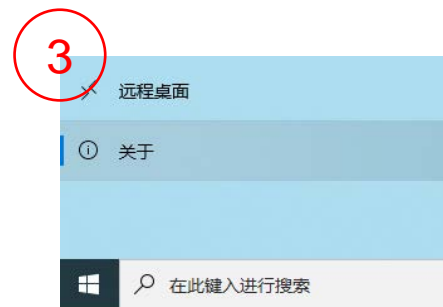
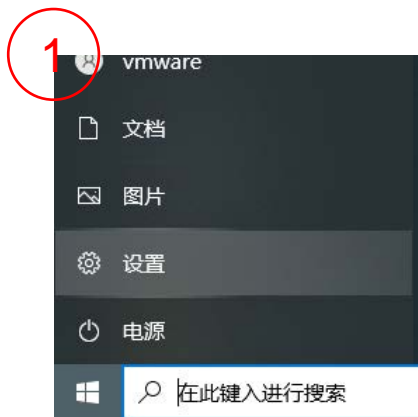
记录3个路径：

- 1) 安装JDK时的安装路径，对应后续步骤中的 `JAVA_HOME`  
– 本示例为 `G:\java\JDK\jdk-11.0.13\`
- 2) JDK安装路径下的lib文件，对应后续步骤中的 `CLASSPATH`  
– 本示例为 `G:\java\JDK\jdk-11.0.13\lib`
- 3) JDK安装路径下的bin文件，对应后续步骤中的 `Path`  
– 本示例为 `G:\java\JDK\jdk-11.0.13\bin`



# 环境配置—环境变量配置

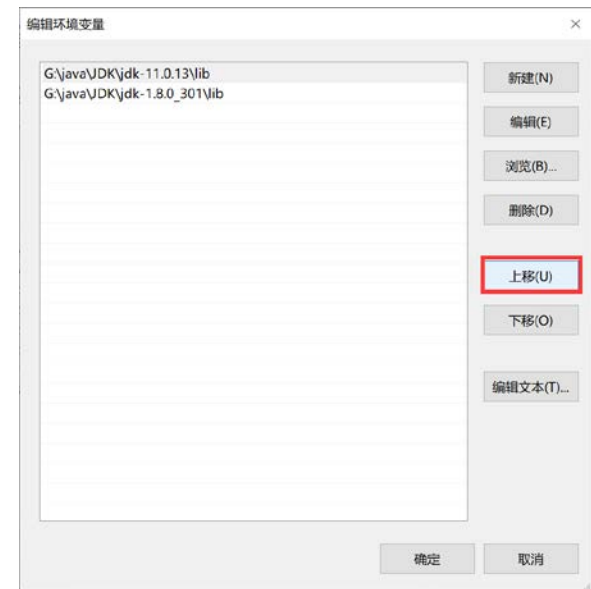
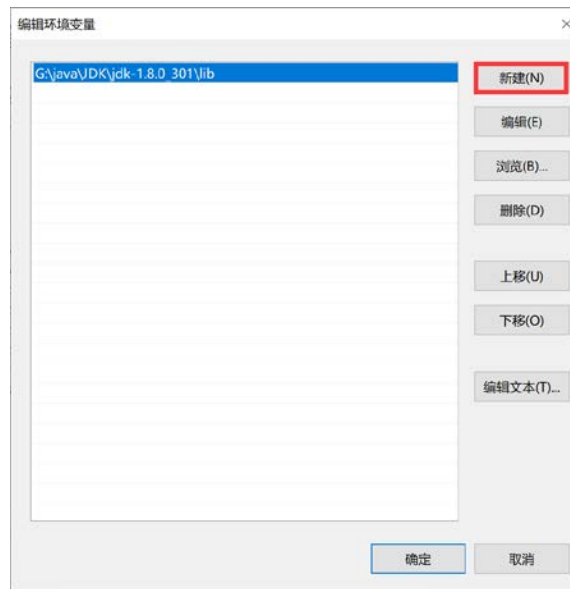
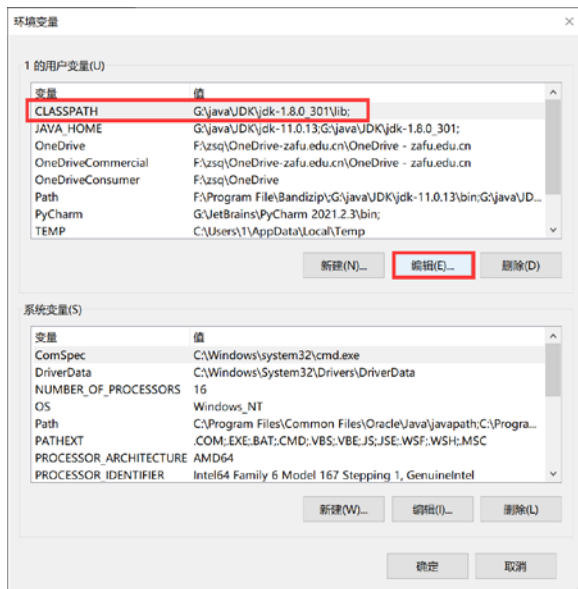
2. 打开 设置 --> 系统 --> 关于 --> 高级系统设置 --> 环境变量





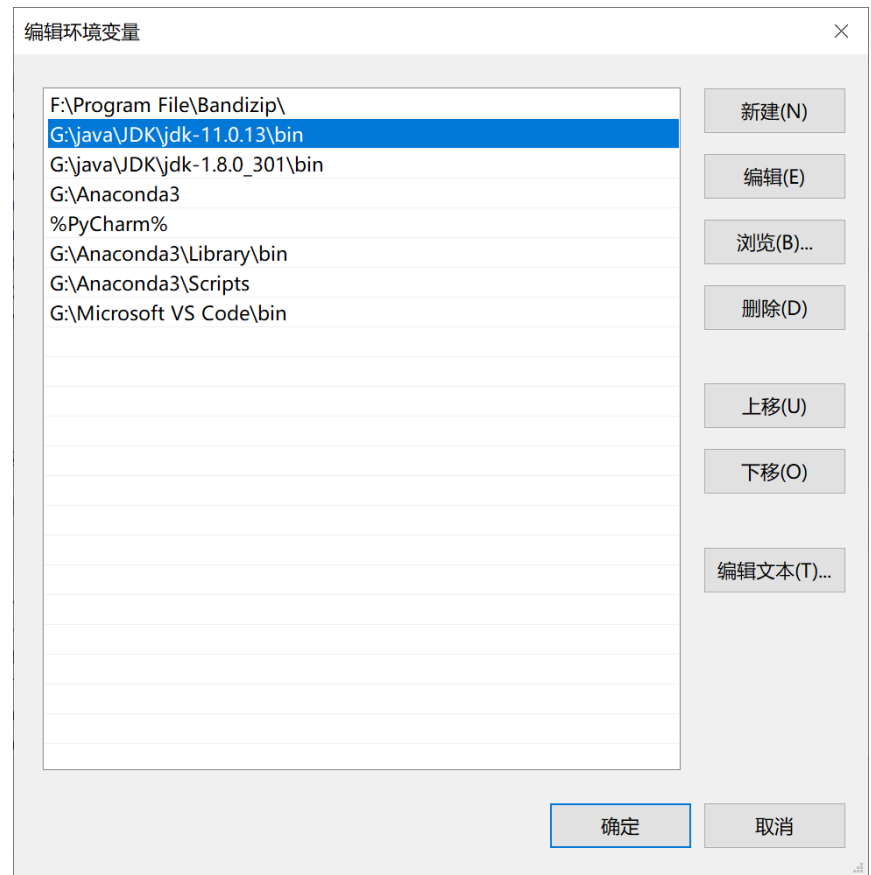
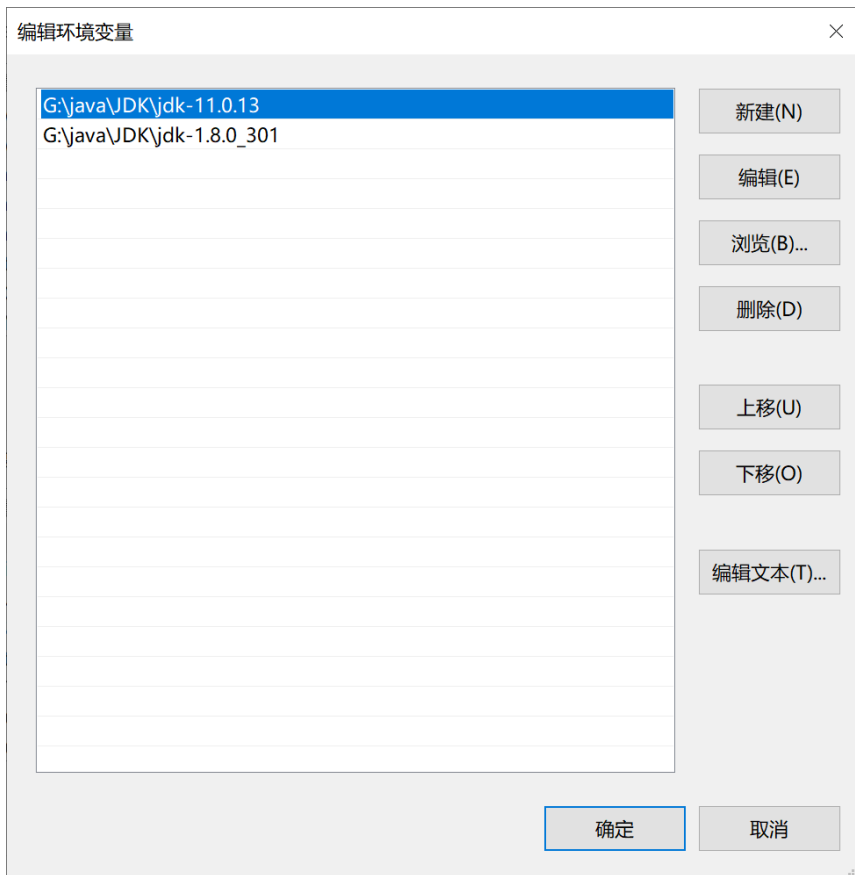
# 环境配置—环境变量配置

3. 编辑 “CLASSPATH” 环境变量，选中 “CLASSPATH”，点击“编辑”，在弹出的窗口选择“新建”，然后输入步骤1中的记录值点“上移”，将新建的路径调整到最顶端。



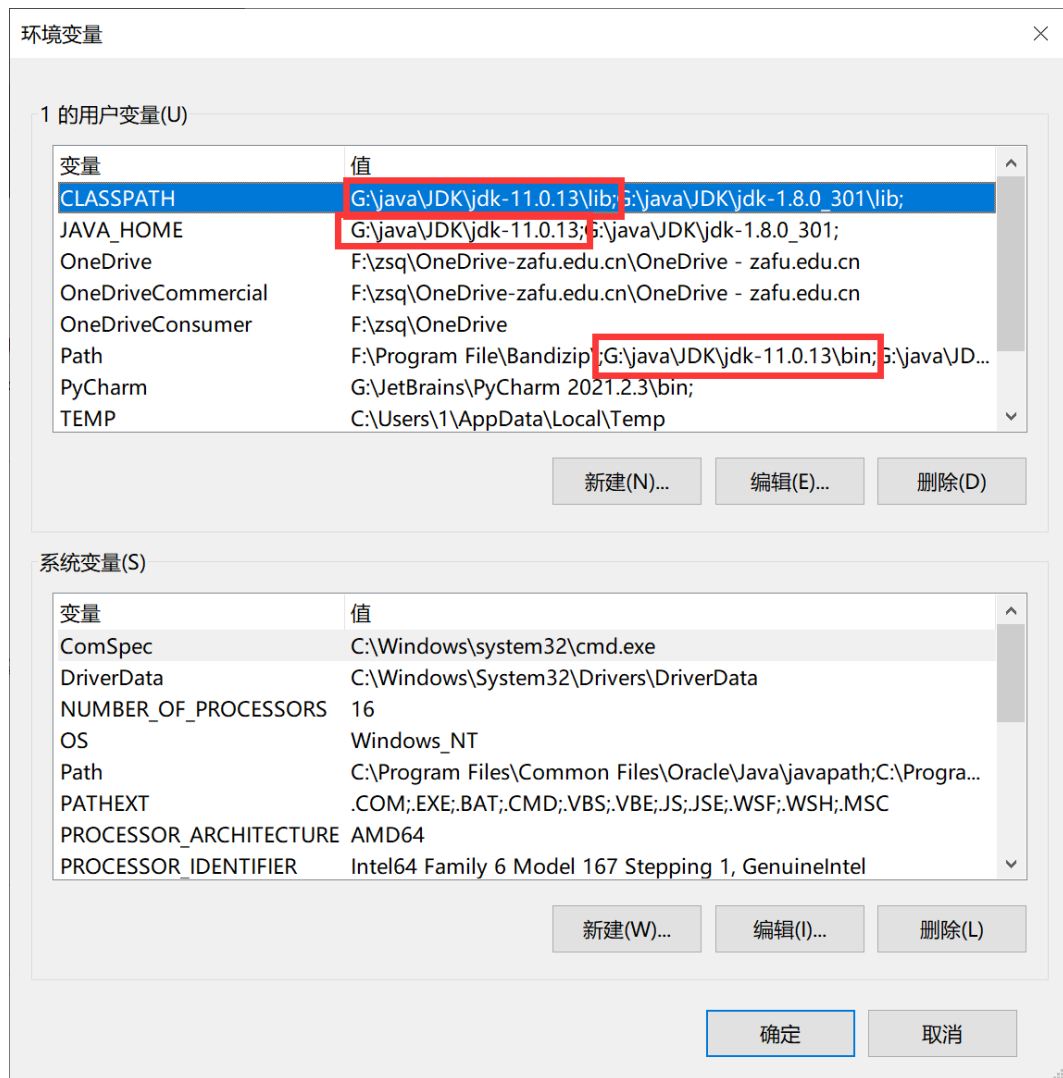
# 环境配置—环境变量配置

4. 依次编辑 “JAVA\_HOME” 和 “Path” 环境变量。



# 环境配置—环境变量配置

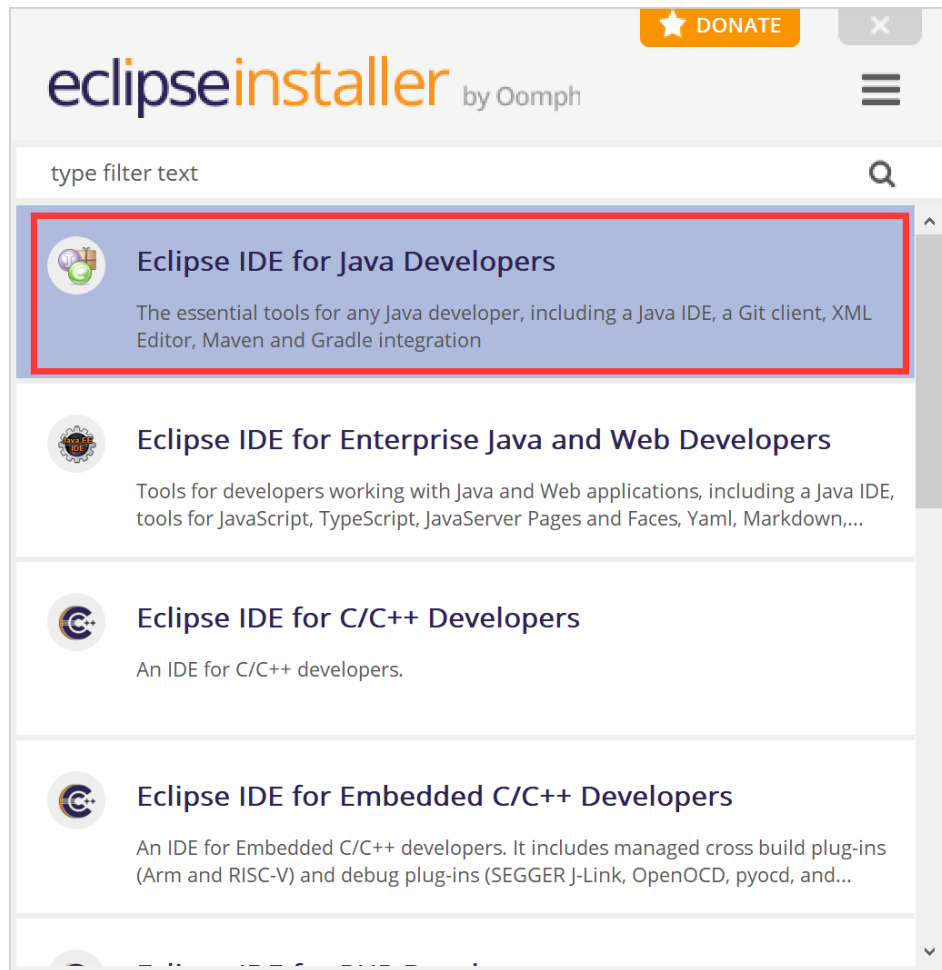
5. 最终完成界面如右图所示，点击“确定”



# 环境配置—安装IDE 【涉及到的安装包均已提供】

使用的IDE: Eclipse

1. 双击安装包（Windows系统选择.exe后缀，Mac系统选择.dmg后缀，本示例将以Windows系统为例），选择“Eclipse IDE for Java Developers”开始安装。

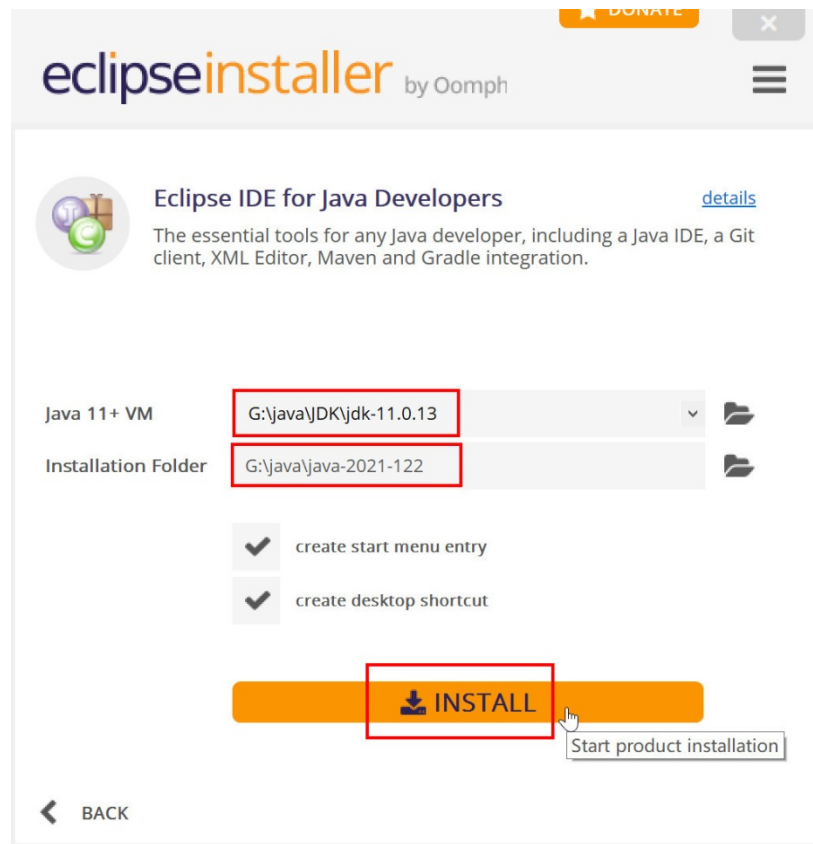


# 环境配置—安装IDE

2. 设置相关路径。

Java 11+ VM 处选择 JDK安装路径，  
Installation Folder 为eclipse安装路  
径，可自由选择，本示例安装至  
G:\java\eclipseNew 路径下

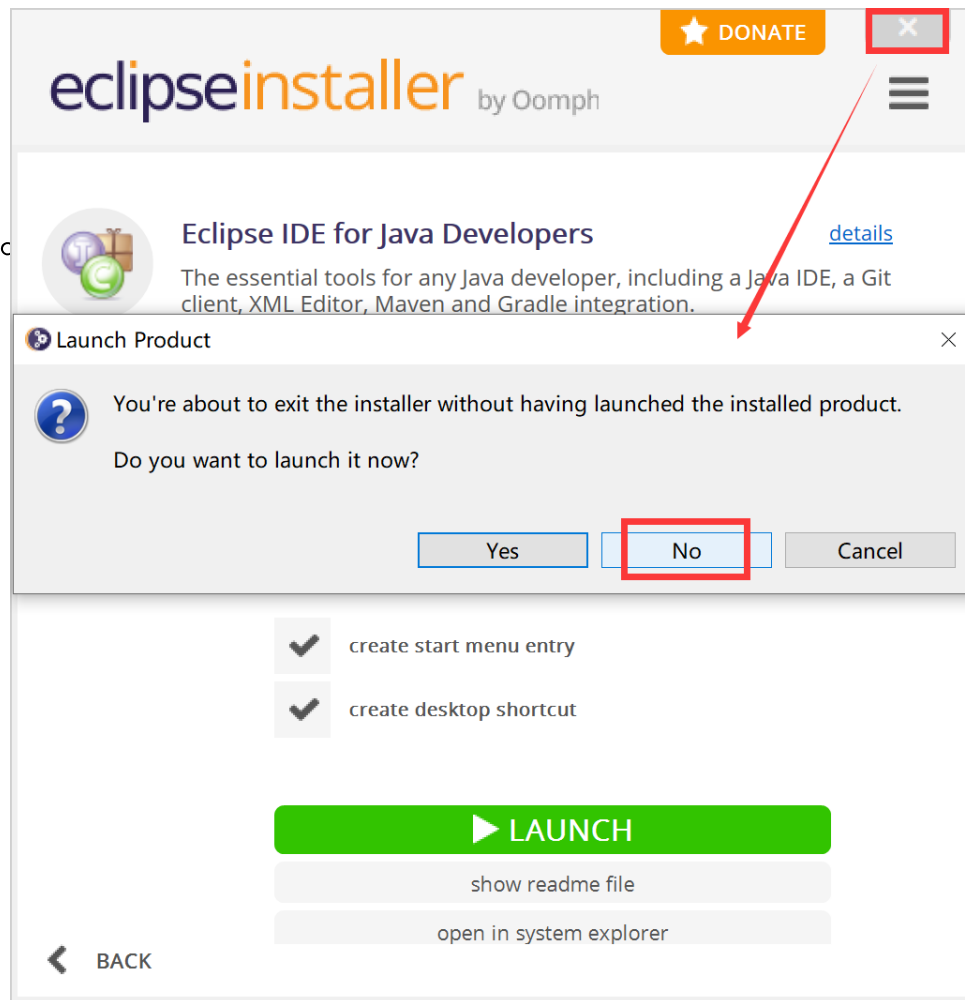
3. 点击“INSTALL”开始安装。



# 环境配置—安装IDE

4. 等待Eclipse安装，完成后不要直接运行程序。

点击关闭按钮，弹框选择“**No**”



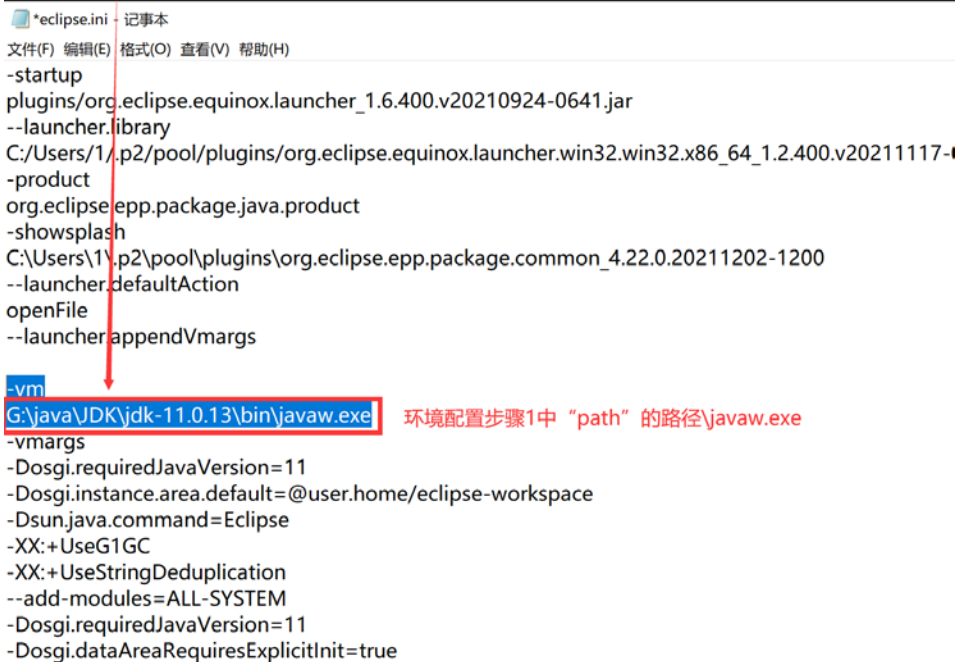
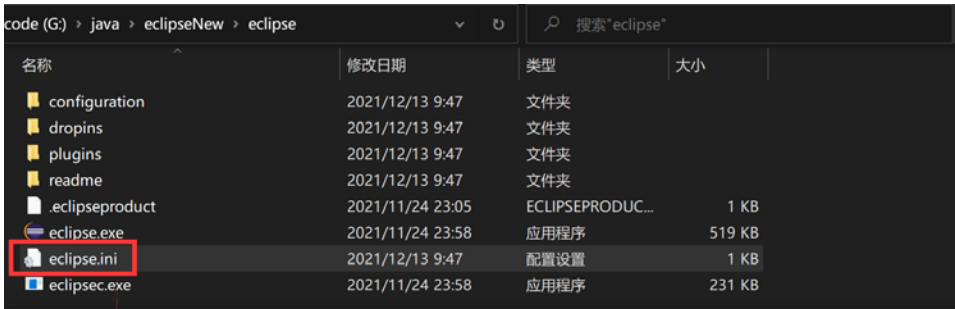
# 环境配置—安装IDE

5. 找到eclipse的安装位置，该目录下有一个“eclipse.ini”文件，用记事本打开，在“-vmargs”前添加两行代码，保存更改。

-vm  
%path%\javaw.exe

本示例为：

-vm  
G:\java\JDK\jdk-11.0.13\bin\javaw.exe



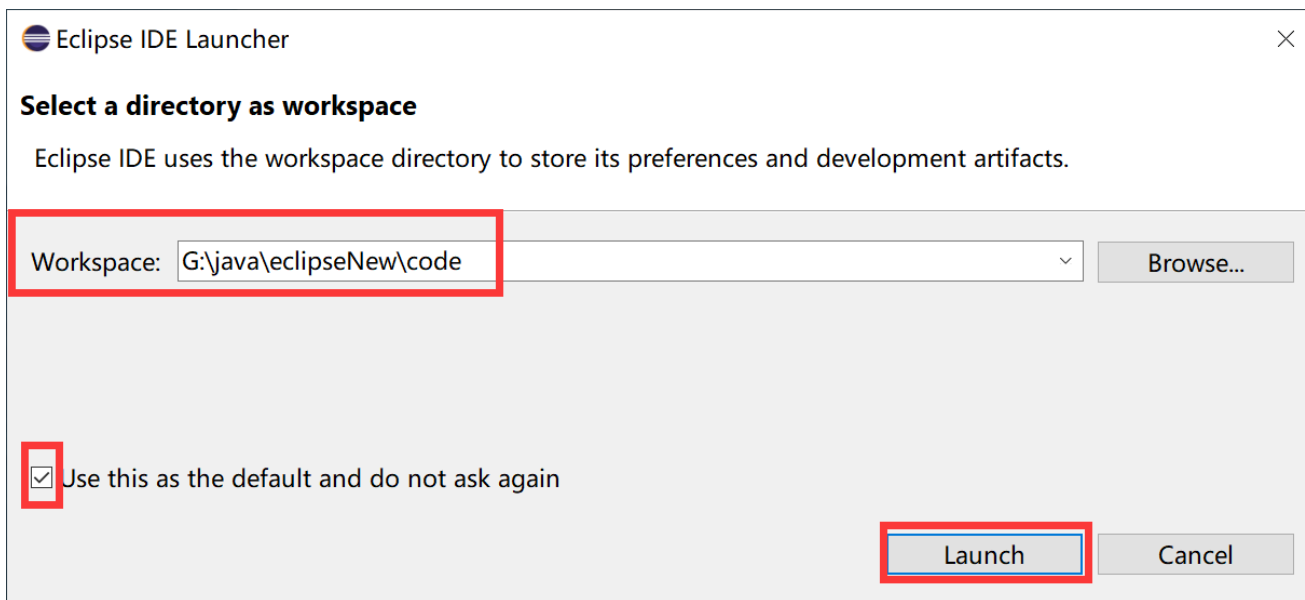
环境配置步骤1中“path”的路径javaw.exe

# 环境配置—Eclipse相关配置

## 1. 启动eclipse，配置workspace。

双击桌面快捷方式启动Eclipse。第一次启动时，需要提供工作空间目录（用于存放Eclipse存储所有项目数据和一些配置信息），可以自由选择/创建目录，或者直接接受默认目录。勾选“Use this as the default and do not ask again”，点击“Launch”

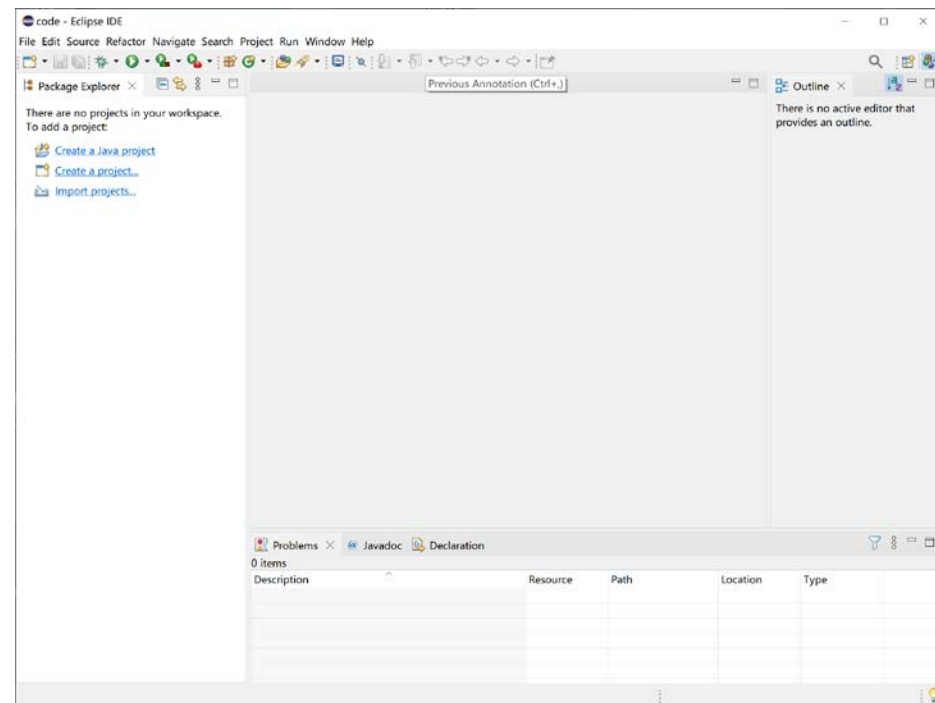
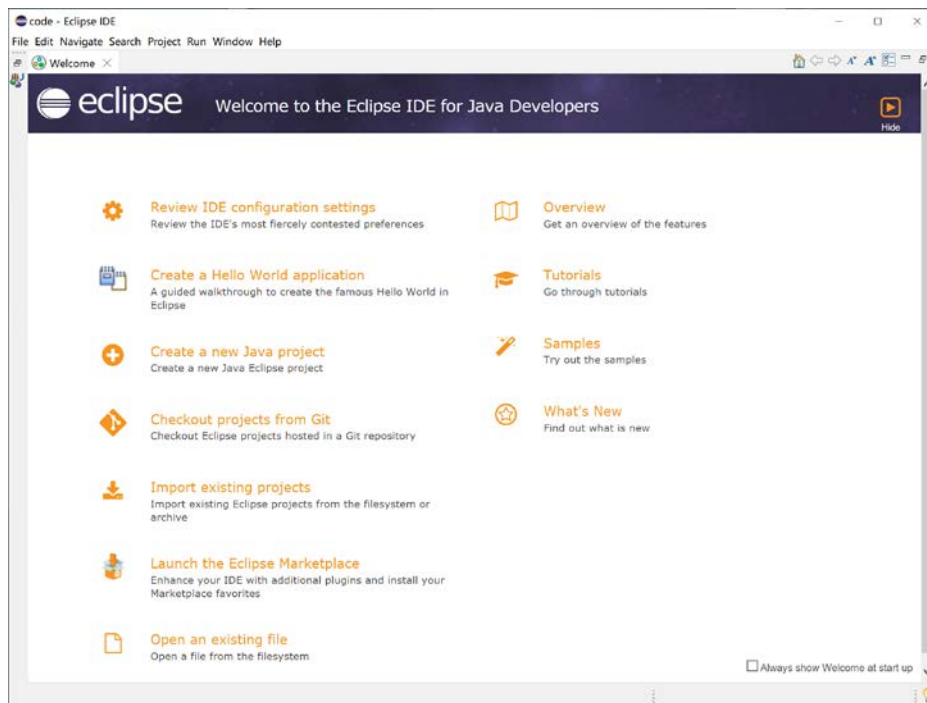
本示例将工作空间目录设置为 G:\java\eclipseNew\code





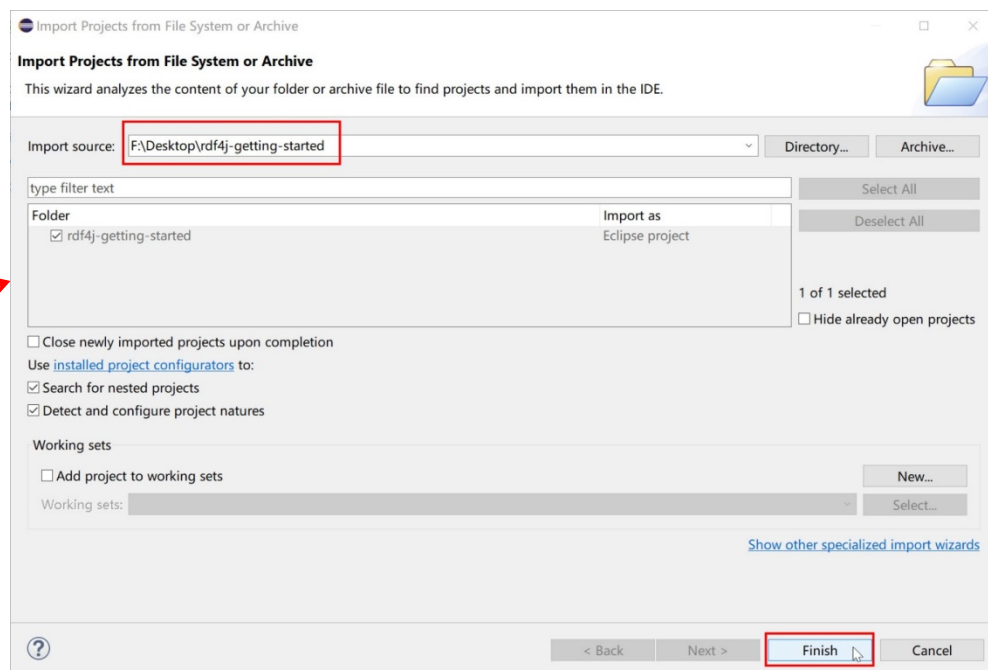
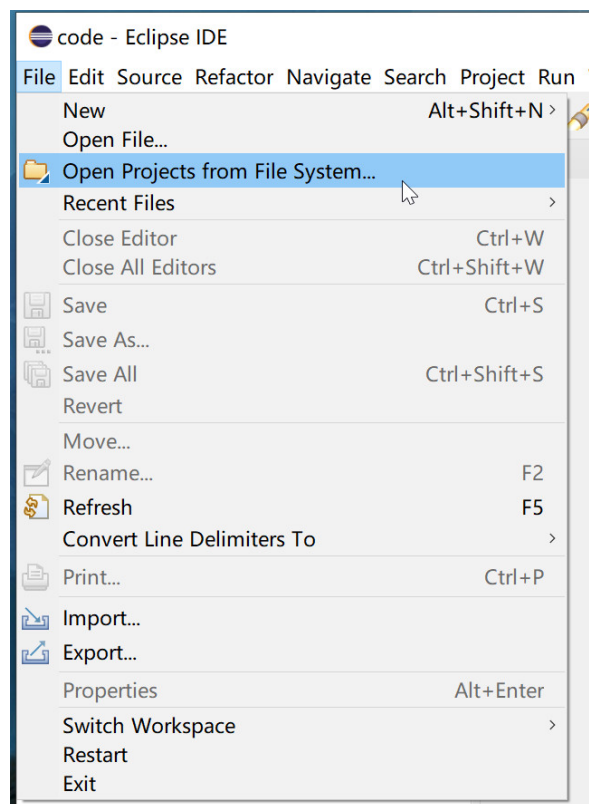
# 环境配置—Eclipse相关配置

2. 第一次启动Eclipse会出现welcome页面，如左下图所示，关闭后如右下图所示。



# 环境配置—打开项目文件

1. 在顶部菜单栏中选择File→Open Projects from File System...
2. 选择下发的项目文件路径，点击“Finish”。



# 环境配置—打开项目文件

## 3. 直接替换本地Maven仓库。

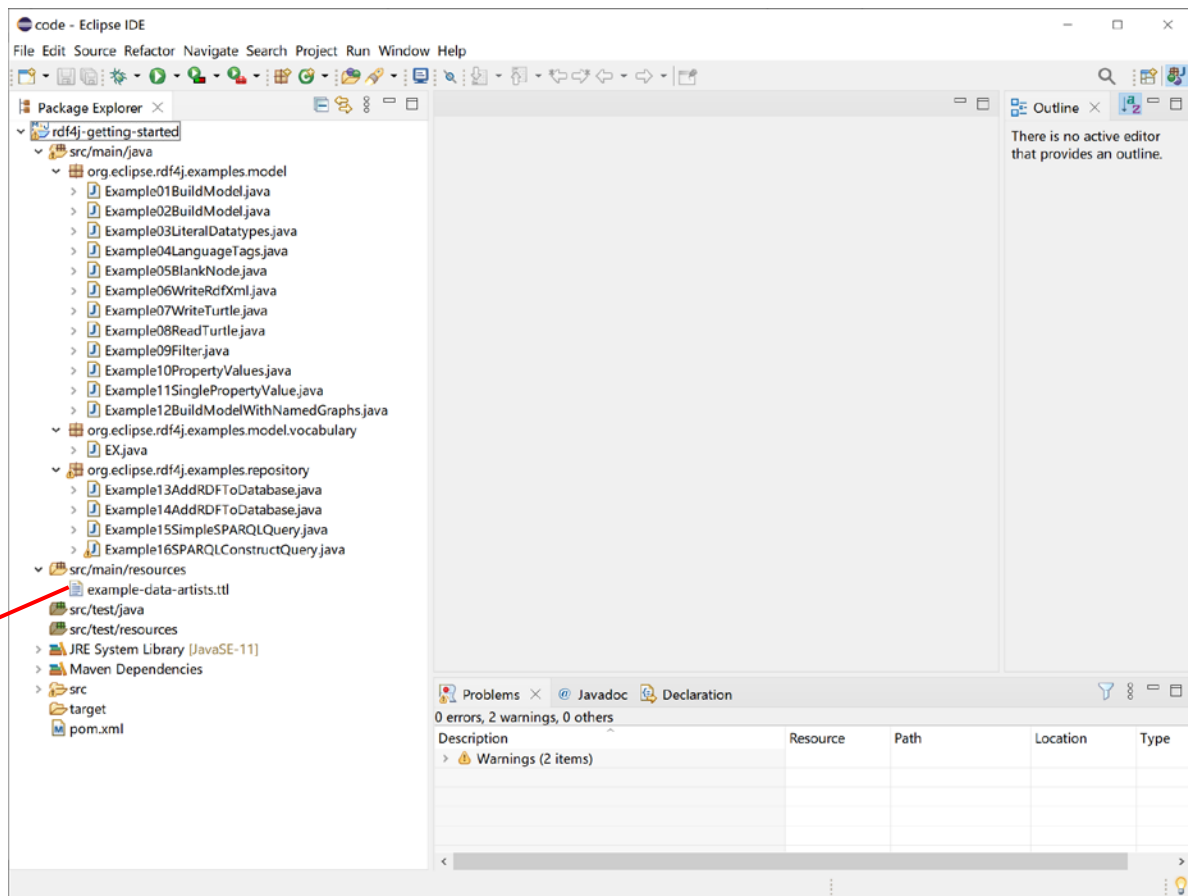
Maven项目将自动下载项目依赖的jar包，这个过程较慢。可将下发的“**repository.zip**”解压后替换本地Maven仓库中的**repository**。本地Maven仓库默认位于**C:\Users\用户名\.m2** 文件夹下。




# 环境配置—打开项目文件

4. 成功打开的项目  
界面如右图所示。

example-data-artists.ttl  
用于存放turtle数据



# .TTL文件内容

 example-data-artists.ttl - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
@prefix ex: <http://example.org/> .
```

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
```

```
ex:Picasso a ex:Artist ;
```

```
    foaf:firstName "Pablo" ;
```

```
    foaf:surname "Picasso";
```

```
    ex:creatorOf ex:guernica ;
```

```
    ex:homeAddress _:node1 .
```

```
    _:node1 ex:street "31 Art Gallery" ;
```

```
        ex:city "Madrid" ;
```

```
        ex:country "Spain" .
```

```
ex:guernica a ex:Painting ;
```

```
    rdfs:label "Guernica";
```

```
    ex:technique "oil on canvas".
```

```
ex:VanGogh a ex:Artist ;
```

```
    foaf:firstName "Vincent" ;
```

```
    foaf:surname "van Gogh";
```

```
    ex:creatorOf ex:starryNight, ex:sunflowers, ex:potatoEaters .
```

```
ex:starryNight a ex:Painting ;
```

```
    ex:technique "oil on canvas";
```

```
    rdfs:label "Starry Night" .
```

```
ex:sunflowers a ex:Painting ;
```

```
    ex:technique "oil on canvas";
```

```
    rdfs:label "Sunflowers" .
```

```
ex:potatoEaters a ex:Painting ;
```

```
    ex:technique "oil on canvas";
```

```
    rdfs:label "The Potato Eaters" .
```

# SPARQL SELECT查询

SPARQL查询语句:

PREFIX ex: <http://example.org/>

PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT ?s ?n

WHERE {

    ?s a ex:Artist;

    foaf:firstName ?n .

}

查询有名字的艺术家的名字

# SPARQL SELECT查询

```
20 * Copyright (c) 2016, 2017 Eclipse RDF4J contributors.
8 package org.eclipse.rdf4j.examples.repository;
9
10 import java.io.IOException;
22
24 * RDF Tutorial example 15: executing a simple SPARQL query on the database
28 public class Example15SimpleSPARQLQuery {
29
30     public static void main(String[] args)
31         throws IOException {
32         // Create a new Repository.
33         Repository db = new SailRepository(new MemoryStore());
34
35         // Open a connection to the database
36         try (RepositoryConnection conn = db.getConnection()) {
37             String filename = "example-data-artists.ttl";
38             try (InputStream input = Example15SimpleSPARQLQuery.class.getResourceAsStream("/" + filename)) {
39                 // add the RDF data from the inputstream directly to our database
40                 conn.add(input, "", RDFFormat.TURTLE);
41             }
42         }
```

line 33 首先建立一个内存仓库db存取数据

line 36 建立一个仓库链接conn连接到内存仓库db

line 38 打开数据文件作为输入流input

line 40 将输入流input通过数据库连接conn加入到内存仓库db中

# SPARQL SELECT查询

```
42
43 // We do a simple SPARQL SELECT-query that retrieves all resources of type `ex:Artist`,
44 // and their first names.
45 String queryString = "PREFIX ex: <http://example.org/> \n";
46 queryString += "PREFIX foaf: <" + FOAF.NAMESPACE + "> \n";
47 queryString += "SELECT ?s ?n \n";
48 queryString += "WHERE { \n";
49 queryString += "    ?s a ex:Artist; \n";
50 queryString += "        foaf:firstName ?n .";
51 queryString += "}";
52
53 TupleQuery query = conn.prepareTupleQuery(queryString);
54
```

line 45 - 51

将SPARQL语句存入java字符串queryString中

line 53

在conn中用SPARQL查询，并将三元组返回到query中



# SPARQL SELECT查询

```
55         // A QueryResult is also an AutoCloseable resource, so make sure it gets closed when done.
56         try (TupleQueryResult result = query.evaluate()) {
57             // we just iterate over all solutions in the result...
58             for (BindingSet solution : result) {
59                 // ... and print out the value of the variable binding for ?s and ?n
60                 System.out.println("?s = " + solution.getValue("s"));
61                 System.out.println("?n = " + solution.getValue("n"));
62             }
63         }
64     } finally {
65         // Before our program exits, make sure the database is properly shut down.
66         db.shutdown();
67     }
68 }
69 }
```

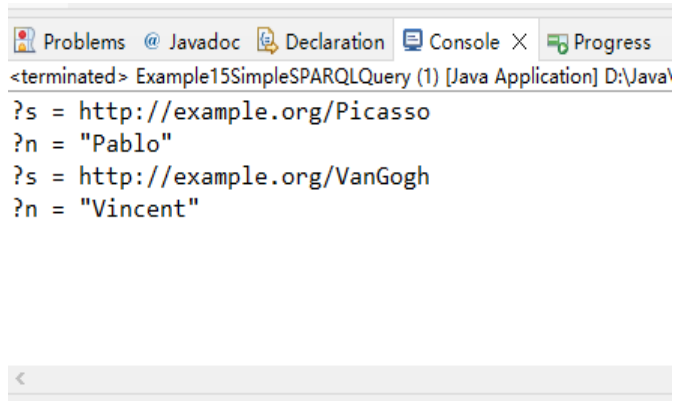
line 58-62                  输出查询到的结果

line 64-67                关闭内存仓库db

# SPARQL SELECT查询

右击 -> Run As -> Java Application 运行

查询返回结果:



The screenshot shows an IDE console window with the following tabs: Problems, Javadoc, Declaration, Console, and Progress. The Console tab is active, displaying the output of a Java application. The output starts with a terminated status and the application name, followed by two rows of query results. Each row contains a variable binding for 's' (a URL) and 'n' (a name). The first row shows 's' as 'http://example.org/Picasso' and 'n' as 'Pablo'. The second row shows 's' as 'http://example.org/VanGogh' and 'n' as 'Vincent'.

```
<terminated> Example15SimpleSPARQLQuery (1) [Java Application] D:\Java\
?s = http://example.org/Picasso
?n = "Pablo"
?s = http://example.org/VanGogh
?n = "Vincent"
```

### 三、作业

# 作业

1. 针对下列自然语言，使用SPARQL查询语句返回结果：

- (1) Who are the creators of Guernica and Sunflowers, respectively?
- (2) List all the artists who live in Spain or other places.
- (3) List all paintings, their names, and the corresponding techniques.