篇文章主要介绍了JDK的Parser来解析Java源代码的相关资料,需要的朋友可以参考下

目录

[1.1. com.github.javaparser.JavaParser; 1](#_Toc2427)

[1.2. 在JDK的tools.jar（ 1](#_Toc30352)

package com.bootdo.jparser;

## com.github.javaparser.JavaParser;

import java.io.File;

import java.io.FileNotFoundException;

import com.github.javaparser.JavaParser;

import com.github.javaparser.ast.CompilationUnit;

## 在JDK的tools.jar（

在JDK中，自带了一套相关的编译API，可以在Java中发起编译流程，解析Java源文件然后获取其语法树，在JDK的tools.jar（OSX下可以在/Library/Java/JavaVirtualMachines/jdk\_version/Contents/Home/lib中找到）中包含着这整套API，但是这却不是Oracle和OpenJDK发布中的公开API，因此对于这套API，并没有官方的正式文档来进行说明。但是，也有不少项目利用了这套API来做了不少事情，例如大名鼎鼎的lombok使用了这套API在Annotation Processing阶段修改了源代码中的语法树，最终结果相当于直接在源文件中插入了新的代码！

由于这套API目前缺少相关文档，使用起来比较困难，例如，解析源代码中的所有变量，并打印出来：

[?](https://blog.csdn.net/GrimRaider/article/details/81390878" \t "https://blog.csdn.net/GrimRaider/article/details/_self)

## jsr269提供annotation processor，允许我们在编译器编译过程中挂钩子。<http://projectlombok.org/> 的许多功能正是基于此实现。

    但有时候可能需要解析语法正确，但没有语义的Java文件（比如对工程中的单个java源文件的方法等元素建索引），这个时候jsr269就不能满足需求了。此时，我们只要语法树（ast）就可以了，也就是说不需要编译通过，只需要语法解析，可选的parser我找到了3个：

-- antlr parser

-- eclipse jdt parser

-- javac parser

下面一个例子使用javac parser来获得ast，并采用visitor模式遍历整颗语法树,提取文件中的所有方法。

注意：

     需要在classpath中引入jdk的tools.jar

     很多类不属于标准api，目前只在openjdk6,7上做过测试

【转】JDK的Parser来解析Java源代码详解\_Java\_GrimRaider的专栏-CSDN博客

java源码解析JavaParser\_Java\_shijiaolong0的博客-CSDN博客

使用openjdk的语法解析器(Parser)解析java源代码\_Java\_zerokkqq的专栏-CSDN博客