Eclipse JDT Tutorial

Sunday, March 03, 2019 11:30 AM

Installation

Install Eclipse (I use Eclispe Luna 4.4.1). Ensure JDK version 1.8 or above is installed. Open command prompt and type java -version to ensure proper JDK installation.

New Project

Open eclipse. Create a new maven project. Choose a simple project. Mention "jdt" as group id, artifact id and name.

Edit the mvn pom.xml file and add the following lines inside the <project></project> node:

```
<dependencies>
  <dependency>
    <groupId>org.eclipse.jdt</groupId>
    <artifactId>org.eclipse.jdt.core</artifactId>
    <version>3.12.2</version>
  </dependency>
 </dependencies>
```

Writing Java Code

First, let us write a simple Java method to read a java file as input.

```
public static String readFromFile(String filePath) {
		StringBuilder stringBuilder = new StringBuilder();
		try {
			BufferedReader reader = new BufferedReader(new FileReader(filePath));
			String line = null;

			String ls = System.getProperty("line.separator");

			while ((line = reader.readLine()) != null) {
				stringBuilder.append(line);
				stringBuilder.append(ls);
			}
			reader.close();
		} catch (Exception e) {
			System.out.println(e.getMessage());
		}

		return stringBuilder.toString();
	}
```

Add any file you wish to analyze into the eclipse java project. For example, you may use this Test.java file:

```
public class Test {
	public static void main(String[] args)
```

```
	{
		String str = "CMI";
		StringBuilder sb = new StringBuilder();

		for(int i = str.length() - 1; i >= 0; i--)
		{
		  if (i == 0) sb.append('S');
		  sb.append(str.charAt(i));
		}
		System.out.println(sb.toString());
	}
}
```

Create a class in src/main/java. Name it as Variable Printer.

Let us read and print our Test file. Add the following main method to the class that you created and run the class as a java application.

```
public static void main(String[] args) {
		String codeFragment = readFromFile("Test.java");
		System.out.println(codeFragment);
	}
```

You should see the contents of Test.java printed on the console.

We are now ready to use Eclipse JDT to analyze Test.java.

Traversing the AST and printing the variable names

Let us count the number of variables.

First we generate the AST for the code. Add the following method to your class.

```
public static ASTNode getASTNode(String codeFragment) {
		ASTParser parser = ASTParser.newParser(AST.JLS8);
		parser.setKind(ASTParser.K_COMPILATION_UNIT);
		parser.setSource(codeFragment.toCharArray());
		parser.setResolveBindings(false);

		ASTNode node = null;
		try {
			node = (CompilationUnit) parser.createAST(null);
		} catch (Exception e) {
			//TODO
			return null;
		}
		    return node;
	}

public static void printVariables(String codeFragment) {

		ASTNode node = getASTNode(codeFragment);

		node.accept(new ASTVisitor() {
			int varCount = 0;
```

```
                @Override
                public boolean visit(VariableDeclarationFragment node) {

                        System.out.println(node.getName().toString());
                        return super.visit(node);
                };
        });
    }
```

Change the main method to

```
public static void main(String[] args) {
        String codeFragment = readFromFile("Test.java");
        printVariables(codeFragment);
    }
```

Run the code and see the variable names getting printed.