

# Homework 1. Building A Simple Search Engine Using Apache Lucene: A Walkthrough

Younghoon Kim

nongaussian@hanyang.ac.kr

# Eclipse & Maven

Eclipse: Download & install

[Eclipse Downloads | The Eclipse Foundation](#)

Maven: Install

[m2e-core/README.md at master · eclipse-m2e/m2e-core \(github.com\)](#)

# Create A New Maven Project

New Maven Project

New Maven project

Select project name and location

☒ Create a simple project (skip archetype selection)

☒ Use default Workspace location

Location: C:\Users\WnongaWeclipse-workspace\EBIR2022T01M01 Browse...

☐ Add project(s) to working set

Working set: More...

Advanced

? < Back Next > Finish Cancel

New Maven Project

New Maven project

Configure project

Artifact

Group Id: edu.hanyang

Artifact Id: MIR202212345M01

Version: 0.0.1-SNAPSHOT

Packaging: jar

Name:

Description:

Parent Project

Group Id:

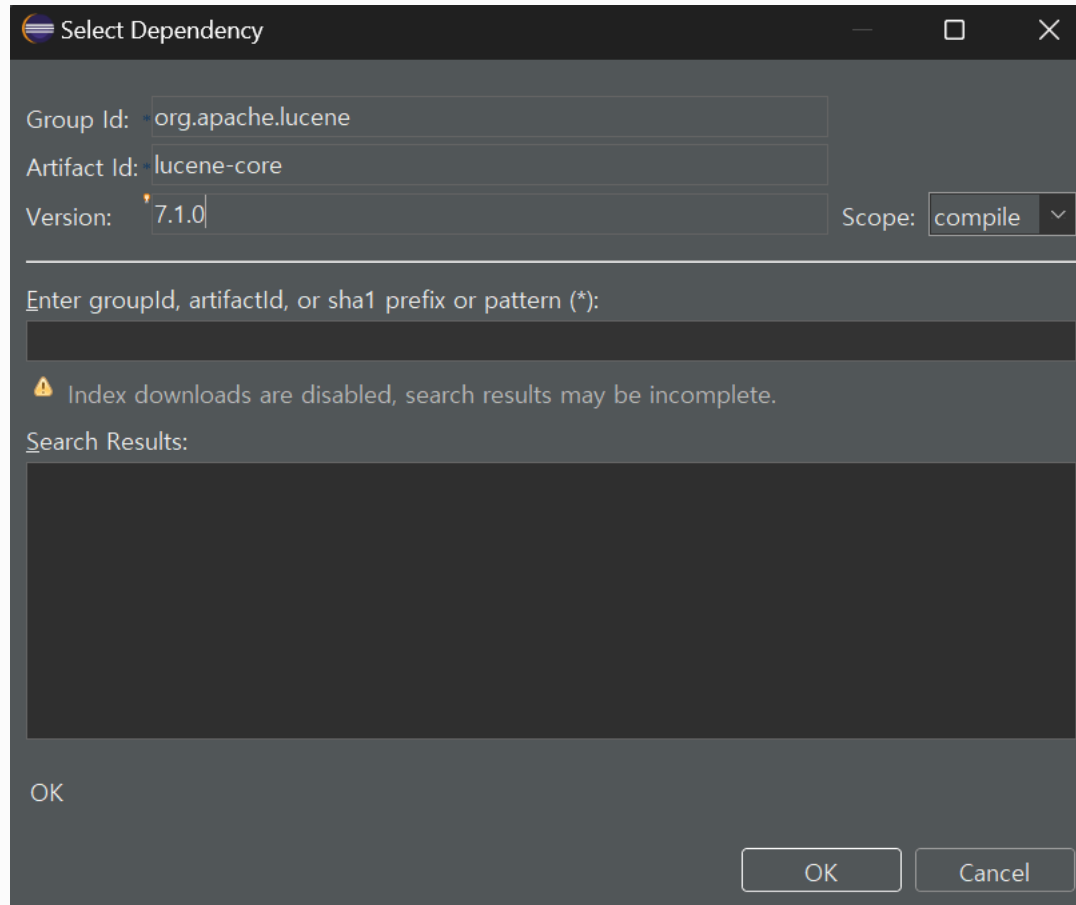
Artifact Id:

Version: Browse... Clear

Advanced

? < Back Next > Finish Cancel

# Add Dependencies To pom.xml



Select Dependency


Group Id:

Artifact Id:

Version:

Scope:

Enter groupId, artifactId, or sha1 prefix or pattern (\*):

 Index downloads are disabled, search results may be incomplete.

Search Results:

OK

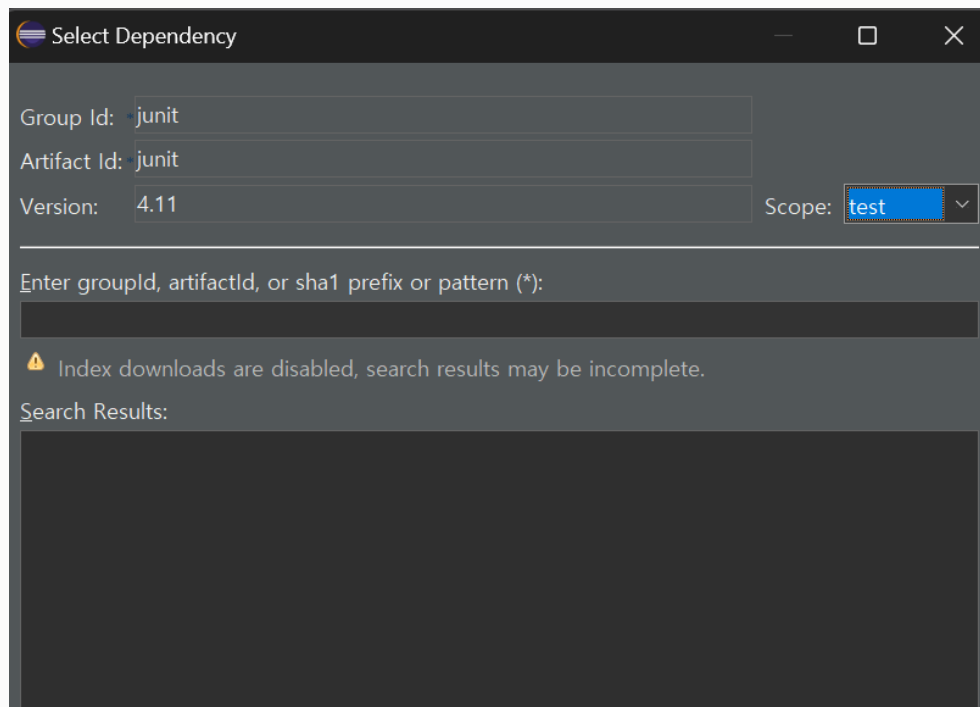
OK Cancel

```
<dependency>  
  <groupId>org.apache.lucene</groupId>  
  <artifactId>lucene-core</artifactId>  
  <version>7.2.1</version>  
</dependency>
```

```
<dependency>  
  <groupId>org.apache.lucene</groupId>  
  <artifactId>lucene-queryparser</artifactId>  
  <version>7.2.1</version>  
</dependency>
```

# Add Dependencies To pom.xml

Add junit of version 4.11 with test scope



Select Dependency

Group Id:

Artifact Id:

Version:  Scope:

Enter groupId, artifactId, or sha1 prefix or pattern (\*):

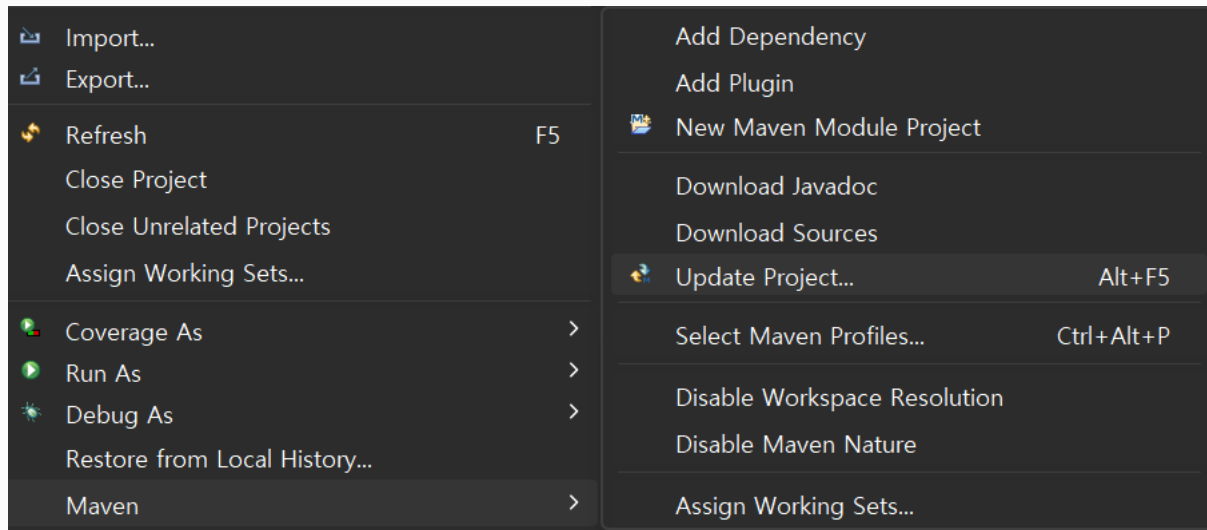
⚠ Index downloads are disabled, search results may be incomplete.

Search Results:

# Update Project

Save the change of pom.xml

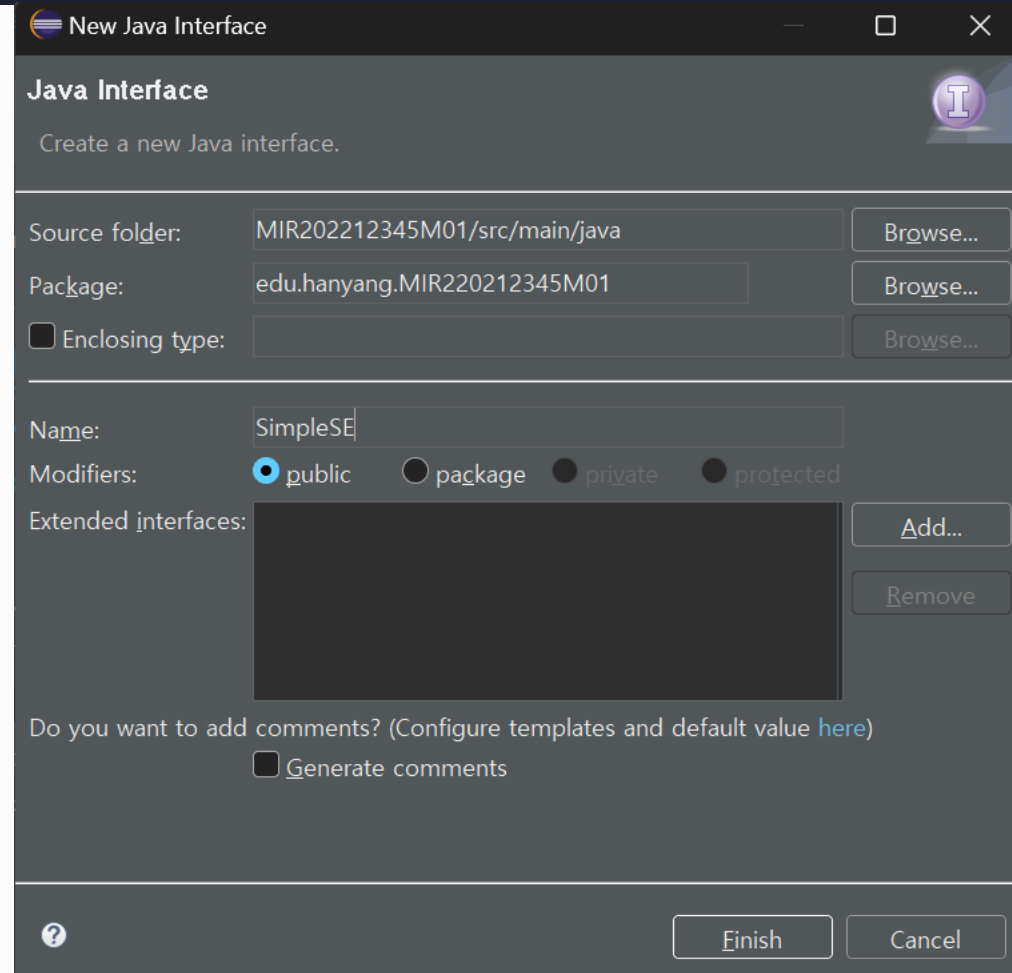
Right click > Maven > Update Project



# Create A New Interface

Class name: SimpleSE

Package: edu.hanyang.<your project name>



The screenshot shows the 'New Java Interface' dialog box. The title bar reads 'New Java Interface'. The main heading is 'Java Interface' with a subtext 'Create a new Java interface.' and an icon of a purple 'I' in a circle. The dialog contains several input fields and buttons:

- Source folder:** A text field containing 'MIR202212345M01/src/main/java' and a 'Browse...' button.
- Package:** A text field containing 'edu.hanyang.MIR202212345M01' and a 'Browse...' button.
- Enclosing type:** A checkbox labeled 'Enclosing type:' followed by an empty text field and a 'Browse...' button.
- Name:** A text field containing 'SimpleSE'.
- Modifiers:** Four radio buttons: 'public' (selected), 'package', 'private', and 'protected'.
- Extended interfaces:** A large empty text area with 'Add...' and 'Remove' buttons to its right.
- Comments:** A checkbox labeled 'Generate comments' with the text 'Do you want to add comments? (Configure templates and default value [here](#))' above it.
- Footer:** A question mark icon on the left and 'Finish' and 'Cancel' buttons on the right.

# Complete SimpleSE Interface

```
package edu.hanyang.MIR220212345M01;

import java.io.IOException;
import org.apache.lucene.analysis.Analyzer;
import org.apache.lucene.queryparser.classic.ParseException;
import org.apache.lucene.store.Directory;

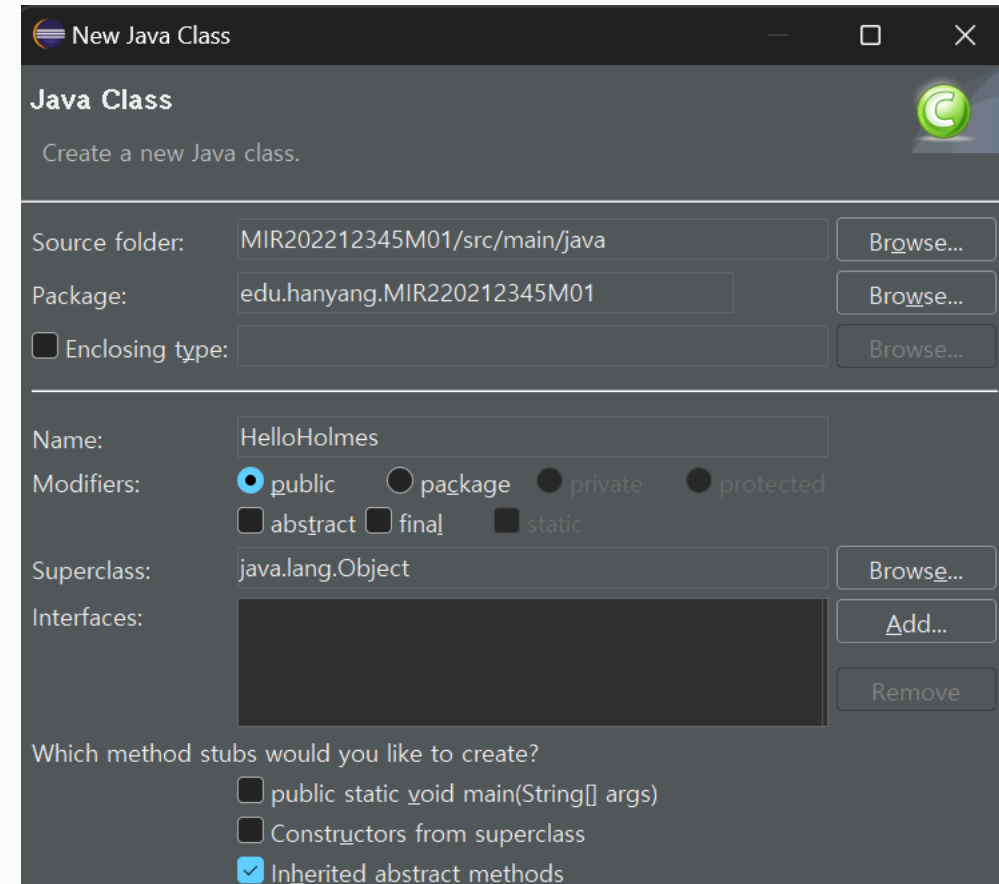
public interface SimpleSE {
    Directory createIndex(String[][] docs, Analyzer analyzer) throws
        IOException;
    String[][] search(Directory index, String querystr, Analyzer analyzer)
        throws ParseException, IOException;
}
```



# Create A New Java Class

Class name: HelloHolmes

Package: edu.hanyang.<your project name>



The screenshot shows the 'New Java Class' dialog box with the following configuration:

- Source folder:** MIR202212345M01/src/main/java
- Package:** edu.hanyang.MIR202212345M01
- Enclosing type:** (empty)
- Name:** HelloHolmes
- Modifiers:** ☒ public, ☐ package, ☐ private, ☐ protected, ☐ abstract, ☐ final, ☐ static
- Superclass:** java.lang.Object
- Interfaces:** (empty list)
- Which method stubs would you like to create?**
  - ☐ public static void main(String[] args)
  - ☐ Constructors from superclass
  - ☒ Inherited abstract methods

# Add Our Junit Class

Download TestHelloHolmes.java from LMS

Copy it into src/test/java/edu/hanyang/<your project name>/

Update the package name in the java file

# Add Our Input Document

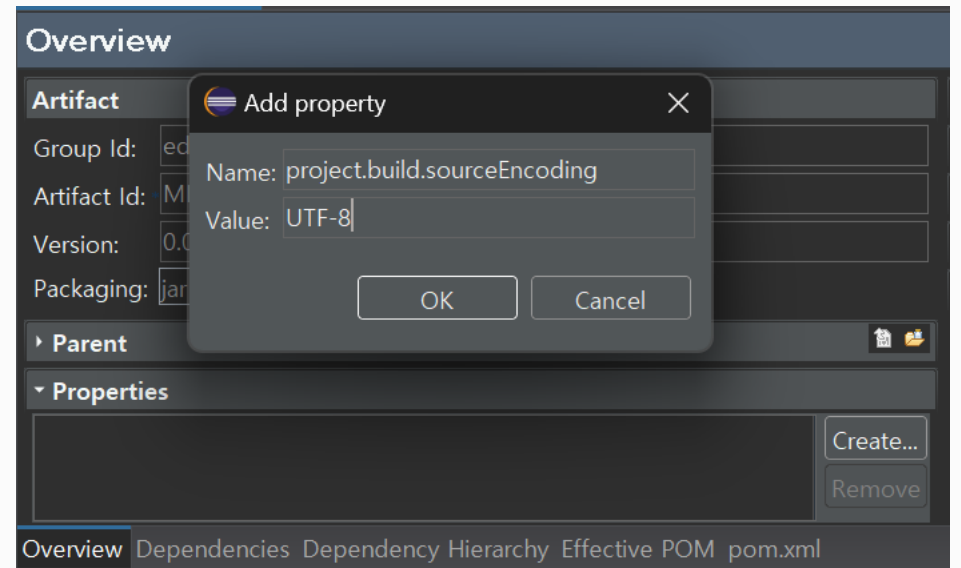
Download 244-8.txt file from LMS

Copy it into /src/test/resources/

# Misc

Add four properties in pom.xml:

- `project.build.sourceEncoding` / UTF-8
- `project.reporting.outputEncoding` / UTF-8
- `maven.compiler.source` / 16
- `maven.compiler.target` / 16



# Run Maven Test

Confirm if you fail at testing as shown in the below:

```
Tests run: 3, Failures: 0, Errors: 3, Skipped: 0
```

```
[INFO] -----  
[INFO] BUILD FAILURE  
[INFO] -----  
[INFO] Total time: 1.648 s  
[INFO] Finished at: 2023-03-24T15:49:45+09:00  
[INFO] -----
```

# Complete HelloHolmes.java

```
package edu.hanyang.MIR220212345M01;

import java.io.IOException;

import org.apache.lucene.analysis.Analyzer;
import org.apache.lucene.document.Document;
import org.apache.lucene.document.Field;
import org.apache.lucene.document.StringField;
import org.apache.lucene.document.TextField;
import org.apache.lucene.index.DirectoryReader;
import org.apache.lucene.index.IndexReader;
import org.apache.lucene.index.IndexWriter;
import org.apache.lucene.index.IndexWriterConfig;
import org.apache.lucene.queryparser.classic.ParseException;
import org.apache.lucene.queryparser.classic.QueryParser;
import org.apache.lucene.search.IndexSearcher;
```

```
import org.apache.lucene.search.Query;
import org.apache.lucene.search.ScoreDoc;
import org.apache.lucene.search.TopScoreDocCollector;
import org.apache.lucene.store.Directory;
import org.apache.lucene.store.RAMDirectory;

public class HelloHolmes implements SimpleSE
{
    public Directory createIndex(String[][] docs, Analyzer analyzer) throws IOException {
        // 1. Index
        Directory index = new RAMDirectory();
        IndexWriterConfig config = new IndexWriterConfig(analyzer);
        IndexWriter w = new IndexWriter(index, config);
        for (String[] doc: docs) {
            addDoc(w, doc[0], doc[1]);
        }
        w.close();
        return index;
    }

    public String[][] search(Directory index, String querystr, Analyzer analyzer) throws
    ParseException, IOException {
        Query q = new QueryParser("txt", analyzer).parse(querystr);

        int hitsPerPage = 10;
```

```

        return index;
    }

    public String[][] search(Directory index, String querystr, Analyzer analyzer) throws
        ParseException, IOException {
        Query q = new QueryParser("txt", analyzer).parse(querystr);

        int hitsPerPage = 10;
        IndexReader reader = DirectoryReader.open(index);
        IndexSearcher searcher = new IndexSearcher(reader);
        TopScoreDocCollector collector = TopScoreDocCollector.create(hitsPerPage);
        searcher.search(q, collector);
        ScoreDoc[] hits = collector.topDocs().scoreDocs;
        String[][] result = new String[hits.length][2];
        for(int i=0; i<hits.length; i++) {
            int docId = hits[i].doc;
            Document d = searcher.doc(docId);
            result[i][0] = d.get("lnum");
            result[i][1] = d.get("txt");
        }
        reader.close();
        return result;
    }

```

```

    public void addDoc(IndexWriter w, String lnum, String txt) throws IOException {
        Document doc = new Document();
        doc.add(new StringField("lnum", lnum, Field.Store.YES));
        doc.add(new TextField("txt", txt, Field.Store.YES));
    }

```



```
IndexReader reader = DirectoryReader.open(index);
IndexSearcher searcher = new IndexSearcher(reader);
TopScoreDocCollector collector = TopScoreDocCollector.create(hitsPerPage);
searcher.search(q, collector);
ScoreDoc[] hits = collector.topDocs().scoreDocs;
String[][] result = new String[hits.length][2];
for(int i=0; i<hits.length; i++) {
    int docId = hits[i].doc;
    Document d = searcher.doc(docId);
    result[i][0] = d.get("lnum");
    result[i][1] = d.get("txt");
}
reader.close();
return result;
}
```

```
public void addDoc(IndexWriter w, String lnum, String txt) throws IOException {
    Document doc = new Document();
    doc.add(new StringField("lnum", lnum, Field.Store.YES));
    doc.add(new TextField("txt", txt, Field.Store.YES));
    w.addDocument(doc);
}
}
```

# Run Maven Test

Results :

Tests run: 3, Failures: 0, Errors: 0, Skipped: 0

```
[INFO] -----  
[INFO] BUILD SUCCESS  
[INFO] -----  
[INFO] Total time: 1.596 s  
[INFO] Finished at: 2023-03-24T15:57:04+09:00  
[INFO] -----
```