<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>cn.devmeteor</groupId>

<artifactId>RuanZhuCode</artifactId>

<version>1.1</version>

<name>RuanZhuCode</name>

<!-- FIXME change it to the project's website -->

<url>http://www.example.com</url>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<maven.compiler.source>1.7</maven.compiler.source>

<maven.compiler.target>1.7</maven.compiler.target>

</properties>

<dependencies>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.11</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.apache.poi</groupId>

<artifactId>poi</artifactId>

<version>4.1.2</version>

</dependency>

<dependency>

<groupId>org.apache.poi</groupId>

<artifactId>ooxml-schemas</artifactId>

<version>1.4</version>

</dependency>

<dependency>

<groupId>org.apache.poi</groupId>

<artifactId>poi-ooxml-schemas</artifactId>

<version>4.1.2</version>

</dependency>

<dependency>

<groupId>org.apache.xmlbeans</groupId>

<artifactId>xmlbeans</artifactId>

<version>3.1.0</version>

</dependency>

<dependency>

<groupId>org.apache.poi</groupId>

<artifactId>poi-examples</artifactId>

<version>4.1.2</version>

</dependency>

</dependencies>

<build>

<pluginManagement><!-- lock down plugins versions to avoid using Maven defaults (may be moved to parent pom) -->

<plugins>

<!-- clean lifecycle, see https://maven.apache.org/ref/current/maven-core/lifecycles.html#clean\_Lifecycle -->

<plugin>

<artifactId>maven-clean-plugin</artifactId>

<version>3.1.0</version>

</plugin>

<!-- default lifecycle, jar packaging: see https://maven.apache.org/ref/current/maven-core/default-bindings.html#Plugin\_bindings\_for\_jar\_packaging -->

<plugin>

<artifactId>maven-resources-plugin</artifactId>

<version>3.0.2</version>

</plugin>

<plugin>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.8.0</version>

</plugin>

<plugin>

<artifactId>maven-surefire-plugin</artifactId>

<version>2.22.1</version>

</plugin>

<plugin>

<artifactId>maven-jar-plugin</artifactId>

<version>3.0.2</version>

</plugin>

<plugin>

<artifactId>maven-install-plugin</artifactId>

<version>2.5.2</version>

</plugin>

<plugin>

<artifactId>maven-deploy-plugin</artifactId>

<version>2.8.2</version>

</plugin>

<!-- site lifecycle, see https://maven.apache.org/ref/current/maven-core/lifecycles.html#site\_Lifecycle -->

<plugin>

<artifactId>maven-site-plugin</artifactId>

<version>3.7.1</version>

</plugin>

<plugin>

<artifactId>maven-project-info-reports-plugin</artifactId>

<version>3.0.0</version>

</plugin>

</plugins>

</pluginManagement>

</build>

</project>

package cn.devmeteor.ruanzhucode;

import org.apache.poi.xwpf.model.XWPFHeaderFooterPolicy;

import org.apache.poi.xwpf.usermodel.\*;

import org.openxmlformats.schemas.wordprocessingml.x2006.main.\*;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.IOException;

import java.math.BigInteger;

import java.text.SimpleDateFormat;

import java.util.\*;

public class App {

private static final String name = "软件著作权代码文档生成器";

private static final String version = "v1.1";

private static final String sourcePath = "F:\\Idea\\RuanZhuCode";

private static final String outputPath = "F:\\Idea\\RuanZhuCode";

private static final String[] myExcludeFiles = new String[]{"F:\\Idea\\RuanZhuCode\\RuanZhuCode.iml", "F:\\Idea\\RuanZhuCode\\README.md"};

private static final String[] myExcludeDirs = new String[]{"F:\\Idea\\RuanZhuCode\\.idea", "F:\\Idea\\RuanZhuCode\\target", "F:\\Idea\\RuanZhuCode\\src\\test"};

public static void main(String[] args) throws IOException {

List<String> excludeDirs = Arrays.asList(myExcludeDirs);

String[] audios = new String[]{"mp3", "wav", "aif", "aiff", "mp1", "mp2", "ra", "ram", "mid", "rmi", "m4a", "wma", "cda", "ogg", "ape", "flac", "aac", "ac3", "mmf", "amr", "m4r", "wavpack"};

String[] videos = new String[]{"avi", "mov", "qt", "asf", "rm", "rmvb", "navi", "divx", ",mp4", "mpeg", "mpg", "flv", "mkv", "3gp", "wmv", "vob", "swf"};

String[] images = new String[]{"webp", "jpg", "png", "ico", "bmp", "gif", "tif", "tga", "pcx", "jpeg", "exif", "fpx", "svg", "psd", "cdr", "pcd", "dxf", "ufo", "eps", "ai", "hdri", "raw", "wmf", "flic", "emf"};

String[] docs = new String[]{"doc", "docx", "xls", "ppt", "pptx", "pdf"};

String[] executable = new String[]{"exe", "apk", "ipa", "app"};

String[] zips = new String[]{"zip", "rar", "arj", "z", "tar", "gz", "iso", "jar"};

List<String> excludeFiles = new ArrayList<>();

excludeFiles.addAll(Arrays.asList(audios));

excludeFiles.addAll(Arrays.asList(videos));

excludeFiles.addAll(Arrays.asList(images));

excludeFiles.addAll(Arrays.asList(docs));

excludeFiles.addAll(Arrays.asList(executable));

excludeFiles.addAll(Arrays.asList(zips));

excludeFiles.addAll(Arrays.asList(myExcludeFiles));

File root = new File(sourcePath);

Queue<File> dirQueue = new ArrayDeque<>();

dirQueue.add(root);

List<File> files = new ArrayList<>();

while (!dirQueue.isEmpty()) {

File dir = dirQueue.poll();

for (File f : dir.listFiles()) {

if (f.isDirectory() && !excludeDirs.contains(f.getAbsolutePath())&&!f.getName().equals(".git"))

dirQueue.add(f);

else if (f.isFile() && !matchExclude(f, excludeFiles))

files.add(f);

}

}

String s = "\n ";

for (File file : files) {

Scanner scanner = new Scanner(new FileInputStream(file));

while (scanner.hasNext())

s += scanner.nextLine() + "\n";

scanner.close();

}

s = s.replaceAll("(?<!:)\\/\\/.\*|\\/\\\*(\\s|.)\*?\\\*\\/", "");

s = s.replaceAll("\\/\\\*(\\s|.)\*?\\\*\\/", "");

s = s.replaceAll("(?m)^\\s\*$(\\n|\\r\\n)", "");

XWPFDocument doc = new XWPFDocument(new FileInputStream("template/template.docx"));

CTSectPr padding = doc.getDocument().getBody().addNewSectPr();

CTPageMar pageMar = padding.addNewPgMar();

pageMar.setLeft(BigInteger.valueOf(1440L));

pageMar.setTop(BigInteger.valueOf(1100L));

pageMar.setRight(BigInteger.valueOf(1440L));

pageMar.setBottom(BigInteger.valueOf(1100L));

createHeader(doc);

createFooter(doc);

Scanner scanner = new Scanner(s);

int total=0;

while (scanner.hasNext()){

total++;

scanner.nextLine();

}

System.out.println("总计："+total+"行");

scanner=new Scanner(s);

while (scanner.hasNext()) {

XWPFParagraph p1 = doc.createParagraph();

XWPFRun r1 = p1.createRun();

r1.setFontFamily("等线 (西文正文)");

r1.setFontSize(10);

r1.setText(scanner.nextLine());

}

scanner.close();

doc.getDocument().getBody().removeP(0);

File file = new File(outputPath+"/"+name+version+".docx");

if (file.exists())

file.delete();

else

file.createNewFile();

FileOutputStream out = new FileOutputStream(file);

doc.getProperties().getCoreProperties().setCreator("RuanZhuCode");

doc.getProperties().getCoreProperties().setLastModifiedByUser("RuanZhuCode");

doc.getProperties().getCoreProperties().setRevision("1");

doc.getProperties().getCoreProperties().setModified(new SimpleDateFormat("yyyy-MM-dd HH:mm:ss").format(new Date()));

doc.write(out);

out.close();

}

private static void createFooter(XWPFDocument doc) {

CTP pageNo = CTP.Factory.newInstance();

XWPFParagraph footer = new XWPFParagraph(pageNo, doc);

CTPPr begin = pageNo.addNewPPr();

begin.addNewPStyle().setVal("style21");

begin.addNewJc().setVal(STJc.CENTER);

pageNo.addNewR().addNewFldChar().setFldCharType(STFldCharType.BEGIN);

pageNo.addNewR().addNewInstrText().setStringValue("PAGE \\\* MERGEFORMAT");

pageNo.addNewR().addNewFldChar().setFldCharType(STFldCharType.SEPARATE);

CTR end = pageNo.addNewR();

CTRPr endRPr = end.addNewRPr();

endRPr.addNewNoProof();

endRPr.addNewLang().setVal("zh-CN");

end.addNewFldChar().setFldCharType(STFldCharType.END);

CTSectPr sectPr = doc.getDocument().getBody().isSetSectPr() ? doc.getDocument().getBody().getSectPr() : doc.getDocument().getBody().addNewSectPr();

XWPFHeaderFooterPolicy policy = new XWPFHeaderFooterPolicy(doc, sectPr);

policy.createFooter(STHdrFtr.DEFAULT, new XWPFParagraph[]{footer});

}

private static void createHeader(XWPFDocument doc) {

CTSectPr sectPr = doc.getDocument().getBody().addNewSectPr();

XWPFHeaderFooterPolicy headerFooterPolicy = new XWPFHeaderFooterPolicy(doc, sectPr);

XWPFHeader header = headerFooterPolicy.createHeader(XWPFHeaderFooterPolicy.DEFAULT);

XWPFParagraph paragraph = header.createParagraph();

paragraph.setBorderBottom(Borders.THICK);

XWPFRun run = paragraph.createRun();

run.setText(name + " " + version + " 源代码");

run.setFontFamily("等线 (中文正文)");

run.setFontSize(9);

}

private static boolean matchExclude(File f, List<String> excludeList) {

for (String e : excludeList)

if (f.getAbsolutePath().equals(e) || f.getName().equals(".gitignore") || f.getName().endsWith("." + e.toLowerCase()))

return true;

return false;

}

}