实现的一种方式

# 修改word模板

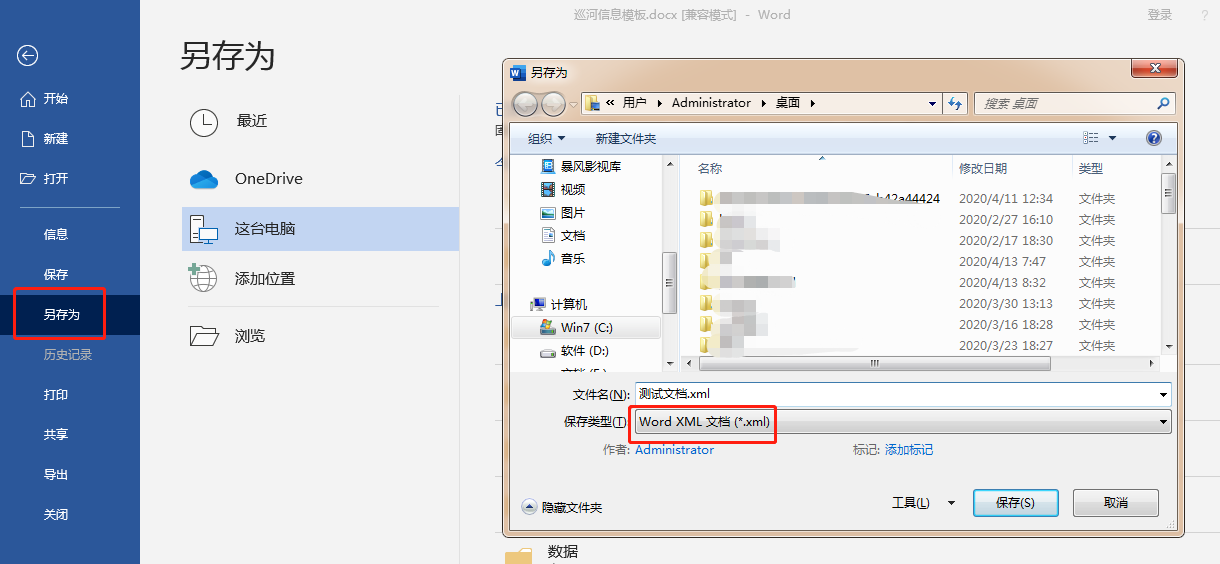
## 1 准备需要生成的word模板



需要填写内容的地方，提前写好，便于下一步的替换。需要图片的地方用一张图片代替。

## 2 将word模板另存为.xml文件，并将.xml文件改为.ftl

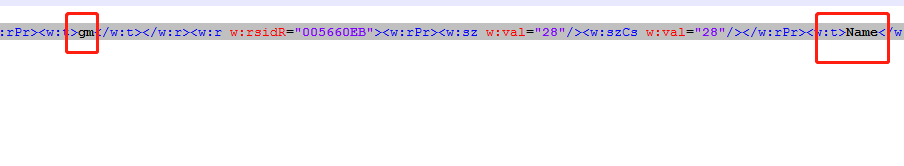
第一步：将word模板另存为.xml 文件

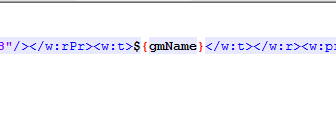


第二步：找的.xml文件改为.ftl文件

》》》》》》》》》》》》》》》》》》》》》》》》》》

第三步：修改.ftl文件中预存字符。将文本字符改为${文本字符}; 例：将gmName改为${gmName} 注： 字符可能不在一起，需要拼接后修改。





第四步：初步修改图片

图片一般是下图所示的0-Z字段。只需要将其全部选中，修改成$  
{字段}，



第五步：完善模板修改

1 ${字段}没加非空校验，故其值不能为空。

* 1. 对文本字段的处理。可以有代码负责，在数据处理的时候经行非空判断将null转为空字符串。
  2. 对与图片的处理。建议将图片放在一个list里，如果图片值为null不向list中添加。另一个好处是可以添加多张图片。
  3. <#if picList?? && (picList?size>0)> </#if> list非空校验

<#list picList as pic></#list> 从list中取值

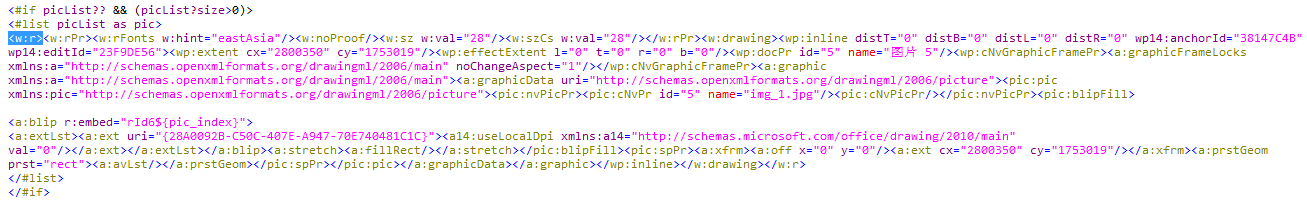
${pic\_index} list的下标 pic\_indx

${pic.字段} 去list中对象中的值 pic.字段；

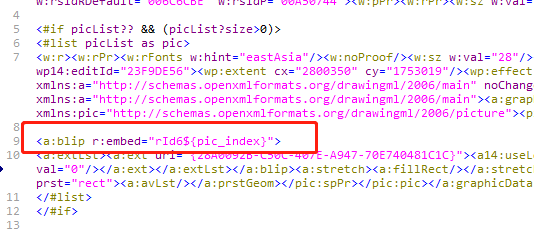
<w:r> 一行的开始

1. 图片模板处理，共有三处

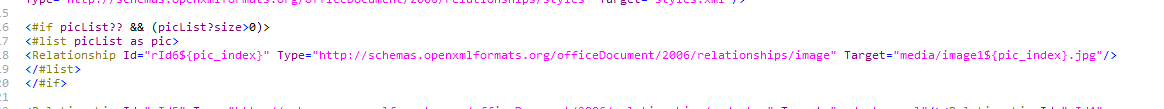
第一处：具体



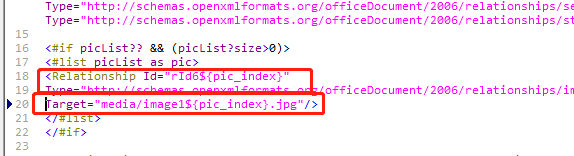
局部：



第二处：

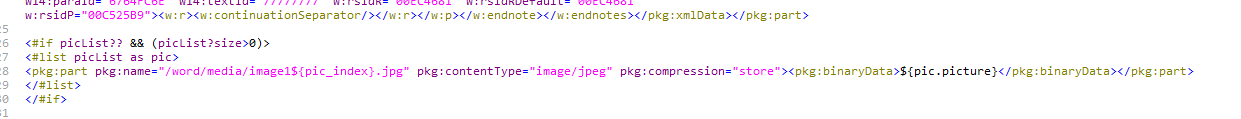


局部

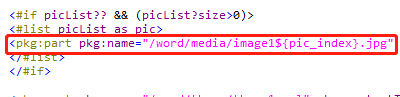


说明 第二处id与第一处对应

第三处：



局部



# Java代码

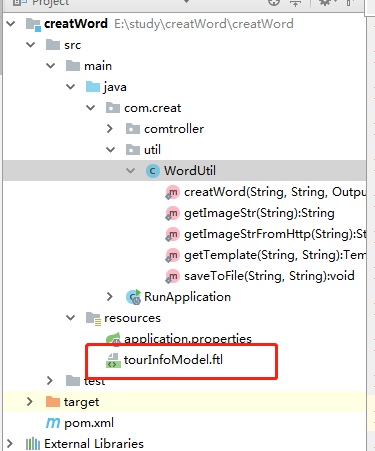
## 1 添加依赖

|  |
| --- |
| <**dependency**>  <**groupId**>org.freemarker</**groupId**>  <**artifactId**>freemarker</**artifactId**>  <**version**>2.3.30</**version**> </**dependency**> |

## 2 导入工具类

|  |
| --- |
| **package** com.creat.util;  **import** freemarker.template.Configuration; **import** freemarker.template.Template; **import** freemarker.template.TemplateException; **import** sun.misc.BASE64Encoder;  **import** java.io.\*; **import** java.net.HttpURLConnection; **import** java.net.URL; **import** java.util.Map;  */\*\*  \** ***@Author:*** *zhouyang  \** ***@Date:*** *2020/4/11  \** ***@Description:*** *根据模板（.ftl）文件生成word的工具类  \*/* **public class** WordUtil {   */\*\*  \** ***@Description:*** *获取模板  \** ***@param pathName*** *模板所在目录  \** ***@param name*** *模板名称  \** ***@return*** *Template  \*/* **private static** Template getTemplate(String pathName, String name){  Configuration configuration = **new** Configuration();  configuration.setDefaultEncoding(**"UTF-8"**);  configuration.setClassForTemplateLoading(WordUtil.**class**, pathName);   Template tempWord = **null**;  **try** {  tempWord = configuration.getTemplate(name,**"UTF-8"**);  } **catch** (IOException e) {  e.printStackTrace();  }  **return** tempWord;  }   */\*\*  \** ***@Description:*** *创建word  \** ***@param pathName*** *模板所在目录  \** ***@param name*** *模板名称  \** ***@param outputStream*** *输出流  \** ***@param dataMap*** *数据map  \*/* **public static void** creatWord(String pathName, String name, OutputStream outputStream, Map dataMap){  Template tempWord = *getTemplate*(pathName, name);  **try** {  tempWord.process(dataMap, **new** BufferedWriter(**new** OutputStreamWriter(  outputStream,**"utf-8"**)));  } **catch** (TemplateException e) {  e.printStackTrace();  } **catch** (IOException e) {  e.printStackTrace();  }  **try** {  outputStream.close();  } **catch** (IOException e) {  e.printStackTrace();  }  }  */\*\*  \** ***@Description:*** *处理本地图片  \** ***@param imgFile*** *本地图片地址  \*/* **public static** String getImageStr(String imgFile) {  **try** {  InputStream in = **null**;  **byte**[] data = **null**;  **try** {  in = **new** FileInputStream(imgFile);  data = **new byte**[in.available()];  in.read(data);  in.close();  } **catch** (Exception e) {  e.printStackTrace();  }  BASE64Encoder encoder = **new** BASE64Encoder();  **return** encoder.encode(data);  } **catch** (Exception e) {  **return null**;  }  }   */\*\*  \** ***@Description:*** *处理网络图片  \** ***@param imgFile*** *网络图片地址  \*/* **public static** String getImageStrFromHttp(String imgFile) {  **try** {  String downPath = **"c:/temp.jpg"** ;  *saveToFile*(imgFile,downPath);  String imageStr = *getImageStr*(downPath);  File file = **new** File(downPath);  **if** (file.exists()){  file.delete();  }  **return** imageStr;  } **catch** (Exception e) {  **return null**;  }  }   */\*\*  \** ***@Description:*** *下载网路图片  \** ***@param destUrl*** *网络图片地址  \** ***@param downPath*** *下载图片地址  \*/* **public static void** saveToFile(String destUrl, String downPath) {  FileOutputStream fos = **null**;  BufferedInputStream bis = **null**;  HttpURLConnection httpUrl = **null**;  URL url = **null**;  **int** BUFFER\_SIZE = 1024;  **byte**[] buf = **new byte**[BUFFER\_SIZE];  **int** size = 0;  **try** {  url = **new** URL(destUrl);  httpUrl = (HttpURLConnection) url.openConnection();  httpUrl.connect();  bis = **new** BufferedInputStream(httpUrl.getInputStream());  fos = **new** FileOutputStream(downPath);  **while** ((size = bis.read(buf)) != -1) {  fos.write(buf, 0, size);  }  fos.flush();  } **catch** (IOException e) {  } **catch** (ClassCastException e) {  } **finally** {  **try** {  fos.close();  bis.close();  httpUrl.disconnect();  } **catch** (IOException e) {  } **catch** (NullPointerException e) {  }  }  } } |

## 将模板导入项目



## 配置文件位置

为了使用方便，可根据需求自行配置

|  |
| --- |
| *# 巡河信息word模板所在目录* **tourInfoModel.pathName** = **/** *# 巡河信息word模板名称* **tourInfoModel.fileName** = **tourInfoModel.ftl** |

## 完善业务代码

核心代码

|  |
| --- |
| @Value(**"${tourInfoModel.pathName}"**) **private** String **pathName**; @Value(**"${tourInfoModel.fileName}"**) **private** String **fileName**; @Override **public** RespData createWord(HttpServletRequest req, HttpServletResponse resp) {  resp.setCharacterEncoding(**"UTF-8"**);  **try** {  req.setCharacterEncoding(**"UTF-8"**);  } **catch** (UnsupportedEncodingException e) {  e.printStackTrace();  }   *//处理数据获得map* Map dataMap = processingTourInfoData();   *//设置下载文件的类型* resp.setContentType(**"application/msword"**);  *//设置响应头信息* **try** {  resp.setHeader(**"Content-Disposition"**,**"attachment;filename="** + java.net.URLEncoder.*encode*(**"巡河记录表.doc"**, **"UTF-8"**));  } **catch** (UnsupportedEncodingException e) {  e.printStackTrace();  }  **try** {  WordUtil.*creatWord*(**pathName**,**fileName**,resp.getOutputStream(),dataMap);  } **catch** (IOException e) {  **return** RespData.*error*(**"文件创建失败"**);  }  **return** RespData.*success*(**"文件下载成功"**); } *//数据处理 模板没加非空校验 故数据处理中必须添加非空校验 模拟数据 省略 需要自己根据业务添加* **private** Map processingTourInfoData(){  Map map = **new** HashMap<>();  List<Map> picList = **new** ArrayList<>();  map.put(**"gmName"**, **"河长名"**);  map.put(**"objName"**, **"河流"**);  map.put(**"adName"**, **"行政区名称"**);  map.put(**"tourTime"**, **"2020年10月12日"**);  map.put(**"tourDuration"**, **"2小时"**);  map.put(**"question"**,**"问题1"**);  *//图片数据处理  /\* Map picMap = new HashMap<>();  String filePath="";  String pictureData = WordUtil.getImageStrFromHttp(filePath);  picMap.put("picture", pictureData);  picList.add(picMap);  map.put("picList",picList);\*/* map.put(**"picList"**,**null**);  **return** map; } |