**private** **int** ORI\_WIDTH = 300;// 590; //源文件宽度

**private** **int** ORI\_HEIGHT = 150;// 360; //源文件高度

**private** **int** X; // 抠图坐标x

**private** **int** Y; // 抠图坐标y

**private** **int** WIDTH; // 模板图宽度

**private** **int** HEIGHT; // 模板图高度

**private** **float** xPercent; // X位置移动百分比

**private** **float** yPercent; // Y位置移动百分比

一些基础数据定义

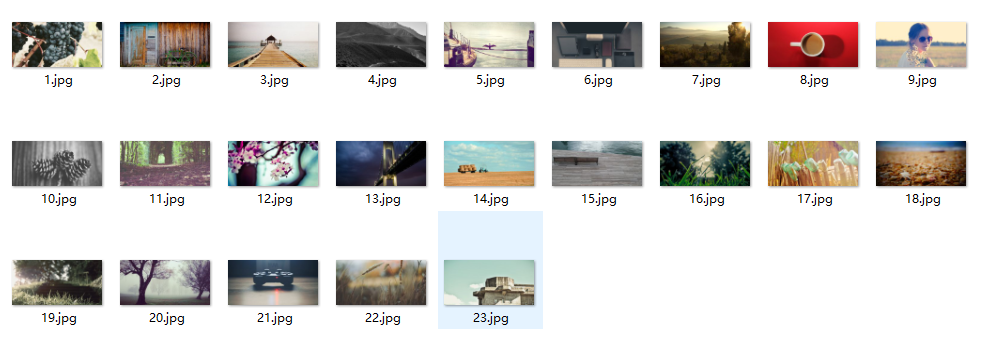
**public** Map<String, String> pictureTemplatesCut(File templateFile, File targetFile, String templateType, String targetType)

这个类的主要方法，根据模板切图

其中templateFile指切下来的图形状,下image前缀同



targetFile指需要被切的原图，后ori前缀同



BufferedImage newImage = **new** BufferedImage(WIDTH, HEIGHT, imageTemplate.getType());

这里生成了一个与模板一样的新图形newImage

generateCutoutCoordinates();

这个函数用于随机生成缺口图片的坐标x,y

并计算xy的位置移动百分比xyPercent，精确到小数点后两位

xPercent = X / ORI\_WIDTH yPercent = Y / ORI\_HEIGHT

BufferedImage targetImageNoDeal = getTargetArea(X, Y, WIDTH, HEIGHT, oriis, oriFiletype);

getTargetArea()用于在x,y处选定一个与模板一致的区域targetImageNoDeal

newImage = DealCutPictureByTemplate(targetImageNoDeal, imageTemplate, newImage);

DealCutPictureByTemplate()用于渲染图片，通过getData方法将targetImageNoDeal区域的图像复制到newImage上

**int**[][] getData(BufferedImage bimg)

这个方法将遍历图片每一个像素，记录rgb值生成对应矩阵

**byte**[] oriCopyImages = DealOriPictureByTemplate (oriImage, imageTemplate, X, Y);

将原图的缺口区域设置为透明，即将oriImage在xy处的imageTemplate区域设置为透明，输出为oriCopyImages

Map<String, String> pictureMap = **new** HashMap<>();

Base64 base64Object = **new** Base64();

pictureMap.put("newImage", base64Object.encodeToString(newImages));// 滑块

pictureMap.put("oriCopyImage", base64Object.encodeToString(oriCopyImages));// 带有缺口的背景图

pictureMap.put("pointX", String.*valueOf*(X));//缺口坐标

pictureMap.put("pointY", String.*valueOf*(Y));

**return** pictureMap;

最后将图片通过org.apache.commons.codec.binary.Base64.encodeToString方法编码为String，保存到Map<String, String>中返回。