

# 1 须知

---

本文主要描述pdf打印相关功能。

所有内容均是针对以下表格对应环境进行的说明，非此环境，本文仅做参考。

本文档仅限内部使用，如有问题可以联系 周家伟（zhoujiawei@sinosoft.com.cn）

简介	备注
本文编写场景：	国元新一代农险业务系统-打印
ireport版本：	1.2.2
ireport工具附件：	链接: <a href="https://pan.baidu.com/s/1gfnfq2f">https://pan.baidu.com/s/1gfnfq2f</a> 密码:eo9s
jasperreports包版本	3.5.3
itext包版本	4.2.0

## 2 整体描述

---

基于jasperreports做pdf打印，需要按以下2步完成：

- pdf模板制作
- java代码编译jrxml模板，并传入相关数据

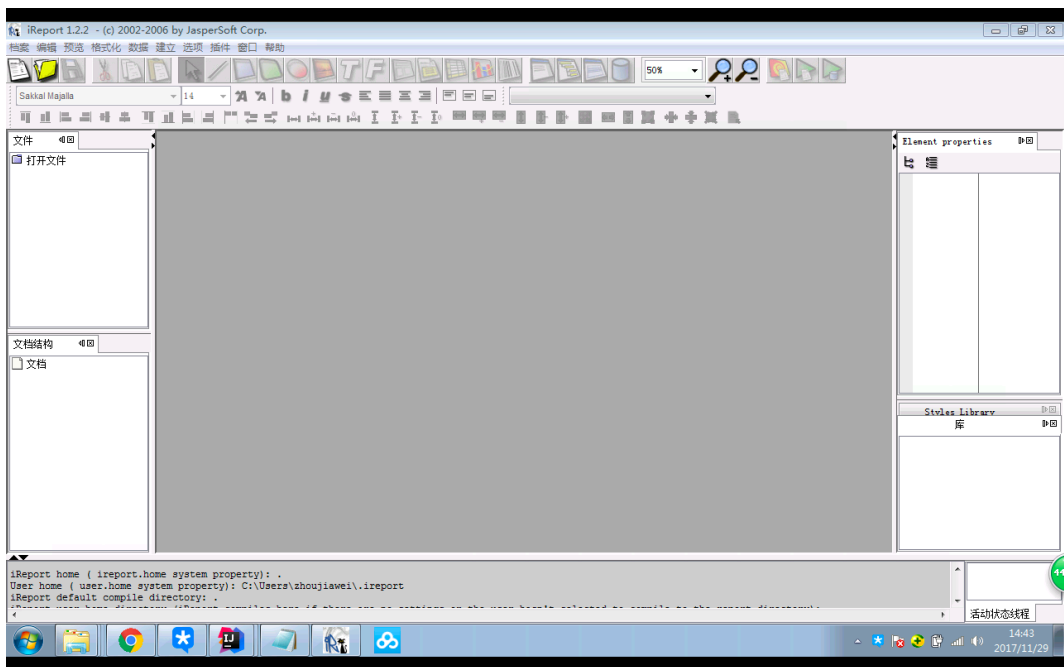
## 3 模板制作

---

### 3.1 工具界面介绍





---

- 主页：



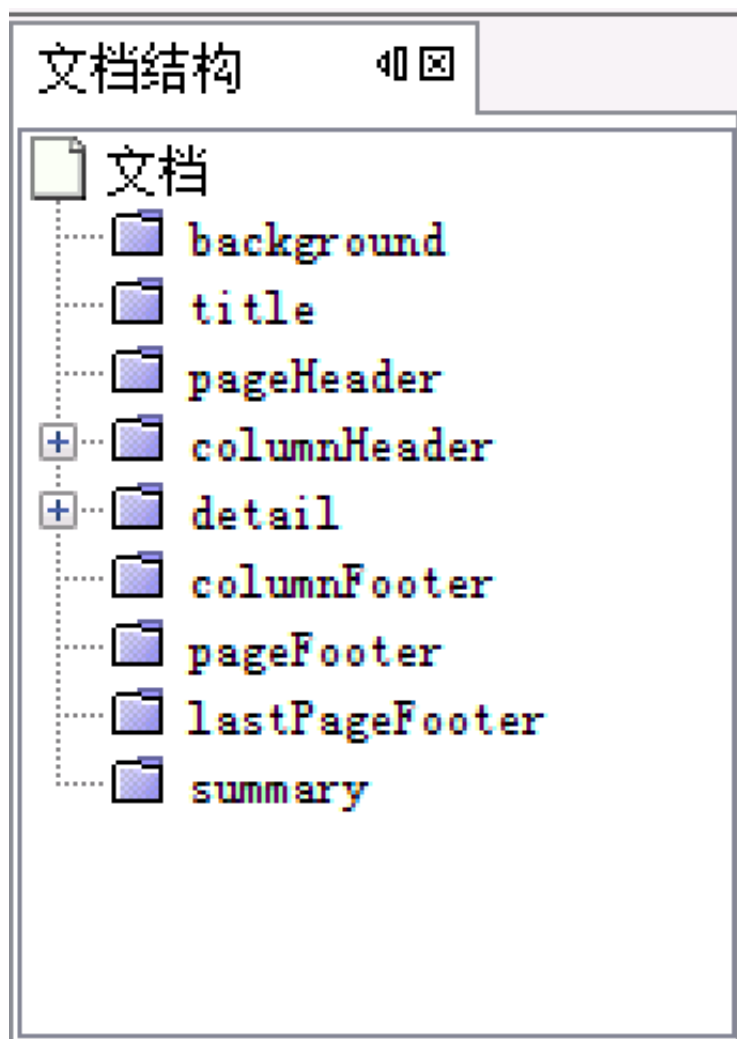
- 常用工具栏：



-  从左至右依次为：新建一个ireport文件；打开已有的ireport文件；保存文件
-  从左至右依次为：线段；矩形；圆角矩形；椭圆形
-  从左至右依次为：图像；不会变动文字；文字字段元
  - 图像：可以在pdf中插入图片，具体使用下面单独介绍。
  - 不会变动文字：就是固定的文字，例如：标题、表头等。
  - 文字字段元：此部分内容是从参数中获取的，可通过代码对pdf模板进行赋值。
-  第一个按钮是：对象属性
  - 对象属性：用于接收入参数据，并给使用。
- 文件管理：

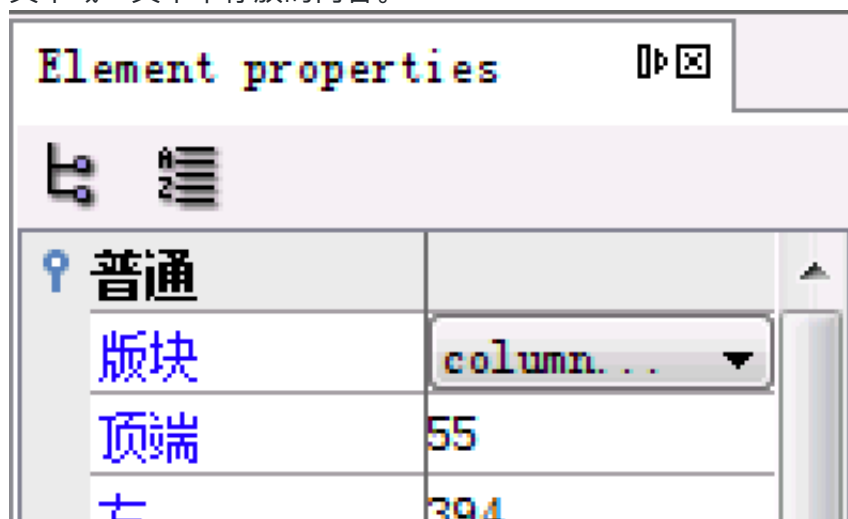











- 文档结构



○ 元素属性设置

- 普通：元素基本设置，例如：宽高等。
- 文本：元素中文本样式设置，例如：字体，字号等。
- 文本域：文本中存放的内容。

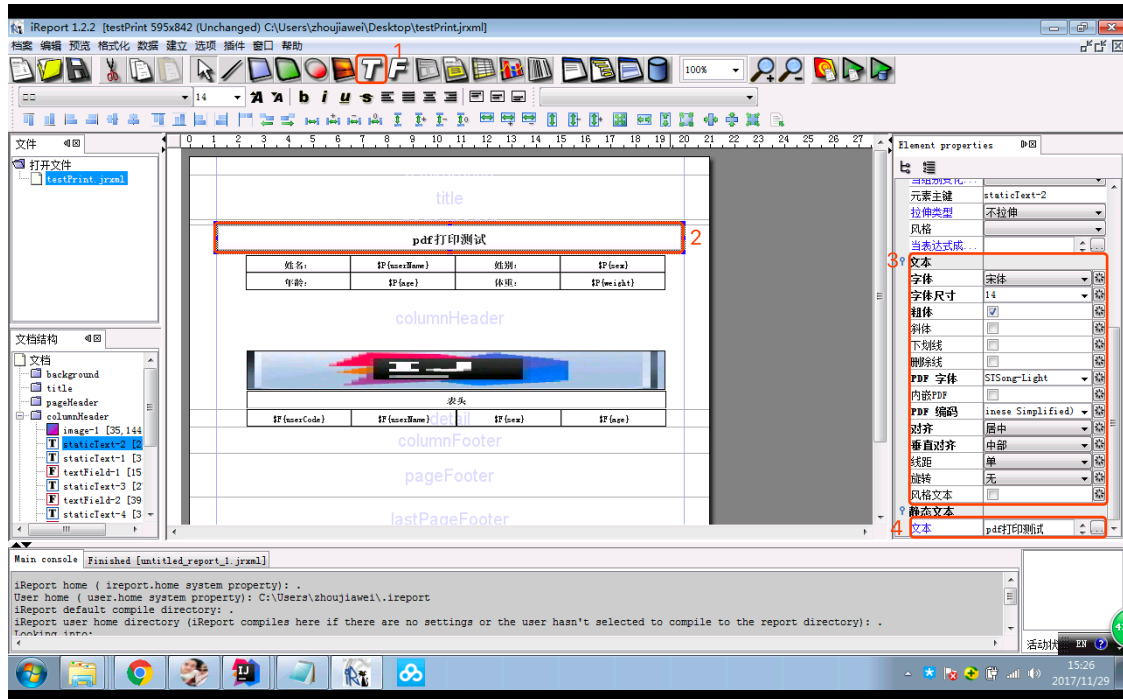


高度	20
宽度	120
前景	<input type="text"/>  
背景	<input type="text"/>  
透明	<input checked="" type="checkbox"/> 
空白时消除	<input type="checkbox"/>
打印在第...	<input type="checkbox"/>
当细目溢...	<input type="checkbox"/>
打印重复值	<input checked="" type="checkbox"/>
定位类型	相对顶... ▼
当组别变...	▼
元素主键	textField-3
拉伸类型	不拉伸 ▼
风格	▼
当表达式...	<input type="text"/>   
 文本	
字体	宋体 ▼ 
字体大小	10 

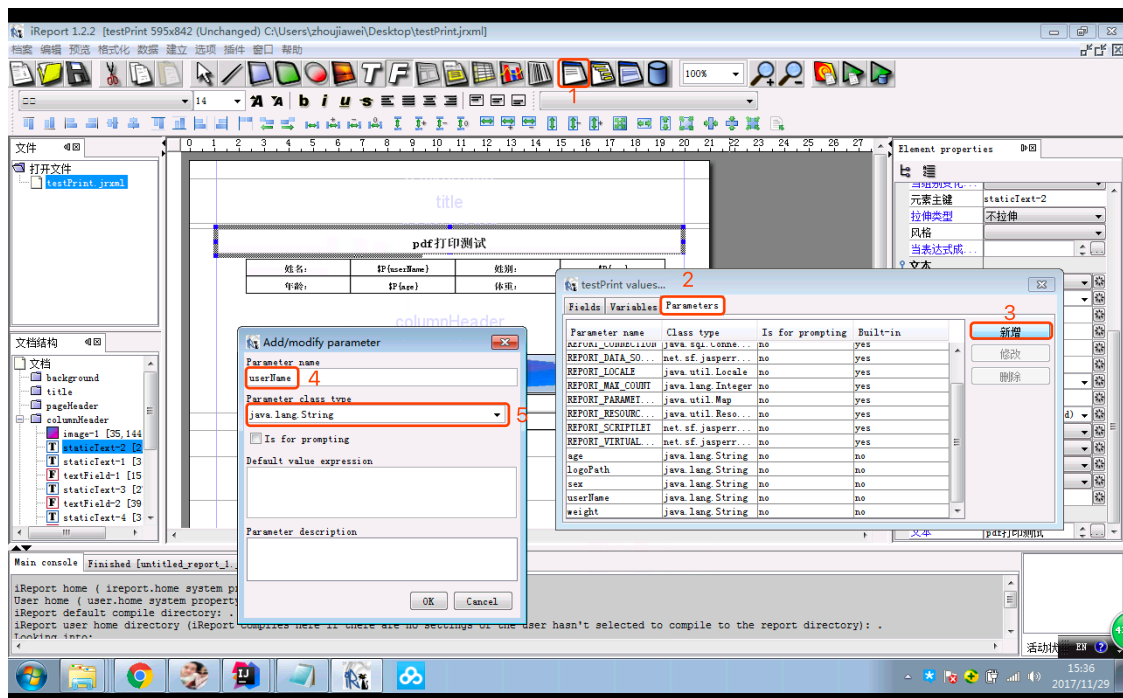
## 3.2 工具使用介绍

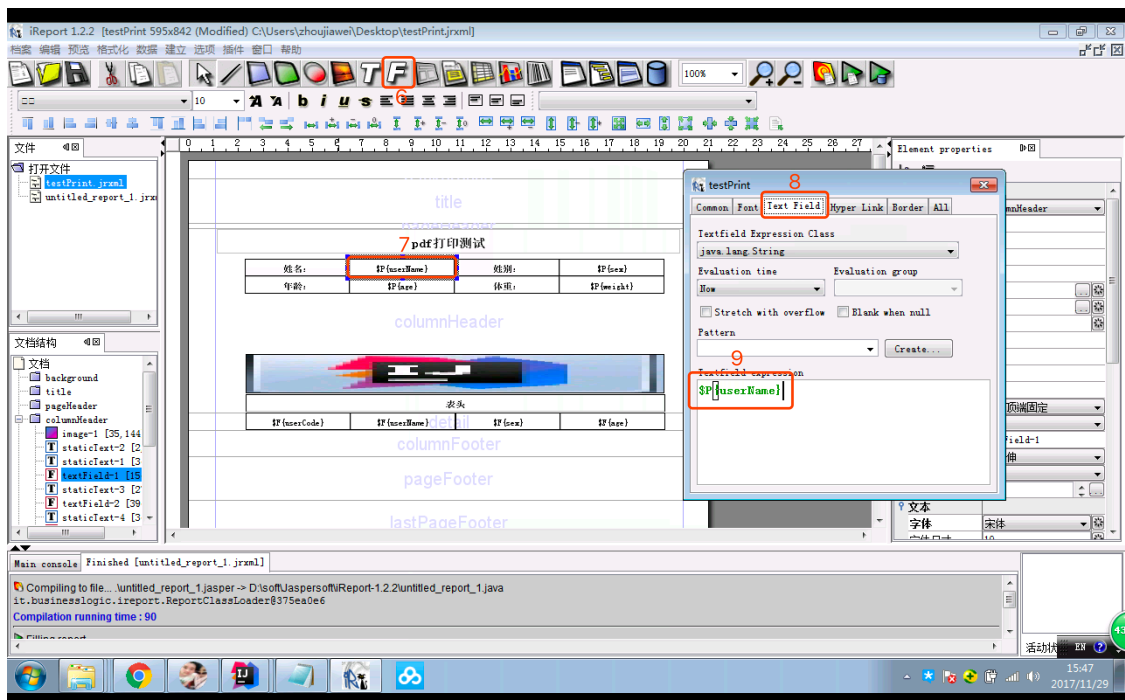
### 3.2.1 不变动文字

不变动文字，使用不变动文字画出，然后设置对于内容和样式即可，如下图所示：



### 3.2.2 文字字段元（动态文本内容）

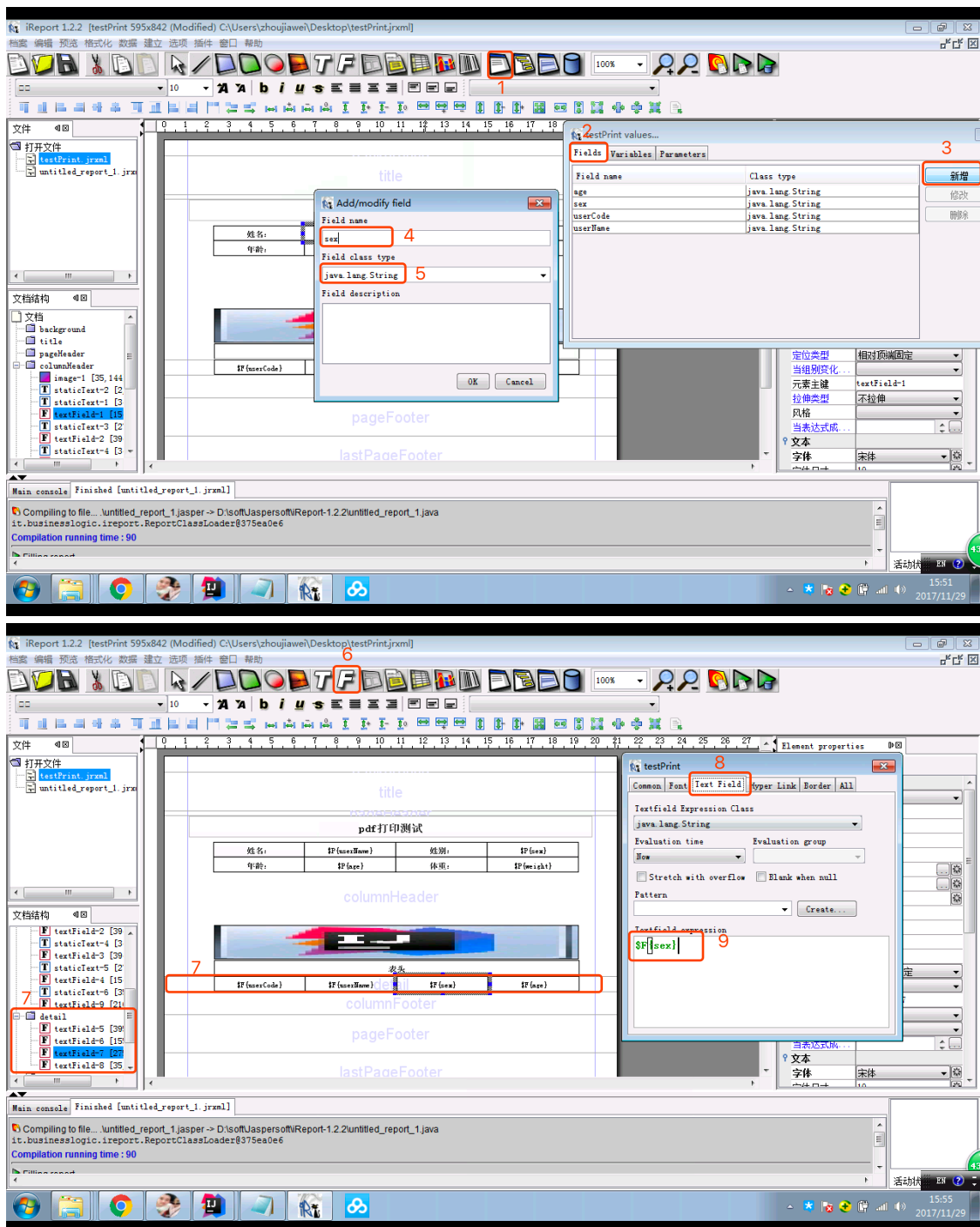




如上图所示：

- 1.点击对象属性。
- 2.选择Parameters: (入参有2中类型，第一种使用map传参，第二种是使用集合传参，集合中是对象，接收对象可以使用Fields，这个后面循环部分会讲到)。
- 3.选择新增变量。
- 4.填写变量名，**注意：变量名需要和代码中入参完全一致，且遵循驼峰命名规则。**
- 5.选择属性类型。
- 6.选择文字字段元。
- 7.在对于区域画出对应文本域。
- 8.右击7步的文本域，选择属性->Text Field。
- 9.取值，对于map传入参数，使用 `$P{**}` 取值。

### 3.2.3 文字字段元（循环列表显示）



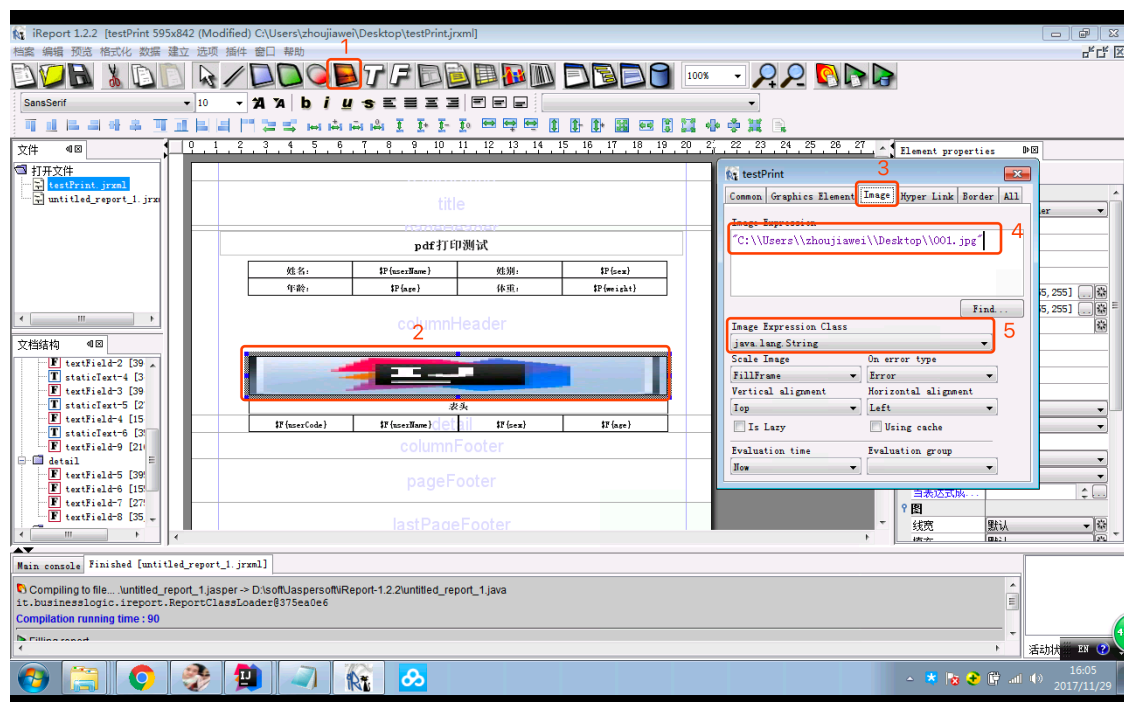
如上图所示：

- 1.选择对象属性。
- 2.选择Fields。
- 3.新增。
- 4.输入变量名称**注意：Fields中的数据都是循环数据，所以此属性变量要与如参集中每个对象中的属性保持一致。**



- 5.设置参数类型。
- 6.选择文字字段元。
- 7.在detail域中绘制模板（注意：只有在detail中绘制的内容才会循环显示）。
- 8.设置属性，选择Text Field。
- 9.设置取值，注意使用 `$F{**}` 取值。

## 3.2.4 图片



如上图所示：

- 1.选择图像。
- 2.绘制模板。
- 3.设置图像属性。
- 4.设置图片路径，注意此路径可以通过参数动态传入。和普通文本使用一致，需要先定义变量，然后使用 `$P{**}` 来获取变量。
- 5.设置参数类型，如果是通过路径来显示图片，此类型为String。

## 4 代码开发

## 4.1 依赖包引入

在对于工程的pom.xml中增加以下依赖：

```
<!-- pdf start-->
<dependency>
  <groupId>jasperreports</groupId>
  <artifactId>jasperreports</artifactId>
  <version>3.5.3</version>
</dependency>
<dependency>
  <groupId>com.lowagie</groupId>
  <artifactId>itext</artifactId>
  <version>4.2.0</version>
</dependency>
<!-- pdf end-->
```

**注意：jasperreports包3.5.3以上版本可能与下文涉及到的工具类不兼容，3.5.3以下的版本可能与itext4.2.0的包不兼容，itext包如果版本太低可能会出现图片无法显示，中文字体无法识别的问题。**

## 4.2 工具类引入

```
package com.sinosoft.demo.core.util;

import com.sinosoft.framework.core.utils.DateUtils;
import net.sf.jasperreports.engine.*;
import net.sf.jasperreports.engine.data.JRBeanCollectionDataSource;
import net.sf.jasperreports.engine.export.FontKey;
import net.sf.jasperreports.engine.export.JRPdfExporter;
import net.sf.jasperreports.engine.export.PdfFont;
import net.sf.jasperreports.engine.util.JRLoader;

import javax.servlet.http.HttpServletResponse;
import java.io.File;
import java.io.InputStream;
import java.net.URL;
import java.util.*;

/**
 * @Description: 单证打印工具类
 * @Author: sucong
```

```

    * @Date: 2017/9/3
    */

    public class PrintUtil {

        /**
         * @description 单张列印方法
         * @param list
         * @param params
         * @param response
         * @throws Exception
         * @author sucong
         * @date 2017/9/3 16:44
         */
        public void printReportPdf(List list, Map<String, Object>
params, HttpServletResponse response) throws Exception
        {
            if(list == null){
                list = new ArrayList<Object>();
                list.add(new Object());
            }
            if(list.isEmpty()){
                list.add(new Object());
            }
            try{
                JRDataSource jrDataSource = new
JRBeanCollectionDataSource(list);
                String templetPath = (String)
params.get("templetPath");
                JasperPrint jprint =
JasperFillManager.fillReport(templetPath, params, jrDataSource);
                JRPdfExporter exporter = new JRPdfExporter();
                Map<FontKey, PdfFont> fontsMap = new
HashMap<FontKey, PdfFont>();
                // 此处用黑体代表STSong-Light在pdf中的粗体, 模板中选择的字体需要
是黑体
                fontsMap.put(new FontKey("黑体", true, false), new
PdfFont("STSong-Light", "UniGB-UCS2-H", true, true, false));
                exporter.setParameter(JRExporterParameter.FONT_MAP,
fontsMap);
                exporter.setParameter(JRExporterParameter.JASPER_PRINT,
jprint);

                exporter.setParameter(JRExporterParameter.OUTPUT_STREAM,
response.getOutputStream());
                exporter.exportReport();
            }catch(Exception e){
                e.printStackTrace();
            }
        }
    }

```

```

        throw e;
    }
}

/**
 * @description 多张列印方法
 * @param listMap
 * @param n
 * @param response
 * @throws Exception
 * @author sucong
 * @date 2017/9/3 16:44
 */
public static void printReportPdfMore(List<Map<String, Object>>
listMap, int n, HttpServletResponse response) throws Exception {
    ArrayList<JasperPrint> jasperPrintList = new
ArrayList<JasperPrint>();
    InputStream inputStream;
    for(int j =0; j<n; j++){
        for (int i = 0; i < listMap.size(); i++) {
            inputStream = (InputStream)
listMap.get(i).get("templetPath");
            ArrayList list;
            list=(ArrayList)listMap.get(i).get("list");
            if(list==null || list.size()==0){
                list=new ArrayList();
                list.add(new Object());
            }
            JRDataSource jrDataSource = new
JRBeanCollectionDataSource(list);
            JasperReport jasperReport =
JasperCompileManager.compileReport(inputStream);

            jasperPrintList.add(JasperFillManager.fillReport(jasperReport,
listMap.get(i), jrDataSource));
        }
    }
    JRPdfExporter exporter = new JRPdfExporter();
    Map<FontKey, PdfFont> fontsMap = new
HashMap<FontKey, PdfFont>();
    //此处用黑体代表STSong-Light在pdf中的粗体，模板中选择的字体需要是黑
    体
    fontsMap.put(new FontKey("黑体", true, false), new
PdfFont("STSong-Light", "UniGB-UCS2-H", true, true, false));
    exporter.setParameter(JRExporterParameter.FONT_MAP,
fontsMap);

    exporter.setParameter(JRExporterParameter.JASPER_PRINT_LIST,

```

```

jasperPrintList);
        exporter.setParameter(JRExporterParameter.OUTPUT_STREAM,
response.getOutputStream());
        exporter.exportReport();
    }

    /**
     * @description 转化日期
     * @param date yyyy-MM-dd
     * @return yyyy年MM月dd日
     * @author sucong
     * @date 2017/9/3 16:44
     */
    public static String formatDateStr(String date){
        if(date!=null && !"".equals(date)){
            String dateArr[] = date.split("-");
            return dateArr[0]+"年"+dateArr[1]+"月"+dateArr[2]+"日";
        }else{
            return "";
        }
    }

    /**
     * @description 转化日期
     * @param date yyyy-MM-dd
     * @return yyyy年MM月dd日
     * @author sucong
     * @date 2017/9/3 16:44
     */
    public static String formatDate(Date date){
        if(date!=null){
            String strDate = DateUtils.formatDate(date);
            String dateArr[] = strDate.split("-");
            return dateArr[0]+"年"+dateArr[1]+"月"+dateArr[2]+"日";
        }else{
            return "";
        }
    }
}

```

## 4.3 服务代码编写

- 全量代码

```

package com.sinosoft.demo.core.printpdf.service.impl;

import com.sinosoft.demo.core.printpdf.dto.PrpDuserDto;
import com.sinosoft.demo.core.printpdf.service.PrintService;
import com.sinosoft.demo.core.util.PrintUtil;
import com.sinosoft.framework.core.service.impl.BaseServiceImpl;
import com.sinosoft.framework.exception.BusinessException;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;
import java.io.InputStream;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

/**
 * @Description: 单证打印service实现类
 * @Author mayibo
 * @Since 2017/8/29 16:28
 */
@Service
@Transactional
public class PrintServiceImpl extends BaseServiceImpl implements
PrintService {

    /** log日志 */
    private static final Logger LOGGER =
LoggerFactory.getLogger(PrintServiceImpl.class);

    private static final String TEXT_NONE = "无";

    @Override
    public void testPrint(HttpServletRequest request,
HttpServletResponse response) throws Exception {
        LOGGER.info("打印测试服务开始");
    }

    //      String templetPath =
this.getClass().getResource("/").getPath() + "template/";
        ClassLoader classLoader = this.getClass().getClassLoader();
        InputStream inputStream=null;
        try {

```

```

        List<Map<String, Object>> listMap =
getOfferParamsForProposal(request, classLoader, inputStream);
        PrintUtil.printReportPdfMore(listMap, 1, response);
    } catch (Exception e) {
        LOGGER.info("打印测试服务报错: " + e.getMessage());
        e.printStackTrace();
        throw new BusinessException("打印测试服务报错");
    } finally {
        if(inputStream!=null){
            try{
                inputStream.close();
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    }
    LOGGER.info("打印测试服务结束");
}

```

```

    public List<Map<String, Object>>
getOfferParamsForProposal(HttpServletRequest request, ClassLoader
classLoader, InputStream inputStream) throws Exception {
    inputStream =
classLoader.getResourceAsStream("template/testPrint.jrxml");
    String userName = request.getParameter("userName");
    String sex = request.getParameter("sex");
    String age = request.getParameter("age");
    String weight = request.getParameter("weight");

    List<Map<String, Object>> listMap = new
ArrayList<Map<String, Object>>();
    Map<String, Object> paramMap = new HashMap<String, Object>();
    paramMap.put("userName", userName);
    paramMap.put("sex", sex);
    paramMap.put("age", age);
    paramMap.put("weight", weight);

    List<PrpDuserDto> prpDuserDtoList = getPrpDuser();

    paramMap.put("list", prpDuserDtoList);

    paramMap.put("logoPath", "template/001.jpg");

    paramMap.put("templetPath", inputStream);
    paramMap.put("isShowImage", "true");
}

```

```
        listMap.add(paramMap);
        return listMap;
    }

    private List<PrpDuserDto> getPrpDuser() {
        List<PrpDuserDto> prpDuserDtoList = new ArrayList<>();
        prpDuserDtoList.add(new PrpDuserDto("0000", "张三", "男", "20"));
        // prpDuserDtoList.add(new PrpDuserDto("0001", "李四", "男", "20"));
        // prpDuserDtoList.add(new PrpDuserDto("0002", "王五", "男", "20"));
        // prpDuserDtoList.add(new PrpDuserDto("0003", "周三", "男", "20"));
        // prpDuserDtoList.add(new PrpDuserDto("0000", "张三", "男", "20"));
        // prpDuserDtoList.add(new PrpDuserDto("0001", "李四", "男", "20"));
        // prpDuserDtoList.add(new PrpDuserDto("0002", "王五", "男", "20"));
        // prpDuserDtoList.add(new PrpDuserDto("0003", "周三", "男", "20"));
        // prpDuserDtoList.add(new PrpDuserDto("0000", "张三", "男", "20"));
        // prpDuserDtoList.add(new PrpDuserDto("0001", "李四", "男", "20"));
        // prpDuserDtoList.add(new PrpDuserDto("0002", "王五", "男", "20"));
        // prpDuserDtoList.add(new PrpDuserDto("0003", "周三", "男", "20"));
        // prpDuserDtoList.add(new PrpDuserDto("0000", "张三", "男", "20"));
        // prpDuserDtoList.add(new PrpDuserDto("0001", "李四", "男", "20"));
        // prpDuserDtoList.add(new PrpDuserDto("0002", "王五", "男", "20"));
        // prpDuserDtoList.add(new PrpDuserDto("0003", "周三", "男", "20"));
        // prpDuserDtoList.add(new PrpDuserDto("0000", "张三", "男", "20"));
        // prpDuserDtoList.add(new PrpDuserDto("0001", "李四", "男", "20"));
        // prpDuserDtoList.add(new PrpDuserDto("0002", "王五", "男", "20"));
        prpDuserDtoList.add(new PrpDuserDto("0003", "周三", "男", "20"));
    }
}
```



```

        return prpDuserDtoList;
    }

}

```

- 模板加载,注意此参数必传。

```

InputStream =
classLoader.getResourceAsStream("template/testPrint.jrxml");

paramMap.put("templetPath", inputStream);

```

- 普通参数传入,注意名称需要与模板定义的一致。

```

paramMap.put("userName", userName);

```

- 集合参数传入,注意dto中的属性名要与模板中\$F参数一致

```

List<PrpDuserDto> prpDuserDtoList = getPrpDuser();
paramMap.put("list", prpDuserDtoList);

```

```

private List<PrpDuserDto> getPrpDuser() {
    List<PrpDuserDto> prpDuserDtoList = new ArrayList<>();
    prpDuserDtoList.add(new PrpDuserDto("0000", "张三", "男", "20"));
    //      prpDuserDtoList.add(new PrpDuserDto("0001", "李四", "男", "20"));
    //      prpDuserDtoList.add(new PrpDuserDto("0001", "李四", "男", "20"));
    //      prpDuserDtoList.add(new PrpDuserDto("0002", "王五", "男", "20"));
    prpDuserDtoList.add(new PrpDuserDto("0003", "周三", "男", "20"));

    return prpDuserDtoList;
}

```

```
}
```

```
public class PrpDuserDto implements Serializable {
    private String userCode;
    private String userName;
    private String sex;
    private String age;

    /**
     *
     * @param userCode
     * @param userName
     * @param sex
     * @param age
     */
    public PrpDuserDto(String userCode, String userName, String
sex, String age) {
        this.userCode = userCode;
        this.userName = userName;
        this.sex = sex;
        this.age = age;
    }
    ...
}
```

- 图片路径传入，其中 `logoPath` 与模板中定义参数需匹配。

```
paramMap.put("logoPath", "template/images/001.jpg");
```

## 4.4 控制器代码编写

- PrintDemoApi

```
package com.sinosoft.demo.api.printpdf;

import com.sinosoft.demo.api.DemoConstant;
import org.springframework.cloud.netflix.feign.FeignClient;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
```

```

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@FeignClient(name = DemoConstant.DEMO_SERVICE_NAME, path =
PrintDemoApi.PATH)
public interface PrintDemoApi {
    String PATH = "print";

    @RequestMapping(value = "testPrint", method =
RequestMethod.GET)
    public void testPrint(HttpServletRequest request,
HttpServletResponse response) throws Exception;
}

```

- PrintDemoController

```

package com.sinosoft.demo.web.printpdf;

import com.sinosoft.demo.api.customer.CustomerApi;
import com.sinosoft.demo.api.printpdf.PrintDemoApi;
import com.sinosoft.demo.core.printpdf.service.PrintService;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@RestController
@RequestMapping(value = PrintDemoApi.PATH)
public class PrintDemoController implements PrintDemoApi{
    private static final org.slf4j.Logger LOGGER =
LoggerFactory.getLogger(PrintDemoController.class);
    @Autowired
    private PrintService printService;

    @Override
    public void testPrint(HttpServletRequest request,
HttpServletResponse response) throws Exception {
        LOGGER.info("保单打印服务");
        printService.testPrint(request, response);
    }
}

```

```
}
```

## 5 相关规范

---

- 每个应用只能有一个PrintServiceImpl,所有打印均由此类处理。
- 每个功能打印的请求控制器放在各自业务模块中，统一调用打印服务处理。
- pdf模板统一放在rest工程 `resources\template` 下。
- 图片统一放在 `resources\template\images` 下。