## 导出时下载的方法

|  |
| --- |
| //controller层中  HSSFWorkbook wb = iSignArchivesService.savesExcelExport(requestContext, list);  ByteArrayOutputStream outputStream = new ByteArrayOutputStream();  wb.write(outputStream);  wb.close();  ExcelUtil.download(outputStream, response, request, "合同档案.xls");  //download实现方法  /\*\*  \* 下载的方法  \*  \* @param byteArrayOutputStream 流  \* @param response 相应  \* @param request 请求  \* @param returnName 文件名称  \* @throws IOException  \*/  public static void download(ByteArrayOutputStream byteArrayOutputStream, HttpServletResponse response,  HttpServletRequest request, String returnName) throws IOException {  // 处理浏览器差异导致的下载文件名称乱码的问题  String userAgent = request.getHeader("USER-AGENT");  String enocodeType = "";  if (StringUtils.contains(userAgent, "Mozilla")) {// google,火狐浏览器  enocodeType = "iso8859-1";  } else {  enocodeType = "utf8";  }  // response.setContentType("application/octet-stream;charset=utf-8");  // response.setContentType("application/vnd.ms-excel");//application/octet-stream  response.setContentType("application/x-download");  returnName = response.encodeURL(new String(returnName.getBytes(), enocodeType)); // 保存的文件名,必须和页面编码一致,否则乱码  response.addHeader("Content-Disposition", "attachment;filename=" + returnName);  response.setContentLength(byteArrayOutputStream.size());  ServletOutputStream outputstream = response.getOutputStream(); // 取得输出流  byteArrayOutputStream.writeTo(outputstream); // 写到输出流  byteArrayOutputStream.close(); // 关闭  outputstream.flush(); // 刷数据  } |

## 设置下拉框

|  |
| --- |
| if (sheet instanceof XSSFSheet) {  XSSFDataValidationHelper dvHelper = new XSSFDataValidationHelper((XSSFSheet)sheet);  XSSFDataValidationConstraint dvConstraint = (XSSFDataValidationConstraint) dvHelper  .createExplicitListConstraint(explicitListValues);  CellRangeAddressList addressList = new CellRangeAddressList(firstRow, lastRow, firstCol, lastCol);  XSSFDataValidation validation = (XSSFDataValidation) dvHelper.createValidation(dvConstraint, addressList);  validation.setSuppressDropDownArrow(true);  validation.setShowErrorBox(true);  sheet.addValidationData(validation);  } else if(sheet instanceof HSSFSheet){  //找到具体的单元格  CellRangeAddressList addressList = new CellRangeAddressList(firstRow, lastRow, firstCol, lastCol);  //生成下拉框内容  DVConstraint dvConstraint = DVConstraint.createExplicitListConstraint(explicitListValues);  //绑定作用区域和下拉框  DataValidation validation = new HSSFDataValidation(addressList, dvConstraint);  validation.setSuppressDropDownArrow(true);  validation.setShowErrorBox(true);  //设置填写不正确的提示信息  validation.createErrorBox("error",errorStr);  //添加到sheet页中  sheet.addValidationData(validation);  } |

## 导入

### 3.1、导入页面

|  |
| --- |
| <#include "../include/header.html">  <!--add by xiaotong.wu@hand-china.com 2017/11/7-->  <style>  .k-clear-selected{  margin: 0px;  margin-bottom: 10px;  margin-left: 12px;  width: 105px;  }  .k-upload-selected{  margin:0px;  margin-bottom: 10px;  margin-left: 12px;  width: 105px;  }  </style>  <!--add end-->  <body>  <script>  viewModel = kendo.observable({  model : {}  });  </script>  <div id="content-container">  <div id="page-content">  <div class="panel" style="padding:0px;">  <form class="form-horizontal">  <div class="panel-body">  <div class="row">  <input type="file" name="files" id="importExcelFile">  </div>  </div>  <div class="panel-body">  <div cla  ss="row">  <span id="errorMessage"></span>  </div>  </div>  <div class="panel-footer text-right">  <a class="btn btn-success" id="download" onclick="downloadExcel()" type="submit">下载模板</a>  <!--<a class="btn btn-success" id="download" onclick="openAttachment(event)" type="button">下载模板</a>-->  <!--modified end-->  </div>  <!--附件上传-->  <div id="open-attachment"></div>  <!--add end-->  </form>  </div>  </div>  </div>  <script id="fileTemplate" type="text/x-kendo-template">  <div>  <p style="margin:0px;">#=name#&nbsp;&nbsp;&nbsp;#=parseInt(size/1024)#kb</p>  <button type='button' class='k-upload-action' style='position: absolute; top: 8px; right: 0;'></button>  </div>  </script>  <!--add end-->  <script>  var baseUrl = "${base.contextPath}/setup/shipment/price";  $("#importExcelFile").kendoUpload({  template: kendo.template($('#fileTemplate').html()),  async:{  saveUrl: baseUrl+"/upload?${\_csrf.parameterName}=${\_csrf.token}",  autoUpload: false  },  multiple: false,  localization: {  select: "请选择文件",  uploadSelectedFiles: "上传",  headerStatusUploading: "上传中...",  headerStatusUploaded: "文件上传成功!"  },  error:function (e) {  kendo.ui.showInfoDialog({  title: '提示信息',  message: '导入失败!'  });  },  success:function (e) {  if(e.response.success){  kendo.ui.showInfoDialog({  title: '提示信息',  message: '添加成功'  }).done(function (event) {  if (event.button == 'OK') {  window.parent.$("#excelWindow").data("kendoWindow").close();  window.parent.$("#grid").data("kendoGrid").dataSource.page(1);  }  })  }else{  kendo.ui.showErrorDialog({  title: '提示信息',  message: '导入失败!<br/>'  }).done(function (event) {  $("#errorMessage").html('<h5>失败信息:</h5><br/>' + e.response.message)  });  }  }  });  function downloadExcel() {  var form = $("<form>"); //定义一个form表单  form.attr('style', 'display:none'); //在form表单中添加查询参数  form.attr('target', '');  form.attr('method', 'GET');  form.attr('action', baseUrl+"/download");  $('body').append(form); //将表单放置在web中  form.submit();  }  //附件上传  function openAttachment(e) {  var sourceType='CUX\_GXP\_CUS\_IMP';  var openAttachment = $("#open-attachment").kendoWindow({  actions: ["Refresh","Close"],  width: 800,  height: 500,  title: '<@spring.message "hap.attachment\_upload"/>',  visible: false,  iframe: true,  modal: true,  content: 'gxp\_attachment\_create.html?sourceType='+sourceType,  }).data("kendoWindow");  openAttachment.center().open();  }  </script>  </body>  </html> |

### 3.2、后台

|  |
| --- |
| //模板下载  @RequestMapping(value = "/setup/shipment/price/download",method = {RequestMethod.GET})  @ResponseBody  public void printTemplate(HttpServletResponse response,HttpServletRequest request) throws IOException {  XSSFWorkbook wb = new XSSFWorkbook();  XSSFRow nRow = null;  XSSFCellStyle nStyle = wb.createCellStyle();  XSSFFont nFont = wb.createFont();  short color = IndexedColors.GREY\_25\_PERCENT.getIndex();  byte rowNo = 0;  int var15 = rowNo+1;  //设置运费价目表sheet页  XSSFSheet shipemntPriceSheet = wb.createSheet("运费价目表");  Cell nCell = null;  int colNo = 0;  nRow = shipemntPriceSheet.createRow(rowNo);  String [] nameArray = new String[]{"合同意向名称(必输)","产品类别名称(必输)","运输方式名称(必输)","起运地编码(必输)","目的地编码(必输)","内贸/外贸(必输)","运费类型编码(必输)","运费备注","运费结算模式(必输)","运费单价","运费总价","运费结算依据(必输)","允许亏吨率","服务公司编码(必输)","生效日期(必输)","失效日期"};  int [] lengthArray = new int[]{14,14,14,14,14,14,14,20,14,14,20,20,14,14,20,14,14};  for (int os = 0;os<nameArray.length;os++){  shipemntPriceSheet.setColumnWidth(os,lengthArray[os]\*272);  nCell = nRow.createCell(colNo++);  nCell.setCellStyle(ExcelUtil.title(wb,nStyle,nFont,color));  nCell.setCellValue(nameArray[os]);  }  ByteArrayOutputStream var16 = new ByteArrayOutputStream();  wb.write(var16);  ExcelUtil.download(var16,response,request,"运费价目导入模板.xlsx");  }  //导入  @RequestMapping(value = "/setup/shipment/price/upload",method = {RequestMethod.POST})  @ResponseBody  public ResponseData upload(HttpServletRequest request, MultipartFile files) throws IOException {  IRequest iRequest = createRequestContext(request);  String fileName = files.getOriginalFilename();  XSSFWorkbook wb = new XSSFWorkbook(files.getInputStream());  ResponseData responseData = new ResponseData();  ArrayList shipmentPriceExcelList = new ArrayList();  XSSFRow nRow = null;  Cell nCell = null;  byte beginRowNo = 1;//定义拿取数据的起始行0开始  //获得运费价目信息sheet页，并存放在shipementPriceExcelList中  XSSFSheet shipmentPriceSheet = wb.getSheetAt(0); //获得sheet  int endRowNo = shipmentPriceSheet.getLastRowNum(); //获得最后一条数据的行号即有多少行数据  nRow = shipmentPriceSheet.getRow(0); //获得sheet的title  short endColNo = nRow.getLastCellNum(); //根据获得的title，来获得sheet中共有多少列数据  for (int rData = beginRowNo;rData <=endRowNo;rData++){//按照每行每列的顺序像list中存放数据  nRow = shipmentPriceSheet.getRow(rData);  if (nRow==null){  nRow = shipmentPriceSheet.createRow(rData);  }  ArrayList e = new ArrayList();  boolean flagCol = false;//判断第一列数据是否为空  for (int j=0;j<=endColNo;j++){  nCell = nRow.getCell(j);  if (nCell!=null){  switch (nCell.getCellType()){  case HSSFCell.CELL\_TYPE\_NUMERIC:  if (HSSFDateUtil.isCellDateFormatted(nCell)){  double d = nCell.getNumericCellValue();  Date date = HSSFDateUtil.getJavaDate(d);  SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd");  e.add(dateFormat.format(date));  }else {  NumberFormat nf = NumberFormat.getInstance();  nf.setGroupingUsed(false);//true时的格式：1,234,567,890  if ((Double)nCell.getNumericCellValue() != null){//数值类型的数据为double,需转化  e.add(nf.format(nCell.getNumericCellValue()));  }else {  e.add("");  }  }  break;  case HSSFCell.CELL\_TYPE\_STRING:  if (StringUtils.isNotBlank(nCell.getStringCellValue())){  e.add(nCell.getStringCellValue());  }else {  e.add("");  }  break;  default:  e.add("");  }  }else {  e.add("");  }  }  shipmentPriceExcelList.add(e);  }  //清除最后几行全空的数据  clearInvalidUploadData(shipmentPriceExcelList);  ResponseData var2 = new ResponseData();  ResponseData var3;  try {  service.insertSheet(iRequest,shipmentPriceExcelList);  return new ResponseData(true);  } catch (RuntimeException var21) {  var2.setMessage("导入失败！出现异常错误:"+var21.getLocalizedMessage());  var2.setSuccess(false);  var3 = var2;  } catch (Exception var22) {  var22.printStackTrace();  var2.setMessage(var22.getMessage());  var2.setSuccess(false);  var3 = var2;  return var3;  } finally {  wb.close();  }  return var3;  } |