Set Local variables from the parameters IN Parsing file

String[] tokens = tools.getParamValue("TemplateRFB").split("\\s");

if (tokens.length < 3) tokens = tools.getParamValue("TemplateRFB").split("!");

applicationName = tokens[0];

fileName = tokens[1];

fileType = tokens[2];

Parsing File:

boolean bDeleteFile = (new Boolean(tools.getParamValue("bDeleteFile"))).booleanValue();

String validateSheet =tools.getParamValue("ValidateSheet");

String xlFileName = tools.getParamValue("FSFileName");

java.io.InputStream xlIS = null;

java.io.InputStream tempIS = null;

String templateName = fileName;

boolean isValid =true;

boolean isValidFlag = false;

if(templateName.indexOf("Requirement")!=-1){

templateName = "Requirements";

}else if(templateName.indexOf("UseCase")!=-1){

templateName = "UseCases";

}else if(templateName.indexOf("ProjectSizing")!=-1){

templateName = "Project Sizing";

}

else{

}

try {

xlIS = new PRInputStream(xlFileName);

java.util.Map xlZipEntryMap = com.pegarules.generated.pega\_appdefinition\_excelintegration.DCOparseZipFile(xlIS);

if (xlZipEntryMap!=null && xlZipEntryMap.size()>0){

String templateErrMsg = "";

//if this is a Project Sizing template, verify that sheet with name "DataExtract" is present.

if (templateName.equals("Project Sizing")){

org.w3c.dom.NodeList sheets = com.pegarules.generated.pega\_appdefinition\_excelintegration.DCOXLGetWorkbookDoc(xlZipEntryMap).getElementsByTagName("sheet");

boolean hasDataExtr = false;

for (int i=0; i<sheets.getLength(); i++){

String currNode = ((org.w3c.dom.Element)sheets.item(i)).getAttribute("name");

//Adding this for validating Project sizing template sheets

if(validateSheet.equals(currNode))

isValidFlag = true;

if (currNode.equals("DataExtract")) hasDataExtr = true;

}

//if template is not correct for ProjectSizing then set isValid 'false'

if(!isValidFlag){

isValid = false;

if(validateSheet.indexOf("Scrum")!=-1)

templateErrMsg = "ErrorOnXLUploadOfScrumTemplate" ;

else

templateErrMsg = "ErrorOnXLUploadOfPegaBPMTemplate";

tools.getPrimaryPage().addMessage(tools.getLocalizedTextForString("pyMessageLabel",templateErrMsg));

return isValid;

}

validateSheet = "";

if (!hasDataExtr){

oLog.debug("Unreadable template: no DataExtract sheet");

tools.getPrimaryPage().addMessage("pzNoDataExtractSheet");

}

}

byte[] tempBytes = new com.pega.pegarules.pub.util.Base64Util().decodeToByteArray(tools.findPage("TemplateFile").getString("pyFileSource"));

tempIS = new java.io.ByteArrayInputStream(tempBytes);

java.util.Map tempZipEntryMap = com.pegarules.generated.pega\_appdefinition\_excelintegration.DCOparseZipFile(tempIS);

//validating the sheet column headings if the sheet name is provided

if(!validateSheet.equals("")){

isValid=com.pegarules.generated.pega\_appdefinition\_excelintegration.pzDCOXLValidateColumnNames(tempZipEntryMap, xlZipEntryMap,validateSheet);

}

//parse data if the validation passes.

if(isValid)

{

com.pegarules.generated.gscoe\_gscoeutilities.DCOXLParseZipEntryMap(tempZipEntryMap, xlZipEntryMap, tools.getParamValue("strType"));

}else

{

oLog.debug("Sheet "+validateSheet +" Column Header validation failed.");

tools.getPrimaryPage().addMessage("pzUnrecognizedFileTemplate\t" + templateName);

}

}

else{

oLog.debug("Unreadable template: no entries.");

tools.getPrimaryPage().addMessage("pzUnrecognizedFileTemplate\t" + templateName);

}

}

catch (Exception e) {

oLog.debug(e);

oLog.debug("Error in parsing Excel file " + xlFileName, e);

//throw new RuntimeException(e);

tools.getPrimaryPage().addMessage("pzUnrecognizedFileTemplate\t" + templateName);

}

finally {

if (xlIS != null) {

try {

xlIS.close();

}

catch(Exception e) {

}

}

if (tempIS != null) {

try {

tempIS.close();

}

catch (Exception e) {

}

}

try{

if(bDeleteFile) {

PRFile uploadedFile = new PRFile(xlFileName);

if(uploadedFile!=null && uploadedFile.exists()){

uploadedFile.delete();

}

}

}

catch(Exception ex)

{

oLog.error("Error while deleting the file");

}

}