AR Account Clearing

Cloud application

Content

[Overview 2](#_Toc39928465)

[Requirements 2](#_Toc39928466)

[Loading biaPostgreSQL in project 3](#_Toc39928467)

[Creating configuration file 3](#_Toc39928468)

[biaPostgreSQL TestConnection 4](#_Toc39928469)

[biaPostgreSQL GetConnCursor 5](#_Toc39928470)

[biaPostgreSQL dbSelect 6](#_Toc39928471)

[biaPostgreSQL - reading SELECT output 7](#_Toc39928472)

[biaPostgreSQL dbExecute 8](#_Toc39928473)

[biaPostgreSQL dbExecuteMany 9](#_Toc39928474)

[Revision 10](#_Toc39928475)

# Overview

The AR Account Clearing cloud-based application automates the process of:

1.

1. identification of items on the customer accounts based on defined accounting criteria
2. clearing of open items on the debitor accounts
3. closing corresponding disputes in DMS
4. closing open service notifications / completing item tasks in QM02/ Requirements

# Requirements

The AR Account Clearing application utilizes the following external libraries to parse, evaluate, and reason on data:

NumPy 1.2.0 – document and posting date calculation

Pandas 1.2.3 – data parsing, evaluation, extraction

PyWin32 3.00 – connection to SAP GUI scripting engine

PyYaml 5.4.1 – reading .yaml configuration files

XlsxWriter 1.3.9 – writing data to .xlsx report files

# biaSAP SAPlogin

The function is used to connect to a given SAP system. The function returns a Session object. Unless an error occurs while connecting to SAP no output is displayed.

Syntax:

from biaSAP import SAPlogin  
  
SAPlogin(SAPsystem)

Example:

from biaSAP import SAPlogin

SAPsystem = "OG ERP: P25 Productive SSO"

Session = SAPlogin(SAPsystem)  
  
Output:

# biaSAP SAPlogout

The procedure is used to log out of a given SAP session. The function executes “/nex” transaction code to terminate all open sessions. No output is displayed.

Syntax:

from biaSAP import SAPlogout  
  
SAPlogout(Session)

Example:

from biaSAP import SAPlogout

SAPlogout(Session)  
  
Output:

# biaSAP ClosePopUp

The function is used to close all SAP GuiModal windows and return the value of Session.ActiveWindow.PopupDialogText.

Syntax:

from biaSAP import ClosePopUp  
  
ClosePopUp(Session)

Example:

from biaSAP import ClosePopUp

ClosePopUp(Session)  
  
Output:

# biaSAP Converting VBScript to Python

In majority of cases the structure of the SAP script in Python will look as follows:

from biaSAP import SAPlogin

from biaSAP import SAPlogout

SAPsystem = "OG ERP: P25 Productive SSO"

Session = SAPlogin(SAPsystem)

Session.StartTransaction("udm\_dispute")

# SAP Session commands

SAPlogout(Session)  
  
In order to start a transaction, use the following Session method:

Session.StartTransaction(TransactionName)

The SAP GUI Window main containers can be controlled by separate objects if needed:

SAP\_GuiMainWindow = Session.Children(0)

SAP\_GuiMenuBar = Session.Children(0).Children(0)

SAP\_GuiToolBar1 = Session.Children(0).Children(1)

SAP\_GuiToolBar2 = Session.Children(0).Children(3)

SAP\_GuiTitleBar = Session.Children(0).Children(2)

SAP\_GuiUserArea = Session.Children(0).Children(4)

SAP\_GuiStatusBar = Session.Children(0).Children(5)

VBScript recorded Session.FindById commands can be used in Python however method/property names and values need to be updated otherwise they will fail to execute. Example of some VBScript commands and their Python equivalent:

#session.findById("wnd[0]").resizeWorkingPane 173, 36, 0

session.findById("wnd[0]").resizeWorkingPane(173, 36, 0)

#session.findById("wnd[0]/tbar[0]/okcd").text = "/nse16"

session.findById("wnd[0]/tbar[0]/okcd").text = "/nse16"

#session.findById("wnd[0]").sendVKey 0

session.findById("wnd[0]").sendVKey(0)

#session.findById("wnd[0]/usr/ctxtDATABROWSE-TABLENAME").text = "TADIR"

session.findById("wnd[0]/usr/ctxtDATABROWSE-TABLENAME").text = "TADIR"

#session.findById("wnd[0]").sendVKey 0

session.findById("wnd[0]").sendVKey(0)

#session.findById("wnd[0]/tbar[1]/btn[8]").press

session.findById("wnd[0]/tbar[1]/btn[8]").press()

Here is a list of several different Session.FindByID commands in Python:

Session.FindById("wnd[0]/shellcont/shell/shellcont[0]/shell/shellcont[1]/shell/shellcont[1]/shell").selectedNode = " 2"

Session.FindById("wnd[0]/shellcont/shell/shellcont[0]/shell/shellcont[1]/shell/shellcont[1]/shell").doubleClickNode(" 2")

Session.FindById("wnd[0]/usr/cntlCLFRM\_CONTAINER/shellcont/shell/shellcont[1]/shell/shellcont[0]/shell/shellcont[0]/shell/shellcont/shell").setCurrentCell(2, "SEL\_ICON1")

Session.FindById("wnd[0]/usr/cntlCLFRM\_CONTAINER/shellcont/shell/shellcont[1]/shell/shellcont[0]/shell/shellcont[0]/shell/shellcont/shell").pressButtonCurrentCell()

Session.FindById("wnd[1]/usr/tabsTAB\_STRIP/tabpNOSV").select()

Session.FindById("wnd[1]/usr/tabsTAB\_STRIP/tabpNOSV/ssubSCREEN\_HEADER:SAPLALDB:3030/tblSAPLALDBSINGLE\_E/btnRSCSEL\_255-SOP\_E[0,0]").press()

Session.findById("wnd[2]/usr/cntlOPTION\_CONTAINER/shellcont/shell").currentCellColumn = "TEXT"

Session.FindById("wnd[2]/tbar[0]/btn[0]").press()

Session.FindById("wnd[1]/usr/tabsTAB\_STRIP/tabpNOSV/ssubSCREEN\_HEADER:SAPLALDB:3030/tblSAPLALDBSINGLE\_E/ctxtRSCSEL\_255-SLOW\_E[1,0]").text = "003"

Session.FindById("wnd[1]/tbar[0]/btn[8]").press()

Session.FindById("wnd[0]/usr/cntlCLFRM\_CONTAINER/shellcont/shell/shellcont[1]/shell/shellcont[0]/shell/shellcont[0]/shell/shellcont/shell").modifyCell(0, "VALUE2", "2")

Session.FindById("wnd[0]/usr/cntlCLFRM\_CONTAINER/shellcont/shell/shellcont[1]/shell/shellcont[0]/shell/shellcont[0]/shell/shellcont/shell").pressButtonCurrentCell()

Session.FindById("wnd[1]/usr/tabsTAB\_STRIP/tabpSIVA/ssubSCREEN\_HEADER:SAPLALDB:3010/tblSAPLALDBSINGLE/ctxtRSCSEL\_255-SLOW\_I[1,0]").text = "1065131"

Session.FindById("wnd[1]/tbar[0]/btn[8]").press()

Session.FindById("wnd[0]/usr/cntlCLFRM\_CONTAINER/shellcont/shell/shellcont[1]/shell/shellcont[0]/shell/shellcont[0]/shell/shellcont/shell").modifyCell(23, "VALUE1", "99999")

Session.FindById("wnd[0]/usr/cntlCLFRM\_CONTAINER/shellcont/shell/shellcont[1]/shell/shellcont[0]/shell/shellcont[1]/shell").pressButton("DO\_QUERY")

CaseListSAPtable = Session.FindById("wnd[0]/usr/cntlCLFRM\_CONTAINER/shellcont/shell/shellcont[1]/shell/shellcont[1]/shell")

NumOfCases = CaseListSAPtable.rowcount

for y in range(0, NumOfCases):

CaseListSAPtable.firstVisibleRow = y

CaseID = CaseListSAPtable.getCellValue(y, "EXT\_KEY")

CaseListSAPtable.doubleClick(y, "EXT\_KEY")

CaseSAPnotes = Session.FindById("wnd[0]/usr/cntlCLFRM\_CONTAINER/shellcont/shell/shellcont[1]/shell/shellcont/shell/shellcont[2]/shell/shellcont[2]/shell/shellcont[1]/shell/shellcont[0]/shell").text

CaseSAPnotes = CaseSAPnotes.replace("\r", "\n")

# biaSAP Watchouts

Here is a list of issues that may arise if not properly handled:

* Multiline free text fields (TextBox/TextArea) in SAP use \r to define new line whilst Python uses \n. It is necessary to make replacement of all \r to \n before any processing of such text.

CaseSAPnotes = Session.FindById("wnd[0]/usr/cntlCLFRM\_CONTAINER/shellcont/shell/shellcont[1]/shell/shellcont/shell/shellcont[2]/shell/shellcont[2]/shell/shellcont[1]/shell/shellcont[0]/shell").text

CaseSAPnotes = CaseSAPnotes.replace("\r", "\n")

* It is recommended to start names all Session.FindByID object methods/properties with lower-case. It’s been proven that some objects methods/properties cannot be find if their name begins with an upper-case letter.  
    
  This will work:

Session.FindById("wnd[1]/usr/tabsTAB\_STRIP/tabpSIVA/ssubSCREEN\_HEADER:SAPLALDB:3010/tblSAPLALDBSINGLE/ctxtRSCSEL\_255-SLOW\_I[1,0]").text = "1065131"

This will not work:

Session.FindById("wnd[1]/usr/tabsTAB\_STRIP/tabpSIVA/ssubSCREEN\_HEADER:SAPLALDB:3010/tblSAPLALDBSINGLE/ctxtRSCSEL\_255-SLOW\_I[1,0]").Text = "1065131"

* Names of SAP object methods in Python always end with ()

# Revision

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Description** |
| 1.0 | 14. 5. 2021 | Dušan Paál | Initial version |