Versteckte User-Exits Invoice-Jobs

Invoice_JOB_StartWorkflow

Bei der Übergabe von Squeeze an den Documents-Workflow kann es zu Problemen mit der Anzahl der erlaubten SOAP-Sessions kommen. Bei einer Überschreitung der erlaubten Sessions (im Standard sind es 3) werden alle Sessions abgebrochen. Wenn ein Workflow bereits gestartet wurde, dann fehlen im Workflow-Verlauf abrupt die Zugriffsberechtigungen. Die Workflow-Skripte lassen den Vorgang in einen Fehler-Status laufen. durch die negative Rückmeldung werden Belege teils mehrfach übergeben.
Bei der Anlage einer neuen Mappe wurde bislang immer direkt der Workflow gestartet. Das führt allerdings dazu, dass über die SOAP-Session nicht nur der Beleg erzeugt wird, sondern auch der Workflow gestartet wird. Je mehr Zeit eine Session benötigt, desto größer ist das Risiko, dass die Sessions abbrechen und es zu Fehlern kommt. Als Lösung wurde bereits eine initiale Verzögerung in den Workflow eingebaut.
Um noch mehr Zeit zu sparen kann der Workflow auch nachgelagert über diesen Job gestartet werden. Das Skript setzt context.superMode(true) und sucht via FileResultset nach allen Rechnungen mit leerer WorkflowID. In einer Schleife wird zunächst nochmal das Attribut "InCirculation" an der Mappe überprüft.

```
/** Diese optionale Funktion wird pro Mappe ausgeführt.
 * Wenn der Rückgabewert ungleich true ist wird der Workflow nicht gestartet!
 * Das UserExit wird auch über die Skripte "Invoice_Action_StartWorkflow" und
"Invoice_Action_Folder_UDA_StartWorkflow" ausgeführt.
 * Diese werden im Kontext eines Benutzers ausgeführt und in den Fällen muss bei Fehlern
context.errorMessage gesetzt werden,
 * damit der Anwender eine Fehlermeldung erhält.
 * @param {Log} log Logging-Objekt * @returns {boolean} true / false (Workflow wird nicht
gestartet)
 * @since Invoice 1.1.015
**/
function ue_beforeStartWorkflow(docFile, log){
log.info("["+docFile.getid()+"] function ue_beforeStartWorkflow(docFile, log)");
return true;
}
```

Invoice_JOB_CheckPostingStatus

Der Job überprüft den Buchungs-Status (Spalte "**PostingStatus**") in der Tabelle "**Invoice_PostingHead**" für alle Rechnungs-Mappen mit der Aktions-ID "**Posting**" und dem Aktions-Status "**TechActionJob**". Aus der Tabelle werden folgende Feldwerte in die Mappe übernommen:

- PostingStatus
- PostingNumber
- PostingPeriod
- PostingDate
- PostingUser
- PostingError

Bei einem Fehler-Status ("**error**") wird der Beleg zur im Feld "**ActionAccessProfile**" hinterlegten Gruppe zur Nachbearbeitung gesendet. Technisch kann in das Feld "ActionAccessProfile" auch ein Benutzer-Login geschrieben werden. Bei erfolgter Buchung (Status "**posted**") wird der Beleg zur nächsten Workflow-Aktion gesendet.

Im Skript wird zunächst das FileResultset ermittelt. Erst wenn mindestens 1 Beleg gefunden wurde wird ein Log erzeugt und erst dann werden auch die folgenden User-Exit-Funktion ausgeführt. Vor dem FileResultset wird "**context.setSuperMode(true)**" gesetzt. Alle User-Exits werden im Super-Mode ausgeführt! Der Job importiert die "**Invoice__ImportLib**". Die UserExits können demnach zur "**Invoice__UserExit_CustomLib**" hinzugefügt werden.

```
/** Diese optionale Funktion wird noch vor der FileResultset-Schleife ausgeführt und kann
verwendet werden,
 * um zum Beispiel den Buchungs-Status aus einem externen System zu übertragen.
 * @param {Log} log Logging-Objekt
 * @returns {} Die Rückgabe wird nicht ausgewertet
 * @since Invoice 1.1.015
**/
function ue_checkPostingStatus_BeforeLoop(log){
  log.info("["+docFile.getid()+"] function ue_checkPostingStatus_BeforeLoop(log)");
}

/** Diese optionale Funktion wird direkt zu Beginn in der Schleife ausgeführt.
 * @param {DocFile} docFile Aktuelle Rechnungs-Mappe im FileResultset
 * @param {Log} log Logging-Objekt
 * @returns {boolean} true / false (durch false wird die Überprüfung der Mappe abgebrochen)
 * @since Invoice 1.1.015
```

```
**/
function ue_checkPostingStatus_DocFileStart(docFile, log){
  log.info("["+docFile.getid()+"] function ue_checkPostingStatus_DocFileStart(docFile, log)");
  return true;
}

/** Diese optionale Funktion wird nur ausgeführt, wenn der Buchungsstatus in der Tabelle "error" zurückliefert.
 * Die Feldwerte werden gesetzt aber die Mappe wurde noch nicht weitergeleitet.
 * @param {DocFile} docFile Aktuelle Rechnungs-Mappe im FileResultset
 * @param {Log} log Logging-Objekt
 * @returns {boolean} true / false (durch false wird die Mappe nicht an die Gruppe zur Überprüfung weitergeleitet)
 * @since Invoice 1.1.015
**/
function ue_checkPostingStatus_OnPostingError(docFile, log){
  log.info("["+docFile.getid()+"] function ue_checkPostingStatus_OnPostingError(docFile, log)");
  return true;
}

/** Diese optionale Funktion wird nur ausgeführt, wenn der Buchungsstatus in der Tabelle "posted" zurückliefert.
 * Die Feldwerte werden gesetzt aber die Mappe wurde noch nicht weitergeleitet.
 * @param {DocFile} docFile Aktuelle Rechnungs-Mappe im FileResultset
 * @param {Log} log Logging-Objekt
 * @returns {boolean} true / false (durch false wird die Mappe nicht zur nächsten Workflow-Aktion weitergeleitet)
 * @since Invoice 1.1.015
**/
function ue_checkPostingStatus_Posted(docFile, log){
  log.info("["+docFile.getid()+"] function ue_checkPostingStatus_Posted(docFile, log)");
  return true;
}

/** Diese optionale Funktion wird nur ausgeführt, wenn der Buchungsstatus in der Tabelle "posted" zurückliefert.
 * Die Feldwerte werden gesetzt und die Mappe erfolgreich weitergeleitet.
 * @param {DocFile} docFile Aktuelle Rechnungs-Mappe im FileResultset
```

```
 * @param {Log} log Logging-Objekt
 * @returns {} Die Rückgabe wird nicht ausgewertet
 * @since Invoice 1.1.015
**/
function ue_checkPostingStatus_PostedAndForwarded(docFile, log){
  log.info("["+docFile.getid()+"] function ue_checkPostingStatus_PostedAndForwarded(docFile, log)");
  return true;
}
```

Invoice_JOB_Archiving

Der Job archiviert die Belege, die eine der folgenden Aktions ID haben:

- Archiving, Archiving1, Archiving2, Archiving3, ... Archiving14, Archiving15

```
/** Diese Funktion archiviert die Rechnungen.
 * Die Funktion wird bei allen Archivierungsvorgängen - auch beim Aussteuern - aufgerufen.
 * Die Funktion wird in der DEXPRO__USerExit_TechActionLib ausgeliefert.
 * @param {string} type Über den Aufruf aus diesem Job wird immer "Job" mitgegeben.
 * @param {DocFile} docFile Aktuelle Rechnungs-Mappe im FileResultset
 * @param {Log} log Optionales Logging-Objekt
 * @returns {boolean} true (Archiviert) / false (Fehler)
**/
function ue_Archiving(type, docFile, log){
  log.info("["+docFile.getid()+"] function ue_Archiving("+type+", docFile, log)");
  if( docFile.archive() ){
    log.info("["+docFile.getid()+"] archived("+docFile.getArchiveKey()+")");
return true;
  }
  log.err("["+docFile.getid()+"] " + docFile.getLastError();
  return false;
}
```

Revision #13
Created 10 August 2022 07:35:34 by Markus Meisner
Updated 12 August 2022 12:11:22 by Markus Meisner