Barcodetrennung

In folgendem Ordner der Squeeze Installation:
SQUEEZE\htdocs\repository\<Mandantenname>\UserExits\Global\

muss die PHP-Datei BarcodeSeparation.php liegen.

Unter Barcode Types sind die möglichen Barcode-Typen aufgeführt die verwendet werden können.

In diesem Beispiel: CODE_128

Darüber hinaus kann der zu berücksichtigende Barcode-Wert weiter eingeschränkt werden:

In diesem Beispiel muss der Barcode 12 Stellen haben und die ersten 4 Stellen müssen "DTER" entsprechen.

```
<?php

use Squeeze\xDataBaseClass;
use Squeeze\xDoc;
use Squeeze\xReturnObject;

/**
 * @param xDoc $xDoc
 * @return xReturnObject
 * @throws Exception
 */
function BarcodeSeparation(xDoc $xDoc, string $nextStep, string $newStatus): xReturnObject
{
    $logger = Logger::getLogger("main");
    $db = xDataBaseClass::getDBConnection();

    $logger->debug('BatchClassId    = ' . $xDoc->batchClassId);
    $logger->debug('DocumentClassId = ' . $xDoc->docClassId);

    if ($xDoc->batchClassId == '1') {
        $result = splitByBarcode($db, $xDoc, $logger);
    }
```

```
    return $result;
}

/**
 * @param PDO $db
 * @param xDoc $xDoc
 * @param Logger $logger
 * @return xReturnObject
 * @throws Exception
 */
function splitByBarcode(PDO $db, xDoc $xDoc, Logger $logger): xReturnObject
{
    // ========================================
    // Barcode Types
    // ========================================
    //  AZTEC
    //  CODABAR
    //  CODE_39
    //  CODE_93
    //  CODE_128
    //  COMPOSITE
    //  DATABAR
    //  DATA_MATRIX
    //  DATABAR_EXP
    //  EAN_2
    //  EAN_5
    //  EAN_8
    //  EAN_13
    //  ITF
    //  ISBN_10
    //  ISBN_13
    //  MAXICODE
    //  PDF_417
    //  QR_CODE
    //  RSS_14
    //  RSS_EXPANDED
    //  UPC_A
    //  UPC_E
    //  UPC_EAN_EXTENSION
    //  UNKNOWN
```

```
    $currentSplit = 0;
    $splits = [];

    // Define which field values to keep
    if ($xDoc->getDocumentFieldByName('ScanUser') !== null) {
        $keepFieldValues[] = ['name' => 'ScanUser', 'value' => $xDoc->getDocumentFieldByName('ScanUser')->getValue()->value];
    }
    if ($xDoc->getDocumentFieldByName('ScanUser') !== null) {
        $keepFieldValues[] = ['name' => 'DocumentType', 'value' => $xDoc->getDocumentFieldByName('DocumentType')->getValue()->value];
    }

    foreach ($xDoc->barcodes as $page => $barcodes) {
        $barcodeType = '';
        $barcodeValue = '';
        $suppressOCR = 0;
        $logger->debug('Checking Barcodes for Page ' . $page);

        if (count($barcodes) == 0) {
            $logger->debug('No Barcode on Page ' . $page);
            if ($page == 1) {
                $currentSplit = $page;
            }
        } else {
            foreach ($barcodes as $key => $barcode) {

                // Suppress OCR if Zero Barcode was found
                //if($barcode['type'] == 'ITF' && $barcode['value'] == '999999'){
                //    $suppressOCR = 1;
                //}

                //$logger->debug($barcode['type']);
                //$logger->debug(substr($barcode['value'], 0, 3));

                if ($barcode['type'] == 'CODE_128' && mb_strlen($barcode['value']) == 12 && substr($barcode['value'], 0, 4) == 'DTER') {
                    $logger->debug('Start Split here! Barcode (' . $barcode['value'] . ') of
```

```
type (' . $barcode['type'] . ') found on page ' . $page);
                    $currentSplit = $page;
                    $barcodeType = $barcode['type'];
                    $barcodeValue = $barcode['value'];
                }
□□□□//elseif($barcode['type'] == 'ITF' && mb_strlen($barcode['value']) == 8 &&
substr($barcode['value'], 0, 1) == '4'){
                //    $logger->debug('Start Split here! Barcode (' . $barcode['value'] . ')
of type (' . $barcode['type'] . ') found on page '. $page);
                //    $currentSplit = $page;
                //    $barcodeType = $barcode['type'];
                //    $barcodeValue = $barcode['value'];
                //}
            }
        }

        // add page
        $splits[$currentSplit][] = ['page' => $page, 'type' => $barcodeType, 'value' =>
$barcodeValue, 'fields' => $keepFieldValues, 'suppressOcr' => $suppressOCR];
    }

    return new xReturnObject(true, 200, 'Barcode Split', $splits);
}
```

Ob die Trennung funktioniert, sieht man wenn in der Bildaufbereitung aus einem Dokument mehrere werden und im Squeeze.log.

Die oben aufgeführten Barcode Typen sind auch die möglichen Werte für Attachment Barcode Typen an der Dokumentenklasse. Wichtig: Der Wert für den Barcode Typ ist case-sensitiv (Groß-Kleinschreibung beachten!).

Ob die Erkennung des Attachment Barcodes funktioniert hat, sieht man daran das ab diesem Barcode kein Text mehr im Viewer markiert werden kann.

Revision #6
Created 27 March 2020 14:24:56 by Achim Redmann
Updated 1 August 2023 11:49:00 by Ulrich Eckhardt