2018 Tax Sale Properties

Notes: This python scripts scrapes a 2018 Tax Sale Properties List in PDF form and creates a csv output file. The following column headers are created and the associated data is collected: **Item Number, Map Number, and Total Tax Due**. The PDF filename is not unique.

PDF

	A	B	C	D	E
1	*2018 TAX SALE PROPERTIES AVAILABLE FOR ASSIGNMENT*				
2	*Steps and guidelines in accepting an assignment of property from the Forfeited Land Commission.*				
3	***Steps:***				
4	*1. Register as a 2018 Tax Sale bidder with the County Tax Collector's office. (Contact 864-596-2597)*				
5	*2. Then contact the Forfeited Land Commission in order to receive an assignment form. (864-562-4273)*				
6	*3. With Assignment Form, remit bid payment in the form of a certified check or cash to Tax Collector.*				
7	***Remember the guidelines for receiving an assignment of property:***				
8	*1. Assignments are prepared on a "first come, first served" basis. The first bidder to present their bidder number and correct form of payment will be assigned a property.*				
9	*2. Once a bidder is assigned a property, then the bidder must adhere to all guidelines that are required by the Tax Collector's office during the redemption period.*				
10	*3. Until the deed is issued conveying the property, the bidder has no rights or access to the properties avaialble for assignment.*				19 PDF 7 26
11					
12	***ITEM #***	**DESCRIPTION**	**DEFAULTING TAXPAYER**	**MAP NUMBER**	**TOTAL TAX DUE**
13	**Item Number**	best known property address	**Owner Name**		**Bid Amount needed**
14	91320	MELVIN HILL RD	QUICK JOSHUA H	1-11-00-015.05	1,363.54
15	86659	HANNON RD	GAINES JERRY A	1-42-00-175.31	206.61
16	90645	26 LADSON ST	OSBY SAM	1-44-02-022.00	2,321.46
17	84746	LADSON ST	CAPERS HENRIETTA FOWLER	1-44-02-101.00	1,596.12
18	86401	BOBO ST	FOLK RACHEL M ETAL	1-44-03-044.00	2,918.23
19	89721	MCDOWELL RD	MCDOWELL JOHNNY R &	2-07-00-107.11	505.94
20	83227	620 REDLINE AVE	ALLIED MORTGAGE OF SOUTH CAROLINA INC	2-14-00-008.00	1,602.51
21	87554	703 BERKELEY ST	HEDDEN STEVEN M	2-14-05-034.00	1,477.33
22	89637	FAIRFIELD ST	MCCLELLAN PHILIP	2-14-09-028.01	1,082.53
23	84496	FAIRFIELD ST	BULSA FELIX	2-14-09-030.00	787.74
24	84497	LEXINGTON ST	BULSA FELIX	2-14-09-088.00	468.17
25	84498	LEXINGTON ST	BULSA FELIX	2-14-09-089.00	468.17
26	85714	RICHLAND ST	DEACONS OF GASSEMBLY BAPTIST CHURCH	2-14-13-003.09	1,655.68
27	86612	404 S FLORIDA AVE	FRIEDMAN ROBERT S JR	2-14-13-061.00	10,652.60
28	87350	JOHNSON CIR	HANDY THOMAS F	2-30-07-034.00	619.77
29	84461	135 MAIN ST	BRYSON CHRISTINE ETAL	2-33-07-062.00	11,374.75
30	90318	TWIN CREEK DR	MOSS ANN P & RICHARD ELLIOTT	2-44-00-040.05	2,831.54
31	90768	SPARROW CT	PANORAMA ESTATES INC	2-44-00-430.00	758.62
32	94184	202 S LINDA ST	WINKLER CORA O	3-10-15-065.00	9,178.02
33	92552	140 PHIFER LINE	SMITH STEVE-CUSTODIAN	3-14-00-076.00	3,612.08
34	90801	241 STONE HILL DR	PARK NICHOLE	3-18-05-021.00	2,314.38

Python Code (20190822_TaxSaleList.py)

```
# Read a 2018 Delinquent Tax List of Real Property PDF and extract

# Tax Map Number and Total Tax Due and dump data into a CSV file

# Accumatch

# by Michael Keller

# Aug 22, 2019

# requires pdfplumber, re, datetime, mwkFunctions


# Libraries

from mwkFunctions import getFiles, printFilename, printNumberOfPages, openFile, writeFile,
stripAmount

import pdfplumber

import re

from datetime import datetime


# Start Timer

start = datetime.now()


# Booleans

##lineStop = False

##lineStart = False

##TaxMap_Found = False

##TotalTax_Found = False

dataFound = False

ItemNumber_Found = False

MapNumber_Found = False

TaxDue_Found = False



# Variables

fn = '2018 Tax Sale FLC Properties 14TH RUN PDF_201907261517124584.pdf'

messageEOL = '. . . end of line'

comma = ','

count = 0

outString = []

outFilename = 'TaxSaleProperties'

outputType = '.csv'
```

```
dataLineStart = 'Item Number'
dataLineStop = 'A B C D E'
colHeadings = 'Item Number,Tax Map Number,Total Tax Due'
ItemNumber = ''
MapNumber = ''
TaxDue = ''

# Regular Expressions
regItemNumber = '\d{5}'
regMapNumber = '\d{1}\D{1}\d{2}\D{1}\d{2}\D{1}\d{3}[.]\d{2}'
regTaxDue = "[ ][,0-9]+['.']\d{2}"

# PDF File Open
print(printFilename(fn))
pdf = pdfplumber.open(fn)

lastPage = len(pdf.pages)
print(printNumberOfPages(lastPage))

fileName = outFilename + outputType
openFile(fileName,colHeadings)

for page in pdf.pages:
    p1 = pdf.pages[count]
    pltext = p1.extract_text()
    text = pltext.splitlines()

##    Break for Testing
##    if(count == 2):
##        break

    for item in text:
        blob = item.strip()

        if(dataFound == True):
            if(blob != dataLineStop):
```

```
                ItemNumber_Found = re.search(regItemNumber,blob)
                if not (ItemNumber_Found is None):
                    ItemNumber = blob[ItemNumber_Found.start():ItemNumber_Found.end()]
                MapNumber_Found = re.search(regMapNumber,blob)
                if not (MapNumber_Found is None):
                    MapNumber = blob[MapNumber_Found.start():MapNumber_Found.end()]
                TaxDue_Found = re.search(regTaxDue,blob)
                if not (TaxDue_Found is None):
                    TaxDue = blob[TaxDue_Found.start():TaxDue_Found.end()]
                    TaxDue = stripAmount(TaxDue,True)
                outString.append(ItemNumber + comma + MapNumber + comma + TaxDue)

        if(dataLineStart in blob):
            dataFound = True

    count = count + 1

## Close PDF
pdf.close()

## Write CSV file
writeFile(fileName,outString)

#end timer
print('Runtime: ' + str(datetime.now() - start))

print(messageEOL)

# EOF
```

Python Code (mwkFunctions.py)

```
## Michael Keller's function module
import os
import re

# Local Variables
_return = '\r'

def getCurrentDirectory():
    os.chdir(os.path.dirname(__file__))
    return os.getcwd()

def getFiles(ext):
    for(dirpath,dirnames,filenames) in os.walk(getCurrentDirectory()):
        return (f for f in filenames if f.endswith(ext))

def printFilename(fn):
    return 'Reading ' + fn + ' file . . .'

def printNumberOfPages(np):
    return 'Number of Pages: ' + str(np)

def readFile(fn):
    # Read existing file
    try:
        file = open(fn,'r')
        print('Opening ' + fn + ' file . . .')
    except FileNotFoundError:
        sys.exit('File: ' + fn + ' was not found')
    except PermissionError:
        sys.exit('Unable able to open file for reading. Do you have permission to open file?')

def openFile(fn,ch):
    # Create output file
    try:
        file = open(fn,'w+')
```

```
        # Write out lines from array
        file.write(ch + _return)
        file.close()
        print('Opening ' + fn + ' file . . .')
    except FileNotFoundError:
        sys.exit('File: ' + fn + ' does not exist')
    except PermissionError:
        sys.exit('Unable able to open output file for writing. Is the file currently open?')

def writeFile(fn,os):
    # Create and append to output file
    try:
        file = open(fn,'a+')
        for taxLine in os:
            file.write(taxLine + _return)
        file.close()
        print('Saving ' + fn + ' file . . .')
    except FileNotFoundError:
        sys.exit('File: ' + fn + ' does not exist')
    except PermissionError:
        sys.exit('Unable able to open output file for writing. Is the file currently open?')

def stripAmount(amt,zero):
    ## Strip dollar sign, commas, and blanks from value
    ## if nothing left after stripping, change value to 0.00
    ## if zero is True, all amounts that are either blank, have
    ## dashes, do not have values, etc. will be converted to '0.00'
    regAmount = "[,0-9]+['.']\d{2}"
    comma = ','
    dollarSign = '$'
    space = ' '
    nospace = ''
    Zero_Value = False

    amt = amt.strip()
    amt = amt.replace(comma,nospace)
```

```
    amt = amt.replace(dollarSign,nospace)
    amt = amt.replace(space,nospace)
    if(zero == True):
        Zero_Value = re.search(regAmount,amt)
        if ((Zero_Value is None) or (len(amt) == 0)):
            amt = '0.00'
    return amt


# EOF
```

CSV Output

	A	B	C
1	Item Number	Tax Map Number	Total Tax Due
2	91320	1-11-00-015.05	1363.54
3	86659	1-42-00-175.31	206.61
4	90645	1-44-02-022.00	2321.46
5	84746	1-44-02-101.00	1596.12
6	86401	1-44-03-044.00	2918.23
7	89721	2-07-00-107.11	505.94
8	83227	2-14-00-008.00	1602.51
9	87554	2-14-05-034.00	1477.33
10	89637	2-14-09-028.01	1082.53
11	84496	2-14-09-030.00	787.74
12	84497	2-14-09-088.00	468.17
13	84498	2-14-09-089.00	468.17
14	85714	2-14-13-003.09	1655.68
15	86612	2-14-13-061.00	10652.6
16	87350	2-30-07-034.00	619.77
17	84461	2-33-07-062.00	11374.75
18	90318	2-44-00-040.05	2831.54
19	90768	2-44-00-430.00	758.62
20	94184	3-10-15-065.00	9178.02
21	92552	3-14-00-076.00	3612.08
22	90801	3-18-05-021.00	2314.38
23	87426	3-20-00-037.01	647.35
24	88739	3-26-13-058.00	1919.43
25	88295	3-29-00-027.03	1038.52
26	87706	3-29-15-020.00	5140.49
27	90884	3-33-00-056.12	402.19
28	88298	3-33-04-026.01	2984.23

Michael Keller
Accumatch
20190822_TaxSaleList
Aug 23, 2019