```
import os
from reportlab.platypus import SimpleDocTemplate, Paragraph, Spacer
from reportlab.lib.styles import getSampleStyleSheet, ParagraphStyle
#from reportlab.lib.pagesizes import a4
from reportlab.lib import colors
from html import escape

from pygments import lex
from pygments.lexers import PythonLexer
from pygments.token import Token

def convert_py_to_pdf_styled(py_file_path, pdf_file_path):
"""
Converts a Python script to a PDF file with IDLE-like syntax highlighting,
line numbers, and background color on an A4 page.

This version correctly applies colors to individual syntax elements.

:param py_file_path: Path to the input .py file.
:param pdf_file_path: Path to the output .pdf file.
"""
if not os.path.exists(py_file_path):
print(f"Error: The file {py_file_path} was not found.")
return

# --- Define IDLE Theme (Customize these colors to match your setup) ---
style_map = {
Token.Keyword: 'orange',
Token.Name.Builtin: 'purple',
Token.Name.Function: 'blue',
Token.Name.Class: 'darkblue',
Token.String: 'green',
Token.Comment: 'red',
Token.Operator: 'black',
Token.Number: 'darkcyan',
'DEFAULT': 'black', # Default text color # Change Text Color
}

# --- ReportLab PDF Setup ---
doc = SimpleDocTemplate(pdf_file_path, #pagesize=a4,
topMargin=50, bottomMargin=50, leftMargin=50, rightMargin=50)

# Custom ParagraphStyle for the code
styles = getSampleStyleSheet()
code_style = ParagraphStyle(
'Code',
parent=styles['Normal'],
fontName='Courier', # Monospaced font
fontSize=9, # Change fontsize
leading=12,
firstLineIndent=0,
leftIndent=0,
)

# --- Read and Process the Python File ---
with open(py_file_path, 'r') as f:
code_lines = f.readlines()

story = []
line_number = 1
```

```
61
62 for line in code_lines:
63 # Create the styled text for the line
64 styled_line = ''
65 # Use pygments to break the line into tokens
66 lexer = PythonLexer()
67 tokens = lex(line, lexer)
68
69 for token_type, token_value in tokens:
70 # Escape HTML special characters to prevent errors in ReportLab's parser
71 escaped_value = escape(token_value)
72
73 # Find the color for the token type, defaulting to 'DEFAULT'
74 color = 'black' # Start with default
75 while token_type not in style_map:
76 token_type = token_type.parent
77 if token_type is None:
78 color = style_map['DEFAULT']
79 break
80 else:
81 color = style_map[token_type]
82
83 styled_line += f'<font color="{color}">{escaped_value}</font>'
84
85 # Prepend the line number with fixed-width formatting
86 line_num_str = f'{line_number: >4} '
87
88 # Combine line number and the styled code
89 full_line_text = f'<font color="gray">{line_num_str}</font>{styled_line}'
90
91 # Create a Paragraph and add it to the story. ReportLab handles wrapping.
92 p = Paragraph(full_line_text, code_style)
93 story.append(p)
94
95 line_number += 1
96
97 # --- Build the PDF ---
98 # To set a background color, we need to draw it on the canvas manually.
99 def on_first_page(canvas, doc):
100 canvas.saveState()
101 # Set your desired IDLE background color here
102 canvas.setFillColor(colors.HexColor('#FEFDFD')) # A slightly off-white like default
IDLE
103 #canvas.setFillColor(colors.HexColor('#FFFFF0')) # Change Background color: Very
light warm white
104 #canvas.setFillColor(colors.HexColor('#1E1E1E')) # Change Background color: Dark
105 canvas.rect(0, 0, doc.width + doc.leftMargin * 2, doc.height + doc.bottomMargin * 2,
fill=1, stroke=0)
106 canvas.restoreState()
107
108 doc.build(story, onFirstPage=on_first_page, onLaterPages=on_first_page)
109 print(f"Successfully converted {py_file_path} to {pdf_file_path}")
110
111
112 if __name__ == '__main__':
113 # 1. Set the name of the Python file you want to convert.
114 # Make sure this file is in the same directory as this script.
115 input_python_file = "PyToPdf6_Working.py"
116
117 # 2. Set the desired name for your output PDF file.
```

```
118 output_pdf_file = "PDF_Out.pdf"
119
120 # 3. Run the script. It will read the input file and create the PDF.
121 convert_py_to_pdf_styled(input_python_file, output_pdf_file)
```