pText now support codeblock style paragraphs:

```
import logging
import unittest
from pathlib import Path

from ptext.io.read.types import Decimal
from ptext.pdf.canvas.color.color import X11Color
from ptext.pdf.canvas.layout.codeblock import CodeBlock
from ptext.pdf.canvas.layout.page_layout import SingleColumnLayout
from ptext.pdf.canvas.layout.paragraph import (
    Paragraph,
)
from ptext.pdf.document import Document
from ptext.pdf.page.page import Page
from ptext.pdf.pdf import PDF
from tests.util import get_output_dir, get_log_dir

logging.basicConfig(
    filename=Path(get_log_dir(), "test-write-codeblock.log"),
    level=logging.DEBUG,
)


class TestWriteCodeblock(unittest.TestCase):
    def __init__(self, methodName="runTest"):
        super().__init__(methodName)
        self.output_dir = Path(get_output_dir(), "test-write-codeblock")

    def test_write_document(self):

        # create output directory if it does not exist yet
        if not self.output_dir.exists():
            self.output_dir.mkdir()

        # create document
        pdf = Document()

        # add page
        page = Page()
        pdf.append_page(page)

        # layout
        layout = SingleColumnLayout(page)

        layout.add(
            Paragraph(
                "pText now support codeblock style paragraphs:",
                font_color=X11Color("YellowGreen"),
                font_size=Decimal(20),
            )
        )

        # read self
        with open(__file__, "r") as self_file_handle:
            file_contents = self_file_handle.read()

        layout.add(
            CodeBlock(
                file_contents,
                font_size=Decimal(5),
            )
        )

        layout.add(
            Paragraph(
                "By default, these LayoutElements are first formatted by black. "
                "The font is Courier, and the background and font_color are adjusted as well.",
                font_size=Decimal(8),
            )
        )

        # determine output location
        out_file = self.output_dir / "output.pdf"

        # attempt to store PDF
        with open(out_file, "wb") as in_file_handle:
            PDF.dumps(in_file_handle, pdf)
```

By default, these LayoutElements are first formatted by black. The font is Courier, and the background and font_color are adjusted as well.