```python
# --- Das Textfeld mit den Zeilennummern stammt von eyllanesc mit leichten VerÄ¤nderungen von mir. --- #
# --- https://stackoverflow.com/questions/50074155/how-to-add-line-number-in-this-texteditor --- #
# --- MIT-Lizenz (frei Benutzung) --- #


from PyQt5.QtWidgets import QWidget, QPlainTextEdit, QTextEdit
from PyQt5.QtGui import QColor, QTextFormat, QPainter, QFont
from PyQt5.QtCore import QRect, pyqtSlot, Qt, QSize


class LineNumberArea(QWidget):
    def __init__(self, editor):
        QWidget.__init__(self, parent=editor)
        self.codeEditor = editor

    def sizeHint(self):
        return QSize(self.codeEditor.line_number_area_width(), 0)

    def paintEvent(self, event):
        self.codeEditor.line_number_area_paint_event(event)


class CodeEditor(QPlainTextEdit):
    def __init__(self, parent=None):
        QPlainTextEdit.__init__(self, parent)
        f = QFont("Consolas", 14)
        f.setBold(True)
        self.setFont(f)
        self.lineNumberArea = LineNumberArea(self)
        self.blockCountChanged.connect(self.updateline_number_area_width)
        self.updateRequest.connect(self.update_line_number_area)
        self.cursorPositionChanged.connect(self.highlight_current_line)
        self.updateline_number_area_width(0)
        self.highlight_current_line()

    def line_number_area_paint_event(self, event):
        painter = QPainter(self.lineNumberArea)
        painter.fillRect(event.rect(), Qt.lightGray)

        block = self.firstVisibleBlock()
        block_number = block.blockNumber()
        top = self.blockBoundingGeometry(block).translated(self.contentOffset()).top()
        bottom = top + self.blockBoundingRect(block).height()
        while block.isValid() and top <= event.rect().bottom():
            if block.isVisible() and bottom >= event.rect().top():
                number = str(block_number + 1)
                painter.setPen(Qt.black)
```

```
48              painter.drawText(-2, top, self.lineNumberArea.width(),
49                               self.fontMetrics().height(),
50                               Qt.AlignRight, number)
51          block = block.next()
52          top = bottom
53          bottom = top + self.blockBoundingRect(block).height()
54          block_number += 1
55
56      def line_number_area_width(self):
57          digits = len(str(self.blockCount()))
58          space = 3 + self.fontMetrics().width("9") * digits
59          return space
60
61      def resizeEvent(self, event):
62          QPlainTextEdit.resizeEvent(self, event)
63          cr = self.contentsRect()
64          self.lineNumberArea.setGeometry(QRect(cr.left(), cr.top(), self.line_number_area_width(), cr.height()))
65
66      @pyqtSlot(int)
67      def updateline_number_area_width(self, new_block_count):
68          self.setViewportMargins(self.line_number_area_width(), 0, 0, 0)
69
70      @pyqtSlot()
71      def highlight_current_line(self):
72          extra_selections = []
73          selection = QTextEdit.ExtraSelection()
74          line_color = QColor(Qt.green).lighter(170)
75          selection.format.setBackground(line_color)
76          selection.format.setProperty(QTextFormat.FullWidthSelection, True)
77          selection.cursor = self.textCursor()
78          selection.cursor.clearSelection()
79          extra_selections.append(selection)
80          self.setExtraSelections(extra_selections)
81
82      @pyqtSlot(QRect, int)
83      def update_line_number_area(self, rect, dy):
84          if dy:
85              self.lineNumberArea.scroll(0, dy)
86          else:
87              self.lineNumberArea.update(0, rect.y(), self.lineNumberArea.width(), rect.height())
88          if rect.contains(self.viewport().rect()):
89              self.updateline_number_area_width(0)
```