

## Arquivo: delegate.py

```
## módulo incluído em modules/contratos/utils.py ##

from PyQt6.QtWidgets import *
from PyQt6.QtGui import *
from PyQt6.QtCore import *
import pandas as pd
import sqlite3

class Dialogs:
    @staticmethod
    def info(parent, title, message):
        QMessageBox.information(parent, title, message)

    @staticmethod
    def warning(parent, title, message):
        QMessageBox.warning(parent, title, message)

    @staticmethod
    def error(parent, title, message):
        QMessageBox.critical(parent, title, message)

    @staticmethod
    def confirm(parent, title, message):
        reply = QMessageBox.question(parent, title, message,
                                     QMessageBox.StandardButton.Yes |
QMessageBox.StandardButton.No,
                                     QMessageBox.StandardButton.No)
        return reply == QMessageBox.StandardButton.Yes

class CustomItemDelegate(QStyledItemDelegate):
    def __init__(self, icons, status_column_index, parent=None):
        super().__init__(parent)
        self.icons = icons
        self.status_column_index = status_column_index

    def paint(self, painter, option, index):
        if index.column() == self.status_column_index:
            situacao = index.model().data(index, Qt.ItemDataRole.DisplayRole)
            icon = self.icons.get(situacao, None)

            if icon:
                icon_size = 24
                icon_x = option.rect.left() + 5
                icon_y = option.rect.top() + (option.rect.height() - icon_size) // 2
                icon_rect = QRect(int(icon_x), int(icon_y), icon_size, icon_size)

                # Obtém o pixmap no tamanho desejado
                pixmap = icon.pixmap(icon_size, icon_size)
                painter.drawPixmap(icon_rect, pixmap)
```

```

        # Ajusta o retângulo para o texto para ficar ao lado do ícone
        text_rect = QRect(
            icon_rect.right() + 5,
            option.rect.top(),
            option.rect.width() - icon_size - 10,
            option.rect.height()
        )
        option.rect = text_rect
    else:
        print(f"Ícone não encontrado para a situação: {situacao}")

    # Chama o método padrão para desenhar o texto ajustado
    super().paint(painter, option, index)

def sizeHint(self, option, index):
    size = super().sizeHint(option, index)
    if index.column() == self.status_column_index:
        size.setWidth(size.width() + 30)
    return size

class CenterAlignDelegate(QStyledItemDelegate):
    def initStyleOption(self, option, index):
        super().initStyleOption(option, index)
        option.displayAlignment = Qt.AlignmentFlag.AlignCenter

etapas = {
    'Planejamento': None,
    'Consolidar Demandas': None,
    'Montagem do Processo': None,
    'Nota Técnica': None,
    'AGU': None,
    'Recomendações AGU': None,
    'Pré-Publicação': None,
    'Sessão Pública': None,
    'Assinatura Contrato': None,
    'Concluído': None
}

```