

## Arquivo: select\_om.py

```
import json
from PyQt6.QtWidgets import QHBoxLayout, QLabel, QComboBox, QMessageBox, QSizePolicy

def create_selecao_om_layout(database_path, dados, load_sigla_om_callback, on_om_changed_callback):
    """
    Cria um layout horizontal para a seleção de OM (Organização Militar).

    Args:
        database_path (str): Caminho para o banco de dados.
        dados (dict): Dicionário com dados iniciais para configuração do layout.
        load_sigla_om_callback (callable): Função para carregar sigla OM no combo.
        on_om_changed_callback (callable): Callback para mudanças na seleção de OM.

    Returns:
        tuple: layout QHBoxLayout configurado e o QComboBox.
    """
    # Criação do layout
    om_layout = QHBoxLayout()

    # Label "OM"
    om_label = QLabel("Organização Militar:")
    om_label.setStyleSheet("font-size: 16px")
    om_layout.addWidget(om_label)

    # ComboBox para sigla OM
    om_combo = QComboBox()
    om_combo.setStyleSheet("font-size: 14px")
    om_combo.setSizePolicy(QSizePolicy.Policy.Expanding, QSizePolicy.Policy.Fixed)

    om_layout.addWidget(om_combo)

    # Determina o valor inicial da sigla OM
    sigla_om = dados.get('sigla_om', 'CeIMBra')

    # Carrega as siglas OM e define o item selecionado
    load_sigla_om_callback(database_path, om_combo, sigla_om)

    # Conecta a função on_om_changed_callback para mudanças no combo
    om_combo.currentTextChanged.connect(lambda: on_om_changed_callback(om_combo, dados, database_path))

    return om_layout, om_combo

def load_sigla_om(json_path, sigla_om_cb, sigla_om):
    """
    Carrega as siglas OM no QComboBox a partir do JSON.
    """
    try:
        print(f"DEBUG: Tentando abrir {json_path}") # Debug

        # Verifica se o arquivo realmente existe
```

```

if not json_path.exists():
    raise FileNotFoundError(f"Arquivo não encontrado: {json_path}")

# Lê o conteúdo do arquivo
with open(json_path, "r", encoding="utf-8") as file:
    content = file.read().strip() # Remove espaços extras
    print(f"DEBUG: Conteúdo do JSON -> {content}") # Debug

    # Se o arquivo estiver vazio, lança um erro
    if not content:
        raise ValueError("Arquivo JSON está vazio!")

data = json.loads(content) # Converte JSON para dicionário

# Verifica se a chave 'organizacoes' existe
if "organizacoes" not in data:
    raise ValueError("JSON inválido: chave 'organizacoes' não encontrada.")

siglas = sorted(set(org["Sigla"] for org in data["organizacoes"]))
sigla_om_cb.addItems(siglas)
sigla_om_cb.setCurrentText(sigla_om) # Define o texto atual do combo
except json.JSONDecodeError as e:
    QMessageBox.warning(None, "Erro", f"Erro ao carregar OM: JSON inválido ({e})")
    print(f"JSON Error: {e}")
except FileNotFoundError as e:
    QMessageBox.warning(None, "Erro", f"Erro ao carregar OM: {e}")
    print(f"File Not Found: {e}")
except Exception as e:
    QMessageBox.warning(None, "Erro", f"Erro ao carregar OM: {e}")
    print(f"Error loading sigla_om: {e}")

def on_om_changed(obj, om_combo, dados, json_path):
    """Callback acionado quando a seleção de OM muda no QComboBox."""
    selected_om = om_combo.currentText()
    print(f"OM changed to: {selected_om}")
    try:
        with open(json_path, "r", encoding="utf-8") as file:
            data = json.load(file)

            # Filtra a organização correspondente à sigla selecionada
            org = next((org for org in data["organizacoes"] if org["Sigla"] == selected_om), None)

            if org:
                dados['uasg'] = str(org["UASG"]) # Converte `uasg` para string
                dados['orgao_responsavel'] = org["Nome"]
                dados['indicativo'] = org["Indicativo"]
                dados['cidade'] = org["Cidade"]

                # Atualiza os valores diretamente no objeto principal
                obj.uasg = str(org["UASG"]) # Converte para string antes de atribuir a `self.uasg`
                obj.orgao_responsavel = org["Nome"]
    
```

```
# Emite o sinal para atualizar o om_label
obj.status_atualizado.emit(obj.uasg, obj.orgao_responsavel)

print(f"Updated dados: uasg={obj.uasg}, orgao_responsavel={obj.orgao_responsavel}")
except Exception as e:
    QMessageBox.warning(None, "Erro", f"Erro ao atualizar dados de OM: {e}")
    print(f"Error updating OM: {e}")
```