

Arquivo: add_button.py

```
from PyQt6.QtWidgets import QPushButton
from PyQt6.QtGui import QIcon, QCursor
from PyQt6.QtCore import QSize, Qt

def add_button_copy(text, icon_name, slot, layout, icons, tooltip=None):
    button = QPushButton()

    # Configurar ícone no botão
    icon = icons.get(icon_name)
    if icon:
        button.setIcon(icon)
    button.setIconSize(QSize(22, 22))

    # Apenas define o texto se for passado
    if text:
        button.setText(text)

    # Aplicando o CSS para o estilo do botão
    button.setStyleSheet("""
        QPushButton {
            background-color: transparent;
            border: none;
            padding: 2px; /* Ajuste de padding interno */
        }
        QPushButton:hover {
            background-color: #2C2F3F;
        }
    """)

    if tooltip:
        button.setToolTip(tooltip)

    button.clicked.connect(slot)

    # Adicionar botão ao layout
    layout.setSpacing(1)
    layout.setContentsMargins(0, 2, 0, 2)
    layout.addWidget(button)
    button.setCursor(Qt.CursorShape.PointingHandCursor)
    return button

def add_button_func(text, icon_name, slot, layout, icons, tooltip=None, button_size=None):
    button = QPushButton()

    # Configurar ícone no botão
    icon = icons.get(icon_name)
    if icon:
        button.setIcon(icon)
    button.setIconSize(QSize(30, 30))
```

```

# Define o tamanho do botão, se especificado
if button_size:
    button.setFixedSize(QSize(*button_size))

# Apenas define o texto se for passado
if text:
    button.setText(text)

# Aplicando o CSS para o estilo do botão
button.setStyleSheet(f"""
    QPushButton {{
        background-color: #F3F3F3;
        color: #333333;
        font-size: 14px;
        font-weight: bold;
        border: 1px solid #CCCCCC;
        padding: 0px 16px;
        border-radius: 5px;
    }}
    QPushButton:hover {{
        background-color: #E0E0E0;
        color: #000000;
    }}
    QPushButton:pressed {{
        background-color: #D6D6D6;
        color: #000000;
    }}
""")

if tooltip:
    button.setToolTip(tooltip)

button.clicked.connect(slot)

layout.addWidget(button)
button.setCursor(Qt.CursorShape.PointingHandCursor)
return button

```

```

def add_button_func_vermelho(text, slot, layout, tooltip=None, button_size=None):
    button = QPushButton(text)

    # Define o tamanho do botão, se especificado
    if button_size:
        button.setFixedSize(QSize(*button_size))

    button.setStyleSheet(f"""
        QPushButton {{
            background-color: #FFCCCC;
            color: #333333;
            font-size: 20px;
            font-weight: bold;
            border: 1px solid #CCCCCC;

```

```

        padding: 0px 16px;
        border-radius: 5px;
    }}
    QPushButton:hover {{
        background-color: #F3F3F3;
        color: #800000; /* Texto vermelho escuro */
    }}
    QPushButton:pressed {{
        background-color: #FF9999;
        color: #660000; /* Texto vermelho mais escuro */
    }}
    """
)

```

```

if tooltip:
    button.setToolTip(tooltip)

```

```

button.clicked.connect(slot)

```

```

layout.addWidget(button)
button.setCursor(Qt.CursorShape.PointingHandCursor)
return button

```

```

def add_button_result(label, icon_name, signal, layout, icons=None, tooltip=None,
additional_click_action=None, button_size=None):

```

```

    button = QPushButton(label)

```

```

    # Verifica se icons não é None antes de tentar obter o ícone

```

```

    if icons and icon_name in icons:

```

```

        button.setIcon(icons.get(icon_name))

```

```

    else:

```

```

        print(f"Aviso: Ícone '{icon_name}' não encontrado ou 'icons' não foi passado.")

```

```

    button.setIconSize(QSize(30, 30))

```

```

    button.clicked.connect(signal.emit)

```

```

    # Adiciona a ação adicional ao clique, se fornecida

```

```

    if additional_click_action:

```

```

        button.clicked.connect(additional_click_action)

```

```

    # Define o tamanho do botão, se especificado

```

```

    if button_size:

```

```

        button.setFixedSize(QSize(*button_size))

```

```

    # Estilo do botão

```

```

    button.setStyleSheet(f"""

```

```

        QPushButton {{

```

```

            background-color: #F3F3F3;

```

```

            color: #333333;

```

```

            font-size: 14px;

```

```

            font-weight: bold;

```

```

            border: 1px solid #CCCCCC;

```

```

            padding: 0px 16px;

```

```

        border-radius: 5px;
    }}
    QPushButton:hover {{
        background-color: #E0E0E0;
        color: #000000;
    }}
    QPushButton:pressed {{
        background-color: #D6D6D6;
        color: #000000;
    }}
    """)

```

```

button.setCursor(QCursor(Qt.CursorShape.PointingHandCursor))
if tooltip:
    button.setToolTip(tooltip)

layout.addWidget(button)
return button

```

```

def add_button(label, icon_name, signal, layout, icons=None, tooltip=None, button_size=None):
    button = QPushButton(label)

```

```

    # Verifica se icons não é None antes de tentar obter o ícone

```

```

    if icons and icon_name in icons:

```

```

        button.setIcon(icons.get(icon_name))

```

```

    else:

```

```

        print(f"Aviso: Ícone '{icon_name}' não encontrado ou 'icons' não foi passado.")

```

```

button.setIconSize(QSize(30, 30)) # Aumenta o tamanho do ícone para sobrepor os limites
button.clicked.connect(signal.emit)

```

```

# Define o tamanho do botão, se especificado

```

```

if button_size:

```

```

    button.setFixedSize(QSize(*button_size))

```

```

# Estilo do botão

```

```

button.setStyleSheet(f"""

```

```

    QPushButton {{
        background-color: #F3F3F3;
        color: #333333;
        font-size: 14px;
        font-weight: bold;
        border: 1px solid #CCCCCC;
        padding: 0px 16px;
        border-radius: 5px;
    }}

```

```

    QPushButton:hover {{
        background-color: #E0E0E0;
        color: #000000;
    }}

```

```

    QPushButton:pressed {{
        background-color: #D6D6D6;
    }}

```

```

        color: #000000;
    }}
    """)

button.setCursor(QCursor(Qt.CursorShape.PointingHandCursor))
if tooltip:
    button.setToolTip(tooltip)

layout.addWidget(button)
return button

def create_button(text, icon, callback, tooltip_text, icon_size=QSize(30, 30)):
    btn = QPushButton(text)

    # Define o ícone e tamanho do ícone, se fornecido
    if icon:
        btn.setIcon(QIcon(icon))
        btn.setIconSize(icon_size)

    # Conecta o callback ao clique, se fornecido
    if callback:
        btn.clicked.connect(callback)

    # Define o estilo de hover para o botão
    btn.setStyleSheet("""
        QPushButton {
            background-color: transparent;
            padding: 8px;
        }
        QPushButton:hover {
            background-color: #222236;
        }
    """)
    btn.setCursor(Qt.CursorShape.PointingHandCursor)
    return btn

```