QColor リファレンス

PySide6における色の定義と操作についてのリファレンス資料です。

概要

`QColor` クラスは、色を表現し操作するためのクラスです。RGB、HSV、CMYK、HSLなど様々な色空間をサポートします。

サンプルアプリケーション

このリファレンスには、3つの実践的なサンプルアプリケーションが含まれています：

1. 基本的な色の作成 (`qcolor_basic.py`)

- RGB値、16進数、色名からの色の作成方法
- 各種色作成方法の比較とデモンストレーション
- コンパクトなウィンドウサイズ（600x400）

2. RGB値の制御 (`qcolor_rgb_control.py`)

- スライダーとスピンボックスによるRGB値の調整
- リアルタイムな色プレビュー
- RGB値と16進数の表示

3. カラーパレットとテーマ (`qcolor_palette.py`)

- Material Designカラーパレットの表示
- ライト/ダークテーマの切り替え機能
- 美しい色の組み合わせの実例

基本的な使用方法

色の作成

```python
from PySide6.QtGui import QColor

# RGB値から作成
color1 = QColor(255, 0, 0)  # 赤
color2 = QColor(0, 255, 0)  # 緑
color3 = QColor(0, 0, 255)  # 青

# 16進数から作成
color4 = QColor("#FF0000")  # 赤
color5 = QColor("#00FF00")  # 緑

# 色名から作成
color6 = QColor("red")
color7 = QColor("blue")
```

カラーサンプルウィジェットの作成

以下は実際のサンプルアプリケーションで使用されている色サンプルの作成方法です：

```
def create_color_sample(self, color, text):
    """色サンプルウィジェットの作成"""
    widget = QWidget()
    widget.setFixedSize(80, 60)

    # 色に基づいてテキストの色を決定
    luminance = (0.299 * color.red() + 0.587 * color.green() + 0.114 * color.blue()) / 255
    text_color = "white" if luminance < 0.5 else "black"

    widget.setStyleSheet(f"""
        QWidget {{
            background-color: {color.name()};
            border: 1px solid #333;
            border-radius: 4px;
        }}
    """)

    layout = QVBoxLayout()
    label = QLabel(text)
    label.setAlignment(Qt.AlignmentFlag.AlignCenter)
    label.setStyleSheet(f"color: {text_color}; font-size: 10px; font-weight: bold;")
    label.setWordWrap(True)
    layout.addWidget(label)
    widget.setLayout(layout)

    return widget
```

RGB値の制御

スライダーによる色の制御

```
# RGB スライダーの設定例
self.r_slider = QSlider(Qt.Orientation.Vertical)
self.r_slider.setRange(0, 255)
self.r_slider.setValue(128)
self.r_slider.valueChanged.connect(self.update_rgb_color)

def update_rgb_color(self):
    """RGB値の変更に基づいて色を更新"""
    r = self.r_slider.value()
    g = self.g_slider.value()
    b = self.b_slider.value()

    self.update_color_display(r, g, b)

def update_color_display(self, r, g, b):
    """色表示の更新"""
    color = QColor(r, g, b)

    # 背景色を更新
    self.color_display.setStyleSheet(f"""
        QLabel {{
            border: 2px solid black;
            background-color: rgb({r}, {g}, {b});
            color: {'white' if (0.299 * r + 0.587 * g + 0.114 * b) < 128 else 'black'};
            font-weight: bold;
        }}
    """)

    # 色情報を更新
    self.color_info.setText(f"RGB({r}, {g}, {b})\n{color.name().upper()}")
```

色空間の変換

RGB値の取得・設定

```
color = QColor(255, 128, 64)

# RGB値の取得
r = color.red()
g = color.green()
b = color.blue()
a = color.alpha()  # アルファ値（透明度）

# RGB値の設定
color.setRed(200)
color.setGreen(100)
color.setBlue(50)
color.setAlpha(200)  # 透明度設定
```

HSV値の操作

```
color = QColor()

# HSV値から設定
color.setHsv(120, 255, 255)  # 色相, 彩度, 明度
```

```
# HSV値の取得
h = color.hue()
s = color.saturation()
v = color.value()
```

テーマとスタイルシート

ライトテーマとダークテーマ

```
def apply_light_theme(self):
    """ライトテーマの適用"""
    self.setStyleSheet("""
        QWidget {
            background-color: #ffffff;
            color: #2c3e50;
        }
        QGroupBox {
            background-color: #f8f9fa;
        }
    """)

def apply_dark_theme(self):
    """ダークテーマの適用"""
    self.setStyleSheet("""
        QWidget {
            background-color: #2c3e50;
            color: #ecf0f1;
        }
        QGroupBox {
            background-color: #34495e;
        }
    """)
```

スタイルシートでの色の使用

```
# ウィジェットのスタイル設定
widget.setStyleSheet("""
    QPushButton {
        background-color: #3498db;
        color: white;
        border: 2px solid #2980b9;
        padding: 10px;
        border-radius: 5px;
        font-weight: bold;
    }
    QPushButton:hover {
        background-color: #2980b9;
    }
""")
```

Material Design カラーパレット

推奨色の組み合わせ

```
# Primary Colors
RED_500 = QColor("#F44336")      # 赤
PINK_500 = QColor("#E91E63")     # ピンク
PURPLE_500 = QColor("#9C27B0")   # 紫
DEEP_PURPLE_500 = QColor("#673AB7")  # 深紫

INDIGO_500 = QColor("#3F51B5")   # インディゴ
BLUE_500 = QColor("#2196F3")     # 青
LIGHT_BLUE_500 = QColor("#03A9F4")   # 水色
CYAN_500 = QColor("#00BCD4")     # シアン

TEAL_500 = QColor("#009688")     # ティール
GREEN_500 = QColor("#4CAF50")    # 緑
LIGHT_GREEN_500 = QColor("#8BC34A")  # 明るい緑
LIME_500 = QColor("#CDDC39")     # ライム

YELLOW_500 = QColor("#FFEB3B")   # 黄色
AMBER_500 = QColor("#FFC107")    # アンバー
ORANGE_500 = QColor("#FF9800")   # オレンジ
DEEP_ORANGE_500 = QColor("#FF5722")  # 深いオレンジ
```

標準色名

色名	RGB値	16進数	日本語
red	(255, 0, 0)	#FF0000	赤
green	(0, 128, 0)	#008000	緑
blue	(0, 0, 255)	#0000FF	青
yellow	(255, 255, 0)	#FFFF00	黄色

色名	RGB値	16進数	日本語
cyan	(0, 255, 255)	#00FFFF	シアン
magenta	(255, 0, 255)	#FF00FF	マゼンタ
black	(0, 0, 0)	#000000	黒
white	(255, 255, 255)	#FFFFFF	白
gray	(128, 128, 128)	#808080	灰色
orange	(255, 165, 0)	#FFA500	オレンジ

サンプルアプリケーションの実行

各サンプルアプリケーションは独立して実行可能です：

```
# 基本的な色の作成
python qcolor_basic.py

# RGB値の制御
python qcolor_rgb_control.py

# カラーパレットとテーマ
python qcolor_palette.py
```

実用的なTips

1. 適切なテキスト色の選択

```python
def get_text_color(background_color):
    """背景色に基づいて適切なテキスト色を返す"""
    luminance = (0.299 * background_color.red() +
                 0.587 * background_color.green() +
                 0.114 * background_color.blue()) / 255
    return "white" if luminance < 0.5 else "black"
```

2. 色の有効性チェック

```python
color = QColor("#FF0000")
if color.isValid():
    print("有効な色です")
else:
    print("無効な色です")
```

3. 色の比較

```python
color1 = QColor(255, 0, 0)
color2 = QColor("#FF0000")

if color1 == color2:
    print("同じ色です")
```

参考リンク

- Qt Documentation - QColor
- PySide6 QColor