```
import sys
from PyQt5.QtWidgets import (QApplication, QMainWindow, QWidget, QLabel, QVBoxLayout, QHBoxLayout,
                             QPushButton, QColorDialog, QComboBox, QSlider, QFileDialog, QAction, QToolBar)
from PyQt5.QtGui import QPainter, QPen, QPixmap, QColor, QPainterPath, QIcon
from PyQt5.QtCore import Qt, QSize, QPoint

class DrawingCanvas(QWidget):
    def __init__(self, parent=None):
        super().__init__(parent)
        self.setFixedSize(800, 500)

        # 创建画布
        self.canvas = QPixmap(self.size())
        self.canvas.fill(Qt.white)

        # 绘图属性
        self.pen_color = QColor(Qt.black)
        self.pen_width = 3
        self.drawing = False
        self.last_point = QPoint()
        self.tool = "pen"  # 默认工具
        self.start_point = QPoint()
        self.shape = "line"  # 默认形状

    def paintEvent(self, event):
        painter = QPainter(self)
        painter.drawPixmap(0, 0, self.canvas)

    def mousePressEvent(self, event):
        if event.button() == Qt.LeftButton:
            self.drawing = True
            self.last_point = event.pos()
            self.start_point = event.pos()

            # 如果是橡皮擦，直接擦除
            if self.tool == "eraser":
                eraser_painter = QPainter(self.canvas)
                eraser_painter.setPen(QPen(Qt.white, self.pen_width * 3))
                eraser_painter.drawPoint(event.pos())
                self.update()

    def mouseMoveEvent(self, event):
```

```
        if (event.buttons() & Qt.LeftButton) and self.drawing:
            painter = QPainter(self.canvas)

            if self.tool == "pen":
                painter.setPen(QPen(self.pen_color, self.pen_width, Qt.SolidLine,
Qt.RoundCap))
                painter.drawLine(self.last_point, event.pos())
                self.last_point = event.pos()
                self.update()
            elif self.tool == "eraser":
                painter.setPen(QPen(Qt.white, self.pen_width * 3))
                painter.drawLine(self.last_point, event.pos())
                self.last_point = event.pos()
                self.update()

    def mouseReleaseEvent(self, event):
        if event.button() == Qt.LeftButton and self.drawing:
            self.drawing = False

            # 绘制形状
            if self.tool == "shape":
                painter = QPainter(self.canvas)
                painter.setPen(QPen(self.pen_color, self.pen_width))
                end_point = event.pos()

                if self.shape == "line":
                    painter.drawLine(self.start_point, end_point)
                elif self.shape == "rect":
                    painter.drawRect(self.start_point.x(), self.start_point.y(),
                                     end_point.x() - self.start_point.x(),
                                     end_point.y() - self.start_point.y())
                elif self.shape == "ellipse":
                    painter.drawEllipse(self.start_point.x(), self.start_point.y(),
                                        end_point.x() - self.start_point.x(),
                                        end_point.y() - self.start_point.y())
                elif self.shape == "triangle":
                    path = QPainterPath()
                    path.moveTo((self.start_point.x() + end_point.x()) / 2, self.start_point.y())
                    path.lineTo(self.start_point.x(), end_point.y())
                    path.lineTo(end_point.x(), end_point.y())
                    path.lineTo((self.start_point.x() + end_point.x()) / 2, self.start_point.y())
                    painter.drawPath(path)

                self.update()
```

```
    def clear_canvas(self):
        self.canvas.fill(Qt.white)
        self.update()

    def save_canvas(self, file_path):
        self.canvas.save(file_path)

class DrawingApp(QMainWindow):
    def __init__(self):
        super().__init__()
        self.setWindowTitle("Python 画图软件")
        self.setGeometry(100, 100, 900, 650)

        # 创建主部件和布局
        self.main_widget = QWidget()
        self.setCentralWidget(self.main_widget)

        # 创建工具栏
        self.toolbar = QToolBar("主工具栏")
        self.toolbar.setIconSize(QSize(24, 24))
        self.addToolBar(Qt.LeftToolBarArea, self.toolbar)

        # 创建画布
        self.canvas = DrawingCanvas()

        # 创建控制面板
        control_panel = QWidget()
        control_panel.setFixedWidth(150)

        # 创建按钮
        self.pen_btn = QPushButton("画笔")
        self.pen_btn.setCheckable(True)
        self.pen_btn.setChecked(True)

        self.eraser_btn = QPushButton("橡皮擦")
        self.eraser_btn.setCheckable(True)

        self.shape_btn = QPushButton("形状")
        self.shape_btn.setCheckable(True)

        # 形状选择
        shape_label = QLabel("选择形状:")
        self.shape_combo = QComboBox()
```

```
self.shape_combo.addItems(["直线", "矩形", "椭圆", "三角形"])

# 颜色选择
color_label = QLabel("选择颜色:")
self.color_btn = QPushButton()
self.color_btn.setStyleSheet("background-color: black;")
self.color_btn.setFixedSize(30, 30)

# 画笔大小
size_label = QLabel("画笔大小:")
self.size_slider = QSlider(Qt.Horizontal)
self.size_slider.setRange(1, 20)
self.size_slider.setValue(3)

# 操作按钮
clear_btn = QPushButton("清空画布")
save_btn = QPushButton("保存图像")

# 布局控制面板
control_layout = QVBoxLayout()
control_layout.addWidget(self.pen_btn)
control_layout.addWidget(self.eraser_btn)
control_layout.addWidget(self.shape_btn)
control_layout.addWidget(shape_label)
control_layout.addWidget(self.shape_combo)
control_layout.addWidget(color_label)
control_layout.addWidget(self.color_btn)
control_layout.addWidget(size_label)
control_layout.addWidget(self.size_slider)
control_layout.addStretch()
control_layout.addWidget(clear_btn)
control_layout.addWidget(save_btn)
control_panel.setLayout(control_layout)

# 主布局
main_layout = QHBoxLayout()
main_layout.addWidget(control_panel)
main_layout.addWidget(self.canvas)
self.main_widget.setLayout(main_layout)

# 连接信号
self.pen_btn.clicked.connect(self.set_pen_tool)
self.eraser_btn.clicked.connect(self.set_eraser_tool)
self.shape_btn.clicked.connect(self.set_shape_tool)
```

```
        self.color_btn.clicked.connect(self.choose_color)
        self.size_slider.valueChanged.connect(self.set_pen_size)
        self.shape_combo.currentIndexChanged.connect(self.set_shape)
        clear_btn.clicked.connect(self.canvas.clear_canvas)
        save_btn.clicked.connect(self.save_image)

        # 设置初始工具
        self.set_pen_tool()

        # 创建菜单栏
        self.create_menu()

    def create_menu(self):
        menubar = self.menuBar()

        # 文件菜单
        file_menu = menubar.addMenu('文件')

        new_action = QAction('新建', self)
        new_action.triggered.connect(self.canvas.clear_canvas)
        file_menu.addAction(new_action)

        save_action = QAction('保存', self)
        save_action.triggered.connect(self.save_image)
        file_menu.addAction(save_action)

        exit_action = QAction('退出', self)
        exit_action.triggered.connect(self.close)
        file_menu.addAction(exit_action)

        # 帮助菜单
        help_menu = menubar.addMenu('帮助')
        about_action = QAction('关于', self)
        about_action.triggered.connect(self.show_about)
        help_menu.addAction(about_action)

    def set_pen_tool(self):
        self.pen_btn.setChecked(True)
        self.eraser_btn.setChecked(False)
        self.shape_btn.setChecked(False)
        self.canvas.tool = "pen"

    def set_eraser_tool(self):
        self.pen_btn.setChecked(False)
```

```
        self.eraser_btn.setChecked(True)
        self.shape_btn.setChecked(False)
        self.canvas.tool = "eraser"

    def set_shape_tool(self):
        self.pen_btn.setChecked(False)
        self.eraser_btn.setChecked(False)
        self.shape_btn.setChecked(True)
        self.canvas.tool = "shape"

    def choose_color(self):
        color = QColorDialog.getColor(self.canvas.pen_color, self, "选择颜色")
        if color.isValid():
            self.canvas.pen_color = color
            self.color_btn.setStyleSheet(f"background-color: {color.name()};")

    def set_pen_size(self, size):
        self.canvas.pen_width = size

    def set_shape(self, index):
        shapes = ["line", "rect", "ellipse", "triangle"]
        self.canvas.shape = shapes[index]

    def save_image(self):
        file_path, _ = QFileDialog.getSaveFileName(
            self, "保存图像", "", "PNG 图像 (*.png);;JPEG 图像 (*.jpg *.jpeg);;所有文件 (*)"
        )
        if file_path:
            self.canvas.save_canvas(file_path)

    def show_about(self):
        QMessageBox.about(self, "关于画图软件",
                          "Python 画图软件\n"
                          "版本 1.0\n\n"
                          "使用 PyQt5 创建的简单绘图应用程序\n"
                          "支持画笔、橡皮擦、形状绘制等功能")

if __name__ == "__main__":
    app = QApplication(sys.argv)
    window = DrawingApp()
    window.show()
    sys.exit(app.exec_())
```