import win32gui

import win32con

import win32api

import os

from ctypes import windll, byref, c\_int, create\_unicode\_buffer, create\_string\_buffer

from pywin.mfc import window, menu, dialog, object

from pywin.mfc import docview

# Define menu and toolbar IDs

ID\_RED = 101

ID\_GREEN = 102

ID\_BLUE = 103

ID\_SHOW = 104

class MyDoc(docview.Doc):

def \_\_init\_\_(self):

docview.Doc.\_\_init\_\_(self)

class MyView(docview.View):

def \_\_init\_\_(self):

docview.View.\_\_init\_\_(self)

self.text = "2052312胡博文"

self.color = win32api.RGB(0, 0, 0)

def OnDraw(self, dc):

dc.SetTextColor(self.color)

dc.TextOut(10, 10, self.text)

class MyFrame(window.FrameWnd):

def \_\_init\_\_(self):

window.FrameWnd.\_\_init\_\_(self, None, "My App")

self.SetIcon(win32gui.LoadIcon(0, win32con.IDI\_APPLICATION))

self.CreateToolBar()

self.CreateStatusBar()

self.SetMenu()

def SetMenu(self):

menuBar = menu.MenuBar()

fileMenu = menu.Menu()

editMenu = menu.Menu()

viewMenu = menu.Menu()

fontMenu = menu.Menu()

colorMenu = menu.Menu()

fileMenu.CreatePopupMenu()

fileMenu.AppendMenu(win32con.MF\_STRING, win32con.ID\_NEW, "&New")

fileMenu.AppendMenu(win32con.MF\_STRING, win32con.ID\_OPEN, "&Open")

fileMenu.AppendMenu(win32con.MF\_SEPARATOR, 0, None)

fileMenu.AppendMenu(win32con.MF\_STRING, win32con.ID\_EXIT, "&Exit")

menuBar.AppendMenu(fileMenu, "&File")

editMenu.CreatePopupMenu()

editMenu.AppendMenu(win32con.MF\_STRING, win32con.ID\_CUT, "Cu&t")

editMenu.AppendMenu(win32con.MF\_STRING, win32con.ID\_COPY, "&Copy")

editMenu.AppendMenu(win32con.MF\_STRING, win32con.ID\_PASTE, "&Paste")

menuBar.AppendMenu(editMenu, "&Edit")

fontMenu.CreatePopupMenu()

fontMenu.AppendMenu(win32con.MF\_STRING, win32con.ID\_FONT\_SMALL, "&Small")

fontMenu.AppendMenu(win32con.MF\_STRING, win32con.ID\_FONT\_MEDIUM, "&Medium")

fontMenu.AppendMenu(win32con.MF\_STRING, win32con.ID\_FONT\_LARGE, "&Large")

viewMenu.AppendMenu(win32con.MF\_POPUP, fontMenu.GetHandle(), "&Font")

colorMenu.CreatePopupMenu()

colorMenu.AppendMenu(win32con.MF\_STRING, ID\_RED, "&Red")

colorMenu.AppendMenu(win32con.MF\_STRING, ID\_GREEN, "&Green")

colorMenu.AppendMenu(win32con.MF\_STRING, ID\_BLUE, "&Blue")

viewMenu.AppendMenu(win32con.MF\_POPUP, colorMenu

viewMenu.AppendMenu(win32con.MF\_SEPARATOR, 0, None)

viewMenu.AppendMenu(win32con.MF\_STRING, ID\_SHOW, "&Show Toolbar")

menuBar.AppendMenu(viewMenu, "&View")

self.SetMenuBar(menuBar)

def CreateToolBar(self):

toolbar = self.CreateSimpleToolBar()

toolbar.LoadToolBar(os.path.join(os.getcwd(), "toolbar.bmp"))

toolbar.SetButtons((ID\_SHOW, ID\_RED, ID\_GREEN, ID\_BLUE))

toolbar.SetButtonInfo(0, ID\_SHOW, win32con.TBSTATE\_ENABLED, 0)

toolbar.SetButtonInfo(1, ID\_RED, win32con.TBSTATE\_ENABLED, 1)

toolbar.SetButtonInfo(2, ID\_GREEN, win32con.TBSTATE\_ENABLED, 2)

toolbar.SetButtonInfo(3, ID\_BLUE, win32con.TBSTATE\_ENABLED, 3)

def OnCmdMsg(self, nID, nCode, pExtra, pHandlerInfo):

if nCode == win32con.CN\_COMMAND:

if nID == ID\_RED:

self.view.color = win32api.RGB(255, 0, 0)

self.view.InvalidateRect(None, True)

elif nID == ID\_GREEN:

self.view.color = win32api.RGB(0, 255, 0)

self.view.InvalidateRect(None, True)

elif nID == ID\_BLUE:

self.view.color = win32api.RGB(0, 0, 255)

self.view.InvalidateRect(None, True)

elif nID == ID\_SHOW:

self.ShowToolBar(not self.IsToolBarVisible())

return window.FrameWnd.OnCmdMsg(self, nID, nCode, pExtra, pHandlerInfo)

if name == "main":

app = dialog.PumpMessages()

doc = MyDoc()

view = MyView()

frame = MyFrame()

frame.CreateView(0, 0, MyView)

frame.ShowWindow(win32con.SW\_SHOW)

app.Run()

class MyView(window.ViewWnd):

def \_\_init\_\_(self):

window.ViewWnd.\_\_init\_\_(self)

self.color = win32api.RGB(0, 0, 0)

def OnDraw(self, dc):

dc.SetTextColor(self.color)

dc.DrawText("2052312胡博文", (0, 0, self.GetClientRect()[2], self.GetClientRect()[3]), win32con.DT\_CENTER | win32con.DT\_VCENTER | win32con.DT\_SINGLELINE)

def OnCmdMsg(self, nID, nCode, pExtra, pHandlerInfo):

if nCode == win32con.CN\_COMMAND:

if nID == ID\_RED:

self.color = win32api.RGB(255, 0, 0)

self.InvalidateRect(None, True)

elif nID == ID\_GREEN:

self.color = win32api.RGB(0, 255, 0)

self.InvalidateRect(None, True)

elif nID == ID\_BLUE:

self.color = win32api.RGB(0, 0, 255)

self.InvalidateRect(None, True)

return window.ViewWnd.OnCmdMsg(self, nID, nCode, pExtra, pHandlerInfo)