2020/07/08_MFC_第8课_CAD04_画图的基本操作

笔记本:	MFC
创建时间:	2020/7/8 星期三 15:35
作者:	ileemi
标签:	画图的基本操作

CAD画图的基本操作

CAD 是以图形为单位，绘图工具是以像素为单位。

选择

- 封闭图形的选中：鼠标在图形的范围内部。
- 直线的选中：鼠标落在直线的周围（需要一定的范围内）。

MFC 对应的API：
矩形：
CRect --> PtInRect

说明：确定指定点是否在正确范围内。如果一个点位于左侧或顶部，或者位于所有四个边内，则该点在正确范围内。右侧或底部的点不正确。

椭圆：
CRect --> PtInRect

直线：
直线外圈一个多边形

绘制多边形的API
CRect -- polygon
CRgn -- PtInRegion

在线条外绘制一个多边形的API -- CRgn --> CreatePolygonRgn

左键单击选中会和绘制图形冲突 -- 解决办法给个**选中**菜单。之后为菜单添加一个事件处理程序。

在 View 试图类中的 LButtonDown 中做文章，当点击选中图形后，将点击的坐标点，通过遍历存储图形的链表（从后往前遍历），取出图形，将点击的坐标点传入到对应的图形中，进行判断，点击的坐标点在四边形的范围内，即表示图形选择成功。为了表示图形选择成功，当图形选择成功后，可以为其添加一个风格样式（在图形类的父类中设定一个绘制选中图形的方法，**OnDrawSelect**，子类自己去实现 ）。让其有辨别度即可。

代码示例：

```
5   //·图像父类
6   class·IShape
7   {
8   public:
9   ··virtual·void·OnDraw(CDC*·pDC)·=·0;··//·绘制图形
10  ··virtual·void·OnDrawSelect(CDC*·pDC)·=·0;··//·当图形被选中，绘制指定样式的图形
11  ··virtual·BOOL·IsSelect(CPoint·pt)·=·0;···//·判断坐标点是否在对应的图形内
12  ··void·MoveShape(CPoint·ptMoveBegin,·CPoint·ptMoveEnd);···//·移动图形
13  ··void·RotateShape();·//旋转图形
14
49  void·CLine::OnDrawSelect(CDC*·pDC)
50  {
51  ··//·当图形被选中后使用这里的样式绘制显示
52  ··CPen·pen;
53  ··pen.CreatePen(
54  ····myPen.GetPenStyle(),
55  ····5,
56  ····RGB(0,·128,·255));
57  ··pDC->SelectObject(&pen);
58
59  ··//·绘制直线
60  ··pDC->MoveTo(GetPointBegin());
61  ··pDC->LineTo(GetPointEnd());
62  }
```

当图形被选中后，在 OnDraw 中进行绘制指定样式的图形，代码示例：

```
145 ··//·3.·在内存中进行绘制
146 ··POSITION·pos·=·m_listShape.GetHeadPosition();··//·获取链表中的头结点
147 ··//·遍历结点数据
148 ··while·(pos)
149 ··{
150 ····auto·Shape·=·m_listShape.GetNext(pos);
151 ····//·判断当前绘制的图形是否被选中
152 ····if·(Shape·==·m_pSelectedShape)
153 ····{
154 ····Shape->OnDrawSelect(&dcMem);·····//·绘制被选中后的状态
155 ····}·
156 ····else
157 ····{
158 ····//·未选中绘制默认样式的图形
159 ····//·将获取到的线绘制到内存中
160 ····Shape->OnDraw(&dcMem);
161 ····}
162 ··}
```

移动

图形的移动需要知道图形移动前的坐标以及移动后的坐标。

WM_LBUTTONDOWN内 保存图形的起点坐标，WM_LBUTTONMOVE内，保存系统的终点坐标。

图形的移动需要在选中图形的前提下，长按鼠标左键。在 IShape 中计算移动前和移动后的坐标差，之后，设置新的坐标点为图形移动后的坐标点。

删除

选中图形，通过链表查找当前选中的图形，进行删除操作。

添加对应的删除菜单（同时相应事件处理程序），添加对应的操作代码即可。
代码示例：

```
424  //·删除图形
425 void·CCADView::OnOptDeleteShape()
426  {
427  ··//·链表图形个数为0，不进行删除操作
428  ··if·(m_pSelectedShape·!=·NULL)
429  ··{
430  ····m_listShape.RemoveAt(m_listShape.Find(m_pSelectedShape));
431  ····delete·m_pSelectedShape;
432  ····m_pSelectedShape·=·NULL;
433  ····InvalidateRect(NULL,·FALSE);····//·刷新客户区
434  ··}
435  }
436
```

旋转

选中图形，通过链表查找当前选中的图形，进行旋转操作。

添加对应的旋转菜单（同时相应事件处理程序），添加对应的操作代码即可。
代码示例：

```
437  //·旋转
438 void·CCADView::OnOptRotateShape()
439  {
440  ··if·(m_pSelectedShape·!=·NULL)
441  ··{
442  ····m_pSelectedShape->RotateShape();
443  ····InvalidateRect(NULL,·FALSE);
444  ··}
445  }
```

```
20  //·旋转
21 void·IShape::RotateShape()
22  {
23  ··rotate(m_ptBegin,·m_ptEnd,·PI·*·90·/·180);
24  }
25
```

IShape 类中的方法

旋转算法：

```
const int LINERANGE = 20;
const double PI = 3.14159265358979323846;
#include "math.h"
inline void rotate(CPoint& beginPos,CPoint& endPos,double dblR)
{
  double x = (beginPos.x + endPos.x)/2.0;
  double y = (beginPos.y + endPos.y)/2.0;

  CPoint pos;

  pos.x = (long)((beginPos.x - x)*cos(dblR) - (beginPos.y - y)*sin(dblR) + x);
  pos.y = (long)((beginPos.x - x)*sin(dblR) - (beginPos.y - y)*cos(dblR) + y);

  beginPos = pos;

  pos.x = (long)((endPos.x - x)*cos(dblR) - (endPos.y - y)*sin(dblR) + x);
  pos.y = (long)((endPos.x - x)*sin(dblR) - (endPos.y - y)*cos(dblR) + y);

  endPos = pos;
}
```

右键菜单

在 MFC 工程中，添加一个菜单资源，给定ID，菜单的消息相应和单文档默认菜单的消息相应用法一致，为菜单添加指定的 "添加事件处理程序"。

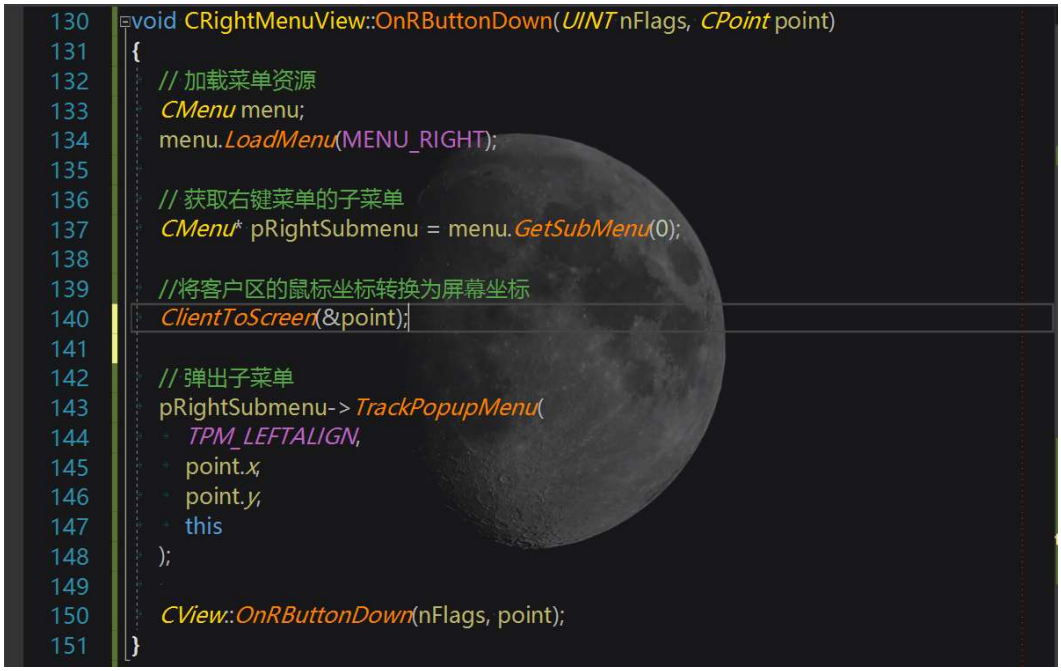
鼠标右键 -- LRButtonDown 中加载指定的菜单。
MFC 对其进行了封装，**CMenu** 类：
loadMenu -- 加载菜单
说明：从可执行文件加载菜单资源并将其附加到CMenu对象

TrackPopupMenu -- 弹出菜单
说明：在指定位置显示一个浮动的弹出菜单，并跟踪弹出菜单上项目的选择。
TrackPopupMenu 参数4 填写响应菜单消息的窗口指针 。

GetSubMenu -- 获取右键菜单（主菜单）的子菜单，其返回值为菜单的下标索引值。主菜单下标从零开始。
说明：检索弹出菜单的指针

弹出的菜单默认坐标为屏幕的坐标，鼠标消息响应的坐标为客户区的坐标，使用 **ClientToScreen** 将显示器上指定点或矩形的客户端坐标转换为屏幕坐标。

参数为指针，其是一个传入传出参数
使用示例：



```
130 void CRightMenuView::OnRButtonDown(UINT nFlags, CPoint point)
131 {
132     // 加载菜单资源
133     CMenu menu;
134     menu.LoadMenu(MENU_RIGHT);
135 
136     // 获取右键菜单的子菜单
137     CMenu* pRightSubmenu = menu.GetSubMenu(0);
138 
139     //将客户区的鼠标坐标转换为屏幕坐标
140     ClientToScreen(&point);
141 
142     // 弹出子菜单
143     pRightSubmenu->TrackPopupMenu(
144         TPM_LEFTALIGN,
145         point.x,
146         point.y,
147         this
148     );
149 
150     CView::OnRButtonDown(nFlags, point);
151 }
```