

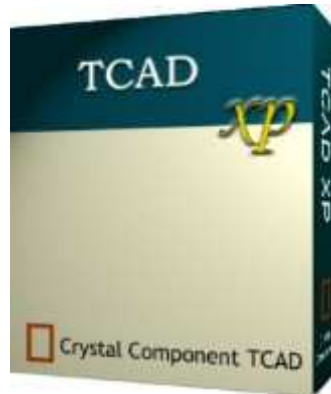
# 1 感谢您

感谢你购买我们的 TCAD 产品，希望能够基于 TCAD 开发出强大的应用软件。

## 2 介绍

### 2.1 什么是 TCAD

TCAD 是一个能帮助你开发矢量图形应用程序的组件。图形能通过鼠标或者代码相互作用。使用起来非常简单、有效、功能强大。她将节省你许多宝贵的时间。



你想在应用程序中加入矢量绘图功能，假如自己从底层写起，发现工作量非常大。现在，你可以使用 TCAD 组件，在应用程序开发中仅通过鼠标控制就可以非常简单的创建、使用矢量图形。TCAD 使你轻松地编程。

## 主要功能

### 图形类型

- 线、单箭头线、双箭头线
- 连接线
- 标尺
- 多义线
- 多边形
- 贝塞尔曲线
- 矩形
- 圆弧、椭圆弧、弦、扇形
- 圆、椭圆
- 文本
- 图像
- 用户自定义图形

### 组合/取消组合

### 支持图层

通过代码创建、移动、旋转图形

保存到文件或数据库

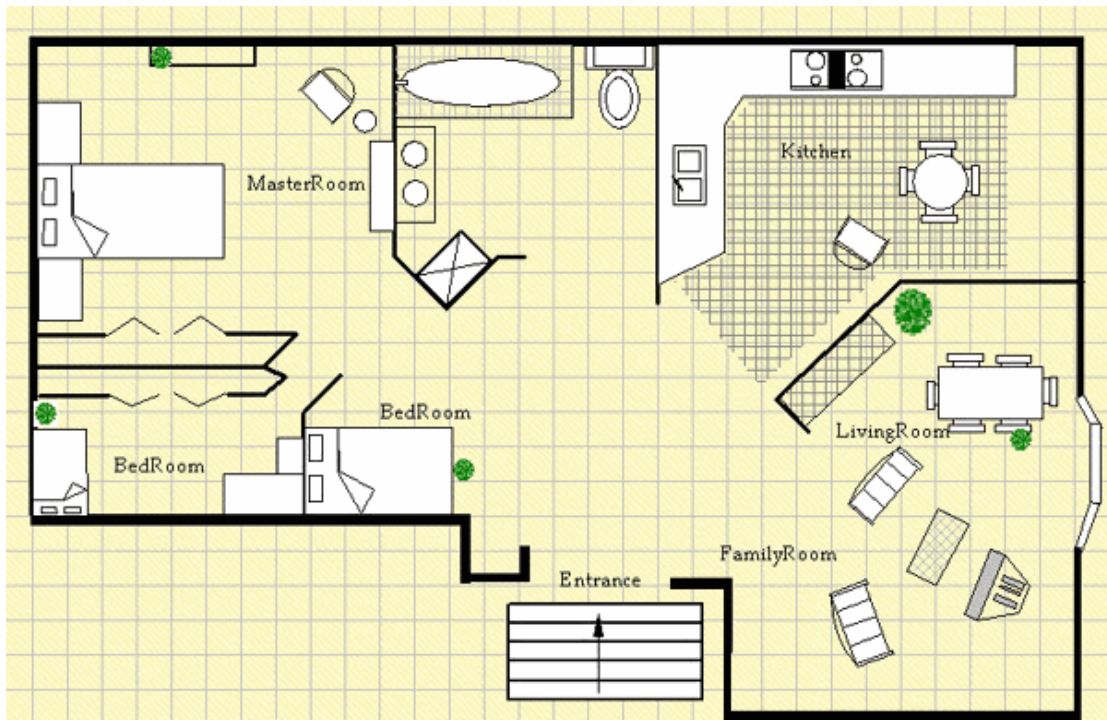
方便地创建用户自定义图形

支持四种坐标系

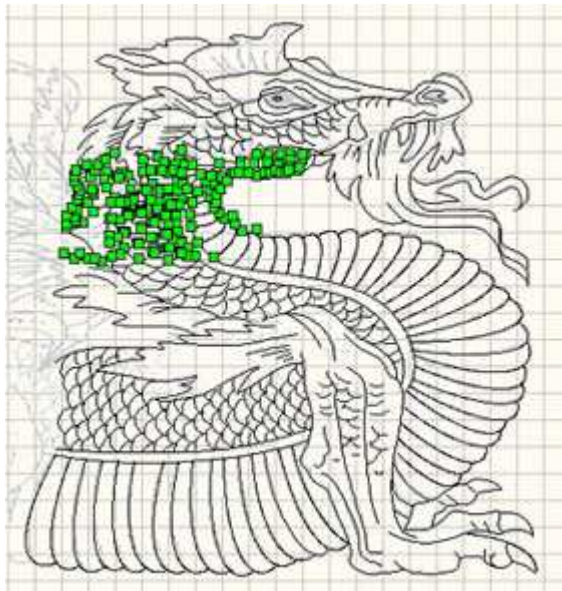
## 详细信息

- 通过鼠标动作或代码创建图形
- 修改已经创建的图形
- 支持图层、打印、删除、可见、不可见
- 使用任何颜色
- 可以使用不同的画笔或画刷
- 具有图形变化相应的事件
- 使用各种格式的纸张类型（如 A0、A1、A2、A3、A4、letter 等等）或自定义类型
- -----
- 能撤销/重复撤销操作步骤
- 剪切、复制、粘贴、删除图形
- 控制图形（移到最前、移到最后等等）
- 通过鼠标或代码旋转、拖动、缩放图形
- 任意类型的调整图形
- 简单的创建用户自定义的联合图形
- 鼠标对充到网格，可以设定网格长度和宽度
- 支持 24 种渐变色
- -----
- 锁定/解锁图形
- 显示或隐藏图形执点
- 组合/撤销组合图形
- 任意比例的图形缩放、浏览
- 当鼠标移到图形时显示提示
- -----
- 通过文件或流（数据库）保存或打开图形
- 打印图形
- -----
- 插入图片
- 像图形一样缩放、旋转、拖动图片
- 输出图形为 WMF、Bitmap、Jpg、Dxf 格式的文件
- 支持用户自定义属性

## 2.2 程序截图

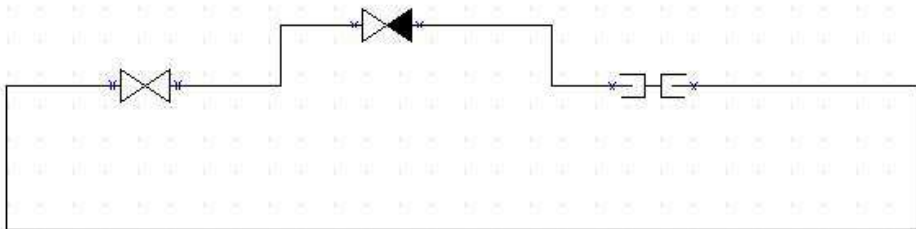


房屋平面图，创建简单、快速，没有使用库

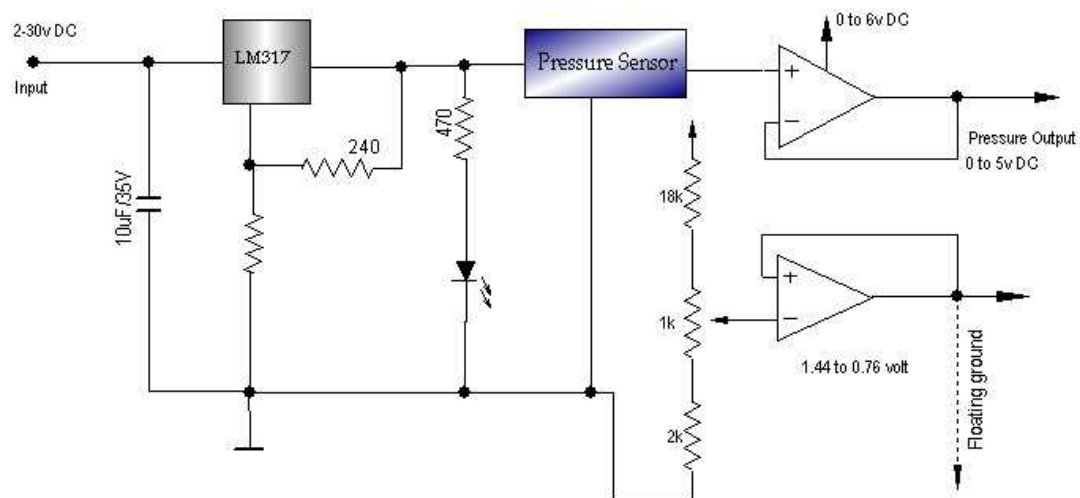


CNC 公司采用 TCAD 设计的矢量图-龙

## TCAD for Delphi & C++ Builder & Kylix & Vcl.net

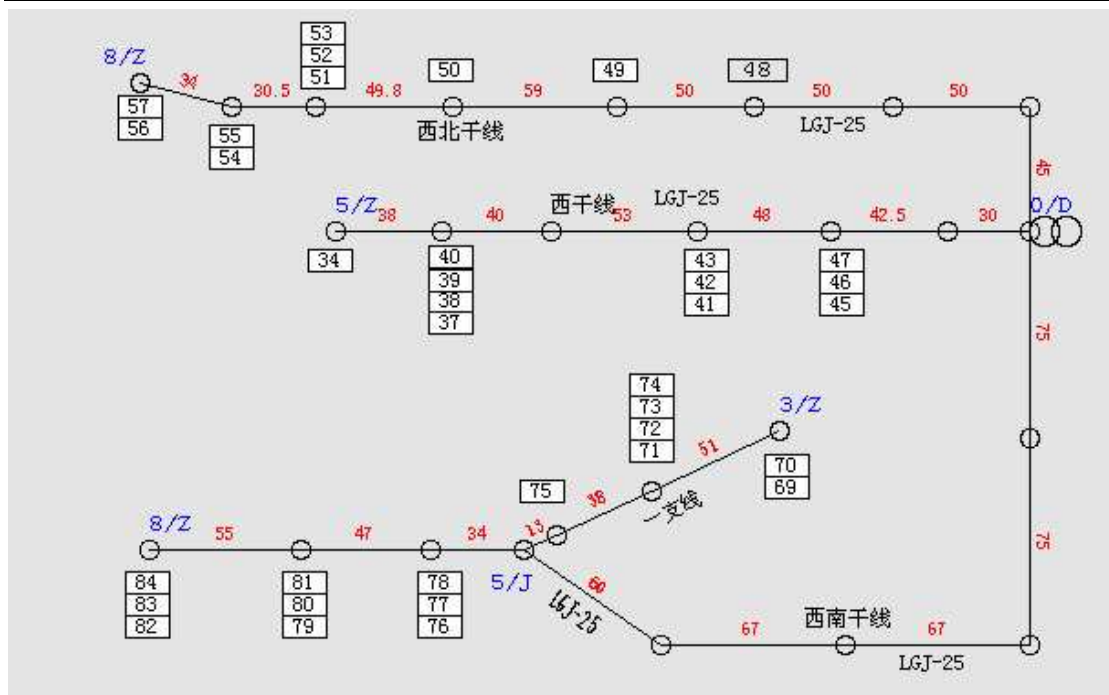


水利设计

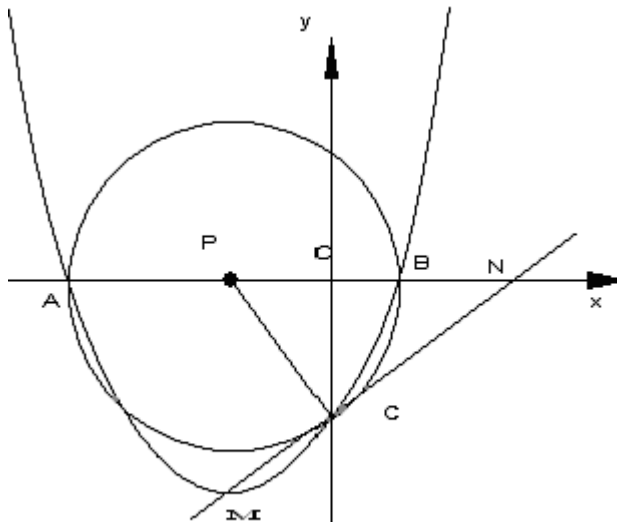


电子线路图，用到库功能，支持连接线。TCAD的连接线功能非常强大

TCAD for Delphi & C++ Builder & Kylix & Vcl.net



项目：低压电力设备和线损管理  
单位：西安工业大学



假如你是老师或学生，TCAD能帮助你画数学图形，你可以拷贝图形到Microsoft Word，使用数学图形库提高工作效率

### 3 定义

```
TMyPoint=record  
    x:single;  
    y:single;  
End;
```

```
TDrawTool  
=(SpLinkLine,SpSelecting,SpRotateSelecting,SpLine,SpRuleLine,SpPolyLine,SpRectangle,Sp  
pLinkPoint,SpLineLinkLine,SpEllipse,SpText,SpImage,SpPolygon,SpPolyBezier,SpElliArc,Sp  
Close,SpPolylinePolygonPointRemoving,SpPolylinePolygonointAdding,SpDragWhole,  
SpBrowse, SpFreeLine,SpBLine);
```

```
TPageStyle = (A0,A1,A2,A3, A4, A5, B3, B4, B5, CustomerPage);
```

```
TDragMode= (dgHorz,dgVert,dgBoth);
```

```
TXYMode =(Mode0,Mode1,Mode2,Mode3);
```

```
    Mode0 = TopLeft  
    Mode1 = BottomLeft  
    Mode2 = BottomRight  
    Mode3 = TopRight
```

```
TGridType =(gPixel,gLine,gNone);
```

```
TUnits = (pixel, mm,dm,m, inch);
```

```
TBkBitmapMode = (Tiled,Stretch, Center, LeftTop);
```

```
TArrowStyle = (ANone,ALeft,ARight,ADouble);
```

```
TBlockLayerMode= (Merge,Import);
```

```
TGradientStyle = ( gsRadialC, gsRadialT, gsRadialB, gsRadialL,  
    gsRadialR, gsRadialTL, gsRadialTR, gsRadialBL, gsRadialBR, gsLinearH,  
    gsLinearV, gsReflectedH, gsReflectedV, gsDiagonalLF, gsDiagonalLB,  
    gsDiagonalRF, gsDiagonalRB, gsArrowL, gsArrowR, gsArrowU, gsArrowD,  
    gsDiamond, gsButterfly,gsNone);
```

```
TArrUserDataRecord=array of Record
```

```
    Key:string;  
    Value:String;
```

```
END;
```

## TCAD for Delphi & C++ Builder & Kylix & Vcl.net

TPenStyleJoinType = (psjtROUND,psjtBEVEL,psjtMITER,psjtMASK);

TPenStyleEndCapType = (psectROUND,psectSQUARE,psectFLAT,psectMASK);

## 4 TMyCAD

TMyCAD 是一个功能强大的二维画图组件，她可以在 Delphi 和 C++ Builer 开发环境中使用。

### 4.1 类图

#### 4.1.1 属性



#### 4.1.2 事件

↔ TMyCAD	
<b>events</b>	
*	OnActionToolToSelecting: TActionToolToSelecting
*	OnClick: TNotifyEvent
*	OnDbClick: TNotifyEvent
*	OnDeleteLayer: TLayerOp
*	OnDragDrop: TNotifyEvent
*	OnDragOver: TNotifyEvent
*	OnMouseDown: TNotifyEvent
*	OnMouseEnter: TNotifyEvent
*	OnMouseEnterShape: TEnterLeaveShape
*	OnMouseLeave: TNotifyEvent
*	OnMouseLeaveShape: TEnterLeaveShape
*	OnMouseMove: TNotifyEvent
*	OnMouseUp: TNotifyEvent
*	OnNewLayer: TLayerOp
*	OnPaint: TNotifyEvent
*	OnShapeAdded: TShapeAdded
*	OnShapeCodeDragging: TShapeCodeDraggingSizing
*	OnShapeCodeRotating: TShapeCodeRotating
*	OnShapeDeleted: TShapeDeleted
*	OnShapeMouseDragged: TShapeMouseDraggingSizingRotating
*	OnShapeMouseDragging: TShapeMouseDraggingSizingRotating
*	OnShapeMouseResized: TShapeMouseDraggingSizingRotating
*	OnShapeMouseResizing: TShapeMouseDraggingSizingRotating
*	OnShapeMouseRotated: TShapeMouseDraggingSizingRotating
*	OnShapeMouseRotating: TShapeMouseDraggingSizingRotating
*	OnShapeSelected: TShapeSelected



### 4.1.3 方法

 TMyCAD	
operations	
+ Create(..)	+ GetShapeByNo(..): TMyShape
+ Destroy	+ GetShapeNoByld(..): Integer
+ AddBlockfromTCADFile(..): Boolean	+ GetShapesCount: Integer
+ AddImageShapeByCode(..): Integer	+ GetShapesCountInALayer(..): Integer
+ AddShape(..): Boolean	+ GroupWorkingShape: Integer
+ AddShapeByCode(..): Integer	+ InVisibleLayerByld(..)
+ AddUserDefineShapefromLib(..): Integer	+ InVisibleLayerByName(..)
+ AlignBottom	+ IsLinked(..): Integer
+ AlignLeft	+ IsTCADFile(..): Boolean
+ AlignRight	+ IsVisibleLayerByld(..): Boolean
+ AlignTop	+ LoadFromFile(..): Boolean
+ BringToFront(..)	+ LoadFromStream(..): Boolean
+ BringToFrontByStep(..)	+ LockUnlockforShapes(..)
+ ClearAllUndoStuff	+ MergeLayers(..): Boolean
+ Copy	+ Move(..)
+ CopyToClipboardAsWmf	+ NewLayer(..): Integer
+ CreateLink(..): Integer	+ Paste
+ Cut	+ PopFromUndoRedoShapeList: Integer
+ DeleteAllLayers: Boolean	+ Print(..)
+ DeleteAllShapes	+ PrintPreview(..)
+ DeleteLayerByld(..): Boolean	+ ReleaseCursorClipArea
+ DeleteLayerByName(..): Boolean	+ RenameShapeName(..)
+ DeleteSelectedShapes: Boolean	+ Resize
+ DeleteShapeByld(..): Boolean	+ Rotate(..)
+ DeSelectedAllShapesByCode	+ SaveToBmp(..)
+ DrawAllShape(..)	+ SaveToBmp(..)
+ FlipHoriz(..)	+ SaveToDxf(..)
+ FlipVert(..)	+ SaveToFile(..): Boolean
+ GetLayerldByName(..): Integer	+ SaveToFileold(..): Boolean
+ GetLayerldByNo(..): Integer	+ SaveToJpg(..)
+ GetLayerNameByld(..): string	+ SaveToStream(..): Boolean
+ GetLayerNoByld(..): Integer	+ SaveToWmf(..)
+ GetLayerNoByName(..): Byte	+ SelectAllShapes
+ GetLayersCount: Byte	+ SelectShapeByCode(..): Boolean
+ GetMaxLayerld: Integer	+ SendToBack(..)
+ GetMemShapesCount: Integer	+ SendToBackByStep(..)
+ GetMyPointByMode(..): TMyPoint	+ SetLayerNameByld(..)
+ GetPointByMode(..): TPoint	+ SetLayerNameByName(..)
+ GetRA1A2By3Points(..)	+ SetMyImage(..): Boolean
+ GetSelectedShape: TMyShape	+ SizeShape(..)
+ GetSelectedShapes: TArrMyShape	+ UnGroupShape(..)
+ GetSelectedShapesCount: Integer	+ VisibleAllLayer
+ GetShapeByld(..): TMyShape	+ VisibleLayerByld(..)
+ GetShapeByName(..): TMyShape	+ VisibleLayerByName(..)

## 4.2 事件

### 4.2.1 OnActionToolToSelecting

**TActionToolToSelecting** = procedure() of object;  
**property** OnActionToolToSelecting: TactionToolToSelecting

**描述:**

当图形工具返回到选择模式时触发OnActionToolToSelecting事件

**例子:**

```
procedure TMainFrm.MyCAD1ActionToolToSelecting;  
begin  
    SelectBtn.Down:=true;  
end;
```

### 4.2.2 OnClick

请查看Delphi 或 C++ Builder帮助文件

### 4.2.3 OnDoubleClick

请查看Delphi 或 C++ Builder帮助文件

### 4.2.4 OnDeleteLayer

**TLayerOp** = procedure(LayerId: integer; LayerName: string; Visible: Boolean) of object;  
**property** OnDeleteLayer:TlayerOp;

**描述:**

当删除一个图层时触发OnDeleteLayer事件

**参数:**

**LayerId**

图层的标识符

**LayerName**

图层的名称

**Visible**

#### 4.2.5 OnDragDrop

**Delphi 语法**

```
type TDragDropEvent = procedure(Sender, Source: TObject; X, Y: Integer) of object;  
property OnDragDrop: TdragDropEvent;
```

**C++ 语法**

```
typedef void __fastcall (__closure *TDragDropEvent)(System::TObject* Sender,  
System::TObject* Source, int X, int Y);  
__property TDragDropEvent OnDragDrop = {read=FOnDragDrop,  
write=FOnDragDrop};
```

**描述:**

当拖动一个对象放下时将触发OnDragDrop事件

当用户放下一个对象时使用OnDragDrop事件指定发生什么，OnDragDrop事件的Source参数是指被放下的对象，Sender参数是指接收放下对象的对象，X和Y参数是指在控制面板上鼠标的坐标。

#### 4.2.6 OnDragOver

**Delphi 语法**

```
Type TDragOverEvent = procedure(Sender, Source: TObject; X, Y: Integer; State:  
TDragState; varAccept: Boolean) of object;  
property OnDragOver: TdragOverEvnet;
```

**C++ 语法**

```
typedef void __fastcall (__closure *TDragOverEvent)(System::TObject* Sender,  
System::TObject* Source, int X, int Y, TDragState State, bool &Accept);  
__property TDragOverEvent OnDragOver = {read=FOnDragOver,  
write=FOnDragOver};
```

**描述:**

当拖动一个对象经过时触发OnDragOver事件，可在此事件中判断是否拖入些对象中。

在OnDragOver事件中，当设置Accept参数为false时拒绝拖入对象。当Accept设置为true时允许拖入对象，通过改变图形的光标来指示是否可拖放对象，可以在OnDragOver设定光标显示。OnDragDrop事件的Source参数是指被放下的对象，Sender参数是指接收放下对象的对象，X和Y参数是指在控制面板上鼠标的坐标。State参数指定在面板上如何移动。

**注意:** 在OnDragOver事件中Accept事件默认值为True，但是如果没有指定OnDragOver事

## TCAD for Delphi & C++ Builder & Kylix & Vcl.net

---

件，当Accept设置为False时是拒绝拖入对象。

### 4.2.7 OnMouseDown

请查看Delphi 或 C++ Builder帮助文件

### 4.2.8 OnMouseEnter

**property** OnMouseEnter:TnotifyEvent

**描述:**

当鼠标进入TMyCAD时触发OnMouseEnter事件

**例子:**

```
procedure TForm1.MyCAD1MouseEnter(Sender: TObject);  
begin  
    Memo1.Lines.Add('-----Enter TCAD Event-----');  
end;
```

### 4.2.9 OnMouseEnterShape

TEnterLeaveShape = procedure (AShape:TMyShape) of object;

**property** OnMouseEnterShape:TNotifyEvent

**描述:**

当鼠标进入某个图形时将触发OnMouseEnterShape事件

**例子:**

```
procedure TForm1.MyCAD1MouseEnterShape(AShape: TMyShape);  
begin  
    Memo1.Lines.Add('-----Enter Shape Event-----');  
    Memo1.Lines.Add('Enter shape, id is : ' + inttostr(AShape.ShapeID));  
end;
```

#### 4. 2. 10 OnMouseLeave

**property** OnMouseLeave:TNotifyEvent

**描述:**

当鼠标离开TMyCAD时触发OnMouseLeave事件

**例子:**

```
procedure TForm1.MyCAD1MouseLeave(Sender: TObject);  
begin  
    Memo1.Lines.Add('-----Leave TCAD Event-----');  
end;
```

#### 4. 2. 11 OnMouseLeaveShape

TEnterLeaveShape = procedure (AShape:TMyShape) of object;

**property** OnMouseLeaveShape:TenterLeaveShape

**描述:**

当鼠标离开某个图形时将触发OnMouseLeaveShape事件

**例子:**

```
procedure TForm1.MyCAD1MouseLeaveShape(ShapeID, LayId: Integer;  
AShape: TMyShape);  
begin  
    Memo1.Lines.Add('----- Leave Shape Event-----');  
    Memo1.Lines.Add('Leave shape, id is : ' + inttostr(AShape.ShapeID));  
end;
```

#### 4. 2. 12 OnMouseMove

请查看Delphi 或 C++ Builder帮助文件

#### 4. 2. 13 OnMouseUp

请查看Delphi 或 C++ Builder帮助文件

#### 4.2.14 OnNewLayer

**TLayerOp** = procedure(LayerId: integer; LayerName: string; Visible: Boolean) of object;  
**property** OnNewLayer: TLayerOp ;

**描述:**

当增加一个图层时触发OnNewLayer事件

#### 4.2.15 OnPaint

请查看Delphi 或 C++ Builder帮助文件

#### 4.2.16 OnShapeAdded

**TShapeAdded** = procedure(LayerId: integer; ShapeID: integer;ShapeName:string;  
AShape:  
TMyShape; ShapeCount: integer) of object;

**描述:**

当增加一个图形时触发OnShapeAdded事件

**例子:**

```
procedure TForm1.MyCAD1ShapeAdded(var AShape: TMyShape;ShapeCount:  
Integer);  
begin  
    Memo1.Lines.Add("You Add a Shape,its id is :" + inttostr(ShapeID) + ',in  
    Layer ' +inttostr(LayerId) + ', it is a ' + AShape.ClassName + ',Now CAD has  
    ' +inttostr(ShapeCount) + 'Shapes')  
end;
```

#### 4.2.17 OnShapeCodeDragging

**TShapeCodeDraggingSizing** = procedure (AShape:TMyShape; dx,dy:Single) of object;  
**property** OnShapeCodeDragging

**描述:**

当通过代码拖动一个图形时触发OnShapeCodeDragging事件

**实例:**

```
procedure TForm1.MyCAD1ShapeCodeRotating(AShape: TMyShape;dx,dy:
single);
begin
    Memo1.Lines.Add('-----code Drag Event-----');
    Memo1.Lines.Add('Shapeld is: '+Inttostr(AShape.Shapeld)+' '+AShape.Name
+' rotating ');
end;
```

#### 4.2.18 OnShapeCodeRotating

**TShapeCodeRotating** = procedure (AShape:TMyShape; AAngle:Single) of object;  
**property** OnShapeCodeRotating

**描述:**

当通过代码旋转图形时触发OnShapeCodeRotating事件

**实例:**

```
procedure
TForm1.MyCAD1ShapeCodeDragging( AShape:TMyShape;AAngle:single);
begin
    Memo1.Lines.Add('-----code Rotate Event-----');
    Memo1.Lines.Add('Shapeld is: '+Inttostr(AShape.Shapeld)+' '+AShape.Name
+' rotating ');
end;
```

#### 4. 2. 19 OnShapeDeleted

**TShapeDeleted** = procedure (ShapeID:integer;LayerId:integer;AShape:TMyShape) of object ;

**property** OnShapeDeleted: TShapeDeleted ;

**描述:**

当删除一个图形时触发OnShapeDeleted事件

#### 4. 2. 20 OnShapeMouseDragged

**TShapeDraggingSizingRotated** = procedure (AShape:TMyShape;FromPoint:TPoint;ToPoint:TPoint ) of object;

**property** OnShapeMouseDragged:TshapeDraggingSizingRotating

**描述:**

当用鼠标拖动图形时触发OnShapeMouseDragged事件


**实例:**

```
procedure TForm1.MyCAD1ShapeMouseDragged( AShape: TMyShape;
FromPoint, ToPoint: TPoint);
begin
    Memo1.Lines.Add('-----Dragged Event-----');
    Memo1.Lines.Add('Shapeld is: '+Inttostr(AShape.ShapeId)+AShape.Name
+' dragged ,'+From x:'+ inttostr(FromPoint.x)+' y:' +inttostr(FromPoint.y)+'To
x:'+ inttostr(ToPoint.x)+' y:' +inttostr(ToPoint.y));
end;
```



#### 4. 2. 21 OnShapeMouseDowning

**TShapeDraggingSizingRotated** = procedure  
(AShape:TMyShape;FromPoint:TPoint;var ToPoint:TPoint ) of object;

**property** OnShapeMouseDowning:TshapeDraggingSizingRotating 

**描述:**


当用鼠标拖动图形时触发OnShapeMouseDowning事件

**实例:**

```
procedure TForm1.MyCAD1ShapeMouseDowning(AShape: TMyShape;  
FromPoint:TPoint; var ToPoint: TPoint);  
begin  
    Memo1.Lines.Add('-----Dragging Event-----');  
    Memo1.Lines.Add('ShapelId is: '+Inttostr(AShape.ShapelId)+AShape.Name  
    +' are dragging!')  
end;
```

#### 4. 2. 22 OnShapeMouseResized

**TShapeDraggingSizingRotating**= procedure  
(AShape:TMyShape;FromPoint:TPoint;var ToPoint:TPoint ) of object;

**property** OnShapeMouseResizing:TshapeDraggingSizingRotating 

**描述:**


当用鼠标改变图形大小后触发OnShapeMouseResized事件

**实例:**

```
procedure TForm1.MyCAD1ShapeMouseResized(AShape: TMyShape;  
FromPoint:TPoint; var ToPoint: TPoint);  
begin  
    Memo1.Lines.Add('-----Resized Event-----');  
    Memo1.Lines.Add('ShapelId is: '+Inttostr(AShape.ShapelId)+AShape.Name  
    +' resized, '+ 'From x:' + intostr(FromPoint.x)+ ' y:' + intostr(FromPoint.y)+  
    'To x:' + intostr(ToPoint.x)+ ' y:' + intostr(ToPoint.y));  
end;
```

#### 4. 2. 23 OnShapeMouseResizing

**TShapeDraggingSizingRotating** = procedure  
(AShape:TMyShape;FromPoint:TPoint;var ToPoint:TPoint ) of object;

**property** OnShapeMouseResizing:TshapeDraggingSizingRotating 

**描述:**


当通过鼠标更改图形大小时触发OnShapeMouseResizing事件

**实例:**

```
procedure TForm1.MyCAD1ShapeMouseResizing(AShape: TMyShape;  
FromPoint:TPoint; var ToPoint: TPoint);  
begin  
    Memo1.Lines.Add('-----Resizing Event-----');  
    Memo1.Lines.Add('Shapeld is: '+Inttostr(AShape.Shapeld)+AShape.Name  
    +' resizing ,'+From x:+ inttostr(FromPoint.x)+' y: '+inttostr(FromPoint.y)+  
    'To x:+ inttostr(ToPoint.x)+' y: '+inttostr(ToPoint.y));  
end;
```

#### 4. 2. 24 OnShapeMouseRotated

**TShapeDraggingSizingRotating**= procedure  
(AShape:TMyShape;FromPoint:TPoint; **var** ToPoint:TPoint ) of object;

**property** OnShapeMouseRotated:TshapeDraggingSizingRotating 

**描述:**

当通过鼠标旋转图形后触发OnShapeMouseRotated事件


**例子:**

```
procedure TForm1.MyCAD1ShapeMouseRotated(AShape: TMyShape;  
FromPoint:TPoint; var ToPoint: TPoint);  
begin  
    Memo1.Lines.Add('-----Rotated Event-----');  
    Memo1.Lines.Add('Shapeld is: '+Inttostr(AShape.Shapeld)+AShape.Name +'  
    rotated ');
```

end;

#### 4.2.25 OnShapeMouseRotating

**TShapeDraggingSizingRotating** = procedure  
(AShape:TMyShape;FromPoint:TPoint; var ToPoint:TPoint ) of object;

**property** OnShapeMouseRotating:TshapeDraggingSizingRotating 

**描述:**

当通过鼠标旋转一个图形时触发OnShapeMouseRotating事件

**实例:**

```
procedure TForm1.MyCAD1ShapeMouseRotating(AShape: TMyShape;  
FromPoint, var ToPoint: TPoint);  
begin  
    Memo1.Lines.Add('-----Rotating Event-----');  
    Memo1.Lines.Add('ShapeId is: '+Inttostr(AShape.ShapeId)+AShape.Name  
    +' rotating ');  
end;
```

#### 4.2.26 OnShapeSelected

**TShapeSelected** = procedure (LastSelectedShape:TMyShape; SelectedShape:  
TMyShape) of object;

**property** OnShapeSelected: TShapeSelected ;

**描述:**

当选择一个图形时触发OnShapeSelected事件，同时可得到图形的信息。

**实例:**

```
procedure TMainFrm.MyCAD1ShapeSelected(LastSelectedShape,  
SelectedShape: TMyShape);  
begin  
    Memo1.Lines.Add('-----Select Event-----');  
    if LastSelectedShape = nil then  
        Memo1.Lines.Add('No last shape selected')
```

```
else
    Memo1.Lines.Add(' Last shape id is ' +
    inttostr>LastSelectedShape.ShapeID)+
    ',in Layer ' + inttostr>LastSelectedShape.LayerId) + ', it is a ' +
    LastSelectedShape.ClassName);
    Memo1.Lines.Add('Now you selected ' + inttostr>SelectedShape.ShapeID)+
    ',in Layer ' + inttostr>LayerId) + ', it is a ' + SelectedShape.ClassName);
end;
```

#### 4.2.27 OnChildShapeSelected

**TChildhapeSelected** = procedure (SelectedChildShape: TMyShape; var AcceptSelect:boolean) of object;

**property** OnChildShapeSelected: TchildhapeSelected

##### 描述:

操作对象为组合图形，当单击子图形时触发的事件，可以得到子图形的信息，及是否允许选择子图形。

##### 实例:

```
procedure TMainFrm.MyCAD1ChildShapeSelected(SelectedChildShape: TMyShape; var
AcceptSelect: Boolean);
begin
    AcceptSelect:=false;
end;
```

#### 4.2.28 OnChildShapeSelected

**TWholeDragOperation** = procedure of object;

**property** OnWholeDragged: TWholeDragOperation

##### 描述:

整体拖动后触发此事件。

#### 4.2.22 OnShapeMouseResized

**TShapeDraggingSizingRotating**= procedure

(AShape:TMyShape;FromPoint:TPoint;var ToPoint:TPoint ) of object;

**property** OnShapeMouseResizeFinished:TshapeDraggingSizingRotating



**描述:**

当用鼠标改变图形大小完成后触发OnShapeMouseResized事件

### 4.3 方法

#### 4.3.1 AddBlockfromTCADFile

**function** AddBlockfromTCADFile(const

FileName:string;BlockLayerMode:TBlockLayerMode):Boolean;

**描述:**

从TCAD文件中加载一个模板

**参数:**

FileName: TCAD文件路径

BlockLayerMode: 添加的模式, 包括合并、追加、导入三种模式, 具体请查看 TblockLayerMode定义

**返回值:**

返回true表示增加成功, 返回false表示增加失败

#### 4.3.2 AddImageShapeByCode

**function**

AddImageShapeByCode(TheName:String;LeftTop:TMyPoint;Abitmap:Tbitmap):integer;

**描述:**

通过代码增加一个图片, 如果你想增加其它图形, 请使用AddShapeByCode 或者 AddCmbShapeByCode过程

**参数:**

TheName: 图片名称

LeftTop: 图片的左上角位置

ABitmap: 想增加到TCAD里的图像

**返回值:**

返回-1增加失败, 返回其它值 (>=0) 增加成功

**例子:**

```
var
  myBitmap:TBitmap;
begin
  {Create a temp bitmap to load form a disk file}
  MyBitmap := TBitmap.Create;
  {Load from files}
  Mybitmap.LoadFromFile( ExtractFilePath(Application.ExeName)
  +'images\1'+'.bmp');
  {a layer0 already exist by default}
  {Add Image into TMyCAD by code}
  MyCAD1.AddImageShapeByCode('MyImage',Point(50,100),MyBitmap);
  {free it because no use}
  MyBitmap.Free;
end;
```

### 4.3.3 AddShapeByCode

**function**

```
AddShapeByCode( Owner:TComponent;ShapeStyle:TDrawTool;ShapeName:string;
ThePoints: array of TMyPoint;TheAngle:Extended=0;OnlyForText:String=""):integer;
```

**描述:**

通过代码增加一个图形, 在自动画图方面非常有用, 假如你想增加图片, 请使用 AddImageByCode方法

**参数:**

Owner: TCAD实例

ShapeStyle: 请查看TDrawTool类型定义

ShapeName: 图形名称

ThePoints: 坐标数组

TheAngle: 图形的角度

OnlyForText: 只在TMyText时使用, 字符串支持双字节

**返回值:**

-1: add failed

else:the ShapeID

**实例:**

```
MyCAD1.AddShapeByCode(myCAD1,SpEllipse, 'EllipseShape',[MyPoint(231,211), MyPoint(340,211),MyPoint(340, 250),MyPoint(231,250)]);
```

### 4.3.4 AddUserDefineShapefromLib

**function**

```
AddUserDefineShapefromLib(ALibManager:TLibManager;UDShapeName:string;CenterX,CenterY:integer;OwnerCAD:TMyCAD): Integer;
```

**描述:**

从库中得到一个用户自定义图形增加到TCAD中，这是TCAD一个重要的功能

**参数:**

ALibManager: 库的实例  
UDShapeName: 用户自定义图形名称  
CenterX, CenterY: 增加图形的中心位置  
OwnerCAD: 要增加到的TCAD实例

**返回值:**

返回-1增加失败，其它 (>=0) 返回增加图形的标识符

**实例:**

```
var  
  mLibManager:TLibManager;  
begin  
  if (FileExists(FileName)) and (IsTCADLibFile(FileName)) then  
  begin  
    mLibManager:=TLibManager.Create;  
    mLibManager.LoadFromFile(LibFileName);  
    MyCAD1.AddUserDefineShapefromLib( mLibManager,  
    'MyShapename',100,100,MyCAD1);  
    mLibManager.Free;  
  end;
```

end;

### 4.3.5 AlignBottom

**procedure** AlignBottom;

**描述:**

当你选择多个图形时，这个过程能使它们在底端对齐

**例子:**

MyCAD1.AlignBottom;

### 4.3.6 AlignHorizontalCenter

**procedure** AlignHorizontalCenter;

**描述:**

当你选择多个图形时，这个过程能使它们水平居中

**例子:**

MyCAD1.AlignHorizontalCenter;

### 4.3.7 AlignLeft

**procedure** AlignLeft;

**描述:**

当你选择多个图形时，这个过程能使它们在左端对齐

**例子:**

MyCAD1.AlignLeft

### 4.3.8 AlignRight

**procedure** AlignRight;

**描述:**

当你选择多个图形时，这个过程能使它们在右端对齐

**例子:**

MyCAD1.AlignRight;



### 4.3.9 AlignTop

**procedure** AlignTop;

**描述:**

当你选择多个图形时，这个过程能使它们在顶端对齐

**例子:**

MyCAD1.AlignTop;

### 4.3.10 AlignVerticalCenter

**procedure** AligVerticalCenter;

**描述:**

当你选择多个图形时，这个过程能使它们垂直居中

**例子:**

MyCAD1.AlignVerticalCenter;

### 4.3.11 BringToFront

**procedure** BringToFront(AShape:TMyShape;NeedSave:boolean=true);

**描述:**

移动选择的图形到最前

**参数:**

Ashape: 要移到最前的图形

NeedSave: 是否放入Undo队列，缺省为true

**例子:**

MyCAD1.BringToFront(MyCAD1.GetSelectedShape);

### 4.3.12 BringToFrontByStep

**procedure** BringToFrontByStep(AShape:TMyShape;NeedSave:boolean=true);

**描述:**

选择的图形向前移一层

**参数:**

Ashape: 要移的图形

NeedSave: 是否保存到undo队列, 用于回退

**例子:**

```
MyCAD1.BringToFrontByStep(MyCAD1.GetSelectedShape);
```

### 4.3.13 ClearAllUndoStuff

```
procedure ClearAllUndoStuff
```

**描述:**

在内存中清空所有的undo队列。在开始一个新绘图时必须使用

**实例:**

```
MyCAD1.ClearAllUndoStuff;
```

### 4.3.14 ClosePolygon

```
procedure ClosePolygon
```

**描述:**

结束当前TMyPolygon绘制

**实例:**

```
MyCAD1. ClosePolygon;
```

### 4.3.15 Copy

```
procedure Copy();
```

**描述:**

在内存中保存被选中的图形, 同时它们也以bitmap图片格式保存在剪贴板中

**实例:**

```
MyCAD1.SelectShapeByCode(0);
```

```
//Select other shape  
MyCAD1.SelectShapeByCode(1,false);  
MyCAD1.Copy;  
MyCAD1.Paste;
```

#### 4.3.16 CopyToClipboardAsWmf

**procedure** CopyToClipboardAsWmf

**描述:**

以WMF格式在剪贴板中保存选择的图形

**实例:**

```
MyCAD1.SelectShapeByCode(0);  
//Select other shape  
MyCAD1.SelectShapeByCode(1,false);  
MyCAD1.CopyToClipboardAsWmf;
```

#### 4.3.17 Create

**constructor** Create(AOwner:Tcomponent);

**描述:**

TMyCAD构造函数，会创建一个名称为Layer0的图层

**实例:**

```
var  
MyCAD:TMyCAD;  
begin  
MyCAD:=TMyCAD.Create(Form1);  
MyCAD.Parent:=Form1;  
end;
```

#### 4.3.18 CreateLink

## TCAD for Delphi & C++ Builder & Kylix & Vcl.net

---

```
function CreateLink(ALinkShapeName:string;SrcShape:TMyShape;  
SrcLinkPtlId:integer;DestShape:TMyShape;DestLinkPtlId:integer): Integer;
```

### 描述:

在源图形和目标图形之间创建一条连接线

### 参数:

**ALinkShapeName:** 连接线名称

**SrcShape:** 源图形, 这是一个有一个连接点的用户自定义图形

**SrcLinkPtlId:** 源图形的连接点Id

**DestShape:** 目标图形, 这是一个有一个连接点的用户自定义图形

**DestLinkPtlId:** 目标图形的连接点Id

### 返回值:

当返回-1时表示增加失败, 返回其它 ( $\geq 0$ ) 值时表示增加成功, 其返回值代表连接线的ShapeId

### 4.3.19 Cut

```
procedure Cut();
```

### 描述:

在内存中保存选择的图形, 删除被选择中的图形, 同时也以bitmap图片格式保存在剪贴板中

### 实例:

```
MyCAD1.SelectShapeByCode(0);
```

```
MyCAD1.Cut;
```

```
MyCAD1.Paste;
```

### 4.3.20 DeleteAllLayers

```
procedue DeleteAllLayers():Boolean;
```

### 描述:

删除所有的图层和图形, 当前的图层标识符是-1

### 返回值:

返回true表示所在图层已删除, 返回false表示删除失败

### 4.3.21 DeleteAllShapes

```
procedure DeleteAllShapes;
```

**描述:**

删除所有的图形，图层仍旧存在

#### 4.3.22 DeleteLayerByID

**function** DeleteLayerByID(ALayerID:integer):Boolean;

**描述:**

通过图层标识符来删除图层，如果所给出的图层标识符不存在，返回false，这个函数会删除该图层所有的图形

#### 4.3.23 DeleteLayerByName

**function** DeleteLayerByName(ALayerName:string):Boolean;

**描述:**

通过图层名称来删除图层，如果图层名称不存在，返回false，这个函数会删除该图层上的所有图形

#### 4.3.24 DeleteSelectedShape

**function** DeleteSelectedShape:boolean;

**描述:**

删除当前所选择的图形

**返回值:**

返回true表示删除成功，返回false表示删除失败

#### 4.3.25 DeleteShapeByID

**function** DeleteShapeByID(ShapeID:Integer):Boolean;

**描述:**

通过图形标识符删除一个图形或一个组合图形

**返回值:**

返回true删除成功，返回false删除失败

**实例:**

DeleteShapeByID(1);

### 4.3.26 DeSelectedAllShapesByCode

**procedure** DeselectedAllShapesByCode;

**描述:**

执行这个命令后，所有图形处于没有被选择状态

**实例:**

MyCAD1.DeselectedAllShapesByCode;

### 4.3.27 Destroy

**destructor** Destroy;

**描述:**

TMyCAD的析构函数，所有TMyCAD的图层及图形都会释放

### 4.3.28 DrawAllShape

**procedure** DrawAllShape(MyCanvas:Tcanvas;ARect:TRect);

**描述:**

指定区域画出所有图形

**参数:**

MyCanvas: 所要重画的画布

ARect: 指定的区域，为了快速重画，不再这个区域内的图形不会被画出来

### 4.3.29 FlipHoriz

**procedure** FlipHoriz(Ashape:TmyShape);



**描述:**

水平翻转一个图形，假如是一个组合图形，子图形也同时翻转

### 4.3.30 FlipVert

**procedure** FlipVert(Ashape:TmyShape);



**描述:**

垂直翻转一个图形，假如是一个组合图形，子图形也同时翻转

### 4.3.31 GetLayerIdByName

```
function GetLayerIdByName(ALayerName:string):integer;
```

**描述:**

通过图层名称得到图层标识符，如果参数ALayerName不存在，返回-1

**实例:**

```
ShowMessage(' The Layers name is:'+MyName' +' its layer Id is:
'+Inttostr(MyCAD1.GetLayerIdByName('MyName')));
```

### 4.3.32 GetLayerIdByNo

```
function GetLayerIdByNo(ALayerNo:integer):integer;
```

**描述:**

通过序号得到图层标识符，如果序号不存在，返回-1

**注意:** 该值从零开始

**实例:**

```
var
    a:integer;
begin
    a:=GetLayerIdByNo(4);
    ShowMessage(' The Layers No is:'+4' +' its layer Id is: '+Inttostr(a));
end;
```

### 4.3.33 GetLayerNameById

```
function GetLayerNameById(ALayerId:integer):integer;
```

**描述:**

通过标识符得到图层Id，如果LayerID不存在，返回-1

**实例:**

```
procedure Form1.Button1Click(Send:TObject)
begin
    ShowMessage(' The Layers Id is:'+8' +' Name is: '+GetLayerNameById(8));
```

```
end;
```

#### 4.3.34 GetLayerNoById

```
function GetLayerNoById(ALayerId:integer):integer;
```

**描述:**

通过标识符得到图层序号，如果标识符不存在，返回-1

**实例:**

```
procedure Form1.Button1Click(Send:TObject)
begin
  ShowMessage(' The Layers Id is:'+4' + ' its layer No is: '+Inttostr(MyCAD1.
    GetLayerNoById(4)));
end;
```

#### 4.3.35 GetLayerNoByName

```
function GetLayerNoByName(ALayerName:string):integer;
```

**描述:**

通过图层名称得到LayerNo，如果给出的图层名称不存在，返回-1

**实例:**

```
procedure Form1.Button1Click(Send:TObject)
begin
  with MyCAD1 do
    ShowMessage(' The Layers name is:'+MyName' + ' layer No is:
      '+Inttostr(GetLayerNoByName('MyName')));
end;
```

#### 4.3.36 GetLayersCount

```
function GetLayerCount():integer;
```

**描述:**

得到TMyCAD当前的图层总数

**实例:**

```
procedure Form1.Button1Click(Send:TObject)
begin
  ShowMessage(' There are '+Inttostr(MyCAD1.GetLayersCount)+' in TMyCAD1!');
end;
```



### 4.3.37 GetMaxLayerId

**function** GetMaxLayerId():integer;

**描述:**

得到最大的LayerId的值，它不是图层的总数，如果当前没有任何图层存在，将返回-1

**实例:**

```
procedure Form1.Button1Click(Sender:TObject)
begin
  ShowMessage(' The max layer Id is '+Inttostr(MyCAD1.GetMaxLayerID));
end;
```

### 4.3.38 GetMemShapesCount

**function** GetMemShapesCount:integer;

**描述:**

得到撤消队列中的图形数量

**实例:**

```
EditPaste1.Enabled:= MyCAD1.GetMemShapesCount > 0;
```

### 4.3.39 GetSelectedShape

**function** GetSelectedShape:TmyShape;

**描述:**

得到选择的图形

**返回值:**

当选择了多个图形时返回nil；当选择的是一个组合图形则返回父图形

**实例:**

```
Shape1:=MyCAD1.GetSelectedShape;
```

### 4.3.40 GetSelectedShapes

**function** GetSelectedShapes:TarrMyShape;

**描述:**

得到选择的图形

**返回值:**

假如选择多个图形，使用这个函数得到图形数组

**实例:**

```
Shape1:=MyCAD1.GetSelectedShapes[0];
```

#### 4.3.41 GetSelectedShapesCount

```
function GetSelectedShapesCount:integer;
```

**描述:**

得到当前选择图形的个数，组合图形算一个图形

#### 4.3.42 GetShapeByID

```
function GetShapeByID(Id:integer):TmyShape;
```

**描述:**

通过ShapeId得到图形实例

**返回值:**

如果图形标识符不存在，返回空值

**实例:**

```
Shape1:=MyCAD1.GetShapeById(0);
```

#### 4.3.43 GetShapeByName

```
function GetShapeByName(const Aname:String):TmyShape;
```

**描述:**

通过图形名称得到这个图形

**返回值:**

如果图形名称不存在，返回空值

**实例:**

```
Shape1:=GetShapeByName('Shape100');
```

#### 4.3.44 GetShapeByNo

**function** GetShapeByNo(AshapeNo:integer):TmyShape;

**描述:**

通过图形序号得到这个图形

**返回值:**

如果图形序号不存在, 返回空值

**实例:**

Shape1:=MyCAD1.GetShapeByNo(10);

#### 4.3.45 GetShapeNoById

**function** GetShapeNoById(AShapeId:Cardinal):Integer;

**描述:**

通过图形标识符得到ShapeNo

**返回值:**

如果图形标识符不存在, 返回-1

**实例:**

Ano:=MyCAD1.GetShapeNoById(0);

#### 4.3.46 GetShapesCount

**function** GetShapesCount:integer;

**描述:**

得到图形总数

**实例:**

ShowMessage('There are ' + IntToStr(MyCAD1.GetShapesCount)+ ' in your form!')

#### 4.3.47 GetShapesCountInALayer

**function** GetShapesCountInALayer(ALayerId:integer):integer;

**描述:**

得到某个图层上的图形总数

#### 4.3.48 GroupWorkingShape

**function** GroupWorkingShape;

**描述:**

组合被选中的一个或多个图形

**返回值:**

-1: 组合图形失败

else: 组合图形的ShapeID

**实例:**

MyCAD1.GroupWorkingShape;

#### 4.3.49 InVisibleLayerById

**procedure** InVisibleLayerById(LayerId:integer);

**描述:**

使图层不可见

#### 4.3.50 InVisibleLayerByName

**procedure** InVisibleLayerByName(LayerName:string);

**描述:**

使图层不可见

#### 4.3.51 IsLinked

**function** IsLinked(Ashape,Bshape:TmyShape):Integer;

**描述:**

确定两个图形之间的链接关系, 如果两个图形之间存在多种关系, 函数只返回第一条连接关系

**返回值:**

返回-1表示两图形之间没有链接, 返回其它 (>=0) 表示链接线图形的ShapeID

#### 4.3.52 IsTCADFile

**function** IsTCADFile(FileName:String):Boolean;

**描述:**

确定一个文件是不是TCADFile

**实例:**

```
if MyCAD1.IsTCADFile( OpenDialog1.FileName) then
begin
  if not MyCAD1.LoadFromFile(OpenDialog1.FileName) then
    ShowMessage('Error when read file');
end
else
  ShowMessage(OpenDialog1.FileName +'is not a TCAD format file');
end;
```

#### 4.3.53 IsVisibleLayerByID

```
function IsVisibleByID(LayerId:integer):Boolean;
```

**描述:**

测试图层是否可见

**返回值:**

返回true表示图层可见，返回false表示图层不可见或不存在

#### 4.3.54 LoadFromFile

```
function LoadFromFile(FileName:String):Boolean;
```

**描述:**

打开一个TCAD绘图文件

**实例:**

```
MyCAD1.LoadFromFile('c:\SaveFile.tcad');
```

#### 4.3.55 LoadFromStream

```
function LoadFromStream(Stream:Tstream):Boolean;
```

**描述:**

从一个流中加载，你也可以从数据库字段或TCP/IP流中读取

**示例:**

```
MyCAD1.LoadFromStream(myStream);
```

### 4.3.56 LockUnlockforShapes

```
procedure LockUnlockforShapes(AShape:TmyShape,Value:boolean);
```

**描述:**

防止图形或者组合图形通过鼠标移动或改变大小,但是可以通过代码移动,当执行这个过程,如果图形被锁定,选择这个图形时,热点是灰色

**参数:**

AShape: 图形

Value: 设定为true时, 锁定图形, 当设为false时图形解锁

### 4.3.57 MergeLayers

```
function MergeLayers(Layer1,Layer2:integer):Boolean;
```

**描述:**

把Layer2并入到Layer1, 并删除原来的Layer2, 当前图层是Layer1

**返回值:**

true: 合并成功

false: 合并失败

**实例:**

```
if MyCAD1.NewLayer(1,2) then  
    ShowMessage('Merge ok!');
```

### 4.3.58 Move

```
procedure Move(AShape:TmyShape;Dx,Dy:integer);
```

**描述:**

通过代码移动一个图形或组合图形,同时触发OnShapeDragging事件

Dx Dy 是一个相对值,当你设置负数时,例如 -10, -20, 图像就从当前位置向左移动10个像素和向上移动20像素。当你设置正数时,例如 10, 20, 图像就从当前位置向右移动10个像素和向下移动20像素。

**实例:**

```
AShape:= MyCAD1.GetSelectedShape;  
if AShape <> nil then  
    MyCAD1.Move(AShape,20,40);
```

### 4.3.59 NewLayer

```
function NewLayer(ALayerName:string;aVisible:Boolean=true):integer;
```

**描述:**

在TCAD中增加一个新的图层，返回图层的标识符，同时当前图层是新创建的图层

**返回值:**

-1: 创建失败

其它: 返回新创建图层的标识符

**实例:**

```
if MyCAD1.NewLayer (' mapLayer')> -1 then  
    ShowMessage('Add ok');
```

### 4.3.60 Paste

```
procedure Paste();
```

**描述:**

粘贴图形，它们有4个像素偏移

**实例:**

```
MyCAD1.SelectShapeBycode(0);  
MyCAD1.Cut;  
MyCAD1.Paste;
```

### 4.3.61 PopfromUndoRedoShapeList

```
procedure PopformUndoRedoShapeList;
```

**描述:**

撤消一步

**实例:**

```
MyCAD1.PopfromUndoRedoShapeList;
```

### 4.3.62 Print

```
procedure Print(ALayers:array of Integer;UserScale:double=1.0);
```

**描述:**

打印当前的绘图

**参数:**

**ALayer:** 假如你如用所有图层, 请用[], 否则使用[1,2,3], 或者你可以创建一个数组, 设定你要打印的图层

**UserScale:** 自定义比例

**实例:****Delphi 语法**

```
//print all shapes in 50% scale.
```

```
MyCAD1.Print([],0.5)
```

**C++ 语法**

```
//print all shapes in 100% scale.
```

```
Void __fastcall TmainFrm::BitBtnClick(Tobject * Sender)
```

```
{  
    int p[1];  
    p[0]=NULL;  
    MyCAD1->Print(p,0,1.0);  
}
```

```
//print the shapes in 0,1 layers in 100% scale.
```

```
Void __fastcall TmainFrm::BitBtnClick(Tobject * Sender)
```

```
{  
    int p[2];  
    p[0]=0;  
    p[1]=1;  
    MyCAD1->Print(p,2,1.0);  
}
```

### 4.3.63 PrintPreview

```
procedure PrintPreview(ALayers:array OF  
Integer;Abitmap:Tbitmap;ScaleforPreview:double=1.0);
```

**描述:**

以图片方式预览

**参数:**

**ALayer:** 假如你如用所有图层, 请用[], 否则使用[1,2,3], 或者你可以创建一个数组, 设定你要打印的图层

**Abitmap:** 画布

**UserScale:** 显示比例



实例:

**Delphi 语法**

```
//print preview all shapes in 100% scale.  
MyCAD1.PrintPreview ([],Image1.Picture.Bitmap,1.0);
```

**C++ 语法**

```
//print preview all shapes in 100% scale.  
Void __fastcall TmainFrm:BitBtn1Click(Tobject *Sender)  
{  
    int p[1];  
    p[0]=NULL;  
    MyCAD1->PrintPreview(p,0,Image1->Picture->Bitmap,1.0);  
}  
//print preview the shapes in 0,1 layers in 100% scale.  
Void __fastcall TmainFrm:BitBtn1Click(Tobject * Sender)  
{  
    int p[2];  
    p[0]=0;  
    p[1]=1;  
    MyCAD1->Print(p,2,Image1->Picture->Bitmap,1.0);  
}
```

### 4.3.64 RenameShapename

```
procedure RenameShapeName(const OldName,NewName:String);
```

**描述:**

图形改名

**实例:**

```
MyCAD1.RenameShapeName('Shape100','MyNewShape');
```

### 4.3.65 Rotate

```
procedure Rotate(AShape:TmyShape;AAngle:Single);
```

**描述:**

通过代码旋转一个图形或一个组合图形，**AAngle**单位是弧度，这个方法对于自动处理方面非常有用

**实例:**

```
这个例子能使图形旋转30度  
AShape:= MyCAD1.GetSelectedShape;
```

```
if AShape <> nil then  
    MyCAD1.Rotate (AShape, 30*pi/180);
```

#### 4.3.66 SaveToBmp

```
procedure SaveToBmp(BmpFileName:String;saveBackColor:Boolean=false);
```

**描述:**  
以BMP格式保存TCAD文件

**实例:**  
MyCAD1.SavetoWMF('c:\SavedFile.bmp');

#### 4.3.67 SaveToBmp-2

```
procedure  
SaveToBmp(Bmp:Tbitmap;NewWidth,NewHeight:integer;saveBackColor:Boolean=false):o  
verload;
```

**描述:**  
以BMP格式保存TCAD文件，可以自己设定大小

**实例:**  
MyCAD1.SavetoWMF('c:\SavedFile.bmp',100,200);

#### 4.3.68 SaveToDxf

```
procedure SaveToDXF(DXFFileName:String);
```

**描述:**  
以AutoCAD R12 DXF的文件格式保存

**实例:**  
CAD1.SavetoDXF('c:\SavedFile.dxf');

#### 4.3.69 SaveToFile

```
function SaveToFile(FileName:String):Boolean;
```

**描述:**  
以TCAD文件格式保存

**实例:**

```
CAD1.SavetoFile('c:\SavedFile');
```

#### 4.3.70 SaveToJpg

**procedure**

```
SaveToJpeg(JpegFileName:String;Quality:integer=80;saveBackColor:Boolean=false)
```

**描述:**

以JPEG的格式保存CAD画图，设置参数Quality可以改变图片像素，默认是80

**实例:**

```
CAD1.SavetoJPG('c:\SavedFile.jpg');
```

#### 4.3.71 SaveToStream

```
function SaveToStream(Stream:Tstream):Boolean;
```

**描述:**

保存到流，你可以保存所有的内容到数据库字段或内存流

**实例:**

```
MyCAD1.SaveToStream(myStream);
```

#### 4.3.72 SaveToWmf

```
procedure SaveToWmf(WMFFileName:String);
```

**描述:**

以WMF的格式保存TCAD画图

**实例:**

```
CAD1.SavetoWMF('c:\SavedFile.WMF');
```

#### 4.3.73 SelectAllShapes

```
procedure SelectAllShapes;
```

**描述:**

选择所有的图形

**示例:**

```
MyCAD1.SelectAllShapes;
```

#### 4.3.74 SelectShapeByCode

**function**

SelectShapeByCode(ShapeId:integer;RemovePrevSelectedShape:boolean=true):Boolean

**描述:**

像鼠标操作一样通过代码选择图形，通过图形标识符来操作特定的图形，假如 RemovePreSelectedShape 的值为 true，则不会清除当前图形的选择状态

**返回值:**

true: 表示图形被选择  
false: 没有此标识符的图形

**实例:**

```
MyCAD1.SelectShapeByCode(2);
```

**注意:**

- 如是要图形是组合图形，只显示组合图形的热点

#### 4.3.75 SendtoBack

**procedure** SendToBack(AShape:TmyShape;NeedSave:Boolean=true);

**描述:**

移到选择的图形到最后

**参数:**

AShape: 想操作的图形  
NeedSave: 是否放入Undo队列，缺省为true

**实例:**

```
MyCAD1.SendToBack(MyCAD1.GetSelectedShape);
```

#### 4.3.76 SendToBackByStep

**procedure** SendToBackByStep(AShape:TmyShape;NeedSave:Boolean=true);

**描述:**

选择的图形向后移动一层

**参数:**

AShape: 想要操作的图形  
NeedSave: 是否放入Undo队列，缺省为true

实例:

```
MyCAD1.SendToBackByStep(MyCAD1.GetSelectedShape);
```

#### 4.3.77 SetLayerNameById

```
procedure SetLayerNameById(const NewLayerName:string;ALayerID:integer);
```

描述:

能过LayerId更改图层名称

#### 4.3.78 SetLayerNameByName

```
procedure SetLayerNameByName(const NewLayerName,OldLayerName:string);
```

描述:

能过图层原来的名称设置图层新名称

#### 4.3.79 SetMyImage

```
funcation SetMyImage(ShapeID:integer;Abitmap:Tbitmap):Boolean;
```

描述:

在TmyImage中添加一个图形

实例:

```
if AShape.ClassName = 'TMyImage' then
begin
  if OpenPictureDialog1.Execute then
  begin
    myBitmap := TBitmap.Create;
    mybitmap.LoadFromFile(OpenPictureDialog1.FileName);
    if MyCAD1.SetMyImage(ShapeId, MyBitmap) then
      Memo1.Lines.Add('Bitmap be loaded into Shape' + Inttostr(ShapeID))
    else
      Memo1.Lines.Add('Bitmap NOT be loaded into Shape' +
        Inttostr(ShapeID));
    MyBitmap.Free;
  end;
end;
```

返回值:

true: 添加图片成功

false: 添加图片失败

#### 4.3.80 SizeShape

**procedure**

SizeShape(AShape:TmyShape;ASelectedHotId:integer;AMovPt:TMyPoint);

**描述:**

改变一个图形或组合图形的大小，如同通过鼠标改变一个图形的大小

**参数:**

AShape: 要想改变大小的图形

AselectedHotId: 选择热点的标识符

AMovPt: 目标位置

**实例:**

MyCAD1.SizeShape(MyCAD1.GetSelectedShape,0,MyPoint(100,100));

#### 4.3.81 UngroupShape

**procedure** UngroupShape(AShape:TmyShape;NeedSaved:boolean=true);

**描述:**

取消一个组合图形

**参数:**

AShape: 想要取消组合的图形

NeedSaved: 是否放入Undo队列，缺省为true

**实例:**

MyCAD1.UngroupShape(MyCAD1.GetSelectedShape);

#### 4.3.82 VisibleAllLayer

**procedure** VisibleAllLayer;

**描述:**

使所有图层可见

#### 4.3.83 VisibleLayerByID

**procedure** VisibleLayerByID(LayerID:integer);

**描述:**

使指定标识符的图层可见

#### 4.3.84 VisibleLayerByName

**procedure** VisibleLayerByName(LayerName:string);

**描述:**

使某一图层可见

#### 4.3.84 SelectAllShapesByLayerId

*xp.b*

**procedure** SelectAllShapesByLayerId(const ALayerId:integer);

**描述:**

选定指定层的所有图形

**示例代码:**

```
MyCAD1.SelectAllShapesByLayerId(0);
```

#### 4.3.85 Tilt

*xp.e*

**function**

Tilt(operateShape:TMyShape:centerPoint,fromPoint,toPoint:TMyPoint;Constrain: Boolean):single;

**描述:**

指定中心点、起始点、目的点来通过代码旋转图形

**参数:**

*operateShape*: 目标图形

*centerPoint*: 中心点

*fromPoint*: 起始点

*toPoint*: 目标点

*Constrain*: 旋转度标志

**示例代码:**

```
var
    oShape: TMyShape;
begin
    oShape := MyCAD1.GetSelectedShape;
    MyCAD1.Tilt(oShape, oShape.GetPoint(0), oShape.GetPoint(1), oShape.GetPoint(
        2), false);
end;
```

### 4.3.86 GetShapesByLayerId

**function** GetShapesByLayerId(layerId: Byte): TArrMyShape;

**描述:**

得到一个图层中的所有图形

**参数:**

layerId: 图层的ID

**返回值:**

图形对象数据

### 4.3.87 PasteFromMyCAD



**procedure** PasteFromMyCAD(AMyCAD: TMyCAD);

**描述:**

从其它的MyCAD拷贝图形。.

**示例:**

```
MyCAD1.PasteFromMyCAD(MyCAD2);
```

#### 4.3.88 AverageHeight

**procedure** AverageHeight(shapeArray: TArrMyShape);

**描述:**

平均分布高度。

**示例:**

```
MyCAD1.AverageHeight(MyCAD1.WorkingShapes);
```

#### 4.3.89 AverageWidth

**procedure** AverageWidth(shapeArray: TArrMyShape);

**描述:**

平均分布宽度。

**示例:**

```
MyCAD1.AverageHeight(MyCAD1.WorkingShapes);
```

#### 4.3.90 PreviewSelectedShapes

**procedure** PreviewSelectedShapes(ABitmap:TBitmap; ScaleforPreview:double = 1.0);

**描述:**

预览选中的图形。

**参数:**

ABitmap: 预览的画布

ScaleforPreview: 预览的缩入比例

#### 4.3.91 PrintSelectedShapes

**procedure** PrintSelectedShapes(UserScale: double = 1.0);

**描述:**

打印选中的图形。

**参数:**

UserScale: 打印的缩入比例

## 4.4 属性

### 4.4.1 ArrowAngle

**property** ArrowAngle:integer

**描述:**

设置线和双箭头线的箭头角度，其值在0到359之间

**示例:**

MyCAD1.ArrowAngle:=180;

### 4.4.2 ArrowLength

**property** ArrowLength:Byte;

**描述:**

设置线和箭头线的箭头长度，其值在10到50之间

### 4.4.3 ArrowOffset

**property** ArrowOffset:byte

**描述:**

设置箭头的偏移量，其值在0到255之间，默认值是0

**示例:**

MyCAD1.ArrowOffset:=0;



MyCAD1.ArrowOffset:=16;



### 4.4.4 ArrowStyle

**property** ArrowStyle:TarrowStyle

**描述:**

设置线和双箭头线的箭头类型

Anone: 线

ALeft: 箭头在左端

ARight: 箭头在右端

ADouble: 双箭头线

#### 4.4.5 BkBitmap

**property** BkBitmap:Tbitmap

**描述:**

设置TCAD的背景图片，如果要清除掉，设置BkBitmap:=nil;

**示例:**

例子演示了加载一个指定图片到MyCAD

**var**

mybitmap:TBitmap;

**begin**

if OpenPictureDialog1.Execute **then**

**begin**

myBitmap:=TBitmap.Create;

mybitmap.LoadFromFile(OpenPictureDialog1.FileName);

MyCAD1.BkBitmap:=mybitmap;

MyBitmap.Free;

**end;**

**end;**

#### 4.4.6 BkBitmapMode

**property** BkBitmapMode:TbkBitmapMode

**TBkBitmapMode=** (Tiled,Stretch,Center,LeftTop) ;

**描述:**

设置背景图片显示样式

Tiled: 平铺

Stretch: 拉伸

Center: 居中

LeftTop: 左上角

#### 4.4.7 Brush

**property** Brush:TBrush;

**描述:**

设置TMyCAD的画刷

#### 4.4.8 Canvas

**property** Canvas:TCanvas;

**描述:**

画布允许通过一些提供的画布对象在上面画图，一些操作在TMyCAD上是不允许的，画图操作不会起效，当重画事件发生时TmyShapes的内容将会清除掉。画布对象自动建立，而且属性只读

#### 4.4.9 ColorOfBackground

**property** ColorOfBackground:Tcolor

**描述:**

指定背景颜色

**示例:**

MyCAD1.ColorOfBackground:=clBule;

#### 4.4.10 ColorOfHot

**property** ColorOfHot:TColor

**描述:**

指定方块热点的颜色

**示例:**

MyCAD1.ColorOfHot:=clRed;

#### 4.4.11 CrossLine

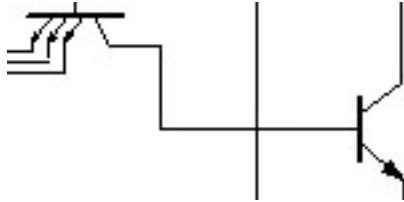
**property** CrossLine:Boolean

**描述:**

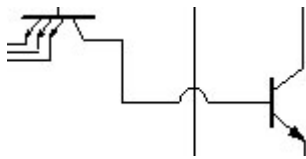
用于连接线功能，当设置为True时显示交叉线特征

示例:

```
MyCAD1.CrossLine:=false;
```



```
MyCAD1.CrossLine:=true;
```



#### 4.4.12 CurrentLayerId

```
property CurrentLayerId:integer;
```

描述:

得到当前图层标识符，从0开始的。将会在这个图层上新增图形

示例:

```
MyCAD1.CurrentLayerID:=2;  
ShowMessage (IntToStr('You are drawing on layerID:'+MyCAD1.CurrentLayerID));  
MyCAD1.CurrentLayerID:=3;  
ShowMessage (IntToStr('You are drawing on layerID:'+MyCAD1.CurrentLayerID));  
// if there is no layerid is 3,MyCAD1.CurrentLayerID still equal 2
```

#### 4.4.13 DiskFileVersion

只读属性，用于查看TMyCAD版本的兼容性

#### 4.4.14 DragMode

```
property DragMode:TdragMode
```

描述:

拖动图形的方式，当设置值为dgHorz时图形只能以水平方向拖动，当设置值为dgVert时图形只能以垂直方向拖动

示例:

```
MyCAD1.DragMode:=dgHorz;
```

#### 4.4.15 Enable

**property** Enabled:Boolean;

**描述:**

TCAD是否响应鼠标、键盘和时间事件，但对代码操作仍然有效

使用Enabled属性改变用户对TMyCAD的可用性，使控制失效，设置Enabled属性为false，不可用的TMyCAD显示颜色较暗，假如Enable属性为false，TMyCAD忽视鼠标、键盘和时间事件

当设置Enabled属性为true时，TMyCAD显示正常，用户可以使用TMyCAD

#### 4.4.16 Font

**property** Font:TFont;

**描述:**

设置字体。

#### 4.4.17 GridOperation

**Property** GridOperation:TGridOperation;



**描述:**

设置TMyCAD的网格属性

#### 4.4.18 HotShow

**property** HotShow:boolean;

**描述:**

true: 热点显示

false: 热点隐藏，即使图形被选中也隐藏

示例:

**Delphi 语法:**

```
MyCAD1.HotShow:=true;
```

**C++ 语法:**

```
MyCAD1->HotShow=true;
```

#### 4.4.19 HotSize

**property** HotSize:byte

**描述:**

设置热点大小

示例:

**Delphi 语法:**

```
MyCAD1.HotSize:=6;
```

**C++ 语法:**

```
MyCAD1->HotSize=6;
```

#### 4.4.20 LabelValue

**property** LabelValue:TLabel

**描述:**

显示当前图形的参数，可以显示线类图形的长度，矩形类图形的宽和高

示例:

```
MyCAD1.LabelValue:=Form1.Label2;
```

#### 4.4.21 LabelXY

**property** LabelXY:TLabel

**描述:**

显示当前鼠标的坐标，随着鼠标的移动而改变

示例:

**Delphi 语法:**

```
MyCAD.LabelXy:=form1.label1;
```

**C++ 语法:**

```
MyCAD->LabelXy=Form->Label1;
```

#### 4.4.22 LinkLineDrawStyle

**property** LinkLineDrawStyle:TLinkLineDrawStyle

**描述:**

定义连接线类型，横平竖直或直连线

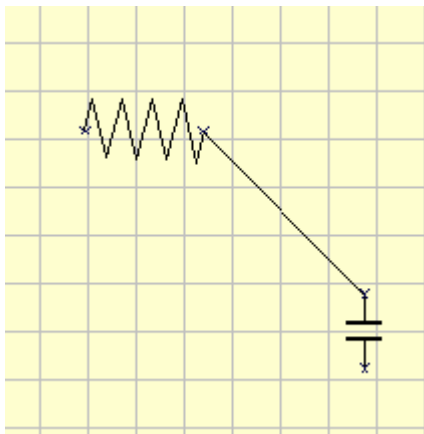
**实例:**

**Delphi 语法:**

```
MyCAD1.LinkLineDrawStyle:=lldsFree;
```

**C++ 语法:**

```
MyCAD1->LinkLineDrawStyle=lldsFree;
```



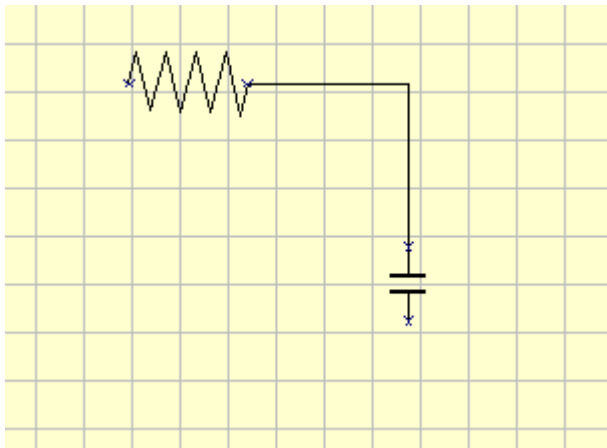
**Delphi 语法:**

```
MyCAD1.LinkLineDrawStyle:=lldsHV;
```

**C++ 语法:**

```
MyCAD1->LinkLineDrawStyle=lldsHV;
```





#### 4.4.23 OperateAllLayer

**property** OperateAllLayer:boolean;

**描述:**

你是否想让鼠标的行为作用于当前层或所有层，默认值为true

**示例:**

```
MyCAD1.OperateAllLayer:=true;
```

#### 4.4.24 PageFoot

**property** PageFoot:string;

**描述:**

设置打印时页脚的文字

**示例:**

**Delphi 语法:**

```
MyCAD1.PageFoot:='Designed by hongbin.fei,2004.12';
```

**C++ 语法:**

```
MyCAD1->PageFoot=" Designed by hongbin.fei,2004.12";
```

#### 4.4.25 PageFootAlignment

**property** PageFootAlignment:Talignment;

**描述:**

设置页脚文本排列

使用Alignment来改变TMyCAD上的文本格式，Alignment可以选择下列常量值:

taLeftJustify: 左边显示

taCenter: 居中显示

taRightJustify: 右边显示

#### 4.4.26 PageFootFont

**property** PageFootFont:string;

**描述:**

设置页脚文本字体

#### 4.4.27 PageFootToBottom

**property** PageFootToBottom:integer;

**描述:**

页脚边距，单位像素

#### 4.4.28 PageHead

**property** PageHead:string;

**描述:**

设置页标题

**示例:**

**Delphi 语法:**

MyCAD1.PageHead:='Designed yb hongbin.fei,2004.12';

**C++ 语法:**

MyCAD1->PageHead=" Designed yb hongbin.fei,2004.12";

#### 4.4.29 PageHeadAlignment

**property** PageHeadAlignment:Talignment;

**描述:**

设置页眉文本排列

#### 4.4.30 PageHeadFont

**property** PageHeadFont:TCADFont;

**描述:**

设置页眉文本字体

#### 4.4.31 PageHeadToTop

**property** PageHeadToTop:integer;

**描述:**

页眉边距，单位是像素

#### 4.4.32 PageHeight

**property** PageHeight:Cardinal

**描述:**

设置面布的高度，单位是厘米，当你设置PageHeight时，PageStyle会自动改变

**实例:**

MyCAD1.PageHeight:=210

#### 4.4.33 PageOrientation

**property** PageOrientation:TprinterOrientation;

**描述:**

可定义画布的方向为横向或纵向，同时 PageWidth 和 PageHeight 会自动改变，在PageStyle设置为自定义页面时，改变画布的方向不会自动更改PageWidth 和 PageHeight属性

**示例:**

MyCAD1.PageOrientation: =poLandscape;

#### 4.4.34 PageStyle

**property** PageStyle:TpageStyle;

**描述:**

## TCAD for Delphi & C++ Builder & Kylix & Vcl.net

---

设置页面类型，包括A0、A1、A2、A3、A4、A5、B3、B4、B5、CustomerPage，在设置页面类型时PageWidth和PageHeight会自动改变

实例：

```
MyCAD1.PageStyle:=A4;
```

### 4.4.35 PageWidth

**property** PageWidth:Cardinal

描述：

设置画布（页面）的宽度，单位是厘米，如果你改变页面类型，PageWidth会自动改变

### 4.4.36 Pen

**property** Pen:TPen;

描述：

设置所需的画笔

### 4.4.37 PrintABorder

**property** PrintABorder:Boolean;

描述：

是否显示预览/打印边框

### 4.4.38 PrintABordertoBottom

**property** PrintABordertoBottom:integer;

描述：

设置边框的下边距，单位是像素

### 4.4.39 PrintABordertoLeft

**property** PrintABordertoLeft:integer;

描述：

设置边框的左边距，单位是像素

#### 4.4.40 PrintABordertoRight

**property** PrintABordertoRight:integer;

**描述:**

设置边框的右边距，单位是像素

#### 4.4.41 PrintABordertoTop

**property** PrintABordertoTop:integer;

**描述:**

设置边框上边距，单位是像素

#### 4.4.42 PrintBackground

**property** PrintBackground:Boolean;

**描述:**

打印画布背景，当设置true时打印同显示一样

#### 4.4.43 Ratio

**property** Ratio:double;

**描述:**

定义比率，该属性非常有用，它出现在LabelValue属性中，包括线的长度和矩形、圆形等其它图形的面积

#### 4.4.44 ResizeEnable

**property** ResizeEnable:boolean;

**描述:**

是否可以改变图形大小，在不允许改变图形大小的情况下它非常有用

#### 4.4.45 ReturnToSelecting

**property** ReturnToSelecting:Boolean

**描述:**

## TCAD for Delphi & C++ Builder & Kylix & Vcl.net

---

当设置为true时，在画完一个图形时，ShapeTool自动返回到选择状态，同时触发OnActionToolToSelecting事件，否则设置为false时，ShapeTool仍然是画图状态，你可以连续地画图

### 实例：

```
MyCAD1.ReturnToSelecting:=false;  
你可以连续的画一些图形
```

### 4.4.46 RotateConstraintDegree

```
property RotateConstraintDegree:Integer;
```

#### 描述：

设置图形旋转时的约束角度，当值为0时，可以以任意角度旋转。

### 4.4.47 RotateEnable

```
property RotateEnable:boolean;
```

#### 描述：

是否能旋转图形，在不允许旋转图形的情况下它非常有用

### 4.4.48 ShapeTool

```
property ShapeTool:TDrawTool
```

#### 描述：

设置当前的画图工具

#### 实例：

```
//Set Line draw tool  
MyCAD1.ShapeTool:=SpLine;  
//Set Link Line draw tool  
MyCAD1.ShapeTool:=SpLinkLine;
```

注意：当ShapeTool设置为SpClose时将关闭TCAD的画图、拖动、改变大小和旋转功能

#### 4.4.49 ShowHint

**property** ShowHint:Boolean;

**描述:**

当鼠标指向一个图形时是否显示提示信息，显示Caption

**示例:**

**Delphi 语法:**

```
MyCAD1.ShowHint:=true;
```

**C++ 语法:**

```
MyCAD1->ShowHint=true;
```

#### 4.4.50 ShowHotLink

**property** ShowHotLink:Boolean;

**描述:**

是否在TMyCAD中图形中显示连接点

**示例:**

**Delphi 语法:**

```
MyCAD1.ShowHotLink:=true;
```

**C++ 语法:**

```
MyCAD1->ShowHotLink=true;
```

#### 4.4.51 Snap

**property** Snap:Boolean;

**描述:**

是否对齐鼠标到网格和其它图形，帮助你画图

**示例:**

```
MyCAD1.Snap:=true;
```

#### 4.4.52 SnapPixels

**property** SnapPixels:Byte

**描述:**

鼠标和最近网格之间的像素小于这个值时，鼠标会捕获这个网格，这个值不能大于

#### 4.4.53 SnapShape

**property SnapShape: Boolean;**

**描述:**

当拖动和改变图形大小时，是否对齐到其它图形，有助于画图

**实例:**

```
MyCAD1.SnapShape:=true;
```

#### 4.4.54 TheUnit

**property TheUnit: Tunits**

**描述:**

设置单位类型，它会出现在LabelValue属性中线的长度或矩形等图形的面积

#### 4.4.55 UndoRedoSize

**property UndoRedoSize: byte**

**描述:**

设置撤消动作保存的步骤，它将占用系统资源

**示例:**

```
MyCAD1.Zoom:=0.50;
```

#### 4.4.56 Version

**property Version: string;**

**描述:**

显示TMyCAD的版本，只读属性

**示例:**

```
ShowMessage('Installed TMyCAD version is:'+MyCAD1.Version);
```

#### 4.4.57 Visible

**property Visible: Boolean;**



**描述:**

是否显示TMyCAD

在运行时使用Visible属性来控制可见性，设置true时可出，否则不可见  
它会调用Show方法来设置可见，调用Hide方法来隐藏

#### 4.4.58 XMMode

**property** XYMode

**描述:**

有4种选择模式，适合你的需求  
请查看定义里的TxyMode

#### 4.4.59 Zoom

**property** Zoom:Double

**描述:**

设置缩放比例，宽度、高度和背景都会自动缩放，当设置为1时，TMyCAD以实际大小显示

**实例:**

```
MyCAD1.Zoom:=0.50;
```

#### 4.4.60 PreZoom

**property** PreZoom:Double

**描述:**

记录上一次的Zoom值。

#### 4.4.61 MouseEffect

**property** MouseEffect:Boolean

**描述:**

指定是否可以自定义光标

**实例:**

```
MyCAD1.MouseEffect := true;
```

#### 4.4.62 LockBound

**property** LockBound:Boolean

**描述:**

指定当拖动一个图形时是否锁定在画布内

**实例:**

```
MyCAD1.LockBound := true;
```

#### 4.4.63 UserData

**property** UserData:TuserData;

**描述:**

你可以增加你自己的数据，它非常有用

**示例:**

```
UserData.AddKeyandValue('Weight','20kg');
```

#### 4.4.64 DragTrace

**property** DragTrace:Boolean;

**描述:**

你可以在拖动图形时是否显示拖动路径

**示例:**

```
MyCAD1.DragTrace:=false;
```

#### 4.4.65 ArcMode

**property** ArcMode:TarcMode;

**描述:**

有两种类型选择, amCircle 或 amEllipse , 允许在画弧线之前选 ArcMode 类型。

amCircle:



amEllipse:



#### 4.4.66 DisableTextInput

**property** DisableTextInput:Boolean;

**描述:**

当这个值设置为True时,用鼠标添加文本对象时会隐藏文本输入框。

**示例:**

```
MyCAD1.DisableTextInput:=true;
```

#### 4.4.67 PenStyleEndCapType

**property** PenStyleEndCapType:TPenStyleEndCapType

**描述:**

设置线条对象结尾点的方式

**取值:**

psectROUND  
psectSQUARE  
psectFLAT  
psectMASK

#### 4.4.67 PenStyleJoinType

**property** PenStyleJoinType:TPenStyleJoinType

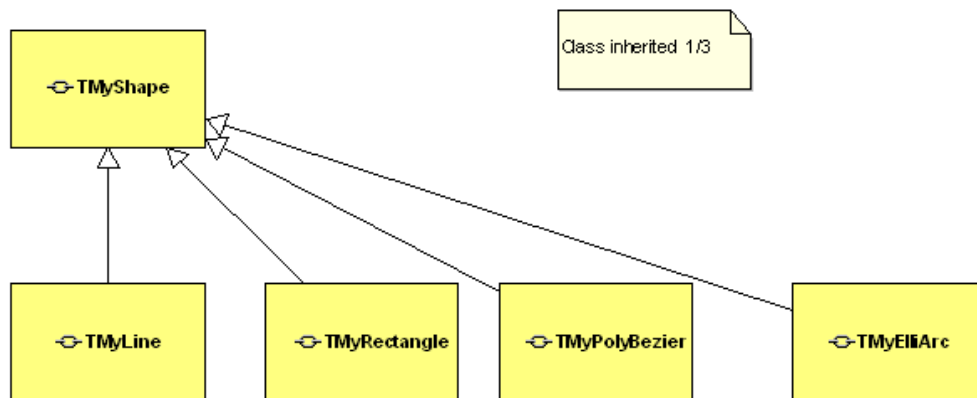
**描述:**

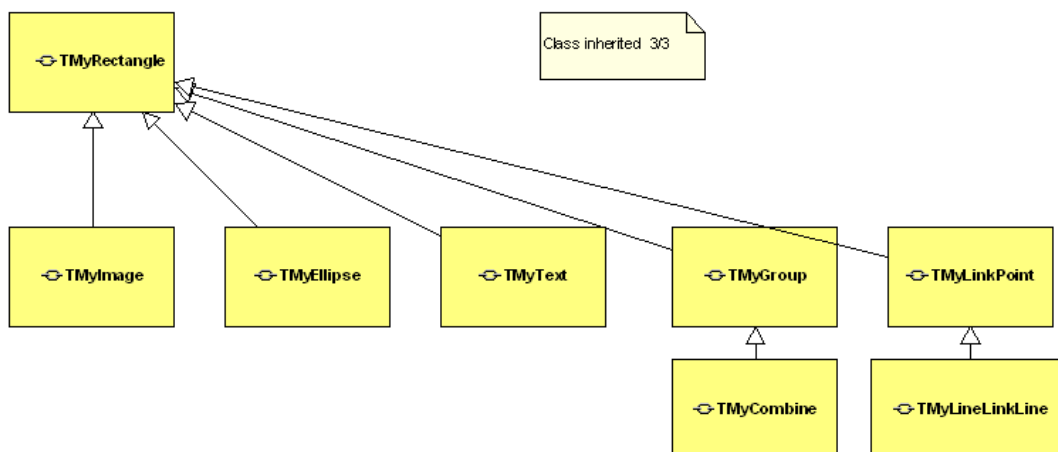
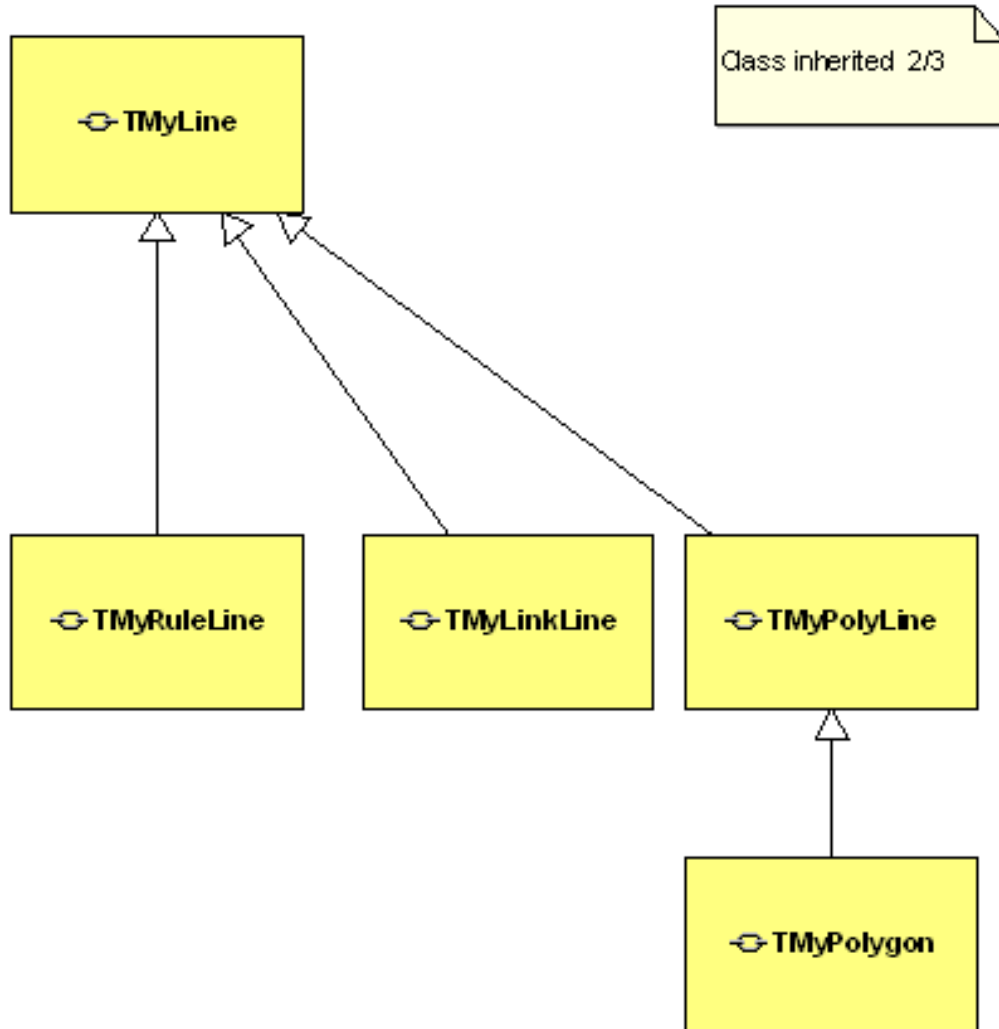
设置矩形对象连接点方式

**取值:**

- psectROUND
- psectSQUARE
- psectFLAT
- psectMASK

## 5 图形类继承图





## 6 TMyShape

### 6.1 类图



### 6.2 字段域

#### 6.2.1 CenterPoint

**CenterPoint:TMyPoint;**

**描述:**

一个图形的中心点，在TMyCAD中会自动维护

## 6.2.2 ChildShapesNo

**ChildShapesNo:integer;**

**描述:**

假如一个图形是一个单独的图形，那么它的值是空值，假如这是一个组合图形或联合图形，这个字段域会存储子图形的序号

## 6.2.3 LayerID

**LayerID:integer**

**描述:**

指出这个图形属于哪个图层，一个图形必须在一个图层上

## 6.2.4 ParentShapesNo

**ParentShapesNo:integer;**

**描述:**

父图形的序号，如果这个图形没有父图形，这个值为-1

## 6.2.5 Shapeld

**Shapeld:integer;**

**描述:**

图形的标识符，删除或增加一个图形，它不能更改，是一个自增字段域，它在 TMyCAD中维护，只读

## 6.2.6 ShapeNo

**ShapesNo:integer;**

**描述:**

这个图形的序号，当删除、新增、移前、移后等操作时这个值会改变，在 TMyCAD中维护，对于你来说它只能读取

## 6.2.7 TextOutPoint

**TextOutPoint:TMyPoint;**

**描述:**

这是写文本所在的位置，当一个图形创建时，在图形的中下方位置，你可以通过鼠标或代码来改变这个位置

## 6.2.8 ThePoints

**ThePoints:TArrMyPoint**

**描述:**

一个图形的位置点数组，不同的图形有不同的位置点个数，它和XyMode没有关系。右上角位置是(0,0)，它是TMyPoint类型

## 6.3 方法

### 6.3.1 Assign

**procedure Assign(Source:TMyShape);virtual;**

**描述:**

拷贝一个图形对象

**实例:**

AShape.Assign(Bshape);

### 6.3.2 ComputerCenterPoint

**function ComputerCenterPoint:TMyPoint;**

**描述:**

它能计算出一个图形的中心点

### 6.3.3 Create

**constructor Create(Aowner:TMyCAD);virtual;**

**描述:**

构造函数



内部的数据结构已经初始化

### 6.3.4 Destroy

**destructor Destroy;override;**

**描述:**

析构函数

首先释放自己所有的字段域，最后调用Destroy方法

### 6.3.5 Draw

**procedure Draw(MyCanvas:TCanvas);virtual;**

**描述:**

在MyCanvas上画一个图形

### 6.3.6 GetCenterPoint

**function GetCenterPoint:TMyPoint;**

**描述:**

返回一个图形的中心点

**返回:**

一个图形的中心点

### 6.3.7 GetCenterPointInZoom

**function GetCenterPointInzoom:TMyPoint;**

**描述:**

返回当前缩放比例的中心点

### 6.3.8 GetHeight

**function GetHeight:Single;**

**描述:**

返回一个图形的外接矩形高度，单位是像素

### 6.3.9 GetLeftBottom

**function GetLeftBottom:TMyPoint;**

**描述:**

返回一个外矩形的左下角的坐标，单位是TMyPoint

### 6.3.10 GetLeftTop

**function GetLeftTop:TMyPoint;**

**描述:**

返回外矩形的左上角的坐标，单位是TMyPoint

### 6.3.11 GetLinkPoint

**function GetLinkPoint(PointID:integer):TMyPoint;**

**描述:**

通过PointId得到连接点

**参数:**

给出连接线的标识符，其值 PointId>=0 and PointId<=GetLinkPointsCount-1

### 6.3.12 GetLinkPointInZoom

**function GetLinkPointZoom(PointID:integer):TMyPoint;**

**描述:**

在当前比例下得么连接点，详细信息请查看GetLinkPoint方法

### 6.3.13 GetMyHeight

**function GetMyHeight:Single;**

**描述:**

得到图形的实际高度，它不考虑角度，单位是像素

### 6.3.14 GetMyWidth

**function GetMyWidth:Single;**

**描述:**

得到图形的实际宽度，不考让角度，单位是像素

### 6.3.15 GetPoint

**function GetPoint(PointID:integer):TMyPoint;public**

**描述:**

通过点标识符得到点

**参数:**

PointID: 给出点标识符，其值必须在 PointID>=0 and PointID<=GetPointsCount-1 范围内

**返回:**

点对角

**实例:**

得到一个图形的第一个点  
if AShape.GetPointsCount>0 then  
  Apoint:=AShape.GetPoint(0);

### 6.3.16 GetPointInZoom

**function GetPointInZoom(PointID:integer):TMyPoint;public**

**描述:**

得到当前缩放比例中的点。详细信息请查看GetPoint

### 6.3.17 GetPointsCount

**function GetPointsCount:Integer;public**

**描述:**

得到一个图形的点数

### 6.3.18 GetRightBottom

**function GetRightBottom:TMyPoint;**

**描述:**

返回一个图形的外矩形的右下角边距，单位是TMyPoint

### 6.3.19 GetRightTop

**function GetRightBottom:TMyPoint;**

**描述:**

返回一个图形的外矩形的右上角的坐标，单位是TMyPoint

### 6.3.20 GetShapeld

**function GetShapeld:integer;**

**描述:**

得到一个图形的标识符

注意：标识符从0开始的

### 6.3.21 GetWidth

**function GetWidth:Single;**

**描述:**

返回一个图形外接矩形的宽度，单位是像素

### 6.3.22 HasChildShapes

**function HasChildShapes:Boolean;**

**描述:**

判断一个图形是否有子图形

**返回值:**

true: 图形有子图形

false: 图形没有子图形

### 6.3.23 HasLinkShapes

**function HasLinkShapes:Boolean;**

**描述:**

判断一个图形是否有连接的图形

**返回值:**

true: 图形有连接的图形

false: 图形没有连接的图形

### 6.3.24 HasParentShape

**function HasParentShape:Boolean;**

**描述:**

判断一个图形是否有父图形

**返回值:**

### 6.3.25 LoadFromStream

**procedure LoadFromStream(Astream:Tstream):virtual;**

**描述:**

从流中加载一个图形，像TmyLine对象等可重载此方法

### 6.3.26 SaveToStream

**procedure SaveToStream(Astream:Tstream):virtual;**

**描述:**

保存一个图形到流，像TmyLine对象等可覆盖此方法

### 6.3.27 IsClickedMe

**function IsClickedMe(MyCanvas:TCanvas;APoint: TPoint): Boolean; virtual;**

**描述:**

判断点是否在图形中

**返回值:**

true : 在图形中. false : 不在图形中

## 6.4 属性

### 6.4.1 Angle

**property Angle:single;**

**描述:**

设置图形的角度。单位是弧度

### 6.4.2 Brush

**property Brush:Tbrush;**

**描述:**

设置你所需TMyShape的画刷

**新增功能:**

在XP。B版本中，闭合的图形支持Bitmap填充

**示例代码:**

```
var
    tmpBitmap:TBitmap;
begin
    tmpBitmap := TBitmap.Create;
    tmpBitmap.LoadFromFile('d:\test\tmp.bmp');
    MyCAD1.GetSelectedShape.Brush.Bitmap := tmpBitmap;
    MyCAD1.Repaint;
end;
```

### 6.4.3 Caption

**property Caption:string;**

**描述:**

当创建图形的标记图形时，Name属性同这个图形的名称，你可以改变它

实例:

```
AShape.Caption:='it is a line';
```

#### 6.4.4 CaptionShow

**property CaptionShow:boolean;**

描述:

设置信息提示是否显示, 提示信息就是Caption属性值

#### 6.4.5 ColorBegin

**property ColorBegin:TColor;**

描述:

设置渐变类型图形的开始颜色

#### 6.4.6 ColorEnd

**property ColorEnd:TColor;**

描述:

设置渐变类型图形的结束颜色

#### 6.4.7 Font

**property Font:TFont;**

描述:

TFont用于设置文本显示风格, TFont定义了一个指定高度、类型、属性等的集合, TFont已成为Windows惯用的字体

#### 6.4.8 GradientStyle

**property GradientStyle:TGradientStyle;**

描述:

设置渐变色类型

示例:

```
AShape.GradientStyle:=gsRadialC;
```

### 6.4.9 Info

**property Info:string;**

描述:

它存储字符串, 你可以根据自己需要设置

### 6.4.10 IsFlipHorz

**property IsFlipHorz:Boolean;** 

描述:

改变图形的翻转状态, 当设置成true, 图形水平翻转

### 6.4.11 IsFlipVert

**property IsFlipVert:Boolean;** 

描述:

改变图形的翻转状态, 当设置成true, 图形垂直翻转

### 6.4.12 Locked

**property Lock:boolean;**

描述:

锁定图形, 热点区域将以灰色显示

### 6.4.13 Name

**property Name:TComponentName;**

代码引用中指定一个组件

描述:

在一个应用程序中使用使用Name属性来确定一个图形, 通常, 一个新图形被定义成Shape1、Shape2等等。



#### 6.4.14 Owner

**property Owner:TMyCAD;**

**描述:**

指定图形属于哪个TMyCAD

#### 6.4.15 Pen

**property Pen:TPen**

**描述:**

设定图形所需的画笔格式

#### 6.4.16 Tag

**property Tag:Longint;**

**描述:**

Tag属性是为了方便开发人员提供的,它可以用来存储一个附加的整型数据或者能配合像组件引用或指针这样的32位值

#### 6.4.17 UserData

**property UserData:TuserData;**

**描述:**

你可以增加你自己的数据,它非常有用

**示例:**

```
UserData.AddKeyandValue('Weight','20kg');
```

#### 6.4.18 Visible

**property Visible:boolean;**

**描述:**

当设置为false时,图形不能被选中、改变大小、旋转、组合

### 6.4.19 MultiInfo

property MultiInfo:TStrings; 

**描述:**

显示图形的多行信息

**注意:** 如果此属性值为空, 则Info属性仍有效。

### 6.4.20 MultiInfoAlignment

property MultiInfoAlignment:TAlignment; 

**描述:**

多行信息的水平对齐方式, 以最长的字符串为标准, 进行左对齐、居中对齐、右对齐

### 6.4.21 LogHeight

property LogHeight: Integer

**描述:**

设置字体高度, LogUse设置True时有效。

### 6.4.22 LogUse

property LogUse: Boolean

**描述:**

设置值为True时, LogHeight 和 LogWidth有效。

### 6.4.23 LogWidth

property LogWidth: Integer

**描述:**

设置字体宽度, LogUse设置True时有效。

#### 6.4.24 ResizeEnable

**property** ResizeEnable:boolean;

**描述:**

是否可以改变图形大小, 在不允许改变图形大小的情况下它非常有用

#### 6.4.24 HotShow

**property** HotShow:boolean;

**描述:**

**true:** 热点显示

**false:** 热点隐藏, 即使图形被选中也隐藏

**示例:**

**Delphi 语法:**

shape.HotShow:=true;

**C++ 语法:**

shape->HotShow=true;

#### 6.4.24 RotateEnable

**property** RotateEnable:boolean;

**描述:**

是否能旋转图形, 在不允许旋转图形的情况下它非常有用

#### 6.4.25 PenStyleEndCapType

**property** PenStyleEndCapType:TPenStyleEndCapType

**描述:**

设置线条对象结尾点的方式

**取值:**

psectROUND

psectSQUARE

psectFLAT  
psectMASK

### 6.4.26 PenStyleJoinType

**property** PenStyleJoinType:TPenStyleJoinType

**描述:**

设置矩形对象连接点方式

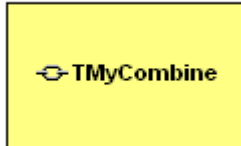
**取值:**

psectROUND  
psectSQUARE  
psectFLAT  
psectMASK

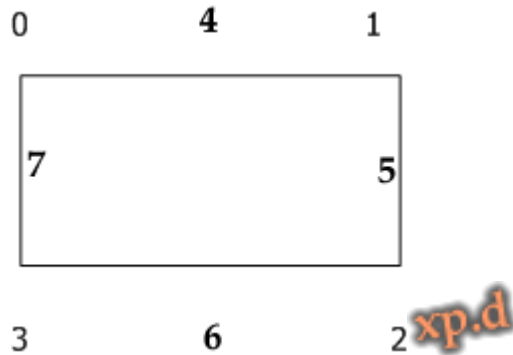
## 7 TMyCombine

这是联合图形的类，它定义了属性、事件和方法。

### 7.1 类图




坐标位置:



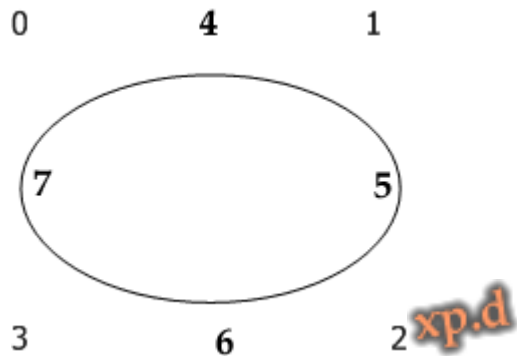
## 8 TMyEllipse

这是一个椭圆（圆）形的类，它定义了属性、事件、方法。

### 8.1 类图


<b>operations</b>
+ Draw(..) + GetInfo: string

坐标位置:



## 8.2 方法

### 8.2.1 Draw

```
procedure Draw(MyCanvas:TCanvas);override;
```

**描述:**

在MyCanvas上面一个椭圆形或圆形

### 8.2.2 GetCenterPoint

```
function GetCenterPoint:TMyPoint;override
```

**描述:**

函数GetCenterPoint重载父类的GetCenterPoint, 它返回椭圆或圆的中心点位置

### 8.2.3 GetCenterPointInZoom

```
function GetCenterPointInZoom:TMyPoint;override
```

**描述:**

函数GetCenterPointInZoom覆盖父类的GetCenterPointInZoom, 它返回当场缩放比例下椭圆或圆的中点位置

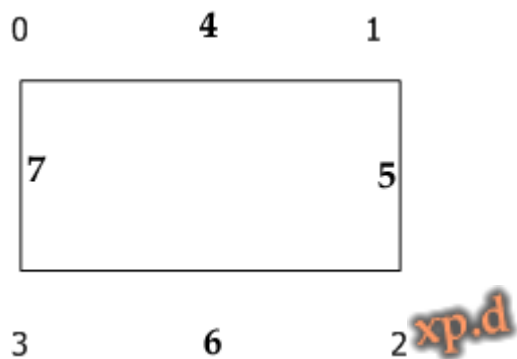
## 9 TMyGroup

这是组合图形的一个类，它定义了属性、事件、方法。

### 9.1 类图



坐标位置:



### 9.2 方法

#### 9.2.1 Draw

```
procedure Draw(MyCanvas:TCanvas);override;
```

**描述:**

在MyCanvas上画

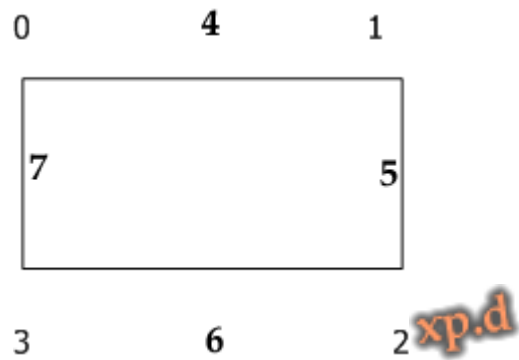
## 10 TMyImage

这是一个图片图形的类，它为一个图片图形定义了属性、事件、方法。

### 10.1 类图

☞ TMyImage
<b>attributes</b>
<ul style="list-style-type: none"> <li>* Bitmap: TBitmap</li> <li>* Border: Boolean</li> <li>* Brightness: Integer</li> <li>* Contrast: Integer</li> <li>* GrayScale: Boolean</li> <li>* Transparent: Boolean</li> </ul>
<b>operations</b>
<ul style="list-style-type: none"> <li>+ Create(..)</li> <li>+ Destroy</li> <li>+ Assign(..)</li> <li>+ Draw(..)</li> <li>+ LoadFromStream(..)</li> <li>+ SaveToStream(..)</li> </ul>

坐标位置:



## 10.2 属性

### 10.2.1 Bitmap

**property Bitmap:TBitmap**

**描述:**

为图片图形设置图像，如果要清除图片，设置Bitmap:=nil;

**示例:**

这个例子演示了在加载TMyImage时指定一个图片

**var**

mybitmap:TBitmap;

**begin**

if OpenPictureDialog1.Execute **then**

**begin**

myBitmap:=TBitmap.Create;

mybitmap.LoadFromFile(OpenPictureDialog1.FileName);

(AShape as TMyImage).Bitmap:=mybitmap;

MyBitmap.Free;

**end;**



end;

### 10.2.2 Border

property Border: Boolean; 

描述:

设置TmyImage图片是否显示边框

实例:

```
AShape.Pen.Width:=2;  
AShape.Pen.Color:=clRed;  
AShape.Border:=true;
```

### 10.2.3 Brightness

property Brightness: integer; 

描述:

设置TmyImage图片的亮度，其值 $-255 \leq \text{Brightness} \leq 255$

实例:

```
AShape.Brightness:=45;
```

### 10.2.4 Contrast

property Constrast: integer; 

描述:

设置TmyImage图片的对比度，其值 $-100 \leq \text{constrast} \leq 100$

实例:

```
AShape.Constrast:=45;
```

## 10.2.5 Grayscale

**property Grayscale:boolean;**



**描述:**

设置TmyImage图片是否显示灰度

**实例:**

```
AShape.Grayscale:=true;
```

## 10.2.6 Transparent

**property Transparent:Boolean**

**描述:**

设置TMyImage的图片的背景是否透明

## 10.2.7 TransparentColor

**property TransparentColor:TColor**

**描述:**

使用 TransparentColor 来画位图的透明色。当 TransparentMode 设置为 tmAuto 时，TransparentColor 返回位图的左下角的像素颜色。

当指定 TransparentColor 时，TransparentMode 自动设置为 tmFixed。如果你希望使用左下角的颜色设置透明色，可以设置 TransparentMode 为 tmAuto。

## 10.2.7 TransparentMode

**property TransparentMode:TTransparentMode**

**描述:**

当 TransparentMode 设置为 tmAuto, TransparentColor 返回位图左下角的像素颜色。当 TransparentMode 设置为 tmFixed, 将使用 TransparentColor 指定的颜色设置位图的透明色。

## 10.3 方法

### 10.3.1 Assign

**procedure Assign(Source:TMyShape);override;**

**描述:**

拷贝一个指定的图片图形

**示例:**

```
AShape.Assign(BShape);
```

### 10.3.2 Create

```
constructor Create(Aowner:TMyCAD);override;
```

**描述:**

构造函数  
初始化内部数据结构

### 10.3.3 Destroy

```
destructor Destroy;override;
```

**描述:**

先释放所有字段，最后调用Destroy方法

### 10.3.4 Draw

```
procedure Draw(MyCanvas:TCanvas);override;
```

**描述:**

在MyCanvas上画一个图片图形

### 10.3.5 LoadFromStream

```
procedure LoadFromStream(Astream:Tstream);override;
```

**描述:**

从流中加载一个图片图形

### 10.3.6 SaveToStream

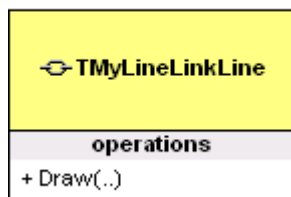
```
procedure SaveToStream(Astream:TStream);virtual;
```

**描述:**

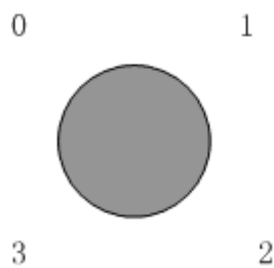
## 11 TMyLineLinkLine

这是一个连接点的类，它定义了属性、事件、方法

### 11.1 类图



坐标位置:



### 11.2 方法

#### 11.2.1 Draw

```
procedure Draw(MyCanvas:TCanvas);override;
```

**描述:**

在MyCanvas上画一个连接点

## 12 TMyElliArc

这是一个圆弧图形的类，它定义了属性、事件、方法

### 12.1 类图

↔ TMyElliArc	
attributes	
* ArcMode: TArcMode	
* ArcStyle: TArcStyle	
operations	
+ Create(..)	
+ Assign(..)	
+ Draw(..)	
+ FlipVert	
+ GetCenterPoint: TMyPoint	
+ GetCenterPointInZoom: TMyPoint	
+ GetInfo: string	
+ LoadFromStream(..)	
+ SaveToStream(..)	

坐标位置:



## 12.2 属性

### 12.2.1 ArcMode

**property ArcMode:TarcMode;**

**描述:**

有两种类型选择, amCircle 或 amEllipse

amCircle:



amEllipse:



## 12.2.2 ArcStyle

**property ArcStyle:TArcStyle;**

**描述:**

当ArcMode属性为amCircle时有3种选择值

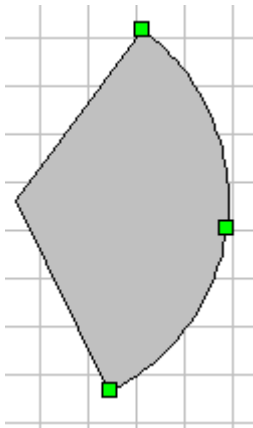
asArc:



asChord:



asSector:



## 12.3 方法

### 12.3.1 Assign

```
procedure Assign(Source:TMyShape);override;
```

描述:

拷贝一个圆弧图形

示例:

```
AShape.Assign(BShape);
```

### 12.3.2 Create

```
constructor Create(Aowner:TMyCAD);override;
```

描述:

构造函数

初始化内部数据结构, 属性ArcStyle的默认值是asArc, 属性ArcMode的默认值是amCircle

### 12.3.3 Draw

```
procedure Draw(MyCanvas:TCanvas);override;
```

描述:

在MyCanvas上画一个圆弧图形

### 12.3.4 GetCenterPoint

```
function GetCenterPoint:TMyPoint;override
```

描述:

GetCenterPoint重载了父类的这个方法, 它返回一个图形的中心点位置

### 12.3.5 GetCenterPointInZoom

```
function GetCenterPointZoom:TMyPoint;override
```

描述:

重载了父类的GetCenterPointInZoom, 它返回当前缩放比例下的中心点位置

### 12.3.6 LoadFromStream

```
procedure LoadFromStream(Astream:Tstream);override;
```

描述:

从流中加载一个图形



### 12.3.7 SaveToStream

**procedure SaveToStream(Astream:TStream);virtual;**

**描述:**

保存一个图形到流

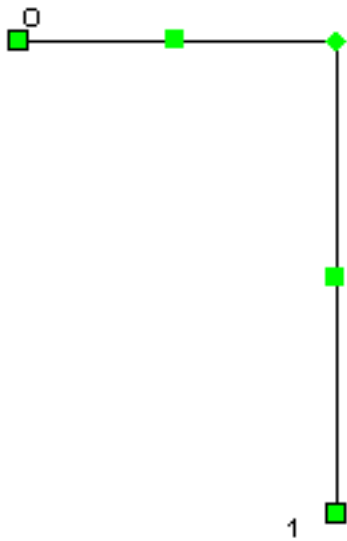
## 13 TmyLinkLine

连接线的类，它定义了属性、事件、方法，TCAD xp 标准版不支持连线功能

### 13.1 类图



坐标位置:



## 13.2 属性

### 13.2.1 LinkLineDrawStyle

property LinkLineDrawStyle:TLinkLineDrawStyle

描述:

定义连接线类型

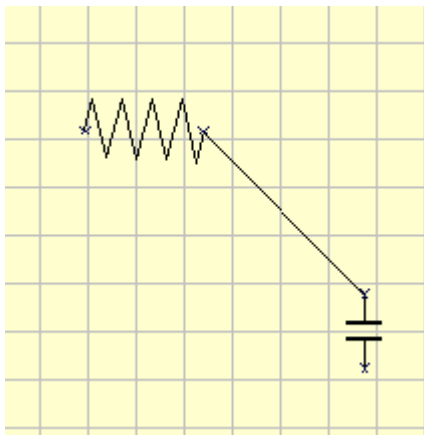
示例:

**Delphi 语法:**

AShape.LinkLineDrawStyle:=lidsFree;

**C++ 语法:**

AShape->LinkLineDrawStyle=lidsFree;

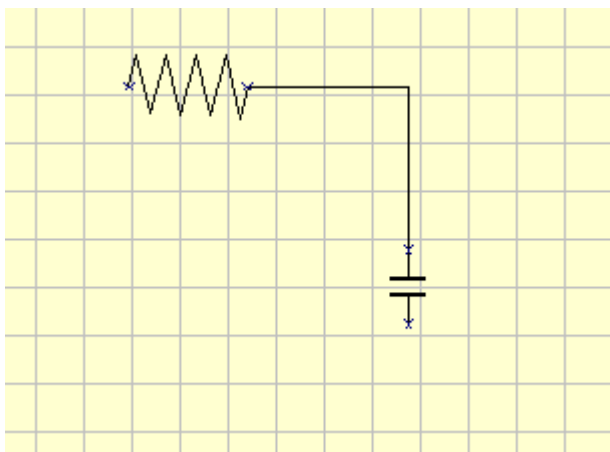


**Delphi 语法:**

```
AShape.LinkLineDrawStyle:=lldsHV;
```

**C++ 语法:**

```
AShape.LinkLineDrawStyle=lldsHV;
```



### 13.2.2 StartSpPtlId

```
property StartSpPtlId:integer;
```

**描述:**

起始图形的连接点标识符，可读写，如果没有起始图形，其值为-1

### 13.2.3 StartSpNo

```
property StartSpNo:integer;
```

**描述:**

起始图形的指针，其值为-1时，则表示没有起始图形

### 13.2.4 EndSpNo

```
property EndSpNo:integer;
```

**描述:**

结束图形的指针，其值为-1表示没有结束图形

### 13.2.5 EndSpPtlId

**property EndSpPtId:integer;**

**描述:**

结束图形连接点的标识符，运行时属性、可读写，如果没有结束图形，其值为-1

## 13.3 方法

### 13.3.1 Assign

**procedure Assign(Source:TmySource):override;**

**描述:**

拷贝一个线形，方法重载父类Assign方法

### 13.3.2 Create

**constructor Create(AOwner:TMyCAD):override;**

**描述:**

构造函数，覆盖父类的构造函数。先调用父类的构造函数，然后再初始化内部数据

### 13.3.3 Draw

**procedure Draw(MyCanvas:TCanvas):override;**

**描述:**

在MyCanvas上画线

### 13.3.4 LoadFromStream

**procedure LoadFromStream(Astream:TStream):override;**

**描述:**

从流中加载一个线形图形

### 13.3.5 CreateDestLink

**function CreateDestLink(AShapeId:integer;AshapeLinkId:integer):Boolean;**

**描述:**

AShapeId: 图形标识符;

AshapeLinkId: 图形的连接点标识符

它能以水平或垂直的方式连接两个图形, 通常用于流程图、电力图等

**返回值:**

true: 创建成功

false: 创建失败

### 13.3.6 CreateSrcLink

**function CreateSrcLink(AShapeId:integer;AshapeLinkId:integer):Boolean;**

**描述:**

它可以同一个外部图形创建一个连接关系

**参数:**

AShapeId: 起始图形的标识符

AshapeLinkId: 图形连接点的标识符

**返回值:**

true: 创建成功

false: 创建失败

### 13.3.7 GetEndPoint

**function GetEndPoint:TMyPoint;**

**描述:**

得到结束端图形的连接点

**返回值:**

假如没有结束端图形, 返回连接线自身的最后一点。否则返回结束端图形连接点

### 13.3.8 GetEndShape

**function GetEndShape:TMyShape;**

**描述:**

得到结束端图形

**返回值:**

nil: 没有结束图形

其它: 结束图形

### 13.3.9 GetStartPoint

**function GetStartPoint:TMyPoint;**

**描述:**

得到开始端图形的连接点

**返回值:**

返回起始连接图形的连接点，假如没有起始图形，返回连接线自身的最后一点

### 13.3.10 RemoveDestLink

**procedure RemoveDestLink;**

**描述:**

删除连接的目标图形的连接关系

### 13.3.11 RemoveSrcLink

**procedure RemoveSrcLink;**

**描述:**

删除连接的源图形的连接关系

### 13.3.12 GetStartShape

**function GetStartShape:TMyShape;**

**描述:**

得到起始连接的图形

**返回值:**

nil: 没有起始连接图形

else: 得到起始连接图形实例

### 13.3.13 RemoveAllLink

**procedure RemoveAllLink;**

**描述:**

删除所有的连接关系

### 13.3.14 SaveToStream

**procedure SaveToStream(Astream:TStream);virtual;**

**描述:**

保存一个图形到流

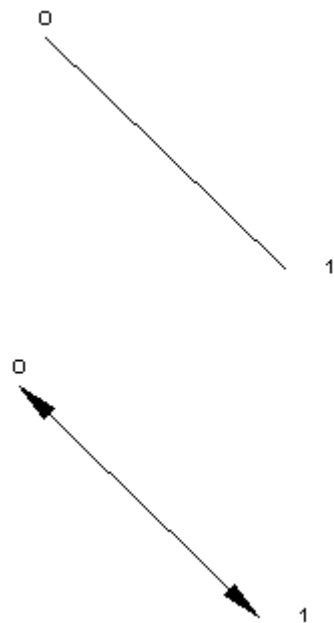
## 14 TMyLine

线图形的类，它定义了属性、事件、方法

### 14.1 类图

← TMyLine	
attributes	
* ArrowAngle: Integer	
* ArrowLength: Byte	
* ArrowOffset: Byte	
* ArrowStyle: TArrowStyle	
operations	
+ Create(..)	
+ Assign(..)	
+ Draw(..)	
+ GetInfo: string	
+ IsClickedMeBeforeWhichPoint(..): Boolean	
+ LoadFromStream(..)	
+ SaveToStream(..)	

坐标位置:



## 14.2 属性

### 14.2.1 ArrowAngle

**property ArrowAngle:integer;**

**描述:**

设置线和多段线的箭头角度，其值在0-359之间

**示例:**

MyCAD1.ArrowAngle:=10;



### 14.2.2 ArrowLength

**property ArrowLength:Byte;**

**描述:**

设置线和多段线的箭头长度，其值在10-50之间

### 14.2.3 ArrowOffset

**property ArrowOffset:byte;**

**描述:**

当画带箭头的线时，ArrowOffset指定箭头到线末端的偏移的像素，其值在0-255之间，默认值0

### 14.2.4 ArrowStyle

**property ArrowStyle:TarrowStyle;**

**描述:**

设置线和多段线的箭头类型

Anone:线

ALeft: 带左箭头的线或多段线

ARight: 带右箭头的线或多段线

ADouble: 两边都带箭头的线或多段线

## 14.3 方法

### 14.3.1 Assign

**procedure Assign(Source:TMyShape);override;**

**描述:**

拷贝一个线图形

**示例:**

AShape.Assign(BShape);

### 14.3.2 Create

**constructor Create(AOwner:TMyCAD);override;**

**描述:**

构造函数  
内部数据结构初始化

### 14.3.3 Draw

**procedure Draw(MyCanvas:TCanvas);override;**

**描述:**

在画布上画一线形

### 14.3.4 GetInfo

**function GetInfo:string;override**

**描述:**

覆盖基类的GetInfo，它返回线的长度，改变画布的大小，其值也会改变

### 14.3.5 IsClickedMeBeforeWhichPoint

**function IsClickedMeBeforeWhichPoint(var BeforeWhichPointID:integer;APoint:  
TPoint): Boolean;**

**描述:**

判断鼠标单击位置是否在线上，如果返回false，表示不在线上。

### 14.3.6 LoadFromStream

**procedure LoadFromStream(Astream:TStream);override;**

**描述:**

从流中加载一个图形

### 14.3.7 SaveToStream

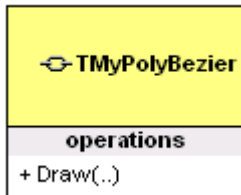
**procedure SaveToStream(Astream:TStream);virtual;**

**描述:**

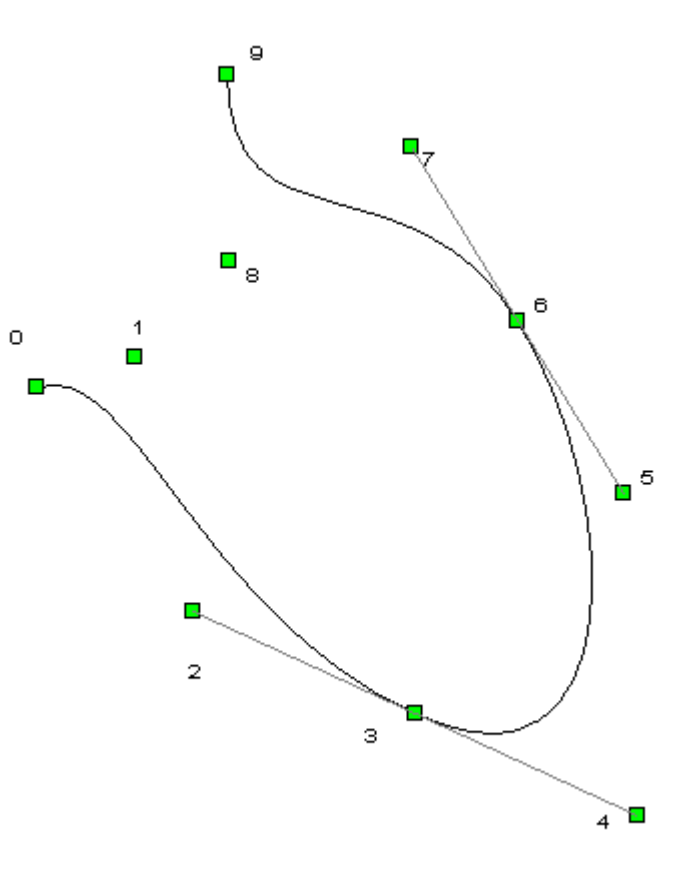
## 15 TMyPolyBezier

这是一个贝塞尔曲线图形的类，它定义了属性、事件、方法。

### 15.1 类图



坐标位置像线图形，不过多于两点



### 15.2 方法

#### 15.2.1 Draw

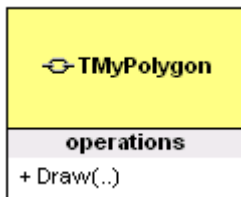
```
procedure Draw(MyCanvas:TCanvas):override;
```

描述:

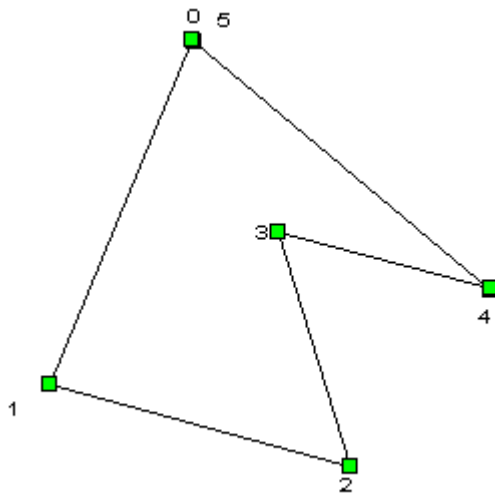
## 16 TMyPolygon

闭合多段线的类，它定义了属性、事件、方法。

### 16.1 类图



从标位置:



### 16.2 方法

#### 16.2.1 Draw

```
procedure Draw(MyCanvas:TCanvas);override;
```

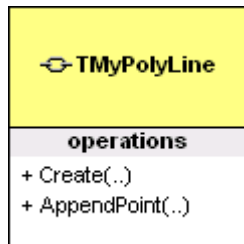
**描述:**

在画布上画多边形

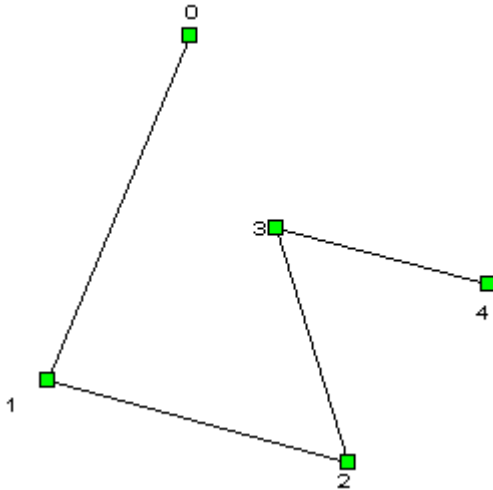
## 17 TMyPolyLine

多段线的类，它定义了属性、事件、方法

### 17.1 类图



坐标位置:



### 17.2 方法

#### 17.2.1 Create

**constructor Create(AOwner: TMyCAD); override;**

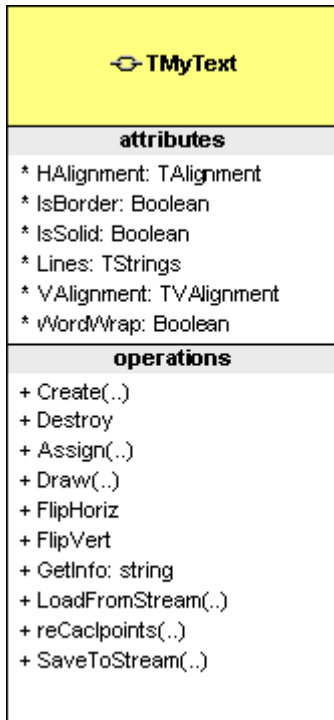
**描述:**

内部数据结构初始化，ArrowStyle默认值为asNone

## 18 TMyText

文本的类，它定义了属性、事件、方法

### 18.1 类图



从标位置:

0            4            1



3            6            2 xp.d

### 18.2 属性

#### 18.2.1 Halignment

property Halignment: TAlignment; xp.a

描述:

决定以外矩形为准文本的水平对齐方式

使用TAlignment的选项值来格式TMyText的文本水平对齐方式，其可选值如下：

taLeftJustify	左对齐
taCenter	中间对齐
taRightJustify	右对齐

### 18.2.2 IsBorder

**property IsBorder: Boolean;**

**描述：**

其值为真时，文本显示边框

### 18.2.3 IsSolid

**property IsSolid: Boolean;**

**描述：**

其值为真时，显示实心文本

### 18.2.4 Lines

**property Lines: TStrings;**

**描述：**

使用Lines在外接矩形内显示多行文本

**注意：**当Lines为空时，文本显示Info属性的值。

### 18.2.5 VAlignment

**property VAlignment: TAlignment;**

**描述：**

决定以外矩形为准文本的垂直对齐方式

使用TAlignment的选项值来格式TMyText的文本垂直对齐方式，其可选值如下：

vaTop	左对齐
vaMiddle	中间对齐

## 18.2.6 WordWrap

property WordWrap: Boolean; 

### 描述:

当WordWrap设置为True时, 允许多行显示文本, WordWrap为True时, 文本长度超出外接矩形时显示超出的文本在新的一行

当WordWrap设置为False时, 限制文本显示在一行, WordWrap为False时, 文本长度超出外接矩形时将显示在矩形外。

## 18.2.6 CharacterSpace

property CharacterSpace: Shortint;

### 描述:

为Info属性设置字符间距。

## 18.3 方法

### 18.3.1 Assign

procedure Assign(Source: TMyShape); override; 

### 描述:

拷贝一个文本图形属性

### 18.3.2 SaveToStream

procedure SaveToStream(Astream: TStream); virtual;

### 描述:

保存一个图形到流

### 18.3.3 LoadFromStream

procedure LoadFromStream(Astream: TStream); override;

### 描述:

从流中加载一个文本



### 18.3.4 Create

**constructor Create(AOwner:TMyCAD);override;**

**描述:**

构造函数

初始化内部数据结构, 属性IsBorder值为false, IsSolid值为true

### 18.3.5 Destroy

**procedure Draw(MyCanvas:TCanvas);override;**

**描述:** 首先释放自己的内存, 然后调用父类的Destroy方法

### 18.3.6 Draw

**procedure Draw(MyCanvas:TCanvas);override;**

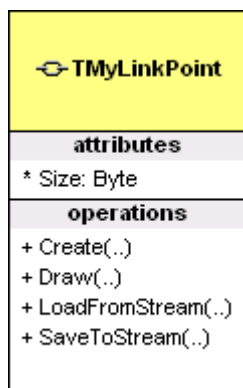
**描述:**

在画布上画一个文本图形

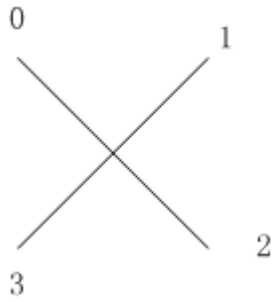
## 19 TMyLinkPoint

这是一个连接点的类, 它定义了属性、事件、方法, 它只对库有用

### 19.1 类图



坐标位置:



## 19.2 属性

### 19.2.1 Size

**property Size:byte;**

**描述:**

设置大小

## 19.3 方法

### 19.3.1 SaveToStream

**procedure SaveToStream(Astream:TStream);virtual;**

**描述:**

保存一个连接点图形到流

### 19.3.2 LoadFromStream

**procedure LoadFromStream(Astream:TStream);override;**

**描述:**

从流中加载一个连接点

### 19.3.3 Create

**constructor Create(AOwner:TMyCAD);override;**

**描述:**

初始化内部数据结构，大小默认值为8个像素

### **19.3.4 Draw**

**procedure Draw(MyCanvas:TCanvas);override;**

**描述:**

在画布上画一个连接点

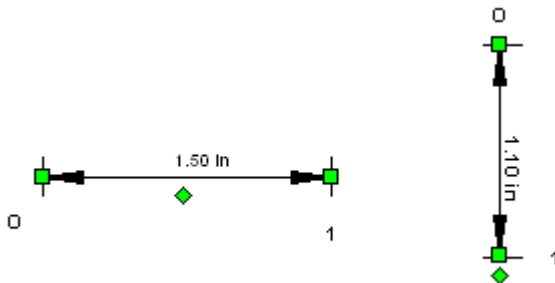
## **20 TMyRuleLine**

线段图形的类，它定义了属性、事件、方法。

### **20.1 类图**

 <b>TMyRuleLine</b>
<b>attributes</b>
* ShowUserInfo: Boolean * TickStyle: TTickStyle * UserInfo: string
<b>operations</b>
+ Create(..) + Assign(..) + Draw(..) + LoadFromStream(..) + SaveToStream(..)

坐标位置:



## 20.2 属性

### 20.2.1 UserInfo

**property UserInfo:string;**

**描述:**

设置用户信息

### 20.2.2 ShowUserInfo

**property ShowUserInfo:boolean;**

**描述:**

当其值为true时，线段显示用户信息，否则显示线段长度

### 20.2.3 TickStyle

**property TickStyle:TTickStyle;**

**描述:**

`TtickStyle=(tsLine,tsNone);`

当设置为tsLine时，在线段两端显示两条小线

当设置为tsNone时，不显示两条小线

## 20.3 方法

### 20.3.1 Assign

`procedure Assign(Source:TMyShape);override;`

描述:

拷贝一条线段

### 20.3.2 Create

`constructor Create(AOwner:TMyCAD);override;`

描述:

初始化内部数据结构，其属性值ArrowStyle默认为ADouble、UserInfo默认为空、ShowUserInfo默认为false、TickStyle默认为tsLine

### 20.3.3 LoadFromStream

`procedure LoadFromStream(Astream:TStream);override;`

描述:

从流中加载一条线段

### 20.3.4 Draw

`procedure Draw(MyCanvas:TCanvas);override;`

描述:

在画布上画一条线段

### 20.3.5 SaveToStream

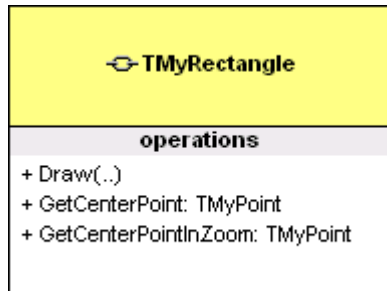
`procedure SaveToStream(Astream:TStream);virtual;`

描述:

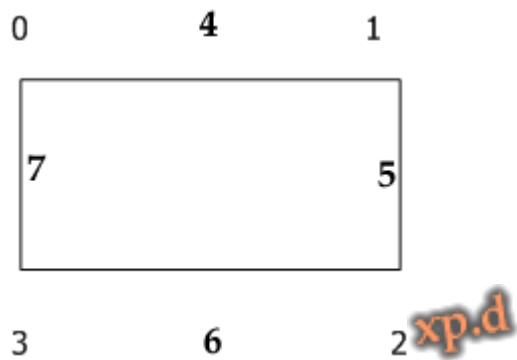
## 21 TMyRectangle

矩形图形的类，它为矩形定义了属性、事件、方法。

### 21.1 类图



坐标位置:



### 21.2 方法

#### 21.2.1 Draw

```
procedure Draw(MyCanvas:TCanvas);override;
```

**描述:**

在画布上画一个矩形图形

#### 21.2.2 GetCenterPoint

```
function GetCenterPoint:TMyPoint;override;
```

**描述:**

重载基类的GetCenterPoint, 它返回矩形的中心点位置

### 21.2.3 GetCenterPointInZoom

**function GetCenterPointInZoom:TMyPoint;override;**

**描述:**

重载基类的GetCenterPointInZoom, 它返回当前缩放比例下矩形中心点位置

### 21.2.4 GetInfo

**function GetInfo:String;override;**

**描述:**

重载基类的GetInfo, 它返回矩形的面积, 当改变矩形所有者的单位时, 矩形面积会改变。

## 21.3 属性

### 21.3.1 AssociateSideResizing

**Property AssociateSideResizing:Boolean**

**描述:**

该属性指定当拖动矩形边框时, 相邻的边是否关联。

**实例:**

MyCAD1.AssociateSideResizing:=true;

### 21.3.2 ShowSideHot

**Property ShowSideHot:Boolean**

**描述:**

是否显示矩形两边的热点

**实例:**

MyCAD1.ShowSideHot:=true;

### 21.3.2 ShowHorizontalSideHot

**Property** ShowHorizontalSideHot:Boolean

描述:

是否显示矩形水平两边的热点

实例:

```
MyCAD1.ShowHorizontalSideHot:=true;
```

### 21.3.2 ShowVerticalSideHot

**Property** ShowVerticalSideHot:Boolean

描述:

是否显示矩形垂直两边的热点

实例:

```
MyCAD1.ShowVerticalSideHot:=true;
```

## 22 TUserData

这个类能让你存储一个图形或组合图形自己的数据。

### 22.1 类图



<b>↔ TUserData</b>
<b>attributes</b>
* UserDataRecord: TArrUserDataRecord
<b>operations</b>
+ Create + AddKeyandValue(..): Boolean + Assign(..) + ChangeValueByKey(..) + ClearAll + DeleteRecordByKey(..) + GetCount: Integer + GetKeyByNo(..): string + GetValueByKey(..): string + ReNameTitle(..)

## 22.2 属性

### 22.2.1 UserDataRecords

**property UserDataRecords:TarrUserDataRecord;**

**描述:**

存储自定义属性

## 22.3 方法

### 22.3.1 Create

**constructor Create;**

**描述:**

初始化内部数据

### 22.3.2 AddKeyAndValue

**function AddKeyAndValue(ConstAKey,Avalue:string):boolean;**

**描述:**

在自定义数据实例中增加一个关键字和值

**返回值:**

true: 增加成功

false: 增加失败

示例:

```
UserData1.AddKeyValue('Name','John');
```

### 22.3.3 Assign

```
procedure Assign(Source:TUserData);
```

**描述:**

拷贝另一个自定义数据实例

**参数:**

Source: 其它存在的用户自定义数据实例

示例:

```
UserData1.Assign(UserData0);
```

### 22.3.4 ChangeValueByKey

```
function ChangeValueByKey(const Akey:string;AValue:string):boolean;
```

**描述:**

通过关键字来修改其值, 如果关键字不存在, 返回false, 存在返回true

**参数:**

Akey:关键字

AValue: 新值

示例:

```
If UserData1.ChangeValueByKey('Name','Rose') then  
  ShowMessage('Changed!');
```

### 22.3.5 ClearAll

**procedure ClearAll;**

**描述:**

清除所有数据

### 22.3.6 DeleteRecordByKey

**function DeleteRecordByKey(AKey:String):Boolean;**

**描述:**

通过关键字删除一条记录

### 22.3.7 GetCount

**function GetCount:integer;**

**描述:**

得到记录总数

### 22.3.8 GetKeyByNo

**function GetKeyByNo(const ANo:integer):string;**

**描述:**

通过序号得到关键字，如果序号不存在，返回false，否则返回true

**参数:**

ANo: 记录在记录集中的序号，从0开始的

**示例:**

```
ShowMessage('the first key name is:'+UserData1.GetKeyByNo(0));
```

### 22.3.9 GetValueByKey

**function GetValueByKey(const AKey:string):string;**

**描述:**

通过关键字得到值，如果关键字不存在，返回false，否则返回true

**参数:**

AKey: 关键字

**示例:**

```
ShowMessage('the first key name is: ' + UserData1.GetKeyByNo(0)+' Value: '+UserData1.GetValueByKey(UserData1.GetKeyByNo(0) ));
```

### 22.3.10 ReNameKey

**function ReNameKey(OldKey,NewKey:string):boolean;**

**描述:**

重命名一个关键字，如果关键字不存在，返回false，否则返回true

## 23 TGridObject

↔ TGridObject
<b>attributes</b>
* GridColor: TColor
* GridHeight: Byte
* GridPenSize: Byte
* GridShow: Boolean
* GridType: TGridType
* GridWidth: Byte
<b>operations</b>
+ Create
+ Destroy



### 23.1 GridColor

**Property GridColor:Tcolor;**

**描述:**

设置网格颜色

**实例:**

```
MyCAD1.GridOperation.GridColor:=clBlack;
```

### 23.2 GridHeight

**Property GridHeight:byte;**

**描述:**

设置网格高度，单位是像素

**实例:**

MyCAD1.GridOperation.GridHeight:=12;

### 23.3 GridPenSize

**property GridPenSize:Byte;**

**描述:**

设置网格线的宽度

**实例:**

MyCAD1.GridOperation.GridPenSize:=3;

### 23.4 GridShow

**Property GridShow:Boolean;**

**描述:**

设置是否显示网格

**实例:**

MyCAD1.GridOperation.GridShow:=false;

### 23.5 GridType

**Property GridType:TGridType;**

**描述:**

设置网格类型

实例:

```
MyCAD1.GridOperation.GridType:=gLine;
```

## 23.6 GridWidth

**Property GridWidth:byte;**

描述:

设置网格宽度，单位是像素

实例:

```
MyCAD1.GridOperation.GridWidth:=12;
```

## 23.4 GridInZoom

**Property GridInZoom:Boolean;**

描述:

设置网格是不是随画布的缩放而缩放。

实例:

```
MyCAD1.GridOperation.GridInZoom:=true;
```

## 24 TMyLibrary

### 24.1 属性

#### 24.1.1 cmbShapesCount

**property cmbShapesCount:Integer;**

描述:

存储库文件有多少个库对象。

#### 24.1.2 LibFileName

**property LibFileName:string;**

描述:

存储库文件路径。

## 24.2 方法

### 24.1.1 AppendNewShape

```
procedure AppendNewShape(CmbShapeName:String;mMyShapes:TArrMyShape;  
mUserDataRecords:TUserData;BaseWidth,BaseHeight:integer);
```

**描述:**

增加一个新的库对象到库文件。

**参数:**

CmbShapeName: 设置对象的名称。

mMyShapes: 库对象子图形对象数组。

mUserDataRecords: 用户自定义数据。

BaseWidth: 设置库对象预览图标的宽度。

BaseHeight: 设置库对象预览图标的高度。

### 24.2.2 Clear

```
procedure Clear;
```

**描述:**

清除图形库的文件流对象。

### 24.2.3 Create

1. **constructor** Create; overload;

**描述:**

创建库的文件流和 TMyCAD 对象。

2. **constructor** Create(FileName:string); overload;

**描述:**

基于已存在的库文件来创建文件流对象和 TMyCAD 对象。

### 24.2.4 DeleteShape

```
procedure DeleteShape(CmbShapeName:String);
```

**描述:**

根据名称删除库对象。

### 24.2.5 Destroy

**destructor** Destroy; override;

**描述:**

释放库文件流和 TMyCAD 对象。

### 24.2.6 GetCmbShape

**function** GetCmbShape(CmbShapeName: String; var mLinkPoints: TArrMyPoint;  
var mUserDataRecords:TUserData ; var mMyShapes: TArrMyShape;var  
mWidth,mHeight:integer; IsconverttoLink:boolean=false): Boolean;

**描述:**

得到库对象。

**参数:**

CmbShapeName: 指定库对象的名称。  
mLinkPoints: 存储连接点。  
mUserDataRecords: 存储用户自定义数据。  
mMyShapes: 存储库对象的子图形。  
mWidth: 指定库对象预览图标的宽度  
mHeight: 指定库对象预览图标的高度  
IsconverttoLink: 默认值为 false.

**返回值:**

true: 成功。  
false: 失败。

### 24.2.7 GetCmbshapelImage

**procedure** GetCmbShapelImage(CmbShapeName: String; Square: double; ABitmap:  
TBitmap;BackColor:TColor);

**描述:**

得到库对象的预览图像。

**参数:**

CmbShapeName: 库对象名称。



**Square:** 预览图像容器的宽度。

**ABitmap:** 临时的图像画布。

**BackColor:** 预览图像容器的背景颜色。

### 24.2.8 GetCmbShapesNameByID

```
function GetCmbShapesNameByID(Id:integer): string;
```

**描述:**

通过 ID 来得到库对象。

**参数:**

Id: 库对象的序列号。

**返回值:**

得到库对象的名称。

### 24.2.9 GetExistShape

```
function GetExistShape(CmbShapeName:string;var Position,Size:integer):  
    Boolean;
```

**描述:**

得到已经存在的库对象。

**参数:**

**CmbShapeName:** 库对象名称。

**Position:** 文件流的起始位置。

**Size:** 库对象的文件流大小。

**返回值:**

true: 成功。

false: 失败。

### 24.2.10 InsertNewShape

```
procedure InsertNewShape(FromPosition:integer;CmbShapeName:String;
```

## TCAD for Delphi & C++ Builder & Kylix & Vcl.net

---

```
mMyShapes:TArrMyShape;mUserDataRecords:TUserData;BaseWidth,  
BaseHeight:integer);
```

### 描述:

插入一个新的库对象到库文件流。

### 参数:

FromPostion: 文件流的起始位置。

CmbShapeName: 库对象名称。

mMyShapes: 库对象的子图形。

mUserDataRecords: 用户自定义数据。

BaseWdith: 指定库对象预览图标的宽度。

BaseHeight: 指定库对象预览图标的高度。

### 24.2.11 LoadFromFile

```
function LoadFromFile(FileName:String): Boolean;
```

### 描述:

加载一个库文件。

### 参数:

FileName: 文件路径。

### 返回值:

true: 成功。

false: 失败。

### 24.2.12 LoadFromOldFile

```
function LoadFromOldFile(FileName:String): Boolean;
```

### 描述:

加载一个旧的库文件。

### 参数:

FileName: 库文件路径。

### 返回值:

true: 成功。

false: 失败。

### 24.2.13 MoveCmbShapeBeforeByName

**function** MoveCmbShapeBeforeByName(thisCmbShapeName:string;  
referCmbShapeName:string): Boolean;

**描述:**

移动一个库对象到另一个库对象之前。

**参数:**

thisCmbShapeName: 移动的库对象。  
referCmbShapeName: 另一个库对象。

**返回值:**

true: 成功。  
false: 失败。

### 24.2.14 ReadCmbShapeName

**function** ReadCmbShapeName(FromPosition :LongInt;var Count,  
NowPosition:LongInt): string;

**描述:**

从流中读取库对象。

**参数:**

FromPosition: 文件流的起始位置。  
Count: 库对象名称的流大小。  
NowPosition: 存储当前文件流的位置。

**返回值:**

库对象名称。

### 24.2.15 ReadCmbShapeSize

**function** ReadCmbShapeSize(FromPosition :LongInt;var Count,  
NowPosition:LongInt): Integer;

**描述:**

读库对象文件流的大小。

**参数:**

FromPosition: 文件流的起始位置。  
Count: 库对象的流大小。  
NowPosition: 存储文件流的当前位置。

**返回值:**

库对象的流大小。

### 24.2.16 ReadLibTag

```
function ReadLibTag(var Count,NowPosition:integer): string;
```

**描述:**

读取库文件的版本标签。

**参数:**

Count: 版本标签的流大小。

NowPosition: 存储当前流的位置。

**返回值:**

版本标签。

### 24.2.17 ReadShapes

```
function ReadShapes(var mMyShapes:TArrMyShape;var mLinkPoints:TArrMyPoint;  
    FromPosition :LongInt;var Count,NowPosition:LongInt; mWidth,  
    mHeight:integer; IsConverttoLink:boolean=false): Boolean;
```

**描述:**

从文件流中读取图形。

**参数:**

mMyShapes: 存储图形的数组。

mLinkPoints: 存储连接点的数组。

FromPostion: 文件流的起始点。

Count: 库对象的文件流大小。

NowPosition: 存储文件流的当前流的位置。

mWidth: 预览图标的宽度。

mHeight: 预览图标的高度。

**返回值:**

true: 成功。  
false: 失败。

### 24.2.18 ReadUserDataRecords

```
function ReadUserDataRecords(FromPosition :LongInt;var  
    mUserDataRecords:TUserData;var Count,NowPosition:LongInt): Boolean;
```

**描述:**

从流中读取用户自定义数据。

**参数:**

FromPosition: 流的起始位置。  
mUserDataRecords: 存储用户自定义数据的数组。  
Count: 用户自定义数据流的大小。  
NowPosition: 存储文件流的当前位置。

**返回值:**

true: 成功。  
false: 失败。

### 24.2.19 ReadWidthHeight

```
function ReadWidthHeight(FromPosition :LongInt;var mWidth,mHeight:integer;  
    var Count,NowPosition:LongInt): Boolean;
```

**描述:**

读取库对象预览图像的宽度和高度。

**参数:**

FromPosition: 流的起始位置。  
mWidth: 存储库对象预览对象的宽度。  
mHeight: 存储库对象预览对象的高度。  
Count: 预览对象的宽度和高度的流大小。  
NowPosition: 流对象的当前位置。

**返回值:**

true: 成功。  
false: 失败。

### 24.2.20 RenameCmbShapeName

**function** RenameCmbShapeName(const OldName,NewName:string): Boolean;

**描述:**

重命名一个库对象。

**参数:**

OldName: 库对象的旧名称。  
NewName: 库对象的新名称。

**返回值:**

true: 成功。  
false: 失败。

### 24.2.21 RepalceExistShape

**function** RepalceExistShape(CmbShapeName:string;mMyShapes:TArrMyShape;  
mUserDataRecords: TUserData; BaseWidth,BaseHeight:integer): Boolean;

**描述:**

替换一个已存在的库对象。

**参数:**

CmbShapeName: 库对象的名称。  
mMyShapes: 存储图形的数组。  
mUserDataRecords: 存储用户自定义数据的数组。  
BaseWidth: 库对象预览图像的宽度。  
BaseHeight: 库对象预览图像的高度。

**返回值:**

true: 成功。  
false: 失败。

### 24.2.22 ReplaceoldTagwithNewTag

**procedure** ReplaceoldTagwithNewTag;

**描述:**

用新的版本标签代替旧的。

### 24.2.23 SaveToFile

**procedure** SavetoFile(FileName:String);

**描述:**

保存文件流到一个文件。

**参数:**

FileName: 文件路径。

### 24.2.24 WriteCmbShapeName

**function** WriteCmbShapeName(CmbShapeName:String;FromPosition :LongInt;var  
Count,NowPosition:LongInt): Boolean;

**描述:**

写库对象名称。

**参数:**

CmbShapeName: 库对象名称。

FromPosition: 流的起始位置。

Count: 库对象名称的流大小。

NowPosition: 存储流的当前位置。

**Return:**

true: 成功。

false: 失败。

### 24.2.25 WriteCmbShapeSize

**function** WriteCmbShapeSize(ASize:integer;FromPosition :LongInt;var Count,  
NowPosition:LongInt): Boolean;

**描述:**

写库对象的大小。

**参数:**

ASzie: 库对象的大小。

FromPosition: 流的起始位置。

Count: 存储库对象的流的大小。

NowPosition: 存储流的当前位置。

**返回值:**

true: 成功。

false: 失败。

### 24.2.26 WriteLibTag

**function** WriteLibTag(var Count,NowPosition:integer): Boolean;

**描述:**

写库对象的版本标签。

**参数:**

Count: 标签的大小。

NowPosition: 存储流的当前位置。

**返回值:**

true: 成功。

false: 失败。

### 24.2.27 WriteShapes

**function** WriteShapes(mMyShapes:TArrMyShape;BaseWidth, baseHeight:integer;  
FromPosition :LongInt;var Count,NowPosition:LongInt): Boolean;

**描述:**

写库对象的图形到流中。

**参数:**

mMyShapes: 存储所有图形的数组。

BaseWidth: 库对象预览图形的宽度。

BaseHeight: 库对象预览图形的高度。

FromPosition: 流的起始点。

Count: 图形数组的流大小。

NowPosition: 存储流的当前位置。

**返回值:**



true: 成功。  
false: 失败。

### 24.2.28 WriteUserDataRecords

```
function WriteUserDataRecords(var mUserDataRecords:TUserData; FromPosition  
:LongInt;var Count,NowPosition:LongInt): Boolean;
```

**描述:**

写用户自定义数据到文件流。

**参数:**

mUserDataRecords: 存储用户自定义数据。  
FromPosition: 流的起始位置。  
Count: 用户自定义数据的流大小。  
NowPosition: 存储流的当前位置。

**返回值:**

true: 成功。  
false: 失败。

### 24.2.29 WriteWidthHeight

```
function WriteWidthHeight(BaseWidth, baseHeight:integer;FromPosition  
:LongInt;var Count,NowPosition:LongInt): Boolean;
```

**描述:**

写库对象的预览图像的宽和高到流中。

**参数:**

BaseWidth: 预览图像的宽。  
BaseHeight: 预览图像的高。  
FromPosition: 流的起始位置。  
Count: 预览图像的宽和高的流大小。  
NowPosition: 存储流的当前位置。

**返回值:**

true: 成功。

false: 失败。

## 24.3 字段

### 24.3.1 IsOldLib

IsOldLib: Boolean;

**描述:**

指定库文件是旧的还是新的。

### 24.3.2 LibStream

LibStream: TMemoryStream;

**描述:**

存储库文件的文件流。

### 24.3.3 mcad

mcad: TMyCAD;

**描述:**

存储TMyCAD的对象。

## 24 关于Crystal Component

我们的宗旨是为您提供有效控件，使你能简单、方便的制作应用程序。从1998年起，我们就一直提供图形图像控件。

**我们的网址**

<http://www.codeidea.com>，你可以通过该网址得到所有有关我们产品的信息。

**Email:**

[webmaster@codeidea.com](mailto:webmaster@codeidea.com)

**电话:**

+86 572 2607144

**地址:**

湖州市青铜路699号科技创业园A座303-304室

**邮编:**

313000