交互式CAD系统开发基础系列丛书

交互式CAD系统开发基础系列丛书

交互式CAD系统开发基础系列丛书

# 用VB.NET和VC#.NET开发交互式CAD系统

苏金明　周建斌　编著

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

交互式CAD系统开发基础系列丛书

# 交互式CAD系统开发基础系列丛书

交互式CAD系统开发基础系列丛书

1. 用 Visual Basic 开发交互式 CAD 系统
2. 用 Visual C++.NET 开发交互式 CAD 系统
3. 用 VB.NET 和 VC#.NET 开发交互式 CAD 系统

ISBN 7-5053-9443-6

9 787505 394438 >

责任编辑：王昌铭
封面设计：闫欢玲



ISBN 7-5053-9443-6 定价：38.00 元 （含光盘）

交互式 CAD 系统开发基础系列丛书

# 用 VB.NET 和 VC#.NET 开发交互式 CAD 系统

苏金明　周建斌　编著

電子工業出版社
Publishing House of Electronics Industry
北京 • BEIJING

## 内 容 简 介

本书主要结合 VB.NET 和 VC#.NET 两种语言介绍了创建交互式 CAD 系统的基本思路和技术，分别给出了两种语言的小系统完整代码，并讨论了技巧实现的其他可能性以及系统代码的改进方法。

本书前 3 章主要介绍语言基础和.NET 框架基础，第 4 章至第 8 章结合一个 CAD 小系统的创建详细地介绍了交互式 CAD 系统的组织思路和基本技术，第 9 章至第 11 章介绍了更多的技巧实现方法和系统优化方法，第 12 章结合 CAD 编程进行了一些设计模式方面的讨论；写作过程中注意了循序渐进的讲解原则，内容适合不同学习阶段的读者。

书中所有示例程序都通过调试，并放在随书的光盘上，以便于学习和交流。

本书可供从事图形学、CAD 技术以及编程技术的有关工程技术人员、程序员、大学生、研究生阅读参考。



**图书在版编目（CIP）数据**

用 VB.NET 和 VC#.NET 开发交互式 CAD 系统/苏金明，周建斌编著. —北京：电子工业出版社，2004.1
(交互式 CAD 系统开发基础系列丛书)
ISBN 7-5053-9443-6

Ⅰ. 用… Ⅱ. ①苏…②周… Ⅲ. ①BASIC 语言—程序设计②C 语言—程序设计③计算机辅助设计
Ⅳ. ①TP312②TP391.72

中国版本图书馆 CIP 数据核字（2003）第 112512 号

责任编辑：王昌铭
印　　刷：北京天竺颖华印刷厂
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编　100036
经　　销：各地新华书店
开　　本：787×1 092　1/16　印张：22.5　字数：576 千字
印　　次：2004 年 1 月第 1 次印刷
印　　数：5000 册　　定价：38.00 元　（含光盘 1 张）

凡购买电子工业出版社的图书，如有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系。联系电话：（010）68279077。质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

# 前　言

交互式 CAD 技术是目前应用十分广泛的一种绘图技术，它的主要特点是可以把创建和编辑图形的动态过程展示出来，因而具有更强的交互性能。为了使更多的人了解和掌握这一技术，我们编写了这套交互式 CAD 系统开发基础系列丛书。丛书一共 3 本，分别结合不同的语言进行介绍。本书是其中的第 3 本，主要介绍用 VB.NET 和 VC#.NET 开发交互式 CAD 系统。

## 本书的内容

本书结合 VB.NET 和 VC#.NET 两种语言介绍了创建交互式 CAD 系统的基本思路和技术。

第 1 章至第 3 章主要介绍后面各章可能用到的语言基础和.NET 框架基础。其中：第 1 章对.NET 进行了比较简略的介绍；第 2 章介绍了 VB.NET 和 VC#.NET 的面向对象编程方法，需要指出的是，VB.NET 已经是完全面向对象的编程语言了；第 3 章介绍了 GDI+编程的一些方法和效果，GDI+是 GDI 的改进版本。

第 4 章至第 8 章结合一个 CAD 小系统的创建详细地介绍了交互式 CAD 系统的组织思路和基本技术。这部分的第 4 章介绍了系统中对象的提取、组织及代码实现情况，给出了一些相关的 UML 类图；然后建立通用坐标系，确定它与页面坐标系、设备坐标系之间的转换关系。第 5 章和第 6 章介绍通过建立基本图元类和交互绘图类来实现图元的鼠标交互绘制。第 7 章介绍在基本图元类中添加图元拾取的方法，添加选择图元的交互绘图类，实现图元的拾取、选择和删除。第 8 章讲解了通过对图元的控制点进行几何变换来实现图元变换，包括图元的平移变换、旋转变换、镜像变换和比例变换。

第 9 章介绍了 GDI+提供的一些交互绘图技巧，包括线形图元和区域的拾取、图元的几何变换等。

第 10 章介绍与图元相交关系有关的一些交互技术，给出了一个相交线与直线段相交的实例，讨论了用矩形框拾取图元的思路和方法，最后结合 GDI+提供的对象方法介绍了计算其他图元与曲线图元交点的思路和方法。

第 11 章介绍了对第 4 章至第 8 章建立的小系统进行优化的一些手段，包括建立强键值的集合类、获得 For Each、纠正圆整错误、用 GDI+进行交互绘图、界面美化和数据存盘、加载等。

第 12 章结合 CAD 编程技巧讨论了几种设计模式的应用，主要讨论了状态模式、访问者模式、模板方法模式和策略模式。另外，还简略地介绍了工厂方法模式、观察者模式、命令模式和记事模式。

## 本书的特色

### 1. 采用了面向对象的编程技术

本书主要的实例都采用了面向对象的编程技术，其中第 4 章至第 8 章建立的 CAD 小系统是一个比较完整的程序。它不仅仅演示了创建 CAD 系统的思路和方法，还是一个很好的

面向对象编程实例。

**2．实例丰富**

本书实例比较丰富，其特点就是用实例“说话”。全书示例程序分 VB.NET 和 VC#.NET 两种语言，大大小小各有 50 多个，而且所有的示例程序都在 CAD 的语境中进行绘制，基本上没有与 CAD 无关的实例。这样做的好处是可以为读者创造一个学习的氛围，不分散注意力。

**3．循序渐进的讲解方式**

在章节安排上注意了循序渐进原则。首先介绍一些基础知识，然后是 CAD 系统的创建，最后介绍更多的技巧实现方法和优化方法，并结合 CAD 编程讨论设计模式的应用。如果您对 VB.NET 或 VC#.NET 已经有了比较多的了解，而且熟悉 GDI+ 和互操作，可以直接从第 4 章开始阅读。不过建议您还是从头开始过一遍。

## 本书的读者

所有关心图形学、CAD 技术的大学生、研究生和其他相关人员都可以是本书的读者。要求读者最好有一定的.NET 编程基础和面向对象编程经验。对于初学者，前面 3 章提供了快速入门的捷径。

为方便您学习本书的内容，我们将所有示例程序都放在随书的光盘上。所有程序都通过调试。

写作过程中，始终得到黄国明的支持，深表感谢！苏华惠做了部分录入工作，刘玉珊帮助搜集资料，一并表示感谢！

由于作者水平有限，书中难免存在谬误和缺点，谨请批评指正。请通过电子邮件与我们联系：

苏金明：s_jm@263.net.cn

周建斌：zjb@cdut.edu.cn

作　者

2003.10.12

# 目　录

# 第 1 章　.NET 基 础

本书不是 VB.NET 和 VC#.NET 方面的入门教程，不会对.NET 环境和语言的特点进行系统的介绍。但是，有一些我们后面要用到的与 VB 6.0 等有显著差别的特点还是需要说明一下。这部分内容包括.NET 的开发环境、基本语法特点和名字空间等。

## 1.1　.NET 开发环境

VB.NET、VC#.NET 和 VC++.NET 等共用一个.NET 平台，所以安装新的 Visual Studio 版本时它们全部被安装。新建项目时显示如图 1-1 所示的“新建项目”对话框。在“项目类型”列表框中选择语言类型，在“模板”文件列表框中选择具体的应用类型。

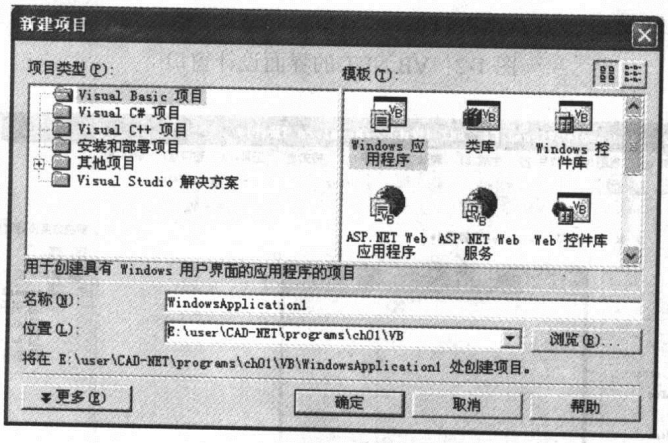


图 1-1　“新建项目”对话框

VB.NET 和 VC#.NET 的界面设计窗口如图 1-2 和图 1-3 所示，二者大同小异。VB.NET 和 VC#.NET 中的界面设计方法大致相同，不分别介绍。下面主要介绍 VB.NET 与 VB 6.0 在界面设计方面的差别。与 6.0 版本相比，VB.NET 中主要有以下几方面的不同。

### 1．窗体的变化

虽然在本书的 VB 篇中曾经介绍过，窗体也是类，但是在 VB 6.0 中，窗体和类基本上是分开来讲的。在 VB.NET 中，窗体是类这个问题就公开了，如果新建的窗体名称为 Form1，则会自动创建一个 Form1 类，它就相当于 6.0 中的窗体模块。另外，添加了一些窗体属性。在 VB 6.0 中，将窗体置于顶部需要调用 API 函数，现在不必了，将它的 Topmost 属性设置为 True 就行了。设置 Visible 属性，可以控制窗体的可见性；利用 Opacity 属性，可以将窗体设置为透明；利用 StartPosition 属性，可以设置窗体启动时在屏幕上的显示位置。现在有了模式对话框的概念，如果用 ShowDialog 方法调用窗体时将该窗体设置为有模式的对话框，则必须关闭该窗体以后才能对其他窗体进行操作。

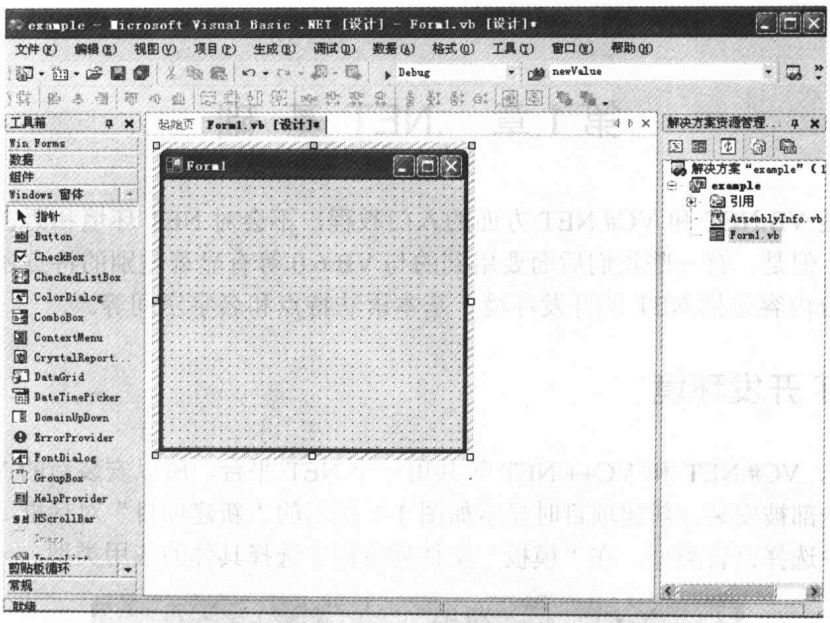


图 1-2　VB.NET 的界面设计窗口

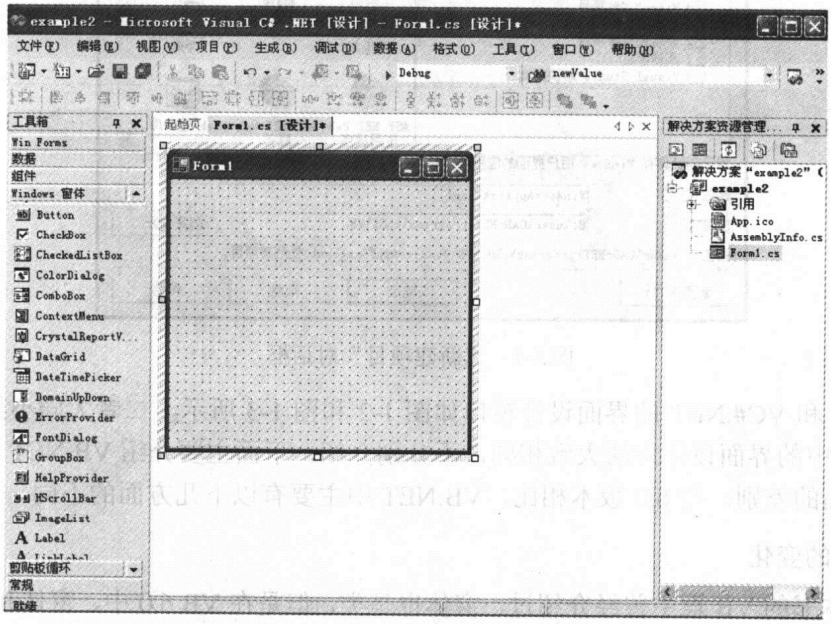


图 1-3　VC#.NET 的界面设计窗口

### 2. 菜单制作方法的变化

VB 6.0 中使用对话框进行菜单制作，现在分别使用 MainMenu 控件和 ContextMenu 控件制作主菜单和弹出式菜单。菜单项的内容直接在窗体上输入，菜单项的属性，如名称、快捷键、单选、核选等都在属性窗口中设定。这样，菜单的制作就与其他控件的制作统一起来了。

### 3. 控件设计的变化

部分控件的功能有了一些变化。比如我们常用的图片框控件，现在也能像图像框那样 Stretch 了。以前为了得到图片框的外框和图像框的 Stretch，我是将它们组合起来用的，现在不必了，设置 SizeMode 属性就行了。在 VB.NET 中，Timer, ImageList 这样一些不在窗体上显示的控件全部被放到窗体下面的一个空白面板中去了。这样，窗体上要干净一些，对这类控件的管理也方便一些。另外还有一些其他的变化，如后面要讲到的工具栏和状态栏等。详细内容需要参考其他专门介绍语言的书。

管理工具方面也有一些变化。如 VB 6.0 中管理窗体、类和公共模块等的窗口是工程窗口，现在用的是解决方案管理器。以前的工程现在称为项目。解决方案是一个比项目更大的概念，它可以包含多个项目。实际上，解决方案和项目是两个容器，它们对创建的类、引用、数据连接和文件等进行管理。

VB 6.0 中有一个对象浏览器，对 VB 中的所有对象进行显示。.NET 中有一个类视图窗口，如图 1-4 所示。它只显示项目中建立的类、接口和模块的信息。

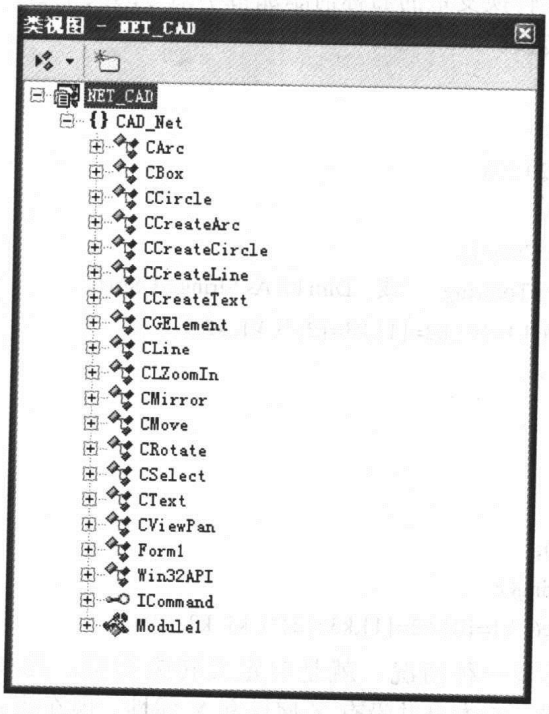


图 1-4　类视图窗口

## 1.2　基本语法

### 1.2.1　数据类型与转换

.NET 中，通用数据类型为 Object 型，VB 6.0 中的 Vairant 类型已经取消了。对象类型赋值时不再需要 Set 关键字。例如：

【VB.NET】

```
Dim obj1 As New Object()
Dim obj2 As Object = obj1
```

【VC#.NET】

```
object obj1=new object();
object obj2 = obj1;
```

数据类型的转换有隐式转换和显式转换两种。隐式转换不需要任何转换形式，直接采用A=B 的形式，系统会自动进行转换。显式转换则通过转换函数或其他的转换形式对数据类型进行强制转换。对于数值类型的转换而言，低精度类型向高精度类型转换影响不大，但高精度类型向低精度类型转换要损失部分信息。在 VB.NET 中，两种类型的转换都可以通过隐式转换完成。但 VC#.NET 是强类型的语言，要求后一种转换必须是显式转换。在 VB.NET 中，可以通过转换函数进行显式转换。如将数值转换为单精度型时使用 CSng 函数，转换为双精度型时使用 CDbl 函数，转换为字符型时使用 CStr 函数等。在 VC#.NET 中则需要进行强类型转换，转换时需要在要转换变量的名称前添加带小括号的类型名。下面的代码演示了数据类型的隐式转换和显式转换。

【VB.NET】

```
Dim i As Integer = 10
Dim j As Double = 20.756
Dim k1 As Single = i
Dim k2 As Single = CSng(j)
Dim k3 As String = j.ToString    '或  Dim k3 As String =CStr(j)
Console.WriteLine(("k1={0},k2={1},k3={2}"), k1, k2, k3)
```

【VC#.NET】

```
int i = 10;
double j = 20.756;
float k1 = i;
float k2 = (float)(j);
string k3 = j.ToString();
Console.WriteLine(("k1={0},k2={1},k3={2}"), k1, k2, k3);
```

进行显式转换时还有一种情况，就是自定义转换类型。比如下面的代码中，p2 是Object(object)类型的变量，它本身并没有 X 属性和 Y 属性。现在将 p2 指定为 p1 的引用，因为 p1 是 Point 类型的数据，有 X 属性和 Y 属性，所以现在 p2 也有这两个属性。但是调用时必须对它进行强制类型转换，并且转换为 Point 型。VB.NET 中使用 CType 函数进行自定义类型转换，VC#.NET 中则进行强类型转换。

【VB.NET】

```
Dim p1 As New Point(10, 10)
Dim p2 As Object = p1
Console.WriteLine("x={0},y={1}", CType(p2, Point).X, CType(p2, Point).Y)
```

【VC#.NET】

```
Point p1=new Point(10, 10);
object p2= new object();
p2 = p1;
Console.WriteLine("x={0},y={1}", ((Point)(p2)).X,  ((Point)(p2)).Y);
```

### 1.2.2 变量

这里主要讲两个问题，一个是变量的声明，另一个是变量的作用范围。VB.NET 与 VB 6.0 不同，可以采用下面的形式声明变量：

```
Dim a, b, c As Single, d As Double
```

即同一类型的变量可以用逗号间隔，连续输入变量名。不同类型的变量也可以在一行中进行声明。上面的变量在 VC#.NET 中做如下声明：

```
float a, b, c;
double d;
```

在 VB.NET 和 VC#.NET 中，变量的作用范围可以局限于块，如 For 循环块或者 If 块等。在块外使用该变量时，需要重新进行声明。如下面的代码段中，在 For 循环中声明了一个单精度型的 xx 变量，利用它进行求和计算。该变量只在 For 循环内部有效，在外部使用它时会报错，认为它是一个没有声明的变量。

【VB.NET】

```
Dim sum As Single, i As Integer
For i = 0 To 9
    Dim xx As Single=10+i
    sum += xx
Next
sum += xx
```

【VC#.NET】

```
float sum;
for (int i = 0;i<=9;i++)
{
    float xx=10+i;
    sum += xx;
}
sum+=xx;
```

### 1.2.3 数组

.NET 中，数组的索引值是从 0 开始的，而且必须从 0 开始。因为基数固定，所以声明数组时可以直接像下面这样声明：

【VB.NET】

```
Dim Value(9) As Single
```

【VC#.NET】

```
float[] Value;
```

使用 New(new)关键字创建数组，如下所示：

【VB.NET】

```
Dim Value() As Single = New Single()
```

【VC#.NET】

```
float[] Value=new float[3]
```

在创建数组的同时，可以对数组进行初始化。如下所示，将数组元素的初始化值用大括号括起来，然后直接放在创建数组的代码后面就行了。对数组进行初始化是一个好的编程习惯。

【VB.NET】

```
Dim Value() As Single = New Single() {2, 4, 7}
```

【VC#.NET】

```
float[] Value=new float[10]{2,4,7};
```

数组还有一些高级特性，后面讲述过程的参数传递时将会讲到，过程函数可以传回一个数组。

### 1.2.4 过程

.NET 中的过程与某些语言中的过程的一个很明显的不同就在于，缺省时它的参数是按值传递的，而其他语言如 VB 6.0 是按地址进行传递的。VB.NET 和 VC#.NET 中的过程一般具有下面的形式。注意，现在函数使用 Return(return)语句返回值。

【VB.NET】

```
Private Sub Proc(ByVal x As Single, ByVal y As Single)
    '功能代码　比如　y=x*x
End Sub
Private Function Func(ByVal x As Single, ByVal y As Single) As Single
    Dim aVal As Single
    '功能代码
    Return aVal
End Function
```

【VC#.NET】

```
private void Proc(float x , float y)
{
    //功能代码　比如　y=x*x;
}
```