عنوان: **پیاده سازی پروژه نقاشی (Paint) به صورت شی گرا #5**
نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۱/۱۲/۰۳ ۰:۵۰
آدرس: www.dotnettips.info
برچسب‌ها: C#, OOP, Project, Paint

در ادامه مطلب پیاده سازی پروژه نقاشی (Paint) به صورت شی گرا #4 به تشریح مابقی کلاس‌های برنامه می‌پردازیم.

در این پست به شرح کلاس Rectangle جهت رسم مستطیل و Square جهت رسم مربع می‌پردازیم

```
using System.Drawing;

namespace PWS.ObjectOrientedPaint.Models
{
    /// <summary>
    /// Rectangle
    /// </summary>
    public class Rectangle : Shape
    {
        #region Constructors (2)

        /// <summary>
        /// Initializes a new instance of the <see cref="Rectangle" /> class.
        /// </summary>
        /// <param name="startPoint">The start point.</param>
        /// <param name="endPoint">The end point.</param>
        /// <param name="zIndex">Index of the z.</param>
        /// <param name="foreColor">Color of the fore.</param>
        /// <param name="thickness">The thickness.</param>
        /// <param name="isFill">if set to <c>true</c> [is fill].</param>
        /// <param name="backgroundColor">Color of the background.</param>
        public Rectangle(PointF startPoint, PointF endPoint, int zIndex, Color foreColor, byte
thickness, bool isFill, Color backgroundColor)
            : base(startPoint, endPoint, zIndex, foreColor, thickness, isFill, backgroundColor)
        {
            ShapeType = ShapeType.Rectangle;
        }

        /// <summary>
        /// Initializes a new instance of the <see cref="Rectangle" /> class.
        /// </summary>
        public Rectangle()
        {
            ShapeType = ShapeType.Rectangle;
        }

        #endregion Constructors

        #region Methods (1)

        // Public Methods (1)

        /// <summary>
        /// Draws the specified g.
        /// </summary>
        /// <param name="g">The g.</param>
        public override void Draw(Graphics g)
        {
            if (IsFill)
                g.FillRectangle(BackgroundBrush, StartPoint.X, StartPoint.Y, Width, Height);
            g.DrawRectangle(Pen, StartPoint.X, StartPoint.Y, Width, Height);
            base.Draw(g);
        }

        #endregion Methods
    }
}
```

کلاس Rectangle از کلاس پایه طراحی شده در ^ ارث بری دارد. این کلاس ساده بوده و تنها شامل یک سازنده و متد ترسیم شی مستطیل می‌باشد.

کلاس بعدی کلاس Square می‌باشد، که از کلاس بالا (Rectangle) ارث بری داشته است، کدهای این کلاس را در زیر مشاهده می‌کنید.

```
using System;
using System.Drawing;

namespace PWS.ObjectOrientedPaint.Models
{
    /// <summary>
    /// Square
    /// </summary>
    public class Square : Rectangle
    {
#region Constructors (2)

        /// <summary>
        /// Initializes a new instance of the <see cref="Square" /> class.
        /// </summary>
        /// <param name="startPoint">The start point.</param>
        /// <param name="endPoint">The end point.</param>
        /// <param name="zIndex">Index of the z.</param>
        /// <param name="foreColor">Color of the fore.</param>
        /// <param name="thickness">The thickness.</param>
        /// <param name="isFill">if set to <c>true</c> [is fill].</param>
        /// <param name="backgroundColor">Color of the background.</param>
        public Square(PointF startPoint, PointF endPoint, int zIndex, Color foreColor, byte thickness,
bool isFill, Color backgroundColor)
        {
            float x = 0, y = 0;
            float width = Math.Abs(endPoint.X - startPoint.X);
            float height = Math.Abs(endPoint.Y - startPoint.Y);
            if (startPoint.X <= endPoint.X && startPoint.Y <= endPoint.Y)
            {
                x = startPoint.X;
                y = startPoint.Y;
            }
            else if (startPoint.X >= endPoint.X && startPoint.Y >= endPoint.Y)
            {
                x = endPoint.X;
                y = endPoint.Y;
            }
            else if (startPoint.X >= endPoint.X && startPoint.Y <= endPoint.Y)
            {
                x = endPoint.X;
                y = startPoint.Y;
            }
            else if (startPoint.X <= endPoint.X && startPoint.Y >= endPoint.Y)
            {
                x = startPoint.X;
                y = endPoint.Y;
            }
            StartPoint = new PointF(x, y);
            var side = Math.Max(width, height);
            EndPoint = new PointF(x+side, y+side);
            ShapeType = ShapeType.Square;
            Zindex = zIndex;
            ForeColor = foreColor;
            Thickness = thickness;
            BackgroundColor = backgroundColor;
            IsFill = isFill;
        }

        /// <summary>
        /// Initializes a new instance of the <see cref="Square" /> class.
        /// </summary>
        public Square()
        {
            ShapeType = ShapeType.Square;
        }

#endregion Constructors
    }
}
```

این کلاس شامل دو سازنده می‌باشد که سازنده دوم فقط نوع شی را تعیین می‌کند و بقیه کارهای آن مانند مستطیل است، در واقع می‌توان از یک دیدگاه گفت که مربع یک مستطیل است که اندازه طول و عرض آن یکسان است. در سازنده اول (نحوه ترسیم

شکل) ابتدا نقاط ابتدا و انتهای رسم شکل تعیین شده و سپس با توجه به پارامترهای محاسبه شده نوع شی جهت ترسیم و دیگر خصوصیات کلاس مقدار دهی می‌شود، با این تفاوت که در نقطه EndPoint طول و عرض مربع برابر با بزرگترین مقدار طول و عرض وارد شده در سازنده کلاس تعیین شده و مربع شکل می‌گیرد. مابقی متدهای ترسیم و ... طبق کلاس پایه مستطیل و Shape تعیین می‌شود.

مطالب قبل:

پیاده سازی پروژه نقاشی (Paint) به صورت شی گرا #1

پیاده سازی پروژه نقاشی (Paint) به صورت شی گرا #2

پیاده سازی پروژه نقاشی (Paint) به صورت شی گرا #3

پیاده سازی پروژه نقاشی (Paint) به صورت شی گرا #4

نظرات خوانندگان

نویسنده: کاوه احمدی
تاریخ: ۱۳۹۱/۱۲/۰۳ ۱۰:۵۸

امروز فرصتی دست داد نگاهی اجمالی به این پروژه بیندازم. به نظرم کد نوشته شده تا به اینجا شی گرا محسوب نمی‌شود. یعنی برخی اهدافی که به واسطه آن پارادایم شی گرایی شکل گرفته در آن رعایت نشده است.
به طور مشخص منظورم متد DrawPreview است که در بخش سوم در کلاس Helpers نوشته شده. تکرار کد شدیدی که در دستور switch این متد دیده می‌شود به سادگی قابل حذف است. کد فوق 2 مشکل اساسی دارد: اول آنکه با زیاد شدن تعداد اشیای قابل رسم، این دستور switch بسیار طولانی شده (با تکرار کد) و کد ناخوانا می‌شود و دوم آنکه با اضافه شدن هر شی قابل رسم جدید به پروژه یک case باید به این دستور اضافه شود. یعنی تغییر در یک بخش از نرم‌افزار منجر به تغییر در سایر بخش‌ها (کلاس Helpers) می‌شود. بدیهی است پارادایم شی‌گرا برای جلوگیری از چنین مسائلی شکل گرفته. در غیر این صورت این کد همان کدهای ساخت‌یافته است که در قالب کلاس نوشته شده. به نظر می‌آید بهتر باشد یک اینترفیس drawable در نظر گرفته می‌شد، در این متد از آن استفاده می‌شد و اشیای قابل رسم آنرا پیاده‌سازی می‌کردند. یک راه بسیار ساده و کارامد

نویسنده: محسن
تاریخ: ۱۳۹۱/۱۲/۰۳ ۱۱:۲۵

البته میشه این قسمت کلاس پایه رو که if و else و switch زیاد داره، با توجه به مطلب « کمپین ضد IF ! » بهبود بخشید.

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۱/۱۲/۰۳ ۲۳:۴

پاسخ شما کاملا صحیح است، توی همون پست گفتم که خیلی خوشحال میشم دوستان ایده ای به من بدهند که این قسمت با استفاده از Action و Func طراحی کنم، خودم راهی به ذهنم نرسید. بله کاملا شما درست می‌فرمایید، راه حل چیست؟

پ. ن: البته این متد کاملا قابل حذف است و می‌تواند در سیستم استفاده نشود. فقط جهت پیش نمایش رسم اشیا بکار می‌رود.

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۱/۱۲/۰۵ ۱۲:۵

@کاوه احمدی
من خیلی سعی کردم طبق الگوی « کمپین ضد IF ! » عمل کردم، و پیش رفتم درست شد، اما به دلیل اینکه در زمان رسم شی در برنامه کاربری (اینترفیس) در زمان MouseMove پیش نمایش شی رسم می‌شود و در زمان MouseUp خود شی رسم می‌شود این امکان نداشتم تا از شی نمونه سازی کنم و طبق اون الگو پیش برم ، لطفا در صورتی که روشم اشتباست اصلاح بفرمایین.
موفق وموید باشید.

نویسنده: محسن
تاریخ: ۱۳۹۱/۱۲/۰۶ ۱۶:۵۵

سلام

می‌تونید از ابزارهای تزریق وابستگی برای تامین وهله مورد نیاز استفاده کنید.