عنوان: **پیاده سازی پروژه نقاشی (Paint) به صورت شی گرا #1**
نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۱/۱۱/۰۷ ۹:۴۰
آدرس: www.dotnettips.info
برچسب‌ها: OOP, C#.NET, Project

قصد داریم در طی چند پست متوالی، یک پروژه Paint را به صورت شی گرا پیاده سازی کنیم. خوب، پروژه ای که می‌خواهیم پیاده سازی کنیم باید دارای این امکانات باشه که مرحله به مرحله پیش میریم و پروژه کامل در نهایت در قسمت پروژه‌ها ی همین سایت قرار خواهد گرفت.

قابلیت ترسیم اشیا روی بوم گرافیکی دلخواه
قابلیت جابجایی اشیا
قابلیت تغییر رنگ اشیا
ترسیم اشیا توپر و تو خالی
تعیین پهنای خط شی ترسیم شده
تعیین رنگ پس زمینه در صورت تو پر بودن شی
قابلیت پیش نمایش رسم شکل در زمان ترسیم اشیا
توانایی انتخاب اشیا
تعیین عمق شی روی بوم گرافیکی مورد نظر
ترسیم اشیایی مانند خط، دایره، بیضی، مربع، مستطیل، لوزی، مثلث
قابلیت تغییر اندازه اشیا ترسیم شده

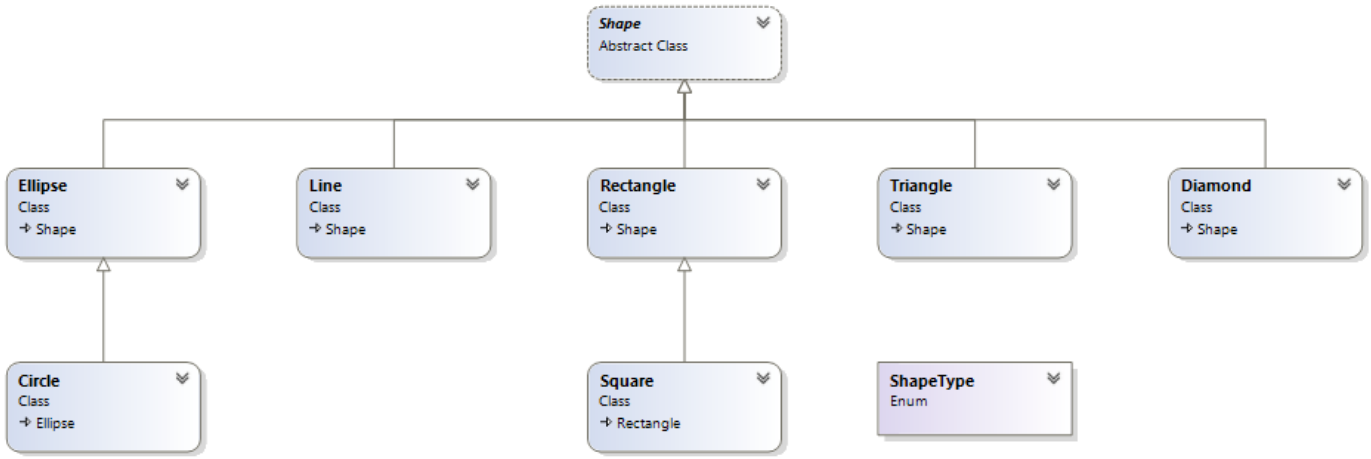
خوب برای شروع ابتدا یک پروژه از نوع Windows Application ایجاد می‌کنیم (البته برای این قسمت می‌توانیم یک پروژه Class Library ایجاد کنیم)

سپس یک پوشه به نام Models به پروژه اضافه نمایید.

خوب در این پروژه یک کلاس پایه به نام Shape در نظر می‌گیریم.

همه اشیا ما دارای نقطه شروع، نقطه پایان، رنگ قلم، حالت انتخاب، رنگ پس زمینه، نوع شی، .... می‌باشند که بعضی از خصوصیات را توسط خصوصیات دیگر محاسبه می‌کنیم. مثلا خاصیت Width و Height و X و Y توسط خصوصیات نقطه شروع و پایان می‌توانند محاسبه شوند.

ساختار کلاس‌های پروژه ما به صورت زیر است که مرحله به مرحله کلاس‌ها پیاده سازی خواهند شد.



با توجه به تصویر بالا (البته این تجزیه تحلیل شخصی من بوده و دوستان به سلیقه خود ممکن است این ساختار را تغییر دهند)

نوع شمارشی ShapeType: در این نوع شمارشی انواع شی‌های موجود در پروژه معرفی شده است

محتوای این نوع به صورت زیر تعریف شده است:

```
namespace PWS.ObjectOrientedPaint.Models
{
    /// <summary>
    /// Shape Type in Paint
    /// </summary>
    public enum ShapeType
    {
        /// <summary>
        /// هیچ
        /// </summary>
        None = 0,
        /// <summary>
        /// خط
        /// </summary>
        Line = 1,
        /// <summary>
        /// مربع
        /// </summary>
        Square = 2,
        /// <summary>
        /// مستطیل
        /// </summary>
        Rectangle = 3,
        /// <summary>
        /// بیضی
        /// </summary>
        Ellipse = 4,
        /// <summary>
        /// دایره
        /// </summary>
        Circle = 5,
        /// <summary>
        /// لوزی
        /// </summary>
        Diamond = 6,
        /// <summary>
        /// مثلث
        /// </summary>
        Triangle = 7,
    }
}
```

انشاالله در پست‌های بعدی ما بقی کلاس‌ها به مرور پیاده سازی خواهند شد.

نظرات خوانندگان

نویسنده: علی صداقت
تاریخ: ۱۳۹۱/۱۱/۰۷ ۱۷:۱۷

مبحث مفید و جالبیست. منتظر ادامه این مبحث هستیم. موفق باشید.

نویسنده: Parham
تاریخ: ۱۳۹۱/۱۱/۰۸ ۱۰:۳۷

نوع شمارشی ShapeType یک فایل کد ساده است، درسته؟

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۱/۱۱/۱۲ ۳:۲

بله درسته

نویسنده: masoud
تاریخ: ۱۳۹۱/۱۱/۱۴ ۱۴:۶

با تشکر اقای فتح الهی ممنون میشم این بحث رو تا آخر پیش ببرید استفاده میکنیم.

نویسنده: مسعود بهرامی
تاریخ: ۱۳۹۲/۰۸/۳۰ ۳:۲۰

قرار دادن کلاس Square به عنوان sub type زیر کلاس Rectangle اصل LSP رو نقض می‌کنه بهتره که هر دو از Shape به ارث برسن با بهتره بگم Implement کنن

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۲/۰۸/۳۰ ۱۰:۴۰

بله کاملا درسته از نظر شی گرای یک مربع نمیتونه مستطیل باشه، این پروژه هنوز تمام نشده و در خروجی نهایی اصلاح خواهد شد الان قابلیت ویرایش وجود نداره.
البته پست بعدی پیاده سازی پروژه نقاشی (Paint) به صورت شی گرا #2 نگاه کنین شاید نظراون عوض شد.
موفق و موید باشید

نویسنده: خوزستان
تاریخ: ۱۳۹۲/۰۸/۳۰ ۱۱:۳۰

اصل SPR , LSP ,.... از کجا و چه منابعی برای اینا موجود هست ؟
آیا کتابهای مهندسی نرم افزار باید خوند یا کلن مبحث جداگانه‌ای داره ؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۸/۳۰ ۱۲:۷

از اینجا شروع کنید. 5 قسمت هست:

اصول طراحی شی گرا SOLID - #بخش اول اصل SRP

نویسنده: خوزستان
تاریخ: ۱۳۹۲/۰۸/۳۰ ۱۳:۱

تشکر .
آیا کتاب فارسی در این باره وجود دارد ؟
اساتید کتابی هست که معرفی کنن ؟

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۲/۰۸/۳۰ ۱۳:۵۹

توی گوگل SOLID در شی گرا جستجو کنین

یا توی همین سایت این برچسب دنبال کنید

نویسنده: مسعود بهرامی
تاریخ: ۱۳۹۲/۰۹/۰۱ ۳:۵۷

سلام مهندس فتح الهی عزیز من خوندم کامل ولی به نظر من بهتره از هم جدا شدن تا اصل LSP رو نقض نکنن درسته رفتار ترسیمشون مثل همه و دیگه تو کلاس مربع شما تابع رسم رو ننوشتین ولی باعث نقض LSPشدین به نظر من بهتر هردوشون از یه Abstract Class دیگه یه ارث برسن و تابع Draw این دوشکل که مثل هم هستش رو بزارین اونجا با اینکار هم LSP رعایت شده و هم تکرا تابع Draw رو هم ندارین
در ضمن تابع رسم پیش نمایش اشکالات اساسی از نظر OOP داشت که من با اجازتون اونو زیر پست خودش میزارم Re factor که به نظرم بهتر میومد. دست گلتونم درد نکنه

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۲/۰۹/۰۱ ۱۲:۸

خوشحال میشم کد شمارو ببینم
منتظر روش شما هستم من که خودم چیزی به ذهنم نرسید