**گزارش ChildAnimation**

**هدف از ایجاد کنترل:**

این کنترل برای نمایش ورودی به صورت انیمیشن ایجاد شد ، این کنترل خصوصیات زیر را نیز دارد:

- قابلیت اجرا روی .NET Framework 3.5 دارد.

- از ویندوز XP به بالا را پشتیبانی می کند.

- برای ترسیم و متحرک سازی از GPU (پردازنده ی کارت گرافیک) استفاده می کند.

این کنترل بعد از تشخیص ویندوز و همچنین کارت گرافیکی مورد نظر ابتدا Graphic engine ای را که مناسب از نظر سرعت و کیفیت می باشد را انتخاب کرده و فایل Dll مربوط به آن را بارگذاری می کند.

**نحوه کارکرد:**

نحوه کارکرد این کنترل به این شکل است که بعد از تعریف و پیاده سازی کنترل در ویندوز فرم با فراخوانی تابع NewSample(value) مقادیر و تصاویر با توجه به مقدار ورودی تنظیم شده و عملیات ترسیم انجام می شود. علاوه بر این، این کنترل تابعی به نام Reset() نیز دارد که فراخوانی این تابع باعث می شود که مقادیر کنترل با مقادیر پیش فرض مقدار دهی شود.

قابل توجه است که درصورتیکه تغییر در مقادیر ورودی باعث تغییر در محل قرار گرفتن المانهای انیمیشن شود عملیات ترسیم انجام می گردد.

این کنترل شامل کلاسهای زیر می باشد که در ادامه به توضیح هر یک می پردازیم.

**کلاس AnimationObject :**

این کلاس، کلاس پایه در این طراحی می باشد. تعریف کلی این کلاس یک شی متحرک می باشد. متغییرهای این کلاس عبارتند از :

texture : که تصاویر شی در این متغییر نگه داری می شود. این متغییر آرایه ای از تصاویر است.

StartPoint : نقطه شروع حرکت شی

EndPoint : نقطه پایان حرکت شی

CurrentPoint : موقعیت جاری شی در کنترل

Device : محیطی که رسم شی در آن استفاده می شود.

ImageSize : سایز تصویر شی.

RotateAngle : این متغییر مقداری که این شی دوران خواهد یافت را در خود ذخیره می کند.

RotateOrigin: محل نسبی قرار گرفتن محور دوران را مشخص می کند.

VerticalMirror : این متغییر مشخص می کند که آیا شی نسبت به محور عمودی دوران کرده است یا نه.

و همچنین شامل متدهای زیر می باشد:

Draw() : تصویر شی را در محل CurrentPoint رسم می کند.

Dispose() : وظیفه آزاد سازی منابع اشغال شده بر عهده ی این متد می باشد.

**کلاس AppleAndBirdScenario :**

این کلاس پیاده سازی یک سناریو می باشد. در این سناریو با توجه به مقدار ورودی کنترل سیب توسط پسر بچه به هوا پرتاب می شود. در صورتی که به مقداری مشخص رسید پرنده هایی که روی درخت نشسته اند به سمت زمین پرواز کرده و کنار پسربچه می نشینند.

در این کلاس هر یک از اشیائی که نقشی در این سناریو دارند در تابع Initialize() تعریف شده و منتظر فراخوانی تابع NewSample() می مانند. این تابع ابتدا تابع DetectCollision() جهت تشخیص تصادم و سپس تابع UpdatePosition() برای به روز رسانی محل قرار گرفتن اشیا و وضعیت و ... اشیا را اجرا می کند. بعد از آماده شدن اجرای تابع UpdateScene() باعث می شود که سکانس به روز رسانی شود.

**کلاس BallonAndBirdScenario :**

این کلاس پیاده سازی یک سناریو می باشد. در این سناریو با توجه به مقدار ورودی کنترل بالون به سمت بالا پرواز می کند. در صورتی که به مقداری مشخص رسید پرنده هایی که روی سیم تیر نشسته اند به سمت زمین پرواز می کنند.

در این کلاس هر یک از اشیائی که نقشی در این سناریو دارند در تابع Initialize() تعریف شده و منتظر فراخوانی تابع NewSample() می مانند. این تابع ابتدا تابع DetectCollision() جهت تشخیص تصادم و سپس تابع UpdatePosition() برای به روز رسانی محل قرار گرفتن اشیا و وضعیت و ... اشیا را اجرا می کند. بعد از آماده شدن اجرای تابع UpdateScene() باعث می شود که سکانس به روز رسانی شود.

در هر دو کلاس BallonAndBirdScenario و AppleAndBirdScenario تابع Dispose() پیاده سازی شده است. که وظیفه آزادسازی منابع استفاده شده در کنترل را به عهده دارد.

**لایه بندی**

IDrawing Interface

Child Animation Control

Graphic Engine D3D9

Graphic Engine D2D1

Graphic Engine MDX

Directx

GPU

کلاسی به اسم GraphicEngine وظیفه تجزیه و تحلیل بهترین گزینه از نظر سرعت و کیفیت با توجه به مشخصات کارت گرافیکی و نوع ویندوز را بر عهده دارد.

درصورتی که نیاز به اضافه نمودن روشی جدید برای ترسیم خط و تصاویر و متن بر روی کنترل بود کافیست اولا شرایط انتخاب در کلاس GraphicEngine درج شود .

دوما کلاسی به روش جدید IDrawing Interface ایجاده شده و پیاده سازی شود.