附加属性：某个属性本来不属于某个对象，但由于某种需求而被后来附加上。也就是把对象放入一个特定环境后对象才具有的属性（表现出来就是被环境赋予的属性）。

附加属性在本质是依赖项属性，只是注册和包装器有所不同。

附加属性的应用：

当我们定义一个人的类Human，人在学校School中具有Grade属性，而在公司Company则可能具有Project属性。我们不可能在Human中既定义Grade属性又定义Project属性。这时可以利用附加属性。

例：程序attached\_property\_test1

Human.cs

// Copyright 2016.刘珅珅

// author：刘珅珅

// 附加属性

using *System*;

using *System*.*Collections*.*Generic*;

using *System*.*Linq*;

using *System*.*Text*;

using *System*.*Threading*.*Tasks*;

using *System*.*Windows*;

namespace attached\_property\_test1

{

class Human:*DependencyObject*

{

}

// 学校

class School:*DependencyObject*

{

// 附加属性的包装器

public static int GetGrade(*DependencyObject* obj)

{

return (int)obj.*GetValue*(GradeProperty);

}

public static void SetGrade(*DependencyObject* obj, int value)

{

obj.*SetValue*(GradeProperty, value);

}

// 附加属性

public static readonly *DependencyProperty* GradeProperty =

*DependencyProperty*.*RegisterAttached*("Grade", typeof(int),

typeof(School), new *UIPropertyMetadata*(0));

}

// 公司

class Company : *DependencyObject*

{

// 附加属性的包装器

public static int GetProject(*DependencyObject* obj)

{

return (int)obj.*GetValue*(ProjectProperty);

}

public static void SetProject(*DependencyObject* obj, int value)

{

obj.*SetValue*(ProjectProperty, value);

}

// 附加属性

public static readonly *DependencyProperty* ProjectProperty =

*DependencyProperty*.*RegisterAttached*("Project", typeof(int),

typeof(Company), new *UIPropertyMetadata*(0));

}

}

MainWindow.xaml

<Window x:Class="attached\_property\_test1.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

Title="MainWindow" Height="350" Width="525">

<StackPanel>

<Button x:Name="button1" Click="School\_Click" Margin="5" Width="50" Height="30" Content="School"/>

<Button x:Name="button2" Click="Company\_Click" Margin="5" Width="70" Height="30" Content="Company" />

</StackPanel>

</Window>

MainWindows.xaml.cs

using *System*;

using *System*.*Collections*.*Generic*;

using *System*.*Linq*;

using *System*.*Text*;

using *System*.*Threading*.*Tasks*;

using *System*.*Windows*;

using *System*.*Windows*.*Controls*;

using *System*.*Windows*.*Data*;

using *System*.*Windows*.*Documents*;

using *System*.*Windows*.*Input*;

using *System*.*Windows*.*Media*;

using *System*.*Windows*.*Media*.*Imaging*;

using *System*.*Windows*.*Navigation*;

using *System*.*Windows*.*Shapes*;

namespace attached\_property\_test1

{

/// <summary>

/// MainWindow.xaml 的交互逻辑

/// </summary>

public partial class MainWindow : *Window*

{

public MainWindow()

{

*InitializeComponent*();

}

private void School\_Click(object sender, *RoutedEventArgs* e)

{

Human human = new Human();

// 在School附加了Grade属性

School.SetGrade(human, 6);

int grade = School.GetGrade(human);

*MessageBox*.*Show*(grade.*ToString*());

}

private void Company\_Click(object sender, *RoutedEventArgs* e)

{

// 在Company中附加了Project属性

Human human = new Human();

Company.SetProject(human, 5);

int project = Company.GetProject(human);

*MessageBox*.*Show*(project.*ToString*());

}

}

}