https://docs.microsoft.com/zh-cn/dotnet/framework/wpf/graphics-multimedia/animation-overview

**Animation Overview**

## Introducing Animations

Animation is an illusion that is created by quickly cycling through a series of images, each slightly different from the last.

Animation on a computer is similar. For example, a program that makes a drawing of a rectangle fade out of view might work as follows.

 The program creates a timer.

 The program checks the timer at set intervals to see how much time has elapsed.

 Each time the program checks the timer, it computes the current opacity value for the rectangle based on how much time has elapsed.

 The program then updates the rectangle with the new value and redraws it.

Prior to WPF, Microsoft Windows developers had to create and manage their own timing systems or use special custom libraries. WPF includes an efficient timing system that is deeply integrated into the WPF framework

WPF handles all the behind-the-scenes work of managing a timing system and redrawing the screen efficiently. It provides timing classes that enable you to focus on the effects you want to create, instead of the mechanics of achieving those effects

## WPF Property Animation System

in WPF, you animate objects by applying animation to their individual properties.

For a property to have animation capabilities, it must meet the following three requirements:

 It must be a dependency property.

 It must belong to a class that inherits from [DependencyObject](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.dependencyobject) and implements the [IAnimatable](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.ianimatable) interface.

 There must be a compatible animation type available. (If WPF does not provide one, you can create your own. See the [Custom Animations Overview](https://docs.microsoft.com/zh-cn/dotnet/framework/wpf/graphics-multimedia/custom-animations-overview).)

## Example: Make an Element Fade In and Out of View

1. create a [Rectangle](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.shapes.rectangle) element
2. Create a DoubleAnimation

Because the [Opacity](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement.opacity) property is of type [Double](https://docs.microsoft.com/zh-cn/dotnet/api/system.double), you need an animation that produces double values. A [DoubleAnimation](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.doubleanimation) is one such animation. A [DoubleAnimation](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.doubleanimation) creates a transition between two double values

1. Create a Storyboard

To apply an animation to an object, you create a [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard) and use the [TargetName](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard.targetname) and [TargetProperty](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard.targetproperty) attached properties to specify the object and property to animate

<Storyboard>

<DoubleAnimation

From="1.0" To="0.0" Duration="0:0:1"

AutoReverse="True" RepeatBehavior="Forever" />

</Storyboard>

The [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard) has to know where to apply the animation. Use the [Storyboard.TargetName](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard.targetname) attached property to specify the object to animate.

1. Associate the Storyboard with a Trigger
   1. The easiest way to apply and start a [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard) in XAML is to use an event trigger.
   2. Create a [BeginStoryboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.beginstoryboard) object and associate your storyboard with it. A [BeginStoryboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.beginstoryboard) is a type of [TriggerAction](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.triggeraction) that applies and starts a [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard).
   3. Create an [EventTrigger](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.eventtrigger) and add the [BeginStoryboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.beginstoryboard) to its [Actions](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.eventtrigger.actions) collection
   4. Add the [EventTrigger](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.eventtrigger) to the [Triggers](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement.triggers) collection of the Rectangle.

4 (Code): Associate the Storyboard with an Event Handler

myRectangle.Loaded += new RoutedEventHandler(myRectangleLoaded);

private void myRectangleLoaded(object sender, RoutedEventArgs e)

{

myStoryboard.Begin(this);

}

## Animation Types

Because animations generate property values, different animation types exist for different property types

they follow a strict naming convention that makes it easy to differentiate between them:

* <*Type*>Animation

Known as a "From/To/By" or "basic" animation, these animate between a starting and destination value, or by adding an offset value to its starting value

* <Type>AnimationUsingKeyFrames

Key frame animations are more powerful than From/To/By animations because you can specify any number of target values and even control their interpolation method.

<Type>AnimationUsingPath

Path animations enable you to use a geometric path in order to produce animated values.

### Animations Are Timelines

an animation is a [Timeline](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.timeline), it also represents a segment of time. An animation also calculates output values as it progresses through its specified segment of time (or [Duration](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.timeline.duration)).

Three frequently used timing properties are [Duration](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.timeline.duration), [AutoReverse](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.timeline.autoreverse), and [RepeatBehavior](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.timeline.repeatbehavior).

### Applying and Starting Storyboards

To start a [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard) in code, you can use an [EventTrigger](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.eventtrigger) or use the [Begin](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard.begin) method of the [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard) class.

# Animation and Timing System Overview

## Timelines and Clocks

a [Timeline](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.timeline), doesn't do anything other than just describe a segment of time. It's the timeline's [Clock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.clock) object that does the real work. an animation is a type of [Timeline](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.timeline) that produces output values.

an animation class describes how output values should be calculated, but it’s the [Clock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.clock) that was created for the animation that drives the animation output and applies it to properties.

When you animate by using a [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard) or the [BeginAnimation](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.animatable.beginanimation) method, clocks are automatically created for your timelines and animations

You can also create a [Clock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.clock) explicitly by using the [CreateClock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.timeline.createclock) method of your [Timeline](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.timeline).

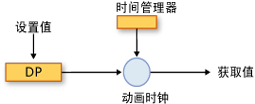
If the [Timeline](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.timeline) contains child timelines, it creates [Clock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.clock) objects for them as well. The resulting [Clock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.clock) objects are arranged in trees that match the structure of the [Timeline](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.timeline) objects tree from which they are created.

## Clocks and the Time Manager

it’s the time manager that manages the [Clock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.mediaplayer.clock) objects created for your timelines.. A time manager is automatically created for each WPF application and is invisible to the application developer. The time manager is the root of a tree of [Clock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.mediaplayer.clock) objects and controls the flow of time in that tree

The time manager "ticks" many times per second;

The following illustration shows the relationship between the time manager, and [AnimationClock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.animationclock), and an animated dependency property.



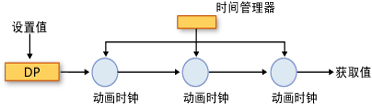
When the time manager ticks, it updates the time of every [Active](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.clockstate#System_Windows_Media_Animation_ClockState_Active) [Clock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.clock) in the application. If the [Clock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.clock) is an [AnimationClock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.animationclock), it uses the [GetCurrentValue](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.animationtimeline.getcurrentvalue) method of the [AnimationTimeline](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.animationtimeline) from which it was created to calculate its current output value

#### Clock Groups

A [ClockGroup](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.clockgroup) is created for timelines that group other timelines, such as the [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard) class, which groups animations and other timelines.

#### Composition

It's possible to associate multiple clocks with a single property, in which case each clock uses the output value of the preceding clock as its base value.



## Current Values and Base Values of Properties

An animatable property can have two values: a base value and a current value. When you set property using its CLR accessor or the [SetValue](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.dependencyobject.setvalue) method, you set its base value. When a property is not animated, its base and current values are the same.

When you animate a property, the [AnimationClock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.animationclock) sets the property's current value. Retrieving the property's value through its CLR accessor or the [GetValue](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.dependencyobject.getvalue) method returns the output of the [AnimationClock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.animationclock) when the [AnimationClock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.animationclock) is [Active](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.clockstate#System_Windows_Media_Animation_ClockState_Active) or [Filling](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.clockstate#System_Windows_Media_Animation_ClockState_Filling). You can retrieve the property's base value by using the [GetAnimationBaseValue](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.ianimatable.getanimationbasevalue) method.

# From/To/By Animations Overview

A From/To/By animation is a type of [AnimationTimeline](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.animationtimeline) that creates a transition between a starting value and an ending value

## From/To/By Animation Types

use the following naming convention:

<Type> Animation

Where <Type> is the type of value that the class animates.

## Target Values

you can also specify only a starting value, a destination value, or an offset value. In these cases, the animation obtains the missing target value from the property that is being animated

**Starting Value**

If you specify only the [From](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.doubleanimation.from) property, the animation transitions from that value to the base value of the animated property

**Ending Value**

If you use the [To](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.doubleanimation.to) property by itself, the animation obtains its starting value from the property that is being animated or from the output of another animation that is applied to the same property

**Offset Value**

If you specify only the [By](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.doubleanimation.by) property, the animation adds the offset value to the base value of the property or to the output of another animation.

# Key-Frame Animations Overview

while a From/To/By animation creates a transition between two values, a single key-frame animation can create transitions among any number of target values

A key-frame animation's target values are described using key frames objects

When the animation runs, it transitions between the frames you specified.

An animation's interpolation method defines how it transitions from one value to the next. There are three types of interpolations: discrete, linear, and splined.

<DoubleAnimationUsingKeyFrames

Storyboard.TargetName="MyAnimatedTranslateTransform"

Storyboard.TargetProperty="X"

Duration="0:0:10">

<LinearDoubleKeyFrame Value="0" KeyTime="0:0:0" />

<LinearDoubleKeyFrame Value="350" KeyTime="0:0:2" />

<LinearDoubleKeyFrame Value="50" KeyTime="0:0:7" />

<LinearDoubleKeyFrame Value="200" KeyTime="0:0:8" />

</DoubleAnimationUsingKeyFrames>

## Key-Frame Animation Types

naming convention:

<Type> AnimationUsingKeyFrames

Where <Type> is the type of value that the class animates

## Target Values (key frames) and Key Times

Key frame types adhere to the following naming convention:

<InterpolationMethod><Type> KeyFrame

Where <InterpolationMethod> is the interpolation method the key frame uses and <Type> is the type of value that the class animates

Every key frame type provides these two properties.

* The [Value](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.ikeyframe.value) property specifies the target value for that key-frame.
* The [KeyTime](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.ikeyframe.keytime) property specifies when (within the animation's [Duration](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.timeline.duration)) a key frame's [Value](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.ikeyframe.value) is reached.

When a key frame animation begins, iterates through its key frames in the order defined by their [KeyTime](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.ikeyframe.keytime) properties.

If there is no key frame at time 0, the animation creates a transition between the target property's current value and the [Value](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.ikeyframe.value) of the first key frame; otherwise, the animation's output value becomes the value of the first key frame.

The animation creates a transition between the [Value](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.ikeyframe.value) of the first and second key frames using the interpolation method specified by the second key frame. The transition starts at the first key frame's [KeyTime](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.ikeyframe.keytime) and ends when the second key frame's [KeyTime](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.ikeyframe.keytime) is reached.

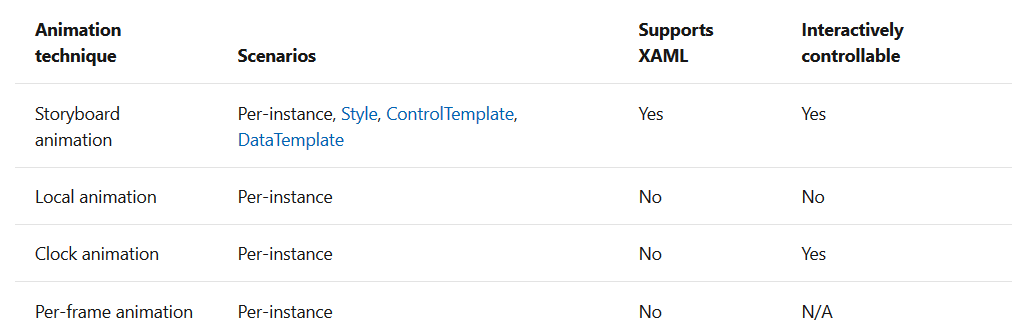
If the animation's [Duration](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.timeline.duration) is [Automatic](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.duration.automatic) or its [Duration](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.timeline.duration) is equal to the time of the last key frame, the animation ends. Otherwise, if the animation's [Duration](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.duration) is greater than the key time of the last key frame, the animation holds the key frame value until it reaches the end of its [Duration](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.duration).

# Property Animation Techniques Overview

## Different Ways to Animate

Because there are many different scenarios for animating properties, WPF provides several approaches for animating properties.

For each approach, the following table indicates whether it can be used per-instance, in styles, in control templates scenarios



## Storyboard Animations

For an object to be animated by a [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard), it must be a [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) or [FrameworkContentElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkcontentelement), or it must be used to set a [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) or [FrameworkContentElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkcontentelement). For more details

A [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard) is a special type of container [Timeline](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.timeline) that provides targeting information for the animations it contains. To animate with a [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard), you complete the following three steps.

 Declare a [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard) and one or more animations.

 Use the [TargetName](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard.targetname) and [TargetProperty](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard.targetproperty) attached properties to specify the target object and property of each animation.

 (Code only) Define a [NameScope](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.namescope) for a [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) or [FrameworkContentElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkcontentelement). Register the names of the objects to animate with that [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) or [FrameworkContentElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkcontentelement).

 Begin the [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard).

There are two ways to begin a [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard): you can use the [Begin](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard.begin) method provided by the [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard) class, or you can use a [BeginStoryboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.beginstoryboard) action

A [BeginStoryboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.beginstoryboard) action can be used in an [EventTrigger](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.eventtrigger), property [Trigger](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.trigger), or a [DataTrigger](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.datatrigger).

Local animations provide a convenient way to animate a dependency property of any [Animatable](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.animatable) object

Unlike a [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard) animation, a local animation can animate an object that isn't associated with a [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) or a [FrameworkContentElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkcontentelement). You also don't have to define a [NameScope](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.namescope) for this type of animation.

 Create an [AnimationTimeline](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.animationtimeline) object.

 Use the [BeginAnimation](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.animatable.beginanimation) method of the object that you want to animate to apply the [AnimationTimeline](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.animationtimeline) to the property that you specify.

// Apply the animation to the brush's Color property.

myBrush.BeginAnimation(SolidColorBrush.ColorProperty, myColorAnimation);

aButton.Background = myBrush;

## Clock Animations

Use [Clock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.mediaplayer.clock) objects when you want to animate without using a [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard) and you want to create complex timing trees or interactively control animations after they start

To apply a single [Clock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.clock) to a property, you complete the following steps.

1. Create an [AnimationTimeline](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.animationtimeline) object.
2. Use the [CreateClock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.animationtimeline.createclock) method of the [AnimationTimeline](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.animationtimeline) to create an [AnimationClock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.animationclock).
3. Use the [ApplyAnimationClock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.animatable.applyanimationclock) method of the object that you want to animate to apply the [AnimationClock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.animationclock) to the property you specify.

// Create a clock the for the animation.

AnimationClock myClock = myAnimation.CreateClock();

// Associate the clock the ScaleX and

// ScaleY properties of the button's

// ScaleTransform.

myScaleTransform.ApplyAnimationClock(

ScaleTransform.ScaleXProperty, myClock);

To create a timing tree and use it animate properties, you complete the following steps.

1. Use [ParallelTimeline](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.paralleltimeline) and [AnimationTimeline](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.animationtimeline) objects to create the timing tree.
2. Use the [CreateClock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.timelinegroup.createclock) of the root [ParallelTimeline](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.paralleltimeline) to create a [ClockGroup](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.clockgroup).
3. Iterate through the [Children](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.clockgroup.children) of the [ClockGroup](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.clockgroup) . For each [AnimationClock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.animationclock) child, use the [ApplyAnimationClock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.animatable.applyanimationclock) method of the object that you want to animate to apply the [AnimationClock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.animationclock) to the property you specify

# Storyboards Overview

## What Is a Storyboard?

Other timeline classes（Container timelines） are provided to help you organize sets of timelines, and to apply timelines to properties.，making it easy to organize and control complex timing behaviors 。Container timelines derive from the [TimelineGroup](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.timelinegroup) class, and include [ParallelTimeline](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.paralleltimeline) and [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard).

## Where Can You Use a Storyboard?

A [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard) can be used to animate dependency properties of animatable classes .However, because storyboarding is a framework-level feature, the object must belong to the [NameScope](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.namescope) of a [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) or a [FrameworkContentElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkcontentelement).

## Targeting Framework Elements, Framework Content Elements, and Freezables

<https://docs.microsoft.com/zh-cn/dotnet/framework/wpf/graphics-multimedia/bitmap-effects-overview>

The previous section mentioned that, for an animation to find its target, it must know the target's name and the property to animate

For the [TargetName](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.setter.targetname) property to work, the targeted object must have a name. Assigning a name to a [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) or a [FrameworkContentElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkcontentelement) in XAML is different than assigning a name to a [Freezable](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.freezable) object.

To enable the targeting of a framework element or a framework content element in XAML, you set its [Name](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement.name) property. In code, you also need to use the [RegisterName](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.namescope.registername) method to register the element's name with the element for which you've created a [NameScope](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.namescope).

Rectangle myRectangle = new Rectangle();

myRectangle.Name = "MyRectangle";

// Create a name scope for the page.

NameScope.SetNameScope(this, new NameScope());

this.RegisterName(myRectangle.Name, myRectangle);

Storyboard.SetTargetName(myDoubleAnimation, myRectangle.Name);

Storyboard.SetTargetProperty(myDoubleAnimation,

new PropertyPath(Rectangle.WidthProperty));

[Freezable](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.freezable) types are those classes that inherit from the [Freezable](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.freezable) class. Examples of [Freezable](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.freezable) include [SolidColorBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.solidcolorbrush), [RotateTransform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.rotatetransform), and [GradientStop](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.gradientstop).

To enable the targeting of a [Freezable](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.freezable) by an animation in XAML, you use the [x:Name Directive](https://docs.microsoft.com/zh-cn/dotnet/framework/xaml-services/x-name-directive) to assign it a name. In code, you use the [RegisterName](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.namescope.registername) method to register its name with the element for which you've created a [NameScope](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.namescope).

<SolidColorBrush x:Name="MySolidColorBrush" Color="Blue" />

[Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard) objects use name scopes to resolve the [TargetName](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard.targetname) property. For more information about WPF name scopes, see [WPF XAML Namescopes](https://docs.microsoft.com/zh-cn/dotnet/framework/wpf/advanced/wpf-xaml-namescopes). If the [TargetName](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard.targetname) property is omitted, the animation targets the element on which it is defined, or, in the case of styles, the styled element.

## Indirect Targeting

There are times a [Freezable](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.freezable) can't be targeted directly by an animation, such as when the [Freezable](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.freezable) is declared as a resource or used to set a property value in a style

Instead of setting the [TargetName](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard.targetname) property with the name of the [Freezable](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.freezable), you give it the name of the element to which the [Freezable](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.freezable) "belongs."

To animate the brush, you would set the animation's [TargetProperty](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard.targetproperty) with a chain of properties that starts at the property of the framework element or framework content element the [Freezable](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.freezable) was used to set and ends with the [Freezable](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.freezable) property to animate.

<ColorAnimation

Storyboard.TargetName="Rectangle01"

Storyboard.TargetProperty="Fill.Color"

From="Blue" To="AliceBlue" Duration="0:0:1" />

DependencyProperty[] propertyChain =

new DependencyProperty[]

{Rectangle.FillProperty, SolidColorBrush.ColorProperty};

string thePath = "(0).(1)";

PropertyPath myPropertyPath = new PropertyPath(thePath, propertyChain);

Storyboard.SetTargetProperty(myColorAnimation, myPropertyPath);

Sometimes you need to target a freezable contained in a collection or array.

To target a freezable contained in a collection, you use the following path syntax.

|  |
| --- |
| *ElementPropertyName* .Children[ *CollectionIndex* ]. *FreezablePropertyName* |

<TransformGroup x:Key="MyTransformGroupResource"

x:Shared="False">

<ScaleTransform />

<RotateTransform />

</TransformGroup>

<DoubleAnimation

Storyboard.TargetName="Rectangle02"

Storyboard.TargetProperty="RenderTransform.Children[1].Angle"

From="0" To="360" Duration="0:0:1" />

## Interactively Controlling a Storyboard in XAML

If you give the [BeginStoryboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.beginstoryboard) a name by specifying its [Name](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.beginstoryboard.name) property, you make it a controllable storyboard. You can then interactively control the storyboard after it's started

<EventTrigger RoutedEvent="Button.Click" SourceName="BeginButton">

<BeginStoryboard Name="MyBeginStoryboard">

<Storyboard>

<DoubleAnimation

Storyboard.TargetName="MyRectangle"

Storyboard.TargetProperty="(Rectangle.Opacity)"

From="1.0" To="0.0" Duration="0:0:5" />

</Storyboard>

</BeginStoryboard>

</EventTrigger>

<EventTrigger RoutedEvent="Button.Click" SourceName="PauseButton">

<PauseStoryboard BeginStoryboardName="MyBeginStoryboard" />

</EventTrigger>

<EventTrigger RoutedEvent="Button.Click" SourceName="ResumeButton">

<ResumeStoryboard BeginStoryboardName="MyBeginStoryboard" />

</EventTrigger>

## Interactively Controlling a Storyboard by Using Code

The following list shows the methods that can be used to manipulate a [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard) after it has started:

* [Pause](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard.pause)
* [Resume](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard.resume)
* [Seek](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard.seek)
* [SkipToFill](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard.skiptofill)
* [Stop](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard.stop)
* [Remove](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard.remove)

## Animate in a Style

You can use [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard) objects to define animations in a [Style](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.style).

You don't specify a [TargetName](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard.targetname); the [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard) always targets the element to which the [Style](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.style) is applied. To target [Freezable](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.freezable) objects, you must use indirect targeting.

 You can't specify a [SourceName](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.eventtrigger.sourcename) for an [EventTrigger](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.eventtrigger) or a [Trigger](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.trigger).

 In XAML, you can't declare event handlers for [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard) or animation events.

## Animate in a ControlTemplate

You can use [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard) objects to define animations in a [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate).

The [TargetName](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard.targetname) may only refer to child objects of the [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate). If [TargetName](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard.targetname) is not specified, the animation targets the element to which the [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate) is applied.

 The [SourceName](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.eventtrigger.sourcename) for an [EventTrigger](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.eventtrigger) or a [Trigger](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.trigger) may only refer to child objects of the [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate).

 In XAML, you can't declare event handlers for [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard) or animation events.

# Timing Behaviors Overview

## Timeline Types

https://docs.microsoft.com/zh-cn/dotnet/framework/wpf/graphics-multimedia/timing-behaviors-overview

A [Timeline](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.timeline) represents a segment of time. It provides properties that enable you to specify the length of that segment, when it should start, how many times it will repeat, how fast time progresses in that segment, and more.

Classes that inherit from the timeline class provide additional functionality, such as animation and media playback. WPF provides the following [Timeline](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.timeline) types.

## Properties that Control the Length of a Timeline

### The Duration Property

### The RepeatBehavior Property

### The AutoReverse Property

<DoubleAnimation

Storyboard.TargetName="MyRectangle" Storyboard.TargetProperty="Width"

From="0" To="100" Duration="0:0:5"

RepeatBehavior="2"

AutoReverse="True" />

As a result, the [DoubleAnimation](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.doubleanimation) plays for 20 seconds: forward for five seconds, backwards for five seconds, forward for 5 seconds again, and then backwards for five seconds.

## The BeginTime Property

The [BeginTime](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.timeline.begintime) property enables you to specify when a timeline starts. A timeline's begin time is relative to its parent timeline

You may also set a timeline's begin time to null, which prevents the timeline from starting. In WPF, you specify null using the [x:Null Markup Extension](https://docs.microsoft.com/zh-cn/dotnet/framework/xaml-services/x-null-markup-extension).

Note that the begin time is not applied each time a timeline repeats because of its [RepeatBehavior](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.timeline.repeatbehavior) setting.

## The FillBehavior Property

When a [Timeline](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.timeline) reaches the end of its total active duration, the [FillBehavior](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.timeline.fillbehavior) property specifies whether it stops or holds its last value

# Bitmap Effects Overview

Bitmap effects enable designers and developers to apply visual effects to rendered Windows Presentation Foundation (WPF) content

Bitmap effects ([BitmapEffect](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.effects.bitmapeffect) object) are simple pixel processing operations. A bitmap effect takes a [BitmapSource](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.imaging.bitmapsource) as an input and produces a new [BitmapSource](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.imaging.bitmapsource) after applying the effect

effects can be set as properties on live [Visual](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.visual) objects, such as a [Button](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.button) or [TextBox](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.textbox). The pixel processing is applied and rendered at run-time. In this case, at the time of rendering, a [Visual](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.visual) is automatically converted to its [BitmapSource](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.imaging.bitmapsource) equivalent and is fed as input to the [BitmapEffect](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.effects.bitmapeffect). The output replaces the [Visual](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.visual) object's default rendering behavior. This is why [BitmapEffect](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.effects.bitmapeffect) objects force visuals to render in software only i.e. no hardware acceleration on visuals when effects are applied

## How to Apply an Effect

[BitmapEffect](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.effects.bitmapeffect) is a property on [Visual](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.visual).

[BitmapEffect](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement.bitmapeffect) can be set to a single [BitmapEffect](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.effects.bitmapeffect) object or multiple effects can be chained by using the [BitmapEffectGroup](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.effects.bitmapeffectgroup) object.

<Button Width="200">You Can't Read This!

<Button.BitmapEffect>

<!-- <BitmapEffectGroup> would go here if you wanted to apply more

then one effect to the Button. However, in this example only

one effect is being applied so BitmapEffectGroup does not need

to be included. -->

<!-- The larger the Radius, the more blurring. The default range is 20.

In addition, the KernelType is set to a box kernel. A box kernel

creates less disruption (less blur) then the default Gaussian kernel. -->

<BlurBitmapEffect Radius="10" KernelType="Box" />

</Button.BitmapEffect>

</Button>

# WPF Brushes Overview

Everything visible on your screen is visible because it was painted by a brush.

## Painting with a Brush

A [Brush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.brush) "paints" an area with its output

Most visual objects enable you to specify how they are painted. The following table lists some common objects and properties with which you can use a [Brush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.brush).

| **Class** | **Brush properties** |
| --- | --- |
| [Border](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.border) | [BorderBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.border.borderbrush), [Background](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.border.background) |
| [Control](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.control) | [Background](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.control.background), [Foreground](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.control.foreground) |
| [Panel](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.panel) | [Background](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.panel.background) |
| [Pen](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.pen) | [Brush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.pen.brush) |
| [Shape](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.shapes.shape) | [Fill](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.shapes.shape.fill), [Stroke](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.shapes.shape.stroke) |
| [TextBlock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.textblock) | [Background](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.textblock.background) |

## Paint with a Solid Color

A [SolidColorBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.solidcolorbrush) paints an area with a solid [Color](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.color). There are a variety of ways to specify the [Color](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.solidcolorbrush.color) of a [SolidColorBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.solidcolorbrush): for example, you can specify its alpha, red, blue, and green channels or use one of the predefined color provided by the [Colors](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.colors) class.

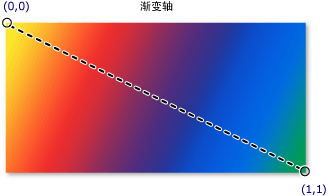
## Paint with a Linear Gradient

A [LinearGradientBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.lineargradientbrush) paints an area with a linear gradient. You use [GradientStop](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.gradientstop) objects to specify the colors in the gradient and their positions.

### The Gradient Axis

By manipulating the brush's [StartPoint](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.lineargradientbrush.startpoint) and [EndPoint](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.lineargradientbrush.endpoint), you can create horizontal and vertical gradients, reverse the gradient direction, condense the gradient spread, and more.

The default [StartPoint](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.lineargradientbrush.startpoint) of a [LinearGradientBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.lineargradientbrush) is (0,0), and its default [EndPoint](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.lineargradientbrush.endpoint) is (1,1), which creates a diagonal gradient starting at the upper-left corner and extending to the lower-right corner of the area being painted.



The following example shows how to create a horizontal gradient by specifying the brush's [StartPoint](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.lineargradientbrush.startpoint) and [EndPoint](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.lineargradientbrush.endpoint).

<Rectangle Width="200" Height="100">

<Rectangle.Fill>

<LinearGradientBrush StartPoint="0,0.5" EndPoint="1,0.5">

<GradientStop Color="Yellow" Offset="0.0" />

<GradientStop Color="Red" Offset="0.25" />

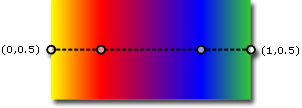
<GradientStop Color="Blue" Offset="0.75" />

<GradientStop Color="LimeGreen" Offset="1.0" />

</LinearGradientBrush>

</Rectangle.Fill>

</Rectangle>



## Paint with a Radial Gradient

<https://docs.microsoft.com/zh-cn/dotnet/framework/wpf/graphics-multimedia/painting-with-solid-colors-and-gradients-overview#radial-gradients>

## Paint with an Image

An [ImageBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.imagebrush) paints an area with a [ImageSource](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.imagesource).

ImageBrush myBrush = new ImageBrush();

myBrush.ImageSource =

new BitmapImage(new Uri(@"sampleImages\pinkcherries.jpg", UriKind.Relative));

exampleRectangle.Fill = myBrush;

<Rectangle Width="75" Height="75">

<Rectangle.Fill>

<ImageBrush ImageSource="sampleImages\pinkcherries.jpg" />

</Rectangle.Fill>

</Rectangle>

## Paint with a Drawing

A [DrawingBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawingbrush) paints an area with a [Drawing](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawing). A [Drawing](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawing) can contain shapes, images, text, and media. Shapes inside a drawing brush may themselves be painted with a solid color, gradient, image, or even another [DrawingBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawingbrush).

A [Drawing](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawing) object describes visible content, such as a shape, bitmap, video, or a line of text. Different types of drawings describe different types of content. The following is a list of the different types of drawing objects.

* [GeometryDrawing](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometrydrawing) – Draws a shape.
* [ImageDrawing](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.imagedrawing) – Draws an image.
* [GlyphRunDrawing](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.glyphrundrawing) – Draws text.
* [VideoDrawing](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.videodrawing) – Plays an audio or video file.
* [DrawingGroup](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawinggroup) – Draws other drawings. Use a drawing group to combine other drawings into a single composite drawing.

## Paint with a Visual

A [VisualBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.visualbrush) paints an area with a [Visual](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.visual) object.

## Paint using Predefined and System Brushes

For convenience, Windows Presentation Foundation (WPF) provides a set of predefined and system brushes that you can use to paint objects.

For a list of available predefined brushes, see the [Brushes](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.brushes) class

For a list of available system brushes, see the [SystemColors](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.systemcolors) class.

A system brush is a [SolidColorBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.solidcolorbrush) object that paints an area with the specified system color.

You can use system brushes as either a static or a dynamic resource.

## Common Brush Features

[Brush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.brush) objects provide an [Opacity](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.brush.opacity) property

if a brush has an opacity value of 0.5 and a color used in the brush also has an opacity value of 0.5, the output color has an opacity value of 0.25.

It's more efficient to change the opacity value of a brush than it is to change the opacity of an entire element using its [UIElement.Opacity](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement.opacity) property.

You can rotate, scale, skew, and translate a brush's content by using its [Transform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.brush.transform) or [RelativeTransform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.brush.relativetransform) properties.

# Transforms Overview

## What Is a Transform?

A [Transform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.transform) defines how to map, or transform, points from one coordinate space to another coordinate space. This mapping is described by a transformation [Matrix](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.matrix), which is a collection of three rows with three columns of [Double](https://docs.microsoft.com/zh-cn/dotnet/api/system.double) values.

By manipulating matrix values, you can rotate, scale, skew, and move (translate) an object.

Although Windows Presentation Foundation (WPF) enables you to directly manipulate matrix values, it also provides several [Transform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.transform) classes that enable you to transform an object without knowing how the underlying matrix structure is configured.

## Transform Classes

Windows Presentation Foundation (WPF) provides the following 2-D [Transform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.transform) classes for common transformation operations

[RotateTransform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.rotatetransform)

[ScaleTransform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.scaletransform)

[SkewTransform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.skewtransform)

[TranslateTransform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.translatetransform)

[AngleX](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.skewtransform.anglex) skews x-axis values relative to the original coordinate system

For creating more complex transformations, Windows Presentation Foundation (WPF) provides the following two classes:

[TransformGroup](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.transformgroup)

Groups multiple [TransformGroup](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.transformgroup) objects into a single [Transform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.transform) that you can then apply to transform properties.

[MatrixTransform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.matrixtransform)

Creates custom transformations that are not provided by the other [Transform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.transform) classes. When you use a [MatrixTransform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.matrixtransform), you manipulate a Matrix directly.

## Common Transformation Properties

One way to transform an object is to declare the appropriate [Transform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.transform) type and apply it to the transformation property of the object. Different types of objects have different types of transformation properties

| **Type** | **Transformation properties** |
| --- | --- |
| [Brush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.brush) | [Transform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.brush.transform), [RelativeTransform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.brush.relativetransform) |
| [ContainerVisual](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.containervisual) | [Transform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.containervisual.transform) |
| [DrawingGroup](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawinggroup) | [Transform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawinggroup.transform) |
| [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) | [RenderTransform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement.rendertransform), [LayoutTransform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement.layouttransform) |
| [Geometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometry) | [Transform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometry.transform) |
| [TextEffect](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.texteffect) | [Transform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.texteffect.transform) |
| [UIElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement) | [RenderTransform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement.rendertransform) |

## Transformations and Coordinate Systems

When you transform an object, you do not just transform the object, you transform coordinate space in which that object exists. By default, a transform is centered at the origin of the target object's coordinate system: (0,0).

## Transforming a FrameworkElement

To apply transformations to a [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement), create a [Transform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.transform) and apply it to one of the two properties that the [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) class provides:

* [LayoutTransform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement.layouttransform) – A transform that is applied before the layout pass. After the transform is applied, the layout system processes the transformed size and position of the element.
* [RenderTransform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement.rendertransform) – A transform that modifies the appearance of the element but is applied after the layout pass is complete. By using the [RenderTransform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement.rendertransform) property instead of the [LayoutTransform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement.layouttransform) property, you can obtain performance benefits.

The value of the [RenderTransformOrigin](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement.rendertransformorigin) property is relative to the size of the button. As a result, the rotation is applied to the center of the button

<Button Content="Rotated Button"

RenderTransformOrigin="0.5,0.5">

<Button.RenderTransform>

<RotateTransform Angle="45" />

</Button.RenderTransform>

</Button>

## Animating Transformations

Because they inherit from the [Animatable](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.animatable) class, the [Transform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.transform) classes can be animated.

## Freezable Features

Because it inherits from the [Freezable](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.freezable) class, [Transform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.transform) objects can be declared as [resources](https://docs.microsoft.com/zh-cn/dotnet/framework/wpf/advanced/xaml-resources), shared among multiple objects, made read-only to improve performance,

# Brush Transformation Overview

The Brush class provides two transformation properties: [Transform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.brush.transform) and [RelativeTransform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.brush.relativetransform). The properties enable you to rotate, scale, skew, and translate a brush's contents

## Differences between the Transform and RelativeTransform Properties

When you apply a transform to a brush's [Transform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.brush.transform) property, you need to know the size of the painted area if you want to transform the brush contents about its center

使用brush的[Transform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.brush.transform) 属性时，先将brush的content输出到paint area，然后对paint area做变换

When you apply a transform to a brush's [RelativeTransform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.brush.relativetransform) property, that transform is applied to the brush before its output is mapped to the painted area

The following list describes the order in which a brush’s contents are processed and transformed.

1. Process the brush’s contents. For a [GradientBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.gradientbrush), this means determining the gradient area. For a [TileBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.tilebrush), the [Viewbox](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.tilebrush.viewbox) is mapped to the [Viewport](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.tilebrush.viewport). This becomes the brush’s output.
2. Project the brush’s output onto the 1 x 1 transformation rectangle.
3. Apply the brush’s [RelativeTransform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.brush.relativetransform), if it has one.
4. Project the transformed output onto the area to paint.
5. Apply the brush’s [Transform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.transform), if it has one.

Because the [RelativeTransform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.brush.relativetransform) is applied while the brush’s output is mapped to a 1 x 1 rectangle，if you used a [RotateTransform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.rotatetransform) to rotate the brush's output 45 degrees about its center, you'd give the [RotateTransform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.rotatetransform) a [CenterX](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.rotatetransform.centerx) of 0.5 and a [CenterY](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.rotatetransform.centery) of 0.5.

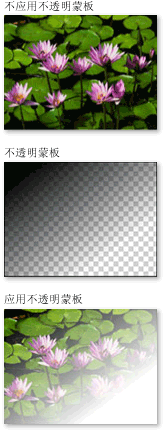
未完

# Opacity Masks Overview

To create an opacity mask, you apply a [Brush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.brush) to the [OpacityMask](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement.opacitymask) property of an element or [Visual](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.visual).

## Creating Visual Effects with Opacity Masks

The alpha channel of each of the brush's pixels are then used to determine the resulting opacity of the element or visual's corresponding pixels; the actual color of the brush is ignored.



## Creating an Opacity Mask

To create an opacity mask, you create a [Brush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.brush) and apply it to the [OpacityMask](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement.opacitymask) property of an element or visual。You can use any type of [Brush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.brush) as an opacity mask.

Because all of the colors in the [Colors](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.colors) class, except [Transparent](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.colors.transparent), are fully opaque, they can be used to simply define a starting color for a gradient opacity mask.

For additional control over alpha values when defining an opacity mask, you can specify the alpha channel of colors using ARGB hexadecimal notation in markup or using the [Color.FromScRgb](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.color.fromscrgb) method.

## Specifying Gradient Stops for an Opacity Mask

### Specifying Color Opacity in "XAML"

In Extensible Application Markup Language (XAML), you use ARGB hexadecimal notation to specify the opacity of individual colors. ARGB hexadecimal notation uses the following syntax:

# **aa** *rrggbb*

The aa in the previous line represents a two-digit hexadecimal value used to specify the opacity of the color.

## Using an Image as an Opacity Mask

To use an image as an opacity mask, use an [ImageBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.imagebrush) to contain the image

When creating an image to be used as an opacity mask, save the image in a format that supports multiple levels of transparency, such as Portable Network Graphics (PNG)

<Image

Height="150"

Width="200"

Source="sampleImages/Waterlilies.jpg"

HorizontalAlignment="Left"

Margin="10"

Grid.Column="2" Grid.Row="1">

<Image.OpacityMask>

<ImageBrush ImageSource="sampleImages/tornedges.png"/>

</Image.OpacityMask>

</Image>

## Painting an Area with a Solid Color

### Using a SolidColorBrush in "XAML"

<!-- This button's background is painted with a red SolidColorBrush,

described using a named color. -->

<Button Background="Red">A Button</Button>

<!-- This button's background is painted with a red SolidColorBrush,

described using hexadecimal notation. -->

<Button Background="#FFFF0000">A Button</Button>

<!-- Both of these buttons' backgrounds are painted with red

SolidColorBrush objects, described using object element

syntax. -->

<Button>A Button

<Button.Background>

<SolidColorBrush Color="Red" />

</Button.Background>

</Button>

### Painting with a SolidColorBrush in Code

myButton.Background = Brushes.Red;

SolidColorBrush mySolidColorBrush = new SolidColorBrush();

mySolidColorBrush.Color = Colors.Red;

myButton.Background = mySolidColorBrush;

The static [FromArgb](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.color.fromargb) enables you to specify the color's alpha, red, green, and blue values.

SolidColorBrush mySolidColorBrush = new SolidColorBrush();

mySolidColorBrush.Color =

Color.FromArgb(

255, // Specifies the transparency of the color.

255, // Specifies the amount of red.

0, // specifies the amount of green.

0); // Specifies the amount of blue.

myButton.Background = mySolidColorBrush;

## Painting an Area with a Gradient

A gradient brush paints an area with multiple colors that blend into each other along an axis.

The gradient examples in this topic use the default coordinate system for setting start points and end points. The default coordinate system is relative to a bounding box: 0 indicates 0 percent of the bounding box and 1 indicates 100 percent of the bounding box. You can change this coordinate system by setting the [MappingMode](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.gradientbrush.mappingmode) property to the value [Absolute](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.brushmappingmode#System_Windows_Media_BrushMappingMode_Absolute).

You may change the orientation and size of the line using the brush's [StartPoint](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.lineargradientbrush.startpoint) and [EndPoint](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.lineargradientbrush.endpoint) properties.

## Radial Gradients

A radial gradient brush's axis is defined by a circle; its colors "radiate" outward from its origin.

The [GradientOrigin](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.radialgradientbrush.gradientorigin) specifies the start point of a radial gradient brush's gradient axis. The gradient axis radiates from the gradient origin to the gradient circle. A brush's gradient circle is defined by its [Center](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.radialgradientbrush.center), [RadiusX](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.radialgradientbrush.radiusx), and [RadiusY](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.radialgradientbrush.radiusy) properties.

[RadiusX](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.radialgradientbrush.radiusx), and [RadiusY](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.radialgradientbrush.radiusy)也是相对值，相对paint area 的长和宽

<RadialGradientBrush

GradientOrigin="0.5,0.5" Center="0.5,0.5"

RadiusX="0.5" RadiusY="0.5">

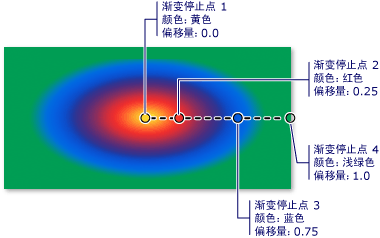
<GradientStop Color="Yellow" Offset="0" />

<GradientStop Color="Red" Offset="0.25" />

<GradientStop Color="Blue" Offset="0.75" />

<GradientStop Color="LimeGreen" Offset="1" />

</RadialGradientBrush>



# Drawing Objects Overview

A [Drawing](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawing) object describes visible content, such as a shape, bitmap, video, or a line of text. Different types of drawings describe different types of content. The following is a list of the different types of drawing objects.

* [GeometryDrawing](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometrydrawing) – Draws a shape.
* [ImageDrawing](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.imagedrawing) – Draws an image.
* [GlyphRunDrawing](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.glyphrundrawing) – Draws text.
* [VideoDrawing](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.videodrawing) – Plays an audio or video file.
* [DrawingGroup](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawinggroup) – Draws other drawings. Use a drawing group to combine other drawings into a single composite drawing.

WPF provides other types of objects that are capable of drawing shapes, bitmaps, text, and media.

when should you use [Drawing](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawing) objects? When you can sacrifice framework level features to gain performance benefits or when you need [Freezable](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.freezable) features.

[Drawing](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawing) objects lack support for [Layout](https://docs.microsoft.com/zh-cn/dotnet/framework/wpf/advanced/layout), input, and focus, they provide performance benefits

## Draw a Shape

To draw a shape, you use a [GeometryDrawing](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometrydrawing). A geometry drawing's [Geometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometrydrawing.geometry) property describes the shape to draw, its [Brush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometrydrawing.brush) property describes how the interior of the shape should be painted, and its [Pen](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometrydrawing.pen) property describes how its outline should be drawn.

## Draw an Image

To draw an image, you use an [ImageDrawing](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.imagedrawing). An [ImageDrawing](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.imagedrawing) object's [ImageSource](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.imagedrawing.imagesource) property describes the image to draw, and its [Rect](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.imagedrawing.rect) property defines the region where the image is drawn.

A [DrawingGroup](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawinggroup) also enables you to apply opacity masks, transforms, bitmap effects, and other operations to its contents

## Display a Drawing as an Image

## Play Media (Code Only)

## Draw Text

To draw text, you use a [GlyphRunDrawing](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.glyphrundrawing) and a [GlyphRun](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.glyphrun).

GlyphRun theGlyphRun = new GlyphRun(

new GlyphTypeface(new Uri(@"C:\WINDOWS\Fonts\TIMES.TTF")),

0,

false,

13.333333333333334,

new ushort[]{43, 72, 79, 79, 82, 3, 58, 82, 85, 79, 71},

new Point(0, 12.29),

new double[]{

9.62666666666667, 7.41333333333333, 2.96,

2.96, 7.41333333333333, 3.70666666666667,

12.5866666666667, 7.41333333333333,

4.44, 2.96, 7.41333333333333},

null,

null,

null,

null,

null,

null

);

GlyphRunDrawing gDrawing = new GlyphRunDrawing(Brushes.Black, theGlyphRun);

A [GlyphRun](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.glyphrun) is a low-level object intended for use with fixed-format document presentation and print scenarios. A simpler way to draw text to the screen is to use a [Label](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.label) or a [TextBlock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.textblock).

## Composite Drawings

To display a [Drawing](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawing) with an [Image](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.image) control, use a [DrawingImage](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawingimage) as the [Image](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.image) control's [Source](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.image.source) and set the [DrawingImage](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawingimage) object's [DrawingImage.Drawing](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawingimage.drawing) property to the drawing you want to display.

//

// Use a DrawingImage and an Image control

// to display the drawing.

//

DrawingImage geometryImage = new DrawingImage(aGeometryDrawing);

// Freeze the DrawingImage for performance benefits.

geometryImage.Freeze();

Image anImage = new Image();

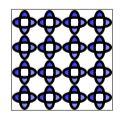
anImage.Source = geometryImage;

anImage.HorizontalAlignment = HorizontalAlignment.Left;

## Paint an Object with a Drawing

A [DrawingBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawingbrush) is a type of brush that paints an area with a drawing object. You can use it to paint just about any graphical object with a drawing

The following examples uses a [DrawingBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawingbrush) to paint the [Fill](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.shapes.shape.fill) of a [Rectangle](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.shapes.rectangle) with a pattern created from a [GeometryDrawing](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometrydrawing). This example produces the following output.



DrawingBrush patternBrush = new DrawingBrush(aGeometryDrawing);

patternBrush.Viewport = new Rect(0, 0, 0.25, 0.25);

patternBrush.TileMode = TileMode.Tile;

patternBrush.Freeze();

//

// Create an object to paint.

//

Rectangle paintedRectangle = new Rectangle();

paintedRectangle.Width = 100;

paintedRectangle.Height = 100;

paintedRectangle.Fill = patternBrush;

# Geometry Overview

## What Is a Geometry?

The [Geometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometry) class and the classes which derive from it enable you to describe the geometry of a 2-D shape. These geometric descriptions have many uses, such defining a shape to paint to the screen or defining hit-test and clip regions. You can even use a geometry to define an animation path.

## Geometries vs. Shapes

For one, the [Geometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometry) class inherits from the [Freezable](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.freezable) class while the [Shape](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.shapes.shape) class inherits from [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement). Because they are elements, [Shape](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.shapes.shape) objects can render themselves and participate in the layout system, while [Geometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometry) objects cannot.

Although [Shape](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.shapes.shape) objects are more readily usable than [Geometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometry) objects, [Geometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometry) objects are more versatile. While a [Shape](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.shapes.shape) object is used to render 2-D graphics,

### The Path Shape

One [Shape](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.shapes.shape), the [Path](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.shapes.path) class, actually uses a [Geometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometry) to describe its contents. By setting the [Data](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.shapes.path.data) property of the [Path](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.shapes.path) with a [Geometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometry) and setting its [Fill](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.shapes.shape.fill) and [Stroke](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.shapes.shape.stroke) properties, you can render a [Geometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometry).

## Common Properties That Take a Geometry

The following table lists several classes that have properties that take a [Geometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometry) object.

| **Type** | **Property** |
| --- | --- |
| [DoubleAnimationUsingPath](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.doubleanimationusingpath) | [PathGeometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.doubleanimationusingpath.pathgeometry) |
| [DrawingGroup](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawinggroup) | [ClipGeometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawinggroup.clipgeometry) |
| [GeometryDrawing](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometrydrawing) | [Geometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometrydrawing.geometry) |
| [Path](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.shapes.path) | [Data](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.shapes.path.data) |
| [UIElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement) | [Clip](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement.clip) |

The classes which derive from the [Geometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometry) class can be roughly grouped into three categories: simple geometries, path geometries, and composite geometries

## Simple Geometry Types

LineGeometry

EllipseGeometry

RectangleGeometry

## Path Geometries

The [PathGeometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.pathgeometry) class and its lightweight equivalent, the [StreamGeometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.streamgeometry) class, provide the means to describe multiple complex figures composed of arcs, curves, and lines

At the heart of a [PathGeometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.pathgeometry) is a collection of [PathFigure](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.pathfigure) objects, so named because each figure describes a discrete shape in the [PathGeometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.pathgeometry). Each [PathFigure](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.pathfigure) is itself comprised of one or more [PathSegment](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.pathsegment) objects, each of which describes a segment of the figure.

The segments within a [PathFigure](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.pathfigure) are combined into a single geometric shape with the end point of each segment being the start point of the next segment

Unlike a [PathGeometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.pathgeometry), the contents of a [StreamGeometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.streamgeometry) do not support data binding, animation, or modification.

## Composite Geometries

Composite geometry objects can be created using a [GeometryGroup](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometrygroup), a [CombinedGeometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.combinedgeometry), or by calling the static [Geometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometry) method [Combine](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometry.combine).

The [CombinedGeometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.combinedgeometry) object and the [Combine](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometry.combine) method performs a Boolean operation to combine the area defined by two geometries

The [GeometryGroup](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometrygroup) class creates an amalgamation of the [Geometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometry) objects it contains without combining their area. Any number of [Geometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometry) objects can be added to a [GeometryGroup](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometrygroup).

## Combined Geometries

the [CombinedGeometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.combinedgeometry) object and the [Combine](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometry.combine) method combine the area defined by the geometries they contain. The [GeometryCombineMode](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometrycombinemode) enumeration specifies how the geometries are combined

[Union](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometrycombinemode#System_Windows_Media_GeometryCombineMode_Union), [Intersect](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometrycombinemode#System_Windows_Media_GeometryCombineMode_Intersect), [Exclude](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometrycombinemode#System_Windows_Media_GeometryCombineMode_Exclude), and [Xor](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometrycombinemode#System_Windows_Media_GeometryCombineMode_Xor).

<Path.Data>

<!-- Combines two geometries using the union combine mode. -->

<CombinedGeometry GeometryCombineMode="Union">

<CombinedGeometry.Geometry1>

<EllipseGeometry RadiusX="50" RadiusY="50" Center="75,75" />

</CombinedGeometry.Geometry1>

<CombinedGeometry.Geometry2>

<EllipseGeometry RadiusX="50" RadiusY="50" Center="125,75" />

</CombinedGeometry.Geometry2>

</CombinedGeometry>

</Path.Data>

# Path Markup Syntax

## StreamGeometry and PathFigureCollection Mini-Languages

WPF provides two classes that provide mini-languages for describing geometric paths: [StreamGeometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.streamgeometry) and [PathFigureCollection](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.pathfigurecollection).

You use the [StreamGeometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.streamgeometry) mini-language when setting a property of type [Geometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.geometry)

<Path Stroke="Black" Fill="Gray"

Data="M 10,100 C 10,300 300,-200 300,100" />

You use the [PathFigureCollection](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.pathfigurecollection) mini-language when setting the [Figures](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.pathgeometry.figures) property of a [PathGeometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.pathgeometry).

<Path Stroke="Black" Fill="Gray">

<Path.Data>

<PathGeometry Figures="M 10,100 C 10,300 300,-200 300,100" />

</Path.Data>

</Path>

## Move Command

Specifies the start point of a new figure.

M startPoint  
  
- or -  
  
m startPoint

## Draw Commands

A draw command can consist of several shape commands.

# Path Animations Overview

A path animation is a type of [AnimationTimeline](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.animationtimeline) that uses a [PathGeometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.pathgeometry) as its input. As the path animation progresses, it reads the x, y, and angle information from the path and uses that information to generate its output.

One way to move an object along a path is to use a [MatrixTransform](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.matrixtransform) and a [MatrixAnimationUsingPath](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.matrixanimationusingpath) to transform an object along a complex path.

## Path Animation Types

WPF provides the following path animation classes.

[DoubleAnimationUsingPath](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.doubleanimationusingpath)

[MatrixAnimationUsingPath](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.matrixanimationusingpath)

[PointAnimationUsingPath](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.pointanimationusingpath)

A [PointAnimationUsingPath](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.pointanimationusingpath) generates [Point](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.point) values from the x- and y-coordinates of its [PathGeometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.pointanimationusingpath.pathgeometry)

A [DoubleAnimationUsingPath](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.doubleanimationusingpath) generates [Double](https://docs.microsoft.com/zh-cn/dotnet/api/system.double) values from its [PathGeometry](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.doubleanimationusingpath.pathgeometry). By setting the [Source](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.doubleanimationusingpath.source) property, you can specify whether the [DoubleAnimationUsingPath](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.doubleanimationusingpath) uses the x-coordinate, y-coordinate, or angle of the path as its output.

# Imaging Overview

WPF Imaging enables developers to display, transform, and format images.

There are two ways to access the WPF Imaging API, a managed component and an unmanaged component. The unmanaged component provides the following features.

## Displaying Images in WPF

Images can be displayed using an [Image](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.image) control, painted on a visual using an [ImageBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.imagebrush), or drawn using an [ImageDrawing](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.imagedrawing).

### Using the Image Control

<Image Width="200">

<Image.Source>

<!-- To save significant application memory, set the DecodePixelWidth or

DecodePixelHeight of the BitmapImage value of the image source to the desired

height and width of the rendered image. If you don't do this, the application will

cache the image as though it were rendered as its normal size rather then just

the size that is displayed. -->

<!-- Note: In order to preserve aspect ratio, only set either DecodePixelWidth

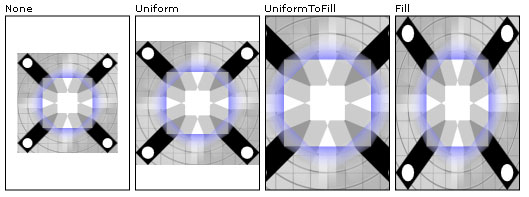
or DecodePixelHeight but not both. -->

<BitmapImage DecodePixelWidth="200"

UriSource="C:\Documents and Settings\All Users\Documents\My Pictures\Sample Pictures\Water Lilies.jpg" />

</Image.Source>

The [Stretch](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.image.stretch) property controls how an image is stretched to fill its container.



### Painting with Images

Images can also be displayed in an application by painting with a [Brush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.brush). To paint with images, use an [ImageBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.imagebrush).

<Button.Background>

<ImageBrush ImageSource="sampleImages\blueberries.jpg" />

</Button.Background>

# Controls

a class does not need to inherit from the [Control](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.control) class to have a visible presence.

Classes that inherit from the [Control](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.control) class contain a [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate),

## Changing the Appearance of a Control

 Change the value of a property of the control.

 Create a [Style](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.style) for the control.

 Create a new [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate) for the control.

WPF gives you the ability to specify the appearance of controls wholesale（批量）, instead of setting properties on each instance in the application, by creating a [Style](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.style).

# WPF Content Model

Windows Presentation Foundation (WPF) is a presentation platform that provides many controls and control-like types whose primary purpose is to display different types of content

The content model describes what content can be used in a control

This topic also lists the content properties for each content model. A content property is a property that is used to store the content of the object.

## Classes That Contain Arbitrary Content

Some controls can contain an object of any type

| **Class that contains arbitrary content** | **Content** |
| --- | --- |
| [ContentControl](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.contentcontrol) | A single arbitrary object. |
| [HeaderedContentControl](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.headeredcontentcontrol) | A header and a single item, both of which are arbitrary objects. |
| [ItemsControl](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.itemscontrol) | A collection of arbitrary objects. |
| [HeaderedItemsControl](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.headereditemscontrol) | A header and a collection of items, all of which are arbitrary objects. |

The [ContentControl](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.contentcontrol) class contains a single piece of arbitrary content. Its content property is [Content](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.contentcontrol.content).

The [ItemsControl](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.itemscontrol) class inherits from [Control](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.control)，Its content properties are [ItemsSource](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.itemscontrol.itemssource) and [Items](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.itemscontrol.items). [ItemsSource](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.itemscontrol.itemssource) is typically used to populate the [ItemsControl](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.itemscontrol) with a data collection。If you do not want to use a collection to populate the [ItemsControl](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.itemscontrol), you can add items by using the [Items](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.itemscontrol.items) property.

## Classes That Contain a Collection of UIElement Objects

The [Panel](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.panel) class positions and arranges child [UIElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement) objects. Its content property is [Children](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.panel.children).

## Classes That Affect the Appearance of a UIElement

The [Decorator](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.decorator) class applies visual effects onto or around a single child [UIElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement). Its content property is [Child](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.decorator.child).

## Classes That Enable Users to Enter Text

[TextBox](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.textbox)，[RichTextBox](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.richtextbox)，[PasswordBox](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.passwordbox)

## Classes That Display Your Text

[TextBlock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.textblock) to display small amounts of text. If you want to display large amounts of text, use the [FlowDocumentReader](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.flowdocumentreader), [FlowDocumentPageViewer](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.flowdocumentpageviewer), or [FlowDocumentScrollViewer](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.flowdocumentscrollviewer) controls.

## Classes That Format Your Text

[TextElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.documents.textelement) and its related classes allow you to format text. [TextElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.documents.textelement) objects contain and format text in [TextBlock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.textblock) and [FlowDocument](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.documents.flowdocument) objects

## ContextMenu Control

A [ContextMenu](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.contextmenu) is attached to a specific control. The [ContextMenu](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.contextmenu) element enables you to present users with a list of items that specify commands or options that are associated with a particular control,

### To customize cell validation feedback

Set the column's [EditingElementStyle](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.datagridboundcolumn.editingelementstyle) property to a style appropriate for the column's editing control. Because the editing controls are created at run time, you cannot use the [Validation.ErrorTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.validation.errortemplate) attached property like you would with simple controls.

<DataGrid.Resources>

<Style x:Key="errorStyle" TargetType="{x:Type TextBox}">

<Setter Property="Padding" Value="-2"/>

<Style.Triggers>

<Trigger Property="Validation.HasError" Value="True">

<Setter Property="Background" Value="Red"/>

<Setter Property="ToolTip"

Value="{Binding RelativeSource={RelativeSource Self},

Path=(Validation.Errors)[0].ErrorContent}"/>

</Trigger>

</Style.Triggers>

</Style>

</DataGrid.Resources>

<DataGrid.Columns>

<DataGridTextColumn Header="Course Name"

Binding="{Binding Name, TargetNullValue=(enter a course name)}"/>

<DataGridTextColumn Header="Course ID"

EditingElementStyle="{StaticResource errorStyle}"

Binding="{Binding Id, ValidatesOnExceptions=True}"/>

<DataGridTextColumn Header="Start Date"

EditingElementStyle="{StaticResource errorStyle}"

Binding="{Binding StartDate, ValidatesOnExceptions=True,

StringFormat=d}"/>

<DataGridTextColumn Header="End Date"

EditingElementStyle="{StaticResource errorStyle}"

Binding="{Binding EndDate, ValidatesOnExceptions=True,

StringFormat=d}"/>

</DataGrid.Columns>

### To validate multiple values in a single row

Implement a [ValidationRule](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.validationrule) subclass that checks multiple properties of the bound data object. In your [Validate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.validationrule.validate) method implementation, cast the value parameter value to a [BindingGroup](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.data.bindinggroup) instance. You can then access the data object through the [Items](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.data.bindinggroup.items) property.

public override ValidationResult Validate(object value,

System.Globalization.CultureInfo cultureInfo)

{

Course course = (value as BindingGroup).Items[0] as Course;

if (course.StartDate > course.EndDate)

{

return new ValidationResult(false,

"Start Date must be earlier than End Date.");

}

else

{

return ValidationResult.ValidResult;

}

}

Add the validation rule to the [DataGrid.RowValidationRules](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.datagrid.rowvalidationrules) collection.

<DataGrid.RowValidationRules>

<local:CourseValidationRule ValidationStep="UpdatedValue"/>

</DataGrid.RowValidationRules>

### To customize row validation feedback

Set the [DataGrid.RowValidationErrorTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.datagrid.rowvalidationerrortemplate) property. This property enables you to customize row validation feedback for individual [DataGrid](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.datagrid) controls

# How to: Group, sort, and filter data in the DataGrid control

To group, sort, and filter the data in a [DataGrid](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.datagrid), you bind it to a [CollectionView](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.data.collectionview) that supports these functions. You can then work with the data in the [CollectionView](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.data.collectionview) without affecting the underlying source data. The changes in the collection view are reflected in the [DataGrid](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.datagrid) user interface (UI).

a collection of Task objects is bound to a [CollectionViewSource](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.data.collectionviewsource). The [CollectionViewSource](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.data.collectionviewsource) is used as the [ItemsSource](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.itemscontrol.itemssource) for the [DataGrid](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.datagrid).

# Styles and templates in WPF

## Styles

You can think of a [Style](https://docs.microsoft.com/en-us/dotnet/api/system.windows.style) as a convenient way to apply a set of property values to multiple elements. You can use a style on any element that derives from [FrameworkElement](https://docs.microsoft.com/en-us/dotnet/api/system.windows.frameworkelement) or [FrameworkContentElement](https://docs.microsoft.com/en-us/dotnet/api/system.windows.frameworkcontentelement) such as a [Window](https://docs.microsoft.com/en-us/dotnet/api/system.windows.window) or a [Button](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.button).

## ControlTemplates

In WPF, the [ControlTemplate](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.controltemplate) of a control defines the appearance of the control.

## DataTemplates

In WPF, you use a [DataTemplate](https://docs.microsoft.com/en-us/dotnet/api/system.windows.datatemplate) to define the visual representation of data. Basically, what you put into a [DataTemplate](https://docs.microsoft.com/en-us/dotnet/api/system.windows.datatemplate) determines what the data looks like in the rendered application.

the [DataType](https://docs.microsoft.com/en-us/dotnet/api/system.windows.datatemplate.datatype) property is similar to the [TargetType](https://docs.microsoft.com/en-us/dotnet/api/system.windows.style.targettype) property of the [Style](https://docs.microsoft.com/en-us/dotnet/api/system.windows.style)

<Window.Resources>

<!-- .... other resources .... -->

<!--DataTemplate to display Photos as images

instead of text strings of Paths-->

<DataTemplate DataType="{x:Type local:Photo}">

<Border Margin="3">

<Image Source="{Binding Source}"/>

</Border>

</DataTemplate>

</Window.Resources>

## Triggers

A trigger sets properties or starts actions, such as an animation, when a property value changes or when an event is raised. [Style](https://docs.microsoft.com/en-us/dotnet/api/system.windows.style), [ControlTemplate](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.controltemplate), and [DataTemplate](https://docs.microsoft.com/en-us/dotnet/api/system.windows.datatemplate) all have a Triggers property that can contain a set of triggers. There are various types of triggers.

### PropertyTriggers

A [Trigger](https://docs.microsoft.com/en-us/dotnet/api/system.windows.trigger) that sets property values or starts actions based on the value of a property is called a property trigger

the [Trigger](https://docs.microsoft.com/en-us/dotnet/api/system.windows.trigger) class also has the [EnterActions](https://docs.microsoft.com/en-us/dotnet/api/system.windows.triggerbase.enteractions) and [ExitActions](https://docs.microsoft.com/en-us/dotnet/api/system.windows.triggerbase.exitactions) properties that enable a trigger to perform actions.

### EventTriggers and Storyboards

Another type of trigger is the [EventTrigger](https://docs.microsoft.com/en-us/dotnet/api/system.windows.eventtrigger), which starts a set of actions based on the occurrence of an event

<EventTrigger RoutedEvent="Mouse.MouseEnter">

<EventTrigger.Actions>

<BeginStoryboard>

<Storyboard>

<DoubleAnimation

Duration="0:0:0.2"

Storyboard.TargetProperty="MaxHeight"

To="90" />

</Storyboard>

</BeginStoryboard>

</EventTrigger.Actions>

</EventTrigger>

# DataTrigger

Represents a trigger that applies property values or performs actions when the bound data meets a specified condition.

<DataTrigger Binding="{Binding Path=State}" Value="WA">

<Setter Property="Foreground" Value="Red" />

</DataTrigger>

## Shared resources and themes

A typical Windows Presentation Foundation (WPF) application might have multiple user interface (UI) resources that are applied throughout the application. Collectively, this set of resources can be considered the theme for the application. WPF provides support for packaging user interface (UI) resources as a theme by using a resource dictionary that is encapsulated as the [ResourceDictionary](https://docs.microsoft.com/en-us/dotnet/api/system.windows.resourcedictionary) class.

WPF theme resources are stored in embedded resource dictionaries. These resource dictionaries must be embedded within a signed assembly

You can define resource dictionaries as individual files that enable you to reuse a theme across multiple applications

<ResourceDictionary.MergedDictionaries>

<ResourceDictionary Source="Shared.xaml" />

</ResourceDictionary.MergedDictionaries>

# Data Templating Overview

<https://docs.microsoft.com/en-us/dotnet/framework/wpf/data/data-templating-overview>

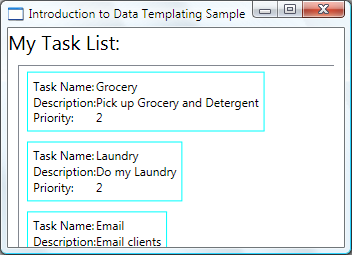
the [ListBox](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.listbox) by default calls ToString when trying to display the objects in the collection.

The solution is to define a [DataTemplate](https://docs.microsoft.com/en-us/dotnet/api/system.windows.datatemplate). One way to do that is to set the [ItemTemplate](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.itemscontrol.itemtemplate) property of the [ListBox](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.listbox) to a [DataTemplate](https://docs.microsoft.com/en-us/dotnet/api/system.windows.datatemplate).

The [DataTemplate](https://docs.microsoft.com/en-us/dotnet/api/system.windows.datatemplate) class has a [DataType](https://docs.microsoft.com/en-us/dotnet/api/system.windows.datatemplate.datatype) property ,Note that in this case the x:Key is set implicitly. Therefore, if you assign this [DataTemplate](https://docs.microsoft.com/en-us/dotnet/api/system.windows.datatemplate) an x:Key value, you are overriding the implicit x:Key and the [DataTemplate](https://docs.microsoft.com/en-us/dotnet/api/system.windows.datatemplate) would not be applied automatically.

If you are binding a [ContentControl](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.contentcontrol) to a collection of Task objects, the [ContentControl](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.contentcontrol) does not use the above [DataTemplate](https://docs.microsoft.com/en-us/dotnet/api/system.windows.datatemplate) automatically. This is because the binding on a [ContentControl](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.contentcontrol) needs more information to distinguish whether you want to bind to an entire collection or the individual objects.

If your [ContentControl](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.contentcontrol) is tracking the selection of an [ItemsControl](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.itemscontrol) type, you can set the [Path](https://docs.microsoft.com/en-us/dotnet/api/system.windows.data.binding.path) property of the [ContentControl](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.contentcontrol) binding to "/" to indicate that you are interested in the current item



We can set [HorizontalContentAlignment](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.control.horizontalcontentalignment) to [Stretch](https://docs.microsoft.com/en-us/dotnet/api/system.windows.horizontalalignment#System_Windows_HorizontalAlignment_Stretch) on the [ListBox](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.listbox) to make sure the width of the items takes up the entire space

### Use DataTriggers to Apply Property Values

<DataTemplate.Triggers>

<DataTrigger Binding="{Binding Path=TaskType}">

<DataTrigger.Value>

<local:TaskType>Home</local:TaskType>

</DataTrigger.Value>

<Setter TargetName="border" Property="BorderBrush" Value="Yellow"/>

</DataTrigger>

</DataTemplate.Triggers>

An alternative way to achieve the same effect is to bind the [BorderBrush](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.border.borderbrush) property to the TaskType property and use a value converter to return the color based on the TaskType value

### What Belongs in a DataTemplate?

if the properties that your Setters are concerned with are not properties of elements that are within the current [DataTemplate](https://docs.microsoft.com/en-us/dotnet/api/system.windows.datatemplate), it may be more suitable to set the properties using a [Style](https://docs.microsoft.com/en-us/dotnet/api/system.windows.style) that is for the [ListBoxItem](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.listboxitem) class

our [DataTemplate](https://docs.microsoft.com/en-us/dotnet/api/system.windows.datatemplate) is concerned with only the presentation and appearance of the data objects. In most cases, all other aspects of presentation, such as what an item looks like when it is selected or how the [ListBox](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.listbox) lays out the items, do not belong in the definition of a [DataTemplate](https://docs.microsoft.com/en-us/dotnet/api/system.windows.datatemplate)

## Choosing a DataTemplate Based on Properties of the Data Object

when you have a [CompositeCollection](https://docs.microsoft.com/en-us/dotnet/api/system.windows.data.compositecollection) of different types or collections with items of different types , you can define different data templates with different [DataType Property](https://docs.microsoft.com/en-us/dotnet/framework/wpf/data/data-templating-overview#Styling_DataType) for different data objects

collection of the same type of data objects you can create a [DataTemplate](https://docs.microsoft.com/en-us/dotnet/api/system.windows.datatemplate) and then use triggers to apply changes based on the property values of each data object

Some scenarios may require you to create a different [DataTemplate](https://docs.microsoft.com/en-us/dotnet/api/system.windows.datatemplate) for data objects that are of the same type

create a subclass of [DataTemplateSelector](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.datatemplateselector) and override the [SelectTemplate](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.datatemplateselector.selecttemplate) method.

public class TaskListDataTemplateSelector : DataTemplateSelector

{

public override DataTemplate

SelectTemplate(object item, DependencyObject container)

{

FrameworkElement element = container as FrameworkElement;

if (element != null && item != null && item is Task)

{

Task taskitem = item as Task;

if (taskitem.Priority == 1)

return

element.FindResource("importantTaskTemplate") as DataTemplate;

else

return

element.FindResource("myTaskTemplate") as DataTemplate;

}

return null;

}

}

We can then declare the TaskListDataTemplateSelector as a resource:

To use the template selector resource, assign it to the [ItemTemplateSelector](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.itemscontrol.itemtemplateselector) property of the [ListBox](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.listbox)

<ListBox Width="400" Margin="10"

ItemsSource="{Binding Source={StaticResource myTodoList}}"

ItemTemplateSelector="{StaticResource myDataTemplateSelector}"

HorizontalContentAlignment="Stretch"/>

Notice this example uses the [DataTemplate](https://docs.microsoft.com/en-us/dotnet/api/system.windows.datatemplate).[Resources](https://docs.microsoft.com/en-us/dotnet/api/system.windows.frameworktemplate.resources) property. Resources defined in that section are shared by the elements within the [DataTemplate](https://docs.microsoft.com/en-us/dotnet/api/system.windows.datatemplate).

<DataTemplate x:Key="importantTaskTemplate">

<DataTemplate.Resources>

<Style TargetType="TextBlock">

<Setter Property="FontSize" Value="20"/>

</Style>

</DataTemplate.Resources>

<Border Name="border" BorderBrush="Red" BorderThickness="1"

Padding="5" Margin="5">

<DockPanel HorizontalAlignment="Center">

<TextBlock Text="{Binding Path=Description}" />

<TextBlock>!</TextBlock>

</DockPanel>

</Border>

</DataTemplate>

## Styling and Templating an ItemsControl

In order to know when it is not suitable to use a [DataTemplate](https://docs.microsoft.com/en-us/dotnet/api/system.windows.datatemplate) it is important to understand the different style and template properties provided by the [ItemsControl](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.itemscontrol).

<ItemsControl Margin="10"

ItemsSource="{Binding Source={StaticResource myTodoList}}">

<!--The ItemsControl has no default visual appearance.

Use the Template property to specify a ControlTemplate to define

the appearance of an ItemsControl. The ItemsPresenter uses the specified

ItemsPanelTemplate (see below) to layout the items. If an

ItemsPanelTemplate is not specified, the default is used. (For ItemsControl,

the default is an ItemsPanelTemplate that specifies a StackPanel.-->

<ItemsControl.Template>

<ControlTemplate TargetType="ItemsControl">

<Border BorderBrush="Aqua" BorderThickness="1" CornerRadius="15">

<ItemsPresenter/>

</Border>

</ControlTemplate>

</ItemsControl.Template>

<!--Use the ItemsPanel property to specify an ItemsPanelTemplate

that defines the panel that is used to hold the generated items.

In other words, use this property if you want to affect

how the items are laid out.-->

<ItemsControl.ItemsPanel>

<ItemsPanelTemplate>

<WrapPanel />

</ItemsPanelTemplate>

</ItemsControl.ItemsPanel>

<!--Use the ItemTemplate to set a DataTemplate to define

the visualization of the data objects. This DataTemplate

specifies that each data object appears with the Proriity

and TaskName on top of a silver ellipse.-->

<ItemsControl.ItemTemplate>

<DataTemplate>

<DataTemplate.Resources>

<Style TargetType="TextBlock">

<Setter Property="FontSize" Value="18"/>

<Setter Property="HorizontalAlignment" Value="Center"/>

</Style>

</DataTemplate.Resources>

<Grid>

<Ellipse Fill="Silver"/>

<StackPanel>

<TextBlock Margin="3,3,3,0"

Text="{Binding Path=Priority}"/>

<TextBlock Margin="3,0,3,7"

Text="{Binding Path=TaskName}"/>

</StackPanel>

</Grid>

</DataTemplate>

</ItemsControl.ItemTemplate>

<!--Use the ItemContainerStyle property to specify the appearance

of the element that contains the data. This ItemContainerStyle

gives each item container a margin and a width. There is also

a trigger that sets a tooltip that shows the description of

the data object when the mouse hovers over the item container.-->

<ItemsControl.ItemContainerStyle>

<Style>

<Setter Property="Control.Width" Value="100"/>

<Setter Property="Control.Margin" Value="5"/>

<Style.Triggers>

<Trigger Property="Control.IsMouseOver" Value="True">

<Setter Property="Control.ToolTip"

Value="{Binding RelativeSource={x:Static RelativeSource.Self},

Path=Content.Description}"/>

</Trigger>

</Style.Triggers>

</Style>

</ItemsControl.ItemContainerStyle>

</ItemsControl>



# Customizing the Appearance of an Existing Control by Creating a ControlTemplate

A [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate) specifies the visual structure and visual behavior of a control. When you create a [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate), you replace the appearance of an existing control without changing its functionality.

You cannot apply a [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate) to a [UserControl](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.usercontrol).

## When You Should Create a ControlTemplate

You create a [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate) when you want to customize the control's appearance beyond what setting the other properties on the control will do.

## Changing the Visual Structure of a Control

A [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate) must have only one [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) as its root element. The root element usually contains other [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) objects

### Preserving the Functionality of a Control's Properties by Using TemplateBinding

The [TemplateBinding](https://docs.microsoft.com/zh-cn/dotnet/framework/wpf/advanced/templatebinding-markup-extension) markup extension binds a property of an element that is in the [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate) to a public property that is defined by the control.

<Grid Margin="4" Background="{TemplateBinding Background}">

<!--Use a ContentPresenter to display the Content of

the Button.-->

<ContentPresenter

HorizontalAlignment="{TemplateBinding HorizontalContentAlignment}"

VerticalAlignment="{TemplateBinding VerticalContentAlignment}"

Margin="4,5,4,4" />

</Grid>

[ContentPresenter](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.contentpresenter) does not have a property named HorizontalContentAlignment, but [Control.HorizontalContentAlignment](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.control.horizontalcontentalignment) can be bound to [FrameworkElement.HorizontalAlignment](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement.horizontalalignment). When you template bind a property, be sure that the target and source properties are the same type.

The [Control](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.control) class defines several properties that must be used by the control template to have an effect on the control when they are set. How the [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate) uses the property depends on the property. The [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate) must use the property in one of the following ways:

* An element in the [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate) template binds to the property.
* An element in the [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate) inherits the property from a parent [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement).

| **Property** | **Usage method** |
| --- | --- |
| [Background](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.control.background) | Template binding |
| [BorderThickness](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.control.borderthickness) | Template binding |
| [BorderBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.control.borderbrush) | Template binding |
| [FontFamily](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.control.fontfamily) | Property inheritance or template binding |
| [FontSize](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.control.fontsize) | Property inheritance or template binding |
| [FontStretch](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.control.fontstretch) | Property inheritance or template binding |
| [FontWeight](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.control.fontweight) | Property inheritance or template binding |
| [Foreground](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.control.foreground) | Property inheritance or template binding |
| [HorizontalContentAlignment](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.control.horizontalcontentalignment) | Template binding |
| [Padding](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.control.padding) | Template binding |
| [VerticalContentAlignment](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.control.verticalcontentalignment) | Template binding |

if the [ContentPresenter](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.contentpresenter) is in the [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate) of a [ContentControl](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.contentcontrol), the [ContentPresenter](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.contentpresenter) will automatically bind to the [ContentTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.contentcontrol.contenttemplate) and [Content](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.contentcontrol.content) properties.

Even though the [Foreground](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.control.foreground) and [FontSize](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.control.fontsize) properties are not template bound, setting them has an effect because their values are inherited.

<StackPanel>

<Button Style="{StaticResource newTemplate}"

Background="Navy" Foreground="White" FontSize="14"

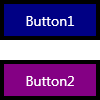
Content="Button1"/>

<Button Style="{StaticResource newTemplate}"

Background="Purple" Foreground="White" FontSize="14"

Content="Button2" HorizontalContentAlignment="Left"/>

</StackPanel>



## Changing the Appearance of a Control Depending on Its State

A visual behavior describes the control appearance when it is in a certain state

The button's functionality is to raise the [Click](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.primitives.buttonbase.click) event when it is clicked, but the button's visual behavior is to change its appearance when it is pointed to or pressed.

You use [VisualState](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.visualstate) objects to specify the appearance of a control when it is in a certain state. A [VisualState](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.visualstate) contains a [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard) that changes the appearance of the elements that are in the [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate).

the control's logic changes state by using the [VisualStateManager](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.visualstatemanager).

When the control enters the state that is specified by the [VisualState.Name](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.visualstate.name) property, the [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard) begins. When the control exits the state, the [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard) stops.

<!--Change the border of the button to red when the

mouse is over the button.-->

<VisualState x:Name="MouseOver">

<Storyboard>

<ColorAnimation Storyboard.TargetName="BorderBrush"

Storyboard.TargetProperty="Color"

To="Red" />

</Storyboard>

</VisualState>

The control is responsible for defining the states as part of its control contract

The following table lists the states that are specified for the [Button](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.button).

| **VisualState Name** | **VisualStateGroup Name** | **Description** |
| --- | --- | --- |
| Normal | CommonStates | The default state. |
| MouseOver | CommonStates | The mouse pointer is positioned over the control. |
| Pressed | CommonStates | The control is pressed. |
| Disabled | CommonStates | The control is disabled. |
| Focused | FocusStates | The control has focus. |
| Unfocused | FocusStates | The control does not have focus. |

States in the same state group are mutually exclusive.

You add [VisualState](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.visualstate) objects to [VisualStateGroup](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.visualstategroup) objects. You add [VisualStateGroup](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.visualstategroup) objects to the [VisualStateGroups](https://msdn.microsoft.com/library/system.windows.visualstatemanager.visualstategroups.aspx) attached property.

<ControlTemplate TargetType="Button">

<Border Name="RootElement">

<VisualStateManager.VisualStateGroups>

<!--Define the states and transitions for the common states.

The states in the VisualStateGroup are mutually exclusive to

each other.-->

<VisualStateGroup x:Name="CommonStates">

<!--The Normal state is the state the button is in

when it is not in another state from this VisualStateGroup.-->

<VisualState x:Name="Normal" />

<!--Change the SolidColorBrush, BorderBrush, to red when the

mouse is over the button.-->

<VisualState x:Name="MouseOver">

<Storyboard>

<ColorAnimation Storyboard.TargetName="BorderBrush"

Storyboard.TargetProperty="Color"

To="Red" />

</Storyboard>

</VisualState>

<!--Change the SolidColorBrush, BorderBrush, to Transparent when the

button is pressed.-->

<VisualState x:Name="Pressed">

<Storyboard>

<ColorAnimation Storyboard.TargetName="BorderBrush"

Storyboard.TargetProperty="Color"

To="Transparent"/>

</Storyboard>

</VisualState>

<!--The Disabled state is omitted for brevity.-->

</VisualStateGroup>

</VisualStateManager.VisualStateGroups>

<Border.Background>

<SolidColorBrush x:Name="BorderBrush" Color="Black"/>

</Border.Background>

<Grid Background="{TemplateBinding Background}" Margin="4">

<ContentPresenter

HorizontalAlignment="{TemplateBinding HorizontalContentAlignment}"

VerticalAlignment="{TemplateBinding VerticalContentAlignment}"

Margin="4,5,4,4" />

</Grid>

</Border>

</ControlTemplate>

# Data Binding Overview in WPF

## What is data binding?

Data binding is the process that establishes a connection between the application UI and the data it displays.

if you want to specify the source on your individual bindings explicitly, you have the following options

[Source](https://docs.microsoft.com/en-us/dotnet/api/system.windows.data.binding.source)

|  |  |
| --- | --- |
| [RelativeSource](https://docs.microsoft.com/en-us/dotnet/api/system.windows.data.binding.relativesource) |  |

[ElementName](https://docs.microsoft.com/en-us/dotnet/api/system.windows.data.binding.elementname)

### Specifying the path to the value

If your binding source is an object, you use the [Binding.Path](https://docs.microsoft.com/en-us/dotnet/api/system.windows.data.binding.path#System_Windows_Data_Binding_Path) property to specify the value to use for your binding. If you are binding to XML data, you use the [Binding.XPath](https://docs.microsoft.com/en-us/dotnet/api/system.windows.data.binding.xpath#System_Windows_Data_Binding_XPath) property to specify the value 。in some cases, it may be applicable to use the [Path](https://docs.microsoft.com/en-us/dotnet/api/system.windows.data.binding.path) property even when your data is XML.

### Binding and BindingExpression

The [Binding](https://docs.microsoft.com/en-us/dotnet/api/system.windows.data.binding?view=netframework-4.8) class is the high-level class for the declaration of a binding. The [BindingExpression](https://docs.microsoft.com/en-us/dotnet/api/system.windows.data.bindingexpression?view=netframework-4.8) class is the underlying object that maintains the connection between the binding source and the binding target. A [Binding](https://docs.microsoft.com/en-us/dotnet/api/system.windows.data.binding?view=netframework-4.8) contains all the information that can be shared across several [BindingExpression](https://docs.microsoft.com/en-us/dotnet/api/system.windows.data.bindingexpression?view=netframework-4.8) objects. A [BindingExpression](https://docs.microsoft.com/en-us/dotnet/api/system.windows.data.bindingexpression?view=netframework-4.8) is an instance expression that cannot be shared and that contains all the instance information about the [Binding](https://docs.microsoft.com/en-us/dotnet/api/system.windows.data.binding?view=netframework-4.8).

## Binding to collections

However, to set up dynamic bindings so that insertions or deletions in the collection update the UI automatically, the collection must implement the [INotifyCollectionChanged](https://docs.microsoft.com/en-us/dotnet/api/system.collections.specialized.inotifycollectionchanged) interface.

WPF provides the [ObservableCollection<T>](https://docs.microsoft.com/en-us/dotnet/api/system.collections.objectmodel.observablecollection-1) class, which is a built-in implementation of a data collection that exposes the [INotifyCollectionChanged](https://docs.microsoft.com/en-us/dotnet/api/system.collections.specialized.inotifycollectionchanged) interface

### Collection views

#### What Are collection views?

A collection view is a layer on top of a binding source collection that allows you to navigate and display the source collection based on sort, filter, and group queries, without having to change the underlying source collection itself. A collection view also maintains a pointer to the current item in the collection. If the source collection implements the [INotifyCollectionChanged](https://docs.microsoft.com/en-us/dotnet/api/system.collections.specialized.inotifycollectionchanged) interface, the changes raised by the [CollectionChanged](https://docs.microsoft.com/en-us/dotnet/api/system.collections.specialized.inotifycollectionchanged.collectionchanged) event are propagated to the views.

#### Using a default view

WPF also creates a default collection view for every collection used as a binding source. If you bind directly to a collection, WPF binds to its default view

To improve performance, collection views for ADO.NET [DataTable](https://docs.microsoft.com/en-us/dotnet/api/system.data.datatable) or [DataView](https://docs.microsoft.com/en-us/dotnet/api/system.data.dataview) objects delegate sorting and filtering to the [DataView](https://docs.microsoft.com/en-us/dotnet/api/system.data.dataview), which causes sorting and filtering to be shared across all collection views of the data source.

#### Current item pointers

Because WPF binds to a collection only by using a view (either a view you specify, or the collection's default view), all bindings to collections have a current item pointer. When binding to a view, the slash ("/") character in a Path value designates the current item of the view

#### Master-Detail binding scenario

The notion of a current item is useful not only for navigation of items in a collection, but also for the master-detail binding scenario

when a singleton object (the [ContentControl](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.contentcontrol) in this case) is bound to a collection view, it automatically binds to the [CurrentItem](https://docs.microsoft.com/en-us/dotnet/api/system.windows.data.collectionview.currentitem) of the view.

### Providing visual feedback

One way to provide such feedback is to set the [Validation.ErrorTemplate](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.validation.errortemplate) attached property to a custom [ControlTemplate](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.controltemplate).

<ControlTemplate x:Key="validationTemplate">

<DockPanel>

<TextBlock Foreground="Red" FontSize="20">!</TextBlock>

<AdornedElementPlaceholder/>

</DockPanel>

</ControlTemplate>

The [AdornedElementPlaceholder](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.adornedelementplaceholder) element specifies where the control being adorned should be placed.

If your [Binding](https://docs.microsoft.com/en-us/dotnet/api/system.windows.data.binding) has associated validation rules but you do not specify an [ErrorTemplate](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.validation.errortemplate) on the bound control, a default [ErrorTemplate](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.validation.errortemplate) will be used to notify users when there is a validation error

### Validation process

Validation usually occurs when the value of a target is transferred to the binding source property. This transfer occurs on [TwoWay](https://docs.microsoft.com/en-us/dotnet/api/system.windows.data.bindingmode#System_Windows_Data_BindingMode_TwoWay) and [OneWayToSource](https://docs.microsoft.com/en-us/dotnet/api/system.windows.data.bindingmode#System_Windows_Data_BindingMode_OneWayToSource) bindings

The following items describe the *validation* process. If a validation error or other type of error occurs at any time during this process, the process is halted.

1 The binding engine checks if there are any custom [ValidationRule](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.validationrule) objects defined whose [ValidationStep](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.validationrule.validationstep) is set to [RawProposedValue](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.validationstep#System_Windows_Controls_ValidationStep_RawProposedValue) for that [Binding](https://docs.microsoft.com/en-us/dotnet/api/system.windows.data.binding),

2 The binding engine then calls the converter, if one exists.

3 binding engine checks if there are any custom [ValidationRule](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.validationrule) objects defined whose [ValidationStep](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.validationrule.validationstep) is set to [ConvertedProposedValue](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.validationstep#System_Windows_Controls_ValidationStep_ConvertedProposedValue) for that [Binding](https://docs.microsoft.com/en-us/dotnet/api/system.windows.data.binding),

4 The binding engine sets the source property.

5 The binding engine checks if there are any custom [ValidationRule](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.validationrule) objects defined whose [ValidationStep](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.validationrule.validationstep) is set to [UpdatedValue](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.validationstep#System_Windows_Controls_ValidationStep_UpdatedValue) for that [Binding](https://docs.microsoft.com/en-us/dotnet/api/system.windows.data.binding), If a [DataErrorValidationRule](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.dataerrorvalidationrule) is associated with a binding and its [ValidationStep](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.validationrule.validationstep) is set to the default, [UpdatedValue](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.validationstep#System_Windows_Controls_ValidationStep_UpdatedValue), the [DataErrorValidationRule](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.dataerrorvalidationrule) is checked at this point. At this point any binding that has the [ValidatesOnDataErrors](https://docs.microsoft.com/en-us/dotnet/api/system.windows.data.binding.validatesondataerrors) set to true is checked

6 The binding engine checks if there are any custom [ValidationRule](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.validationrule) objects defined whose [ValidationStep](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.validationrule.validationstep) is set to [CommittedValue](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.validationstep#System_Windows_Controls_ValidationStep_CommittedValue) for that [Binding](https://docs.microsoft.com/en-us/dotnet/api/system.windows.data.binding)

If a [ValidationRule](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.validationrule) does not pass at any time throughout this process, the binding engine creates a [ValidationError](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.validationerror) object and adds it to the [Errors](https://msdn.microsoft.com/library/system.windows.controls.validation.errors.aspx) collection of the bound element.

When [Errors](https://msdn.microsoft.com/library/system.windows.controls.validation.errors.aspx) is not empty, the [HasError](https://msdn.microsoft.com/library/system.windows.controls.validation.haserror) attached property of the element is set to true. Also, if the [NotifyOnValidationError](https://docs.microsoft.com/en-us/dotnet/api/system.windows.data.binding.notifyonvalidationerror) property of the [Binding](https://docs.microsoft.com/en-us/dotnet/api/system.windows.data.binding) is set to true, then the binding engine raises the [Validation.Error](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.validation.error) attached event on the element.

# Walkthrough: Caching Application Data in a WPF Application

<https://docs.microsoft.com/zh-cn/dotnet/framework/wpf/advanced/walkthrough-caching-application-data-in-a-wpf-application>

# INotifyPropertyChanged Interface

<https://docs.microsoft.com/zh-cn/dotnet/api/system.componentmodel.inotifypropertychanged?view=netframework-4.8>

如果使用 CallerMemberName 特性，则调用 NotifyPropertyChanged 方法不必将属性名称指定为字符串参数。 有关详细信息, 请参阅[调用方信息](https://msdn.microsoft.com/library/9cb2b8c0-c4f6-44b8-9c90-38948455b373)。

# Threading Model

As a result, the majority of WPF developers won't have to write an interface that uses more than one thread. Because multithreaded programs are complex and difficult to debug

This topic discusses threading by using the [BeginInvoke](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.threading.dispatcher.begininvoke) method for asynchronous calls. You can also make asynchronous calls by calling the [InvokeAsync](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.threading.dispatcher.invokeasync) method, which take an [Action](https://docs.microsoft.com/zh-cn/dotnet/api/system.action) or [Func<TResult>](https://docs.microsoft.com/zh-cn/dotnet/api/system.func-1) as a parameter

## Overview and the Dispatcher

The UI thread queues work items inside an object called a [Dispatcher](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.threading.dispatcher). The [Dispatcher](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.threading.dispatcher) selects work items on a priority basis and runs each one to completion. Every UI thread must have at least one [Dispatcher](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.threading.dispatcher), and each [Dispatcher](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.threading.dispatcher) can execute work items in exactly one thread.

Most classes in WPF derive from [DispatcherObject](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.threading.dispatcherobject). At construction, a [DispatcherObject](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.threading.dispatcherobject) stores a reference to the [Dispatcher](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.threading.dispatcher) linked to the currently running thread.

A background thread can ask the UI thread to perform an operation on its behalf. It does this by registering a work item with the [Dispatcher](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.threading.dispatcher) of the UI thread. The [Dispatcher](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.threading.dispatcher) class provides two methods for registering work items: [Invoke](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.threading.dispatcher.invoke) and [BeginInvoke](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.threading.dispatcher.begininvoke). Both methods schedule a delegate for execution.

The [Dispatcher](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.threading.dispatcher) orders the elements in its queue by priority.

## Threads in Action: The Samples

### A Single-Threaded Application with a Long-Running Calculation

Most graphical user interfaces (GUIs) spend a large portion of their time idle while waiting for events that are generated in response to user interactions。The WPF threading model doesn’t allow input to interrupt an operation happening in the UI thread。This means you must be sure to return to the [Dispatcher](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.threading.dispatcher) periodically to process pending input events before they get stale.

If we break up the task of calculation into manageable chunks, we can periodically return to the [Dispatcher](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.threading.dispatcher) and process events. We can give WPF an opportunity to repaint and process input.

By using the [BeginInvoke](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.threading.dispatcher.begininvoke) method, we can schedule prime number checks in the same queue that UI events are drawn from.

In this case, we choose the [SystemIdle](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.threading.dispatcherpriority#System_Windows_Threading_DispatcherPriority_SystemIdle) priority. The [Dispatcher](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.threading.dispatcher) will execute this delegate only when there are no important events to process.

### Handling a Blocking Operation with a Background Thread

We can use [Invoke](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.threading.dispatcher.invoke) or [BeginInvoke](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.threading.dispatcher.begininvoke) to insert delegates into the [Dispatcher](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.threading.dispatcher) of the UI thread. Eventually, these delegates will be executed with permission to modify UI elements

When the button is clicked, we display the clock drawing and start animating it. We disable the button. We invoke the FetchWeatherFromServer method in a new thread, and then we return,

private void ForecastButtonHandler(object sender, RoutedEventArgs e)

{

// Change the status image and start the rotation animation.

fetchButton.IsEnabled = false;

fetchButton.Content = "Contacting Server";

weatherText.Text = "";

hideWeatherImageStoryboard.Begin(this);

// Start fetching the weather forecast asynchronously.

NoArgDelegate fetcher = new NoArgDelegate(

this.FetchWeatherFromServer);

fetcher.BeginInvoke(null, null);

}

### Multiple Windows, Multiple Threads

Windows Explorer works in this fashion. Each new Explorer window belongs to the original process, but it is created under the control of an independent thread.

When the user clicks the "new window" button, we launch a copy of our window in a separate thread. This way, long-running or blocking operations in one of the windows won’t lock all the other windows.

private void ThreadStartingPoint()

{

Window1 tempWindow = new Window1();

tempWindow.Show();

System.Windows.Threading.Dispatcher.Run();

}

WPF automatically creates a new [Dispatcher](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.threading.dispatcher) to manage the new thread. All we have to do to make the window functional is to start the [Dispatcher](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.threading.dispatcher).

# Using Delegates (C# Programming Guide)

https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/delegates/using-delegates

A [delegate](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/delegate) is a type that safely encapsulates a method, similar to a function pointer in C and C++. Unlike C function pointers, delegates are object-oriented, type safe, and secure. The type of a delegate is defined by the name of the delegate. The following example declares a delegate named Del that can encapsulate a method that takes a [string](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/string) as an argument and returns [void](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/void):

public delegate void Del(string message);

The [Delegate](https://docs.microsoft.com/en-us/dotnet/api/system.delegate?view=netframework-4.8) class is the base class for delegate types. However, only the system and compilers can derive explicitly from the [Delegate](https://docs.microsoft.com/en-us/dotnet/api/system.delegate?view=netframework-4.8) class or from the [MulticastDelegate](https://docs.microsoft.com/en-us/dotnet/api/system.multicastdelegate?view=netframework-4.8) class. The [Delegate](https://docs.microsoft.com/en-us/dotnet/api/system.delegate?view=netframework-4.8) class is not considered a delegate type; it is a class used to derive delegate types.

The common language runtime provides an Invoke method for each delegate type, with the same signature as the delegate. The Invoke method is useful in [reflection](https://docs.microsoft.com/en-us/dotnet/framework/reflection-and-codedom/reflection?view=netframework-4.8) when you want to find the signature of the delegate type.

The common language runtime provides each delegate type with BeginInvoke and EndInvoke methods, to enable asynchronous invocation of the delegate.

Once a delegate is instantiated, a method call made to the delegate will be passed by the delegate to that method

When a delegate is constructed to wrap an instance method, the delegate references both the instance and the method. A delegate has no knowledge of the instance type aside from the method it wraps,

When a delegate is constructed to wrap a static method, it only references the method.

A delegate can call more than one method when invoked. This is referred to as multicasting

MethodClass obj = new MethodClass();

Del d1 = obj.Method1;

Del d2 = obj.Method2;

Del d3 = DelegateMethod;

//Both types of assignment are valid.

Del allMethodsDelegate = d1 + d2;

allMethodsDelegate += d3;

At this point allMethodsDelegate contains three methods in its invocation list—Method1, Method2, and DelegateMethod. The original three delegates, d1, d2, and d3, remain unchanged. When allMethodsDelegate is invoked, all three methods are called in order

If the delegate has a return value and/or out parameters, it returns the return value and parameters of the last method invoked.

Delegates with more than one method in their invocation list derive from [MulticastDelegate](https://docs.microsoft.com/en-us/dotnet/api/system.multicastdelegate), which is a subclass of System.Delegate

To register for an event, the recipient creates a method designed to handle the event, then creates a delegate for that method and passes the delegate to the event source. The source calls the delegate when the event occurs. The delegate then calls the event handling method on the recipient, delivering the event data.

# Delegates with Named vs. Anonymous Methods (C# Programming Guide)

A [delegate](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/delegate) can be associated with a named method. When you instantiate a delegate by using a named method, the method is passed as a parameter, for example:

// Declare a delegate:

delegate void Del(int x);

// Define a named method:

void DoWork(int k) { /\* ... \*/ }

// Instantiate the delegate using the method as a parameter:

Del d = obj.DoWork;

C# enables you to instantiate a delegate and immediately specify a code block that the delegate will process when it is called. The block can contain either a lambda expression or an anonymous method

# Anonymous functions (C# Programming Guide)

An anonymous function is an "inline" statement or expression that can be used wherever a delegate type is expected. You can use it to initialize a named delegate or pass it instead of a named delegate type as a method parameter.

You can use a [lambda expression](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/statements-expressions-operators/lambda-expressions) or an [anonymous method](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/operators/delegate-operator) to create an anonymous function

// Original delegate syntax required

// initialization with a named method.

TestDelegate testDelA = new TestDelegate(M);

// C# 2.0 provides a simpler way to declare an instance of Del.

Del del2 = Notify;

// C# 2.0: A delegate can be initialized with

// inline code, called an "anonymous method." This

// method takes a string as an input parameter.

TestDelegate testDelB = delegate(string s) { Console.WriteLine(s); };

// C# 3.0. A delegate can be initialized with

// a lambda expression. The lambda also takes a string

// as an input parameter (x). The type of x is inferred by the compiler.

TestDelegate testDelC = (x) => { Console.WriteLine(x); };

# Calling Synchronous Methods Asynchronously

https://docs.microsoft.com/en-us/dotnet/standard/asynchronous-programming-patterns/calling-synchronous-methods-asynchronously?view=netframework-4.8

The .NET Framework enables you to call any method asynchronously. To do this you define a delegate with the same signature as the method you want to call; the common language runtime automatically defines BeginInvoke and EndInvoke methods for this delegate, with the appropriate signatures.

The BeginInvoke method initiates the asynchronous call. It has the same parameters as the method that you want to execute asynchronously, plus two additional optional parameters. The first parameter is an [AsyncCallback](https://docs.microsoft.com/en-us/dotnet/api/system.asynccallback) delegate that references a method to be called when the asynchronous call completes. The second parameter is a user-defined object that passes information into the callback method

BeginInvoke returns an [IAsyncResult](https://docs.microsoft.com/en-us/dotnet/api/system.iasyncresult), which can be used to monitor the progress of the asynchronous call.

The parameters of EndInvoke include the out and ref parameters of the method that you want to execute asynchronously, plus the [IAsyncResult](https://docs.microsoft.com/en-us/dotnet/api/system.iasyncresult) returned by BeginInvoke.

The code examples in this topic demonstrate four common ways to use BeginInvoke and EndInvoke to make asynchronous calls. After calling BeginInvoke you can do the following:

 Do some work and then call EndInvoke to block until the call completes.

 Obtain a [WaitHandle](https://docs.microsoft.com/en-us/dotnet/api/system.threading.waithandle) using the [IAsyncResult.AsyncWaitHandle](https://docs.microsoft.com/en-us/dotnet/api/system.iasyncresult.asyncwaithandle) property, use its [WaitOne](https://docs.microsoft.com/en-us/dotnet/api/system.threading.waithandle.waitone) method to block execution until the [WaitHandle](https://docs.microsoft.com/en-us/dotnet/api/system.threading.waithandle) is signaled, and then call EndInvoke.

 Poll the [IAsyncResult](https://docs.microsoft.com/en-us/dotnet/api/system.iasyncresult) returned by BeginInvoke to determine when the asynchronous call has completed, and then call EndInvoke.

 Pass a delegate for a callback method to BeginInvoke. The method is executed on a [ThreadPool](https://docs.microsoft.com/en-us/dotnet/api/system.threading.threadpool) thread when the asynchronous call completes. The callback method calls EndInvoke.

// The delegate must have the same signature as the method

// it will call asynchronously.

public delegate string AsyncMethodCaller(int callDuration, out int threadId);

// Create the delegate.

AsyncMethodCaller caller = new AsyncMethodCaller(ad.TestMethod);

// Initiate the asychronous call.

IAsyncResult result = caller.BeginInvoke(3000,

out threadId, null, null);

// Call EndInvoke to wait for the asynchronous call to complete,

// and to retrieve the results.

string returnValue = caller.EndInvoke(out threadId, result);

// Initiate the asychronous call.

IAsyncResult result = caller.BeginInvoke(3000,

out threadId, null, null);

Thread.Sleep(0);

Console.WriteLine("Main thread {0} does some work.",

Thread.CurrentThread.ManagedThreadId);

// Wait for the WaitHandle to become signaled.

result.AsyncWaitHandle.WaitOne();

// Perform additional processing here.

// Call EndInvoke to retrieve the results.

string returnValue = caller.EndInvoke(out threadId, result);

// Close the wait handle.

result.AsyncWaitHandle.Close();

You can use the [IsCompleted](https://docs.microsoft.com/en-us/dotnet/api/system.iasyncresult.iscompleted) property of the [IAsyncResult](https://docs.microsoft.com/en-us/dotnet/api/system.iasyncresult) returned by BeginInvoke to discover when the asynchronous call completes.

// Initiate the asychronous call.

IAsyncResult result = caller.BeginInvoke(3000,

out threadId, null, null);

// Poll while simulating work.

while(result.IsCompleted == false) {

Thread.Sleep(250);

Console.Write(".");

}

// Call EndInvoke to retrieve the results.

string returnValue = caller.EndInvoke(out threadId, result);

If the thread that initiates the asynchronous call does not need to be the thread that processes the results, you can execute a callback method when the call completes. The callback method is executed on a [ThreadPool](https://docs.microsoft.com/en-us/dotnet/api/system.threading.threadpool) thread.

In the callback method, you can cast the [IAsyncResult](https://docs.microsoft.com/en-us/dotnet/api/system.iasyncresult), which is the only parameter of the callback method, to an [AsyncResult](https://docs.microsoft.com/en-us/dotnet/api/system.runtime.remoting.messaging.asyncresult) object.

IAsyncResult result = caller.BeginInvoke(3000,

out dummy,

new AsyncCallback(CallbackMethod),

"The call executed on thread {0}, with return value \"{1}\".");

// The callback method must have the same signature as the

// AsyncCallback delegate.

static void CallbackMethod(IAsyncResult ar)

{

// Retrieve the delegate.

AsyncResult result = (AsyncResult) ar;

AsyncMethodCaller caller = (AsyncMethodCaller) result.AsyncDelegate;

// Retrieve the format string that was passed as state

// information.

string formatString = (string) ar.AsyncState;

// Define a variable to receive the value of the out parameter.

// If the parameter were ref rather than out then it would have to

// be a class-level field so it could also be passed to BeginInvoke.

int threadId = 0;

// Call EndInvoke to retrieve the results.

string returnValue = caller.EndInvoke(out threadId, ar);

// Use the format string to format the output message.

Console.WriteLine(formatString, threadId, returnValue);

}

### Configuring the Application Definition for MSBuild

WPF generates this code for you when the markup file of your application definition is configured as an MSBuild ApplicationDefinition item

## Getting the Current Application

the [Application](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.application) class is implemented as a singleton class which creates a single instance of itself and provides shared access to it with the static [Current](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.application.current) property.

## Application Lifetime

After [Run](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.application.run) is called and the application is initialized, the application is ready to run. This moment is signified when the [Startup](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.application.startup) event is raised:

<Application

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

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

x:Class="SDKSample.App"

Startup="App\_Startup" />

void App\_Startup(object sender, StartupEventArgs e)

{

// Open a window

MainWindow window = new MainWindow();

window.Show();

}

You need to handle the [Startup](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.application.startup) event to open a [Window](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.window) if you need to instantiate it using a non-parameterless constructor, or you need to set its properties or subscribe to its events before showing it, or you need to process any command-line arguments that were supplied when the application was launched.

During application initialization, WPF retrieves the command-line arguments from the operating system and passes them to the [Startup](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.application.startup) event handler via the [Args](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.startupeventargs.args) property of the [StartupEventArgs](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.startupeventargs) parameter.

### Application Activation and Deactivation

An application becomes the active application in the following circumstances:

* It is launched and shows a [Window](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.window).
* A user switches from another application by selecting a [Window](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.window) in the application.

Likewise, an application can become inactive in the following circumstances:

* A user switches to another application from the current one.
* When the application shuts down.

#### Session Ending

The shutdown conditions that are described by the [ShutdownMode](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.application.shutdownmode) property are specific to an application. In some cases, though, an application may shut down as a result of an external condition. The most common external condition occurs when a user ends the Windows session by the following actions:

* Logging off
* Shutting down
* Restarting
* Hibernating

To detect when a Windows session ends, you can handle the [SessionEnding](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.application.sessionending) event, as illustrated in the following example.

void App\_SessionEnding(object sender, SessionEndingCancelEventArgs e)

{

// Ask the user if they want to allow the session to end

string msg = string.Format("{0}. End session?", e.ReasonSessionEnding);

MessageBoxResult result = MessageBox.Show(msg, "Session Ending", MessageBoxButton.YesNo);

// End session, if specified

if (result == MessageBoxResult.No)

{

e.Cancel = true;

}

}

If the user does not want the session to end, the code sets [Cancel](https://docs.microsoft.com/zh-cn/dotnet/api/system.componentmodel.canceleventargs.cancel) to true to prevent the Windows session from ending.

#### Exit

When an application shuts down, it may need to perform some final processing, such as persisting application state. For these situations, you can handle the [Exit](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.application.exit) event,

#### Exit Code

Applications are mostly launched by the operating system in response to a user request. However, an application can be launched by another application to perform some specific task. When the launched application shuts down, the launching application may want to know the condition under which the launched application shut down. In these situations, Windows allows applications to return an application exit code on shutdown. By default, WPF applications return an exit code value of 0.

To change the exit code, you can call the [Shutdown(Int32)](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.application.shutdown#System_Windows_Application_Shutdown_System_Int32_) overload, which accepts an integer argument to be the exit code:

// Shutdown and return a non-default exit code

Application.Current.Shutdown(-1);

You can detect the value of the exit code, and change it, by handling the [Exit](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.application.exit) event. The [Exit](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.application.exit) event handler is passed an [ExitEventArgs](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.exiteventargs) which provides access to the exit code with the [ApplicationExitCode](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.exiteventargs.applicationexitcode) property.

# WPF Application Resource, Content, and Data Files

Microsoft Windowsapplications often depend on files that contain non-executable data, such as Extensible Application Markup Language (XAML), images, video, and audio.

This support revolves around a specific set of application data file types, including:

* **Resource Files**: Data files that are compiled into either an executable or library WPF assembly.
* **Content Files**: Standalone data files that have an explicit association with an executable WPF assembly.
* **Site of Origin Files**: Standalone data files that have no association with an executable WPF assembly.

One important distinction to make between these three types of files is that resource files and content files are known at build time; an assembly has explicit knowledge of them

## Resource Files

If an application data file must always be available to an application, the only way to guarantee availability is to compile it into an application's main executable assembly or one of its referenced assemblies. This type of application data file is known as a resource file.

You should use resource files when:

* You don't need to update the resource file's content after it is compiled into an assembly.
* You want to simplify application distribution complexity by reducing the number of file dependencies.
* Your application data file needs to be localizable (see [WPF Globalization and Localization Overview](https://docs.microsoft.com/zh-cn/dotnet/framework/wpf/advanced/wpf-globalization-and-localization-overview)).

### Configuring Resource Files

In WPF, a resource file is a file that is included in an Microsoft build engine (MSBuild) project as a Resource item.

<Project xmlns="http://schemas.microsoft.com/developer/msbuild/2003" ... >

...

<ItemGroup>

<Resource Include="ResourceFile.xaml" />

</ItemGroup>

...

</Project>

### Using Resource Files

To load a resource file, you can call the [GetResourceStream](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.application.getresourcestream) method of the [Application](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.application) class, passing a pack URI that identifies the desired resource file. [GetResourceStream](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.application.getresourcestream) returns a [StreamResourceInfo](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.resources.streamresourceinfo) object, which exposes the resource file as a [Stream](https://docs.microsoft.com/zh-cn/dotnet/api/system.io.stream) and describes its content type.

// Navigate to xaml page

Uri uri = new Uri("/PageResourceFile.xaml", UriKind.Relative);

StreamResourceInfo info = Application.GetResourceStream(uri);

System.Windows.Markup.XamlReader reader = new System.Windows.Markup.XamlReader();

Page page = (Page)reader.LoadAsync(info.Stream);

this.pageFrame.Content = page;

you can let WPF take care of opening and converting the [Stream](https://docs.microsoft.com/zh-cn/dotnet/api/system.io.stream) by loading a resource file directly into the property of a type using code or xaml

Uri pageUri = new Uri("/PageResourceFile.xaml", UriKind.Relative);

this.pageFrame.Source = pageUri;

<Frame Name="pageFrame" Source="PageResourceFile.xaml" />

### Application Code Files as Resource Files

you can set the [Application.StartupUri](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.application.startupuri) property with a pack URI that references the window or page that you would like to load when an application starts.

<Application

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

StartupUri="SOOPage.xaml" />

You can do this when a XAML file is included in an MSBuild project as a Page item.

<Project xmlns="http://schemas.microsoft.com/developer/msbuild/2003" ... >

...

<ItemGroup>

<Page Include="MainWindow.xaml" />

</ItemGroup>

...

</Project>

## Content Files

A content file is distributed as a loose file alongside an executable assembly. Although they are not compiled into an assembly, assemblies are compiled with metadata that establishes an association with each content file.

You should use content files when your application requires a specific set of application data files that you want to be able to update without recompiling the assembly that consumes them

### Configuring Content Files

To add a content file to a project, an application data file must be included as a Content item. Furthermore, you need to set the MSBuild CopyToOutputDirectory metadata element to specify that the content file is copied to a location that is relative to the built assembly. I

<ItemGroup>

<Content Include="ContentFile.xaml">

<CopyToOutputDirectory>PreserveNewest</CopyToOutputDirectory>

</Content>

</ItemGroup>

### Using Content Files

To load a content file, you can call the [GetContentStream](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.application.getcontentstream) method of the [Application](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.application) class, passing a pack URI that identifies the desired content file.

you can let WPF take care of opening and converting the [Stream](https://docs.microsoft.com/zh-cn/dotnet/api/system.io.stream) by loading a resource file directly into the property of a type using code.

## Site of Origin Files

there are times when you may want to establish either an implicit or non-existent relationship between an assembly and an application data file, including when:

* A file doesn't exist at compile time.
* You don't know what files your assembly will require until run time.
* You want to be able to update files without recompiling the assembly that they are associated with.
* Your application uses large data files, such as audio and video, and you only want users to download them if they choose to.

It is possible to load these types of files by using traditional URI schemes, such as the file:/// and http:// schemes.

<Image Source="file:///C:/DataFile.bmp" />

<Image Source="http://www.datafilewebsite.com/DataFile.bmp" />

However, the file:/// and http:// schemes require your application to have full trust.

### Configuring Site of Origin Files

If your site of origin files are non-existent or unknown at compile time, you need to use traditional deployment mechanisms for ensuring the required files are available at run time, including using either the XCopy command-line program or the Microsoft Windows Installer.

If you do know at compile time the files that you would like to be located at the site of origin, but still want to avoid an explicit dependency, you can add those files to an MSBuild project as None item.

<Project xmlns="http://schemas.microsoft.com/developer/msbuild/2003" ... >

...

<None Include="PageSiteOfOriginFile.xaml">

<CopyToOutputDirectory>Always</CopyToOutputDirectory>

</None>

...

</Project>

### Using Site of Origin Files

To load a site of origin file, you can call the [GetRemoteStream](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.application.getremotestream) method of the [Application](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.application) class, passing a pack URI that identifies the desired site of origin file.

you can let WPF take care of opening and converting the [Stream](https://docs.microsoft.com/zh-cn/dotnet/api/system.io.stream) by loading a resource file directly into the property of a type using code.

# Pack URIs in WPF

<https://docs.microsoft.com/zh-cn/dotnet/framework/wpf/app-development/pack-uris-in-wpf>

In Windows Presentation Foundation (WPF), uniform resource identifiers (URIs) are used to identify and load files in many ways, including the following:

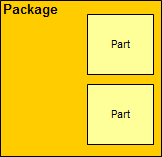
* Specifying the user interface (UI) to show when an application first starts.
* Loading images.
* Navigating to pages.
* Loading non-executable data files.

Furthermore, URIs can be used to identify and load files from a variety of locations, including the following:

* The current assembly.
* A referenced assembly.
* A location relative to an assembly.
* The application's site of origin.

## The Pack URI Scheme

The pack URI scheme is used by the [Open Packaging Conventions](https://go.microsoft.com/fwlink/?LinkID=71255) (OPC) specification, which describes a model for organizing and identifying content. The key elements of this model are packages and parts, where a package is a logical container for one or more logical parts. The following figure illustrates this concept.

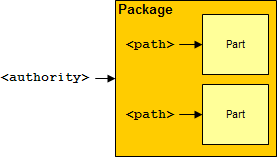


The pack URI scheme uses "pack" as its scheme, and contains two components: authority and path. The following is the format for a pack URI.

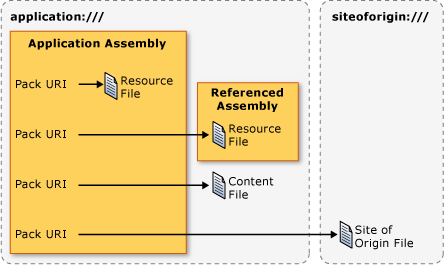
pack://*authority*/*path*

The *authority* specifies the type of package that a part is contained by, while the *path* specifies the location of a part within a package.

This concept is illustrated by the following figure:



Packages and parts are analogous to applications and files, where an application (package) can include one or more files (parts), including:

* Resource files that are compiled into the local assembly.
* Resource files that are compiled into a referenced assembly.
* Resource files that are compiled into a referencing assembly.
* Content files.
* Site of origin files.
* To access these types of files, WPF supports two authorities: application:/// and siteoforigin:///. The application:/// authority identifies application data files that are known at compile time, including resource and content files. The siteoforigin:/// authority identifies site of origin files. The scope of each authority is shown in the following figure.
* 

### Local Assembly Resource File

The pack URI for a resource file that is compiled into the local assembly uses the following authority and path:

* **Authority**: application:///.
* **Path**: The name of the resource file, including its path, relative to the local assembly project folder root.

The following example shows the pack URI for a XAML resource file that is located in the root of the local assembly's project folder.

pack://application:,,,/ResourceFile.xaml

 **Authority**: application:///.

 **Path**: The name of a resource file that is compiled into a referenced assembly. The path must conform to the following format:

*AssemblyShortName*{*;Version*]{*;PublicKey*];component/*Path*

* + **/Path**: the name of the resource file, including its path, relative to the root of the referenced assembly's project folder.

The following example shows the pack URI for a XAML resource file that is located in the root of the referenced assembly's project folder.

pack://application:,,,/ReferencedAssembly;component/ResourceFile.xaml

## Content File Pack URIs

The pack URI for a content file uses the following authority and path:

* **Authority**: application:///.
* **Path**: The name of the content file, including its path relative to the file system location of the application's main executable assembly.

The following example shows the pack URI for a XAML content file, located in the same folder as the executable assembly.

pack://application:,,,/ContentFile.xaml

## Site of Origin Pack URIs

The pack URI for a site of origin file uses the following authority and path:

* **Authority**: siteoforigin:///.
* **Path**: The name of the site of origin file, including its path relative to the location from which the executable assembly was launched.

The following example shows the pack URI for a XAML site of origin file, stored in the location from which the executable assembly is launched.

pack://siteoforigin:,,,/SiteOfOriginFile.xaml

## Page Files

XAML files that are configured as MSBuild Page items are compiled into assemblies in the same way as resource files. Consequently, MSBuild Page items can be identified using pack URIs for resource files.

## Absolute vs. Relative Pack URIs

A fully qualified pack URI includes the scheme, the authority, and the path, and it is considered an absolute pack URI. As a simplification for developers, XAML elements typically allow you to set appropriate attributes with a relative pack URI, which includes only the path.

For example, consider the following absolute pack URI for a resource file in the local assembly.

pack://application:,,,/ResourceFile.xaml

The relative pack URI that refers to this resource file would be the following.

/ResourceFile.xaml

By default, a relative pack URI is considered relative to the location of the markup or code that contains the reference。If a leading backslash is used, however, the relative pack URI reference is then considered relative to the root of the application

## Pack URI Resolution

The format of pack URIs makes it possible for pack URIs for different types of files to look the same. For example, consider the following absolute pack URI.

pack://application:,,,/ResourceOrContentFile.xaml

This absolute pack URI could refer to either a resource file in the local assembly or a content file

WPF resolves URIs for resource files in local assemblies and content files by using the following heuristics:

the path of the pack URI refers to a content file.?

the path of the pack URI refers to a resource file.?

### Using Pack URIs in Code

You specify a pack URI in code by instantiating the [Uri](https://docs.microsoft.com/zh-cn/dotnet/api/system.uri) class and passing the pack URI as a parameter to the constructor. This is demonstrated in the following example.

Uri uri = new Uri("pack://application:,,,/File.xaml");

By default, the [Uri](https://docs.microsoft.com/zh-cn/dotnet/api/system.uri) class considers pack URIs to be absolute. Fortunately, the [Uri(String, UriKind)](https://docs.microsoft.com/zh-cn/dotnet/api/system.uri.-ctor#System_Uri__ctor_System_String_System_UriKind_) overload of the [Uri](https://docs.microsoft.com/zh-cn/dotnet/api/system.uri) class constructor accepts a parameter of type [UriKind](https://docs.microsoft.com/zh-cn/dotnet/api/system.urikind) to allow you to specify whether a pack URI is either absolute or relative.

// Absolute URI (default)

Uri absoluteUri = new Uri("pack://application:,,,/File.xaml", UriKind.Absolute);

// Relative URI

Uri relativeUri = new Uri("/File.xaml",

UriKind.Relative);

## Alternatives to Writing a New Control

WPF enables to you customize existing controls by using its rich content model, styles, templates, and triggers. The following list gives examples of how these features can be used to create custom and consistent experiences without having to create a new control.

**Rich Content.** Many of the standard WPF controls support rich content. For example, the content property of a [Button](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.button) is of type [Object](https://docs.microsoft.com/zh-cn/dotnet/api/system.object), there is less need to create a new control or to modify an existing control to support a complex visualization.

**Styles.** A [Style](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.style) is a collection of values that represent properties for a control. By using styles, you can create a reusable representation of a desired control appearance and behavior without writing a new control.

**Data Templates.**

**Control Templates**

**Triggers**

## Models for Control Authoring

WPF provides three general models for creating a control, each of which provides a different set of features and level of flexibility. The base classes for the three models are [UserControl](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.usercontrol), [Control](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.control), and [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement).

### Deriving from UserControl

The simplest way to create a control in WPF is to derive from [UserControl](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.usercontrol).

if your control inherits from [UserControl](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.usercontrol), people who use your control will not be able to use a [DataTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.datatemplate) or [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate) to customize its appearance

#### Benefits of Deriving from UserControl

Consider deriving from [UserControl](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.usercontrol) if all of the following apply:

* You want to build your control similarly to how you build an application.
* Your control consists only of existing components.
* You don't need to support complex customization.

### Deriving from Control

When you create a control that inherits from the [Control](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.control) class, you define its appearance by using templates.

the UI and logic of your control are properly decoupled, a user of your control can redefine the control's [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate) to customize its appearance

#### Benefits of Deriving from Control

* You want the appearance of your control to be customizable via the [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate).
* You want your control to support different themes.

### Deriving from FrameworkElement

there are times when a control's appearance requires more than the functionality of simple element composition. For these scenarios, basing a component on [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) is the right choice.

There are two standard methods for building [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement)-based components: direct rendering and custom element composition.

Direct rendering involves overriding the [OnRender](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement.onrender) method of [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) and providing [DrawingContext](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawingcontext) operations that explicitly define the component visuals.

Custom element composition involves using objects of type [Visual](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.visual) to compose the appearance of your component

When an application looks for a resource, it looks at three levels in the following order:

 The element level.

The system starts with the element that references the resource and then searches resources of the logical parent and so forth until the root element is reached.

 The application level.

Resources defined by the [Application](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.application) object.

 The theme level.

Theme-level dictionaries are stored in a subfolder named Themes. The files in the Themes folder correspond to themes. For example, you might have Aero.NormalColor.xaml, Luna.NormalColor.xaml, Royale.NormalColor.xaml, and so on. You can also have a file named generic.xaml. When the system looks for a resource at the themes level, it first looks for it in the theme-specific file and then looks for it in generic.xaml.

When your control is in an assembly that is separate from the application, you must put your global resources at the element level or at the theme level. Both methods have their advantages.

#### Defining Resources at the Element Level

You can define shared resources at the element level by creating a custom resource dictionary and merging it with your control’s resource dictionary.

<UserControl.Resources>

<ResourceDictionary>

<ResourceDictionary.MergedDictionaries>

<ResourceDictionary Source="Dictionary1.xaml"/>

</ResourceDictionary.MergedDictionaries>

</ResourceDictionary>

</UserControl.Resources>

The disadvantage to this approach is that a [ResourceDictionary](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.resourcedictionary) object is created each time you reference it.

if you have 10 custom controls in your library and merge the shared resource dictionaries for each control by using XAML, you create 10 identical [ResourceDictionary](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.resourcedictionary) objects.

#### Defining Resources at the Theme Level

WPF enables you to create resources for different Windows themes. As a control author, you can define a resource for a specific theme to change your control's appearance depending on what theme is in use.

These files must be in a folder named Themes that is a subfolder of the folder that contains the control.

# Creating a Control That Has a Customizable Appearance

## Parts and States Model

The parts and states model specifies how to define the visual structure and visual behavior of a control. To follow the parts and states model, you should do the following:

* Define the visual structure and visual behavior in the [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate) of a control.
* Follow certain best practices when your control's logic interacts with parts of the control template.
* Provide a control contract to specify what should be included in the [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate).

## Defining the Visual Structure and Visual Behavior of a Control in a ControlTemplate

The visual structure of a control is the composite of [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) objects that make up the control. The visual behavior is the way the control appears when it is in a certain state.

## Using Parts of the ControlTemplate in Code

A [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate) author might omit [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) or [VisualState](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.visualstate) objects, either purposefully or by mistake, but your control's logic might need those parts to function properly.

Your control should not throw an exception or report an error if a [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement), [VisualState](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.visualstate), or [VisualStateGroup](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.visualstategroup) is missing from the [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate)

### Anticipate Missing FrameworkElement Objects

If a custom [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate) omits the [TextBlock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.textblock) or buttons, it is acceptable that the control loses some of its functionality, but you should be sure that your control does not cause an error.

The following practices will ensure that your control responds properly to missing [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) objects:

 Set the x:Name attribute for each [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) that you need to reference in code.

 Define private properties for each [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) that you need to interact with.

 Subscribe to and unsubscribe from any events that your control handles in the [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) property's set accessor.

 Set the [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) properties that you defined in step 2 in the [OnApplyTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement.onapplytemplate) method. This is the earliest that the [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) in the [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate) is available to the control. Use the x:Name of the [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) to get it from the [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate).

 Check that the [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) is not null before accessing its members. If it is null, do not report an error.

public override void OnApplyTemplate()

{

UpButtonElement = GetTemplateChild("UpButton") as RepeatButton;

DownButtonElement = GetTemplateChild("DownButton") as RepeatButton;

//TextElement = GetTemplateChild("TextBlock") as TextBlock;

UpdateStates(false);

}

### Use the VisualStateManager to Manage States

The control's logic is responsible for changing the control's state. The following example shows that the NumericUpDown control calls the [GoToState](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.visualstatemanager.gotostate) method to go into the Positive state when Value is 0 or greater

if (Value >= 0)

{

VisualStateManager.GoToState(this, "Positive", useTransitions);

}

else

{

VisualStateManager.GoToState(this, "Negative", useTransitions);

}

When a control calls [GoToState](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.visualstatemanager.gotostate) to change its state, the [VisualStateManager](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.visualstatemanager) does the following:

 If the [VisualState](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.visualstate) that the control is going to has a [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard), the storyboard begins. Then, if the [VisualState](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.visualstate) that the control is coming from has a [Storyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.animation.storyboard), the storyboard ends.

 If the control is already in the state that is specified, [GoToState](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.visualstatemanager.gotostate) takes no action and returns true.

 If state that is specified doesn't exist in the [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate) of control, [GoToState](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.visualstatemanager.gotostate) takes no action and returns false.

It is recommended that you do the following to maintain your control's states:

* Use properties to track its state.
* Create a helper method to transition between states.

There are three typical places where the state of a control might change:

* When the [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate) is applied to the [Control](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.control).
* When a property changes.
* When an event occurs.

You should update the state of the control in the [OnApplyTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement.onapplytemplate) method so that the control appears in the correct state when the [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate) is applied.

public override void OnApplyTemplate()

{

UpButtonElement = GetTemplateChild("UpButton") as RepeatButton;

DownButtonElement = GetTemplateChild("DownButton") as RepeatButton;

//TextElement = GetTemplateChild("TextBlock") as TextBlock;

UpdateStates(false);

}

private static void ValueChangedCallback(DependencyObject obj,

DependencyPropertyChangedEventArgs args)

{

NumericUpDown ctl = (NumericUpDown)obj;

int newValue = (int)args.NewValue;

// Call UpdateStates because the Value might have caused the

// control to change ValueStates.

ctl.UpdateStates(true);

// Call OnValueChanged to raise the ValueChanged event.

ctl.OnValueChanged(

new ValueChangedEventArgs(NumericUpDown.ValueChangedEvent,

newValue));

}

protected override void OnGotFocus(RoutedEventArgs e)

{

base.OnGotFocus(e);

UpdateStates(true);

}

## Providing the Control Contract

You provide a control contract so that [ControlTemplate](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.controltemplate) authors will know what to put in the template. A control contract has three elements:

* The visual elements that the control's logic uses.
* The states of the control and the group each state belongs to.
* The public properties that visually affect the control.

To specify what [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) objects the control expects, you use the [TemplatePartAttribute](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.templatepartattribute), which specifies the name and type of the expected elements. To specify the possible states of a control, you use the [TemplateVisualStateAttribute](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.templatevisualstateattribute), which specifies the state's name and which [VisualStateGroup](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.visualstategroup) it belongs to.

[TemplatePart(Name = "UpButtonElement", Type = typeof(RepeatButton))]

[TemplatePart(Name = "DownButtonElement", Type = typeof(RepeatButton))]

[TemplateVisualState(Name = "Positive", GroupName = "ValueStates")]

[TemplateVisualState(Name = "Negative", GroupName = "ValueStates")]

[TemplateVisualState(Name = "Focused", GroupName = "FocusedStates")]

[TemplateVisualState(Name = "Unfocused", GroupName = "FocusedStates")]

public class NumericUpDown : Control

{}

# Adorners Overview

Adorners are a special type of [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement), used to provide visual cues to a user. Adorners can be used to add functional handles to elements or provide state information about a control.

An [Adorner](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.documents.adorner) is a custom [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) that is bound to a [UIElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement). Adorners are rendered in an [AdornerLayer](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.documents.adornerlayer), Rendering of an adorner is independent from rendering of the [UIElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement) that the adorner is bound to. An adorner is typically positioned relative to the element to which it is bound, using the standard 2-D coordinate origin located at the upper-left of the adorned element.

Common applications for adorners include:

* Adding functional handles to a [UIElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement) that enable a user to manipulate the element in some way (resize, rotate, reposition, etc.).
* Provide visual feedback to indicate various states, or in response to various events.
* Overlay visual decorations on a [UIElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement).
* Visually mask or override part or all of a [UIElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement).

# Routed Events Overview

### Top-level Scenarios for Routed Events

However, the added image must not break the hit-testing behavior that causes a button to respond to a [Click](https://docs.microsoft.com/en-us/dotnet/api/system.windows.controls.primitives.buttonbase.click) of its content, even if the user clicks on pixels that are technically part of the image.

In Windows Forms, you would have to attach the same handler multiple times to process events that could be raised from multiple elements. Routed events enable you to attach that handler only once

**Class handling:** Routed events permit a static handler that is defined by the class. This class handler has the opportunity to handle an event before any attached instance handlers can.

**Referencing an event without reflection:** Certain code and markup techniques require a way to identify a specific event. A routed event creates a [RoutedEvent](https://docs.microsoft.com/en-us/dotnet/api/system.windows.routedevent) field as an identifier, which provides a robust event identification technique that does not require static or run-time reflection.

### How Routed Events Are Implemented

A routed event is a CLR event that is backed by an instance of the [RoutedEvent](https://docs.microsoft.com/en-us/dotnet/api/system.windows.routedevent) class and registered with the WPF event system。The connection to the identically named CLR event is accomplished by overriding the add and remove implementations for the CLR event.

public static readonly RoutedEvent TapEvent = EventManager.RegisterRoutedEvent(

"Tap", RoutingStrategy.Bubble, typeof(RoutedEventHandler), typeof(MyButtonSimple));

// Provide CLR accessors for the event

public event RoutedEventHandler Tap

{

add { AddHandler(TapEvent, value); }

remove { RemoveHandler(TapEvent, value); }

}

## Routing Strategies

Bubbling routed events are generally used to report input or state changes from distinct controls or other UI elements.

**Direct:** Only the source element itself is given the opportunity to invoke handlers in response.

Tunneling routed events are often used or handled as part of the compositing for a control, such that events from composite parts can be deliberately suppressed or replaced by events that are specific to the complete control.

## Why Use Routed Events?

Routed event listeners and routed event sources do not need to share a common event in their hierarchy. Any [UIElement](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement) or [ContentElement](https://docs.microsoft.com/en-us/dotnet/api/system.windows.contentelement) can be an event listener for any routed event.

Routed events can also be used to communicate through the element tree, because the event data for the event is perpetuated to each element in the route. One element could change something in the event data, and that change would be available to the next element in the route.

Certain WPF styling and templating features such as [EventSetter](https://docs.microsoft.com/en-us/dotnet/api/system.windows.eventsetter) and [EventTrigger](https://docs.microsoft.com/en-us/dotnet/api/system.windows.eventtrigger) require the referenced event to be a routed event.

Routed events support a class handling mechanism whereby the class can specify static methods that have the opportunity to handle routed events before any registered instance handlers can access them

## Adding and Implementing an Event Handler for a Routed Event

<Button Click="b1SetColor">button</Button>

Routed event handlers can always be added through a helper method [AddHandler](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement.addhandler) (which is the same method that the existing backing calls for add.)

Button b2 = new Button();

b2.AddHandler(Button.ClickEvent, new RoutedEventHandler(Onb2Click));

Button b2 = new Button();

b2.Click += new RoutedEventHandler(Onb2Click2);

### The Concept of Handled

All routed events share a common event data base class, [RoutedEventArgs](https://docs.microsoft.com/en-us/dotnet/api/system.windows.routedeventargs). [RoutedEventArgs](https://docs.microsoft.com/en-us/dotnet/api/system.windows.routedeventargs) defines the [Handled](https://docs.microsoft.com/en-us/dotnet/api/system.windows.routedeventargs.handled) property

For most common handler scenarios, marking an event as handled by setting [Handled](https://docs.microsoft.com/en-us/dotnet/api/system.windows.routedeventargs.handled) to true will "stop" routing for either a tunneling route or a bubbling route, and also for any event that is handled at a point in the route by a class handler.

However, there is a "handledEventsToo" mechanism whereby listeners can still run handlers in response to routed events where [Handled](https://docs.microsoft.com/en-us/dotnet/api/system.windows.routedeventargs.handled) is true in the event data. You can only use the handledEventsToo mechanism in code, or in an [EventSetter](https://docs.microsoft.com/en-us/dotnet/api/system.windows.eventsetter):

* In code, instead of using a language-specific event syntax that works for general CLR events, call the WPF method [AddHandler(RoutedEvent, Delegate, Boolean)](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement.addhandler#System_Windows_UIElement_AddHandler_System_Windows_RoutedEvent_System_Delegate_System_Boolean_) to add your handler. Specify the value of handledEventsToo as true.
* In an [EventSetter](https://docs.microsoft.com/en-us/dotnet/api/system.windows.eventsetter), set the [HandledEventsToo](https://docs.microsoft.com/en-us/dotnet/api/system.windows.eventsetter.handledeventstoo) attribute to be true.

The event still routes to the next listener, but with [Handled](https://docs.microsoft.com/en-us/dotnet/api/system.windows.routedeventargs.handled)=true in its event data, so only handledEventsToo listeners have the opportunity to invoke further handlers.

## Class Handlers

If you are defining a class that derives in some way from [DependencyObject](https://docs.microsoft.com/en-us/dotnet/api/system.windows.dependencyobject), you can also define and attach a class handler for a routed event that is a declared or inherited event member of your class.

## Attached Events in WPF

The element handling the event need not define or inherit the attached event, and neither the object potentially raising the event nor the destination handling instance must define or otherwise "own" that event as a class member.

## Qualified Event Names in XAML

<Border Height="50" Width="300" BorderBrush="Gray" BorderThickness="1">

<StackPanel Background="LightGray" Orientation="Horizontal" Button.Click="CommonClickHandler">

<Button Name="YesButton" Width="Auto" >Yes</Button>

<Button Name="NoButton" Width="Auto" >No</Button>

<Button Name="CancelButton" Width="Auto" >Cancel</Button>

</StackPanel>

</Border>

[Button](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.button) "owns" the event, but the routed event system permits handlers for any routed event to be attached to any [UIElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement) or [ContentElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.contentelement) instance listener

## WPF Input Events

Input events often come in pairs, with one being the bubbling event and the other being the tunneling event

Occasionally, input events only have a bubbling version, or perhaps only a direct routed version.

First, the tunneling event is raised and travels its route. Then the bubbling event is raised and travels its route. The two events literally share the same event data instance

If an element along the tunneling route marked the routed event as handled, the already-handled event data is sent on for the bubbling event, and typical handlers attached for the equivalent bubbling input events will not be invoked.

A routed event handler delegate provides references to two objects

The object where the handler was invoked is the object reported by the sender parameter.

The object where the event was first raised is reported by the [Source](https://docs.microsoft.com/en-us/dotnet/api/system.windows.routedeventargs.source) property in the event data

## EventSetters

In styles, you can include some pre-declared XAML event handling syntax in the markup by using an [EventSetter](https://docs.microsoft.com/en-us/dotnet/api/system.windows.eventsetter). When the style is applied, the referenced handler is added to the styled instance.

<StackPanel

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

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

x:Class="SDKSample.EventOvw2"

Name="dpanel2"

Initialized="PrimeHandledToo"

>

<StackPanel.Resources>

<Style TargetType="{x:Type Button}">

<EventSetter Event="Click" Handler="b1SetColor"/>

</Style>

</StackPanel.Resources>

<Button>Click me</Button>

<Button Name="ThisButton" Click="HandleThis">

Raise event, handle it, use handled=true handler to get it anyway.

</Button>

</StackPanel>

# Attached Events Overview

The concept of an attached event enables you to add a handler for a particular event to an arbitrary element rather than to an element that actually defines or inherits the event. In this case, neither the object potentially raising the event nor the destination handling instance defines or otherwise "owns" the event

In XAML syntax, the attached event is specified not just by its event name, but by its owning type plus the event name, separated by a dot (.)

## How WPF Implements Attached Events

In WPF, attached events are backed by a [RoutedEvent](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.routedevent) field and are routed through the tree after they are raised. Typically, the source of the attached event (the object that raises the event) is a system or service source, 因此，运行引发事件的代码的对象不是元素树的直接部分。

Classes that interact with or use the service can either use the event in the attached event syntax，或选择将附加事件作为路由事件表示，该事件是类集成服务功能的一部分。

Generally, the attached event serves an architecture purpose, but is then forwarded to a non-attached (backed with a CLR event "wrapper") routed event.

the underlying attached event [Mouse.MouseDown](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.mouse.mousedown) can more easily be handled on any given [UIElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement) by using [MouseDown](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement.mousedown) on that [UIElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement) rather than dealing with attached event syntax either in XAML or code. The attached event serves a purpose in the architecture because it allows for future expansion of input devices

## Handling an Attached Event in WPF

In general, a WPF attached event is not very different from a WPF routed event. The differences are how the event is sourced and how it is exposed by a class as a member (which also affects the XAML handler syntax).

More often, the purpose of the event is to enable a composited element to report a state to a parent element in compositing, in which case the event is usually raised in code and also relies on class handling in the relevant parent class.

## Defining Your Own Attached Events as Routed Events

A method **AddEventNameHandler** with two parameters. The first parameter is the instance to which the event handler is added. The second parameter is the event handler to add. The method must be public and static, with no return value.

A method **RemoveEventNameHandler** with two parameters.

The **AddEventNameHandler** implementation for a WPF attached event consists of calling the [AddHandler](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement.addhandler) with the routed event and handler as arguments.

This implementation strategy and the routed event system in general restrict handling for attached events to either [UIElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement) derived classes or [ContentElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.contentelement) derived classes

For example, the following code defines the NeedsCleaning attached event on the owner class Aquarium, using the WPF attached event strategy of declaring the attached event as a routed event.

C#

public static readonly RoutedEvent NeedsCleaningEvent = EventManager.RegisterRoutedEvent("NeedsCleaning", RoutingStrategy.Bubble, typeof(RoutedEventHandler), typeof(AquariumFilter));

public static void AddNeedsCleaningHandler(DependencyObject d, RoutedEventHandler handler)

{

UIElement uie = d as UIElement;

if (uie != null)

{

uie.AddHandler(AquariumFilter.NeedsCleaningEvent, handler);

}

}

public static void RemoveNeedsCleaningHandler(DependencyObject d, RoutedEventHandler handler)

{

UIElement uie = d as UIElement;

if (uie != null)

{

uie.RemoveHandler(AquariumFilter.NeedsCleaningEvent, handler);

}

}

## Raising a WPF Attached Event

if you are defining a custom attached event based on the WPF model of basing attached events on [RoutedEvent](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.routedevent), you can use [RaiseEvent](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement.raiseevent) to raise an attached event from any [UIElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement) or [ContentElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.contentelement). Raising a routed event (attached or not) requires that you declare a particular element in the element tree as the event source; that source is reported as the [RaiseEvent](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement.raiseevent) caller. Determining which element is reported as the source in the tree is your service's responsibility

## Object Lifetime Events

There are four main types of objects in WPF with respect to lifetime events; elements in general, window elements, navigation hosts, and application objects.

## Common Lifetime Events for Elements

Any WPF framework-level element (those objects deriving from either [FrameworkElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement) or [FrameworkContentElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkcontentelement)) has three common lifetime events: [Initialized](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement.initialized), [Loaded](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement.loaded), and [Unloaded](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement.unloaded)

### Initialized

[Initialized](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement.initialized) is raised first, and roughly corresponds to the initialization of the object by the call to its constructor. Because the event happens in response to initialization, you are guaranteed that all properties of the object are set.

the sequence of [Initialized](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement.initialized) being raised by nested elements that are defined in markup appears to occur in order of deepest elements in the element tree first, then parent elements toward the root

When you are writing handlers in response to the [Initialized](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement.initialized) event, you must consider that there is no guarantee that all other elements in the element tree around where the handler is attached have been created, particularly parent elements

### Loaded

[Loaded](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement.loaded) is raised next. The [Loaded](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement.loaded) event is raised before the final rendering, but after the layout system has calculated all necessary values for rendering.

When all elements in the tree are in a state where they are considered loaded, the [Loaded](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement.loaded) event is first raised on the root element. The [Loaded](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement.loaded) event is then raised successively on each child element

### Unloaded

# Marking Routed Events as Handled, and Class Handling

You must go through the extra effort of adding a handler with the handledEventsToo parameter version ([AddHandler(RoutedEvent, Delegate, Boolean)](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement.addhandler" \l "System_Windows_UIElement_AddHandler_System_Windows_RoutedEvent_System_Delegate_System_Boolean_)) in order to handle routed events that are marked handled by earlier participants in the event route.

## Class Handlers and Instance Handlers

Routed events consider two different types of listeners to the event: class listeners and instance listeners. Class listeners exist because types have called a particular [EventManager](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.eventmanager) API ,[RegisterClassHandler](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.eventmanager.registerclasshandler), in their static constructor , or have overridden a class handler virtual method from an element base class.

Each class handler is added to an internal store, and when the event route for an application is constructed, the class handlers are all added to the event route

Class handlers are added to the route such that the most-derived class handler is invoked first, and class handlers from each successive base class are invoked next.

# Preview Events

# Property Change Events

The event itself is sometimes a routed event and is sometimes a standard common language runtime (CLR) event.

### RoutedPropertyChanged Events

Certain events use an event data type and delegate that are explicitly used for property change events. The event data type is [RoutedPropertyChangedEventArgs<T>](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.routedpropertychangedeventargs-1), and the delegate is [RoutedPropertyChangedEventHandler<T>](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.routedpropertychangedeventhandler-1).The event data contains two properties, [OldValue](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.routedpropertychangedeventargs-1.oldvalue) and [NewValue](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.routedpropertychangedeventargs-1.newvalue),

The advantage of routing a property changed event is that the top level of a control can receive property changed events

### DependencyPropertyChanged Events

Events for these property changes are not routed; they are standard CLR events.

[DependencyPropertyChangedEventArgs](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.dependencypropertychangedeventargs) also reports an old and new value for the property.

# Commanding Overview

## What Are Commands?

Commands have several purposes

第一个目的是将语义和调用命令的对象与执行命令的逻辑分开。By using commands, you can bind each type of user action to the same logic.

Another purpose of commands is to indicate whether an action is available. The semantics of a command can be consistent across applications and classes

## Four Main Concepts in WPF Commanding

the command, the command source, the command target, and the command binding:

 The *command* is the action to be executed.

 The *command source* is the object which invokes the command.

 The *command target* is the object that the command is being executed on.

 The *command binding* is the object which maps the command logic to the command.

Commands in WPF are created by implementing the [ICommand](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.icommand) interface. [ICommand](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.icommand) exposes two methods, [Execute](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.icommand.execute), and [CanExecute](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.icommand.canexecute), and an event, [CanExecuteChanged](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.icommand.canexecutechanged).

[Execute](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.icommand.execute) performs the actions that are associated with the command. [CanExecute](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.icommand.canexecute) determines whether the command can execute on the current command target.

The [Execute](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.routedcommand.execute) and [CanExecute](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.routedcommand.canexecute) methods of a [RoutedCommand](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.routedcommand) do not contain the application logic for the command, but rather they raise routed events that tunnel and bubble through the element tree until they encounter an object with a [CommandBinding](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.commandbinding).

The [Execute](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.routedcommand.execute) method on a [RoutedCommand](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.routedcommand) raises the [PreviewExecuted](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.commandmanager.previewexecuted) and the [Executed](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.commandmanager.executed) events on the command target. The [CanExecute](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.routedcommand.canexecute) method on a [RoutedCommand](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.routedcommand) raises the [CanExecute](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.commandmanager.canexecute) and [PreviewCanExecute](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.commandmanager.previewcanexecute) events on the command target. These events tunnel and bubble through the element tree until they encounter an object which has a [CommandBinding](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.commandbinding) for that particular command.

### Command Sources

A command source is the object which invokes the command. Examples of command sources are [MenuItem](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.menuitem), [Button](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.button), and [KeyGesture](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.keygesture).

Command sources in WPF generally implement the [ICommandSource](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.icommandsource) interface.

[ICommandSource](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.icommandsource) exposes three properties: [Command](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.icommandsource.command), [CommandTarget](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.icommandsource.commandtarget), and [CommandParameter](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.icommandsource.commandparameter):

If the [CommandTarget](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.icommandsource.commandtarget) is not set, the element with keyboard focus will be the command target.

Typically, a command source will listen to the [CanExecuteChanged](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.routedcommand.canexecutechanged) event. This event informs the command source that the ability of the command to execute on the current command target may have changed. The command source can query the current status of the [RoutedCommand](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.routedcommand) by using the [CanExecute](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.routedcommand.canexecute) method

### CommandBinding

A [CommandBinding](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.commandbinding) associates a command with the event handlers that implement the command.

The [CommandBinding](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.commandbinding) class contains a [Command](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.commandbinding.command) property, and [PreviewExecuted](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.commandbinding.previewexecuted), [Executed](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.commandbinding.executed), [PreviewCanExecute](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.commandbinding.previewcanexecute), and [CanExecute](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.commandbinding.canexecute) events.

[Command](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.commandbinding.command) is the command that the [CommandBinding](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.commandbinding) is being associated with. The event handlers which are attached to the [PreviewExecuted](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.commandbinding.previewexecuted) and [Executed](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.commandbinding.executed) events implement the command logic.

### Command Target

The command target is the element on which the command is executed. With regards to a [RoutedCommand](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.routedcommand), the command target is the element at which routing of the [Executed](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.commandmanager.executed) and [CanExecute](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.commandmanager.canexecute) starts.

### The CommandManager

The [CommandManager](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.commandmanager) serves a number of command related functions.

It provides a set of static methods for adding and removing [PreviewExecuted](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.commandmanager.previewexecuted), [Executed](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.commandmanager.executed), [PreviewCanExecute](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.commandmanager.previewcanexecute), and [CanExecute](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.commandmanager.canexecute) event handlers to and from a specific element.

The first purpose is to separate the semantics and the object that invokes a command from the logic that executes the command. This allows for multiple and disparate sources to invoke the same command logic, and it allows the command logic to be customized for different targets.

Another purpose of commands is to indicate whether an action is available.

The semantics of a command can be consistent across applications and classes, but the logic of the action is specific to the particular object acted upon

### Commands

Commands in WPF are created by implementing the [ICommand](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.icommand) interface. [ICommand](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.icommand) exposes two methods, [Execute](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.icommand.execute), and [CanExecute](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.icommand.canexecute), and an event, [CanExecuteChanged](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.icommand.canexecutechanged).

The [Execute](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.routedcommand.execute) and [CanExecute](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.routedcommand.canexecute) methods of a [RoutedCommand](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.routedcommand) do not contain the application logic for the command, but rather they raise routed events that tunnel and bubble through the element tree until they encounter an object with a [CommandBinding](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.commandbinding).

### Command Sources

A command source is the object which invokes the command.

[ICommandSource](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.icommandsource) exposes three properties: [Command](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.icommandsource.command), [CommandTarget](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.icommandsource.commandtarget), and [CommandParameter](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.icommandsource.commandparameter)

the command source can query the current status of the [RoutedCommand](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.routedcommand) by using the [CanExecute](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.routedcommand.canexecute) method.

### CommandBinding

A [CommandBinding](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.commandbinding) associates a command with the event handlers that implement the command.

The [CommandBinding](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.commandbinding) class contains a [Command](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.commandbinding.command) property, and [PreviewExecuted](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.commandbinding.previewexecuted), [Executed](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.commandbinding.executed), [PreviewCanExecute](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.commandbinding.previewcanexecute), and [CanExecute](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.commandbinding.canexecute) events.

### Command Target

The command target is the element on which the command is executed. With regards to a [RoutedCommand](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.routedcommand), the command target is the element at which routing of the [Executed](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.commandmanager.executed) and [CanExecute](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.commandmanager.canexecute) starts.

The command source can explicitly set the command target. If the command target is not defined, the element with keyboard focus will be used as the command target.

### The CommandManager

The [CommandManager](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.commandmanager) serves a number of command related functions.

The [CommandManager](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.commandmanager) also provides a means, through the [RequerySuggested](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.commandmanager.requerysuggested) event, to notify a command when it should raise the [CanExecuteChanged](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.icommand.canexecutechanged) event.

# Input Overview

## Input API

The primary input API exposure is found on the base element classes: [UIElement](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement), [ContentElement](https://docs.microsoft.com/en-us/dotnet/api/system.windows.contentelement), [FrameworkElement](https://docs.microsoft.com/en-us/dotnet/api/system.windows.frameworkelement), and [FrameworkContentElement](https://docs.microsoft.com/en-us/dotnet/api/system.windows.frameworkcontentelement). These classes provide functionality for input events related to key presses, mouse buttons, mouse wheel, mouse movement, focus management, and mouse capture, to name a fewKeyboard and Mouse Classes

In addition to the input API on the base element classes, the [Keyboard](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.keyboard) class and [Mouse](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.mouse) classes provide additional API for working with keyboard and mouse input.

Examples of input API on the [Keyboard](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.keyboard) class are the [Modifiers](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.keyboard.modifiers) property, which returns the [ModifierKeys](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.modifierkeys) currently pressed, and the [IsKeyDown](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.keyboard.iskeydown) method, which determines whether a specified key is pressed.

Examples of input API on the [Mouse](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.mouse) class are [MiddleButton](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.mouse.middlebutton), which obtains the state of the middle mouse button, and [DirectlyOver](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.mouse.directlyover), which gets the element the mouse pointer is currently over.

### Stylus Input

## Event Routing

## Handling Input Events

### Keyboard Input Event

Keyboard events occur when the operating system reports key actions that occur while keyboard focus is on an element.

### Mouse Input Event

## Text Input

The [TextInput](https://docs.microsoft.com/en-us/dotnet/api/system.windows.contentelement.textinput) event enables you to listen for text input in a device-independent manner. The keyboard is the primary means of text input, but speech, handwriting, and other input devices can generate text input also.

For keyboard input, WPF first sends the appropriate [KeyDown](https://docs.microsoft.com/en-us/dotnet/api/system.windows.contentelement.keydown)/[KeyUp](https://docs.microsoft.com/en-us/dotnet/api/system.windows.contentelement.keyup) events. If those events are not handled and the key is textual (rather than a control key such as directional arrows or function keys), then a [TextInput](https://docs.microsoft.com/en-us/dotnet/api/system.windows.contentelement.textinput) event is raised.

multiple keystrokes can generate a single character of text input and single keystrokes can generate multi-character strings. This is especially true for languages such as Chinese, Japanese, and Korean which use Input Method Editors (IMEs) to generate the thousands of possible characters in their corresponding alphabets.

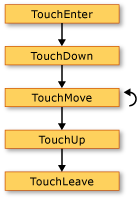
## Touch and Manipulation

New hardware and API in the Windows 7 operating system provide applications the ability to receive input from multiple touches simultaneously

WPF exposes two types of events when touch occurs: touch events and manipulation events. Touch events provide raw data about each finger on a touchscreen and its movement. Manipulation events interpret the input as certain actions

A **manipulation** occurs when touch is interpreted as a physical action that is applied to an object. In WPF, manipulation events interpret input as a translation, expansion, or rotation manipulation.

the touch events are routed events. consider the scenario where a user puts one finger on an element, moves the finger in the element, and then lifts the finger from the element. The following illustration shows the execution of the bubbling events



When more than two fingers are used, the events occur for each finger.

the manipulation events report how the input can be interpreted. There are three types of manipulations, translation, expansion, and rotation. More than one type of manipulation can occur simultaneously.

When you cause objects to respond to manipulations, you can have the object appear to have inertia. This can make your objects simulate the physical world. WPF enables you to simulate this behavior by raising manipulation events after the user's fingers releases the object.

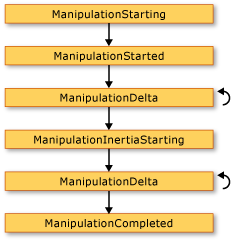
The [UIElement](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement) defines the following manipulation events.

* [ManipulationStarting](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement.manipulationstarting)
* [ManipulationStarted](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement.manipulationstarted)
* [ManipulationDelta](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement.manipulationdelta)
* [ManipulationInertiaStarting](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement.manipulationinertiastarting)
* [ManipulationCompleted](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement.manipulationcompleted)
* [ManipulationBoundaryFeedback](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement.manipulationboundaryfeedback)

By default, a [UIElement](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement) does not receive these manipulation events. To receive manipulation events on a [UIElement](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement), set [UIElement.IsManipulationEnabled](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement.ismanipulationenabled) to true.

#### The Execution Path of Manipulation Events

Consider a scenario where a user "throws" an object. The user puts a finger on the object, moves the finger across the touchscreen for a short distance, and then lifts the finger while it is moving



The [ManipulationStarting](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement.manipulationstarting) event occurs when the user places a finger on the object. Among other things, this event allows you to set the [ManipulationContainer](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.manipulationstartingeventargs.manipulationcontainer) property.

The [ManipulationStarted](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement.manipulationstarted) event occurs next. This event reports the origin of the manipulation.

The [ManipulationDelta](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement.manipulationdelta) event occurs multiple times as a user's fingers move on a touchscreen. The [DeltaManipulation](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.manipulationdeltaeventargs.deltamanipulation) property of the [ManipulationDeltaEventArgs](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.manipulationdeltaeventargs) class reports whether the manipulation is interpreted as movement, expansion, or translation.

The [ManipulationInertiaStarting](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement.manipulationinertiastarting) event occurs when the user's fingers lose contact with the object. This event enables you to specify the deceleration of the manipulations during inertia.

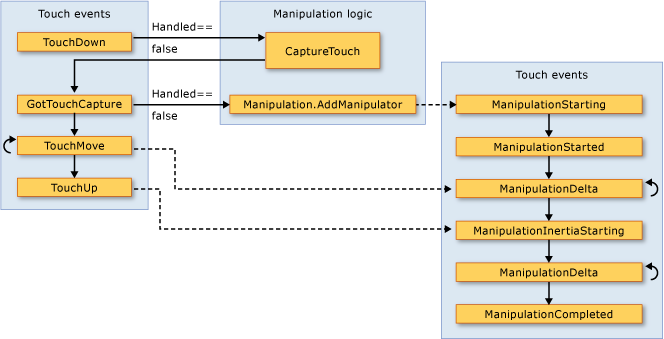
The [ManipulationCompleted](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement.manipulationcompleted) event occurs when the manipulation and any inertia ends.

The [UIElement](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement) also defines the [ManipulationBoundaryFeedback](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement.manipulationboundaryfeedback) event. This event occurs when the [ReportBoundaryFeedback](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.manipulationdeltaeventargs.reportboundaryfeedback) method is called in the [ManipulationDelta](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement.manipulationdelta) event. The [ManipulationBoundaryFeedback](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement.manipulationboundaryfeedback) event enables applications or components to provide visual feedback when an object hits a boundary.

You can cancel the manipulation by calling the [Cancel](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.manipulationstartingeventargs.cancel) method on the event arguments in any manipulation event except [ManipulationBoundaryFeedback](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement.manipulationboundaryfeedback) event. When you call [Cancel](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.manipulationstartingeventargs.cancel), the manipulation events are no longer raised and mouse events occur for touch.

### The Relationship Between Touch and Manipulation Events

A [UIElement](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement) can always receive touch events. When the [IsManipulationEnabled](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement.ismanipulationenabled) property is set to true, a [UIElement](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement) can receive both touch and manipulation events. If the [TouchDown](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement.touchdown) event is not handled (that is, the [Handled](https://docs.microsoft.com/en-us/dotnet/api/system.windows.routedeventargs.handled) property is false), the manipulation logic captures the touch to the element and generates the manipulation events.



 When the first touch device generates a [TouchDown](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement.touchdown) event on a [UIElement](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement), the manipulation logic calls the [CaptureTouch](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement.capturetouch) method, which generates the [GotTouchCapture](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement.gottouchcapture) event.

 When the [GotTouchCapture](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement.gottouchcapture) occurs, the manipulation logic calls the [Manipulation.AddManipulator](https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.manipulation.addmanipulator) method, which generates the [ManipulationStarting](https://docs.microsoft.com/en-us/dotnet/api/system.windows.uielement.manipulationstarting) event.

## Focus

There are two main concepts that pertain to focus in WPF: keyboard focus and logical focus.

### Keyboard Focus

Keyboard focus can be obtained by tabbing to an element or by clicking the mouse on certain elements, such as a [TextBox](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.textbox). Keyboard focus can also be obtained programmatically by using the [Focus](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.keyboard.focus) method on the [Keyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.keyboard) class

In order for an element to obtain keyboard focus the [Focusable](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement.focusable) property and the [IsVisible](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement.isvisible) properties must be set to **true**.

The following example uses [Focus](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.keyboard.focus) to set keyboard focus on a [Button](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.button). The recommended place to set initial focus in an application is in the [Loaded](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement.loaded) event handler.

private void OnLoaded(object sender, RoutedEventArgs e)

{

// Sets keyboard focus on the first Button in the sample.

Keyboard.Focus(firstButton);

}

### Logical Focus

Logical focus refers to the [FocusManager.FocusedElement](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.focusmanager.focusedelement) in a focus scope. There can be multiple elements that have logical focus in an application, but there may only be one element that has logical focus in a particular focus scope.

An element can be turned into a focus scope in Extensible Application Markup Language (XAML) by setting the [FocusManager](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.focusmanager) attached property [IsFocusScope](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.focusmanager.isfocusscope) to true

The [Keyboard](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.keyboard) class is concerned primarily with keyboard focus and the [FocusManager](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.focusmanager) is concerned primarily with logical focus, but this is not an absolute distinction. An element that has keyboard focus will also have logical focus, but an element that has logical focus does not necessarily have keyboard focus.

## Keyboard Navigation

The [KeyboardNavigation](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.keyboardnavigation) class is responsible for implementing default keyboard focus navigation when one of the navigation keys is pressed

The navigation behavior of a navigation container can be changed by setting the attached [KeyboardNavigation](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.keyboardnavigation) properties [TabNavigation](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.keyboardnavigation.tabnavigation), [ControlTabNavigation](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.keyboardnavigation.controltabnavigation), and [DirectionalNavigation](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.keyboardnavigation.directionalnavigation). the possible values are [Continue](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.keyboardnavigationmode#System_Windows_Input_KeyboardNavigationMode_Continue), [Local](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.keyboardnavigationmode#System_Windows_Input_KeyboardNavigationMode_Local), [Contained](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.keyboardnavigationmode#System_Windows_Input_KeyboardNavigationMode_Contained), [Cycle](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.keyboardnavigationmode#System_Windows_Input_KeyboardNavigationMode_Cycle), [Once](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.keyboardnavigationmode#System_Windows_Input_KeyboardNavigationMode_Once), and [None](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.keyboardnavigationmode#System_Windows_Input_KeyboardNavigationMode_None). The default value is [Continue](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.keyboardnavigationmode#System_Windows_Input_KeyboardNavigationMode_Continue), which means the element is not a navigation container.

<Menu KeyboardNavigation.TabNavigation="Cycle">

<MenuItem Header="Menu Item 1" />

<MenuItem Header="Menu Item 2" />

<MenuItem Header="Menu Item 3" />

<MenuItem Header="Menu Item 4" />

</Menu>

## Navigating Focus Programmatically

Additional API to work with focus are [MoveFocus](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement.movefocus) and [PredictFocus](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement.predictfocus).

[MoveFocus](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement.movefocus) changes focus to the next element in the application. A [TraversalRequest](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.traversalrequest) is used to specify the direction. The [FocusNavigationDirection](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.focusnavigationdirection) passed to [MoveFocus](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement.movefocus) specifies the different directions focus can be moved, such as [First](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.focusnavigationdirection#System_Windows_Input_FocusNavigationDirection_First), [Last](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.focusnavigationdirection#System_Windows_Input_FocusNavigationDirection_Last), [Up](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.focusnavigationdirection#System_Windows_Input_FocusNavigationDirection_Up) and [Down](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.focusnavigationdirection#System_Windows_Input_FocusNavigationDirection_Down).

The following example uses [MoveFocus](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.frameworkelement.movefocus) to change the focused element.

// Creating a FocusNavigationDirection object and setting it to a

// local field that contains the direction selected.

FocusNavigationDirection focusDirection = \_focusMoveValue;

// MoveFocus takes a TraveralReqest as its argument.

TraversalRequest request = new TraversalRequest(focusDirection);

// Gets the element with keyboard focus.

UIElement elementWithFocus = Keyboard.FocusedElement as UIElement;

elementWithFocus.MoveFocus(request);

## Focus Events

The events related to keyboard focus are [PreviewGotKeyboardFocus](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.keyboard.previewgotkeyboardfocus), [GotKeyboardFocus](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.keyboard.gotkeyboardfocus) and [PreviewLostKeyboardFocus](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.keyboard.previewlostkeyboardfocus), [LostKeyboardFocus](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.keyboard.lostkeyboardfocus).

If the [PreviewGotKeyboardFocus](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.keyboard.previewgotkeyboardfocus) event or the [PreviewLostKeyboardFocusEvent](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.keyboard.previewlostkeyboardfocusevent) event is handled and [Handled](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.routedeventargs.handled) is set to true, then focus will not change.

The events related to logical focus are [GotFocus](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement.gotfocus) and [LostFocus](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.uielement.lostfocus). These events are defined on the [FocusManager](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.focusmanager) as attached events

## Mouse Position

the WPF input API requires that you specify your frame of reference when you work with coordinates obtained through the mouse. The [GetPosition](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.input.mouse.getposition) method returns the coordinate of the mouse pointer relative to the specified element.

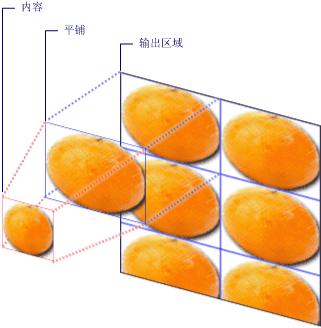
## Mouse Capture

During the drag, the user cannot click without aborting the drag-and-drop, which makes most mouseover cues inappropriate while the mouse capture is held by the drag origin. The input system exposes APIs that can determine mouse capture state, as well as APIs that can force mouse capture to a specific element, or clear mouse capture state

# TileBrush Overview

[TileBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.tilebrush)是[ImageBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.imagebrush), [DrawingBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawingbrush), 和[VisualBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.visualbrush)的父类

Tile brushes provide you with a great deal of control over how an area is painted with an image, drawing, or visual。



[TileBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.tilebrush) 根据[Viewport](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.tilebrush.viewport)，[ViewportUnits](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.tilebrush.viewportunits)和输出区域计算瓷砖（Tile）的位置和大小，不设置的话，瓷砖将拉伸到和输出区域一样大。根据[Stretch](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.tilebrush.stretch)将内容投射到瓷砖上。把瓷砖输出到输出区域，如果瓷砖和区域不一样大，[TileMode](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.tilebrush.tilemode)规定了重复铺排方式。

## Brush Content

 If the brush is an [ImageBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.imagebrush), this content is an image The [ImageSource](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.imagebrush.imagesource) property specifies the contents of the [ImageBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.imagebrush).

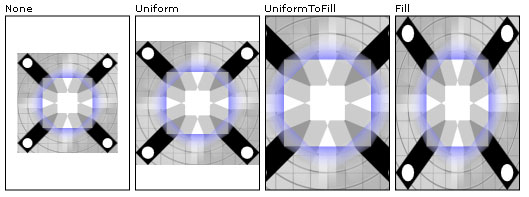
 If the brush is a [DrawingBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawingbrush), this content is a drawing. The [Drawing](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawingbrush.drawing) property specifies the contents of the [DrawingBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.drawingbrush).

 If the brush is a [VisualBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.visualbrush), this content is a visual. The [Visual](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.visualbrush.visual) property specifies the content of the [VisualBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.visualbrush).

You can specify the position and dimensions of [TileBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.tilebrush) content by using the [Viewbox](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.tilebrush.viewbox) property。By default, the [Viewbox](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.tilebrush.viewbox) is configured to completely contain the brush's contents

## The Base Tile

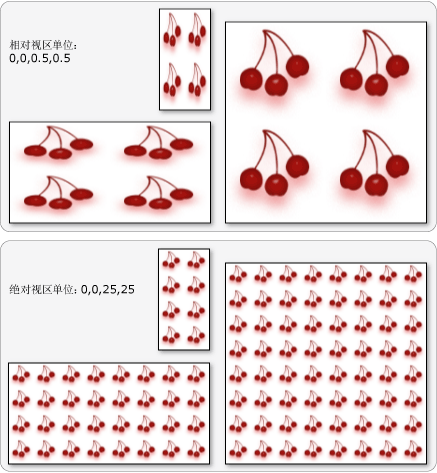
The [Stretch](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.tilebrush.stretch) property controls how [TileBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.tilebrush) content is stretched to fill the base tile



By default, a [TileBrush](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.tilebrush) generates a single tile (the base tile) and stretches that tile to completely fill the output area. You can change the size and position of the base tile by setting the [Viewport](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.tilebrush.viewport) and [ViewportUnits](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.tilebrush.viewportunits) properties.

The [Viewport](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.tilebrush.viewport) property determines the size and position of the base tile, and the [ViewportUnits](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.tilebrush.viewportunits) property determines whether the [Viewport](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.tilebrush.viewport) is specified using absolute or relative coordinates

If the coordinates are relative, they are relative to the size of the output area. The point (0,0) represents the top left corner of the output area, and (1,1) represents the bottom right corner of the output area

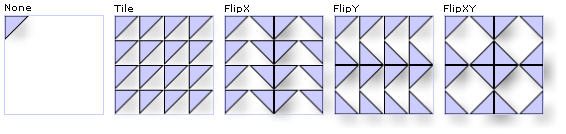


The base tile is located at (0,0) of the output area.

### Tiling Behavior

When a tile brush's tile does not completely fill the output area, its [TileMode](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.media.tilebrush.tilemode) property specifies whether the base tile should be duplicated to fill the output area and, if so, how the base tile should be duplicated.

The following image illustrates the different tiling modes.



# WPF Globalization and Localization Overview

Localization is the translation of application resources into localized versions for the specific cultures that the application supports.

## Best Practices for Globalization and Localization in WPF

### Best Practices for WPF UI Design

Write your UI in XAML; avoid creating UI in code. When you create your UI by using XAML, you expose it through built-in localization APIs.

Avoid using absolute positions and fixed sizes to lay out content; instead, use relative or automatic sizing.

 Use [SizeToContent](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.window.sizetocontent) and keep widths and heights set to Auto.

 Avoid using [Canvas](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.canvas) to lay out UIs.

 Use [Grid](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.grid) and its size-sharing feature

Provide extra space in margins because localized text often requires more space

Enable [TextWrapping](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.textblock.textwrapping) on [TextBlock](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.controls.textblock) to avoid clipping.

Set the xml:lang attribute. This attribute describes the culture of a specific element and its child elements.

Create a customized composite font to obtain better control of fonts that are used for different languages.

### Best Practices for WPF Localization

 Use localization comments to provide extra context for localizers.

Use localization attributes to control localization instead of selectively omitting [Uid](https://docs.microsoft.com/zh-cn/dotnet/api/system.windows.markup.localizer.bamllocalizableresourcekey.uid) properties on elements