# 1、Adorners（装饰器）

### DrawingContextExtensions:扩展绘图方法的类

方法：

|  |  |
| --- | --- |
| DrawArc(DrawingContext,Brush,Pen,Point,double,double,SweepDirection,double,double) | 使用画刷，画笔绘制几何图形 |
| DrawArc(DrawingContext ,Brush ，Pen ，Point,double,double, SweepDirection ,double ,double ) | 使用画刷，画笔绘制几何图形 |
| DrawArc(DrawingContext ,Brush ,Pen ,Point,double ,double,double ,double) | 使用画刷，画笔绘制几何图形 |

### RectangleAdorner:缩放矩形时，矩形中心出现十字准线的装饰器



属性：

|  |  |
| --- | --- |
| Rectangle | 获取或设置矩形 |

方法：

|  |  |
| --- | --- |
| OnRender(DrawingContext ) | 渲染的时候调用 |

### TargetSymbolAdorner



属性：

|  |  |
| --- | --- |
| Position | 获得或设置位置点 |

方法：

|  |  |
| --- | --- |
| OnRender(DrawingContext ) | 渲染TargetSymbolAdorner |

# 2.Class diagrams（类图）

# 3.Controls

### （1）CameraController：相机控制

#### （1.1）CameraController：提供通过鼠标和手势来操作相机的类

<helix:HelixViewport3D CameraMode="Inspect" CameraRotationMode="Trackball">

<helix:DefaultLights/>

<local:SpectrumAnalyser x:Name="analyzer"/>

</helix:HelixViewport3D>

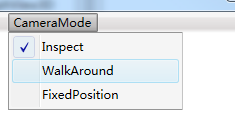
属性：

|  |  |
| --- | --- |
| ActualCamera | 获取实际相机 |
| Camera | 相机 |
| CameraLookDirection | 相机方向 |
| CameraMode | 相机模式 |
| CameraPosition | 相机位置 |
| CameraRotationMode | 相机旋转模式 |
| CameraTarget | 获取或设置向相机目标 |
| CameraUpDirection | 获取或设置相机垂直方向 |
| ChangeFieldOfViewCursor | 获取或设置改变视图光标 |
| DefaultCamera | 默认相机 |
| Enabled | 获取或设置一个值，指示是否启用。 |
| InertiaFactor | 获取或 设置惯性因子 |
| InfiniteSpin | 获取或设置一个值，指示是否为无限旋转 |
| IsActive | 获取一个值是否表示IsActive。 |
| IsChangeFieldOfViewEnabled | 能否改变视野 |
| IsInertiaEnabled | 获取或设置一个值，指示相机操作是否启用惯性 |
| IsMoveEnabled | 能否移动 |
| IsPanEnabled | 能否平移 |
| IsRotationEnabled | 能否旋转 |
| IsTouchZoomEnabled | 能否触摸缩放 |
| IsZoomEnabled | 能否缩放 |
| LeftRightPanSensitivity | 通过左右键获取或设置平移灵敏度 |
| LeftRightRotationSensitivity | 通过左右键获取或设置旋转灵敏度 |
| MaximumFieldOfView | 最大视野 |
| MinimumFieldOfView | 最小视野 |
| ModelUpDirection | 获取或设置模型的垂直方向 |
| MoveSensitivity | 获取或设置移动灵敏度。 |
| PageUpDownZoomSensitivity | 通过页面向上和向下翻页获取或设置缩放的灵敏度 |
| PanCursor | 平移光标 |
| RotateAroundMouseDownPoint | 获取或设置一个值，指示是否绕鼠标向下旋转 |
| RotateCursor | 旋转光标 |
| RotationSensitivity | 旋转灵敏度 |
| ShowCameraTarget | 获取或设置一个值，指示在操作相机时是否显示目标服务器。 |
| SpinReleaseTime | 获取或设置鼠标拖动的最大持续时间以激活旋转 |
| UpDownPanSensitivity | 通过向上和向下键获取或设置平移的灵敏度。 |
| UpDownRotationSensitivity | 通过向上和向下键获取或设置旋转的灵敏度。 |
| Viewport | 视口 |
| ZoomAroundMouseDownPoint | 获取或设置一个值，指示是否放大鼠标左下角 |
| ZoomCursor | 缩放光标 |
| ZoomRectangleCursor | 获取或设置缩放矩形光标 |
| ZoomSensitivity | 获取或设置缩放灵敏度 |
| IsOrthographicCamera | 获取一个值，指示是否是平行相机。 |
| IsPerspectiveCamera | 获取一个值，指示是否是正交相机。 |
| OrthographicCamera | 平行相机 |
| PerspectiveCamera | 正交相机 |

方法：

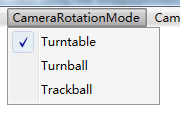
|  |  |
| --- | --- |
| AddMoveForce(double, double , double) | 添加移动强度 |
| AddMoveForce(Vector3D) | 添加移动强度 |
| AddPanForce(double, double) | 添加平移强度 |
| AddPanForce(Vector3D) | 添加平移强度 |
| AddRotateForce(double, double) | 添加旋转强度 |
| AddZoomForce(double) | 添加缩放强度 |
| ChangeDirection(Vector3D, Vector3D, double) | 改变相机方向 |
| ChangeDirection(Vector3D, double) | 改变相机方向 |
| HideRectangle() | 隐藏矩形 |
| HideTargetAdorner() | 隐藏目标adorner |
| LookAt(Point3D, doubl) | 改变look-at点 |
| PushCameraSetting() | 将当前相机设置推入内部堆栈 |
| ResetCamera() | 重置相机 |
| ResetCameraUpDirection() | 重置相机垂直方向 |
| RestoreCameraSetting() | 重置相机设置 |
| ShowRectangle(Rect, Color, Color) | 显示矩形 |
| ShowTargetAdorner(Point) | 显示目标adorner。 |
| StartSpin(Vector, Point, Point3D) | 开始旋转 |
| StopSpin() | 停止旋转 |
| UpdateRectangle(Rect) | 根性矩形 |
| Zoom(double) | 根据增量值缩放 |
| ZoomExtents(double) | 缩放模型范围 |
| RestoreCursor() | 从光标栈恢复光标 |
| SetCursor(Cursor) | 设置光标 |
| OnLookAtChanged() | 触发LookAtChanged事件 |
| OnZoomedByRectangle() | 引发ZoomedByRectangle事件 |
| OnManipulationCompleted(ManipulationCompletedEventArgs) | 触摸完成事件 |
| OnManipulationDelta(ManipulationDeltaEventArgs) | 触摸事件 |
| OnManipulationStarted(ManipulationStartedEventArgs) | 触摸开始事件 |
| OnStylusSystemGesture(StylusSystemGestureEventArgs) |  |
| CameraChanged(DependencyObject, DependencyPropertyChangedEventArgs) | 相机改变事件 |
| ViewportChanged(DependencyObject, DependencyPropertyChangedEventArgs ) | 视口改变事件 |
| BackViewHandler(object, ExecutedRoutedEventArgs ) | 后视图处理事件 |
| BottomViewHandler(object, ExecutedRoutedEventArgs) | 底视图处理事件 |
| CameraControllerLoaded(object, RoutedEventArgs ) | 相机控制器载入时发生 |
| CameraControllerUnloaded(objec, RoutedEventArgs ) | 相机控制器卸载时发生 |
| Clamp(double, double, double) | 限制特定的值 |
| FindPanVector(double, double) | 找到平移向量 |
| FrontViewHandler(object, ExecutedRoutedEventArgs ) | 前视图处理事件 |
| InitializeBindings() | 初始化输入绑定。 |
| LeftViewHandler(object, ExecutedRoutedEventArgs ) | 左视图处理事件 |
| OnCameraChanged() | 相机改变 |
| OnCompositionTargetRendering(object, RenderingEventArgs) | 渲染事件 |
| OnKeyDown(object, KeyEventArgs ) | 按键事件 |
| OnMouseWheel(object, MouseWheelEventArgs ) | 鼠标滚轮事件 |
| OnTimeStep(double) | 按时步骤 |
| OnViewportChanged() | 视图改变 |
| RefreshViewport() | 更新视图 |
| ResetCameraHandler(object, ExecutedRoutedEventArgs ) | 复位照相机事件 |
| RightViewHandler(object, ExecutedRoutedEventArgs) | 右视图事件处理 |
| StopAnimations() | 停止动画 |
| SubscribeEvents() | 订阅事件 |
| TopViewHandler(object, ExecutedRoutedEventArgs ) | 顶视图事件处理。 |
| UnSubscribeEvents() | 未订阅事件 |
| ZoomExtentsHandler(object, ExecutedRoutedEventArgs) | 缩放范围事件处理 |

#### （1.2）CameraMode（相机模式）



|  |  |
| --- | --- |
| Inspect | 固定目标位置，缩放时移动相机 |
| WalkAround | 固定相机位置，放大时沿摄像机移动 |
| FixedPosition | 固定摄像机目标，变焦时改变视角 |

#### （1.3）CameraRotationMode（相机旋转模式）



|  |  |
| --- | --- |
| Turntable | 转盘被限制在两个旋转轴（模型向上和向右） |
| Turnball | 转盘使用三个轴（正方向，右方向和向上方向（左/右边缘） |
| Trackball | 使用虚拟轨迹球 |

#### （1.4）CameraSetting（相机设置）

属性：

|  |  |
| --- | --- |
| FarPlaneDistance | 设置最远可视距离 |
| FieldOfView | 视角 |
| LookDirection | 摄像机方向 |
| NearPlaneDistance | 设置最近可视距离 |
| Position | 指定相机位置 |
| UpDirection | 摄像机的倾斜角度 |
| Width | 设置摄像机的宽度 |

方法：

|  |  |
| --- | --- |
| CameraSetting(ProjectionCamera) | 初始化CameraSetting类的新实例 |
| UpdateCamera(ProjectionCamera) | 将相机更新为此状态。 |

#### （1.5）ManipulationEventArgs（提供操作事件的数据）

属性：

|  |  |
| --- | --- |
| CurrentPosition | 设置当前操作位置 |

方法：

|  |  |
| --- | --- |
| ManipulationEventArgs(Point ) | 初始化ManipulationEventArgs类的新实例 |

#### （1.6）MouseGestureHandler（用于鼠标手势处理程序的抽象基类）

属性：

|  |  |
| --- | --- |
| Origin | 原始触控点 |
| Camera | 设置相机 |
| CameraLookDirection | 设置相机方向 |
| CameraMode | 设置相机模式 |
| CameraPosition | 设置相机位置 |
| CameraTarget | 设置相机目标点 |
| CameraUpDirection | 设置相机倾斜角度 |
| Controller | 设置相机控制 |
| LastPoint | 设置最后一个点（2D屏幕坐标） |
| LastPoint3D | 设置最后一个点（3D球体坐标） |
| ManipulationWatch |  |
| ModelUpDirection | 获取模型向上的方向 |
| MouseDownNearestPoint3D | 获取或设置在最近的命中元素（3D球体坐标）的鼠标点击点。 |
| MouseDownPoint | 获取或设置鼠标点击点（2D屏幕坐标） |
| MouseDownPoint3D | 获取或设置鼠标点击点（3D球面坐标）。 |
| RotationSensitivity | 获得旋转灵敏度 |
| Viewport | 获得视口 |
| ViewportHeight | 视口高度 |
| ViewportWidth | 视口宽度 |
| ZoomSensitivity | 缩放灵敏度 |

方法：

|  |  |
| --- | --- |
| Completed(ManipulationEventArgs ) | 操作完成时发生 |
| Delta(ManipulationEventArgs ) | 操作过程中位置发生改变时发生 |
| Execute(object,ExecutedRoutedEventArgs) | 开始鼠标手势出理 |
| Started(ManipulationEventArgs ) | 操作开始时发生 |
| UnProject(Point, Point3D, Vector3D) | 将2D平面点投影到3D平面 |
| UnProject(Point ) | 通过相机目标点将屏幕（2D）的点投射到平面上的一个点。 |
| CanExecute（） | 能否继续执行 |
| GetCursor(); | 获取手势的光标 |
| GetRay(Point) | 将光线线放入由2D中的位置给出的视图体积（屏幕坐标） |
| OnInertiaStarting(int) | 惯性开始时调用 |
| OnMouseDown(object,MouseButtonEventArgs ) | 当鼠标被按下时发生 |
| OnMouseMove(object, MouseEventArgs ) | 鼠标移动时发生 |
| OnMouseUp(object, MouseButtonEventArgs) | 鼠标释放时发生 |
| Project(Point3D p) | 计算3D点的屏幕位置 |
| SetMouseDownPoint(Point position) | 设置鼠标点击点 |

#### （1.7）PanHandler（处理平移）

属性

|  |  |
| --- | --- |
| panPoint3D | 3D平移起点 |

事件：

|  |  |
| --- | --- |
| PanHandler(CameraController) | 初始化PanHandler的新实例 |
| Delta(ManipulationEventArgs ) | 当操作过程中位置发生改变时发生 |
| Pan(Vector3D) | 用指定的3D矢量（球体坐标）平移相机 |
| Pan(Vector) | 用指定的2D矢量（屏幕坐标）平移相机。 |
| Started(ManipulationEventArgs ) | 当操作开始时发生 |
| CanExecute() | 当命令可以被目标执行 |
| GetCursor() | 获取手势的光标 |
| OnInertiaStarting(int) | 惯性开始时被调用 |

#### （1.8）RotateHandler（处理旋转）

属性：

|  |  |
| --- | --- |
| changeLookAt | Lookat是否改变 |
| rotationAxisX | X旋转轴 |
| rotationAxisY | Y旋转轴 |
| rotationPoint | 旋转点 |
| rotationPoint3D | 3D旋转点 |

方法：

|  |  |
| --- | --- |
| RotateHandler(CameraController, bool) | 初始化RotateHandler类的新实例 |
| Completed(ManipulationEventArgs ) | 当操作完成时发生 |
| Delta(ManipulationEventArgs e) | 操作过程中位置改变时发生 |
| LookAt(Point3D, double) | 改变lookat点 |
| Rotate(Point, Point, Point3D) | 围绕指定点旋转相机 |
| Rotate(Vector) | 旋转 |
| RotateTurnball(Point, Point, Point3D ) | 绕三个轴旋转 |
| RotateTurntable(Vector, Point3D) | 使用转盘旋转旋转相机 |
| Started(ManipulationEventArgs ) | 操作开始后发生 |
| CanExecute() | 能被执行 |
| GetCursor() | 获得光标 |
| OnInertiaStarting(int) | 惯性开始时调用 |
| ProjectToTrackball(Point, double, double) | 将平面点投影到球体坐标 |
| InitTurnballRotationAxes(Point ) | 初始旋转轴 |
| RotateAroundUpAndRight(Point, Point, Point3D) | 围绕照相机向上和向右旋转 |
| RotateTrackball(Point, Point, Point3D) | 旋转轨迹球 |

#### （1.9）ZoomHandler（缩放操作）

属性：

|  |  |
| --- | --- |
| changeFieldOfView | 改变视角 |
| zoomPoint | 缩放点 |
| zoomPoint3D | 3D缩放点 |

方法：

|  |  |
| --- | --- |
| ZoomHandler(CameraController, bool) | 初始化ZoomHandler的新实例 |
| ZoomByChangingCameraPosition(double, Point3D) | 改变相机位置缩放 |
| MoveCameraPosition(Vector3D) | 通过特定向量改变相机位置 |
| ZoomByChangingCameraWidth(double,Point3D) | 改变相机宽度缩放 |
| ZoomByChangingFieldOfView(double) | 改变视角缩放 |
| Completed(ManipulationEventArgs ) | 操作完成时发生 |
| Delta(ManipulationEventArgs ) | 操作时位置改变时发生 |
| Started(ManipulationEventArgs ) | 操作开始后发生 |
| Zoom(double delta) | 缩放视图 |
| Zoom(double delta, Point3D zoomAround) | 围绕指定点缩放视图 |
| CanExecute() | 命令可以被执行 |
| GetCursor() | 获得手势时光标 |
| ChangeCameraDistance(double, Point3D) | 改变相机距离 |

#### （1.10）ZoomRectangleHandler（处理矩形缩放）

属性：

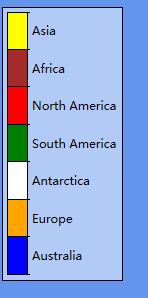
|  |  |
| --- | --- |
| zoomRectangle | 缩放的矩形 |

方法：

|  |  |
| --- | --- |
| ZoomRectangleHandler(CameraController) | 初始化ZoomRectangleHandler的新实例 |
| Completed(ManipulationEventArgs ) | 操作完成时发生 |
| Delta(ManipulationEventArgs ) | 操作时位置改变时发生 |
| Started(ManipulationEventArgs ) | 操作开始后发生 |
| ZoomRectangle(Rect) | 缩放到特定矩形 |
| CanExecute() | 命令可以被执行 |
| GetCursor() | 获得手势时光标 |

### （2）ColorAxis(颜色轴)

#### （2.1）CategorizedColorAxis（为不同类别提供颜色轴）



<h:CategorizedColorAxis Categories="{Binding Categories}" ColorScheme="{Binding ColorScheme2}" />

this.Categories = new[] { "Asia", "Africa", "North America", "South America", "Antarctica", "Europe", "Australia" };

this.ColorScheme2 =BrushHelper.CreateSteppedGradientBrush(new[] { Colors.Yellow, Colors.Brown, Colors.Red, Colors.Green, Colors.White, Colors.Orange, Colors.Blue },false);

属性：

|  |  |
| --- | --- |
| Categories | 获取或设置类别 |

方法：

|  |  |
| --- | --- |
| AddVisuals() | 更新视图 |
| GetTickLabels() | 获取刻度标签 |

#### （2.2）ColorAxis

属性：

|  |  |
| --- | --- |
| BarWidth | 获取或设置颜色条矩形的宽度 |
| ColorScheme | 获取或设置配色方案 |
| FlipColorScheme | 获取或设置配色方案，如果为True，则转换正常颜色方向 |
| Position | 获取或设置位置 |
| TextMargin | 获取或设置文本边距 |
| TickLength | 获取或设置刻度的长度 |
| Canvas | 获得画布 |
| ColorArea | 获得颜色矩形区域 |

方法：

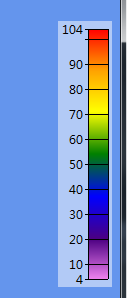
|  |  |
| --- | --- |
| ColorAxis() | 初始化ColorAxis()的新实例 |
| OnApplyTemplate() |  |
| PropertyChanged(DependencyObject, DependencyPropertyChangedEventArgs) | 处理属性改变 |
| AddVisuals() | 添加视图 |
| GetTickLabels() | 获取刻度标签 |
| MeasureOverride(Size) |  |
| UpdateVisuals() | 更新视图 |

#### （2.3）ColorAxisPosition（颜色轴的位置）

属性：

|  |  |
| --- | --- |
| Left | 左边位置 |
| Right | 右边位置 |

#### （2.4）RangeColorAxis



<h:RangeColorAxis Grid.Row="1" Grid.Column="2" Minimum="4" Maximum="104" Step="10" Margin="8" Padding="4 8 4 8" ColorScheme="{Binding ColorScheme}" Background="#80FFFFFF" Position="Right"/>

属性：

|  |  |
| --- | --- |
| FormatProvider | 获取或设置提供的格式 |
| FormatString | 获取或设置格式字符串 |
| Maximum | 获取或设置最大值 |
| MaximumTextureCoordinate | 获取或设置最大纹理坐标 |
| Minimum | 获取或设置最小值 |
| MinimumTextureCoordinate | 获取或设置最小纹理坐标 |
| Step | 设置间隔 |

方法：

|  |  |
| --- | --- |
| AddVisuals() | 更新视图 |
| GetTickLabels() | 获取刻度标签 |
| GetTickValues() | 获取刻度值 |

### （3）Stereo（立方体）

#### （3.1）StereoControl（用立体相机控制的基础类）

属性：

|  |  |
| --- | --- |
| Camera | 获取或设置相机 |
| CameraRotationMode | 获取或设置相机旋转模式 |
| CopyDirectionVector | 获取或设置一个值，指示是否复制方向向量。 |
| CopyUpVector | 获取或设置一个值，指示是否复制方向向量。 |
| CrossViewing | 获取或设置一个值，指示相机是否设置为交叉查看 |
| StereoBase | 获取或设置立体基类 |
| Children | 获取子类 |
| LeftCamera | 获取或设置左边相机 |
| LeftViewport | 获取或设置左边视口 |
| RightCamera | 获取或设置右边相机 |
| RightViewport | 获取或设置右边视口 |

方法：

|  |  |
| --- | --- |
| StereoControl() | 初始化StereoControl()的一个新实例 |
| BindViewports(Viewport3D,Viewport3D) | 绑定视口 |
| BindViewports(Viewport3D Viewport3D, bool, bool) | 绑定视口 |
| Clear() | 清理集合元素 |
| ExportKerkythea(string leftFileName, string rightFileName) | 将视图导出到kerkythea |
| SynchronizeStereoModel() | 同步立体声模型 |
| UpdateCameras() | 更新相机 |
| StereoViewChanged(DependencyObject , DependencyPropertyChangedEventArgs ) | 处理立体视图发生改变 |
| CameraChanged(object, EventArgs) | 处理相机改变事件 |
| ChildrenCollectionChanged(object, NotifyCollectionChangedEventArgs ) | 处理集合元素发生改变 |

### HelixViewport3D（包含Viewport3D和CameraController的控件）

属性：

|  |  |
| --- | --- |
| BackViewGesture | 获取或设置后视图手势 |
| BottomViewGesture | 获取或设置底部视图手势 |
| CameraInertiaFactor | 获取或设置相机惯性因子 |
| CameraInfo | 获取相机信息 |
| CameraMode | 获取或设置相机模式 |
| CameraRotationMode | 获取或设置相机旋转模式 |
| ChangeFieldOfViewCursor | 获取或设置视野改变时的光标 |
| ChangeFieldOfViewGesture | 获取或设置视野改变时的手势 |
| ChangeLookAtGesture | 获取或设置look-at该变时的手势 |
| CoordinateSystemHeight | 获取或设置坐标系视口的高度 |
| CoordinateSystemHorizontalPosition | 获取或设置坐标系视口的水平位置 |
| CoordinateSystemLabelForeground | 获取或设置坐标系标签的颜色 |
| CoordinateSystemLabelX | 获取或设置坐标系X标签 |
| CoordinateSystemLabelY | 获取或设置坐标系Y标签 |
| CoordinateSystemLabelZ | 获取或设置坐标系Z标签 |
| CoordinateSystemVerticalPosition | 获取或设置坐标系视口的垂直位置 |
| CoordinateSystemWidth | 获取或设置坐标系视口的宽度 |
| CurrentPosition | 当前位置 |
| EnableCurrentPosition | 获取或设置值，当前位置值是否可用 |
| CalculateCursorPosition | 获取或设置值，当前光标位置值是否可用 |
| CursorPosition | 得到光标位置 |
| CursorOnElementPosition | 获取最近模型上当前的光标位置。 如果模型未命中，则该位置为null |
| CursorOnConstructionPlanePosition | 获取光标平面上的当前光标位置 |
| ConstructionPlane | 获取或设置定义当前定义平面光标位置的平面 |
| CursorRay | 获取光标 |
| DefaultCamera | 获取或设置默认相机 |
| FieldOfViewText | 获取视图文本 |
| FrameRate | 获取帧速率 |
| FrameRateText | 获取帧速率文本 |
| FrontViewGesture | 获取或设置前视图手势 |
| InfiniteSpin | 获取或设置一个指示是否启用无限旋转的值 |
| InfoBackground | 获取或设置CameraInfo和TriangleCount字段的背景画笔 |
| InfoForeground | 获取或设置信息文本的前景画笔 |
| IsChangeFieldOfViewEnabled | 获取或设置一个值，该值指示是否启用了视野更改。 |
| IsHeadLightEnabled | 获取或设置一个值，指示头灯是否启用 |
| IsInertiaEnabled | 获取或设置一个值，指示相机操作是否启用惯性 |
| IsPanEnabled | 获取或设置一个值，指示平移是否可用 |
| IsMoveEnabled | 获取或设置一个值，指示移动是否可用 |
| IsRotationEnabled | 获取或设置一个值，指示旋转是否可用 |
| IsTouchZoomEnabled | 获取或设置一个值，指示触摸是否可用 |
| IsZoomEnabled | 获取或设置一个值，指示缩放是否可用 |
| LeftRightPanSensitivity | 通过左右键获取或设置平移的灵敏度 |
| LeftRightRotationSensitivity | 通过左右键获取或设置旋转的灵敏度 |
| LeftViewGesture | 获取或设置左视图手势 |
| LookAtChanged |  |
| MaximumFieldOfView | 获取或设置最大视野 |
| MinimumFieldOfView | 获取或设置最小视野 |
| ModelUpDirection | 获取或设置模型的向上方向 |
| Orthographic | 获取或设置一个值，指示是否使用正交相机。 |
| OrthographicToggleGesture | 获取或设置正交切换手势 |
| PageUpDownZoomSensitivity | 通过页面向上和向下翻页获取或设置缩放的灵敏度 |
| PanCursor | 获取或设置平移光标 |
| PanGesture2 | 获取或设置替代平移手势 |
| PanGesture | 获取或设置平移手势 |
| ResetCameraGesture | 获取或设置复位摄像机手势 |
| ResetCameraKeyGesture | 获取或设置重设相机的关键手势 |
| RightViewGesture | 获取或设置右视图手势 |
| RotateAroundMouseDownPoint | 获取或设置一个值，指示是否绕鼠标向下旋转 |
| RotateCursor | 获取或设置旋转光标 |
| RotateGesture2 | 获取或设置代替的旋转手势 |
| RotateGesture | 获取或设置旋转手势 |
| RotationSensitivity | 获取或设置旋转灵敏度 |
| ShowCameraInfo | 获取或设置一个值，指示是否显示相机信息 |
| ShowCameraTarget | 获取或设置一个值，指示是否显示摄像机目标 |
| ShowCoordinateSystem | 获取或设置一个值，指示是否显示坐标系 |
| ShowFieldOfView | 获取或设置一个值，指示是否显示当前视野 |
| ShowFrameRate | 获取或设置一个值，指示是否显示帧速率 |
| ShowTriangleCountInfo | 获取或设置一个值，指示是否显示场景中的三角形总数 |
| ShowViewCube | 获取或设置一个值，指示是否显示视图立方体 |
| Status | 获取或设置状态 |
| SubTitle | 获取或设置子标题 |
| SubTitleSize | 获取或设置子标题大小 |
| TextBrush | 获取或设置文本画笔 |
| TitleBackground | 获取或设置标题背景颜色 |
| TitleFontFamily | 获取或设置标题字体 |
| Title | 获取或设置标题 |
| TitleSize | 获取或设置标题大小 |
| TopViewGesture | 获取或设置上视图手势 |
| TriangleCountInfo | 获取有关三角形数的信息 |
| UpDownPanSensitivity | 通过向上和向下键获取或设置平移的灵敏度 |
| UpDownRotationSensitivity | 通过向上和向下键获取或设置旋转的灵敏度 |
| ViewCubeBackText | 获取或设置多维立方体后视图文本 |
| ViewCubeBottomText | 获取或设置多维立方体底部视图文本 |
| ViewCubeFrontText | 获取或设置多维立方体前视图文本 |
| ViewCubeHeight | 获取或设置多维立方体高度 |
| ViewCubeHorizontalPosition | 获取或设置多维立方体水平位置 |
| ViewCubeLeftText | 获取或设置多维立方体左视图文本 |
| ViewCubeOpacity | 获取或设置多维立方体透明度 |
| ViewCubeRightText | 获取或设置多维立方体右视图文本 |
| ViewCubeTopText | 获取或设置多维立方体上视图文本 |
| ViewCubeVerticalPosition | 获取或设置多维立方体垂直位置 |
| ViewCubeWidth | 获取或设置多维立方体宽度 |
| ZoomAroundMouseDownPoint | 获取或设置一个值，指示是否放大鼠标左下角 |
| ZoomCursor | 获取或设置缩放光标 |
| ZoomExtentsGesture | 获取或设置缩放手势范围 |
| ZoomExtentsWhenLoaded | 获取或设置一个值，指示控件加载时是否缩放扩展盘。 |
| ZoomGesture2 | 获取或设置替代的缩放手势 |
| ZoomGesture | 获取或设置缩放手势 |
| ZoomRectangleCursor | 获取或设置缩放矩形光标 |
| ZoomRectangleGesture | 获取或设置缩放矩形手势 |
| ZoomSensitivity | 获取或设置缩放灵敏度 |
| Camera | 获取或设置相机 |
| CameraController | 得到相机控制器 |
| Children | 获取子元素 |
| DebugInfo | 获取或设置调试信息文本 |
| IsViewCubeEdgeClicksEnabled | 获取或设置视图多边形边框能否点击 |
| Lights | 获取灯光 |
| Viewport | 获得视口 |

方法：

|  |  |
| --- | --- |
| HelixViewport3D() | 初始化HelixViewport3D()的新实例 |
| CameraChanged | 相机属性更改事件 |
| LookAtChanged | 当目标点改变时发生 |
| ChangeCameraDirection(Vector3D, double) | 改变相机方向 |
| Copy() | 将视图复制到剪贴板 |
| CopyXaml() | 将视图复制到剪贴板为xaml格式。 |
| Export(string) | 将视图导出到指定的文件 |
| ExportStereo(string, double) | 将视图导出为具有指定文件名的立体图像 |
| FindNearest(Point, out Point3D, out Vector3D, out DependencyObject) | 找到最近的对象 |
| FindNearestPoint(Point) | 找到最近的点 |
| FindNearestVisual(Point) | 找到通过指定点最接近摄像机射线的 “Visual3D”对象。 |
| LookAt(Point3D, double) | 更改相机以查看指定的点。 |
| LookAt(Point3D, double, double) | 更改相机以查看指定的点。 |
| LookAt(Point3D, Vector3D, double) | 更改相机以查看指定的点。 |
| OnApplyTemplate() |  |
| ResetCamera() | 复位相机 |
| SetView(Point3D, Vector3D, Vector3D, double) | 设置相机的位置和方向 |
| FitView(Vector3D, Vector3D, double) | 设置相机方向，并调整相机位置以使模型适合视图。 |
| ZoomExtents(double) | 缩放到屏幕的范围 |
| ZoomExtents(Rect3D, double) | 缩放到屏幕的范围 |
| OnLookAtChanged() | 触发LookAtChanged事件 |
| OnZoomedByRectangle() | 触发ZoomedByRectangle事件 |
| OnCameraChanged() | 处理相机改变 |
| OnHeadlightChanged() | 处理“IsHeadLightEnabled”属性更改事件。 |
| OnItemsChanged(NotifyCollectionChangedEventArgs ) | 当项目改变的时候被触发 |
| OnItemsSourceChanged(IEnumerable, IEnumerable) | 当项目源改变时被触发 |
| OnMouseMove(MouseEventArgs) | 鼠标移动事件 |
| RaiseCameraChangedEvent() | 触发相机改变事件 |
| UpdateCursorPosition(Point) | 更新光标位置 |
| AddItems(IEnumerable ) | 添加指定项目 |
| CameraPropertyChanged(object, EventArgs ) | 处理当前相机改变事件 |
| CompositionTargetRendering(object, EventArgs ) | 处理CompositionTarget控件的渲染事件 |
| CopyHandler(object, ExecutedRoutedEventArgs ) | 处理复制命令 |
| OnCameraRotationModeChanged() | 处理相机模式更改 |
| OnControlLoaded(object, RoutedEventArgs ) | 处理载入事件 |
| OnControlUnloaded(object, RoutedEventArgs ) | 处理卸载事件 |
| OnOrthographicChanged() | 处理更改 为Orthographic相机事件。 |
| OnShowFrameRateChanged() | 处理帧速率更改事件 |
| OnShowTriangleCountInfoChanged() | 处理帧速率显示信息更改事件 |
| OrthographicToggle(object, ExecutedRoutedEventArgs ) | 处理平行相机更改命令 |
| RemoveItems(IEnumerable) | 移除指定项目 |
| SubscribeToRenderingEvent() | 订阅渲染事件 |
| UnsubscribeRenderingEvent() | 取消订阅渲染事件 |
| UpdateCameraInfo() | 更新相机信息 |
| UpdateFieldOfViewInfo() | 更新视野信息 |
| UpdateRenderingEventSubscription() | 更新订阅渲染事件 |
| ViewCubeViewportMouseEnter(object, MouseEventArgs ) | 处理鼠标进入多维立方体事件 |
| ViewCubeViewportMouseLeave(object, MouseEventArgs ) | 处理鼠标离开多维立方体事件 |

### IHelixViewport3D（三维视口界面）

属性：

|  |  |
| --- | --- |
| Camera | 获得相机 |
| CameraController | 获得相机控制器 |
| Lights | 获得灯光 |
| Viewport | 获得视口 |

方法：

|  |  |
| --- | --- |
| Copy() | 将视图复制到剪贴板 |
| CopyXaml() | 将视图以xaml格式复制到剪贴板 |
| Export(string fileName) | 将视图导出到指定的文件名 |
| ZoomExtents(double | 缩放范围 |

### InputBindingX（输入绑定支持绑定的手势）

属性：

|  |  |
| --- | --- |
| Gezture | 获得或设置手势 |

方法：

|  |  |
| --- | --- |
| GeztureChanged(DependencyObject, DependencyPropertyChangedEventArgs) | 手势改变时发生 |
| OnGeztureChanged() | 手势改变时调用 |

## 4.Converters

### （1）EnumToBooleanConverter（枚举类型转布尔类型接口）

方法：

|  |  |
| --- | --- |
| Convert(object, Type, object, CultureInfo) | 转换值。如果返回null，则使用null值 |
| ConvertBack(object, Type, object , CultureInfo) | 转换值。如果返回null，则使用null值 |

### （2）LinearConverter（线性转换）

属性：

|  |  |
| --- | --- |
| B | 获得或设置值 |
| M | 获得或设置值 |

方法：

|  |  |
| --- | --- |
| Convert(object, Type, object, CultureInfo) | 转换值，方法返回null，则null为有效值 |
| ConvertBack(object, Type, object , CultureInfo) | 转换值，方法返回null，则null为有效值 |
| GetDoubleValue(object, double) | 得到双精度的值 |

### （3）NotNullToVisibilityConverter（非空值转换为Visiblity值的转换）

属性：

|  |  |
| --- | --- |
| Inverted | 获取或设置一个值，表示该转换器是否反相 |

方法：

|  |  |
| --- | --- |
| NotNullToVisibilityConverter() | 初始化一个NotNullToVisibilityConverter()的新实例 |
| Convert(object, Type, object, CultureInfo) | 转换值，方法返回null，则null为有效值 |
| ConvertBack(object, Type, object , CultureInfo) | 转换值，方法返回null，则null为有效值 |

### （4）SelfProvider（提供实例本身的MarkupExtension）

方法：

|  |  |
| --- | --- |
| ProvideValue(IServiceProvider) | 在派生类中实现时，返回一个设置为此标记扩展的目标属性值的对象 |

### （5）StringToBooleanConverter（字符串向布尔类型的转换器）

方法：

|  |  |
| --- | --- |
| Convert(object, Type, object, CultureInfo) | 转换值，方法返回null，则null为有效值 |
| ConvertBack(object, Type, object , CultureInfo) | 转换值，方法返回null，则null为有效值 |

# 5.Exporters（导出）

### （1）BitmapExporter（将Viewport3D导出为.bmp，.png或.jpg文件）

属性：

|  |  |
| --- | --- |
| Background | 获取或设置背景色 |
| Format | 获取或设置文件名（png、jpg。bmp） |
| OversamplingMultiplier | 获取或设置过采样乘数 |

方法：

|  |  |
| --- | --- |
| BitmapExporter() | 初始化BitmapExporter的新实例 |
| Export(Viewport3D, Stream) | 导出指定的图形 |
| Export(Visual3D, Stream) | 导出指定的视口 |
| Export(Model3D, Stream) | 导出指定的模型 |

### （2）ColladaExporter（将3D图形导出为Collada文件）

属性：

|  |  |
| --- | --- |
| Author | 获取或设置作者 |
| Comments | 获取或设置注释 |
| Copyright | 获取或设置版权 |

方法：

|  |  |
| --- | --- |
| Export(Viewport3D, Stream) | 导出指定的视口 |
| XmlWriter Create(Stream) | 在指定的流上创建一个新的XmlWriter。 |
| Close(XmlWriter) | 关闭此导出 |
| ExportModel(XmlWriter, GeometryModel3D, Transform3D) | 导出模型 |
| WriteAssets(XmlWriter,Viewport3D) | 写入文件资产 |
| BindMaterial(XmlWriter, string, string) | 绑定指定的几何形状和材料 |
| ExportEffect(XmlWriter, GeometryModel3D) | 导出指定模型的效果 |
| ExportEffect(XmlWriter, Material ) | 导出指定材料的效果 |
| ExportGeometry(XmlWriter, GeometryModel3D Transform3D) | 导出指定模型的几何形状 |
| ExportLight(XmlWriter, Light ) | 导出指定灯光 |
| ExportMaterial(XmlWriter, GeometryModel3D) | 导出指定模型的材料。 |
| ExportMaterial(XmlWriter, Material) | 导出指定材料 |
| ExportNode(XmlWriter, GeometryModel3D, Transform3D) | 将指定的模型导出为节点 |
| ExportSceneNode(XmlWriter, Model3D, Transform3D) | 将指定的模型导出为场景节点 |
| WriteColor(XmlWriter, Color) | 写入指定的颜色 |
| WriteDouble(XmlWriter, string, double) | 写入指定的元素值 |
| WriteMatrix(XmlWriter, string, Matrix3D) | 写入指定的元素矩阵 |
| WritePhongMaterial(XmlWriter, string, Color ) | 写一个电子材料 |

### （3）Exporter（导出）

方法：

|  |  |
| --- | --- |
| Export(Viewport3D, Stream) | 导出指定的视口 |
| Export(Visual3D, Stream) | 导出指定的图形 |
| Export(Model3D, Stream) | 导出指定的模型 |
| Create(Stream) |  |
| Close(T) |  |
| ExportCamera(T, Camera) | 导出相机 |
| ExportHeader(T) | 导出标题 |
| ExportLight(T, Light, Transform3D) | 导出灯光 |
| ExportModel(T, GeometryModel3D, Transform3D) | 导出模型 |
| ExportViewport(T, Viewport3D) | 导出视口 |
| RenderBrush(Stream, Brush, int w, int, int) | 将画笔渲染为JPG图像 |
| RenderBrush(Stream, Brush, int w, int) | 将画笔渲染为PNG图像 |
| RenderBrush(Brush, int ,int) | 渲染指定的画笔 |
| Encode(RenderTargetBitmap, Stream) | 将指定的位图编码为PNG图像 |
| Encode(RenderTargetBitmap, Stream, int) | 将指定的位图编码为JPG图像 |

### （4）Exporters（包含所有支持的导出的列表）

方法：

|  |  |
| --- | --- |
| Create(string) | 基于指定路径的扩展创建导出器 |

### （5）IExporter（指定导出3D模型的功能）

方法：

|  |  |
| --- | --- |
| Export(Viewport3D, Stream) | 导出指定的视口（包括型号，相机和灯） |
| Export(Visual3D, Stream); | 导出指定的图形 |
| Export(Model3D, Stream); | 导出指定的模型 |

### KerkytheaExporter（将3D图形导出到Kerkythea 输入文件）

属性：

|  |  |
| --- | --- |
| RenderSettings | 渲染设置（RayTracer, PhotonMap, MetropolisLightTransport） |
| Aperture | 获取或设置光圈 |
| BackgroundColor | 获取或设置背景色 |
| FileCreator | 获取或设置纹理文件创建者 |
| FocalLength | 获取或设置焦点的长度 |
| FocusDistance | 获取或设置焦距 |
| Height | 获取或设置高度 |
| LensSamples | 获取或设置镜头样品 |
| LightMultiplier | 获取或设置光乘数 |
| Name | 获取或设置名字 |
| ReflectionColor | 获取或设置反射的颜色 |
| Reflections | 获取或设置一个值，表示这个KerkytheaExporter是否是反射 |
| RenderSetting | 获取或设置渲染设置 |
| ShadowColor | 获取或设置阴影颜色 |
| Shadows | 获取或设置一个值，指示此KerkytheaExporter”是否为阴影 |
| SoftShadows | 获取或设置一个值，指示[软阴影] |
| TextureHeight | 获取或设置纹理的高度 |
| TexturePath | 获取或设置纹理路径 |
| TextureWidth | 获取或设置纹理的宽度 |
| Threads | 获取或设置线程 |
| Width | 获取或设置宽度 |

方法：

|  |  |
| --- | --- |
| KerkytheaExporter() | 初始化KerkytheaExporter的一个新实例 |
| ExportMesh(KerkytheaWriter, MeshGeometry3D ) | 导出网格 |
| RegisterMaterial(Material, Stream) | 注册材料 |
| WriteMetropolisLightTransport(KerkytheaWriter, string) |  |
| WriteStandardRayTracer(KerkytheaWriter , string) | 写入标准的射线追踪器属性 |
| WriteThreadedRaytracer(KerkytheaWriter, int) |  |
| KerkytheaWriter Create(Stream) | 创建指定流的作者。 |
| ExportCamera(KerkytheaWriter, Camera ) | 导出相机 |
| ExportHeader(KerkytheaWriter) | 导出文本标题 |
| ExportLight(KerkytheaWriter, Light, Transform3D ) | 导出灯光 |
| ExportModel(KerkytheaWriter, GeometryModel3D, Transform3D) | 导出模型 |
| ExportViewport(KerkytheaWriter, Viewport3D) | 导出指定视图 |
| Close(KerkytheaWriter) | 关闭导出 |
| CreateTransform(Point3D, Vector3D, Vector3D) | 创建转换 |
| ToKerkytheaString(Point) | 转换点到字符串格式 |
| ToKerkytheaString(Point3D | 转换点到字符串格式 |
| ToKerkytheaString(Vector3D vector) | 转换3D向量到字符串格式 |
| ToKerkytheaString(Color ) | 转换颜色到字符串格式 |
| ExportMapChannel(KerkytheaWriter, MeshGeometry3D) | 从指定的网格导出地图通道（纹理坐标） |
| ExportMaterial(KerkytheaWriter, string, Material, IList<double>) | 导出材料 |
| ExportMaterial(KerkytheaWriter, Material) | 导出指定材料 |
| GetSolidColor(Brush, Color) | 获取单一画刷 |
| GetTexture(KerkytheaWriter , Brush, string | 获取画笔的纹理 |
| GetUniqueName(KerkytheaWriter, DependencyObject, string) | 获取唯一名字 |
| WriteBitmapTexture(KerkytheaWriter, string, string) | 写一个位图纹理 |
| WriteConstantTexture(KerkytheaWriter, string, Color) | 写一个不变的纹理 |
| WriteDielectricMaterial(KerkytheaWriterr, string,Color?,Color?, double,double, string) | 写一个电介质材料 |
| WriteWeight(KerkytheaWriter, string, double) | 写一个重量 |
| WriteWhittedMaterial(KerkytheaWriter ， string, string, Color?, Color?, Color?, double double) | 写一个白色的材料 |
| ValueOrDefault(double, double) | 默认值 |
| KerkytheaWriter(Stream) | 初始化KerkytheaWriter的新实例 |
| WriteStartElement(string) | 写入起始元素 |
| WriteAttributeString(string ,string) | 写入属性字符串 |
| WriteEndElement() | 写入最终元素 |
| WriteFullEndElement() | 写入完整的元素 |
| WriteStartDocument() | 写入开始文件 |
| void Close() | 关闭此实例。 |
| WriteParameter(string, string, string ) | 写一个参数 |
| WriteParameter(string, string) | 写入一个字符串参数 |
| WriteParameter(string, Color) | 写入一个颜色参数 |
| WriteParameter(string, bool) | 写入一个布尔参数 |
| WriteParameter(string, double) | 写入一个双精度参数 |
| WriteParameter(string, int) | 写入一个整形参数 |
| WriteTransform(string, Matrix3D ) | 写一个转换矩阵 |
| WriteEndObject() | 写最终对象 |
| WriteObject(string, string , string, string) | 写对象 |
| WriteStartObject(string, string, string, string) | 写入起始对象 |
| WriteEndDocument() | 写入最终文件 |
| Write(XmlNode) | 写入指定的XML节点 |
| TryGetTexture(Brush, out string) | 尝试获取指定画笔的纹理 |
| AddTexture(Brush, string) | 添加指定的纹理 |
| GetUniqueName(string, string) | 获得一个唯一的名字 |

### ObjExporter（导出为OBJ文件）

属性：

|  |  |
| --- | --- |
| ExportNormals | 获取或设置一个值，指示是否导出法线 |
| FileCreator | 取或设置纹理图像和素材文件创建者 |
| UseDissolveForTransparency | 获取或设置一个值，指示是否使用“d”作为透明度（默认为“Tr”） |
| Comment | 获取或设置注释 |
| MaterialsFile | 获取或设置材料文件 |
| SwitchYZ | 获取值，指示是否转换Y和 Z坐标值 |
| TextureFolder | 获取或设置纹理文件夹 |
| TextureExtension | 获取或设置纹理扩展 |
| TextureSize | 获取或设置纹理大小 |
| TextureQualityLevel | 获取或设置纹理质量级别（用于JPEG编码） |

类：ObjWriters

|  |  |
| --- | --- |
| ObjWriter |  |
| MaterialsWriter |  |

方法：

|  |  |
| --- | --- |
| ExportMesh(StreamWriter, MeshGeometry3D, Transform3D) | 导出网格 |
| ObjWriters Create(Stream) | 创建指定流 |
| Close(ObjWriters) | 关闭导出 |
| ExportModel(ObjWriters， GeometryModel3Dl, Transform3D) | 导出模型 |
| ExportMaterial(StreamWriter, string, Material, Material) | 导出材料 |
| ToColorString(Color) | 将颜色转换为字符串 |

### PovRayExporter（导出PovRay文件）

方法：

|  |  |
| --- | --- |
| StreamWriter Create(Stream) | 创建指定的流 |
| Close(StreamWriter) | 关闭导出 |
| ExportCamera(StreamWriter,Camera) | 导出相机 |
| ExportLight(StreamWriter, Light, Transform3D) | 导出灯光 |
| ExportModel(StreamWriter, GeometryModel3D, Transform3D) | 导出模型 |

### RenderManExporter（导出为Render文件）

1. 方法：

|  |  |
| --- | --- |
| StreamWriter Create(Stream) | 创建指定的流 |
| Close(StreamWriter) | 关闭导出 |
| ExportCamera(StreamWriter,Camera) | 导出相机 |
| ExportLight(StreamWriter, Light, Transform3D) | 导出灯光 |
| ExportModel(StreamWriter, GeometryModel3D, Transform3D) | 导出模型 |

### （10）StlExporter（导出为STereoLithography二进制文件）

方法：

|  |  |
| --- | --- |
| BinaryWriter Create(Stream) | 创建BinaryWriter来写STereoLithography二进制文件 |
| Close(BinaryWriter) | 关闭BinaryWriter |
| Export(Viewport3D, Stream) | 导出指定视口 |
| Export(Visual3D, Stream) | 导出指定Visual3D |
| Export(Model3D, Stream) | 导出指定Model3D |
| ExportHeader(BinaryWriter, int) | 写一个空的STL标题 |
| ExportModel(BinaryWriter, GeometryModel3D, Transform3D) | 将GeometryModel3D写入STL二进制格式的 “BinaryWriter”。 |

### （11）VrmlExporter（导出VRML97文件）

属性：

|  |  |
| --- | --- |
| Title | 获取或设置标题 |

方法：

|  |  |
| --- | --- |
| StreamWriter Create(Stream) | 给指定流创建Writer |
| Close(StreamWriter) | 关闭导出Writer |
| ExportModel(StreamWriter, GeometryModel3D, Transform3D) | 导出模型 |

### X3DExporter（导出X3D文件）

属性：

|  |  |
| --- | --- |
| Metadata | 获取元数据 |
| Title | 设置标题 |

方法：

|  |  |
| --- | --- |
| XmlWriter Create(Stream) | 创建指定流的Writer |
| Close(XmlWriter) | 关闭导出 |
| ExportHeader(XmlWriter) | 导出标题 |
| ExportModel(XmlWriter, GeometryModel3D, Transform3D ) | 导出模型 |

### XamlExporter（导出为XMAL）

属性：

|  |  |
| --- | --- |
| CreateResourceDictionary | 获取或设置一个值指示是否创建资源字典 |

方法：

|  |  |
| --- | --- |
| WrapInResourceDictionary(object) | 在资源字典中包装指定的对象 |
| Export(Viewport3D, Stream) | 导出指定的视口 |
| Export(Visual3D, Stream) | 导出指定的视图 |
| Export(Model3D, Stream) | 导出指定的模型 |
| XmlWriter Create(Stream) | 为指定流创建XmlWriter |
| Close(XmlWriter) | 关闭导出 |

## 6.ExtensionMethods（扩展方法）

### （1）AnimationExtensions（动画扩展）

方法：

|  |  |
| --- | --- |
| AnimateOpacity(this, double, double) | 动画指定对象的不透明度 |

### （2）Matrix3DExtensions（Matrix3D变换扩展）

方法：

|  |  |
| --- | --- |
| Inverse(this Matrix3D) | 返回反向矩阵 |
| ToArray(this Matrix3D) | 转换Matrix3D为二维 |
| ToString(this Matrix3D, int) | 将矩阵转换为字符串，“\ t”和“\ n”作为分隔符。 |
| ToString(this Matrix3D, string, int) | 将矩阵转换为字符串 |
| ToString(this Matrix3D, string, string, string, int, CultureInfo) | 将矩阵转换为字符串 |
| ConvertToString(this Matrix3D, string, int) | 将矩阵转换为字符串 |
| ConvertToString(this Matrix3D, string, string, string, in, CultureInfo) | 将矩阵转换为字符串 |

### （3）Point3DExtensions（point3D扩展）

方法：

|  |  |
| --- | --- |
| DistanceTo(this Point3D, Point3D) | 计算从p1到p2的距离。 |
| DistanceToSquared(this Point3D, Point3D) | 计算从p1到p2的平方距离。 |
| ToVector3D(this Point3D) | 转换Point3D到Vector3D |
| Multiply(this Point3D, double) | 将指定点与标量相乘 |
| Sum(params Point3D[]) | 计算指定点 |

### （4）StringExtensions(字符串扩展)

方法：

|  |  |
| --- | --- |
| SplitOnWhitespace(this string) | 用空格拆分字符串 |
| EnumerateToString(this IEnumerable, string, string) | 从可枚举的项目中创建一个字符串 |

### （5）Vector3DExtensions（Vector3D扩展）

方法：

|  |  |
| --- | --- |
| FindAnyPerpendicular(this Vector3D) | 找到指定向量的垂直向量 |
| IsUndefined(this Vector3D) | 确定指定的向量是否未定义（NaN，NaN，NaN） |
| ToPoint3D(this Vector3D) | 转换Vector3D到Point3D |

## 7.Geometry（几何形状）

### （1）BoundingSphere（边界球）

属性：

|  |  |
| --- | --- |
| Center | 圆的中心点 |
| Radius | 圆的直径 |

方法：

|  |  |
| --- | --- |
| CreateFromPoints(IEnumerable<Point3D>) | 从一组点创建一个有界的球体 |
| CreateFromRect3D(Rect3D) | 从 “Rect3D”创建一个 “BoundingSphere”。 |
| CreateMerged(BoundingSphere, BoundingSphere) | 创建一个合并的边界球体 |
| Contains(Point3D) | 确定指定点是否在球体内 |
| DistanceFrom(Point3D) | 计算从点到球面上最近点的距离。 |
| Intersects(BoundingSphere) | 确定球体是否与指定球体相交 |
| RayIntersection(Ray3D, out Point3D[]) | 获取与指定光线的交点 |

### （2）LineSegment（线段）

属性：

|  |  |
| --- | --- |
| P1 | 获取线段的第一个点 |
| P2 | 获取线段的第二个点 |

方法：

|  |  |
| --- | --- |
| AreLineSegmentsIntersecting(Point, Point, Point, Point) | 检查两个线段是否有交点 |
| IntersectsWith(LineSegment) | 指示指定的线段是否与当前线段相交。 |

### （3）Mesh3D（3D网格）

属性：

|  |  |
| --- | --- |
| Edges | 获得边缘 |
| Faces | 获得面 |
| TextureCoordinates | 获得纹理坐标 |
| Vertices | 获取顶点 |

方法：

|  |  |
| --- | --- |
| AddFace(params int[]) | 添加面 |
| Clone() | 创建一个新对象，该对象是当前实例的副本 |
| FindCentroid(int) | 找到指定面的质心 |
| FindFaceFromEdge(int, int) | 从边缘找到面 |
| GetBounds() | 获得边界 |
| GetFaceNormal(int) | 获得面的向量 |
| GetNeighbourVertices(int) | 获得相邻顶点 |
| IsQuadrilateralMesh() | 确定网格是否仅包含四边形面 |
| IsTriangularMesh() | 确定网格是否仅包含三角形面 |
| Quadrangulate() | 将网格转换为四边形网格 |
| ToMeshGeometry3D(bool, double, List<int>) | 将网格转换为MeshGeometry3D |
| Triangulate(bool ) | 对指定的网格进行三角测量 |
| UpdateEdges() | 更新边缘 |
| FindMidpoint(int, int) | 找到指定边的中点 |

### （4）Plane3D（平面3D）

属性：

|  |  |
| --- | --- |
| Normal | 法线 |
| Position | 位置 |

方法：

|  |  |
| --- | --- |
| LineIntersection(Point3D, Point3D) | 找到平面和一条线之间的交点 |

### （5）Polygon(2D多边形)

属性：

|  |  |
| --- | --- |
| Points | 获取或设置点 |

方法：

|  |  |
| --- | --- |
| Triangulate（） | 使用扫描线算法对多边形进行三角测量 |

### （6）Polygon3D（3D多边形）

属性：

|  |  |
| --- | --- |
| Points | 获取或设置点 |

方法：

|  |  |
| --- | --- |
| Flatten() | 使多边形为平面 |
| GetNormal() | 获取多边形的法线 |
| IsPlanar() | 确定这个多边形是否是平面的 |

### （7）Ray3D（3D伞形）

属性:

|  |  |
| --- | --- |
| Direction | 设置方向 |
| Origin | 设置起始点 |

事件：

|  |  |
| --- | --- |
| GetNearest(Point3D ) | 得到最接近指定点的射线点 |
| PlaneIntersection(Point3D,Vector3D) | 找到与平面的交点 |
| PlaneIntersection(Point3D, Vector3D , out Point3D) | 找到与平面的交点，如果找到返回True |

### （8）Triangle（二维平面三角形）

属性：

|  |  |
| --- | --- |
| P1 | 得到三角形第一个点 |
| P2 | 得到三角形第二个点 |
| P3 | 得到三角形第三个点 |

方法：

|  |  |
| --- | --- |
| IsCompletelyInside(Rect rect) | 检查指定的矩形是否完全在当前三角形内 |
| IsRectCompletelyInside(Rect rect) | 检查指定的矩形是否完全在当前三角形内 |
| IsPointInside(Point p) | 检查指定点是否在三角形内 |
| IntersectsWith(Rect rect) | 指示指定的矩形是否与当前三角形相交 |

## 8.Helpers

### （1）ScreenSpace

#### （1.1）Billboard

#### （1.2）BillboardGeometryBuilder（为Billboard的集合构建网格图形）

方法：

|  |  |
| --- | --- |
| CreateIndices(int) | 创建三角形索引 |
| GetPositions(IList<Billboard>, Vector) | 使用当前屏幕转换获取Billboard位置。 |
| GetPinPositions(IList<Billboard>, Vector, double) | 使用当前屏幕转换获取Billboard位置 |

#### （1.3）CohenSutherlandClipping（提供针对矩形的线剪辑）

方法：

|  |  |
| --- | --- |
| ClipLine(ref double x0, ref double y0, ref double x1, ref double y1) | 使用（xmin，ymin）和（xmax，ymax）对角地限制的剪辑矩形来计算点（x，y）的位代码 |
| IsInside(double x, double y) | 根据位置返回一个值 |
| ComputeOutCode(double, double) | 计算出位代码 |

#### （1.4）LineGeometryBuilder（构建线段集合的网格几何图形）

方法：

|  |  |
| --- | --- |
| CreateIndices(int) | 创建三角形索引。 |
| CreatePositions(IList<Point3D>,double, double, ohenSutherlandClipping) | 创建指定线段的位置 |

#### （1.5）PointGeometryBuilder（构建点集合的几何形状）

方法：

|  |  |
| --- | --- |
| CreateIndices(int) | 创建三角形索引。 |
| CreatePositions(IList<Point3D>,double, double, ohenSutherlandClipping) | 创建指定线段的位置 |

#### （1.6）ScreenGeometryBuilder（用于构建平面几何形状的基类）

方法：

|  |  |
| --- | --- |
| UpdateTransforms() | 更新变换 |

### （2）WeakEventListeners

#### （2.1）RenderingEventListener（提供将CompositionTarget.Rendering事件传递给指定处理程序的弱事件监听器）

#### （2.2）WeakEventListener{TEventManager,TEventArgs}（提供一个弱事件监听器，它将指定事件管理器的事件传递给指定的事件处理程序）

方法：

|  |  |
| --- | --- |
| IWeakEventListener.ReceiveWeakEvent(Type, object, EventArgs ) | 从集中式事件管理器接收事件 |

### （3）WeakEventManagers

#### （3.1）AttachedProperties（提供附加属性）

方法：

|  |  |
| --- | --- |
| GetName(this DependencyObject) | 得到模型名称 |
| SetName(this DependencyObject, string) | 设置模型名称 |

#### （3.2）BrushHelper（提供创建画笔的方法）

方法：

|  |  |
| --- | --- |
| ChangeOpacity(Brush, double) | 创建具有指定透明度的画笔的副本 |
| CreateGradientBrush(params Color[]) | 从给定的颜色创建渐变画笔 |
| CreateGradientBrush(IList<Color>, bool) | 从颜色列表中创建渐变画笔 |
| SolidColorBrush CreateGrayBrush(double) | 创建一个灰色的画笔 |
| CreateHsvBrush(double, bool) | 创建一个HSV的画笔 |
| CreateRainbowBrush(bool) | 创建彩虹画刷 |
| CreateSteppedGradientBrush(IList<Color>, bool) | 从颜色列表中创建“渐变”画笔。 |
| CreateSteppedGradientBrush(LinearGradientBrush) | 创建阶梯式渐变画笔 |
| CreateSteppedGradientBrush(LinearGradientBrush, int) | 创建阶梯式梯度刷 |
| CreateSteppedHsvBrush(int) | 创建一个逐步的HSV画刷 |

#### （3.3）CameraHelper（提供Camera派生类的扩展方法）

方法：

|  |  |
| --- | --- |
| AnimateTo(this ProjectionCamera, Point3D, Vector3D, Vector3D, double) | 动画相机的位置和方向 |
| AnimateWidth(this OrthographicCamera, double, double) | 动画正交宽度 |
| ChangeDirection(this ProjectionCamera, Vector3D, Vector3D, double) | 改变相机方向 |
| Copy(this ProjectionCamera, ProjectionCamera, bool) | 复制指定的相机，如有必要，转换视场/宽度 |
| CopyDirectionOnly(this ProjectionCamera, ProjectionCamera, double) | 复制相机源的方向 |
| GetInfo(this Camera) | 获取有关指定摄像头的信息字符串 |
| LookAt(this ProjectionCamera, Point3D, double) | 设置摄像机目标点 |
| LookAt(this ProjectionCamera, Point3D, Vector3D , double) | 设置摄像机目标点和摄像机方向 |
| LookAt(this ProjectionCamera, Point3D, Vector3D, Vector3D, double) | 设置摄像机目标点和方向 |
| LookAt(this ProjectionCamera, Point3D, double, double) | 设置摄像机目标点和相机距离 |
| Reset(this Camera) | 重置指定的正摄相机 |
| Reset(this PerspectiveCamera) | 重置指定的正摄相机 |
| Reset(this OrthographicCamera) | 重置指定的正摄相机 |
| GetViewMatrix(this Camera) | 获取相机的视图变换矩阵 |
| GetProjectionMatrix(this Camera, double) | 获取指定相机的投影矩阵 |
| GetTotalTransform(this Camera, double) | 获得组合的视图和投影变换 |
| GetInverseTransform(this Camera, double) | 获取逆相机变换 |
| FitView(this ProjectionCamera, Viewport3 double) | 适合当前视图中的当前场景 |
| FitView(this ProjectionCamera, Viewport3D, Vector3D, Vector3D, double) | 适合当前视图中的当前场景 |
| ZoomExtents(this ProjectionCamera, Viewport3D, double) | 缩放以适应指定视口的范围 |
| ZoomExtents(this ProjectionCamera, Viewport3, Rect3D, double) | 缩放以适应指定的边界矩形 |
| FitView(this ProjectionCamera, Viewport3D, Rect3D, Vector3D, Vector3D, double) | 适合当前视图中指定的边界矩形 |
| ZoomExtents(ProjectionCamera, Viewport3D, Point3D, double, double) | 缩放以适应指定的球体 |
| FitView(ProjectionCamer, Viewport3D, Point3D, double, Vector3D, Vector3D, double) | 使视图适合于边界球 |
| ZoomToRectangle(this ProjectionCamera, Viewport3D, Rect) | 将相机缩放到指定的矩形 |

#### （3.4）CanonicalSplineHelper

方法：

|  |  |
| --- | --- |
| CreateSpline(IList<Point3D>, double, IList<double>, bool , double) | 创建点的样条 |
| Segment(IList<Point3D>, Point3D, Point3D, Point3D, Point3D, double, double ,double) | 分割 |

#### （3.5）ColorHelper（颜色辅助方法）

方法：

|  |  |
| --- | --- |
| ChangeAlpha(this Color, byte) | 更改颜色的alpha值 |
| ChangeIntensity(this Color, double) | 改变强度 |
| ColorDifference(Color, Color) | 计算两个颜色之间的差异 |
| ColorToHex(Color) | 将“Color”转换为十六进制字符串 |
| ColorToHsv(Color) | 将“Color”转换为HSV值（双精度） |
| ColorToHsvBytes(Color) | 将“Color”转换为HSV值（字节） |
| ColorToUint(Color) | 将“Color”转换为无符号整形 |
| Complementary(Colo) | 计算互补色 |
| HexToColor(string) | 将十六进制字符串转换为 “Color”。 |
| HsvToColor(byte, byte, byte) | 将HSV转换为 “Color”。 |
| HsvToColor(double[]) | 从指定的HSV创建一个颜色。 |
| HsvToColor(double, double , double) | 从HSV转换为Color |
| HueDifference(Color, Color) | 计算两个 “Color”之间的色相差 |
| Interpolate(Color, Color, double) | 两个“Color”之间的线性插值 |
| UIntToColor(uint) | 转换无符号整型到Color |

#### （3.6）ContourHelper（提供通过3顶点平面计算轮廓面的功能。）

方法：

|  |  |
| --- | --- |
| ContourFacet(int, int, int, out Point3D[], out Vector3D[], out Point[], out int[]) | 通过3顶点构面创建轮廓面 |
| CalculatePoint(double, double, double, double) | 计算一个新的点坐标 |
| GetContourFacet() | 获取当前构面的 “ContourFacetResult”。 |
| SetData(int, int, int) | 初始化面数据，并从指定的三角形索引中计算出 “sides”值 |
| CreateNewPosition(int, int) | 计算由两个三角形索引指定的边的平面交叉点的位置 |
| CreateNewNormal(int, int) | 计算由两个三角形索引指定的边的平面交叉点的法线 |
| CreateNewTexture(int, int) | 计算由两个三角形索引指定的边的平面交点处的纹理坐标。 |
| IsSideAlone(int) | 确定指定索引处的顶点是否位于其他两个顶点的相反侧。 |
| AllSidesBelowContour() | 确定平面的所有边是否在轮廓之下 |

#### （3.7）DependencyPropertyEx（提供通用依赖属性寄存器方法）

#### （3.8）DoubleKeyDictionary（双键字典）

属性：

|  |  |
| --- | --- |
| OuterDictionary | 获取或设置OuterDictionary |

方法：

|  |  |
| --- | --- |
| Clear() | 清除字典 |
| Add(K key1, T key2, V value) | 添加指定键值 |
| ContainsKey(K index1, T index2) | 确定指定的字典是否包含该键 |
| Equals(DoubleKeyDictionary<K, T, V>) | 等于指定值 |
| GetEnumerator() | 获取枚举器 |
| Remove(K key1, T key2) | 移除指定键值 |
| IEnumerable.GetEnumerator() | 返回一个遍历集合的枚举器 |

类：DoubleKeyPairValue

属性：

|  |  |
| --- | --- |
| Key1 | 键值1 |
| Key2 | 键值2 |
| Value | 获取或设置值 |

方法：

|  |  |
| --- | --- |
| ToString() | 返回代表此实例的 “System.String”。 |

#### （3.9）ElementSortingHelper（通过透明度和与相机的距离对元素进行排序。）

方法：

|  |  |
| --- | --- |
| AlphaSort(Point3D, Model3DCollection, Transform3D) | 从近到远进行排序，当场景重新排序时使用不同透明度 |
| GetDistanceSquared(Point3D, Visual3D) | 获得距离平方 |
| GetDistanceSquared(Point3D, GeometryModel3D) | 获得距离平方 |
| IsTransparent(Visual3D) | 确定指定的视觉是否透明 |
| IsTransparent(Model3D) | 确定指定的模型是否透明 |
| IsTransparent(GeometryModel3D) | 确定指定的模型是否透明 |
| IsTransparent(Materia) | 确定指定材料的任何部分是否透明 |
| IsTransparent(Brush) | 确定指定的画笔是否透明。 |
| SortModel(Point3D, IList<Visual3D>) | 排序场景 - 第一个不透明对象，然后按照距离相机排列的透明对象 |

#### （3.10）GradientBrushes（包含一组预定义的渐变画笔）

属性：

|  |  |
| --- | --- |
| BlueWhiteRed | 蓝白色渐变刷。 |
| Hue | 色相梯度刷 |
| HueStripes | 12色条纹的色相梯度刷 |
| Rainbow | 彩虹渐变刷 |
| RainbowStripes | 12条彩虹刷 |

#### （3.12）GZipHelper（提供使用Gzip流压缩文件的方法）

方法：

|  |  |
| --- | --- |
| Compress(string) | 使用标准的zlib压缩压缩文件 |

#### （3.13）IBoundsIgnoredVisual3D（定义visual3D类型，它的边界将被忽略）

#### （3.14）LoopSubdivision（细分方案 Loop，Warren）

属性：

|  |  |
| --- | --- |
| Scheme | 获取或设置细分方案 |
| Positions | 获得位置 |

方法：

|  |  |
| --- | --- |
| ToMeshGeometry3D() | 创建几何形状 |
| ToMesh3D() | 创建网格 |
| Add(int v0, in, int) | 增加三角形 |
| GetEdgeVertice(int, int, int ) | 获得一个边缘点 |
| SetEdgeVertice(int, int, int, int) | 设置一个边缘点 |
| Sum(IEnumerable<int>) | 指定向量的和 |
| Sqr(double) | 平方值 |
| Subdivide(int) | 细分这个实例 |
| Subdivide() | 细分这个实例 |
| AddEdgeVertice(int, int, int) | 添加一个边缘顶点 |

#### （3.15）MaterialHelper（提供创建材料的方法）



创建该对象的方法：  
 public Flag(string image)

{

Mesh = new MeshGeometry3D();

Material = MaterialHelper.CreateImageMaterial(image);

Damping = 0.98;

integrator = new VerletIntegrator() { Iterations = 4, Damping = this.Damping };

WindSpeed = 6;

WindDirection = 180;

PoleHeight = 12;

Height = 3;

Length = 4;

Mass = 0.8;

}

private void CreateFlag()

{

Flag = new Flag("Examples/Cloth/FlagOfNorway.png");

Flag.Init();

FlagModel = new GeometryModel3D

{

Geometry = Flag.Mesh,

Material = Flag.Material,

BackMaterial = Flag.Material

};

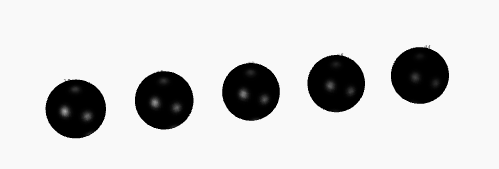
}

方法：

|  |  |
| --- | --- |
| ChangeOpacity(Material, double) | 改变材料透明度 |
| CreateImageMaterial(string, double, UriKind, bool) | 从指定的位图文件创建一个材质 |
| CreateImageMaterial(BitmapImage, double, bool) | 从指定的图像创建材质 |
| CreateEmissiveImageMaterial(string, Brush, UriKind, bool) | 创建发光图片材料。 |
| CreateEmissiveImageMaterial(BitmapImage, Brush, bool) | 从指定的图像创建发光材料 |
| CreateMaterial(Color) | 创建指定颜色的材料 |
| CreateMaterial(Color, double) | 创建指定颜色和透明度的材料 |
| CreateMaterial(Brush, double, byte, bool) | 使用指定的画笔创建材质为漫反射材质 |
| CreateMaterial(Brush, double, double, byte, bool) | 使用指定的画笔创建材质为漫反射材质 |
| CreateMaterial(Brush, Brush, Brush, double, double, bool) | 使用指定的漫反射和镜面光创建材质 |
| GetFirst<T>(Material) | 获取指定类型的第一个材料 |
| GetImage(string, UriKind) | 从指定的uri获取图像 |

#### （3.16）Materials(包含一组预定义的材料)

镜面强度：



<helix:SphereVisual3D Center="-6,0,0" Radius="1" Material="{helix:Material Black, SpecularIntensity=0.4}"/>

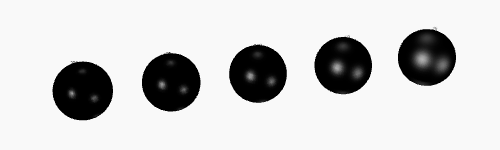
<helix:SphereVisual3D Center="-3,0,0" Radius="1" Material="{helix:Material Black, SpecularIntensity=0.6}"/>

<helix:SphereVisual3D Center="0,0,0" Radius="1" Material="{helix:Material Black, SpecularIntensity=0.8}"/>

<helix:SphereVisual3D Center="3,0,0" Radius="1" Material="{helix:Material Black, SpecularIntensity=0.9}"/>

<helix:SphereVisual3D Center="6,0,0" Radius="1" Material="{helix:Material Black, SpecularIntensity=1.00}"/>

镜面光焦度：



<helix:SphereVisual3D Center="-6,0,0" Radius="1" Material="{helix:Material Black, SpecularPower=25}"/>

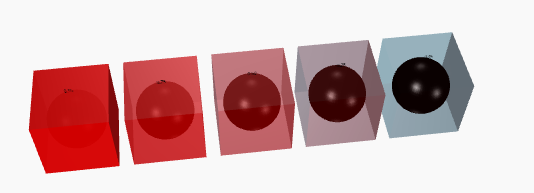
<helix:SphereVisual3D Center="-3,0,0" Radius="1" Material="{helix:Material Black, SpecularPower=50}"/>

<helix:SphereVisual3D Center="0,0,0" Radius="1" Material="{helix:Material Black, SpecularPower=100}"/>

<helix:SphereVisual3D Center="3,0,0" Radius="1" Material="{helix:Material Black, SpecularPower=150}"/>

<helix:SphereVisual3D Center="6,0,0" Radius="1" Material="{helix:Material Black, SpecularPower=200}"/>

透明度：



<helix:CubeVisual3D Center="-6,0,0" SideLength="2.5" Material="{helix:Material Red, Opacity=0.05}"/>

<helix:CubeVisual3D Center="-3,0,0" SideLength="2.5" Material="{helix:Material Red, Opacity=0.25}"/>

<helix:CubeVisual3D Center="0,0,0" SideLength="2.5" Material="{helix:Material Red, Opacity=0.5}"/>

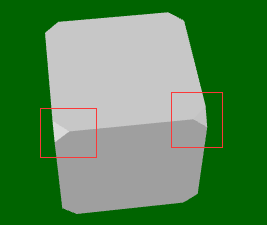
<helix:CubeVisual3D Center="3,0,0" SideLength="2.5" Material="{helix:Material Red, Opacity=0.75}"/>

<helix:CubeVisual3D Center="6,0,0" SideLength="2.5" Material="{helix:Material Red, Opacity=0.95}"/>

属性：

|  |  |
| --- | --- |
| BlackMaterial | Black |
| DarkGrayMaterial | DarkGray |
| GrayMaterial | Gray |
| LightGrayMaterial | LightGray |
| WhiteMaterial | White |
| HueMaterial | Hue |
| RainbowMaterial | Rainbow |
| RedMaterial | Red |
| OrangeMaterial | Orange |
| YellowMaterial | Yellow |
| GreenMaterial | Green |
| BlueMaterial | Blue |
| IndigoMaterial | Indigo |
| VioletMaterial | Violet |
| BrownMaterial | Brown |
| GoldMaterial | Brown |

#### （3.17）MeshBuilder（构建MeshGeometry3D对象）



倒角立方体：

private ModelVisual3D CreateDice()

{

var diceMesh = new MeshBuilder();

diceMesh.AddBox(new Point3D(0, 0, 0), 1, 1, 1);

for (int i = 0; i < 2; i++)

for (int j = 0; j < 2; j++)

for (int k = 0; k < 2; k++)

{

var points = new List<Point3D>();

diceMesh.ChamferCorner(new Point3D(i - 0.5, j - 0.5, k - 0.5), 0.1, 1e-6, points);

//foreach (var p in points)

// b.ChamferCorner(p, 0.03);

}

return new ModelVisual3D { Content = new GeometryModel3D { Geometry = diceMesh.ToMesh(), Material = Materials.White } };

}

属性：

|  |  |
| --- | --- |
| Normals | 获取网格的法向量 |
| Positions | 获取网格的位置集合 |
| TextureCoordinates | 获取网格的纹理坐标 |
| TriangleIndices | 获取三角形索引 |
| CreateNormals | 获取或设置一个指示是否创建法向量的值 |
| CreateTextureCoordinates | 获取或设置一个指示是否创建纹理坐标的值 |

方法：

|  |  |
| --- | --- |
| GetCircle(int, bool) | 获取一个圆形 |
| GetCircleSegment(int, double I, double) | 获取一个圆圈段。 |
| AddArrow(Point3D, Point3D, double, double, int) | 向网格添加箭头 |
| AddBoundingBox(Rect3, double) | 将边框的边添加为圆柱 |
| AddBox(Point3D, double, double, double) | 添加与X，Y和Z轴对齐的框 |
| AddBox(Rect3D) | 添加与X，Y和Z轴对齐的框 |
| AddBox(Point3D, double, double, double, BoxFaces) | 添加一个带有指定面的框，与X，Y和Z轴对齐 |
| AddBox(Point3D, Vector3D, Vector3D y, double xlength, double ylength, double, BoxFaces) | 添加一个带指定面的框，与指定的轴对齐 |
| AddCone(Point3, Vector3D, doubl, double , double, bool, bool,int) | 添加（可能带头）的锥体 |
| AddCone(Point3D, Point3D, double, bool, int) | 添加锥体 |
| AddCubeFace(Point3D, Vector3D, Vector3D, double, double, double) | 添加立方体面 |
| AddCylinder(Point3D, Point, double, int) | 向网格添加圆柱体 |
| AddDodecahedron(Point3D, Vector3D, Vector3D, double) | 生成十二面体 |
| AddEdges(IList<Point3D>, IList<int>, double , int) | 将边缘集合添加为圆柱体 |
| AddExtrudedGeometry(IList<Point>, Vector3D, Point3D, Point3D) | 添加指定曲线的挤压表面 |
| AddPolygon(IList<Point>, Vector3D, Vector3D, Point3D) | 添加多边形 |
| AddExtrudedSegments(IList<Point>, Vector3D, Point3, Point3D) | 添加指定线段的挤压表面 |
| AddLoftedGeometry(IList<IList<Point3D>>,  IList<IList<Vector3D>>, IList<IList<Point>>) | 添加一个抬高的表面 |
| AddNode(Point3D, Vector3D, Point) | 添加一个节点 |
| AddPipe(Point3D, Point3D, double, double, int) | 添加（可能是中空的）管道 |
| AddPolygon(IList<Point3D>) | 添加多边形 |
| AddPyramid(Point3D, double, double, bool) | 添加一个金字塔 |
| AddPyramid(Point3D, Vector3D, Vector3D, double, double , bool) | 添加一个金字塔 |
| AddOctahedron(Point3D, Vector3, Vector3, double, double) | 添加八面体 |
| AddTetrahedron(Point3D, Vector3D, Vector3D, double) | 添加四面体 |
| AddQuad(Point3D, Point3D, Point3D, Point3D) | 添加四边形多边形 |
| AddQuad(Point3, Point3D, Point3D, Point3D, Point, Point, Point, Point) | 添加四边形多边形 |
| AddQuads(IList<Point3D>, IList<Vector3D>, IList<Point>) | 添加四边形多边形列表 |
| AddRectangularMesh(IList<Point3D>, int) | 添加一个矩形网格（m×n个点） |
| AddRectangularMesh(Point3D[,], Point[,], bool, bool) | 添加由二维数组阵列定义的矩形网格。 |
| AddRegularIcosahedron(Point3D, double, bool) | 添加常规二十面体 |
| AddSurfaceOfRevolution(Point3D, Vector3D, IList<PointIList<int>,IList<double>) | 增添旋转的表面 |
| AddRevolvedGeometry(IList<Point>, IList<double>, Point3D, Vector3D, int) | 增添旋转的表面 |
| AddSubdivisionSphere(Point3D, double, int) | 添加一个球体（通过细分一个常规的二十面体） |
| AddSphere(Point3D, double, int, int) | 添加一个球体 |
| AddEllipsoid(Point3D, double, double, double, int, int) | 添加椭球 |
| AddTriangle(Point3D, Point3D, Point3D) | 添加一个三角形 |
| AddTriangle(Point3D, Point3D, Point3D, Poin, Point, Point) | 添加一个三角形 |
| AddTriangleFan(IList<int>) | 添加一个三角形风扇 |
| AddTriangleFan(IList<Point3D>, IList<Vector3D>, IList<Point>) | 向网格添加三角形风扇 |
| AddTriangleStrip(IList<Point3D>, IList<Vector3D>, IList<Point>) | 向网格添加三角形条 |
| AddPolygon(IList<int>) | 添加一个由顶点索引指定的多边形（使用三角形风扇） |
| AddPolygonByCuttingEars(IList<int>) | 添加由顶点索引定义的多边形 |
| AddPolygonByTriangulation(IList<int>) | 添加由顶点索引定义的多边形（使用扫描线算法） |
| AddTriangles(IList<Point3D>,IList<Vector3D>, IList<Point>) | 添加一个三角形列表 |
| AddTube(IList<Point3D>, double[], double[], int, bool) | 添加管道 |
| AddTube(IList<Point3D>, double, int, bool) | 添加管道 |
| AddTube(IList<Point3D>, IList<double>, IList<double>, IList<Point>, boo, bool) | 添加一个带有自定义部分的管 |
| AddTube(IList<Point3D>, IList<double>, IList<double>, IList<double>, IList<Point>, Vector3D, bool, bool) | 添加一个带有自定义部分的管 |
| Append(MeshBuilder) | 添加指定网格 |
| Append(MeshGeometry3D mesh) | 添加指定网格 |
| Append(IList<Point3D>,IList<int> IList<Vector3D>, IList<Point> ) | 附加指定的点和三角形 |
| ChamferCorner(Point3D, double, double, IList<Point3D>) | 倒角立方体 |
| CheckPerformanceLimits() | 检查性能限制 |
| Scale(double, double double) | 缩放位置（和法向量） |
| SubdivideLinear(bool) | 执行网格的线性细分 |
| ToMesh(bool) | 将几何转换为 “MeshGeometry3D”。 |
| GetUnitSphere(int) | 从缓存获取单位球体 |
| AddRectangularMeshNormals(int index0, int, int s) | 为矩形网格添加法向量 |
| AddRectangularMeshTextureCoordinates(int, int) | 添加矩形网格的纹理坐标。 |
| AddRectangularMeshTriangleIndices(int, int rows, int, bool) | 为矩形网格添加三角形索引 |
| AddRectangularMeshTriangleIndices(int, int, int, bool, bool) | 为矩形网格添加三角形索引 |
| FindCornerNormal(Point3D, double) | 查找指定角的平均值 |
| NoSharedVertices() | 确保没有三角形共享相同的顶点 |
| Subdivide4() | 将每个三角形细分为四个子三角形。 |
| SubdivideBarycentric() | 将每个三角形细分为六个三角形。 在每个三角形的中点添加一个顶点。 |
| AddTorus(double, double, int, int ) | 添加环面 |

#### （3.18）MeshGeometryHelper

方法：

|  |  |
| --- | --- |
| CalculateNormals(IList<Point3D>, IList<int>) | 计算法向量 |
| CalculateNormals(MeshGeometry3D) | 计算法向量 |
| FindBorderEdges(MeshGeometry3D) | 找到只连接到一个三角形的边 |
| FindEdges(MeshGeometry3D) | 查找网格中的所有边缘（每个边缘仅包含一次） |
| FindSharpEdges(MeshGeometry3D, double) | 找到相邻三角形法向量之间的角度的所有边。 |
| NoSharedVertices(MeshGeometry3D) | 创建一个没有顶点共享的新的网格 |
| Simplify(MeshGeometry3D, double) | 简化指定的网格 |
| Validate(MeshGeometry3D) | 验证指定的网格 |
| Cut(MeshGeometry3D, Point3, Vector3D) | 用指定的平面切割网格 |
| GetContourSegments(MeshGeometry3D, Point3D, Vector3D) | 获取轮廓段 |
| CombineSegments(IList<Point3D>, double) | 组合细分 |
| CreateKey(uin, uint) | 从两个32位索引创建一个64位密钥 |
| ReverseKey(ulong, out uint, out uint) | 从64位密钥提取两个32位索引 |
| FindConnectedSegment(IList<Point3D>, Point3, double) | 查找到指定点的最近连接的段 |

#### （3.19）Model3DHelper（为Model3D对象提供扩展方法）

方法：

|  |  |
| --- | --- |
| GetTransform(this Model3D, Model3D, Transform3D) | 得到变换 |
| Traverse<T>(this Model3D, Action<T, Transform3D>) | 遍历Model3D树，并在指定类型的每个Model3D上调用指定的操作。 |
| Traverse<T>(this Model3D, Transform3D, Action<T, Transform3D>) | 遍历Model3D树，并在指定类型的每个Model3D上调用指定的操作。 |

#### （3.20）StereoHelper（包含立体视图的辅助方法）

方法：

|  |  |
| --- | --- |
| CalculateStereoBase(double, double, double, double, double) | 使用完整的Bercovitz公式计算立体声基座 |
| CalculateStereoBase(double, double, double, double) | 使用完整的Bercovitz公式计算立体声基座 |
| CreateClone(Visual3D) | 创建Visual3D的克隆 |
| FindFocalLength(double, double) | 找到给定视场和格式的焦距 |
| UpdateStereoCameras(PerspectiveCamera, PerspectiveCamera, PerspectiveCamera, doubl bool, bool, bool) | 根据中央相机更新左右相机 |

#### （3.21）TextCreator（创建文本标签模型或视觉效果。）

方法：

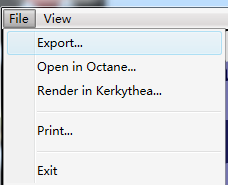
|  |  |
| --- | --- |
| CreateTextLabelModel3D(string, Brush, bool, double, Point3D, Vector3D, Vector3D) | 创建一个包含文本标签的Visual3D元素。 |
| CreateTextLabelModel3D(string, Brus, bool, double, Point3D, Vector3D, Vector3D) | 为文本标签创建模型 |

#### （3.22）Transform3DHelper（Transform3D的辅助方法

方法：

|  |  |
| --- | --- |
| CombineTransform(Transform3D, Transform3D) | 组合两种变换 |

#### （3.23）Viewport3DHelper（提供Viewport3D的扩展方法）



private void Export\_Click(object sender, RoutedEventArgs e)

{

var d = new SaveFileDialog();

d.Filter = Exporters.Filter;

d.DefaultExt = Exporters.DefaultExtension;

if (!d.ShowDialog().Value)

{

return;

}

Viewport3DHelper.Export(view1.Viewport, d.FileName);

}

方法：

|  |  |
| --- | --- |
| Copy(this Viewport3D, int) | 将指定的视口复制到剪贴板 |
| Copy(this Viewport3D, double, doubl, Brush, int) | 将指定的视口复制到剪贴板 |
| CopyXaml(this Viewport3D) | 将视口复制为XMAL剪贴板 |
| Export(this Viewport3D, string, Brus) | 导出指定的视口。 |
| ExportStereo(this Viewport3D, string, double, Brush) | 导出指定的视口。 |
| FindHits(this Viewport3D, Point) | 找到指定命中的位置 |
| FindHits(this Viewport3D, Rect, SelectionHitMode) | 找到指定命中的矩形 |
| FindNearest(this Viewport3D, Point, out Point3D, out Vector3D, out DependencyObject) | 找到最近的视觉，命中点及其在指定位置的法线 |
| FindNearestPoint(this Viewport3D, Point) | 找到指定位置处最近点的坐标 |
| FindNearestVisual(this Viewport3D, Point) | 找到距离指定位置最近的Visual3D。 |
| GetCameraTransform(this Viewport3DVisual) | 获取相机变换 |
| GetCameraTransform(this Viewport3) | 获取相机变换（视口和投影） |
| GetLights(this Viewport3D) | 获得所有光 |
| GetRay(this Viewport3D, Point) | 在指定位置获取光线 |
| GetTotalTransform(this Viewport3DVisual) | 获取总体变换（摄像机和视口） |
| GetTotalTransform(this Viewport3D) | 获取总体变换（摄像机和视口） |
| GetTransform(this Viewport3D, Visual3D) | 获取指定视觉的总体变换。 |
| GetViewMatrix(this Viewport3D) | 获取视图矩阵 |
| GetProjectionMatrix(this Viewport3D) | 获取投影矩阵 |
| GetViewportTransform(this Viewport3DVisual) | 获取视口转换。 |
| GetViewportTransform(this Viewport3D) | 获取视口转换。 |
| Point2DtoPoint3D(this Viewport3D, Point, out Point3D, out Point3D) | 将位置转换为近距离剪切平面处的Point3D |
| Point2DtoRay3D(this Viewport3D, Point) | 将2D点转换为射线。 |
| Point3DtoPoint2D(this Viewport3D, Point3D) | 将Point3D转换为Point2D。 |
| Print(this Viewport3D, string) | 打印指定的视口 |
| RenderBitmap(this Viewport3D, Brush, int) | 将视口呈现为位图 |
| RenderBitmap(this Viewport3D, doubl, doubl, Brush, int) | 将视口呈现为位图 |
| ResizeAndArrange(this Viewport3D, double, double) | 调整和排列视口。 |
| SaveBitmap(this Viewport3D, string,rush, int, BitmapExporter.OutputFormat) | 将视口保存到文件。 |
| SaveStereoBitmap(this Viewport3D, string, double, Brush, int) | 将 “Viewport3D”保存到左/右位图 |
| SaveBitmap(this Viewport3D, Stream, Brush, int, BitmapExporter.OutputForma) | 将 “Viewport3D”保存到位图。 |
| SearchFor<T>(this IEnumerable<Visual3D>) | 在给定类型T的对象的Visual3D集合中递归搜索 |
| UnProject(this Viewport3D, Point, Point3D, Vector3D) | 将一个点从屏幕（2D）转换到平面上的一个点（3D） |
| UnProject(this Viewport3D, Point ) | 通过相机目标点将一个点从屏幕（2D）转换到平面上的一个点。 |
| GetTotalNumberOfTriangles(this Viewport3D) | 获取视口中三角形的总数 |
| CopyBitmap(BitmapSource, WriteableBitmap, int, int) | 复制位图 |
| ExportKerkythea(this Viewport3, string, Brush) | 导出为Kerkythe文件 |
| ExportKerkythea(this Viewport3D, string, Brush, in, int) | 导出为Kerkythe文件 |
| ExportObj(this Viewport3D, string) | 导出为OBJ文件 |
| ExportX3D(this Viewport3D, string) | 导出为3D文件 |
| ExportCollada(this Viewport3D, string) | 导出为Collada文件 |
| ExportStl(this Viewport3D, string) | 导出为STL文件 |
| ExportXaml(this Viewport3D, string) | 导出为XAML |
| GetGlobalHitPosition(RayHitTestResult, Viewport3D) | 将命中位置转换为全局（视口）坐标 |
| GetNormalHit(RayMeshGeometry3DHitTestResult) | 获得命中测试结果的法线 |
| SearchFor(IEnumerable<Visual3D>, Type, IList<Model3D) | 递归搜索给定类型的对象 |
| SearchFor(IEnumerable<Model3D>, Type, IList<Model3D>) | 搜索指定类型的型号 |

类：RectangleHitResult

属性：

|  |  |
| --- | --- |
| Model | 获得命中模型 |

类：HitResult

属性：

|  |  |
| --- | --- |
| Distance | 获得距离 |
| Mesh | 获得网格 |
| Model | 获得模型 |
| Normal | 获取或设置法线 |
| Position | 获取或设置位置 |
| RayHit | 获取或设置射线命中 |
| Visual | 获得图形 |

#### （3.24）Visual3DHelper（提供Visual3D对象的扩展方法）

方法：

|  |  |
| --- | --- |
| Find<T>(DependencyObject) | 找到指定类型的第一个子类 |
| FindBounds(this Visual3DCollection) | 找到Visual3D集合的边框。 |
| FindBounds(this Visual3D, Transform3D) | 找到指定视觉的边框 |
| GetTransform(this Visual3D) | 获取指定视觉的总变换 |
| GetViewport3D(this Visual3D) | 从指定的视觉获取父级 “Viewport3D。 |
| GetViewportTransform(this Visual3D) | 获取到视口空间的转换。 |
| IsAttachedToViewport3D(this Visual3D) | 确定视觉是否附加到Viewport3D。 |
| Traverse<T>(this Visual3DCollection, Action<T, Transform3D>) | 遍历Visual3D / Model3D树，并在指定类型的每个Model3D上调用指定的操作。 |
| Traverse<T>(this Visual3, Action<T, Transform3D>) | 遍历Visual3D / Model3D树，并在指定类型的每个Model3D上调用指定的操作。 |
| GetTransformTo(this Visual3D, Model3D) | 获取从指定的Visual3D到指定的Model3D的转换 |
| GetViewport(this Visual3D) | 获取指定视觉的视口。 |
| GetChildren(this Visual3D) | 获取子类 |
| GetModel(this Visual3D) | 获取指定的Visual3D的模型。 |
| Traverse<T>(Visual3D, Transform3D, Action<T, Transform3D>) | 遍历视觉树，并在指定类型的每个对象上调用指定的操作。 |

#### （3.25）XamlHelper（提供帮助方法来生成xaml）

方法：

|  |  |
| --- | --- |
| GetXaml(Viewport3D) | 获取指定视口的xaml。 |
| GetXaml(object) | 获取指定对象的xaml。 |

## 9.Importers（导入）

### （1）IModelReader（模型阅读器接口类）

方法：

|  |  |
| --- | --- |
| Read(string) | 从指定的路径读取模型 |
| Read(Stream) | 从指定的流中读取模型 |

### （2）Importers：

默认文件导入扩展名为.obj；所有支持被导入的文件扩展名为\*.3ds;\*.lwo;\*.obj;\*.objx;\*.stl;\*.off

### （3）LwoReader（灯光对象阅读器）

属性：

|  |  |
| --- | --- |
| Materials | 获取材料 |
| Meshes | 获取网格 |
| Surfaces | 获取表面 |
| Points | 获取点 |

方法：

|  |  |
| --- | --- |
| Read(Stream) | 从指定的流中读取模型 |
| BuildModel() | 构建模型 |
| ReadChunkId(BinaryReader) | 阅读块ID |
| ReadChunkSize(BinaryReader) | 阅读块大小 |
| ReadData(BinaryReader, int) | 读取一个块的数据块 |
| ReadFloat(BinaryReader) | 阅读浮点型 |
| ReadInt(BinaryReader) | 阅读整型 |
| ReadPoints(BinaryReade, int) | 阅读点 |
| ReadPolygons(BinaryReader, int) | 阅读多边形 |
| ReadShortInt(BinaryReaderr) | 阅读短整形 |
| ReadString(BinaryReader, int | 读一个字符串 |
| ReadSurface(BinaryReader, int) | 阅读表面 |

### （4）ModelImporter（从文件导入模型）

属性：

|  |  |
| --- | --- |
| DefaultMaterial | 获取或设置默认材料 |

方法：

|  |  |
| --- | --- |
| Load(string, Dispatcher = null, bool) | 从指定路径导入模型 |

Xaml：

<Window x:Class="Helix\_Car\_Demo.MainWindow"

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

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

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:Helix\_Car\_Demo"

xmlns:helix="http://helix-toolkit.org/wpf"

mc:Ignorable="d"

WindowState="Maximized" WindowStyle="None"

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

<Grid>

<helix:HelixViewport3D Name="viewport3D" Grid.Row="0" ShowViewCube="False" CameraRotationMode="Trackball" InfiniteSpin="False" ZoomExtentsWhenLoaded="True" MouseDoubleClick="viewport3D\_MouseDoubleClick" PreviewMouseWheel="viewPort3D\_PreviewMouseWheel">

<helix:DefaultLights/>

</helix:HelixViewport3D>

</Grid>

Cs：

public partial class MainWindow : Window

{

ModelVisual3D device;

public MainWindow()

{

InitializeComponent();

viewport3D.RotateGesture = new MouseGesture(MouseAction.LeftClick);

device= new ModelVisual3D();

device.Content = getModel(@"TestObj\BrandenburgGate\BrandenburgGate.Obj");

viewport3D.Children.Add(device);

viewport3D.Children.Add(new LightVisual3D());

}

public Model3D getModel(string path)

{

Model3D device = null;

try

{

ModelImporter import = new ModelImporter();

device = import.Load(path);

}

catch (Exception e)

{

e.ToString();

}

return device;

}

}

### （5）ModelReader（模型阅读类）

属性：

|  |  |
| --- | --- |
| Freeze | 获取或设置一个值，该值指示模型是否可用 |
| DefaultMaterial | 获取或设置默认材料 |
| Dispatcher | 获得线程 |
| Directory | 获得文件夹 |
| TexturePath | 获取或设置文本路径 |

方法：

|  |  |
| --- | --- |
| Read(string) | 从指定路径读取模型 |
| Read(Stream ) | 从指定流读取模型 |
| Dispatch(Action) | 在线程上调用指定的操作 |

### （6）ObjReader（Wavefront .obj文件阅读器）

属性：

|  |  |
| --- | --- |
| IgnoreErrors | 获取或设置一个值，指示是否忽略错误 |
| SwitchYZ | 获取或设置一个值，指示是否切换Y和Z坐标 |
| SkipTransparencyValues | 获取或设置一个值，指示是否跳过素材文件中的透明度值。 |
| IsSmoothingDefault | 设置一个值指示平滑是否为默认 |
| Groups | 获取文件的组。 |
| Materials | 获取导入的材料文件中的材料 |
| CurrentMaterial | 获取或设置当前材料 |
| CurrentGroup | 获取当前组。 |
| Normals | 获取或设置法线 |
| Points | 获取或设置点 |
| Reader | 获取或设置流阅读器 |
| TextureCoordinates | 获取或设置纹理坐标 |

方法：

|  |  |
| --- | --- |
| Read(string) | 从指定的路径读取模型。 |
| Read(Strea) | 从指定的流读取模型 |
| ReadZ(string) | 读取GZipStream压缩的OBJ文件 |
| ColorParse(string) | 解析一个颜色的字符串。 |
| DoubleParse(string) | 解析一个包含double值的字符串 |
| Split(string) | 使用空格（输入）分隔指定的字符串作为分隔符 |
| SplitLine(string, out string, out string) | 在关键字和参数中分割一行 |
| AddGroup(string) | 添加具有指定名称的组。 |
| EnsureNewMesh() | 确保创建一个新的网格。 |
| SetSmoothingGroup(string) | 设置平滑组号。 |
| AddFace(string) | 添加一个面 |
| AddNormal(string) | 添加一个法线 |
| AddTexCoord(string) | 添加纹理坐标 |
| AddVertex(strin) | 添加顶点 |
| BuildModel() | 构建模型 |
| GetMaterial(string) | 获取具有指定名称的材料 |
| LoadMaterialLib(string) | 加载材料库。 |
| SetMaterial(string) | 设置当前组的材料 |

类：Group

属性：

|  |  |
| --- | --- |
| Material | 设置材料 |
| MeshBuilder | 获取当前网格的网格构建器。 |
| Name | 获取或设置组名 |

方法：

|  |  |
| --- | --- |
| CreateModels() | 构建模型组 |

类：MaterialDefinition

属性：

|  |  |
| --- | --- |
| AlphaMap | 获取或设置Alpha贴图。 |
| Ambient | 获取或设置环境颜色 |
| AmbientMap | 获取或设置环境贴图。 |
| BumpMap | 获取或设置凹凸贴图 |
| Diffuse | 获取或设置漫反射颜色 |
| DiffuseMap | 获取或设置漫反射贴图 |
| Dissolved | 获取或设置不透明度值。 |
| Illumination | 获取或设置照明。 |
| Name | 获取或设材料名称 |
| Specular | 获取或设置镜面颜色 |
| SpecularCoefficient | 获取或设置镜面系数 |
| SpecularMap | 获取或设置镜面图。 |
| Material | 获取或设置材料 |

方法：

|  |  |
| --- | --- |
| GetMaterial(string) | 从指定的路径获取材料。 |
| CreateMaterial(string) | 创建材料。 |
| CreateTextureBrush(string) | 创建纹理画笔。 |

### （7）OffReader（提供对象文件格式（OFF）阅读器）

属性：

|  |  |
| --- | --- |
| Faces | 获得平面 |
| Vertices | 获得顶点 |

方法：

|  |  |
| --- | --- |
| CreateMesh() | 从加载文件中创建网格 |
| CreateMeshGeometry3D() | 从加载的文件创建一个 “MeshGeometry3D”对象。 |
| CreateModel3D() | 从加载的文件创建一 “Model3DGroup”。 |
| Load(Stream) | 从指定的流中加载模型 |
| Read(Stream) | 从指定的流中读取模型。 |
| GetIntValues(string) | 字符串中解析整数值 |
| GetValues(string) | 从字符串中解析双精值 |
| RemoveComments(string) | 从行中删除注释 |

### （8）StLReader（为StereoLithography.StL文件提供导出）

属性：

|  |  |
| --- | --- |
| Header | 获取文件标题 |
| Materials | 获取材料 |
| Meshes | 获取网格 |

方法：

|  |  |
| --- | --- |
| Read(Stream) | 从指定的流中读取模型。 |
| ToModel3D() | 建立模型 |
| ParseLine(string, out string, out string) | 解析来自指定行的ID和值。 |
| ParseNormal(string) | 解析一个法线的字符串 |
| ReadFloat(BinaryReader) | 读取一个浮点数（4字节） |
| ReadLine(StreamReader, string) | 从流读取器读取一行。 |
| ReadUInt16(BinaryReader) | 读取16位无符号整数。 |
| ReadUInt32(BinaryReader) | 读取32位无符号整数。 |
| TryParseVertex(string, out Point3) | 从字符串中解析顶点 |
| ReadFacet(StreamReaderr, strin) | 阅读一个平面。 |
| ReadTriangle(BinaryReader) | 从二进制STL文件读取三角形。 |
| TryReadAscii(Stream) | 从指定的流中读取ASCII格式的模型。 |
| TryReadBinary(Stream) | 从指定的二进制流中读取模型。 |

### （9）StudioReader（3D Studio文件阅读器）

方法:

|  |  |
| --- | --- |
| Read(Stream) | 从指定的流中读取模型。 |
| ReadChunkId(BinaryReader) | 读取chunk id |
| ReadChunkSize(BinaryReader) | 读取chunk 大小 |
| ReadColor(BinaryReader) | 读取颜色 |
| ReadData(BinaryReader, int) | 读取数据 |
| ReadFaceList(BinaryReader) | 读取平面列表 |
| ReadFaceSets(BinaryReader, int) | 读取平面设置 |
| ReadMatMap(BinaryReader, int) | 读取材料贴图 |
| ReadMaterial(BinaryReader, int) | 读取材料 |
| ReadObject(BinaryReade int) | 读取object |
| ReadString(BinaryReader) | 读取字符串 |
| ReadTexCoords(BinaryReader) | 读取纹理坐标 |
| ReadTransformation(BinaryReader) | 读取转换 |
| ReadTriangularMesh(BinaryReader, int) | 读取三角形网格 |
| ConvertFaceIndices(List<int>, List<int>) | 转换平面索引 |
| ReadVector(BinaryReader) | 读一个向量 |
| ReadVertexList(BinaryReader) | 读取顶点列表。 |

类：Mesh

|  |  |
| --- | --- |
| Positions | 顶点 |
| TriangleIndices | 定义三角形 |
| TextureCoordinates | 纹理坐标 |
| Material | 材料 |
| BackMaterial | 背面材料 |

方法：

|  |  |
| --- | --- |
| CreateModel() | 构建模型 |

类：FaceSet

属性：

|  |  |
| --- | --- |
| Faces | 获取或设置平面 |
| Name | 获取或设置材料名称 |

## 10.Physics

### （1）MarkupExtensions

#### GradientExtension（材料的标记扩展）

方法：

|  |  |
| --- | --- |
| ProvideValue(IServiceProvider) | 返回指定类型的渐变画笔 |

#### ImageBrushExtension（图像画刷的扩展）

方法：

|  |  |
| --- | --- |
| ProvideValue(IServiceProvider) | 在派生类中实现时，返回一个设置为此标记扩展的目标属性值的对象 |

#### ImageMaterialExtension（图像材料的标记扩展）

属性：

|  |  |
| --- | --- |
| IsEmissive | 获取或设置一个值，指示此材料是否发光。 |
| Opacity | 获取或设置不透明度。 |
| UriKind | 获取或设置URI的种类。 |

方法：

|  |  |
| --- | --- |
| ProvideValue(IServiceProvider) | 在派生类中实现时，返回一个设置为此标记扩展的目标属性值的对象 |

#### KeyGestureExtension（关键和鼠标手势的标记扩展）

方法：

|  |  |
| --- | --- |
| ProvideValue(IServiceProvider) | 提供一个值 |

#### LinearGradientBrushExtension（LinearGradientBrush的标记扩展）

方法：

|  |  |
| --- | --- |
| ProvideValue(IServiceProvider) | 返回线性渐变画笔 |

#### MaterialExtension（材料的标记扩展）

属性：

|  |  |
| --- | --- |
| Opacity | 获取或设置不透明度。 |
| SpecularIntensity | 获得或设置镜面强度 |
| SpecularPower | 获取或设置镜面光焦度 |

方法：

|  |  |
| --- | --- |
| ProvideValue(IServiceProvider) | 在派生类中实现时，返回一个设置为此标记扩展的目标属性值的对象。 |

#### PerspectiveCameraExtension（创建透视相机的标记扩展）

属性：

|  |  |
| --- | --- |
| FieldOfView | 视野 |
| LookDirection | 获取或设置相机方向 |
| Position | 获取或设置位置 |
| UpDirection | 获取或设置垂直方向 |

方法：

|  |  |
| --- | --- |
| ProvideValue(IServiceProvider) | 在派生类中实现时，返回一个设置为此标记扩展的目标属性值的对象。 |

#### TranslateExtension（创建翻译变换的标记扩展。）

属性：

|  |  |
| --- | --- |
| Offset | 获取或设置偏移量。 |

方法：

|  |  |
| --- | --- |
| ProvideValue(IServiceProvider) | 在派生类中实现时，返回一个设置为此标记扩展的目标属性值的对象。 |

### （2）VerletIntegrator（提供一个verlet集成器）

属性：

|  |  |
| --- | --- |
| Accelerations | 获取或设置加速度 |
| Constraints | 获取或设置约束。 |
| Damping | 获取或设置阻尼 |
| InverseMass | 获取或设置反质量 |
| Iterations | 获取或设置迭代。 |
| Positions | 获取或设置Positions。 |
| Positions0 | 获取或设置position0。 |

方法：

|  |  |
| --- | --- |
| AddConstraint(int, int, double) | 添加约束 |
| AddFloor(double) |  |
| AddSphere(Point3D, double) | 添加球体 |
| ApplyGravity(Vector3D) | 应用重力。 |
| CreateConstraintsByMesh(MeshGeometry3D, double) | 通过网格创建约束 |
| FixPosition(int) | 修正指定的位置。 |
| Init(MeshGeometry3D) | 进入指定的网格 |
| Resize(int) | 调整数组的大小 |
| SetForce(int, Vector3D) | 设置力 |
| SetInverseMass(double) | 设置反质量 |
| TimeStep(double) |  |
| TransferPositions(MeshGeometry3D) | 转换位置 |
| Integrate(double) | 整合 |
| SatisfyConstraints(int) | 满足约束 |

类：Constraint

方法：

|  |  |
| --- | --- |
| Satisfy(VerletIntegrator, int) | 满足约束 |

类：SphereConstraint

属性：

|  |  |
| --- | --- |
| Center | 获取或设置中心 |
| Index | 获取或设置索引 |
| Radius | 获取或设置半径 |
| RadiusSquared | 获取或设置半径平方。 |

方法：

|  |  |
| --- | --- |
| Satisfy(VerletIntegrator, int) | 满足约束 |

类：FloorConstraint

属性：

|  |  |
| --- | --- |
| Friction | 获得或设置摩擦 |
| Index | 索引 |

方法：

|  |  |
| --- | --- |
| Satisfy(VerletIntegrator, int) | 满足约束 |

类：DistanceConstraint

属性：

|  |  |
| --- | --- |
| Index1 | 索引1 |
| Index2 | 索引2 |
| Iterations | 获取或设置迭代 |
| RelaxationFactor | 获取或设置松弛因子 |
| Restlength | 获取或设置剩余长度 |

方法：

|  |  |
| --- | --- |
| Satisfy(VerletIntegrator, int) | 满足约束 |

## 11.Resources（资源）

## 12.SelectionCommands（选择命令）

### （1）ModelsSelectedByPointEventArgs

提供PointSelectionCommand的ModelsSelected事件的事件数据。

属性：

|  |  |
| --- | --- |
| Position | 获取选择的矩形 |

### （2）ModelsSelectedByRectangleEventArgs

提供RectangleSelectionCommand的ModelsSelected事件的事件数据。

属性：

|  |  |
| --- | --- |
| Rectangle | 获取选择的矩形 |

### （3）ModelsSelectedEventArgs

提供SelectionCommand的ModelsSelected事件的事件数据。

属性：

|  |  |
| --- | --- |
| SelectedModels | 获取所选模型 |
| AreSortedByDistanceAscending | 获取一个值，指示所选模型是按照距离升序排列的 |

### （4）PointSelectionCommand

提供一个在拖动鼠标时显示矩形并引发返回矩形中包含的模型的事件的命令

方法：

|  |  |
| --- | --- |
| Started(ManipulationEventArgs) | 操作开始后发生 |
| Completed(ManipulationEventArgs) | 操作完成后发生 |
| GetCursor() | 获取手势的光标 |

### （5）RectangleSelectionCommand

提供一个在拖动鼠标时显示矩形并引发返回矩形中包含的模型的事件的命令

方法：

|  |  |
| --- | --- |
| Started(ManipulationEventArgs) | 操作开始时触发 |
| Delta(ManipulationEventArgs) | 操作过程中执行 |
| Completed(ManipulationEventArgs) | 操作完成时触发 |
| GetCursor() | 获取光标 |
| HideRectangle() | 隐藏选择的矩形框 |
| UpdateRectangle() | 更新矩形框 |
| ShowRectangle() | 显示矩形框 |

### （6）SelectionCommand（为选择命令提供抽象基类）

属性：

|  |  |
| --- | --- |
| SelectionHitMode | 获取或设置选择命中模式。 |
| MouseDownPoint | 获取鼠标点击点（2D屏幕坐标） |

方法：

|  |  |
| --- | --- |
| Execute(object) | 执行命令 |
| CanExecute(object) | 检查命令是否可以执行 |
| Started(ManipulationEventArgs) | 操作开始时发生。 |
| Delta(ManipulationEventArgs) | 在操作期间更改位置时发生 |
| Completed(ManipulationEventArgs) | 操作完成时发生。 |
| OnModelsSelected(ModelsSelectedEventArgs) | 触发鼠标选择事件 |
| GetCursor(); | 获取手势的光标。 |
| OnMouseDown(object) | 当按下鼠标按钮时调用 |
| OnMouseUp(object, MouseButtonEventArgs) | 当鼠标按钮被释放时调用 |
| OnMouseMove(object, MouseEventArgs) | 当鼠标移动到控件上时调用。 |
| OnCanExecutedChanged(object, EventArgs) | 当执行条件改变时调用 |

### （7）SelectionHitMode（选择模式）

属性：

|  |  |
| --- | --- |
| Touch | 选择触摸选择范围的模型 |
| Inside | 选择完全在范内的模型 |

## 13.ShaderEffects（阴影）

### （1）AnaglyphEffect（提供浮雕混合效果）

属性：

|  |  |
| --- | --- |
| LeftInput | 获取或设置左输入画笔。 |
| Method | 获取或设置浮雕法。 |
| Offset | 获取或设置水平偏移 |
| RightInput | 获取或设置右输入画笔 |
| ShaderMethod | 获取或设置着色器方法。 |

方法：

|  |  |
| --- | --- |
| AnaglyphMethodChanged(DependencyObject, DependencyPropertyChangedEventArgs) | 浮雕方法改变 |

### （2）AnaglyphMethod（浮雕方法）

属性：

|  |  |
| --- | --- |
| True = 0 | 真实浮雕 |
| Gray = 1 | 灰度浮雕 |
| Color = 2 | 全色浮雕 |
| HalfColor = 3 | 半色浮雕 |
| Optimized = 4, | 优化的浮雕 |
| Dubois = 5 | Dubois浮雕 |

### （3）InterlacedEffect（提供交错混合效果）

属性：

|  |  |
| --- | --- |
| LeftInput | 获取或设置左输入画笔。 |
| EvenLeft | 获取或设置一个值，指示偶数行是否应为左视图或右视图。 |
| Offset | 获取或设置水平偏移 |
| RightInput | 获取或设置右输入画笔 |
| ShaderMethod | 获取或设置着色器方法。 |

方法：

|  |  |
| --- | --- |
| EvenLeftChanged(DependencyObject, DependencyPropertyChangedEventArgs) | 浮雕方法改变 |

## 14.Themes（主题）

## 15.Visual3Ds

### （1）Composite（综合）

#### （1.1）BoundingBoxVisual3D（显示指定边框的线框的可视元素）

属性：

|  |  |
| --- | --- |
| BoundingBox | 获取或设置边界框 |
| Diameter | 获取或设置直径。 |
| Fill | 获取或设置边框的画笔。 |

方法：

|  |  |
| --- | --- |
| OnBoxChanged() | 更新边框 |
| OnFillChanged() | 当填充更改时调用。 |
| BoxChanged(DependencyObject, DependencyPropertyChangedEventArgs) | 当边框尺寸更改时调用 |
| FillChanged(DependencyObject, DependencyPropertyChangedEventArgs) | 当填充更改时调用。 |
| AddEdge(Point3D, Point3D) | 增加边缘 |

#### （1.2）ClonedVisual3D（一个视觉元素克隆另一个视觉元素的所有子元素）

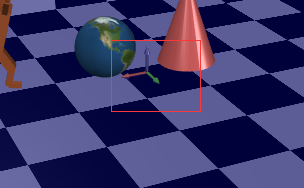
属性：

|  |  |
| --- | --- |
| Source | 获取或设置源 |

方法：

|  |  |
| --- | --- |
| SourceChanged(DependencyObject, DependencyPropertyChangedEventArgs) | 来源发生改变 |
| OnSourceChanged() | 来源发生改变 |

#### （1.3）CoordinateSystemVisual3D（在X，Y和Z方向显示带有箭头的坐标系的视觉元素）



<ht:CoordinateSystemVisual3D/>

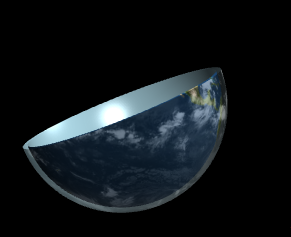
属性：

|  |  |
| --- | --- |
| ArrowLengths | 获取或设置箭头长度 |
| XAxisColor | 获取或设置x轴颜色 |
| YAxisColor | 获取或设置Y轴颜色 |
| ZAxisColor | 获取或设置Z轴颜色 |

方法：

|  |  |
| --- | --- |
| GeometryChanged(DependencyObject, DependencyPropertyChangedEventArgs) | 几何图形改变 |
| OnGeometryChanged() | 几何图形改变 |

#### （1.4）CuttingPlaneGroup（将切平面运用于视觉元素所有子类）



<h:CuttingPlaneGroup x:Name="cuttingGroup1">

<h:CuttingPlaneGroup.CuttingPlanes>

<h:Plane3D Normal="0,0.5,-1"/>

</h:CuttingPlaneGroup.CuttingPlanes>

<h:SphereVisual3D Radius="1" Material="{h:ImageMaterial 'pack://application:,,,/Examples/Earth/earth.jpg'}" ThetaDiv="180" PhiDiv="90" />

<h:SphereVisual3D Radius="1.05" Material="{Binding Clouds}" ThetaDiv="90" PhiDiv="45"/>

</h:CuttingPlaneGroup>

属性：

|  |  |
| --- | --- |
| CuttingPlanes | 获取或设置切平面 |
| IsEnabled | 获取或设置一个值，指示是否启用切割。 |
| Operation | 获取或设置切割操作 |

方法：

|  |  |
| --- | --- |
| OnCompositionTargetRendering(object, RenderingEventArgs) | 当组合目标渲染事件发生时调用 |
| IsEnabledChanged(DependencyObject, DependencyPropertyChangedEventArgs) | 处理更改为=“IsEnabled”属性。 |
| OperationChanged(DependencyObject, DependencyPropertyChangedEventArgs) | 处理更改为Operation属性。 |
| ApplyCuttingGeometries(bool | 运用切平面 |
| ApplyCuttingPlanesToModel(GeometryModel3D, Transform3D, bool) | 将切平面应用于模型 |
| Intersect(MeshGeometry3D, GeneralTransform3D, Plane3D, bool) | 用指定的平面相交指定的源网格几何。 |

#### （1.5）Expander3D（将所有内容相对于指定的来源进行转换的视觉元素。）

属性：

|  |  |
| --- | --- |
| Expansion | 获取或设置扩展因子 |
| ExpandOrigin | 获取或设置扩展的来源 |

方法：

|  |  |
| --- | --- |
| ExpandTo(double value, double animationTime) | 扩展到指定的值 |
| ExpansionChanged(DependencyObject d, DependencyPropertyChangedEventArgs e) | 扩展改变 |
| OnExpansionChanged() | 扩展 |
| Expand(GeometryModel3D model, Transform3D transformation) | 扩展 |

#### （1.6）Exploder3D（一个视觉元素，将所有的内容元素都扩张）

属性：

|  |  |
| --- | --- |
| IsExploding | 获取或设置一个值，指示此实例是否正在扩大 |

方法：

|  |  |
| --- | --- |
| IsExplodingChanged(DependencyObject, DependencyPropertyChangedEventArgs) | 当扩张状态改变时调用。 |
| OnIsExplodingChanged() | 当扩张状态改变时发生 |

#### （1.7）FileModelVisual3D（显示从文件加载的模型的可视元素）



<ht:FileModelVisual3D x:Name="model1" Source="Examples/Contour/vwbus.3ds"/>

属性：

|  |  |
| --- | --- |
| ModelLoaded | 模型加载时发生。 |
| DefaultMaterial | 获取或设置默认材料 |
| Source | 获取或设置源文件名。 |

方法：

|  |  |
| --- | --- |
| SourceChanged(DependencyObject obj, DependencyPropertyChangedEventArgs args) | 源文件改变 |
| OnModelLoaded() | 模型加载时发生。 |
| SourceChanged() | 当源或默认材料更改时调用。 |

#### LightVisual3D（显示指定光源的视觉元素。）

属性：

|  |  |
| --- | --- |
| Light | 获取或设置光。 |

方法：

|  |  |
| --- | --- |
| LightChanged(DependencyObject obj, DependencyPropertyChangedEventArgs args) | 灯光改变 |
| OnLightChanged() | 灯光改变时发生。 |

#### MeshNormalsVisual3D（显示指定网格几何的法线的视觉元素）

属性：

|  |  |
| --- | --- |
| Color | 获取或设置法线的颜色。 |
| Diameter | 获取或设置法线箭头的直径 |
| Mesh | 获取或设置网格 |

方法：

|  |  |
| --- | --- |
| MeshChanged(DependencyObject obj, DependencyPropertyChangedEventArgs args) | 网格改变 |
| OnMeshChanged() | 更新视图 |

#### MeshVisual3D（显示Mesh3D网格的视觉元素）

属性：

|  |  |
| --- | --- |
| EdgeDiameter | 获取或设置图形直径 |
| EdgeMaterial | 获取或设置图形材料 |
| FaceBackMaterial | 获取或设置背面材料 |
| FaceMaterial | 获取或设置正面材料 |
| Mesh | 获取或设置网格 |
| SharedVertices | 获取或设置一个值，指示是否共享顶点（平滑阴影 |
| ShrinkFactor | 获取或设置收缩因子 |
| TriangleIndexToFaceIndex | 获取或设置从三角形索引到面部索引的映射。 |
| VertexMaterial | 获取或设置顶点材质。 |
| VertexRadius | 获取或设置顶点半径。 |
| VertexResolution | 获取或设置顶点分辨率（细分数）。 |

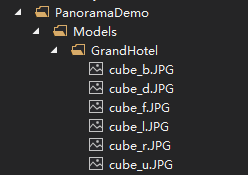
方法：

|  |  |
| --- | --- |
| MeshChanged(DependencyObject obj, DependencyPropertyChangedEventArgs args) | 网格改变 |
| UpdateVisuals() | 更新视图 |

#### PanoramaCube3D（显示全景立方体或天空盒的视觉元素）

#### 

#### <ht:PanoramaCube3D Source="Examples\PanoramaDemo\Models\GrandHotel\"/>



#### 属性：

|  |  |
| --- | --- |
| AutoCenter | 获取或设置一个值，指示[自动中心] |
| ShowSeams | 获取或设置一个值，指示是否显示接缝。 |
| Size | 获取或设置多维立方体的大小。 |
| Source | 获取或设置全景/天空盒目录或文件前缀 |

方法：

|  |  |
| --- | --- |
| SourceChanged(DependencyObject, DependencyPropertyChangedEventArgs) | 源文件改变 |
| GeometryChanged(DependencyObject, DependencyPropertyChangedEventArgs) | 几何图形改变 |
| AddCubeSide(Vector3D, Vector3D, string) | 添加立方体 |
| UpdateModel() | 更新模型 |

#### RenderingModelVisual3D（为ModelVisual3D对象提供一个抽象基类，用于监听CompositionTarget.Rendering事件）

方法：

|  |  |
| --- | --- |
| SubscribeToRenderingEvent() | 订阅CompositionTarget.Rendering事件。 |
| UnsubscribeRenderingEvent() | 取消订阅CompositionTarget.Rendering事件 |
| OnCompositionTargetRendering(object, RenderingEventArgs) | 处理CompositionTarget.Rendering事件。 |

#### SortingVisual3D（指定SortingVisual3D的排序方法）

属性：

|  |  |
| --- | --- |
| CheckForOpaqueVisuals | 获取或设置一个值，指示是否检查是否存在不透明的子视觉效果。 |
| IsSorting | 获取或设置一个值，指示此实例是否正在排序。 当视觉从Viewport3D中删除时，此属性应设置为false以取消订阅呈现事件。 |
| Method | 获取或设置排序方法 |
| SortingFrequency | 获取或设置排序频率（Hz） |

方法：

|  |  |
| --- | --- |
| OnCompositionTargetRendering(object sender, RenderingEventArgs e) | 组合target\_呈现 |
| IsSortingChanged(DependencyObject d, DependencyPropertyChangedEventArgs e) | 排序更改 |
| GetCameraDistance(Visual3D c, Point3D cameraPos, Transform3D transform) | 获取指定视觉距离相机的距离。 |
| IsVisualTransparent(Visual3D visual) | 确定指定的视觉效果是否透明 |
| OnIsSortingChanged() | 排序更改 |
| SortChildren() | 排序内容 |

#### VectorFieldVisual3D（显示矢量场的视觉元素）

属性：

|  |  |
| --- | --- |
| Diameter | 获取或设置直径 |
| Directions | 获取或设置方向。 |
| Fill | 获取或设置填充 |
| HeadLength | 获取或设置头的长度 |
| Positions | 获取或设置位置 |
| ThetaDiv | 获取或设置箭头的分割数 |

方法：

|  |  |
| --- | --- |
| UpdateModel() | 更新模型 |
| CreateBodyTransform(Point3D, Vector3D) | 创造变形 |
| CreateHeadTransform(Point3D, Vector3D) | 创建标题变换 |
| CreateGeometry() | 创建几何 |

#### ViewCubeVisual3D（显示多维数据集的立方体）

属性：

|  |  |
| --- | --- |
| BackText | 获取或设置背面文本 |
| BottomText | 获取或设置底部文本。 |
| Center | 获取或设置中心 |
| FrontText | 获取或设置前面文本 |
| LeftText | 获取或设置左面文本 |
| ModelUpDirection | 获取或设置模型垂直方向 |
| RightText | 获取或设置右面文本 |
| Size | 获取或设置大小 |
| TopText | 获取或设置顶部文本 |
| IsEnabled | 获取或设置视图立方体是否可用 |
| Viewport | 取或设置由视图立方体控制的视口 |

方法：

|  |  |
| --- | --- |
| OnClicked(Vector3D, Vector3D) | 触发点击事件 |
| VisualModelChanged(DependencyObject, DependencyPropertyChangedEventArgs) | VisualModel属性已更改 |
| UpdateVisuals() | 更新视图 |
| AddEdges() | 创建几何 |
| AddEdge(Point3D, double, double, double, Vector3D) | 添加边 |
| AddCorners() | 添加角 |
| EdgesMouseLeaves(object, MouseEventArg) | 鼠标离开 |
| EdggesMouseEnters(object, MouseEventArgs) | 鼠标进入 |
| CornersMouseLeave(object, MouseEventArgs) | 鼠标离开 |
| CornersMouseEnters(object, MouseEventArgs) | 鼠标进入 |
| AddCubeFace(Vector3D, Vector3D, Brush, string) | 添加立方体面 |
| FaceMouseLeftButtonDown(objec, MouseButtonEventArgs) | 左键按下 |

类：ClickedEventArgs

属性：

|  |  |
| --- | --- |
| LookDirection | 设置摄像机方向 |
| UpDirection | 设置相机垂直方向 |

右下角的视图立方体，默认为显示；

隐藏：

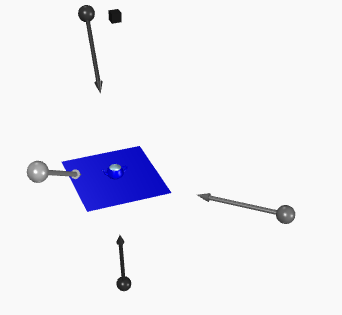
<h:HelixViewport3D ShowViewCube="False">

...

<h:HelixViewport3D>

### LightSetups（灯光设置）

#### DefaultLights（一个视觉元素，包含一个包含4个定向灯的光模型）



<helix:DefaultLights ShowLights="True"/>

方法：

|  |  |
| --- | --- |
| AddLights(Model3DGroup) | 添加灯到元素中 |

#### （1.2）DirectionalHeadLight（代表定向头灯）

#### （1.3）GenericHeadLight（为在相机空间中操作的灯提供基础类）

属性：

|  |  |
| --- | --- |
| Brightness | 获取或设置大灯的亮度。 如果设置，此属性将覆盖 “Color”属性。 |
| Color | 获取或设置大灯的颜色。 如果 “亮度”设置为<c> NaN </ c>，则使用此属性。 |
| Position | 获取或设置大灯的位置（在相机空间中） |

方法：

|  |  |
| --- | --- |
| OnVisualParentChanged(DependencyObject) | 当3-D可视对象的父级更改时调用。 |
| AddLights(Model3DGroup) | 将灯添加到元素。 |
| Update(DependencyObject, DependencyPropertyChangedEventArgs ) | 更新灯光 |
| CameraChanged(object, EventArgs) | 处理相机改变 |
| Update() | 更新此实例。 |

#### （1.4）LightSetup（光模型的抽象基类。）

属性：

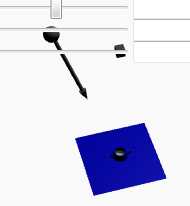
|  |  |
| --- | --- |
| ShowLights | 获取或设置一个值，指示是否显示灯视觉效果 |

方法：

|  |  |
| --- | --- |
| SetupChanged(DependencyObject, DependencyPropertyChangedEventArgs) | 设置更改 |
| AddLights(Model3DGroup); | 将灯添加到元素。 |
| OnSetupChanged() | 处理灯光设置的变化 |
| OnShowLightsChanged() | 显示灯改变时调用 |
| ShowLightsChanged(DependencyObject, DependencyPropertyChangedEventArgs) | 显示灯改变 |

#### （1.5）SpotHeadLight（点光源）

#### （1.6）SunLight（自然光）



<helix:SunLight Altitude="{Binding Altitude}" Azimuth="{Binding Azimuth}" Brightness="{Binding Brightness}" Ambient="{Binding Ambient}" ShowLights="True"/>

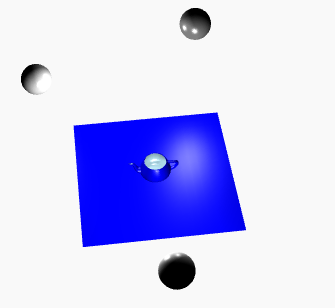
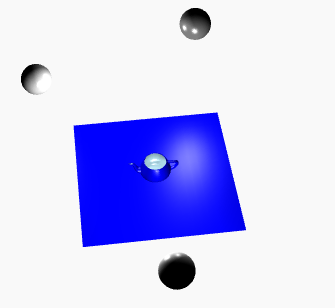
属性：

|  |  |
| --- | --- |
| Altitude | 获取或设置高度角（度） |
| Ambient | 获取或设置环境亮度。 |
| Azimuth | 获取或设置方位角（度） |
| Brightness | 获取或设置亮度 |

方法：

|  |  |
| --- | --- |
| AddLights(Model3DGroup) | 将灯添加到元素。 |

#### （1.7）ThreePointLights（包含三点光源设置的视觉元素）



<helix:ThreePointLights ShowLights="True"/>

属性：

|  |  |
| --- | --- |
| Distance | 获取或设置距离 |
| FillLightAngle | 获取或设置填充光角度。 |
| FillLightSideAngle | 获取或设置填充光侧角 |
| FrontDirection | 获取或设置前方向 |
| KeyLightAngle | 获取或设置关键光角度。 |
| KeyLightBrightness | 获取或设置关键光亮度。 |
| KeyLightSideAngle | 获取或设置关键灯侧角 |
| KeyToFillLightRatio | 获取或设置键以填充光比 |
| KeyToRimLightRatio | 获取或设置边框光比的键。 |
| RimLightAngle | 获取或设置边框光角度 |
| Target | 获取或设置目标 |
| UpDirection | 获取或设置方向。 |

方法：

|  |  |
| --- | --- |
| AddLights(Model3DGroup) | 将灯添加到元素。 |

### Manipulators

### （2.1）BindableRotateManipulator（表示包含可围绕轴旋转的操纵器的视觉元素）

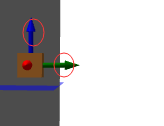
属性：

|  |  |
| --- | --- |
| Axis | 获取或设置旋转轴 |
| Diameter | 获取或设置直径。 |
| InnerDiameter | 获取或设置内径 |
| Length | 获取或设置圆柱体的长度。 |
| Pivot | 获取或设置操纵器的枢轴点 |
| InternalPivotPoint | 获取或设置内部枢轴点。 |

方法：

|  |  |
| --- | --- |
| UpdateGeometry() | 更新几何图形。 |
| OnMouseDown(MouseButtonEventArgs) | 鼠标按下 |
| OnMouseMove(MouseEventArgs) | 鼠标移动 |
| PositionChanged(DependencyPropertyChangedEventArgs) | 位置改变 |
| ValueChanged(DependencyPropertyChangedEventArgs) | 根据旋转值的变化更新目标变换 |

#### （2.2）BindableTranslateManipulator（表示一个可视元素，其中包含一个可以沿着轴平移的操纵器。）



<ht:TranslateManipulator Color="Black" Position="-8,0,0" Offset="0,0,0.5" Length="1" Value="{Binding Height, ElementName=box1, Converter={StaticResource Multiplier}}" Direction="0,0,1"/>

属性：

|  |  |
| --- | --- |
| Diameter | 获取或设置操纵器箭头的直径。 |
| Length | 获取或设置操纵器箭头的长度。 |
| Direction | 获取或设置变换方向 |

方法：

|  |  |
| --- | --- |
| UpdateGeometry() | 更新几何图形。 |
| OnMouseDown(MouseButtonEventArgs) | 鼠标按下 |
| OnMouseMove(MouseEventArgs) | 鼠标移动 |
| PositionChanged(DependencyPropertyChangedEventArgs) | 位置改变 |
| ValueChanged(DependencyPropertyChangedEventArgs | 根据旋转值的变化更新目标变换 |
| GetNearestPoint(Point, Point3D, Vector3D) | 获得平移轴上最近的点 |

#### （2.3）CombinedManipulator（表示显示平移和旋转操纵器的视觉元素）

属性：

|  |  |
| --- | --- |
| CanRotateY | 获取或设置一个值，指示此实例是否可以旋转Y |
| CanRotateX | 获取或设置一个值，指示此实例是否可以旋转X. |
| CanRotateZ | 获取或设置一个值，指示此实例是否可以旋转Z |
| CanTranslateX | 获取或设置一个值，指示此实例是否可以平移X. |
| CanTranslateY | 获取或设置一个值，指示此实例是否可以平移Y. |
| CanTranslateZ | 获取或设置一个值，指示此实例是否可以平移Z. |
| Diameter | 获取或设置直径 |
| Offset | 获取或设置视觉的偏移 |
| Pivot | 获取或设置操纵器的枢轴点 |
| Position | 获取或设置操纵器的位置。 |
| TargetTransform | 获取或设置目标变换 |

方法：

|  |  |
| --- | --- |
| Bind(ModelVisual3D ) | 将此操纵器绑定到给定的Visual3D。 |
| UnBind() | 释放此操纵器的绑定 |
| UpdateChildren() | 更新内容的视觉效果 |
| ChildrenChanged(DependencyObject, DependencyPropertyChangedEventArgs) | 处理与内容视觉相关的属性的更改 |

#### （2.4）Manipulator（为操纵者提供抽象基类）

属性：

|  |  |
| --- | --- |
| Color | 获取或设置操纵器的颜色。 |
| Material | 获取或设置操纵器的材料 |
| BackMaterial | 获取或设置操纵器背面的材料 |
| Offset | 获取或设置视觉的偏移（该矢量被添加到位置点） |
| Position | 获取或设置操纵器的位置。 |
| TargetTransform | 获取或设置目标变换。 |
| Value | 获取或设置操纵器值。 |
| Camera | 获取或设置相机。 |
| HitPlaneNormal | 获取或设置命中平面法线 |
| Model | 获取或设置模型。 |
| ParentViewport | 获取或设置父视口。 |

方法：

|  |  |
| --- | --- |
| Bind(ModelVisual3D) | 将此操纵器绑定到给定的Visual3D。。 |
| UnBind() | 释放该操纵器的绑定。 |
| UpdateGeometry(DependencyObject, DependencyPropertyChangedEventArgs) | 当与几何图形相关的属性更改时调用 |
| GetHitPlanePoint(Point, Point3, Vector3D) | 在命中平面上投射点 |
| UpdateGeometry(); | 更新几何图形 |
| OnMouseDown(MouseButtonEventArgs) | 鼠标按下 |
| OnMouseUp(MouseButtonEventArgs) | 鼠标弹起 |
| PositionChanged(DependencyPropertyChangedEventArgs) | 位置改变 |
| ValueChanged(DependencyPropertyChangedEventArgs) | 值改变 |
| ToLocal(Point3D) | 坐标转换 |
| ToWorld(Point3D) | 坐标转换（球到面） |
| ToWorld(Vector3D) | 坐标转换面到球） |
| ColorChanged() | 颜色改变 |

#### （2.5）RotateManipulator（表示包含可围绕轴旋转的操纵器的视觉元素）

属性：

|  |  |
| --- | --- |
| Axis | 获取或设置旋转轴 |
| Diameter | 获取或设置直径。 |
| InnerDiameter | 获取或设置内径 |
| Length | 获取或设置圆柱体的长度。 |
| Pivot | 获取或设置操纵器的枢轴点 |

方法：

|  |  |
| --- | --- |
| UpdateGeometry() | 更新几何图形。 |
| OnMouseDown(MouseButtonEventArgs) | 鼠标按下 |
| OnMouseMove(MouseEventArgs) | 鼠标移动 |

#### （2.6）TranslateManipulator（表示一个可视元素，其中包含一个可以沿着轴平移的操纵器。）

属性：

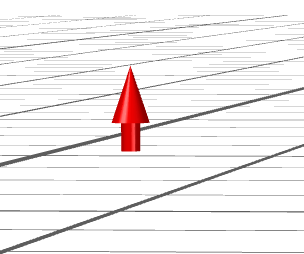
|  |  |
| --- | --- |
| Diameter | 获取或设置操纵器箭头的直径。 |
| Length | 获取或设置操纵器箭头的长度。 |
| Direction | 获取或设置变换方向 |

方法：

|  |  |
| --- | --- |
| UpdateGeometry() | 更新几何图形。 |
| OnMouseDown(MouseButtonEventArgs) | 鼠标按下 |
| OnMouseMove(MouseEventArgs) | 鼠标移动 |
| GetNearestPoint(Point, Point3D, Vector3D) | 获得平移轴上最近的点。 |

### MeshVisuals

#### （3.1）ArrowVisual3D（显示箭头的视觉元素。）



<helix:ArrowVisual3D Point1="5 0 -0.45" Point2="5 0 0.45" Diameter="0.2" Fill="Red" Visible="{Binding IsChecked, ElementName=ArrowVisible}"/>

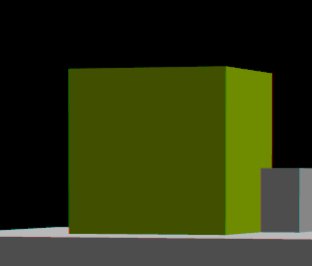
属性：

|  |  |
| --- | --- |
| Diameter | 获取或设置直径。 |
| Direction | 获取或设置方向 |
| HeadLength | 获取或设置头部的长度（相对于箭头圆柱体的直径） |
| Origin | 获取或设置原点。 |
| Point1 | 获取或设置箭头的起始点。 |
| Point2 | 获取或设置箭头的终点。 |
| ThetaDiv | 获取或设置箭头周围的分割数。 |

方法：

|  |  |
| --- | --- |
| Tessellate() | 做细分并返回 “MeshGeometry3D” |

#### （3.2）BoxVisual3D（一个呈现框的视觉元素。）



<h:BoxVisual3D Width="20" Height="20" Length="20" Center="0,0,24" Fill="Yellow"/>

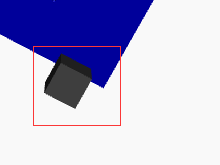
属性：

|  |  |
| --- | --- |
| Center | 获取或设置框的中心。 |
| BottomFace | 获取或设置一个值，指示是否包括底面。 |
| Height | 获取或设置高度（沿局部z轴） |
| Length | 获取或设置框的长度（沿着局部x轴）。 |
| TopFace | 获取或设置一个值，指示是否包括顶面 |
| Width | 获取或设置框的宽度（沿着本地y轴） |

方法：

|  |  |
| --- | --- |
| Tessellate() | 做细分并返回 “MeshGeometry3D” |

#### （3.3）CubeVisual3D（显示立方体的视觉元素）



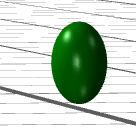
#### 属性：

|  |  |
| --- | --- |
| Center | 获取或设置立方体中心。 |
| SideLength | 获取或设置多边形边的长度 |

方法：

|  |  |
| --- | --- |
| Tessellate() | 做细分并返回 “MeshGeometry3D” |

#### （3.4）EllipsoidVisual3D（显示轴对齐椭圆体的视觉元素）



<helix:EllipsoidVisual3D Center="2 0 0" Fill="Green" RadiusX="0.45" RadiusY="0.3" RadiusZ="0.45" Visible="{Binding IsChecked, ElementName=EllipsoidVisible}"/>

属性：

|  |  |
| --- | --- |
| Center | 获取或设置椭圆的中心（这将设置元素的变换） |
| PhiDiv | 获取或设置phi方向的分割数（从“顶”到“底”）。 |
| RadiusX | 获取或设置椭球的X赤道半径 |
| RadiusY | 获取或设置椭球的Y赤道半径 |
| RadiusZ | 获取或设置椭球的Z赤道半径 |
| ThetaDiv | 获取或设置θ方向（围绕球体）的分割数。 |

方法：

|  |  |
| --- | --- |
| Tessellate() | 做细分并返回 “MeshGeometry3D” |

#### （3.5）ExtrudedVisual3D（沿着路径挤压断面的视觉元素。）

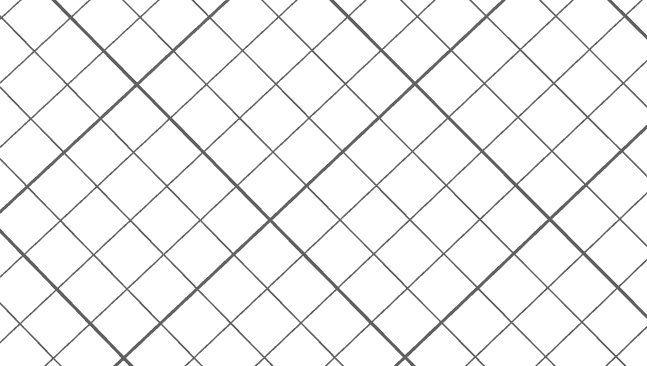
属性：

|  |  |
| --- | --- |
| Diameters | 沿路径获取或设置直径 |
| Angles | 沿路径获取或设置直径。 |
| IsPathClosed | 获取或设置一个值，指示路径是否关闭 |
| IsSectionClosed | 获取或设置一个值，指示该部分是否关闭。 |
| Path | 获取或设置路径 |
| Section | 获取或设置部分。 |
| SectionXAxis |  |
| TextureCoordinates | 沿着路径获取或设置纹理坐标（仅X） |

方法：

|  |  |
| --- | --- |
| Tessellate() | 做细分并返回 “MeshGeometry3D” |

#### （3.6）GridLinesVisual3D（显示一组网格线的视觉元素。）



<helix:GridLinesVisual3D Center="6 0 -0.5" Fill="Gray" Visible="{Binding IsChecked, ElementName=GridLinesVisible}"/>

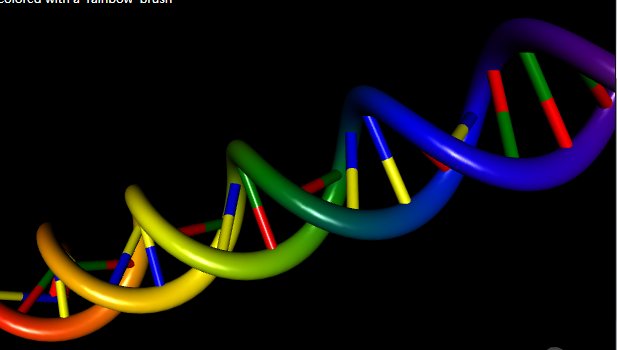
属性：

|  |  |
| --- | --- |
| Center | 获取或设置网格的中心 |
| Length | 获取或设置网格区域的长度。 |
| LengthDirection | 获取或设置网格的长度方向。 |
| MajorDistance | 获取或设置主网格线之间的距离 |
| MinorDistance | 获取或设置次网格线之间的距离。 |
| Normal | 获取或设置网格平面的法向量 |
| Thickness | 获取或设置网格线的宽度。 |
| Width | 获取或设置网格区域的宽度（垂直于长度方向）。 |

方法：

|  |  |
| --- | --- |
| Tessellate() | 做细分并返回 “MeshGeometry3D” |
| IsMultipleOf(double, double) | 确定y是否为d的倍数。 |
| AddLineX(MeshBuilder, double, double, double, double) | 加线X |
| AddLineY(MeshBuilde, double, double, double, double) | 加线Y |
| GetPoint(double, double) | 在平面得到一点。 |

#### （3.7）HelixVisual3D（显示螺旋的视觉元素）



<helix:HelixVisual3D Radius="2" Diameter="0.5" Turns="3" Length="30" Fill="{StaticResource RainbowBrush}"/>

<helix:HelixVisual3D Radius="2" Diameter="0.5" Turns="3" Length="30" Phase="180" Fill="{StaticResource RainbowBrush}"/>

属性：

|  |  |
| --- | --- |
| Origin | 获取或设置原点 |
| Length | 获取或设置长度 |
| Diameter | 获取或设置直径。 |
| Phase | 获取或设置相位 |
| Radius | 获取或设置半径 |
| Turns | 获取或设置圈数。 |

方法：

|  |  |
| --- | --- |
| Evaluate(double, double, out Point) | 计算表面。 |

#### （3.8）MeshElement3D（表示包含一个GeometryModel3D和前后材质的元素的基类。）

属性：

|  |  |
| --- | --- |
| BackMaterial | 获得或设置背部材料 |
| Fill | 获取或设置填充画笔。 此画笔将用于“材质”和“背面材质” |
| Material | 获取或设置材料 |
| Visible | 获取或设置一个值，指示此 “MeshElement3D”是否可见。 |
| Model | 获取几何模型 |

方法：

|  |  |
| --- | --- |
| BeginEdit() | 开始对对象进行编辑 |
| CancelEdit() | 取消最后一次编辑 |
| EndEdit() | 结束编辑 |
| UpdateModel() | 更新模型 |
| VisibleChanged(DependencyObject d, DependencyPropertyChangedEventArgs e) | 可见标志已更改 |
| GeometryChanged(DependencyObject d, DependencyPropertyChangedEventArgs e) | 几何图形改变 |
| MaterialChanged(DependencyObject d, DependencyPropertyChangedEventArgs e) | Material或BackMaterial属性已更改。 |
| OnFillChanged() | 填充属性已更改。 |
| OnGeometryChanged() | 处理几何或可见状态的变化 |
| OnMaterialChanged() | 处理材料/背面材料的变化 |
| Tessellate(); | 做细分并返回 “MeshGeometry3D”。 |
| FillChanged(DependencyObject, DependencyPropertyChangedEventArgs) | 当填充更改时调用 |

#### （3.9）MeshGeometryVisual3D（可以绑定到MeshGeometry3D的视觉元素。）

属性：

|  |  |
| --- | --- |
| MeshGeometry | 获取或设置 “MeshGeometry3D”定义的形状。 |

方法：

|  |  |
| --- | --- |
| Tessellate() | 做细分并返回 “MeshGeometry3D”。 |

#### （3.10）ParametricSurface3D（为矩形网格上评估的参数曲面提供基类。）

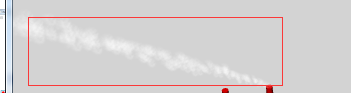
属性：

|  |  |
| --- | --- |
| MeshSizeU | 以u方向获取或设置网格尺寸。 |
| MeshSizeV | 以v方向获取或设置网格尺寸。 |

方法：

|  |  |
| --- | --- |
| Tessellate() | 做细分并返回 “MeshGeometry3D”。 |
| Evaluate(double u, double v, out Point textureCoord); | 在指定的u，v参数下评估曲面 |
| AddTriangle(MeshGeometry3D mesh, int i1, int i2, int i3) | 添加三角形。 |
| IsDefined(Point3D point) | 确定是否定义了指定的点。 |

#### （3.11）ParticleSystem（在WPF的限制内渲染一个简单的粒子系统）



<h:ParticleSystem Texture="{StaticResource SmokeBrush}" Position="{Binding TopPosition, ElementName=chimney1}" AliveParticles="1000" Acceleration="10"/>

属性：

|  |  |
| --- | --- |
| AliveParticles | 获取或设置活动粒子的数量。 |
| Position | 获取或设置位置。 |
| StartDirection | 获取或设置起始方向 |
| Acceleration | 获取或设置加速度。 |
| AccelerationDirection | 获取或设置加速度方向 |
| AccelerationSpreading | 获取或设置加速度展开角度。 |
| StartRadius | 获取或设置起始半径 |
| StartSize | 获取或设置起始大小 |
| StartVelocity | 获取或设置起始速度。 |
| VelocityDamping | 获取或设置速度阻尼因子。 |
| StartVelocityRandomness | 获取或设置起始速度随机性 |
| StartSpreading | 获取或设置开始扩展。 |
| LifeTime | 获取或设定生命周期 |
| AngularVelocity | 获取或设置角速度。 |
| SizeRate | 获取或设置大小率。 |
| FadeOutTime | 获取或设置淡出时间。 |
| EmitRate |  |
| Texture | 获取或设置纹理。 |

方法：

|  |  |
| --- | --- |
| TextureChanged() | 纹理更改时更新材质。 |
| OnVisualParentChanged(DependencyObject oldParent) | 当3-D可视对象的父级更改时调用 |
| EmitOne() | 发出一个粒子 |
| Update(double time) | 更新系统 |
| AdjustListLength<T>(IList<T> list, int targetLength) | 调整指定列表的长度。 |
| CreateRandomVector(Vector3D z, double spreading) | 创建一个随机向量 |
| OnCompositionTargetRendering(object sender, RenderingEventArgs eventArgs) | 处理CompositionTarget.Rendering事件 |

类：Particle

#### （3.12）PieSliceVisual3D（一个视觉元素，显示由中心，法线，向上向量，内半径，外半径，起始和终止角度定义的平坦饼状切片）



<helix:PieSliceVisual3D Center="-1 0 0" OuterRadius="0.45" InnerRadius="0.1" Fill="Turquoise" Visible="{Binding IsChecked, ElementName=PieSliceVisible}"/>

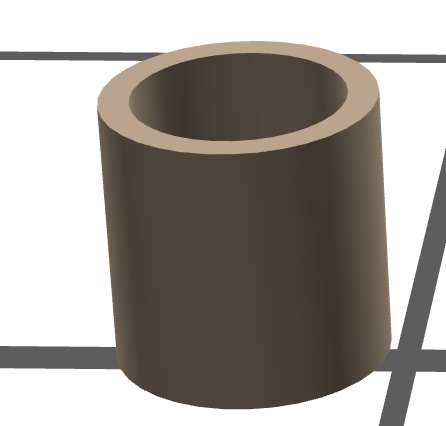
属性：

|  |  |
| --- | --- |
| Center | 获取或设置中心。 |
| EndAngle | 获取或设置结束角度。 |
| InnerRadius | 获取或设置内半径。 |
| Normal | 获取或设置法线 |
| OuterRadius | 获取或设置外半径 |
| StartAngle | 获取或设置开始角度。 |
| ThetaDiv | 获取或设置切片的角分割数 |
| UpVector | 获取或设置向量 |

方法：

|  |  |
| --- | --- |
| Tessellate() | 做细分并返回 “MeshGeometry3D” |

#### （3.13）PipeVisual3D（显示两点之间管道的视觉元素）



<helix:PipeVisual3D Point1="-2 0 -0.45" Point2="-2 0 0.45" Diameter="0.9" InnerDiameter="0.7" Fill="Bisque" Visible="{Binding IsChecked, ElementName=PipeVisible}"/>

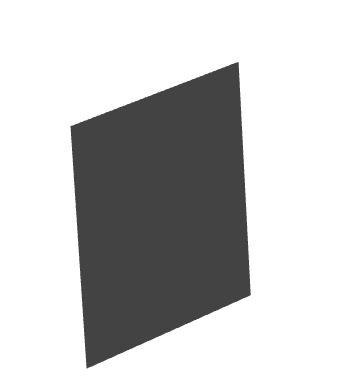
属性：

|  |  |
| --- | --- |
| Diameter | 获取或设置（外）直径。 |
| InnerDiameter | 获取或设置内径 |
| Point1 | 获取或设置开始点 |
| Point2 | 获取或设置结束点。 |
| ThetaDiv | 获取或设置切片的角分割数 |

方法：

|  |  |
| --- | --- |
| Tessellate() | 做细分并返回 “MeshGeometry3D” |

#### （3.14）QuadVisual3D（显示四边形多边形的视觉元素）



<helix:QuadVisual3D Point1="-3 -0.45 -0.45" Point2="-3 0.45 -0.45" Point3="-3 0.45 0.45" Point4="-3 -0.45 0.45" Fill="Gray" Visible="{Binding IsChecked, ElementName=QuadVisible}"/>

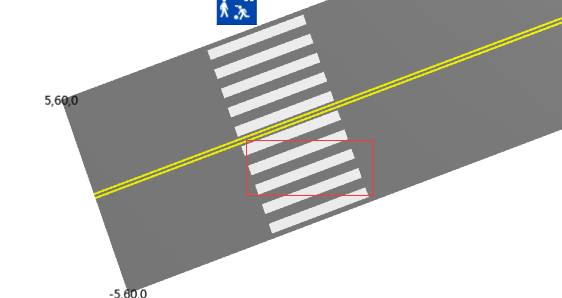
属性：

|  |  |
| --- | --- |
| Point1 | 获取或设置第一个点 |
| Point2 | 获取或设置第二个点 |
| Point3 | 获取或设置第三个点 |
| Point4 | 获取或设置第四个点 |

方法：

|  |  |
| --- | --- |
| Tessellate() | 做细分并返回 “MeshGeometry3D” |

#### （3.15）RectangleVisual3D（一个可视元素，显示由原点，法线，长度和宽度定义的3D矩形。）



<ht:RectangleVisual3D Width="0.1" Length="60" Origin="0.1,30,0.01" LengthDirection="0,1,0" Normal="0,0,1" Material="{ht:Material Yellow}"/>

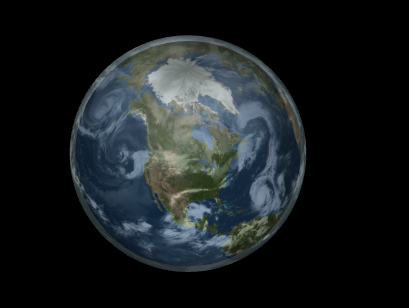
属性：

|  |  |
| --- | --- |
| DivLength | 获取或设置“长度”方向上的分割数。 |
| DivWidth | 获取或设置“宽度”方向上的分割数 |
| Length | 获取或设置长度 |
| LengthDirection | 获取或设置长度方向。 |
| Normal | 获取或设置平面的法向量。 |
| Origin | 获取或设置平面的中心点 |
| Width | 获取或设置宽度 |

方法：

|  |  |
| --- | --- |
| Tessellate() | 做细分并返回 “MeshGeometry3D” |
| CoerceDivValue(DependencyObject, object) | 强化分数值 |

#### （3.16）SphereVisual3D（显示由中心和半径定义的球体的视觉元素）



<t:SphereVisual3D Radius="1" Material="{t:ImageMaterial 'pack://application:,,,/Examples/Earth/earth.jpg'}" ThetaDiv="180" PhiDiv="90" />

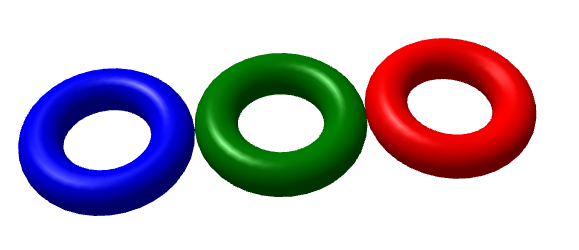
属性：

|  |  |
| --- | --- |
| Center | 获取或设置球体的中心。 |
| PhiDiv | 获取或设置phi方向的分割数（从“顶”到“底”） |
| Radius | 获取或设置球体的半径。 |
| ThetaDiv | 取或设置θ方向（围绕球体）的分割数 |

方法：

|  |  |
| --- | --- |
| Tessellate() | 做细分并返回 “MeshGeometry3D” |

#### （3.17）TorusVisual3D（显示由两个直径（圆环和它的管）定义的环面的视觉元素。）



<ht:TorusVisual3D x:Name="torus1" TorusDiameter="{Binding Value, ElementName=torusDiameterSlider}" TubeDiameter="{Binding Value, ElementName=tubeDiameterSlider}"

ThetaDiv="{Binding Value, ElementName=thetaDivSlider}" PhiDiv="{Binding Value, ElementName=phiDivSlider}" Fill="Red" />

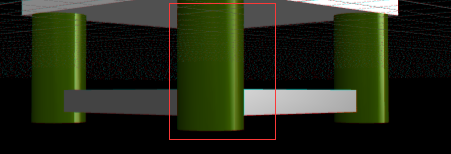
属性：

|  |  |
| --- | --- |
| TorusDiameter | 获取或设置（环面）直径。 |
| PhiDiv | 获取或设置phi方向的分割数（从“顶”到“底”） |
| TubeDiameter | 获取或设定管径。 |
| ThetaDiv | 取或设置θ方向（围绕球体）的分割数 |

方法：

|  |  |
| --- | --- |
| Tessellate() | 做细分并返回 “MeshGeometry3D” |

#### （3.18）TruncatedConeVisual3D（显示由原点，高度，法线，基底和顶部半径定义的截锥的视觉元素）

  
<h:TruncatedConeVisual3D BaseRadius="5" TopRadius="5" BaseCap="True" Origin="-20,-20,-10" Height="20" Fill="Green"/>

属性：

|  |  |
| --- | --- |
| BaseCap | 获取或设置一个值，指示是否包括基本封顶 |
| BaseRadius | 获取或设置基准半径。 |
| Height | 获取或设置高度 |
| Normal | 获取或设置法线 |
| Origin | 获取或设置原点 |
| ThetaDiv | 取或设置θ方向（围绕球体）的分割数 |
| TopCap | 获取或设置一个值，指示是否包括顶帽 |
| TopRadius | 获取或设置顶部半径 |

方法：

|  |  |
| --- | --- |
| Tessellate() | 做细分并返回 “MeshGeometry3D” |

#### （3.19）TubeVisual3D（沿着指定路径显示管的视觉元素）



<helix:TubeVisual3D Path="-7 0 -0.45 -7 0.1 0 -7 0 0.45" Diameter="0.1" Fill="HotPink" Visible="{Binding IsChecked, ElementName=TubeVisible}"/>

属性：

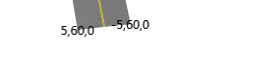
|  |  |
| --- | --- |
| Diameter | 获取或设置管的直径 |
| ThetaDiv | 获取或设置管子周围的分隔数 |
| AddCaps | 获取或设置创建Caps指示符 |

方法：

|  |  |
| --- | --- |
| Tessellate() | 做细分并返回 “MeshGeometry3D” |
| SectionChanged(DependencyObject, DependencyPropertyChangedEventArgs) | 该部分已更改。 |
| OnSectionChanged | 更新该部分。 |

### （4）ScreenSpaceVisuals

#### （4.1）BillboardVisual3D



<ht:BillboardTextVisual3D Position="5,0,0" Text="5,0,0" DepthOffset="0.01"/>



<ht:BillboardVisual3D Position="6,10,3" Width="40" Height="40" Material="{ht:ImageMaterial Examples/Billboard/roadsign.png, IsEmissive=true}"/>

属性：

|  |  |
| --- | --- |
| DepthOffset | 获取或设置深度偏移。 |
| Height | 获取或设置高度。 |
| HorizontalAlignment | 获取或设置水平对齐。 |
| IsRendering | 获取或设置一个值，指示是否呈现此实例。 |
| Material | 获取或设置材料。 |
| Position | 获取或设置Billboard的位置（中心）。 |
| VerticalAlignment | 获取或设置垂直对齐。 |
| Width | 获取或设置宽度。 |
| Mesh | 获取或设置网格。 |
| Model | 获取或设置模型。 |

方法：

|  |  |
| --- | --- |
| OnMaterialChanged() | 材料改变。 |
| GeometryChanged(DependencyObject , DependencyPropertyChangedEventArgs ) | 几何改变。 |
| OnCompositionTargetRendering(object , RenderingEventArgs ) | 组合target\_呈现。 |
| OnVisualParentChanged(DependencyObject) | 当3-D可视对象的父级更改时调用 |
| UpdateGeometry（） | 更新几何图形。 |
| UpdateTransforms() | 更新转换。 |
| MaterialChanged(DependencyObject , DependencyPropertyChangedEventArgs ) | 颜色改变 |
| OnGeometryChanged() | 几何图形更改时调用。 |

#### （4.2）BoundingBoxWireFrameVisual3D

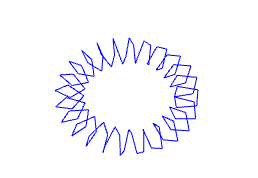
属性：

|  |  |
| --- | --- |
| BoundingBox | 获取或设置边界框 |

方法：

|  |  |
| --- | --- |
| OnBoxChanged() | 更新边界框。 |
| BoxChanged(DependencyObject , DependencyPropertyChangedEventArgs ) | 当边界框更改时调用。 |

#### （4.3）LinesVisual3D



<h:LinesVisual3D Points="{Binding LinePoints}" Color="Blue"/>

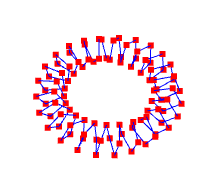
属性：

|  |  |
| --- | --- |
| Thickness | 获取或设置线条的宽度。 |

方法：

|  |  |
| --- | --- |
| UpdateGeometry() | 更新几何。 |
| UpdateTransforms() | 更新转换。 |

#### （4.4）PointsVisual3D



<h:PointsVisual3D Points="{Binding Points}" Color="Red" Size="6" />

属性：

|  |  |
| --- | --- |
| Size | 获取或设置点的大小。 |

方法：

|  |  |
| --- | --- |
| UpdateGeometry() | 更新几何图形。 |
| UpdateTransforms() | 更新转换。 |

#### （4.5）ScreenSpaceVisual3D

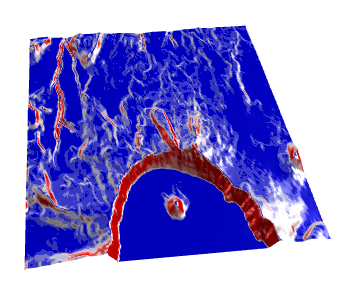
属性：

|  |  |
| --- | --- |
| Color | 获取或设置颜色。 |
| DepthOffset | 获取或设置深度偏移。 |
| IsRendering | 获取或设置一个值，指示是否呈现此实例。 |
| Points | 获取或设置点集合。 |
| Clipping | 获取或设置剪切对象。 |
| Mesh | 获取或设置网格。 |
| Model | 获取或设置模型。 |

方法：

|  |  |
| --- | --- |
| GeometryChanged(object , DependencyPropertyChangedEventArgs ) | 几何属性更改时调用。 |
| PointsChanged(DependencyObject , DependencyPropertyChangedEventArgs ) | 当点变化时调用。 |
| OnCompositionTargetRendering(object , RenderingEventArgs ) | 组合target\_呈现。 |
| OnVisualParentChanged(DependencyObject ) | 当3-D可视对象的父级更改时调用。 |
| UpdateGeometry(); | 更新几何图形。 |
| UpdateTransforms(); | 更新转换。 |
| ColorChanged(object , DependencyPropertyChangedEventArgs ) | 当颜色变化时更改材料。 |
| HandlePointsChanged(object , EventArgs ) | 处 “Points”集合中的更改。 |
| ColorChanged() | 当颜色变化时更改材料 |
| UpdateClipping() | 更新剪辑对象。 |

### Terrain（地形）



<t:TerrainVisual3D Source="Examples/Terrain/Models/crater\_0513.btz"/>

#### （5.1）Textures

##### MapTexture

属性：

|  |  |
| --- | --- |
| Bottom | 获取或设置底部。 |
| Left | 获取或设置左边。 |
| Right | 获取或设置右边。 |
| Top | 获取或设置顶部。 |

方法：

|  |  |
| --- | --- |
| Calculate(TerrainModel , MeshGeometry3D ) | 计算指定模型的纹理。 |

##### SlopeDirectionTexture

属性：

|  |  |
| --- | --- |
| Brush | 获取或设置画笔。 |

方法：

|  |  |
| --- | --- |
| Calculate(TerrainModel , MeshGeometry3D ) | 计算指定模型的纹理。 |

##### SlopeTexture

属性：

|  |  |
| --- | --- |
| Brush | 获取或设置画笔。 |

方法：

|  |  |
| --- | --- |
| Calculate(TerrainModel , MeshGeometry3D ) | 计算指定模型的纹理。 |

##### TerrainTexture

属性：

|  |  |
| --- | --- |
| Material | 获取或设置材料。 |
| TextureCoordinates | 获取或设置纹理坐标。 |

方法：

|  |  |
| --- | --- |
| Calculate(TerrainModel , MeshGeometry3D ) | 计算指定模型的纹理。 |

#### （5.2）TerrainModel

属性：

|  |  |
| --- | --- |
| Bottom | 获取或设置底部。 |
| Data | 获取或设置数据。 |
| Height | 获取或设置高度。 |
| Left | 获取或设置左边。 |
| MaximumZ | 获取或设置最大Z。 |
| MinimumZ | 获取或设置最小Z。 |
| Offset | 获取或设置偏移量。 |
| Right | 获取或设置右边。 |
| Texture | 获取或设置纹理。 |
| Top | 获取或设置顶部。 |
| Width | 获取或设置宽度。 |

方法：

|  |  |
| --- | --- |
| CreateModel(int ) | 创建MAP的3D模型。 |
| Load(string ) | 加载指定的文件。 |
| ReadTerrainFile(Stream ) | 读取.bt（二进制MAP）文件。 |
| ReadTerrainFile(string ) | 读取指定的.bt MAP文件。 |
| ReadZippedFile(string ) | 阅读一个gzip压缩的.bt文件.. |

#### （5.3）TerrainVisual3D

属性：

|  |  |
| --- | --- |
| Source | 获取或设置源map文件。 |

方法：

|  |  |
| --- | --- |
| SourceChanged(DependencyObject, DependencyPropertyChangedEventArgs ) | 来源改变。 |
| UpdateModel() | 更新模型。 |

### （6）Text

#### （6.1）BillboardTextGroupVisual3D（包含文本billboards集合的视觉元素）

属性：

|  |  |
| --- | --- |
| Background | 获取或设置背景 |
| BorderBrush | 获取或设置边框画笔。 |
| IsEnabled | 获取或设置一个值，指示是否启用此对象的更新 |
| PinBrush | 获取或设置针刷。 |
| PinWidth | 获取或设置“pin”的宽度。 |
| BorderThickness | 获取或设置边框宽度 |
| FontFamily | 获取或设置字体系列。 |
| FontSize | 获取或设置字体的大小。 |
| FontWeight | 获取或设置字体重量。 |
| Foreground | 前景色 |
| HeightFactor | 获取或设置高度因子。 |
| IsRendering | 获取或设置一个值，指示是否呈现此实例 |
| Items | 获取或设置项目 |
| Padding | 获取或设置填充。 |
| Offset | 设置偏移量 |

方法：

|  |  |
| --- | --- |
| OnCompositionTargetRendering(object, RenderingEventArgs) | 处理CompositionTarget.Rendering事件。 |
| OnVisualParentChanged(DependencyObject) | 当3-D可视对象的父级更改时调用。 |
| UpdateGeometry() | 更新几何图形 |
| UpdateTransforms() | 更新转换 |
| VisualChanged(DependencyObject, DependencyPropertyChangedEventArgs) | 视觉外观发生变化 |
| CreateElement(string) | 创建指定文本的元素 |
| VisualChanged() | 更新视觉外观（纹理和几何） |

#### （6.2）BillboardTextItem（代表Billboard文字项目）

属性：

|  |  |
| --- | --- |
| DepthOffset | 获取或设置深度偏移。 |
| WorldDepthOffset | 获取或设置球体坐标中的深度偏移量 |

#### （6.3）BillboardTextVisual3D（定义材料的类型）

属性：

|  |  |
| --- | --- |
| Background | 获取或设置背景 |
| BorderBrush | 获取或设置边框画笔。 |
| BorderThickness | 获取或设置边框宽度 |
| FontFamily | 获取或设置字体系列。 |
| FontSize | 获取或设置字体的大小。 |
| FontWeight | 获取或设置字体重量。 |
| Foreground | 前景色 |
| HeightFactor | 获取或设置高度因子。 |
| MaterialType | 获取或设置材料类型 |
| Padding | 获取或设置填充。 |
| Text | 获取或设置文本 |

方法：

|  |  |
| --- | --- |
| VisualChanged(DependencyObject, DependencyPropertyChangedEventArgs) | 视觉外观发生变化 |
| VisualChanged() | 当视觉外观改变时更新文本块 |

#### （6.4）SpatialTextItem（表示空间文本项）

属性：

|  |  |
| --- | --- |
| TextDirection | 获取或设置文本方向 |
| UpDirection | 获取或设置垂直方向。 |

#### （6.5）TextGroupVisual3D（提供一个显示文本集合的视觉元素）

属性：

|  |  |
| --- | --- |
| Background | 获取或设置背景画笔。 |
| BorderBrush | 获取或设置边框画笔 |
| BorderThickness | 获取或设置边框宽度 |
| FontFamily | 获取或设置字体系列。 |
| FontSize | 获取或设置字体的大小（如果未设置，则使用Height属性。 |
| FontWeight | 获取或设置字体重量 |
| Foreground | 前景 |
| Height | 获取或设置文本的高度。 |
| IsDoubleSided | 获取或设置一个值，指示此文本视觉是否是双面的 |
| IsFlipped | 获取或设置一个值，指示文本是否应翻转（水平镜像） |
| Items | 获取或设置项目 |
| Padding | 获取或设置填充 |

方法：

|  |  |
| --- | --- |
| VisualChanged() | 视觉改变 |
| VisualChanged(DependencyObject, DependencyPropertyChangedEventArgs) | 视觉改变 |
| CreateElement(string) | 为指定的文本创建一个元素TextBlock或 “FrameworkElement”包装 “TextBlock”。 |
| OptimizeSize(UIElement, double, double) | 优化面板的大小。 |
| CreateTextMaterial(IEnumerable<TextItem>,Func<string, FrameworkElement>, Brus, out Dictionary<string, FrameworkElement>, out Dictionary<FrameworkElement, Rect>) | 创建文本材料 |

#### （6.6）TextItem

属性：

|  |  |
| --- | --- |
| HorizontalAlignment | 获取或设置水平对齐 |
| Position | 获取或设置位置。 |
| Text | 获取或设置文本 |
| VerticalAlignment | 获取或设置垂直对齐 |

#### （6.7）TextVisual3D



<h:TextVisual3D Foreground="Black" Background="LightBlue" BorderBrush="DarkBlue" BorderThickness="1" Height="2" Padding="2" FontFamily="Arial" FontWeight="Normal" FontSize="48" IsDoubleSided="True" Position="2,1,0" UpDirection="0,0,1" TextDirection="0,1,0" HorizontalAlignment="Center" VerticalAlignment="Center" Text="TextVisual3D"> </h:TextVisual3D>

属性：

|  |  |
| --- | --- |
| Background | 获取或设置背景画笔。 |
| BorderBrush | 获取或设置边框画笔 |
| BorderThickness | 获取或设置边框宽度 |
| FontFamily | 获取或设置字体系列。 |
| FontSize | 获取或设置字体的大小（如果未设置，则使用Height属性。 |
| FontWeight | 获取或设置字体重量 |
| Foreground | 前景 |
| Height | 获取或设置文本的高度。 |
| HorizontalAlignment | 获取或设置水平对齐 |
| IsDoubleSided | 获取或设置一个值，指示此文本视觉是否是双面的 |
| Padding | 获取或设置填充 |
| Position | 获取或设置文本的位置 |
| Text | 获取或设置文本 |
| TextDirection | 获取或设置文本方向 |
| UpDirection | 获取或设置文本的垂直方向。 |
| VerticalAlignment | 获取或设置垂直对齐。 |

方法：

|  |  |
| --- | --- |
| VisualChanged() | 视觉改变 |
| VisualChanged(DependencyObject, DependencyPropertyChangedEventArgs) | 视觉改变 |