Working with Input from Touch Screens

Article09/23/2009

Windows Media Center in Windows 7 supports touch input, which allows for a more natural interaction with the computer represented by identifying and tracking one or more contact points on the input device. Touch input involves the user touching the screen and dragging a finger in any direction, which is referred to as a *gesture*. The dragging movement in one direction is referred to as *panning*. To enable your application to support touch input, use the new GesturePanHandler input handler, which provides gesture-tracking behavior to a UI object in a Windows Media Center application, including event firing so that your application can respond appropriately.

Note Gesture handling is disabled by default, so to enable it, set the **TouchInteractive** property to true. To enable panning in the scrolling handler, set the **GesturePanEnabled** property in the ScrollingHandler to true.

MCML supports additional behavior to allow both the user and your application to respond to touch input in different ways.

Using Inertia and Friction

Inertia is a feature that allows variation in movement, which begins when the user releases their touch from the screen, the gesture propels the UI or object to move at a certain velocity, and then slows down according to the amount of friction that was specified.

- If your application supports friction, use the GestureInertiaSettings element to specify the settings for friction and velocity.
- Use the **IsPanning** and **IsPanningInertia** properties of the **GesturePanHandler** element to determine whether and how the user is panning.

Constraining Movement to an Axis

Movement can be constrained (locked) to move along the X or Y axis. This feature is useful for UI that should move in a particular direction. For example on the Windows Media Center Music Library, panning is always left or right along the Y-axis.

Movement can also be constrained to a dominant axis as determined by the user's initial gesture, allowing your application to interpret the user's intent without enforcing precise movements. For example, on the Windows Media Center Start menu, users can pan up or

DOCKET ALARM Find authenticated court documents without watermarks at docketalarm.com.

down to choose a menu strip. Or, they can pan left or right to choose an application tile on a menu strip. If the user gestures right with a slight upward incline, the dominant axis is the X-axis.

You can specify these settings in the **GesturePanHandler** element.

Snapping to Ratchet Points

Ratchet points are like evenly-spaced markers that indicate to your application each time the user pans a certain distance. When a ratchet point is crossed, an event is fired to notify the application. You can use the ratchet events to trigger an action to occur, such as scrolling more data on the screen. Ratcheting is also useful for determining what to do when the gesture ends, such as snapping the UI backward or forward to a known location. The Windows Media Center Start menu strips behave this way—the application tiles are spaced apart, and as the user pans left or right, the focus snaps to a tile rather than to a place between them when panning stops.

- You can specify the size of ratchet points, thereby determining the space between them.
- Ratchet offset values are used to determine where panning has stopped with respect to the last ratchet point that was crossed, and your application can respond accordingly. When movement crosses a ratchet point, the ratchet offset is reset to 0.
- The AutoRatchet feature automatically snaps the UI to the last or next ratchet point when the gesture ends (the user has finished panning and inertia has ended).

  For example, **RatchetSize.Width** is 100 pixels and **AutoRatchetSize.Width** is 75 pixels; if the current offset is 80 when the gesture ends, the Ratcheted event fires, as if the user panned the remaining 20 pixels to reach that ratchet point. However, if the current offset is 60 pixels when the gesture ends, the Ratcheted event does not fire.

Use the GestureRatchetSettings to specify ratchet settings.

Disabling Animation

When your application uses animation, note that the animation can interfere with panning movement. The gesture displaces the target visual, and the view item tries to apply an animation to arrive at the destination, so the gestures look delayed or staggered. It is recommended that you disable animations while the user is actively panning (check

DOCKET ALARM Find authenticated court documents without watermarks at docketalarm.com.

whether **IsPanning**="true" for the **GesturePanHandler**, and then set **AnimationsEnabled**="false" for the view item).

Sample Explorer

- **Input Handlers > Gesture Pan**
- **Input Handlers > Gesture Pan Enabled Scrolling**

See Also

- Sample Explorer
- Working with User Input and Text Entry

DOCKET ALARM Find authenticated court documents without watermarks at docketalarm.com.