Windows Touch 程式開發入門

許煜坤
台灣微軟研究開發處
2010/1/20

Agendas

- “Good, Better, Best” model
- Platforms details
  - Native Win32 APIs – MS Windows SDK 7.0
  - Windows 7 Multi-Touch .Net Interop Sample Library – VS2008 (Windows Form, WPF)
  - Silverlight 3.0
  - WPF 4 – Integrated Foundation for Multi-Touch
- Related contents and resources

Touch Platform Overview

	Good	Better	Best
APIs	**For Free!** • **Panning/zoom gestures** • **Right click gesture**	• **Gesture notifications** • **Pan/zoom/rotate/etc**	• **Raw touch data** • **Manipulation and Inertia processors**
Native Win32	• Controls with standard scrollbars	• WM_GESTURE message	• WM_TOUCH • COM based Manipulation and Inertia Processors
WPF	• WPF 4.0 pan support in ScrollViewer etc.	• Gesture events • Inertia configuration	• Touch events • Manipulation and Inertia Processors
WinForms	• Controls with standard scrollbars	• WM_GESTURE message • P/Invoke	• Manipulation and Inertia Processors in Microsoft.Ink.DLL • Real-time Stylus or Ink Collector

GetSystemMetrics()

- SM_DIGITIZER returns data about available digitizers on the system
- SM_TABLETPC returns data about Tablet functionality
- SM_MAXIMUMTOUCHES yields the largest number of contacts any of the available digitizers supports

Windows Touch Gesture
WM_GESTURE

- Provides notifications when the user performs gestures over your window
- Contains additional information like center of gesture and gesture-specific arguments
- WM_GESTURENOTIFY
  - SetGestureConfig & GESTURECONFIG Structure
- WM_GESTURE
  - GetGestureInfo & GESTUREINFO Structure
    - Gesture command – GID_ZOOM, GID_PAN, GID_ROTATE, GID_TWOFINGERTAP, GID_PRESSANDTAP
  - CloseGestureInfoHandle

Richer Touch Experiences

demo

WM_TOUCH Applications

Windows Touch Input
WM_TOUCH

- Provides raw touch input data, conceptually similar to mouse messages
- Finger painting, custom gestures, feeding higher-level controls (e.g. Manipulations)
- Touch APIs
  - RegisterTouchWindow & UnregisterTouchWindow
  - GetTouchInputInfo & CloseTouchInputHandle
- Decode WM_TOUCH message
  - GetTouchInputInfo & TOUCHINPUT Structure
    - TOUCHEVENTF_DOWN, TOUCHEVENTF_MOVE, TOUCHEVENTF_UP
  - CloseTouchInputHandle

Touch-Optimized Experiences

demo

WM_TOUCH, Manipulations and More

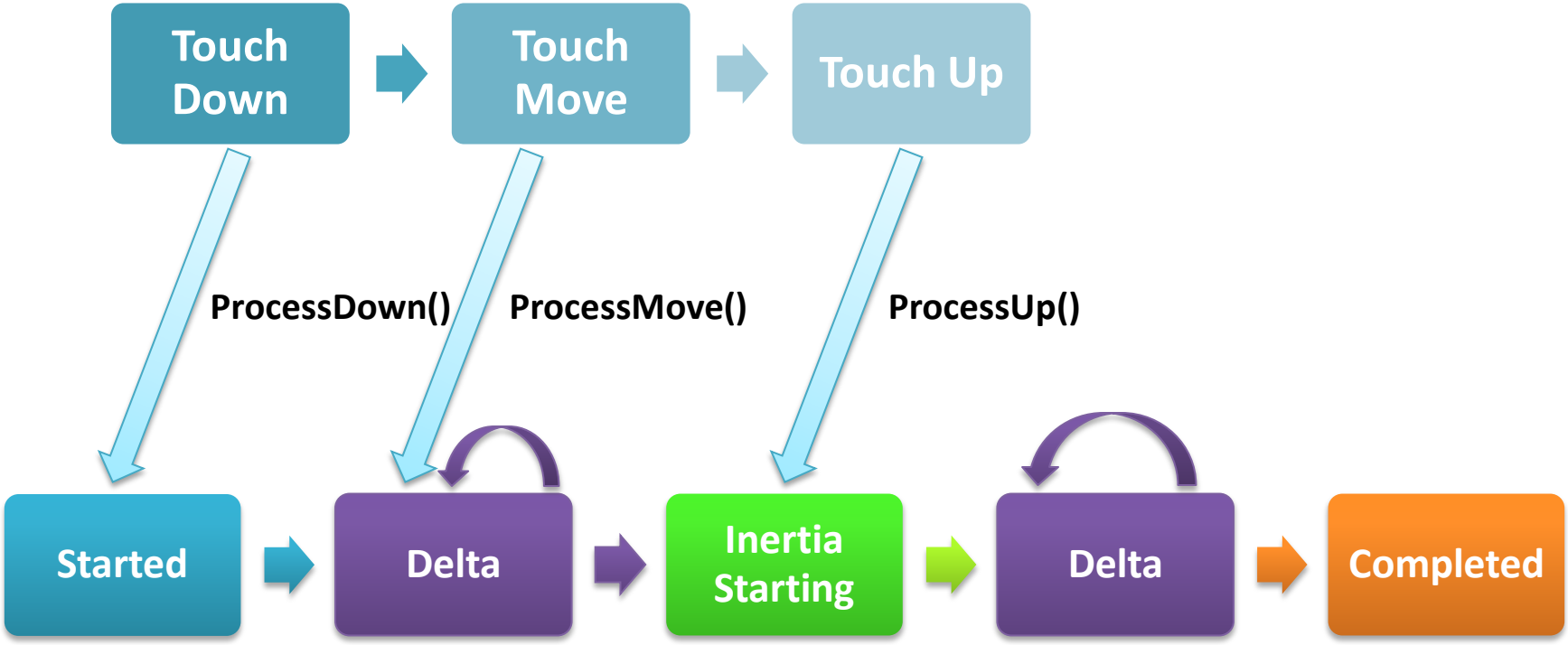
Manipulations in Unmanaged Code

- Implement event sink for the _IManipulationEvents interface
- Pass data received from WM_TOUCH to the IManipulationProcessor
  - ProcessDown
  - ProcessMove
  - ProcessUp
- IManipulationEvents methods
  - ManipulationStatred
  - ManipulationDelta
  - ManipulationCompleted
- IManipulationProcessor properties – Single Finger Rotation
  - PivotPointX, PivotPointY, PivotRadius

Inertia in Unmanaged Code

- Add IInertiaProcessor interface
- Pass data received from WM_TOUCH to the IManipulationProcessor
  - ProcessDown
  - ProcessMove
  - ProcessUp
- IManipulationEvents methods
  - ManipulationStatred
  - ManipulationDelta
  - ManipulationCompleted
- IInertiaProcessor methods
  - Reset
  - Process
  - Complete
- IInertiaProcessor properties
  - Smooth object animation using the Velocity, Deceleration and Displacement properties
  - Controlling object position using elastic bounds

Manipulations/Inertia Flow



WPF 3.5 SP1 Experiences

demo

Real Time Stylus, Manipulations and Inertia

Windows 7 Multi-Touch .Net Interop Sample Library

- Wrappers for Windows Form and WPF 3.5 SP1 apps
- Wrappers for
  - Touch Message
  - Gesture Message
  - ManipulationProcessor
  - ManipulationInertiaProcessor
- Can be downloaded from http://code.msdn.microsoft.com/WindowsTouch (PDC 2008)

Windows 7 Multi-Touch .Net Interop Sample Library - Gesture

- Create GestureHandler
- Implement events handler
  - Pan
  - PanBegin
  - PanEnd
  - Rotate
  - RotateBegin
  - RotateEnd
  - PressAndTapap
  - Zoom
  - ZoomBegin
  - ZoomEnd

Windows 7 Multi-Touch .Net Interop Sample Library - Touch

- Create TouchHandler

- Implement events handler
  - TouchDown
  - TouchMove
  - TouchUp

Windows 7 Multi-Touch .Net Interop Sample Library - Manipulation

- Create TouchHandler or EnableStylusEvents
  - TouchDown, TouchMove, TouchUp
  - StylusDown, StylusMove, StylusUp

- Create Manipulationprocessor
  - Implement events handler
    - ManipulationDelta
    - ManipulationCompleted
    - ManipulationStarted

Windows 7 Multi-Touch .Net Interop Sample Library - Inertia

- Create TouchHandler or EnableStylusEvents
  - TouchDown, TouchMove, TouchUp
  - StylusDown, StylusMove, StylusUp
- Create ManipulationInertiaProcessor
  - Implement events handler
    - ManipulationCompleted
    - ManipulationStatred
    - ManipulationDelta
    - BeforeInertia

Silverlight 3.0 Touch Experiences

demo

Silverlight 3.0

- Add Touch.FrameReported event handler
- GetTouchPoints or GetPrimaryTouchPoint to get TouchPoint
- TouchPoint Class
  - Position
  - TouchAction – Up, Down and Move

Silverlight 3 Touch APIs - Listening

```
public partial class MainPage : UserControl
{
  public MainPage()
  {
    ...
    // listen to touch events from the system
    Touch.FrameReported +=
        new TouchFrameEventHandler(OnFrame);
  }

  void OnFrame(object sender, TouchFrameEventArgs e)
  {
    // enumerate and respond to touch events
  }
}
```

Silverlight 3 Touch APIs - Processing

```
void Touch_FrameReported(object sender, TouchFrameEventArgs e)
{
  TouchPointCollection touchPoints = e.GetTouchPoints(LayoutRoot);

  foreach (TouchPoint tp in touchPoints)
  {
    if (tp.Action == TouchAction.Down) {
      // a new touch has come down
    }
    if (tp.Action == TouchAction.Move) {
      // a previously down touch has moved
    }
    if (tp.Action == TouchAction.Up) {
      // a touch has been removed
    }
  }
}
```

WPF 4.0 Experiences

demo

Manipulations and Inertia

WPF 4.0

- Touch events
  - TouchDown
  - TouchMove
  - TouchUp
  - TouchEnter
  - TouchLeave
- Manipulation events
  - ManipulationStarting
  - ManipulationStarted
  - ManipulationDelta
  - ManipulationCompleted
  - ManipulationInertiaStarting
  - ManipulationBoundaryFeedback

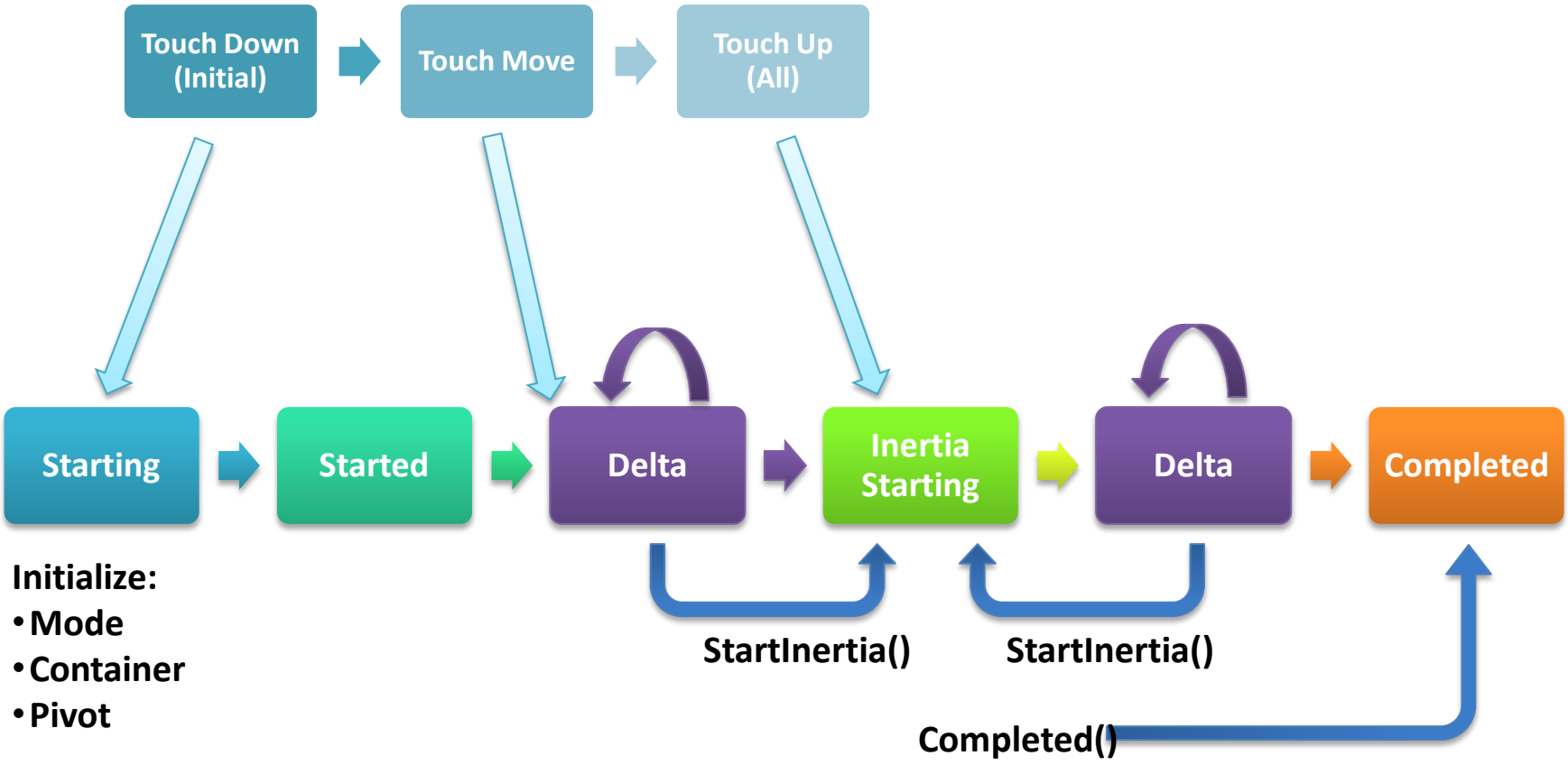
WPF 4 Touch Events

```
public partial class MainWindow : Window
  {
     public MainWindow()
     {
        InitializeComponent();
        // listen to touch events from the system
        this.TouchDown += new EventHandler<TouchEventArgs>(MainWindow_TouchDown);
        this.TouchMove += new EventHandler<TouchEventArgs>(MainWindow_TouchMove);
        this.TouchUp += new EventHandler<TouchEventArgs>(MainWindow_TouchUp);
     }
     void MainWindow_TouchUp(object sender, TouchEventArgs e)
     {
         // a new touch has come down
     }
     void MainWindow_TouchMove(object sender, TouchEventArgs e)
     {
        // a previously down touch has moved
     }
     void MainWindow_TouchDown(object sender, TouchEventArgs e)
     {
        // a touch has been removed
     }
  }
```

WPF 4 Manipulations Events

```
public partial class MainWindow : Window
  {
    public MainWindow()
    {
      InitializeComponent();
      this.ManipulationStarted += new
     EventHandler<ManipulationStartedEventArgs>(MainWindow_ManipulationStarted);
      this.ManipulationInertiaStarting += new
     EventHandler<ManipulationInertiaStartingEventArgs>(MainWindow_ManipulationInertiaStarting);
      this.ManipulationCompleted += new
     EventHandler<ManipulationCompletedEventArgs>(MainWindow_ManipulationCompleted);
      this.ManipulationDelta += new EventHandler<ManipulationDeltaEventArgs>(MainWindow_ManipulationDelta);
    }
    void MainWindow_ManipulationDelta(object sender, ManipulationDeltaEventArgs e)
    {
       // process delta data
    }
    void MainWindow_ManipulationCompleted(object sender, ManipulationCompletedEventArgs e)
    {
      // Manipulations/Inertia complete
    }
    void MainWindow_ManipulationInertiaStarting(object sender, ManipulationInertiaStartingEventArgs e)
    {
      // start inertia
    }
    void MainWindow_ManipulationStarted(object sender, ManipulationStartedEventArgs e)
    {
      // Manipulations started
    }
  }
```

Manipulation Events



Related Content and Resources

- Windows 7 Multi-Touch .Net Interop Sample Library fro WinForm and WPF 3.5 SP1

  http://code.msdn.microsoft.com/WindowsTouch
- WPF 4 Resources: http://connect.microsoft.com/wpf
- Sessions in PDC 2009
  - Windows Touch Deep Dive (Session CL17)

    http://blogs.msdn.com/ansont/archive/2009/12/03/multi-touch-in-wpf-4-part-1.aspx
  - Multitouch on Microsoft Surface and Windows 7 for .NET Developers (Session CL27)

Contact us:

- Forums: http://social.msdn.microsoft.com/Forums/en-US/tabletandtouch/threads/
- Email: twtouch@microsoft.com

Questions ?

Messages & GetMessageExtraInfo()

- GetMessageExtraInfo() returns the extra info associated with a message

- Mouse **up** and **down** messages are tagged with a special signature indicating they came from touch or pen:
  - Mask extra info against 0xFFFFFF80
  - 0xFF515780 for touch, 0xFF515700 for pen

Enable Multi-Touch in WPF 3.5 SP1 by Real Time Stylus

- P/Invoke SetProp Win32 API

```
[DllImport("user32")]
public static extern bool SetProp(IntPtr hWnd, string
  lpString,  IntPtr hData);
```

- Set the window property to enable Multi-Touch input on inking context.

```
WindowInteropHelper windowInteropHelper = new
  WindowInteropHelper((System.Windows.Window) this);
SetProp(windowInteropHelper.Handle,
  "MicrosoftTabletPenServiceProperty", new
  IntPtr(0x01000000));
```

- StylusDown/StylusUp/StylusMove events to handle the touch related events accordingly

Receive WM_TOUCH/WM_GESTURE in Windows Form 3.5 SP1

- P/Invoke Win32 Multi-Touch APIs
- RegisterTouchWindow for WM_TOUCH
- Override WndProc

  protected override void WndProc(ref Message m)
- Handle WM_TOUCH (or WM_GESTURE which is exclusive with WM_TOUCH)