PrimeSense

NITE C# Wrappers - Programmer's Guide

March, 2011

About This Document

This document describes the .NET C# wrappers for PrimeSense's NITE Middleware and for OpenNI.

Table of Contents

1 General

XnVNITE.net - The C# wrapper for NITE covers most of the APIs available in the C++ API of NITE. All the .Net objects mirror C++ objects (and indeed wrap specific objects). The C++ object has the same name, with the XnV prefix (whereas the C# API is in the NITE namespace). Please refer to NITE user guides and NITE API reference guides for detailed description.

1.1 Objects

1.1.1 MessageGenerator

This is a general generator, mirroring XnVMessageGenerator. It exports APIs to Add and Remove listeners, as well as to generate a message (InPtr). See XnVMessageGenerator's documentation in the API reference for more information.

1.1.2 MessageListener

This is a general listener, mirroring XnVMessageListener.
It exports APIs to parse a message (IntPtr), and has an event when a message is parsed, as well as when the listener is activated or deactivated.
See XnVMessageListener's documentation in the API reference for more information.

1.1.3 SessionGenerator

This is a general session generator, mirroring XnVSessionGenerator.
It exports APIs for the following events: SessionStart, SessionEnd and FocusProgress.
It also exports APIs to start tracking at a specific point, and to lose a specific point or all points. A SessionGenerator is a MessageGenerator.
See XnVSessionGenerator's documentation in the API reference for more information.

1.1.4 SessionManager

This is an object mirroring XnVSessionManager. It exports initialization and management of the gestures that start sessions or revive sessions as quick refocus gestures. A SessionManager is a SessionGenerator.
See XnVSessionManager's documentation in the API reference for more information.

1.1.5 Broadcaster

This is an object mirroring XnVBroadcaster. It passes messages it receives to all the listeners that registered to it. A Broadcaster is a MessageListener as well as a MessageGenerator (through IMessageGenerator).
See XnVBroadcaster's documentation in the API reference for more information.

1.1.6 FlowRouter

This is an object mirroring XnVFlowRouter. It passes messages it receives to a single 'active' listener. It also generates activation/deactivation messages when the 'active' listener is changed. A FlowRouter is a MessageListener.
See XnVFlowRouter's documentation in the API reference for more information

1.1.7 PointControl

This is a general listener for point messages. It mirrors XnVPointControl.
PointControl gets the hand points from the messages, and allows inheriting objects to parse the points. A PointControl is a MessageListener.
See XnVPointControl's documentation in the API reference for more information.

1.1.8 PointFilter

This is a general filter for point messages. It mirrors XnVPointFilter.
This is a base class for all filters of point messages. A PointFilter is a PointControl as well a MessageGenerator (through IMessageGenerator).
See XnVPointFilter's documentation in the API reference for more information.

1.1.9 PointDenoiser

This is a point filter that smoothes existing points. It mirrors XnVPointDenoiser.
It receives a point message, and generates a different point message, with a smoothed point. A PointDenoiser is a PointFilter.
See XnVPointDenoiser's documentation in the API reference for more information.

1.1.10 PointArea

This is a point filter that hides points that reside outside of a specific predefined area. It mirrors XnVPointArea. It receives a point message, and passes through only the points within the specific area declared. As far as the point filters/controls downstream, these points don't exist (meaning they get PointDestroy if they ever saw them, etc.). A PointArea is a PointFilter.
See XnVPointArea's documentation in the API reference for more information.

1.1.11 CircleDetector

This is a point control that understands points of the primary hand as circular movements. It mirrors XnVCircleDetector. It exports events of CircleDetected (available every frame when a circle is alive), and NoCircle (when a circle is lost). A CircleDetector is a PointControl.
See XnVCircleDetector's documentation in the API reference for more information.

1.1.12 PushDetector

This is a point control that understands points of the primary hand as push movements (movement along the Z-axis). It mirrors XnVPushDetector.

It exports events for PushDetected (when a movement forward along the Z-axis is detected) and Stabilized (when no movement is detected after a push).A PushDetector is a PointControl.

See XnVPushDetecor’s documentation in the API reference for more information.

1.1.13 SteadyDetector

This is a point control that looks for when a hand point is steady (variance under a certain threshold). It exports events for Steady (when the variance of a hand that isn’t steady is under a threshold for the first time) and NotSteady (when the variance of a hand that is steady is over a threshold for the first time). A SteadyDetector is a PointControl.

See XnVSteadyDetector’s documentation in the API reference for more information.

1.1.14 SwipeDetector

This is a point control that looks for swipe movements of the primary hand along either the Y-axis or the X-axis. It exports events for Swipe in either of the 4 directions, and an event for a general swipe (with the direction as a parameter). A SwipeDetector is a PointControl.

See XnVSwipeDetector’s documentation in the API reference for more information.

1.1.15 WaveDetector

This is a point control that looks for a wave movement of the primary hand. The definition of wave is supplied by the parameters. It exports an event for when a wave is detected. A WaveDetector is a PointControl.

See XnVWaveDetector’s documentation in the API reference for more information.

1.1.16 SelectableSlider1D

This is a point control that uses the primary hand to represents a slider along either the X, Y or Z axes. The slider’s size is determined by configuration, and then the position of the hand is available as a number between 0 and 1 (the ValueChange event). The slider is also split into a certain number of cells, each called an Item. When the hand changes position to be in a different Item, the ItemHover event is raised. When a selection is performed (movement along any of the other axes) the ItemSelected event is raised. A SelectableSlider1D is a PointControl.

See XnVSelectableSlider1D’s documentation in the API reference for more information.

1.1.17 SelectableSlider2D

This is a point control that uses the primary hand to represent a 2D slider on the X/Y plane. The slider's sizes in both axes are determined by configuration, and then the position of the primary hand is available as a number between 0 and 1 for each axis (the ValueChange event). The slider is also split into a number of cells for each axis (n*m), each called an Item. When the hand changes position to be in a different Item, the ItemHover event is raised. When a selection is performed (movement in the Z-axis) the ItemSelected event is raised. A SelectableSlider2D is a PointControl.
See XnVSelectableSlider2D's documentation in the API reference for more information.