Mouse Cursor Sample

*This sample is compatible with the April 2018 XDK*

# Description

This sample demonstrates how to implement mouse controls in Xbox apps and games. Three different mouse controls are demonstrated: (1) absolute mouse control reading the windows mouse, (2) relative mouse control capturing the windows mouse to perform a mouse-look control, and (3) clip cursor mouse control capturing the windows mouse to stay within the window of the app or game.

# Using the sample

This sample uses mouse control for interaction with the UI and game modes. Press the right mouse button to exit the different game modes and return to the UI.

# Implementation notes

Throughout the sample, 3 different mouse control modes are commonly referenced.

## Absolute Mode (default)

In this mode, the mouse cursor is controlled by the OS and is free to move off of the application’s window. This is the default window interaction used daily with most programs. The ICoreWindow event “PointerMoved” is used to monitor and read the mouse movements.

See [Responding to touch input (DirectX and C++)](https://msdn.microsoft.com/en-us/library/windows/apps/xaml/hh994925.aspx)

## Relative Mode

This mode is sometimes referred to as “mouse-look” or “mouse-move” camera mode. Some apps or games might prefer the mouse to be used as a general input device based on the relative movement of the mouse instead of the mouse location. This is commonly seen in first person shooter games.

To enter this mode in the sample, select the first person shooter tile. This mode requires a way to exit the mode and return to absolute mouse mode. In this sample, pressing “Esc” will exit the mode and return to the menu.

To implement this mode, set the system cursor to null to hide it and the mouse movement is read through the ICoreWindow event “MouseMoved”.

See [Developing mouse controls (DirectX and C++)](https://msdn.microsoft.com/en-us/library/windows/apps/xaml/hh994925.aspx)

## Clip Cursor Mode

Clip cursor mode is a variant of absolute mode where the mouse is constrained to the bounds of the application’s window. To enter this mode, select the real-time strategy tile. This mode also requires a method to exit and return to absolute mode. Pressing “Esc” will exit the mode and return to the menu.

To implement this mode, set the system cursor to null to hide it and read mouse movement through the ICoreWindow event “MouseMoved” (similar to relative mode). Then the application must draw its own mouse cursor and manually update the location as well as ensure the cursor never leaves the window. When exiting clip cursor mode and returning to absolute, the system cursor will appear where it was previously hidden. Therefore when the application stops drawing its own cursor, the system cursor must be set to the location that the application cursor was previously being drawn.

# Known issues

# Update history

Initial release February 2018.

Updated in June 2018 to support 4k which requires a dips to pixels conversion of the mouse position.

# Privacy statement

When compiling and running a sample, the file name of the sample executable will be sent to Microsoft to help track sample usage. To opt-out of this data collection, you can remove the block of code in Main.cpp labeled “Sample Usage Telemetry”.

For more information about Microsoft’s privacy policies in general, see the [Microsoft Privacy Statement](https://privacy.microsoft.com/en-us/privacystatement/).