Introducing the Windows Sensor and Location Platform

November 16, 2009

Abstract

This paper provides an overview of the development platform in Windows 7 that facilitates the use of sensors, including location sensors such as global positioning system (GPS) devices.

This information applies to the Windows 7 operating system.

For the latest information, see:
 <http://www.microsoft.com/whdc/device/sensors/default.mspx>

Disclaimer: This is a preliminary document and may be changed substantially prior to final commercial release of the software described herein.

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This White Paper is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

Unless otherwise noted, the example companies, organizations, products, domain names, e-mail addresses, logos, people, places and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, email address, logo, person, place or event is intended or should be inferred.

© 2009 Microsoft Corporation. All rights reserved.

Microsoft and Windows are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

Document History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date | Change |  |  |  |
| November 16, 2009 | Minor revisions to remove mention of web developers and web pages |
| October 24, 2008 | First publication |

Contents

[Introduction 4](#_Toc246140552)

[Sensor Device Overview 4](#_Toc246140553)

[Privacy and Security 6](#_Toc246140554)

[Architecture 6](#_Toc246140555)

[Sensor Class Extension 7](#_Toc246140556)

[Location Sensor Drivers 8](#_Toc246140557)

[Logical Sensor Drivers 8](#_Toc246140558)

[Windows Driver Kit Features for Sensors and Location 9](#_Toc246140559)

[Sensor API 9](#_Toc246140560)

[Windows SDK Features for Sensors 9](#_Toc246140561)

[Location API 9](#_Toc246140562)

[Choosing a Location Sensor 10](#_Toc246140563)

[Windows SDK Features for Location 10](#_Toc246140564)

[User Control Panel 10](#_Toc246140565)

[For More Information 10](#_Toc246140566)

# Introduction

As humans, we rely on our senses to provide us with information about the world around us. When we create machines to take on some of our work, we add sensor mechanisms so that the machines can respond appropriately to changing conditions.

For example, automobile engines typically use a variety of sensors. These sensors are monitored by an onboard computer that continuously adjusts settings, such as engine timing, to maximize power and efficiency. A television may use an ambient light sensor to adjust the brightness of the picture to match changing room conditions. Even something as simple as a doorbell button acts as a rudimentary sensor to detect a human presence at the door.

Although the purely mechanical doorbell fulfills its purpose, the information that complex sensors provide becomes far more powerful when it is combined with software. Modern sensors can provide a lot of data very quickly, and in a variety of formats, so software provides a natural mechanism for making sense of sensor data.

Today, software developers can write programs that use sensors, but a lack of standardization makes programming for sensors an arduous task. After completion, a sensor-based program usually forever depends on a particular type of hardware. Using one or more vertical solutions to enable deployment of a software-based system has limited the integration of sensors with computer hardware, and Windows®-based computers are no exception.

Windows 7 includes native support for sensors, expanded by a new development platform for working with sensors, including location sensors such as GPS devices. The Windows Sensor and Location platform provides a standard way for device manufacturers to expose sensor devices to software developers and consumers, while providing developers with a standardized application programming interface (API) for working with sensors and sensor data.

This document provides an overview of the Windows Sensor and Location platform, discusses its various parts, and describes how the pieces work together to provide a comprehensive system for working with sensors.

# Sensor Device Overview

Sensors come in many configurations and, from a certain perspective, almost anything that provides data about physical phenomena can be called a sensor. Although we usually think of sensors as hardware components, logical sensors can also provide information through emulation of sensor functionality in software or firmware. Also, a single hardware device can contain multiple sensors. The following table shows different kinds of sensors and their use.

|  |  |  |
| --- | --- | --- |
| Sensor type | Sensor placement | Typical use |
| GPS | Built into the computer or connected to the computer | Location and orientation |
| Accelerometer | Built into the computer or attached to a person | Motion, orientation, and location |
| Proximity  | Built into the computer or nearby | Presence and spatial awareness |
| Light | Built into the computer, nearby, or remote | Ambient light conditions, such as room lighting or outdoor daylight |
| RFID | Built into the computer or attached to a person | Identification of physical items, automatic logon, or location of people |
| Compass | Attached to a person | Orientation |
| Camera | Built into the computer, attached to a person, or remote | Recording, object identification, or facial recognition |
| Microphone | Built into the computer, attached to a person, or remote | Ambient noise |
| Temperature | Built into the computer or remote | Indoor or outdoor ambient temperature and cooking temperature |
| Moisture | Remote | Detection of water leaks or lawn sprinkler operation |
| Motion detector | Built into the computer or remote | Security systems or sensing of motion near the computer |
| Traffic | Remote, usually public | Travel time predictions |
| Weather station | Remote, possibly public | Weather conditions and predictions |

The Windows Sensor and Location platform organizes sensors into categories, which represent broad classes of related sensor devices and types, which represent specific kinds of sensors. For example, a sensor in a video game controller that detects the position and movement of a player's hand (perhaps for a video bowling game) would be categorized as an Orientation sensor, but its type would be a 3-D Accelerometer. In code, Windows 7 represents categories and types by using globally unique identifiers (GUIDs), many of which are predefined. Device manufacturers can create new categories and types by defining and publishing new GUIDs.

Location devices comprise one particularly interesting category. By now, most people are familiar with GPS sensors. In Windows 7, a GPS is a type of sensor and is part of the Location category. The Location category includes other sensor types, some of which are software based, such as a sensor that provides location information based on an Internet address, a sensor that determines location based on nearby cell phone towers, or a sensor that that reads information from a connected wireless network hub and looks up the hub location in a database.

# Privacy and Security

The Windows Sensor and Location platform helps to ensure that sensor data remains private, when appropriate, in the following ways.

* Sensors are off by default. The platform design presumes that any sensor can provide personal data, so each sensor is disabled until the user provides explicit consent to access the sensor data through the Location and Other Sensors Control Panel.
* Windows provides disclosure messages to the user. These messages help users to understand how using sensors can affect the privacy of personal data.
* Providing permission for a sensor requires administrator rights.
* When enabled, a sensor device works for all programs that are running under a particular user account (or for all user accounts), including non-interactive users and services, such as ASPNET or SYSTEM. Because the sensor and location platform cannot ensure that data that is exposed to one process is secure from access by another process, users must give permission for all programs to access a particular sensor device (for a particular user account or all accounts on the computer).
* Programs that use sensors can call a method to open a system dialog box that prompts users to enable required sensor devices. This feature makes it easy for developers and users to ensure that sensors work when programs need them, while maintaining user control of disclosure of sensor data.
* Sensor drivers use a special object that processes all I/O requests and ensures that only programs that have user permission can access sensor data.

# Architecture

The Windows Sensor and Location platform consists of the following components:

* The device driver interface (DDI). Windows 7 provides a standard mechanism for sensor devices to connect to the computer and to provide data to other subsystems.
* The Windows Sensor API, which provides a set of methods, properties, and events for working with connected sensors and sensor data.
* The Windows Location API, which is built on the Windows Sensor API and provides a set of programming objects, including scripting objects, for working with location information.
* The Location and Other Sensors Control Panel, which gives users control over settings for sensors, including location sensors.
* Privacy controls that enable users to manage which sensors applications can use.

Figure 1, on the following page, shows the relationship between these components.

Figure 1. Windows Sensor and Location platform architecture

**Device Driver Interface**

Sensor manufacturers can create device drivers to connect sensors to Windows 7. Sensor device drivers are implemented by using the Windows Portable Devices (WPD) driver model, which is based on the Windows user-mode driver framework (UMDF). Because these technologies are established, experienced device driver programmers will find writing a sensor driver to be a familiar task. The Sensor DDI uses certain UMDF and WPD data types and interfaces, and also defines sensor-specific WPD commands and parameters, where appropriate.

## Sensor Class Extension

To help make it easier to write a device driver that exposes a sensor to Windows 7 (and to the Sensor and Location platform in particular), the operating system includes a driver class extension. This COM object is a required component for sensor device drivers. It provides a simple set of interfaces that enable programmers to implement a sensor driver without writing a lot of boilerplate code. Using the class extension provides several additional benefits for programmers, including the following:

* Ensuring that other components of the Windows Sensor and Location platform—such as the Sensor API, Location API, and Control Panel—can discover and enumerate connected sensors.
* Helping to ensure that user privacy is well protected. The class extension enforces appropriate access control restrictions for sensors that handle personal information.
* Providing a standard mechanism for retrieving data from the driver and raising event notifications through the API layers.

Figure 2 shows the relationship between a sensor, its driver, and the sensor driver class extension.

Figure 2. Sensor driver with sensor class extension

The Sensor Class Extension exposes two COM interfaces. The ISensorClassExtension interface provides access to methods that the sensor driver uses to communicate with the sensor platform and, therefore, client applications. The ISensorDriver interface provides callback methods that the class extension uses to retrieve information about features that the sensor supports, as well as the actual sensor data. The class extension also activates callback methods to provide notifications to the sensor driver.

## Location Sensor Drivers

You create drivers for location sensors by using the same driver model and class extension as for any other category of sensor. Location drivers identify the location sensor as belonging to the Location category. They then expose the appropriate data fields for the specific sensor type and identify the type of location data report that the driver provides.

## Logical Sensor Drivers

You can create drivers for logical sensors. These drivers work like other sensor drivers, but do not depend on hardware. The ILogicalSensorManager interface, which is part of the Sensor API, provides methods for managing logical sensor connections.

## Windows Driver Kit Features for Sensors and Location

The Windows 7 Driver Kit includes sample code and documentation to help guide driver developers through the process of creating a sensor or location device driver.

## Sensor API

The Windows Sensor API enables C++ developers to create sensor-based programs by using a set of COM interfaces. The API defines interfaces for performing common sensor programming tasks, including managing sensors, working with collections of sensors, working with individual sensors, and working with data.

ISensorManager includes methods for retrieving sensors by category, type, or ID. The interface also provides a method to open a system dialog box that requests permission to use particular sensors, and methods to register for sensor events. ISensorManagerEvents provides notifications when a sensor connects.

ISensorCollection provides a way to work with collections of sensors and provides typical collection functionality, such as Add, Remove, GetCount, and GetAt(index).

ISensor represents a particular sensor. This interface provides methods for setting and retrieving sensor properties, getting state information, retrieving sensor data, and registering for events. ISensorEvents provides notifications about changes in sensor state, changes in data, sensor disconnects, and custom events for the sensor.

ISensorDataReport represents a data report, which provides a snapshot of data from a particular sensor. Sensor drivers mark data reports with a time stamp so that applications can determine whether a particular set of data meets timeliness requirements. This interface enables developers to access a single data point at a time by using a reference to a PROPERTYKEY or to retrieve multiple data values through a single method call by using a pointer to the WPD IPortableDeviceKeyCollection interface.

### Windows SDK Features for Sensors

The Windows 7 SDK includes headers, documentation, samples, and tools to help guide software developers in using sensors with programs in Windows 7. A single header, named sensors.h, declares common, predefined sensor events, properties, categories, types, and data types. Tools include a virtual sensor, which emulates a connected hardware device, and a sensor monitoring program that displays sensor data and other events.

## Location API

Built on the Sensor platform, the Location API provides an easy way to retrieve data about geographic location while protecting user privacy. The Location API provides its functionality through a set of COM interfaces, which represent objects. These objects can be used by programmers who are familiar with using COM through the C++ programming language, or in scripting languages, such as JScript. Scripting support gives easy access to location data for projects, such as gadgets.

The default COM interface for the Location API, named ILocation, represents a Location object. Through this object you can register or unregister for location data report events, specify the frequency of report events, retrieve a particular location report by type, and open a system dialog box to prompt the user for permission to access location data. Location-enabled programs can receive events when the current location changes and when a status change occurs for a particular data report type.

The Location API defines two location report types. ILatLongReport represents the current latitude and longitude, including error radius data. ICivicAddressReport represents a typical street address and provides individual methods to retrieve address parts, such as the city or state/province. Civic address reports also include information about the level of detail that the report provides. For example, a particular report may contain accurate information about the current city, but not the current street address.

### Choosing a Location Sensor

The Location API uses a predefined set of rules to choose the best location sensor that is available. Except for requesting user permission, you do not need to manage location sensors. Instead, you can focus on making the best use of the available data.

### Windows SDK Features for Location

The Windows 7 SDK includes headers, documentation (including scripting reference documentation), samples, and tools to help guide software developers in using location information with programs in Windows 7.

## User Control Panel

Windows 7 includes a control panel that provides users with the ability to enable or disable sensors systemwide or for each user. Because some sensors can expose sensitive data, this user interface lets users control which programs have access to each sensor. Users can also view sensor properties and change the sensor description that appears in the user interface.

# For More Information

For more information about the Windows Sensor and Location platform, send e-mail to:

sensext@microsoft.com

For more information about location in Windows® 7, send e-mail to:

locext@microsoft.com