**Nvidia Sensor Client**

(NVSC)

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Comment |
| 0.0.1 | 15-APR-2021 | Erik Lilliebjerg | Initial draft. |

**List of Approvers**

|  |  |  |
| --- | --- | --- |
| Approver | Title / Function | |
| Robert Collins | | Sensor Manager |
| Erik Lilliebjerg | | SW Architect |

Table of Contents

[1 Introduction 5](#_Toc69654097)

[1.1 Document Scope 5](#_Toc69654098)

[1.2 Glossary and Acronyms 5](#_Toc69654099)

[2 Functional Overview 5](#_Toc69654100)

[2.1 Overview 5](#_Toc69654101)

[2.1.1 The Sensor List 5](#_Toc69654102)

[2.1.2 Opening a NVSC Session 5](#_Toc69654103)

[2.1.3 Sensor Operation 6](#_Toc69654104)

[2.1.4 Sensor Events 6](#_Toc69654105)

[2.1.5 Closing a NVSC Session 6](#_Toc69654106)

[3 Functions 6](#_Toc69654107)

[3.1 getSensors 6](#_Toc69654108)

[3.2 clientOpen 7](#_Toc69654109)

[3.3 setSensorBatch 7](#_Toc69654110)

[3.4 setSensorAble 8](#_Toc69654111)

[3.5 setSensorFlush 8](#_Toc69654112)

[3.6 getSensorEvent 9](#_Toc69654113)

[3.7 clientClose 9](#_Toc69654114)

Table of Tables

[Table 1: Glossary 5](#_Toc52192096)

# Introduction

## Document Scope

The Nvidia Sensor framework (NVS) was originally designed as an [Android sensor HAL](https://source.android.com/devices/sensors/hal-interface). It has evolved into a superset of the Android sensor API. One of the extensions is the NVS Client API (NVSC), an extension of NVS over IPC. Since NVS is heavily based on the Android sensor API, the Android sensor specification will be referred to for most of the NVS functions and data structures.

## Glossary and Acronyms

Table 1: Glossary

|  |  |
| --- | --- |
| **Abbreviation / Term** | **Definition** |
| NVS | NVidia Sensor framework. |
| NVSC | NVS over IPC. |
| IPC | Inter-Process Communication |

# Functional Overview

## Overview

The following are the typical steps to using NVSC:

1. Locate the desired sensor.
2. Become a client of NVS by opening a session.
3. Operate the desired sensor.
4. Poll for sensor events.
5. When done, the client can either disable the sensor and stop polling for events, or if the client has no more need for the NVS framework it can close the session.

### The Sensor List

The sensor list is a list of all the sensors NVS is in control of. This function can be called without opening an NVSC session thereby minimizing those resources should the desired sensor not be found. The functionality is identical to the Android function [get\_sensors\_list](https://source.android.com/devices/sensors/hal-interface). It is highly recommended to read the [Android sensor specification](https://android.googlesource.com/platform/hardware/libhardware/+/master/include/hardware/sensors.h) for the sensor description in the sensor list sensor\_t structure. Once a desired sensor is identified in this list, the structure member, handle, will be used when referring to the sensor.

### Opening a NVSC Session

Opening an NVSC session and using the client handle in all NVSC subsequent calls allows NVS to track its clients. This allows a single sensor to be used by multiple clients. NVS will use the highest rate request of all the clients using a sensor.

### Sensor Operation

Once a sensor and its handle are identified and an NVSC session is opened, the client can now operate the sensor. The most basic of operations is to set the sensor rate and enable it. This is done with the setSensorBatch and setSensorAble NVSC functions. For the more complex batching and flush operations, see the Android specification [here](https://source.android.com/devices/sensors/batching).

### Sensor Events

Use the getSensorEvent NVSC function to poll for sensor events. It is highly recommended to read the [Android sensor specification](https://android.googlesource.com/platform/hardware/libhardware/+/master/include/hardware/sensors.h) for the sensor data returned in the sensor event sensors\_event\_t structure.

### Closing a NVSC Session

Closing an NVSC session is typically used during system shutdown. Unless a client never intends to use NVSC during the remainder of system uptime, it does not need to close the NVSC session.

When an NVSC session is closed NVS will perform any client cleanup such as disabling the client’s sensors and sensor event polling.

# Functions

## getSensors

**Description**

Returns list of available sensors.

It’s the caller’s responsibility to delete the sensor list memory returned.

See the [Android sensor specification](https://android.googlesource.com/platform/hardware/libhardware/+/master/include/hardware/sensors.h) for the sensor description in the sensor list sensor\_t structure.

**Precondition**

**Parameters**

|  |  |  |
| --- | --- | --- |
| **Parameter Name** | **Input/Output** | **Description** |
| \*\*snsrs | **Input** | Pointer for the pointer to the sensor list.  If this is NULL then just the sensor count is returned. |

**Returns**

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| >= 0 | The number of sensors in the sensor list. |
| -EPERM | Fatal error. NVS is not operational. |
| -EAGAIN | NVS is still initializing. |
| -EBUSY | NVS is encountering errors. This isn’t good but it is possible that another attempt at a later time may work. |
| -ENOMEM | There are system resource errors. |

## clientOpen

**Description**

Open a session with NVS.

The client handle placed at clnts will be used in all subsequent NVSC function calls.

**Precondition**

**Parameters**

|  |  |  |
| --- | --- | --- |
| **Parameter Name** | **Input/Output** | **Description** |
| \*\*clnts | **Input** | Pointer for the client handle. |

**Returns**

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| 0 | Success. |
| -EINVAL | No clnts pointer. |
| -EPERM | Fatal error. NVS is not operational. |
| -EAGAIN | NVS is still initializing. |
| -EBUSY | NVS is encountering errors. This isn’t good but it is possible that another attempt at a later time may work. |
| -ENOMEM | There are system resource errors. |

## setSensorBatch

**Description**

Set sensor rate and timeout.

See Android documentation for [batching](https://source.android.com/devices/sensors/batching).

**Precondition**

**Parameters**

|  |  |  |
| --- | --- | --- |
| **Parameter Name** | **Input/Output** | **Description** |
| \*clnt | **Input** | Client handle. |
| handle | **Input** | Sensor handle. |
| nsPeriod | **Input** | Sensor rate period in nanoseconds. |
| nsTimeout | **Input** | Sensor event timeout in nanoseconds. |

**Returns**

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| 0 | Success. |
| -EINVAL | Invalid input. |
| -EPERM | Fatal error. NVS is not operational. |
| -EFAULT | Client handle unknown. |

## setSensorAble

**Description**

EnAble/DisAble a sensor.

**Precondition**

**Parameters**

|  |  |  |
| --- | --- | --- |
| **Parameter Name** | **Input/Output** | **Description** |
| \*clnt | **Input** | Client handle. |
| handle | **Input** | Sensor handle. |
| enable | **Input** | 0=Disable, 1=Enable |

**Returns**

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| 0 | Success. |
| -EINVAL | Invalid input. |
| -EPERM | Fatal error. NVS is not operational. |
| -EFAULT | Client handle unknown. |

## setSensorFlush

**Description**

Flushes the batching FIFO.

See Android documentation for [batching](https://source.android.com/devices/sensors/batching).

**Precondition**

**Parameters**

|  |  |  |
| --- | --- | --- |
| **Parameter Name** | **Input/Output** | **Description** |
| \*clnt | **Input** | Client handle. |
| handle | **Input** | Sensor handle. |

**Returns**

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| 0 | Success. |
| -EINVAL | Invalid sensor handle. |
| -EPERM | Fatal error. NVS is not operational. |
| -EFAULT | Client handle unknown. |

## getSensorEvent

**Description**

Polls for sensor events.

See [Android documentation](https://source.android.com/devices/sensors/hal-interface) for sensor event polling.

See [here](https://android.googlesource.com/platform/hardware/libhardware/+/master/include/hardware/sensors.h) for the sensors\_event\_t structure definition.

**Precondition**

**Parameters**

|  |  |  |
| --- | --- | --- |
| **Parameter Name** | **Input/Output** | **Description** |
| \*clnt | **Input** | Client handle. |
| \*evnt | **Input** | Pointer to the sensors\_event\_t data buffer. |
| nEvnt | **Input** | The number of sensors\_event\_t events the data buffer can hold. |
| msPoll | **Input** | msPoll < 0: poll will block until events.  msPoll == 0: poll will return if no events.  msPoll > 0: poll returns after events or the millisecond timeout specified by msPoll. |

**Returns**

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| >= 0 | The number of events in put in the data buffer. |
| -EINVAL | Invalid input. |
| -EPERM | Fatal error. NVS is not operational. |
| -EFAULT | Client handle unknown. |

## clientClose

**Description**

Closes an open session with NVS.

**Precondition**

**Parameters**

|  |  |  |
| --- | --- | --- |
| **Parameter Name** | **Input/Output** | **Description** |
| \*clnt | **Input** | Client handle. This is allowed to be NULL for efficiency. |

**Returns**

Void.