NVAPI Open Source SDK
for Driver Release 560

Release Notes

Table of Contents

NVAPI Release Notes

Introduction

NVAPI is NVIDIA Corporation's core software development kit that allows access to NVIDIA GPUs and drivers on all Windows platforms. NVAPI provides support for categories of operations that range beyond the scope of those found in familiar graphics APIs such as DirectX and OpenGL.

This release contains a version of `nvapi.h`, provided under MIT license, to enable open-source re-implementations of NVAPI for Windows emulation environments.

For those interested in developing applications using the NVAPI Developer SDK on Windows, NVIDIA recommends using the NVAPI Developer SDK, which is available at https://developer.nvidia.com/nvapi.

The following files are provided by NVIDIA:

> `nvapi.h`

Starting with release 560, this file is split into the following files, which contain the interface constants, structure definitions, and function prototypes for the NVAPI interface.

- o `nvapi.h`
- o `nvapi_lite_common.h`
- o `nvapi_lite_d3dext.h`
- o `nvapi_lite_salend.h`
- o `nvapi_lite_salstart.h`
- o `nvapi_lite_sli.h`
- o `nvapi_lite_stereo.h`
- o `nvapi_lite_surround.h`

> `nvapi_interface.h`

This file is a reference for mapping the NVAPI identifiers to functions.

> `NvApiDriverSettings.h`
> `NvApiDriverSettings.c`

These files are used to query and change driver settings. For more information, see the NVIDIA Driver Settings Programming Guide.

> `nvHLSLExtns.h`

> `nvHLSLExtnslnternal.h`

> `nvShaderExtnEnums.h`

> `license.txt`

> `nvapi.lib, nvapi64.lib : NVAPI static libraries provided under MIT license.`

These release notes describe the changes made in the NVAPI Open Source Interface for this release.

NVAPI Runtime

The NVAPI runtime (NVAPI DLL) provides the following key functions:

> `nvapi_QueryInterface():`

Maps a 32-bit identifier to a function pointer.

```
void *nvapi_QueryInterface(NvU32 id);
```

The NVAPI application will call `nvapi_QueryInterface()` to get individual NVAPI function pointers from NVAPI DLL.

Refer to `nvapi_interface.h` for a mapping of identifiers to NVAPI function names.

> `NvAPI_Initialize():`

```
NvAPI_Status __cdecl NvAPI_Initialize()
{
    return NVAPI_OK;
}
```

> `NvAPI_Unload():`

```
NvAPI_Status __cdecl NvAPI_Unload()
{
    return NVAPI_OK;
}
```

Changes in NVAPI for Driver Release 560

New Functions

> Added `NvAPI_D3D12_CreateCubinComputeShaderExV2`

> Added `NvAPI_D3D12_GetCudaMergedTextureSamplerObject`

> Added `NvAPI_D3D12_GetCudaIndependentDescriptorObject`

New/Updated Structures

> Added `bIsVRRPossible` to `NV_GET_VRR_INFO_V1`
> Added `bIsVRRRequested` to `NV_GET_VRR_INFO_V1`
> Added `bIsVRRIndicatorEnabled` to `NV_GET_VRR_INFO_V1`
> Added `bIsDisplayInVRRMode` to `NV_GET_VRR_INFO_V1`
> Added `sleepIntervalUs` to `NV_GET_SLEEP_STATUS_PARAMS_V1`
> Added `bUseGameSleep` to `NV_GET_SLEEP_STATUS_PARAMS_V1`
> Added `NVAPI_D3D12_CREATE_CUBIN_SHADER_PARAMS`
> Added `NVAPI_D3D12_GET_CUDA_MERGED_TEXTURE_SAMPLER_OBJECT_PARAMS`
> Added `NVAPI_D3D12_GET_CUDA_INDEPENDENT_DESCRIPTOR_OBJECT_PARAMS`

New/Updated Enums

> Added `NVAPI_D3D12_GET_CUDA_INDEPENDENT_DESCRIPTOR_OBJECT_TYPE`

New/Updated Unions

> None

New Macros

> None

New Errors

> None

TCC Support

> None

MCDM Support

> None

Deprecated NVAPI Functions

> None

NVAPIDriverSettings Additions/Removals

> **>** None

HLSL Extensions

The following header files are provided to expose intrinsic functions that are not present in the HLSL instruction set. For more information, see [Unlocking GPU Intrinsics in HLSL](#).

> `nvShaderExtnEnums.h`

> This header file contains all the shader extension opcodes. The application should call `NvAPI_D3D12_IsNvShaderExtnOpCodeSupported()` or `NvAPI_D3D11_IsNvShaderExtnOpCodeSupported()` to check for support for these NV shader extension opcodes.

> `nvHLSLExtns.h`

> This header file needs to be included in the application HLSL shader code to use the NVIDIA shader extensions.

> `nvHLSLExtnsInternal.h`

> This file contains internal functions that are not to be called by the application directly.

NVAPI Security Information

User administrator privilege is required to access certain driver features per NVIDIA's overall security vision. This helps mitigate the impact of malware.

Each API requiring administrator access will return an `NVAPI_INVALID_USER_PRIVILEGE` error when run with standard user privilege.

The application will require Administrator privileges to access this API, which can be elevated to a higher permission level by selecting "Run as Administrator" in Admin approval mode.

Sample Code

The SDK package contains the `Sample_Code` directory, which provides code examples for the following features:

Feature	Sample Code Subdirectory	Sample Code
Custom timing	`CustomTiming`	`CustomTiming.cpp`
Display color control	`DisplayColorControl`	> `DisplayColorControl.cpp` > `NVHelper.cpp` > `NVHelper.h`
Display configuration	`DisplayConfiguration`	> `DisplayConfiguration.cpp` > `targetver.h`
GPU handle enumeration	`GPUHandleEnumeration`	`gpuHandleEnumeration.c`
QSYNC event registration	`QSYNC_Event_Registration`	`QSYNC_Event_Registration.cpp`
Sync configuration	`Sync_Configuration`	> `Sync_Configuration.cpp` > `targetver.h`
I2C	`i2c`	> `i2c.cpp` > `targetver.h`

NVIDIA Corporation | 2788 San Tomas Expressway, Santa Clara, CA 95051
http://www.nvidia.com