Class name: nvidiaapi

Header:

#include <nvml.h>

#include <string>

#include "gpumonitor.h"

Name space: std

Description: nvidianvml class provides the interface to get clocking data and status information from Nvidia.

Constructors:

**NvidiaNVML** ();

Parameters: None.

Return: None.

Task: Initialize the Nvidia variables, special data structure for storing version, status, flags, and const parameter that will be used.

Destructor:

~ **NvidiaNVML** ():

Parameters: None.

Return: None.

Task: Free the necessary space used in the class.

Key Private Variables:

|  |  |
| --- | --- |
| NvU32 \_gpuCount; | Counts of GPU |

Public Methods:

bool **initNVML**();

Parameters: None.

Return: Indicator of the status of connection.

Task: initialize the connection of NVML interface to the Nvidia.

unsigned int **getGPUCount**();

Parameters: None.

Return: None.

Task: get the number of GPU.

void **shutDownNVML**();

Parameters: None.

Return: None.

Task: close the connection of NVML interface to the Nvidia.

std::string **getGPUName**(unsigned int index);

Parameters: GPU index.

Return: Fan Speed.

Task: access Nvml API to query hardware to get the speed of fan.

int **getGPUTemp**(unsigned int index);

Parameters: GPU index.

Return: Temperature in Celsius degree.

Task: access sensor to get the temperature of target GPU device.

int **getTempLimit**();

Parameters: None

Return: upper bound of temperature which gpu should not be higher than this.

Task: query the temperature upper bound that has been set already.

int **getFanSpeed**(unsigned int index);

Parameters: GPU index.

Return: Fan Speed.

Task: access Nvml API to query hardware to get the speed of fan.

int **getMemClock**(unsigned int index);

Parameters: GPU index.

Return: Memory clocking.

Task: access Nvml API to query Memory clocking, which represents the running memory clocking.

int **getGPUClock**(unsigned int index);

Parameters: GPU index.

Return: GPU clocking.

Task: access Nvml API to query GPU clocking, which represents the running GPU clocking.

int **getPowerDraw**(unsigned int index);

Parameters: GPU index.

Return: Power Draw.

Task: access Nvml API to get the draw of Power.

unsigned int\* **getAllTemp**();

Parameters: None.

Return: An array of GPU temperature.

Task: access Nvml API to query all GPU temperature, which represents the running GPU temperature.

int **getHigherTemp**();

Parameters: None.

Return: Maximum of GPU temperature.

Task: access Nvml API to query all GPU temperature, and return the maximum of the running GPU temperature.

int **getLowerTemp**();

Parameters: None.

Return: Minimum of GPU temperature.

Task: access Nvml API to query all GPU temperature, and return the minimum of the running GPU temperature.

int **getHigherFanSpeed**();

Parameters: None.

Return: Max Fan Speed.

Task: access Nvml API to query all hardware to get the maximum speed of fan among them.

unsigned int\* **getAllFanSpeed**();

Parameters: None.

Return: An array of Fanspeed.

Task: access Nvml API to query to get the speed of fan of all devices.

int **getLowerFanSpeed**();

Parameters: None.

Return: Min Fan Speed.

Task: access Nvml API to query all hardware to get the minimum speed of fan among them.

int **getMemMaxClock**();

Parameters: GPU index.

Return: Memory clocking.

Task: access Nvml API to query Memory clocking, which represents the running memory clocking. And select the max memory clock among them.

unsigned int\* **getAllMemClock**();

Parameters: None.

Return: An array of Memory Clock.

Task: access Nvml API to query to get the Memory Clock of all devices.

int **getMemLowerClock**();

Parameters: GPU index.

Return: Memory clocking.

Task: access Nvml API to query Memory clocking, which represents the running memory clocking. And select the min memory clock among them.

int **getGPUMaxClock**();

Parameters: GPU index.

Return: Max GPU clocking.

Task: access Nvml API to query GPU clocking, which represents the running GPU clocking. And select the max GPU clock among them.

unsigned int\* **getAllGPUClock**();

Parameters: None.

Return: An array of GPU Clock.

Task: access Nvml API to query to get the GPU Clock of all devices.

int **getGPUMinClock**();

Parameters: GPU index.

Return: Min GPU clocking.

Task: access Nvml API to query GPU clocking, which represents the running GPU clocking. And select the min GPU clock among them.

int **getMaxPowerDraw**();

Parameters: GPU index.

Return: Max Power Draw value.

Task: access Nvml API to query power draw, which represents the running power. And select the max power draw among them.

unsigned int\* **getAllPowerDraw**();

Parameters: None.

Return: An array of Power Draw.

Task: access Nvml API to query to get the Power Draw of all devices.

int **getMinPowerDraw**();

Parameters: GPU index.

Return: Min Power Draw value.

Task: access Nvml API to query power draw, which represents the running power. And select the min power draw among them.

int **getPowerDrawSum**();

Parameters: None.

Return: Summation of Power Draw.

Task: access Nvml API to query to get the Power Draw of all devices and sum them up.

QList<GPUInfo> **getStatus**();

Parameters: None.

Return: A list that shows all information related to the device status.

Task: access Nvidia API to query to all kinds of that, store the status information in a list with given order to represent the macro status.