GetDevInfo Documentation

Release 2.0.0

Hamish McIntyre-Bhatty

Jul 11, 2022

CONTENTS

CHAPTER

ONE

LICENSE

CHAPTER

TWO

TABLE OF CONTENTS

2.1 Introduction

This is GetDevInfo's API documentation. Mostly this documentation is derived from docstrings in the source code, with a few exceptions.

2.2 Running GetDevInfo

2.2.1 Calling GetDevInfo from the commandline

Run with:

python3 -m getdevinfo

Warning: Breaking change: Support for calling with "python3 -m getdevinfo.getdevinfo was removed in v2.0.0.

2.2.2 Calling GetDevInfo from a Python program

Run with:

>>>import getdevinfo >>>getdevinfo.get_info()

Note: For versions prior to v2.0.0, you will need to call with "getdevinfo.getdevinfo.get_info()".

2.3 Important: Error reporting in GetDevInfo

GetDevInfo does not make use of a logger. Instead, GetDevInfo will create the file "/tmp/getdevinfo.errors" when errors occur.

Note: There was no error reporting capability at all for versions before v2.0.0.

Note: If you're running GetDevInfo directly from the commandline, the errors are printed to the standard output instead.

This file should be read, the contents reported or logged, and then the file deleted by programs that are calling GetDevInfo, in order to prevent potential leaking of sensitive disk information.

This is not a major security concern, as it is not expected that sensitive data will be contained in the file in most situations. At worst, a disk UUID or serial number may be present in the file with particular error scenarios.

2.4 Documentation for the output format

This module outputs data in a precisely-formatted dictionary object. In order for it to be useful, this format, and the information that is provided in it, needs to be explained precisely.

This format is the same on Linux, macOS, and Cygwin (Windows), but the macOS and Cygwin versions of this library currently have less functionality, so some of the information isn't present on those platforms version. Instead, placeholders like "N/A" or "Unknown" are used. Those instances will be pointed out here.

Note: On Linux and Cygwin, superuser/administrator privileges are required for GetDevInfo to work correctly.

2.4.1 For each device and partition

A sub-dictionary is created with the name of that disk as its key.

For example: To access the info for /dev/disk1s1, use:

```
>>> DISKINFO['/dev/disk1s1']
```

2.4.2 Inside this sub-dictionary (standard devices)

Various information is collected and organised here.

'Name': The disk's name, stored as a string.

'Type': Whether the disk is a "Device" or "Partition", stored as a string.

For example:

```
>>> DISKINFO['/dev/sda']['Type']
>>> "Device"
```

Note: Due to Cygwin limitations, all disks are considered devices on Cygwin.

'HostDevice': The "parent" or "host" device of a partition, stored as a string. For a device, this is always set to "N/A". For an LVM disk, this is the host device of the containing partition. eg: /dev/sdb.

Example 1:

```
>>> DISKINFO['/dev/sda']['HostDevice']
>>> "N/A"
```

Example 2:

```
>>> DISKINFO['/dev/sde5']['HostDevice']
>>> "/dev/sde"
```

Example 3:

```
>>> DISKINFO['/dev/disk1s3']['HostDevice']
>>> "/dev/disk1"
```

'Partitions': All the partitions a device contains, stored as a list. For partitions, this is always set to [].

Example 1:

```
>>> DISKINFO['/dev/sda1']['Partitions']
>>> []
```

Example 2:

```
>>> DISKINFO['/dev/sda']['Partitions']
>>> ["/dev/sda1", "/dev/sda2", "/dev/sda3"]
```

Example 3:

```
>>> DISKINFO['/dev/disk0']['Partitions']
>>> ["/dev/disk0s1", "/dev/disk0s2"]
```

Note: Not yet available on Cygwin.

'Vendor': The device's/partition's vendor. For a device, this is often the brand. For partitions this is more random, but often has something to do with the file system type, or the OS that created the partition.

Example 1:

```
>>> DISKINFO['/dev/sda']['Vendor']
>>> "VBOX"
```

Example 2:

```
>>> DISKINFO['/dev/sda1']['Vendor']
>>> "Linux"
```

Example 3:

```
>>> DISKINFO['/dev/disk0s1']['Vendor']
>>> "VBOX"
```

Note: Not available for all disks yet on Cygwin.

'Product': The device's product information. Often model information such as a model name/number. For a partition, this is always the same as it's host device's product information, prefixed by "Host Device: ".

Example 1:

```
>>> DISKINFO['/dev/sda']['Product']
>>> "ST1000DM003-1CH1"
```

Example 2:

```
>>> DISKINFO['/dev/sda1']['Product']
>>> "Host Device: ST1000DM003-1CH1"
```

Example 3:

```
>>> DISKINFO['/dev/disk0']['Product']
>>> "HARDDISK"
```

Note: Not available for all disks yet on Cygwin.

'Capacity', and 'RawCapacity': The disk's capacity, in both human-readable form, and program-friendly form. Ignored for some types of disks, like optical drives. The human-readable capacity is rounded to make it a 3 digit number. The machine-readable size is measured in bytes, and it is not rounded.

Note: Not available for all disks yet on Cygwin.

Example:

```
>>> DISKINFO['/dev/sda']['Capacity']
>>> "500 GB"
```

```
>>> DISKINFO['/dev/sda']['RawCapacity']
>>> "500107862016"
```

'Description': A human-readable description of the disk. Simply here to make it easier for a human to identify a disk. On Linux, these are the descriptions provided by lshw (except for logical volumes), and they are fairly basic. On macOS, these are generated using information from diskutil. On Cygwin, these are generated and provide information like the drive letter and bus used (eg ATA).

Example 1:

```
>>> DISKINFO['/dev/sda']['Description']
>>> "ATA Disk"
```

Example 2:

```
>>> DISKINFO['/dev/disk1']['Description']
>>> "Internal Hard Disk Drive (Connected through SATA)"
```

'Flags': The disk's capabilities, stored as a list.

Note: Not yet available on macOS, Cygwin, or for logical volumes on Linux.

For example:

```
>>> DISKINFO['/dev/cdrom']['Flags']
>>> ['removable', 'audio', 'cd-r', 'cd-rw', 'dvd', 'dvd-r', 'dvd-ram']
```

'Partitioning': The disk's partition scheme. N/A for partitions and logical volumes.

Note: Not yet available on macOS.

Example 1:

```
>>> DISKINFO['/dev/sda']['Partitioning']
>>> "gpt"
```

Example 2:

```
>>> DISKINFO['/dev/sdb']['Partitioning']
>>> "mbr"
```

'FileSystem': The disk's file system. N/A for devices.

Note: Not yet available on macOS.

Example:

```
>>> DISKINFO['/dev/sda']['FileSystem']
>>> "ext4"
```

'UUID': This disk's UUID. N/A for devices. Length changes based on filesystem type. For example, vfat UUIDs are shorter.

Note: Not yet available on macOS.

Example:

```
>>> DISKINFO['/dev/sda1']['UUID']
>>> XXXX-XXXX
```

'ID': The disk's ID.

Note: Not yet available on macOS or Cygwin.

Example:

```
>>> DISKINFO['/dev/sda']['ID']
>>> "usb-Generic_STORAGE_DEVICE_000000001206-0:1"
```

'BootRecord', 'BootRecordStrings': The MBR/PBR of the disk. Can be useful in identifying the bootloader that resides there, if any. Stored as a string.

Warning: Breaking change: This was a bytestring until GetDevInfo v2.0.0.

Note: Not yet available on macOS.

2.4.3 Inside this sub-dictionary (specifics for LVM disks on Linux)

These are keys that are only present for LVM disks (where "Product" is "LVM Partition").

'Aliases': Any aliases the disk has. LVM disks can often be accessed using multiple different names. This is a list of those names.

Example:

```
>>> DISKINFO['/dev/mapper/fedora/root']['Aliases']
>>> ['/dev/mapper/fedora/root', '/dev/fedora--localhost-root']
```

'LVName': The name of the logical volume.

Example:

```
>>> DISKINFO['/dev/mapper/fedora/root']['LVName']
>>> "root"
```

'VGName': The name of the volume group the logical volume belongs to.

Example:

```
>>> DISKINFO['/dev/mapper/fedora/root']['VGName']
>>> "fedora"
```

'HostPartition': The partition that contains this logical volume.

Example:

```
>>> DISKINFO['/dev/mapper/fedora/root']['HostPartition']
>>> "/dev/sda"
```

Note: Not always available depending on disk configuration.

Warning: "UUID" may or may not be available for certain disks.

Warning: "Capacity" and "RawCapacity" may not be available for certain disks.

Warning: "HostPartition" and "HostDevice" may not be available for certain disks.

2.4.4 Inside this sub-dictionary (NVME disks)

Warning: Various standard keys are not available for NVME disks as they aren't supported by lshw.

2.5 Documentation for the getdevinfo module

This is the part of the package that you would normally import and use. It detects your platform (Linux or macOS), and runs the correct tools for that platform.

For example:

```
>>> import getdevinfo
>>> getdevinfo.getdevinfo.get_info()
```

Or, more concisely:

```
>>> import getdevinfo.getdevinfo as getdevinfo
>>> getdevinfo.get_info()
```

Will run the correct tools for your platform and return the collected disk information as a dictionary.

Note: You can import the submodules directly, but this might result in strange behaviour, or not work on your platform if you import the wrong one. That is not how the package is intended to be used, except if you want to use the get_block_size() function to get a block size, as documented for each platform later.

---

getdevinfo.getdevinfo.**get_info**(*name_main=False*)

This function is used to determine the platform you're using (Linux or macOS) and run the relevant tools. Then, it returns the disk information dictionary to the caller.

Returns: dict, the disk info dictionary.

Raises: Hopefully nothing, but if there is an unhandled error or bug elsewhere, there's a small chance it could propagate to here. If this concerns you, you can wrap this code in a try:, except: clause:

```
>>> try:
>>>     get_info()
>>> except:
>>>     #Handle the error.
```

Usage:

```
>>> disk_info = get_info()
```

getdevinfo.getdevinfo.**run**()

Allows the module to be run with -m.

2.6 Documentation for the linux module

This is the part of the package that contains the tools and information getters for Linux. This would normally be called from the getdevinfo module, but you can call it directly if you like.

Note: You can import this submodule directly, but it might result in strange behaviour, or not work on your platform if you import the wrong one. That is not how the package is intended to be used, except if you want to use the get_block_size() function to get a block size, as documented below.

Warning: Feel free to experiment, but be aware that you may be able to cause crashes, exceptions, and generally weird situations by calling these methods directly if you get it wrong. A good place to look if you're interested in this is the unit tests (in tests/).

Warning: This module won't work properly unless it is executed as root.

getdevinfo.linux.**assemble_lvm_disk_info**(*line_counter*, *testing=False*)

Private, implementation detail.

This function is used to assemble LVM disk info into the dictionary.

Like get_device_info(), and get_partition_info(), it uses some of the helper functions here.

Args:

line_counter (int): The line in the output that informtion for a particular logical volume begins.

Kwargs: testing (bool): Used during unit tests. Default = False.

Usage:

```
>>> assemble_lvm_disk_info(<anInt>)
```

OR:

```
>>> assemble_lvm_disk_info(<anInt>, testing=<aBool>)
```

getdevinfo.linux.**compute_block_size**(*stdout*)

Private, implementation detail.

Used to process and tidy up the block size output from blockdev.

Args: stdout (str): blockdev's output.

Returns: int/None: The block size:

- None - Failed!
- int - The block size.

Usage:

```
>>> compute_block_size(<stdoutFromBlockDev>)
```

getdevinfo.linux.**generate_description**(*disk*)

Private, implementation detail.

This function generates a description for the given disk. This is used when the disk came from lsblk's output, so we don't have a description generated from lshw to use.

Args:

disk (str): The name of a device/partition in the disk info dictionary.

Returns: string (str). A description.

Usage:

```
>>> description = generate_description(<aDiskName>)
```

getdevinfo.linux.**get_block_size**(*disk*)

Public

This function uses the blockdev command to get the block size of the given device.

Args:

disk (str): The partition/device/logical volume that we want the block size for.

Returns: int/None. The block size.

- None - Failed!
- int - The block size.

Usage:

```
>>> block_size = get_block_size(<aDeviceName>)
```

getdevinfo.linux.**get_boot_record**(*disk*)

Private, implementation detail.

This function gets the MBR/PBR of a given disk.

Args: disk (str): The name of a partition/device.

Returns: tuple (string, string). The boot record (raw, any readable strings):

- ("Unknown", "Unknown") - Couldn't read it.
- Anything else - The PBR/MBR and any readable strings therein.

Usage:

```
>>> boot_record, boot_record_strings = get_boot_record(<aDiskName>)
```

getdevinfo.linux.**get_capabilities**(*node*)

Private, implementation detail.

This function gets the capabilities from the structure generated by parsing lshw's XML output.

Args: node: Represents a device/partition.

Returns: list. The capabilities:

- [] - Couldn't find them.
- Anything else - The capabilities - as unicode strings.

Usage:

```
>>> capabilities = get_capabilities(<aNode>)
```

getdevinfo.linux.**get_capacity**(*node*)

Private, implementation detail.

This function gets the capacity from the structure generated by parsing lshw's XML output. Also rounds it to a human- readable form, and returns both sizes.

Args: node: Represents a device/partition.

Returns: tuple (string, string). The sizes (bytes, human-readable):

- ("Unknown", "Unknown") - Couldn't find them.
- Anything else - The sizes.

Usage:

```
>>> raw_size, human_size = get_capacity(<aNode>)
```

getdevinfo.linux.**get_device_info**(*node*)
Private, implementation detail.

This function gathers and assembles information for devices (whole disks). It employs some simple logic and the other functions defined in this module to do its work.

Args:

node: A "node" representing a device, generated from lshw's XML output.

Returns: string. The name of the device.

Usage:

```
>>> host_disk = get_device_info(<aNode>)
```

getdevinfo.linux.**get_file_system**(*node*)
Private, implementation detail.

This function gets the file system from the structure generated by parsing lshw's XML output.

Args: node: Represents a device/partition.

Returns: string. The file system:

- "Unknown" - Couldn't find it.
- Anything else - The file system.

Usage:

```
>>> file_system = get_file_system(<aNode>)
```

getdevinfo.linux.**get_id**(*disk*)
Private, implementation detail.

This function gets the ID of a given partition or device.

Args: disk (str): The name of a partition/device.

Returns: string. The ID:

- "Unknown" - Couldn't find it.
- Anything else - The ID.

Usage:

```
>>> disk_id = get_id(<aDiskName>)
```

getdevinfo.linux.**get_info**()
This function is the Linux-specific way of getting disk information. It makes use of the lshw, blkid, and lvdisplay commands to gather information.

It uses the other functions in this module to acheive its work, and it **doesn't** return the disk infomation. Instead, it is left as a global attribute in this module (DISKINFO).

Raises: Nothing, hopefully, but errors have a small chance of propagation up to here here. Wrap it in a try:, except: block if you are worried.

Usage:

```
>>> get_info()
```

getdevinfo.linux.**get_lv_aliases**(*line*)
Private, implementation detail.

Note: "name" here means path eg /dev/mapper/fedora/root.

This function gets the names of a logical volume. There may be one or more aliases as well as a "default" name. Find and return all of them.

Args: line (int): The line number where the LV name can be found.

Returns: tuple (string, list). The aliases (default_name, all aliases).

Usage:

```
>>> default_name, alias_list = get_lv_aliases(<anLVName>)
```

getdevinfo.linux.**get_lv_and_vg_name**(*volume*)
Private, implementation detail.

Note: "name" here means the names of the logical volume and the volume group by themselves. eg volume "root", in volume group "fedora."

This function gets the name of the logical volume (LV), and the name of the volume group (VG) it belongs to.

Args: volume (str): The path for a logical volume.

Returns: tuple (string, string). The VG, and LV name (vg_name, lv_name):

- ("Unknown", "Unknown") - Couldn't find them.
- Anything else - The VG and LV names.

Usage:

```
>>> vg_name, lv_name = get_lv_and_vg_name(<anLVPath>)
```

getdevinfo.linux.**get_lv_file_system**(*disk*)
Private, implementation detail.

This function gets the file system of a logical volume.

Args: disk (str): The name of a logical volume.

Returns: string. The file system.

Usage:

```
>>> file_system = get_lv_file_system(<anLVName>)
```

getdevinfo.linux.**get_partition_info**(*subnode*, *host_disk*)
Private, implementation detail.

This function gathers and assembles information for partitions. It employs some simple logic and the other functions defined in this module to do its work.

Args:

subnode: A "node" representing a partition, generated from lshw's XML output.

host_disk (str): The "parent" or "host" device. eg: for /dev/sda1, the host disk would be /dev/sda. Used to organise everything nicely in the disk info dictionary.

Returns: string. The name of the partition.

Usage:

```
>>> volume = get_device_info(<aNode>)
```

getdevinfo.linux.**get_partitioning**(*disk*)

Private, implementation detail.

This function gets the partition scheme from the structure generated by parsing lshw's XML output.

Args:

disk (str): The name of a device/partition in the disk info dictionary.

Returns: string (str). The partition scheme:

- "Unknown" - Couldn't find it.
- **"mbr" - Old-style MBR partitioning** for BIOS systems.
- "gpt" - New-style GPT partitioning.

Usage:

```
>>> partitioning = get_partitioning(<aDiskName>)
```

getdevinfo.linux.**get_product**(*node*)

Private, implementation detail.

This function gets the product from the structure generated by parsing lshw's XML output.

Args: node: Represents a device/partition.

Returns: string. The product:

- "Unknown" - Couldn't find it.
- Anything else - The product.

Usage:

```
>>> product = get_product(<aNode>)
```

getdevinfo.linux.**get_uuid**(*disk*)

Private, implementation detail.

This function gets the UUID of a given partition.

Args: disk (str): The name of a **partition**.

Returns: string. The UUID:

- "Unknown" - Couldn't find it.
- Anything else - The UUID.

Usage:

```
>>> uuid = get_uuid(<aPartitionName>)
```

getdevinfo.linux.**get_vendor**(*node*)
Private, implementation detail.

This function gets the vendor from the structure generated by parsing lshw's XML output.

Args: node: Represents a device/partition.

Returns: string. The vendor:

- "Unknown" - Couldn't find it.
- Anything else - The vendor.

Usage:

```
>>> vendor = get_vendor(<aNode>)
```

getdevinfo.linux.**parse_lsblk_output**()
Private, implementation detail.

This function is used to get NVME disk information from the output of lsblk.

Note: This will only remain here until lshw adds support for NVME disk detection - this is a temporary fix.

Usage:

```
>>> parse_lsblk_output()
```

getdevinfo.linux.**parse_lvm_output**(*testing=False*)
Private, implementation detail.

This function is used to get LVM partition information from the output of lvdisplay –maps.

Kwargs: testing (bool): Used during unit tests. Default = False.

Usage:

```
>>> parse_lvm_output()
```

OR:

```
>>> parse_lvm_output(testing=<aBool>)
```

2.7 Documentation for the Cygwin (Windows) module

This is the part of the package that contains the tools and information getters for Windows/Cygwin. This would normally be called from the getdevinfo module, but you can call it directly if you like.

Note: You can import this submodule directly, but it might result in strange behaviour, or not work on your platform if you import the wrong one. That is not how the package is intended to be used.

Warning: Feel free to experiment, but be aware that you may be able to cause crashes, exceptions, and generally weird situations by calling these methods directly if you get it wrong. A good place to look if you're interested in this is the unit tests (in tests/).

Warning: This module won't work properly unless it is executed as root.

getdevinfo.cygwin.**compute_block_size**(*stdout*)
Private, implementation detail.

Used to process and tidy up the block size output from smartctl.

Args: stdout (str): The block size.

Returns: int/None: The block size:

- None - Failed!
- int - The block size.

Usage:

```
>>> compute_block_size(<stdoutFromSmartctl>)
```

getdevinfo.cygwin.**get_block_size**(*disk*)
Public

This function uses the smartctl command to get the block size of the given device.

Args:

disk (str): The partition/device/logical volume that we want the block size for.

Returns: int/None. The block size.

- None - Failed!
- int - The block size.

Usage:

```
>>> block_size = get_block_size(<aDeviceName>)
```

getdevinfo.cygwin.**get_boot_record**(*disk*)
Private, implementation detail.

This function gets the MBR/PBR of a given disk.

Args: disk (str): The name of a partition/device.

Returns: tuple (string, string). The boot record (raw, any readable strings):

- ("Unknown", "Unknown") - Couldn't read it.
- Anything else - The PBR/MBR and any readable strings therein.

Usage:

```
>>> boot_record, boot_record_strings = get_boot_record(<aDiskName>)
```

getdevinfo.cygwin.**get_capabilities**(*disk*)
Private, implementation detail.

This function gets the capabilities from the structure generated by parsing smartctl's output.

Warning: Not yet implemented on Cygwin, returns empty list.

Args: data (dict): Parsed JSON from smartctl.

Returns: list. The capabilities:

- [] - Couldn't find them.
- Anything else - The capabilities - as unicode strings.

Usage:

```
>>> capabilities = get_capabilities(<aNode>)
```

getdevinfo.cygwin.**get_capacity**(*data*)

Private, implementation detail.

This function gets the vendor from the structure generated by parsing smartctl's output. Also rounds it to a human-readable form, and returns both pieces of data.

Args: data: Parsed JSON from smartctl.

Returns: tuple (string, string). The sizes (bytes, human-readable):

- ("Unknown", "Unknown") - Couldn't find them.
- Anything else - The sizes.

Usage:

```
>>> raw_size, human_size = get_capacity(<smartctl-data>)
```

getdevinfo.cygwin.**get_description**(*data*, *disk*)

Private, implementation detail.

This function creates a description from the structure generated by parsing smartctl's output.

Args: data: Parsed JSON from smartctl. disk (str): Name of a device/partition.

Returns:

string. The description: This may contain various bits of info, or not, depending on what macOS knows about the disk.

Usage:

```
>>> description = get_description(<smartctl-data>, <aDisk>)
```

getdevinfo.cygwin.**get_device_info**(*host_disk*)

Private, implementation detail.

This function gathers and assembles information for devices (whole disks). It employs some simple logic and the other functions defined in this module to do its work.

Note: Functionality not yet complete.

Args: host_disk: The name of the device.

Returns: string. The name of the device.

Usage:

```
>>> host_disk = get_device_info(<aNode>)
```

getdevinfo.cygwin.**get_file_system**(*output*)

Private, implementation detail.

This function gets the file system from blkid's output.

Args: output (list): Output from blkid.

Returns: string. The file system:

- "Unknown" - Couldn't find it.
- Anything else - The file system.

Usage:

```
>>> file_system = get_file_system(<blkid-output>)
```

getdevinfo.cygwin.**get_id**(*disk*)

Private, implementation detail.

This function gets the ID of a given partition or device.

Warning: Not yet implemented on Cygwin.

Args: disk (str): The name of a partition/device.

Returns: string. The ID:

- "Unknown" - Couldn't find it.
- Anything else - The ID.

Usage:

```
>>> disk_id = get_id(<aDiskName>)
```

getdevinfo.cygwin.**get_info**()

This function is the Cygwin-specific way of getting disk information. It makes use of the smartctl and blkid commands to gather information.

It uses the other functions in this module to acheive its work, and it **doesn't** return the disk infomation. Instead, it is left as a global attribute in this module (DISKINFO).

Raises: Nothing, hopefully, but errors have a small chance of propagation up to here here. Wrap it in a try:, except: block if you are worried.

Usage:

```
>>> get_info()
```

getdevinfo.cygwin.**get_partition_info**(*subnode*, *host_disk*)

Private, implementation detail.

This function gathers and assembles information for partitions. It employs some simple logic and the other functions defined in this module to do its work.

> **Warning:** Not yet implemented on Cygwin - returns None.

Args:

subnode: A "node" representing a partition, generated from lshw's XML output.

host_disk (str): The "parent" or "host" device. eg: for /dev/sda1, the host disk would be /dev/sda. Used to organise everything nicely in the disk info dictionary.

Returns: string. The name of the partition.

Usage:

```
>>> volume = get_device_info(<aNode>)
```

getdevinfo.cygwin.**get_partitioning**(*output*)

Private, implementation detail.

This function gets the partition scheme from blkid's output.

Args: output (list): Output from blkid.

Returns: string (str). The partition scheme:

- "Unknown" - Couldn't find it.
- **"mbr" - Old-style MBR partitioning** for BIOS systems.
- "gpt" - New-style GPT partitioning.

Usage:

```
>>> partitioning = get_partitioning(<blkid-output>)
```

getdevinfo.cygwin.**get_product**(*data*)

Private, implementation detail.

This function gets the product from the structure generated by parsing smartctl's output.

Args: data: Parsed JSON from smartctl.

Returns: string. The product:

- "Unknown" - Couldn't find it.
- Anything else - The product.

Usage:

```
>>> product = get_product(<smartctl-data>)
```

getdevinfo.cygwin.**get_uuid**(*output*)

Private, implementation detail.

This function gets the UUID from blkid's output.

Args: output (list): Output from blkid.

Returns: string. The UUID:

- "Unknown" - Couldn't find it.
- Anything else - The UUID.

Usage:

```
>>> uuid = get_uuid(<blkid-output>)
```

getdevinfo.cygwin.**get_vendor**(*data*)
Private, implementation detail.

This function gets the vendor from the structure generated by parsing smartctl's output.

Args: data: Parsed JSON from smartctl.

Returns: string. The vendor:

- "Unknown" - Couldn't find it.
- Anything else - The vendor.

Usage:

```
>>> vendor = get_vendor(<smartctl-data>)
```

2.8 Documentation for the macos module

This is the part of the package that contains the tools and information getters for macOS. This would normally be called from the getdevinfo module, but you can call it directly if you like.

Note: You can import this submodule directly, but it might result in strange behaviour, or not work on your platform if you import the wrong one. That is not how the package is intended to be used, except if you want to use the get_block_size() function to get a block size, as documented below.

Warning: Feel free to experiment, but be aware that you may be able to cause crashes, exceptions, and generally weird situations by calling these methods directly if you get it wrong. A good place to look if you're interested in this is the unit tests (in tests/).

Warning: This module won't work properly unless it is executed as root.

getdevinfo.macos.**compute_block_size**(*disk*, *stdout*)
Private, implementation detail.

Used to process and tidy up the block size output from diskutil info.

Args: stdout (str): diskutil info's output.

Returns: int/None: The block size:

- None - Failed!
- int - The block size.

Usage:

```
>>> compute_block_size(<stdoutFromDiskutil>)
```

getdevinfo.macos.**get_block_size**(*disk*)

Public

This function uses the diskutil info command to get the block size of the given device.

Args:

disk (str): The partition/device that we want the block size for.

Returns: int/None. The block size.

- None - Failed!
- int - The block size.

Usage:

```
>>> block_size = get_block_size(<aDeviceName>)
```

getdevinfo.macos.**get_boot_record**(*disk*)

Not yet implemented, returns ("Unknown", "Unknown").

getdevinfo.macos.**get_capabilities**(*disk*)

Not yet implemented, returns "Unknown"

getdevinfo.macos.**get_capacity**()

Private, implementation detail.

This function gets the capacity of the disk currently referenced in the diskutil info output we're storing. You can't really use this standalone. Also rounds it to a human-readable form, and returns both sizes.

Returns: tuple (string, string). The sizes (bytes, human-readable):

- ("Unknown", "Unknown") - Couldn't find them.
- Anything else - The sizes.

Usage:

```
>>> raw_size, human_size = get_capacity()
```

getdevinfo.macos.**get_description**(*disk*)

Private, implementation detail.

This function generates a human-readable description of the given disk.

Args: disk (str): Name of a device/partition.

Returns:

string. The description: This may contain various bits of info, or not, depending on what macOS knows about the disk.

Usage:

```
>>> description = get_description(<aDisk>)
```

getdevinfo.macos.**get_device_info**(*disk*)

Private, implementation detail.

This function gathers and assembles information for devices (whole disks). It employs some simple logic and the other functions defined in this module to do its work.

Args: disk (str): The name of a device, without the leading /dev. eg: disk1

Returns: string. The name of the device.

Usage:

```
>>> host_disk = get_device_info(<aNode>)
```

getdevinfo.macos.**get_file_system**(*disk*)
Not yet implemented, returns "Unknown".

getdevinfo.macos.**get_id**(*disk*)
Not yet implemented, returns "Unknown".

getdevinfo.macos.**get_info**()
This function is the macOS-specific way of getting disk information. It makes use of the diskutil list, and diskutil info commands to gather information.

It uses the other functions in this module to acheive its work, and it **doesn't** return the disk infomation. Instead, it is left as a global attribute in this module (DISKINFO).

Raises: Nothing, hopefully, but errors have a small chance of propagation up to here here. Wrap it in a try:, except: block if you are worried.

Usage:

```
>>> get_info()
```

getdevinfo.macos.**get_partition_info**(*disk*, *host_disk*)
Private, implementation detail.

This function gathers and assembles information for partitions. It employs some simple logic and the other functions defined in this module to do its work.

Args:

disk (str): The name of a partition, without the leading /dev. eg: disk1s1

host_disk (str): The "parent" or "host" device. eg: for /dev/disk1s1, the host disk would be /dev/disk1. Used to organise everything nicely in the disk info dictionary.

Returns: string. The name of the partition.

Usage:

```
>>> volume = get_device_info(<aDisk>, <aHostDisk>)
```

getdevinfo.macos.**get_partitioning**(*disk*)
Not yet implemented, returns "Unknown".

getdevinfo.macos.**get_product**(*disk*)
Private, implementation detail.

This function gets the product of the given disk.

Args: disk (str): Name of a device/partition.

Returns: string. The product:

- "Unknown" - Couldn't find it.
- Anything else - The product.

Usage:

```
>>> product = get_product(<aDisk>)
```

getdevinfo.macos.**get_uuid**(*disk*)
Not yet implemented, returns "Unknown".

getdevinfo.macos.**get_vendor**(*disk*)
Private, implementation detail.

This function gets the vendor of the given disk.

Args: disk (str): Name of a device/partition.

Returns: string. The vendor:

- "Unknown" - Couldn't find it.
- Anything else - The vendor.

Usage:

```
>>> vendor = get_vendor(<aDisk>)
```

getdevinfo.macos.**is_partition**(*disk*)
Private, implementation detail.

This function determines if a disk is a partition or not.

Args: disk (str): Name of a device/partition.

Returns: bool:

- True - Is a partition.
- False - Not a partition.

Usage:

```
>>> is_a_partition = is_partition(<aDisk>)
```

2.9 GNU Free Documentation License

GNU Free Documentation License Version 1.3, 3 November 2008

Copyright (C) 2000, 2001, 2002, 2007, 2008 Free Software Foundation, Inc. <https://fsf.org/>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "publisher" means any person or entity that distributes copies of the Document to the public.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties–for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See https://www.gnu.org/licenses/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free

Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

"Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

"CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

"Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright (c) YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with…Texts." line with this:

> with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

CHAPTER

THREE

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

g

INDEX

I

P

R