TPG-256A Pressure Daemon Documentation

Release 0.2.0

Carlos Vigo

May 26, 2020

CONTENTS

To be completed. . .

You can also check the `official documentation`

CHAPTER

ONE

README

1.1 TPG - 256A Pressure Monitor

The **tpg-256a-pressure-monitor** is a readout and logging software for the Pfeiffer TPG-256A Pressure Controller. The package includes a device driver which manages the connection and communication through RS-232 serial port. The application is run as a command-line

1.2 Dependencies

The package **tpg-256a-pressure-monitor** has the following pre-requisites:

- Python 3.6 and pip 10.0 are the minimum required versions to build and install **tpg-256a-pressure-monitor** and its dependencies. It is recommended to install and run **tpg-256a-pressure-monitor** (and any other package, for that matter) under a virtual environment.
- libpq, a C library that implements connections to the PostgreSQL backend server. It is used by the package lab-utils to connect to the PostgreSQL backend, manage database structure and save new data entries.

1.3 Getting Started

TODO: pip package available now

1.4 Usage

To use a the **tpg-256a-pressure-monitor** package, TODO

1.5 Authors

- **Carlos Vigo** - *Initial work* - GitLab

1.6 Contributing

Please read our contributing policy for details on our code of conduct, and the process for submitting pull requests to us.

1.7 Versioning

We use Git for versioning. For the versions available, see the tags on this repository.

1.8 License

This project is licensed under the GNU GPLv3 License

1.9 Built With

- PyCharm Community Edition - The IDE used
- Sphinx - Documentation

1.10 Acknowledgments

- Nobody so far

CHAPTER

TWO

API REFERENCE

Description

Readout and logging application for the Pfeiffer TPG-256A Pressure Readout.

Modules

tpg_256a_pressure_monitor	Readout and logging application for the Pfeiffer TPG-256A Pressure Readout.
Daemon	Daemon TCP server.
Monitor	Background monitoring thread based on the `threading` library.
TPG_256A	Driver for the Pfeiffer TPG 256A.

2.1 Daemon

Description

Daemon TCP server. The server will run indefinitely listening on the specified TCP (see the `Server` documentation). When a client connects and sends a message string, the message parser will call the appropriate method. The following commands are supported by the parser (options must be used with a double dash - -):

quit		Stops the daemon and cleans up database and serial port
status		TODO: Not implemented yet
tpg_256a	on/off/restart	Connects / disconnects / restarts the TPG 256A device
	test	Performs a serial port test and returns the device firmware
	config {file}	Reloads the default (or given) config file (logging is stopped)
	gauge-info	Returns gauge type, status and latest value
	single-readout	Performs a single read-out to the device (logging is stopped)
logging	start / stop	Launches or stops the separate device monitoring thread
	terminal	Prints output to the terminal with *info* level
	use-database	Enables data saving to a PostgreSQL database

Classes

Daemon	Base class of the daemon, derived from `Server`.

2.1.1 Daemon

Description

class tpg_256a_pressure_monitor.Daemon.**Daemon**(*config_file: str = None*, *pid_file_name: str = None*, *host: str = None*, *port: int = None*, *device_config_file: str = None*)

Base class of the daemon, derived from `Server`. The daemon holds pointers to the *device* driver and the *monitor* thread, and communicates with them upon message reception.

Attributes

Daemon.device	Device handler.
Daemon.monitor	Monitor thread.

Daemon.device

Daemon.**device: tpg_256a_pressure_monitor.TPG_256A.TPG_256A = None**

Device handler.

Daemon.monitor

Daemon.**monitor: tpg_256a_pressure_monitor.Monitor.Monitor = None**

Monitor thread.

Methods

Daemon.__init__	Initializes the *Daemon* object.
Daemon.logging	Manages the *logging thread*.
Daemon.quit	Stops the daemon, called with message 'quit'.
Daemon.status	TODO
Daemon.tpg_256a	Modifies or checks the status of the TPG 256A *device*.
Daemon.update_parser	Sets up the message `parser`.

Daemon.__init__

Daemon.**__init__**(*config_file: str = None*, *pid_file_name: str = None*, *host: str = None*, *port: int = None*, *device_config_file: str = None*)

Initializes the *Daemon* object. The *device* constructor is called: serial connection is established and hardware information is retrieved from the controller.

Parameters

- **config_file** (*str, optional*) – See parent class Server.
- **pid_file_name** (*str, optional*) – See parent class Server.
- **host** (*int, optional*) – See parent class Server.
- **port** (*int, optional*) – See parent class Server.
- **device_config_file** (*str, optional*) – Configuration file for the TPG-256A *device*.

Raises

- **configparser.Error** – Configuration file error
- **LockError** – The PID file could not be locked (see parent class Server).
- **OSError** – Socket errors (see parent class Server).
- **SerialException** – The connection to the *device* has failed
- **IOError** – Communication error, probably message misspelt.
- ***StateError*** – *device* was in the wrong state, e.g. already ON.

Daemon.logging

Daemon.**logging**()

Manages the *logging thread*. Provides functionality to:

- Start and stop the thread.
- Enable or disable database usage.
- Enable or disable terminal output.

Daemon.quit

Daemon.**quit**()

Stops the daemon, called with message 'quit'. The method overrides the original quit() to do proper clean-up of the monitoring *thread* and the *device* handler.

Daemon.status

Daemon.**status**()
TODO

Daemon.tpg_256a

Daemon.**tpg_256a**()
Modifies or checks the status of the TPG 256A *device*. Provides functionality to:

- Connect and disconnect the controller.
- Retrieve hardware information.
- Reload device configuration.
- Perform a single read-out of the gauges

Daemon.update_parser

Daemon.**update_parser**()
Sets up the message parser.

2.2 Monitor

Description

Background monitoring thread based on the threading library. A *Monitor* object starts a background thread which reads out the TPG-256A *device* every second. The data can then be printed to the terminal and/or saved to a PostgreSQL database using the lab_utils.database library. The monitoring thread is intended to be self-sustainable and will try to deal with unexpected errors (usually issues with communication to the device), recover, log them and keep running.

Classes

Monitor	Manages a background thread which logs data from the TPG-256A *device*.

2.2.1 Monitor

Description

class tpg_256a_pressure_monitor.Monitor.**Monitor**(*device:* tpg_256a_pressure_monitor.TPG_256A.TPG_256A, *name: str = 'Monitor Thread', database_flag: bool = False, database_config_file: str = None, terminal_flag: bool = False, table_name: str = 'pressure'*)
Manages a background thread which logs data from the TPG-256A *device*.

Attributes

Monitor.column_list	List of data labels to save.
Monitor.database_flag	Database usage flag.
Monitor.db	Database object.
Monitor.device	*TPG-256 A* handler.
Monitor.run_flag	Flag to signal the thread to stop.
Monitor.table_name	Name of the PostgreSQL table where data will be saved.
Monitor.terminal_flag	Terminal output flag.

Monitor.column_list

Monitor.**column_list: List[str] = None**
List of data labels to save.

Monitor.database_flag

Monitor.**database_flag: threading.Event = False**
Database usage flag.

Monitor.db

Monitor.**db: lab_utils.database.Database = None**
Database object.

Monitor.device

Monitor.**device: tpg_256a_pressure_monitor.TPG_256A.TPG_256A = None**
TPG-256 A handler.

Monitor.run_flag

Monitor.**run_flag: threading.Event = None**
Flag to signal the thread to stop.

Monitor.table_name

Monitor.**table_name: str = 'pressure'**
Name of the PostgreSQL table where data will be saved.

Monitor.terminal_flag

Monitor.**terminal_flag: threading.Event = False**
Terminal output *flag*.

Methods

Monitor.__init__	Initializes the *Monitor* object.
Monitor.prepare_database	Ensures the *database* is ready to accept data from the TPG-256A *device*.
Monitor.print_string	Prints the retrieved data to the terminal.
Monitor.run	Monitoring method start upon object creation.
Monitor.save_to_database	Saves the latest pressure data to the PostgreSQL *database*.
Monitor.stop	Clears the *run_flag* to signal the background thread to stop.

Monitor.__init__

Monitor.**__init__**(*device:* tpg_256a_pressure_monitor.TPG_256A.TPG_256A, *name: str = 'Monitor Thread'*, *database_flag: bool = False*, *database_config_file: str = None*, *terminal_flag: bool = False*, *table_name: str = 'pressure'*)

Initializes the *Monitor* object. The constructor checks that the given *device* is initialized. If *database_flag* is set to *True*, the *prepare_database()* method is called, which initializes the *database* object and sets up the connection. A table named *table_name* is created, as well as the necessary *columns* to store the pressure data.

Finally, the method *run()* starts and detaches a background thread which will run indefinitely, reading the TPG-256A *device*. The data is saved to the database if *database_flag* is set to *True*, and it is printed to the terminal if *terminal_flag* is set to *True*.

Parameters

- **device** (*TPG_256A*) – Device handle, must be already initialized and connected.
- **name** (*str, optional*) – Thread name for logging purposes, default is 'Monitor Thread'
- **database_flag** (*bool, optional*) – Save data to a PostgreSQL database, default is 'False'
- **terminal_flag** (*bool, optional*) – Print data to the logging terminal sink with 'info' level, default is 'False'
- **table_name** (*str, optional*) – Name of the PostgreSQL table where the data is saved, default is 'pressure'.

Raises

- ***StateError*** – The supplied *device* was not properly initialized.
- **configparser.Error** – Database configuration file error.
- **psycopg2.DatabaseError** – Database error (connection, access...)

Monitor.prepare_database

Monitor.**prepare_database**(*database_config_file: str = None*)

Ensures the *database* is ready to accept data from the TPG-256A *device*. Initializes the *database* object and sets up the connection. If the table *table_name* does not exist, it is created, as well as the necessary *columns* to store the pressure data. The labels of the columns are retrieved from device's *channel_info*.

Parameters **database_config_file** (*str, optional*) – The configuration file of the database

Raises

- **configparser.Error** – Error reading configuration file.
- **psycopg2.Error** – Base exception for all kinds of database errors.

Monitor.print_string

Monitor.**print_string**()

Prints the retrieved data to the terminal. The log level will be *INFO* if *terminal_flag* is set, and *DEBUG* otherwise.

Monitor.run

Monitor.**run**() → None

Monitoring method start upon object creation. The TPG-256A *device* is read every second in an endless loop. The pressure data may be saved to a PostgreSQL *database* and/or printed to the terminal, if the respective *terminal_flag* and *database_flag* flags were set.

In case of unexpected error (which happens often with the RS-232 communication protocol), the method will try to recover, log any information and continue operations.

To stop logging and break the loop, the *stop()* method should be used to set the *run_flag* flag.

Monitor.save_to_database

Monitor.**save_to_database**()

Saves the latest pressure data to the PostgreSQL *database*. If *database_flag* is not set, the method does nothing.

Raises **psycopg2.Error** – Base exception for all kinds of database errors.

Monitor.stop

Monitor.**stop**() → bool

Clears the *run_flag* to signal the background thread to stop. The thread status is then checked every 0.1 second (up to 5 seconds). Returns *True* if the thread stopped, *False* otherwise.

Returns *True* if the thread is not running within 5 seconds, *False* otherwise.

Return type bool

2.3 TPG_256A

Description

Driver for the Pfeiffer TPG 256A. The device is a six-channel pressure readout and monitor controller.

The *TPG_256A* main class manages the interface to the device and implements some of the available operations through RS-232 communication. The driver implements an auxiliary *Channel* class to hold information about the available gauges. A custom exception *StateError* is used for internal error management.

The driver has been adapted to Python3 from the PyExpLabSys library. More information is also available in the device documentation

Classes

Channel	Simple container to hold channel information.
TPG_256A	Driver implementation for the Pfeiffer TPG-256A.

2.3.1 Channel

Description

class tpg_256a_pressure_monitor.TPG_256A.**Channel**(*gauge_number: int = None*)

Simple container to hold channel information. The setup is read from the *configuration file* (e.g. *label*) or is retrieved from the controller directly (e.g. *gauge_id*).

The following gauge types are supported:

ID	Description
TPR	Pirani Gauge or Pirani Capacitive gauge
IKR9	Cold Cathode Gauge 10E-9
IKR11	Cold Cathode Gauge 10E-11
PKR	FullRange CC Gauge
PBR	FullRange BA Gauge
IMR	Pirani / High Pressure Gauge
CMR	Linear gauge
noSEn	No Sensor
no Sensor	No Sensor
noid	No identifier

The measurement status can take the following values:

Code	Status
0	Measurement data okay
1	Underrange
2	Overrange
3	Sensor error
4	Sensor off (IKR, PKR, IMR, PBR)
5	No sensor (output: 5,2.0000E-2 [mbar])
6	Identification error

Attributes

Channel.active	The gauge should be ON.
Channel.connected	The gauge was detected by the controller.
Channel.data	Latest pressure readout value.
Channel.gauge_id	Gauge type, retrieved from the controller.
Channel.gauge_number	The channel ID number.
Channel.label	Label of the gauge, to be used when logging to a database.
Channel.logging	Data from the gauge should be recorded.
Channel.status_code	Measurement status code.
Channel.status_str	Measurement status description.

Channel.active

Channel.**active: bool = False**
The gauge should be ON.

Channel.connected

Channel.**connected: bool = False**
The gauge was detected by the controller.

Channel.data

Channel.**data: float = None**
Latest pressure readout value.

Channel.gauge_id

Channel.**gauge_id: str = None**
Gauge type, retrieved from the controller.

Channel.gauge_number

Channel.**gauge_number: int = None**
The channel ID number.

Channel.label

Channel.**label: str = ''**
Label of the gauge, to be used when logging to a database.

Channel.logging

Channel.**logging: bool = False**
Data from the gauge should be recorded.

Channel.status_code

Channel.**status_code: int = None**
Measurement status code.

Channel.status_str

Channel.**status_str: str = None**
Measurement status description.

Methods

Channel.__init__	Initialize self.

Channel.__init__

Channel.**__init__**(*gauge_number: int = None*)
Initialize self. See help(type(self)) for accurate signature.

2.3.2 TPG_256A

Description

class tpg_256a_pressure_monitor.TPG_256A.**TPG_256A**(*serial_port: str = None*, *baud_rate: int = None*, *connect: bool = False*, *timeout: float = None*, *config_file: str = None*)
Driver implementation for the Pfeiffer TPG-256A. The device is a six-channel pressure readout and monitor controller. The driver has been adapted to Python3 from the PyExpLabSys library, and implements the following commands (see the device documentation for more information):

Mnemonic	Description
PNR	Program number (firmware version)
PR[1 ... 6]	Pressure measurement (measurement data) gauge [1 ... 6]
TID	Transmitter identification (gauge identification)
SEN,0,0,0,0,0,0	Gauge status

Attributes

TPG_256A.ACK	Acknowledge, chr(6), \x06
TPG_256A.CR	Carriage return, chr(13), \r
TPG_256A.ENQ	Enquiry, chr(5), \x05
TPG_256A.ETX	End text (Ctrl-c), chr(3), \x03
TPG_256A.LF	Line feed, chr(10), \n
TPG_256A.NAK	Negative acknowledge, chr(21), \x15
TPG_256A.baud_rate	Baud rate for serial communication.
TPG_256A.channel_info	Channel information, loaded from the configuration file.
TPG_256A.config_file	Device configuration file
TPG_256A.connected	Status flag.
TPG_256A.serial	Serial port handler.
TPG_256A.serial_port	Physical address of the device file.
TPG_256A.timeout	Time-out for serial connection error.

TPG_256A.ACK

TPG_256A.**ACK = '\x06'**
Acknowledge, chr(6), \x06

TPG_256A.CR

TPG_256A.**CR = '\r'**
Carriage return, chr(13), \r

TPG_256A.ENQ

TPG_256A.**ENQ = '\x05'**
Enquiry, chr(5), \x05

TPG_256A.ETX

TPG_256A.**ETX = '\x03'**
End text (Ctrl-c), chr(3), \x03

TPG_256A.LF

TPG_256A.**LF = '\n'**
Line feed, chr(10), \n

TPG_256A.NAK

TPG_256A.**NAK = '\x15'**
Negative acknowledge, chr(21), \x15

TPG_256A.baud_rate

TPG_256A.**baud_rate: int = 9600**
Baud rate for serial communication.

TPG_256A.channel_info

TPG_256A.**channel_info: List[tpg_256a_pressure_monitor.TPG_256A.Channel] = []**
Channel information, loaded from the configuration file.

TPG_256A.config_file

TPG_256A.**config_file: str = 'conf/tpg_256a.ini'**
Device configuration file

TPG_256A.connected

TPG_256A.**connected: bool = False**
Status flag.

TPG_256A.serial

TPG_256A.**serial: serial.serialposix.Serial = None**
Serial port handler.

TPG_256A.serial_port

TPG_256A.**serial_port: str = '/dev/PfeifferTPG256A'**
Physical address of the device file.

TPG_256A.timeout

TPG_256A.**timeout: float = 1.0**
Time-out for serial connection error.

Methods

TPG_256A.__init__	Initializes the *TPG_256A* object.
TPG_256A.config	Loads the TPG-256A configuration from a file.
TPG_256A.connect	Connects to the TPG-256A Controller.
TPG_256A.disconnect	Closes the connection to the TPG-256A Controller.
TPG_256A.gauge_identification	Reads the gauges identification.
TPG_256A.gauge_status	Reads the gauges status.
TPG_256A.pressure_gauge	Reads the pressure measured by gauge number *gauge*.
TPG_256A.pressure_gauges	Reads the pressure measured by all active gauges.
TPG_256A.program_number	Returns the firmware version.

TPG_256A.__init__

TPG_256A.**__init__**(*serial_port: str = None*, *baud_rate: int = None*, *connect: bool = False*, *timeout: float = None*, *config_file: str = None*)

Initializes the *TPG_256A* object. It calls the *config()* method to set up the device if a *config_file* is given. If the *connect* flag is set to *True*, attempts the connection to the device.

Parameters

- **serial_port** (*str, optional*) – Physical address of the device file, default is 'None'
- **timeout** (*float, optional*) – Serial communication time out, default is 'None'
- **baud_rate** (*int, optional*) – Baud rate for serial communication, default is 'None'
- **connect** (*bool, optional*) – If set, attempt connection to the device, default is *False*
- **config_file** (*str, optional*) – Configuration file, default is 'None'.

Raises

- **configparser.Error** – Configuration file error
- **SerialException** – The connection to the device has failed
- **IOError** – Communication error, probably message misspelt.
- ***StateError*** – Device was in the wrong state.

TPG_256A.config

TPG_256A.**config**(*new_config_file: str = None*)

Loads the TPG-256A configuration from a file. If *new_config_file* is not given, the latest *config_file* is re-loaded; if it is given and the file is successfully parsed, *config_file* is updated to the new value.

Parameters **new_config_file** (*str, optional*) – New configuration file to be loaded.

Raises **configparser.Error** – Configuration file error

TPG_256A.connect

TPG_256A.**connect**()
Connects to the TPG-256A Controller. The methods *gauge_identification()* and *gauge_status()* are called to retrieve hardware information from the device.

Raises

- **SerialException** – The connection to the device has failed.
- **IOError** – Communication error, probably message misspelt.
- ***StateError*** – Device was in the wrong state.

TPG_256A.disconnect

TPG_256A.**disconnect**()
Closes the connection to the TPG-256A Controller.

Raises

- **serial.SerialException** – The connection to the device has failed.
- **IOError** – Communication error, probably message misspelt.
- ***StateError*** – Device was in the wrong state.

TPG_256A.gauge_identification

TPG_256A.**gauge_identification**()
Reads the gauges identification. Saves the information in *channel_info*. Checks that gauges marked as *active* in *channel_info* are available; sets them to inactive otherwise and disables logging.

Raises

- ***StateError*** – Device was in the wrong state.
- **serial.SerialException** – The connection to the device has failed.
- **IOError** – Communication error, probably message misspelt.

TPG_256A.gauge_status

TPG_256A.**gauge_status**()
Reads the gauges status. Checks that gauges marked as *active* in *channel_info* are available; sets them to inactive otherwise.

Raises

- ***StateError*** – Device was in the wrong state.
- **serial.SerialException** – The connection to the device has failed.
- **IOError** – Communication error, probably message misspelt.

TPG_256A.pressure_gauge

TPG_256A.**pressure_gauge**(*gauge_nr*) → Tuple[float, int]
Reads the pressure measured by gauge number *gauge*.

Parameters **gauge_nr** (*int*) – The gauge number, 1 to 6

Returns (value, status code)

Return type [float, int]

Raises

- ***StateError*** – Device was in the wrong state.
- **serial.SerialException** – The connection to the device has failed.
- **IOError** – Communication error, probably message misspelt.
- **ValueError** – Invalid gauge_nr, must be between 1 and 6.

TPG_256A.pressure_gauges

TPG_256A.**pressure_gauges**()
Reads the pressure measured by all active gauges. Saves the data into the *channel_info* list.

Raises

- ***StateError*** – Device was in the wrong state.
- **serial.SerialException** – The connection to the device has failed.
- **IOError** – Communication error, probably message misspelt.

TPG_256A.program_number

TPG_256A.**program_number**() → str
Returns the firmware version.

Returns The firmware version.

Return type str

Raises

- **serial.SerialException** – The connection to the device has failed.
- **IOError** – Communication error, probably message misspelt.

Exceptions

StateError	Mock-up exception to deal with unexpected device status.

2.3.3 StateError

exception tpg_256a_pressure_monitor.TPG_256A.**StateError**

Mock-up exception to deal with unexpected device status. It is used to signal for instance that the device should be connected but it is not at a certain execution point.

CHAPTER

THREE

CHANGELOG

All notable changes to this project will be documented in this file.

The format is based on Keep a Changelog, and this project adheres to Semantic Versioning.

3.1 0.2.0 - 2020-05-26

- Full implementation of the logging module

3.2 0.1.0 - 2020-04-21

- First release of the **tpg-256a-pressure-monitor** package
- Installation instructions and setup

CHAPTER

FOUR

CONTRIBUTING

When contributing to this repository, please first discuss the change you wish to make via issue, email, or any other method with the owners of this repository before making a change.

Please note we have a code of conduct, please follow it in all your interactions with the project.

4.1 Pull Request Process

1. Ensure any install or build dependencies are removed before the end of the layer when doing a build.
2. Update the README.md with details of changes to the interface, this includes new environment variables, exposed ports, useful file locations and container parameters.
3. Increase the version numbers in any examples files and the README.md to the new version that this Pull Request would represent. The versioning scheme we use is SemVer.
4. You may merge the Pull Request in once you have the sign-off of two other developers, or if you do not have permission to do that, you may request the second reviewer to merge it for you.

4.2 Code of Conduct

4.2.1 Our Pledge

In the interest of fostering an open and welcoming environment, we as contributors and maintainers pledge to making participation in our project and our community a harassment-free experience for everyone, regardless of age, body size, disability, ethnicity, gender identity and expression, level of experience, nationality, personal appearance, race, religion, or sexual identity and orientation.

4.2.2 Our Standards

Examples of behavior that contributes to creating a positive environment include:

- Using welcoming and inclusive language
- Being respectful of differing viewpoints and experiences
- Gracefully accepting constructive criticism
- Focusing on what is best for the community
- Showing empathy towards other community members

Examples of unacceptable behavior by participants include:

- The use of sexualized language or imagery and unwelcome sexual attention or advances
- Trolling, insulting/derogatory comments, and personal or political attacks
- Public or private harassment
- Publishing others' private information, such as a physical or electronic address, without explicit permission
- Other conduct which could reasonably be considered inappropriate in a professional setting

4.2.3 Our Responsibilities

Project maintainers are responsible for clarifying the standards of acceptable behavior and are expected to take appropriate and fair corrective action in response to any instances of unacceptable behavior.

Project maintainers have the right and responsibility to remove, edit, or reject comments, commits, code, wiki edits, issues, and other contributions that are not aligned to this Code of Conduct, or to ban temporarily or permanently any contributor for other behaviors that they deem inappropriate, threatening, offensive, or harmful.

4.2.4 Scope

This Code of Conduct applies both within project spaces and in public spaces when an individual is representing the project or its community. Examples of representing a project or community include using an official project e-mail address, posting via an official social media account, or acting as an appointed representative at an online or offline event. Representation of a project may be further defined and clarified by project maintainers.

4.2.5 Enforcement

Instances of abusive, harassing, or otherwise unacceptable behavior may be reported by contacting the project team at [INSERT EMAIL ADDRESS]. All complaints will be reviewed and investigated and will result in a response that is deemed necessary and appropriate to the circumstances. The project team is obligated to maintain confidentiality with regard to the reporter of an incident. Further details of specific enforcement policies may be posted separately.

Project maintainers who do not follow or enforce the Code of Conduct in good faith may face temporary or permanent repercussions as determined by other members of the project's leadership.

4.2.6 Attribution

This Code of Conduct is adapted from the Contributor Covenant, version 1.4, available at http://contributor-covenant.org/version/1/4

CHAPTER

FIVE

GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007Copyright © 2007 Free Software Foundation, Inc. <http://fsf.org/>

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

5.1 Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program–to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: **(1)** assert copyright on the software, and **(2)** offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that

patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

5.2 TERMS AND CONDITIONS

5.2.1 0. Definitions

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that **(1)** displays an appropriate copyright notice, and **(2)** tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

5.2.2 1. Source Code

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that **(a)** is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and **(b)** serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code

for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

5.2.3 2. Basic Permissions

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

5.2.4 3. Protecting Users' Legal Rights From Anti-Circumvention Law

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

5.2.5 4. Conveying Verbatim Copies

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5.2.6 5. Conveying Modified Source Versions

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- **a)** The work must carry prominent notices stating that you modified it, and giving a relevant date.
- **b)** The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".
- **c)** You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- **d)** If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

5.2.7 6. Conveying Non-Source Forms

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- **a)** Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- **b)** Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either **(1)** a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or **(2)** access to copy the Corresponding Source from a network server at no charge.
- **c)** Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- **d)** Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- **e)** Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either **(1)** a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or **(2)** anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

5.2.8 7. Additional Terms

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- **a)** Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- **b)** Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- **c)** Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- **d)** Limiting the use for publicity purposes of names of licensors or authors of the material; or

- **e)** Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- **f)** Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

5.2.9 8. Termination

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

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, you do not qualify to receive new licenses for the same material under section 10.

5.2.10 9. Acceptance Not Required for Having Copies

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

5.2.11 10. Automatic Licensing of Downstream Recipients

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

5.2.12 11. Patents

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either **(1)** cause the Corresponding Source to be so available, or **(2)** arrange to deprive yourself of the benefit of the patent license for this particular work, or **(3)** arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license **(a)** in connection with copies of the covered work conveyed by you (or copies made from those copies), or **(b)** primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

5.2.13 12. No Surrender of Others' Freedom

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

5.2.14 13. Use with the GNU Affero General Public License

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

5.2.15 14. Revised Versions of this License

The Free Software Foundation may publish revised and/or new versions of the GNU General Public 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.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

5.2.16 15. Disclaimer of Warranty

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

5.2.17 16. Limitation of Liability

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

5.2.18 17. Interpretation of Sections 15 and 16

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

5.3 How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year>  <name of author>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.
```

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program>  Copyright (C) <year>  <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type 'show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type 'show c' for details.
```

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see <http://www.gnu.org/licenses/>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <http://www.gnu.org/philosophy/why-not-lgpl.html>.

PYTHON MODULE INDEX

t

INDEX

Symbols

A

B

C

D

E

G

L

M

N

P

Q

R

S

T

U