
PyBluez

Release master

Sep 16, 2019

Contents

1 Table of contents	3
Index	19

Python extension module allowing access to system Bluetooth resources.

1.1 Installing PyBluez

PyBluez can be installed on GNU/Linux, Windows and macOS systems and is compatible with Python 2.7 and 3.

Note: Before you install **PyBluez** please install the dependencies required for your system as described in the sections below.

Installing PyBluez using pip

Open a terminal (command prompt on Windows) and enter

```
pip install pybluez
```

(there are also binaries for Windows platform on PyPI or here - [Unofficial Windows Binaries for Python Extension Packages](#))

For experimental Bluetooth Low Energy support (only for Linux platform - for additional dependencies please take look at: [ble-dependencies](#))

```
pip install pybluez\[ble\]
```

Installing PyBluez from source

Download a stable release from <https://github.com/pybluez/pybluez/releases>

or download the latest version using the links below.

master.zip

master.tar.gz

Extract the zip or tar and cd to the extracted file directory, then:

```
python setup.py install
```

for Bluetooth Low Energy support (GNU/Linux only):

```
pip install -e .\[ble\]
```

1.1.1 GNU/Linux Dependencies

- Python 2.7 or more recent version
- Python distutils (standard in most Python distros, separate package python-dev in Debian)
- BlueZ libraries and header files

1.1.2 Windows Dependencies

- Windows 7/8/8.1/10
- Python 3.5 or more recent version

PyBluez requires a C++ compiler installed on your system to build CPython modules.

For Python 3.5 or higher

- Microsoft Visual C++ 14.0 standalone: Build Tools for Visual Studio 2017 (x86, x64, ARM, ARM64)
- Microsoft Visual C++ 14.0 with Visual Studio 2017 (x86, x64, ARM, ARM64)
- Microsoft Visual C++ 14.0 standalone: Visual C++ Build Tools 2015 (x86, x64, ARM)
- Microsoft Visual C++ 14.0 with Visual Studio 2015 (x86, x64, ARM)

Note: Windows 10 users need to download and install the [Windows 10 SDK](#)

[More details here](#)

- Widcomm BTW development kit 5.0 or later (Optional)

1.1.3 macOS Dependencies

- Xcode
- PyObjc 3.1b or later (<https://pythonhosted.org/pyobjc/install.html#manual-installation>)

1.2 Contributors

Original Author - [Albert Huang](#).

Maintainers:

- [Ryan Govostes](#)
- [Piotr Karulis](#)

Other contributors:

- Travis Peters
- Michal Maruska
- Young-Bo
- Thomas Leveil
- Yurii Shevchuk
- Hugo Sales
- kedos
- Chad Spensky
- clach04
- Boris Belousov
- Ratmir Karabut
- zeripath
- Brennan Ashton
- Haim Daniel
- demanbart
- Clemens Wolff
- Arusekk
- Colin Atkinson
- Vitali Lovich
- bbregeault
- Eduardo Marossi
- Bill Crawford
- Christian Clauss

1.3 Contributing to PyBluez

This project is not currently under active development.

Contributions are strongly desired to resolve compatibility problems on newer systems, address bugs, and improve platform support for various features. Here are some guidelines to follow.

1.3.1 Compatibility Issues

Please submit compatibility issues by opening an [issue](#) explaining the problem clearly and providing information necessary to reproduce the issue, including sample code and logs.

If you have long logs, please post them on <https://gist.github.com> and link to the Gist in your issue.

1.3.2 Bugs

Please submit bug reports by opening an [issue](#) explaining the problem clearly using code examples.

1.3.3 Coding

Todo: Develop PyBluez coding standards and style guides.

1.3.4 Documentation

The documentation source lives in the `docs` folder. Contributions to the documentation are welcome but should be easy to read and understand. All source documents are in restructured text format.

Todo: Develop PyBluez documentation standards and style guides.

1.3.5 Commit messages and pull requests

Commit messages should be concise but descriptive, and in the form of an instructional patch description (e.g., “Add macOS support” not “Added macOS support”).

Commits which close, or intend to close, an issue should include the phrase “fix #234” or “close #234” where #234 is the issue number, as well a short description, for example: “Add logic to support Win10, fix #234”. Pull requests should aim to match or closely match the corresponding issue title.

1.3.6 Copyrights

By submitting to this project you agree to your code or documentation being released under the projects *license*. Copyrights on submissions are owned by their authors. Feel free to append your name to the list of contributors in `contributors.rst` found in the projects `docs` folder as part of your pull request!

1.4 PyBluez API

The Pybluez API provides a suite of classes and functions.

1.4.1 Classes

BluetoothSocket

Todo: Add documentation for the BluetoothSocket methods.

```
class bluetooth.BluetoothSocket (proto=<sphinx.ext.autodoc.importer._MockObject object>,  
                                _sock=None)  
    Bases: object
```

A Bluetooth Socket representing one endpoint of a Bluetooth connection.

Parameters `proto` (*int*) – The protocol the socket will use. The options are HCI, L2CAP, RFCOMM, or SCO. The default is RFCOMM.

Note: RFCOMM is the only protocol available for Windows and macOS systems.

accept ()

Accept a connection.

Returns A tuple containing a *BluetoothSocket* and a Bluetooth address.

Return type tuple

Raises *BluetoothError* – When an attempt to accept a connection fails.

bind (*addrport*)

Bind the socket to a local address and port.

Parameters **addrport** (*tuple*) – A tuple of the form (address str, port int)

Raises *BluetoothError* – When an attempt to bind the socket fails.

close (**args, **kwargs*)

connect (**args, **kwargs*)

connect_ex (**args, **kwargs*)

dup ()

Duplicate the socket

Returns A new *BluetoothSocket* connected to the same system resource.

Return type *BluetoothSocket*

family

fileno (**args, **kwargs*)

get_l2cap_options (*sock, mtu*)

Gets L2CAP options for the specified L2CAP socket. Options are: omtu, imtu, flush_to, mode, fcs, max_tx, txwin_size.

getpeername (**args, **kwargs*)

getsockname (**args, **kwargs*)

getsockopt (**args, **kwargs*)

gettimeout (**args, **kwargs*)

listen (**args, **kwargs*)

makefile (**args, **kwargs*)

proto

recv (**args, **kwargs*)

recvfrom (**args, **kwargs*)

send (**args, **kwargs*)

sendall (**args, **kwargs*)

sendto (**args, **kwargs*)

set_l2cap_mtu (*sock, mtu*)

Adjusts the MTU for the specified L2CAP socket. This method needs to be invoked on both sides of the connection for it to work! The default mtu that all L2CAP connections start with is 672 bytes.

mtu must be between 48 and 65535, inclusive.

set_l2cap_options (*sock, options*)

Sets L2CAP options for the specified L2CAP socket. The option list must be in the same format supplied by `get_l2cap_options()`.

setblocking (**args, **kwargs*)

setl2capsecurity (**args, **kwargs*)

setsockopt (**args, **kwargs*)

settimeout (**args, **kwargs*)

shutdown (**args, **kwargs*)

timeout

type

DeviceDiscoverer

Todo: Add documentation for the DeviceDiscover class and its methods.

class `bluetooth.DeviceDiscoverer` (*device_id=-1*)

Bases: `object`

Skeleton class for finer control of the device discovery process.

To implement asynchronous device discovery (e.g. if you want to do something *as soon as* a device is discovered), subclass `DeviceDiscoverer` and override `device_discovered ()` and `inquiry_complete ()`

cancel_inquiry ()

Call this method to cancel an inquiry in process. `inquiry_complete` will still be called.

device_discovered (*address, device_class, rssi, name*)

Called when a bluetooth device is discovered.

`address` is the bluetooth address of the device

device_class is the Class of Device, as specified in [1] passed in as a 3-byte string

`name` is the user-friendly name of the device if `lookup_names` was set when the inquiry was started. otherwise `None`

This method exists to be overridden.

[1] <https://www.bluetooth.org/foundry/assignnumb/document/baseband>

fileno ()

find_devices (*lookup_names=True, duration=8, flush_cache=True*)

find_devices (*lookup_names=True, service_name=None, duration=8, flush_cache=True*)

Call this method to initiate the device discovery process

lookup_names - set to **True** if you want to lookup the user-friendly names for each device found.

service_name - set to the name of a service you're looking for. only devices with a service of this name will be returned in `device_discovered ()` NOT YET IMPLEMENTED

ADVANCED PARAMETERS: (don't change these unless you know what you're doing)

duration - the number of 1.2 second units to spend searching for bluetooth devices. If `lookup_names` is `True`, then the inquiry process can take a lot longer.

`flush_cache` - return devices discovered in previous inquiries

`inquiry_complete()`

Called when an inquiry started by `find_devices` has completed.

`pre_inquiry()`

Called just after `find_devices` is invoked, but just before the inquiry is started.

This method exists to be overridden

`process_event()`

Waits for one event to happen, and processes it. The event will be either a device discovery, or an inquiry completion.

`process_inquiry()`

Repeatedly calls `process_event()` until the device inquiry has completed.

1.4.2 Functions

`advertise_service`

`bluetooth.advertise_service(sock, name, service_id="", service_classes=[], profiles=[], provider="", description="", protocols=[])`

Advertise a service with the local SDP server.

Parameters

- **`sock`** (`BluetoothSocket`) – The `BluetoothSocket` to use for advertising a service. The socket must be a bound, listening socket.
- **`name`** (`str`) – The name of the service and `service_id` (if specified). This should be a string of the form “XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX”, where each ‘X’ is a hexadecimal digit.
- **`service_classes`** (`list`) – a list of service classes belonging to the advertised service. Each service class is represented by a 16-bit or 128-bit UUID.

UUID Type	Format
Short 16-bit	XXXX
Full 128-bit	XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX

where each ‘X’ is a hexadecimal digit.

There are some constants for standard services, e.g. `SERIAL_PORT_CLASS` that equals to “1101”. Some class constants provided by PyBluez are:

<code>SERIAL_PORT_CLASS</code>	<code>LAN_ACCESS_CLASS</code>
<code>DIALUP_NET_CLASS</code>	<code>HEADSET_CLASS</code>
<code>CORDLESS_TELEPHONY_CLASS</code>	<code>AUDIO_SOURCE_CLASS</code>
<code>AUDIO_SINK_CLASS</code>	<code>PANU_CLASS</code>
<code>NAP_CLASS</code>	<code>GN_CLASS</code>

- **`profiles`** (`list`) – A list of service profiles that this service fulfills. Each profile is a tuple with (uuid, version). Most standard profiles use standard classes as UUIDs.

PyBluez offers a list of standard profiles, for example SERIAL_PORT_PROFILE. All standard profiles have the same name as the classes, except that _CLASS suffix is replaced by _PROFILE.

- **provider** (*str*) – A text string specifying the provider of the service
- **description** (*str*) – A text string describing the service
- **protocols** (*list*) – A list of protocols

Note: A note on working with Symbian smartphones: bt_discover in Python for Series 60 will only detect service records with service class SERIAL_PORT_CLASS and profile SERIAL_PORT_PROFILE

discover_devices

`bluetooth.discover_devices` (*duration=8, flush_cache=True, lookup_names=False, lookup_class=False, device_id=-1, iac=10390323*)

Perform a Bluetooth device discovery.

This function uses the first available Bluetooth resource to discover Bluetooth devices.

Parameters

- **lookup_names** (*bool*) – When set to True `discover_devices()` also attempts to look up the display name of each detected device. (the default is False).
- **lookup_class** (*bool*) – When set to True `discover_devices()` attempts to look up the class of each detected device. (the default is False).

Returns

Returns a list of device addresses as strings or a list of tuples. The content of the tuples depends on the values of `lookup_names` and `lookup_class` as detailed below.

lookup_class	lookup_names	Return
False	False	list of device addresses
False	True	list of (address, name) tuples
True	False	list of (address, class) tuples
True	True	list of (address, name, class) tuples

Return type `list`

find_service

`bluetooth.find_service` (*name=None, uuid=None, address=None*)

Use to find available Bluetooth services.

This function uses the service discovery protocol (SDP) to search for Bluetooth services matching the specified criteria and returns the search results.

The search criteria are defined by passing one or more parameters to the function.

If no criteria are specified then a list of all nearby services detected is returned. If more than one criteria is specified, then the search results will match all the criteria specified.

Parameters

- **name** (*str or None*) – The friendly name of a Bluetooth device.

- **uuid** (*str* or *None*) – A valid 16-bit or 128-bit UUID.

UUID Type	Format
Short 16-bit	XXXX
Full 128-bit	XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX

where each ‘X’ is a hexadecimal digit.

- **address** (*str* or *None*) – The Bluetooth address of a device or “localhost”. If “localhost” is provided the function will search for Bluetooth services on the local machine.

Returns

The search results will be a list of dictionaries. Each dictionary represents a search match having the following key/value pairs.

Key	Value
host	the bluetooth address of the device advertising the service.
name	the name of the service being advertised.
de- scrip- tion	a description of the service being advertised.
provider	the name of the person/organization providing the service.
proto- col	either ‘RFCOMM’, ‘L2CAP’, None if the protocol was not specified, or ‘UNKNOWN’ if the protocol was specified but unrecognized.
port	the L2CAP PSM number if the protocol is ‘L2CAP’, the RFCOMM channel number if the protocol is ‘RFCOMM’, or None if the protocol wasn’t specified.
service- classes	a list of service class IDs (UUID strings). Possibly empty
profiles	a list of profiles the service claims to support. a profile takes the form of (UUID, version) pairs. Possibly empty.
service- id	the Service ID of the service. None if it wasn’t set See the Bluetooth spec for the difference between Service ID and Service Class ID List

Return type *list*

lookup_name

`bluetooth.lookup_name(address, timeout=10)`

Look up the friendly name of a Bluetooth device.

This function tries to determine the friendly name (human readable) of the device with the specified Bluetooth address.

Parameters **address** (*str*) – The Bluetooth address of the device.

Returns The friendly name of the device on success, and None on failure.

Return type *str* or *None*

Raises *BluetoothError* – When the provided address is not a valid Bluetooth address.

stop_advertising

`bluetooth.stop_advertising(sock)`

Try to stop advertising a bluetooth service.

This function instructs the local SDP server to stop advertising the service associated with socket. You should typically call this right before you close socket.

Parameters `sock` (`BluetoothSocket`) – The `BluetoothSocket` to stop advertising the service on.

Raises `BluetoothError` – When SDP fails to stop advertising for some reason.

1.4.3 Exceptions

BluetoothError

exception `bluetooth.BluetoothError`

Bases: `OSError`

Raised when a bluetooth function or method fails for a Bluetooth I/O related reason.

1.5 License

Copyright © 2004-2019 Albert Haug and contributors; see [Contributors](#) for current list.

PyBluez 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 2 of the License, or (at your option) any later version.

PyBluez 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 PyBluez; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

(continues on next page)

(continued from previous page)

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 this service 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 make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. 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.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

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

GNU GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program).

(continues on next page)

(continued from previous page)

Whether that is true depends on what the Program does.

1. You may copy and distribute 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 and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

(continues on next page)

(continued from previous page)

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

(continues on next page)

(continued from previous page)

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), 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 distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the 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 a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions

(continues on next page)

(continued from previous page)

either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, 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.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE 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.

END OF TERMS AND CONDITIONS

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 convey 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 2 of the License, or (at your option) any later version.

(continues on next page)

(continued from previous page)

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, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

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

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

```
Gnomovision version 69, Copyright (C) year name of author
Gnomovision 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, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright interest in the program
`Gnomovision' (which makes passes at compilers) written by James Hacker.
```

```
<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice
```

This 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 Library General Public License instead of this License.

Indices and tables:

- genindex
- modindex
- search

A

accept() (*bluetooth.BluetoothSocket method*), 7
 advertise_service() (*in module bluetooth*), 9

B

bind() (*bluetooth.BluetoothSocket method*), 7
 BluetoothError, 12
 BluetoothSocket (*class in bluetooth*), 6

C

cancel_inquiry() (*bluetooth.DeviceDiscoverer method*), 8
 close() (*bluetooth.BluetoothSocket method*), 7
 connect() (*bluetooth.BluetoothSocket method*), 7
 connect_ex() (*bluetooth.BluetoothSocket method*), 7

D

device_discovered() (*bluetooth.DeviceDiscoverer method*), 8
 DeviceDiscoverer (*class in bluetooth*), 8
 discover_devices() (*in module bluetooth*), 10
 dup() (*bluetooth.BluetoothSocket method*), 7

F

family (*bluetooth.BluetoothSocket attribute*), 7
 fileno() (*bluetooth.BluetoothSocket method*), 7
 fileno() (*bluetooth.DeviceDiscoverer method*), 8
 find_devices() (*bluetooth.DeviceDiscoverer method*), 8
 find_service() (*in module bluetooth*), 10

G

get_l2cap_options() (*bluetooth.BluetoothSocket method*), 7
 getpeername() (*bluetooth.BluetoothSocket method*), 7
 getsockname() (*bluetooth.BluetoothSocket method*), 7
 getsockopt() (*bluetooth.BluetoothSocket method*), 7

gettimeout() (*bluetooth.BluetoothSocket method*), 7

I

inquiry_complete() (*bluetooth.DeviceDiscoverer method*), 9

L

listen() (*bluetooth.BluetoothSocket method*), 7
 lookup_name() (*in module bluetooth*), 11

M

makefile() (*bluetooth.BluetoothSocket method*), 7

P

pre_inquiry() (*bluetooth.DeviceDiscoverer method*), 9
 process_event() (*bluetooth.DeviceDiscoverer method*), 9
 process_inquiry() (*bluetooth.DeviceDiscoverer method*), 9
 proto (*bluetooth.BluetoothSocket attribute*), 7

R

recv() (*bluetooth.BluetoothSocket method*), 7
 recvfrom() (*bluetooth.BluetoothSocket method*), 7

S

send() (*bluetooth.BluetoothSocket method*), 7
 sendall() (*bluetooth.BluetoothSocket method*), 7
 sendto() (*bluetooth.BluetoothSocket method*), 7
 set_l2cap_mtu() (*bluetooth.BluetoothSocket method*), 7
 set_l2cap_options() (*bluetooth.BluetoothSocket method*), 8
 setblocking() (*bluetooth.BluetoothSocket method*), 8
 setl2capsecurity() (*bluetooth.BluetoothSocket method*), 8
 setsockopt() (*bluetooth.BluetoothSocket method*), 8

`settimeout()` (*bluetooth.BluetoothSocket method*), 8
`shutdown()` (*bluetooth.BluetoothSocket method*), 8
`stop_advertising()` (*in module bluetooth*), 12

T

`timeout` (*bluetooth.BluetoothSocket attribute*), 8
`type` (*bluetooth.BluetoothSocket attribute*), 8