Role Playing Game Engine

Version 0.0.1

Remco Bras

Table of Contents

1 About RPGE and obtaining it

1.1 About RPGE and this manual

RPGE is a GNU package, providing an engine for two-dimensional graphical role playing games. It is driven and extended by writing programs in GUILE, a dialect of Scheme. This manual is for RPGE 0.0.1 and aims to document the parts of the workings of RPGE that people using it should know about, the way RPGE is driven using GUILE and how RPGE can be obtained, installed and otherwise used.

1.2 Getting a copy

RPGE can be obtained using anonymous git or from the web. In case you wish to use anonymous git, the following command creates a local copy of the RPGE git repository: `git clone git://git.savannah.nongnu.org/rpge.git`. Those who prefer the web can download a copy of a release from the GNU mirrors, a list of which is kept at `http://www.gnu.org/prep/ftp.html`, or, if necessary, directly from `ftp://ftp.gnu.org` by FTP. Either of these options will allow you to receive a copy of RPGE, but keep in mind that the sources in git will be more current, but probably also buggier than the releases.

1.3 Installation

To install RPGE, your system will, at the time of writing (users of Git are advised to check the file 'README' in the root of the source tree for possibly updated information), need the following:

- An ISO C99-compliant C compiler that does not attempt to link when passed the -c option
- A make program compatible with Makefiles produced by GNU configure scripts
- A UNIX shell to execute GNU configure scripts
- GNU sed
- The SimpleDirectmediaLayer libraries and development files, including the sdl-image and sdl-ttf libraries
- GUILE, including the development files and libraries

If you have the above dependencies, you can install RPGE by moving to the directory containing the source code and running `./configure && make` to build RPGE. Once the build is complete and error-free, RPGE can be installed by running `make install` as a user with administrator privileges.

2 Invoking RPGE

2.1 A summary of command-line options

RPGE is invoked on the command-line using `RPGE` *`options`* in which options is one or more of the following, with their required arguments:

- -v or –version: Prints version information and exits.
- -h or –help: Prints a summary of usage and options and exits.
- -f *file*: Changes the default initialization file handled at startup from .RPGE to *file*

2.2 RPGE initialization files

Once started, RPGE parses command-line options, initializes the libraries it uses and, before entering the frame loop, processes an initialization file. This file, .RPGE by default, contains the names of files containing Scheme code for RPGE to execute, one per line, in sequence. Any lines starting with '#' are ignored.

This facility is used in the sample files to include several common Scheme files. It is advised to use these files, since they provide common interfaces for matters like dealing with events, handling the statistics of mobs and a generic table type. However, it is certainly possible and sometimes useful to replace them. Keep in mind that anything that depends on other files should be loaded after them.

Files loaded by initialization files may create threads, for example to load a certain mechanism to handle events in a loop. To do so, declare (use-modules (ice-9 threads)) at the start of your file and use (make-thread safe-load *filename*) to load the file. The safe-load procedure is a RPGE primitive, which uses a load mutex to ensure that no concurrent loads happen. Since looping mechanisms may run forever, such files should define all their procedures and call (unlock-mutex load-mutex) just before entering their main loops. This ensures that the background script is loaded properly and new loads can happen safely.

When loading a file referenced in an initialization file, RPGE looks for it in the current directory first, after which it checks through a sequence of scheme paths. These are directories that may contain scheme code. By default, this sequence is empty on startup. To fill the sequence, use a so-called colon directive in your initialization file, of the form `scheme-dir:`*`directory`*. This adds *directory*, which should be a complete, relative or absolute, path to the directory (for example /usr/local/share/RPGE/scm or ../), to the scheme paths. A similar facility exists for sprites, so the default sprites can be found by simply referencing them by name. To add an image search directory, use the directive `image-dir:`*`directory`*, which acts like the scheme-dir directive.

Since several common files and directories may be loaded and declared very often, RPGE provides an include directive to tell RPGE to load another configuration file. This does not use any paths yet, so when using it, the path passed to it should be a complete path to the other configuration file. Syntactically, the directive is `include:`*`file`*. To deal with the problem of loading the default libraries on startup, RPGE's build system builds a default configuration file for you using sed. This file is located in conf/default.conf, relative to the RPGE source directory. This file is installed on your system by make install and sets up the default scheme directories, image directories and libraries. RPGE also makes a

skeleton configuration file, conf/skeleton.conf, for you, which includes default.conf and loads the testing game, test.scm. Loading RPGE with this file (using for example `src/RPGE -f conf/skeleton.conf` in the top source directory after running make), will fail until make install has run. This is intentional, as the file is meant to make it easy to work with the included files in any RPGE installation.

3 RPGE's view of the world

3.1 Overview

In RPGE, a world is essentially a flat, 2-dimensional surface, divided into tiles. These tiles are small, rectangular areas, with configurable widths and heights. On these tiles, objects called mobs can move around in any way they wish, possibly blocked by certain tiles while freely passing over others. These mobs are each rendered as a so-called sprite, a 2-dimensional picture. The picture to render is determined by the file loaded for the mob and its animation status.

In the future, this model could be expanded to allow any number of concurrent 'worlds' on a single RPGE instance, something that could be especially useful if RPGE were ever networked. At present though, there is only one world and parallel universes are merely science-fiction.

3.2 Tiles

RPGE tiles need a little more discussion, since they have some interesting properties that users may want to be aware of. First, all tiles in the tilegrid are independent. Changing the data of a single tile does not change the data of the others, with one important exception. The one exception is that changing the image data used by a tile directly (something only possible through either runtime reloading of images or modifying the RPGE source, there is no API function) changes the image data used by all tiles using the image of the same filename and index.

Second, RPGE differentiates between the various directions a mob arrives at a tile from. For example, if a mob is moving right when it hits a blocking tile, that tile will only block if the BLOCK_LEFT flag is set. Similarly, for a mob arriving at any tile, the blocking flag for the corresponding edge is checked.

Third, RPGE has the facilities in place to differentiate between two types of mobs (dubbed 'ground' and 'air' mobs) and block none, one or both of those. However, these flags are unused at the time of writing.

Fourth, though RPGE does check for tile blocking while a mob is moving, mobs moving over tiles while they are being changed may cause some bugs. For example, a mob moving over a tile that does not block while it is being modified to block will probably proceed to move over the tile anyway, as blocking was checked when the mob entered the tile and will not be checked when the mob leaves it.

3.3 Mobs

Mobs, like tiles, have some interesting properties. The most important of these is the ability to add any data you want to any mob you want. This data, dubbed the 'mob user data ', can be queried and set from GUILE. By default, it is the empty list marker and protected from the garbage collector. Since RPGE only allows a single GUILE value of user data for every mob, it is recommended to fill this space with a list, table or other compound data structure so you can put arbitrarily large amounts of data in this space. The example statistics system and other systems that work with mob user data will presume this data is

a table. It is also worth noting that the default mob bootstrap procedure sets this data to an empty table.

This bootstrap procedure is another property of mobs that users can use to enhance mobs. Since systems using mob user data may need initialization, RPGE provides this procedure, which is called by the GUILE procedure make-mob only, not by the primitive create-mob, to set the initial value of this data. The procedure may however do any arbitrary amount of processing or data creation, as long as it takes a single argument representing the mob being created.

Closer to the C side of things, mobs are currently considered as wide and as high as a single tile. Therefore, RPGE will render sprites of SPRITE_WIDTH width and SPRITE_HEIGHT height, both of those macros being equal to the respective dimensions of tiles. When determining what sprite to render, RPGE looks up the image data associated with the image index given in the mob (images are loaded globally in RPGE, to save space) and takes a rect of the specified width and height, starting at a starting point determined by the current animation of the mob. Animations are laid out horizontally in the image data, being right next to each other and SPRITE_WIDTH wide each. Individual frames of a certain animation are in its column, right below each other with no padding. Essentially, RPGE treats the image data as a frame grid, indexed in the horizontal direction by the index of the mob's current animation and in the vertical direction by the index of the mob's current frame. Animations can loop automatically if that is desired.

In addition to being animated, mobs can move. RPGE provides two primitives for making mobs move, the generic move-mob which immediately changes the direction and movement of a mob, regardless of any current velocity. Since this is not very practical if you want mobs to move along a path specified by keying it in asynchronously, RPGE natively supports queueing up movements, which is done using mob-add-movement. If the mob is not moving, this procedure has the same effect as move-mob. Otherwise, the movement is queued up in a FIFO buffer, the first element of which is started when the mob is done with its current movement, and so on. This buffer is not cleared by calls to move-mob. If mobs collide however, the buffer is instantly cleared on both colliding mobs. The rationale behind this is that RPGE should wait for users to issue new commands, since a collision implicitly is a result of being unable to execute movement commands.

Finally, mobs support RPGE's event mechanism, which is currently hardly being used. In the future, this could be used to send an event whenever a mob completes a motion or, more generally, changes its motion. This would be useful for animation purposes.

4 Communication in RPGE

In RPGE, there are two kinds of communication, being communication between RPGE itself and the scripts running in it and communication between the scripts themselves. The latter type of communication can be handled in any way the users want, just like in any multithreaded program. Communication between RPGE and the engine is not so freeform, but is similar to communication in GTK, Xlib or a similar system.

This communication is handled by so-called events, which live on things called 'eventstacks'. Regardless of the specific stack, events themselves are simple messages, basically consisting of something to identify the particular type of message and whatever the user should know about it.

Another property of eventstacks one should be aware of is that they need to be opened before usage. Once opened, it is guaranteed that the user receives any event that arrives after the stack was opened. A stack without users does not accumulate any events at all. Regardless of this last property, it is advised to add any useful event to the appropriate stack, even if users are not guaranteed.

Stacks themselves live in RPGE, being assigned to either a certain object or just being global. There is currently one global stack and every single mob has a stack of its own. Global stacks should be used for events that are not related to anything that has a stack of its own and is a single object, for example a collision between two mobs or a keypress. Stacks assigned to the object are for events related to the object exclusively, for example an event sent when a mob moves across a tile boundary.

4.1 Event Reference

This section lists all current events. Note that events, to GUILE, are pairs whose car holds data identifying the particular type of event and whose cdr holds data associated with the specific event itself.

Global events:

- key-down:This event specifies a keypress. The car is the symbol 'key-down, the cdr is a symbol indicating the key pressed. Usually, this is the letter typed by pressing this particular key. For example, if the a key is pressed without holding down shift, the cdr of the associated event will hold the symbol 'a. Newlines, tabs and spaces are converted to 'newline, 'tab and 'space, respectively.
- collision:This event indicates a collision between two mobs. Its car is the symbol 'collision, its cdr is a pair, holding the two mobs that collided, in no particular order.

Mob events:

- tile-change:This event is sent whenever a mob (technically a mob's center) crosses a tile boundary. Its car is the symbol 'tile-change, its cdr is a pair of pairs, whose car holds a pair of the x and y coordinates (in tile coordinates) the mob had and whose cdr holds a pair of the new x and y coordinates.

Appendix A API Reference

This section describes the entire RPGE GUILE API, all the way from primitives to the stuff implemented in the examples that are part of the RPGE distribution.

A.1 Primitives

Primitives related to mobs:

- `(create-mob` *`x y sprite`*`)`:

  Creates a mob on the tilegrid at point (*x*,*y*). Note that *x* and *y* are tile coordinates, not pixel coordinates. The mob is rendered using *sprite*, specified as a string containing the filename of the sprite to be loaded. Animation data and the like are defaulted to zero.

- `(move-mob` *`mob x y frames`*`)`:

  Move *mob*, over *frames* frames, *x* tiles to the right and *y* tiles down. This modifies the movement data of the mob, canceling any current movement.

- `(add-mob-movement` *`mob x y frames`*`)`:

  Queue up the movement that would be done by the mob if move-mob was called with the same arguments. Queued up movements are executed in order once the mob is done moving.

- `(set-mob-animation` *`mob animation start target framesbetween loop`*`)`:

  Animate *mob*, using animation *animation*, starting at frame *start*, ending at frame *target*, changing to the next frame every *framesbetween* frames. If *loop* is true, the animation loops until stopped. Note that frames and animations are specified by indices, which start at zero.

- `(stop-mob-animation` *`mob`*`)`

  Immediately stop the animation of *mob* in its current state.

- `(get-mob-data` *`mob`*`)`

  Returns the piece of global data associated with *mob*.

- `(set-mob-data` *`mob value`*`)`

  Set the piece of global data associated with *mob* to *value*.

- `(open-mob-events` *`mob`*`)`

  Open the eventstack of *mob* and return the user index associated with this action.

- `(get-mob-event` *`mob user`*`)`

  Get the next event on *mob* for *user*, in which *user* is an index returned by open-mob-events.

- `(close-mob-evetns` *`mob user`*`)`

  Close the eventstack on *mob* for *user*, indicating that *user* does not want to receive any new events on this stack.

- `(destroy-mob` *`mob`*`)`

  Destroy *mob*.

Primitives related to tiles and grids thereof:

- `(create-tile sprite clipping-rect blocking)`

  Creates a tile, using *sprite* as the filename to load a sprite from, *clipping-rect* as a list representing a rectangular clipping area, containing the part of the sprite image that should be used for rendering, and *blocking* as an identifier indicating which mobs are blocked. The types of blocking RPGE currently checks for are block-<direction>, where direction is left, right up or down. The particular direction indicates the edge the mob to be blocked hits, i.e. a tile with block-right for *blocking* blocks mobs attempting to cross its right edge. There is a single combined constant combining all four directions, conveniently named block-all-directions. Other variations can be obtained by simply adding individual constants.
- `(init-tilegrid width height)`

  Resets the main tilegrid to a new grid of dimensions *width* and *height*. This should be called once, when initializing the game map. Note that the tilegrid created by this procedure does not have any tiles set and all tiles should be set manually.
- `(set-tile x y tile)`

  Sets the tile at (*x*,*y*) to *tile*.
- `(set-all-tiles tile)`

  Sets all tiles to *tile*.

Primitives related to windows:

- `(create-window width height x y sprite spritewidth spriteheight)`

  Creates a window *width* pixels wide and *height* pixels high, with its top-left corner at (*x*,*y*) (in pixel coordinates). The window is filled with a rectangular tile, *spritewidth* pixels wide and *spriteheight* pixels wide, taken from the image file *sprite*. Note that the tile is hardwired to have its top-left corner at (0,0).
- `(remove-window window)`

  Destroys *window*.

Font-related primitives:

- `(open-font filename size)`

  Opens the font referenced by *filename*, with a font size of *size*.
- `(close-font font)`

  Closes *font*.

Text-related primitives:

- `(make-text x y text font red green blue)`

  Create floating text at pixel coordinates (*x*,*y*), rendering *text*, using *font*. The text is rendered in the color specified by *red*,*green* and *blue*, where each variable represents the intensity of the color that is its namesake, on a scale from 0 to 255, inclusive.
- `(destroy-text text)`

  Destroy *text*.

Primitives controlling the main camera:

- `(get-camera-x)`

- `(get-camera-y)`

  These return the x and y-coordinates, respectively, of the main camera, in tile coordinates.
- `(set-camera-x` *x*`)`
- `(set-camera-y` *y*`)`

  Set the coordinates of the main camera, in tile coordinates.

Global miscellaneous primitives:

- `(get-global-data)`

  Returns the current value of the global data stored inside RPGE.
- `(set-global-data` *value*`)`

  Sets the current value of the global data, while protecting it from the garbage collector. This should be called with something like a list or a table, so it can be shared across scripts. The default scripts set this to a table and store data in that.
- `(open-global-events)`

  Opens the global eventstack, returning a user index.
- `(get-global-event` *user*`)`

  Returns the next global event in line for *user*.
- `(close-global-events` *user*`)`

  Closes the global eventstack for *user*.
- `(get-argv)`

  Gets the current argument vector of the thread, roughly presumed to be equal to the 'arguments' the script was 'called' with.
- `(load-with-argv` *script arglist*`)`

  Execute *script* in the current thread, passing it *arglist* as its arguments.
- `(safe-load` *script*`)`

  Load *script* safely, that is, avoid crashes due to concurrent load starts and ends.

A.2 Scheme-level APIs

In addition to the primitives defined by the RPGE core, the default RPGE initialization file loads several files that define many Scheme functions for use with RPGE. Usually, these are meant to provide functionality that the developers felt shouldn't be 'hardcoded' in RPGE and can be replaced by an equivalent built on either GUILE itself or a combination of GUILE and the RPGE primitives described in the previous section.

In the default initialization file, files are ordered in such a way that, at the very least, they will be loaded after their dependencies have been loaded. Therefore, if you replace files, it is recommended to check if files loaded after the deleted file will still load and the functions they define will still run. For example, not loading table.guile , which is referenced at the very start of the default initialization file, will break most functions in files loaded later, with the exception of utils.guile, which does not use any tables at all.

This section lists the APIs in the order in which they are loaded by default, roughly the order in which they depend on each other.

Table API (defined in table.guile):

- `(init-table)` Returns a fresh, initialized table.
- `(table? table)` Returns #t if *table* looks like a table. This function is not perfectly accurate.
- `(add-to-table table key value)` Returns *table*, with *key* bound to *value*. This does not modify the original *table*.
- `(add-to-table! table key value)` Returns *table*, modified to include *key* bound to *value*.
- `(get-from-table table key)` Returns the value *key* is bound to in *table*.
- `(set-in-table! table key value)` Bind *key* to *value* in *table*,modifying it. Returns 'error if *key* is not bound in *table*.
- `(multi-key-find-with-list table keylist)` Returns the value bound to the keylist in *table*. Essentially, for a list of length n, this returns the value bound to the successive members of the list in *table* and n-1 sub tables. On error, this returns '() if a table does not contain the right key and 'error if one of the sub tables is not a table.
- `(multi-key-find table . keys)` The same as the above, but using the rest argument keys as the key list.
- `(multi-key-add! table keylist value)` Using the same conventions for sub tables as the above two functions, add *value* to a subtable of *table*.
- `(multi-key-set! table keylist value)` Using the same sub table conventions, modify the value in a subtable of *table*, setting it to *value*.
- `(remove-from-table! table key)` Modifies *table*, removing the binding of *key*.

Key binding API (defined in keys.scm):

- `(bind-key key proc)` Add a new binding, binding *key* to *proc*, where *proc* is a procedure of no arguments.
- `(get-binding key)` Get the procedure bound to *key*, or '() if none can be found.
- `(remove-binding key)` Undo the current binding to *key*.

Mob event API (defined in mob_events.scm):

- `(mob-tracking-init)` Initialize mob event handling.
- `(add-tracked-mob! mob)` Make the mob event system track *mob*, handling events on this mob.
- `(remove-tracked-mob! mob)` Make the mob event system stop tracking *mob*.
- `(init-mob-bindings mob)` Initialize event binding for *mob*.
- `(bind-mob-event mob event proc)` Bind *event* to *proc* on *mob*. *proc* should be a procedure of 1 argument, the event sent by RPGE, which is a pair of a type (usually a symbol) and some associated data.
- `(execute-mob-binding mob event)` Execute the procedure bound to *event* for *mob*, passing it the event. Note that for this procedure, *event* should contain the full event sent by RPGE, but while binding, it should contain the type data for the event.

Mob bootstrapping API (defined in mobutils.scm):

- `(get-mob-bootstrap-proc)` Returns the current mob bootstrap procedure, or '() if none was defined.

- `(set-mob-bootstrap-proc! proc)` Set current mob bootstrap proc to proc, or define it to proc if it was not defined yet.
- `(make-mob x y sprite)` Make a mob using create-mob, passing it the arguments passed. Then, apply the mob boostrap procedure, if any, to the newly made mob. This procedure should be used whenever you 'just want to make a mob'. Use create-mob only if you have an explicit reason to avoid bootstrapping.

Mob statistics API (defined in stats.scm):

- `(stats-init mob)` Initialize the statistics system on the mob. Like many default procedures, this relies on the mob data being a valid table. It is recommended to call this and related procedures in the mob bootstrap procedure.
- `(get-stat mob stat)` Get the value of *stat* for *mob*. If *stat* is not set, this returns '().
- `(set-stat mob stat value)` Set the value of *stat* to *value* for *mob*. If *stat* is not set, this implicitly defines it.
- `(procedural-stats-init)` Initialize the procedural stats system. This requires that the global userdata is a valid table.
- `(add-procedural-stat-proc! stat proc)` Add *stat* as a procedural stat, meaning that get-stat will call *proc* on the mob passed to get-stat if *stat* is requested.
- `(get-stat-proc stat)` Returns the procedure used to calculate *stat*.
- `(remove-procedural-stat-proc! stat)` Remove the procedure bound to *stat*. After this, *stat* will be looked up by value on the mob, rather than being calculated using a procedure.

The above APIs usually have initialization procedures, which need to be called before using them. If the description of such a procedure does not mention any other constraints, it should be presumed that the global userdata needs to be a table and, for procedures that initialize something for a mob, the mob's user data must be a valid table as well.

GNU Free Documentation License

Version 1.2, November 2002

0. PREAMBLE

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

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

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

1. APPLICABILITY AND DEFINITIONS

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

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

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

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released

under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

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

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

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

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

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

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

2. VERBATIM COPYING

   You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and

that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

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

3. COPYING IN QUANTITY

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

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

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

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

4. MODIFICATIONS

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

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

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

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

D. Preserve all the copyright notices of the Document.

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

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

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

H. Include an unaltered copy of this License.

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

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

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

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

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

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

O. Preserve any Warranty Disclaimers.

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

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

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

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

5. COMBINING DOCUMENTS

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

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

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

6. COLLECTIONS OF DOCUMENTS

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

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

7. AGGREGATION WITH INDEPENDENT WORKS

   A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called

an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

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

8. TRANSLATION

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

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

9. TERMINATION

   You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document 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.

10. FUTURE REVISIONS OF THIS LICENSE

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

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

ADDENDUM: How to use this License for your documents

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

```
Copyright (C)  year  your name.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.2
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover
Texts.  A copy of the license is included in the section entitled ``GNU
Free Documentation License''.
```

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

```
with the Invariant Sections being list their titles, with
the Front-Cover Texts being list, and with the Back-Cover Texts
being list.
```

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

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