Dorkalize

F.Borghese & Giacomo

COLLABORATORS			
	TITLE : Dorkalize		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	F.Borghese & Giacomo	August 10, 2022	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

1 Dorkalize **1**

1.1 Dorkalize 1

1.2 What’s this? 2

1.3 Why this? 2

1.4 Disclaimer 3

1.5 Requirements 3

1.6 Installation 3

1.7 Quickstart for programmers 3

1.8 Quickstart for translators 5

1.9 Modifications to C source 6

1.10 Using CatComp without Dorkalize 7

1.11 Main window 7

1.12 Source files tab 8

1.13 Project options tab 8

1.14 Filters tab 9

1.15 Languages tab 10

1.16 String window 10

1.17 Filtered string window 11

1.18 Filter window 12

1.19 Translation window 13

1.20 Option window 13

1.21 Menus 13

1.22 Error messages 14

1.23 History 15

1.24 Distribution 15

1.25 Note for translators 15

1.26 Acknowledgements 16

1.27 Contacts 17

Chapter 1

Dorkalize

1.1 Dorkalize

```
                              DORKALIZE 0.9
                     by
             Francesco Borghese
             Giacomo Di Giacomo
```

What’s this?

Why this?

Disclaimer

Requirements

Installation

Quickstart for programmers

Quickstart for translators

Modifications needed for C code

Windows

Main window

Source files tab

Project options tab

Filters tab

Languages tab

String window

Filtered string window

Filter window

Translated strings window

Option window

Menus

Error messages

History

Distribution

Note for translators

Acknowledgements

Contacts

1.2 What's this?

Dorkalize is a utility that takes C source files and adds support for AmigaOS localization by replacing the strings contained in them with calls to the appropriate OS functions, allows input and editing of their translations, generates the catalog descriptor files used by CatComp to create catalogs, and then spawns CatComp itself to create the catalogs. This allows you to program without cluttering your sources with calls which make them much less readable.

Unlike other programs of the same type, Dorkalize allows you to perform further development of your program on the original sources, without having to deal with the clumsy defines and/or function calls which replace printable strings. Dorkalize will keep track of the modifications you had made to your sources and will recover the localized strings and their translations from the previous version. At every new version, you only have to select which strings of the new ones you entered you want to be localized, enter their translations, and compile the localized sources. Dorkalize will do all the rest, up to creating the catalog.

This program was started to be a replacement for the buggy parsing function of Commodore's "Localize" program, which is used to localize C source programs. At least for us, it has never worked. It has then turned into a powerful utility featuring advanced string filtering, automatic patching of source files, built-in editing and translation of strings, recovery of translated strings from older catalog descriptors, and spawning of CatComp.

1.3 Why this?

Because when we tried to localize our Italian lotto program AmiSuperLotto we discovered, much to our disappointment, that Localize did not work correctly. Specifically, the line numbers to patch were incorrectly reported. In addition to this, we wanted to develop a program to suit our needs (and maybe other programmers') and allow for a quick translation of programs.

1.4 Disclaimer

No warranty is given that Dorkalize will perform the expected task. We take no responsibility for any kind of damage that Dorkalize can cause to its users.

1.5 Requirements

Dorkalize requires any Amiga with at least OS 2.1 and MUI 3.0 or better, the MUI custom classes NList.mcc and BetterString.mcc (included in the distribution archive), and the Commodore localization tool CatComp. You may use Commodore's Localize tool, but this is neither requested nor useful (since it contains more than one bug). You can find CatComp and Localize in the 3.1 Native Developer Kit.

1.6 Installation

To install Dorkalize simply double-click on the "Install" icon and follow the prompts. You need the Commodore Installer to execute the installation script.

1.7 Quickstart for programmers

If you are a programmer, you can use Dorkalize to generate, starting from your nonlocalized sources, a version of the same sources that supports localization.

Note that your programs need some modifications you have to do by yourself before they can be correctly localized.

When you start Dorkalize, you will be presented with the main window. The first time you start Dorkalize you should set up all the options, in particular you should tell Dorkalize where CatComp and, if you wish, Localize are found. This is required to generate the C header file containing the translation information, and the catalog files.

When you have set the global options, you should set the options for your current project. Select the Project options tab

in the
main window
to set
them up. You should specify the following items:

- The descriptor file path and name. This is the .cd file that contains the strings to be localized and that will be used later to generate the catalogs.

- The path and name of the C header output file. This file will be automatically #included in the patched source files when needed. Make sure it resides in the same directory as your localized source files before compiling them.

- The working directory. Dorkalize will store the data needed for automatic string recovery here. Do not delete the .str file if you want to take advantage of this feature.

- The source directory for the translation files. If you have some old translation files (the files with a .ct extension used by CatComp) that you want to recover, put them in a subdirectory of the source directory that has the same name as the language they refer to. For example, if you have a previous .ct file containing the old Italian translations, and you specified "Dev:Workdir/" as source translation directory, you must put the file into "Dev:Workdir/italiano/". Dorkalize expects that the name of this file minus the .ct extension be the same as the name of the project minus the .dprj extension, if this is present.

- The destination directory for the translation and catalog files. Dorkalize will put the translation and catalog files it generates into subdirectories of this directory that have the same name as the languages the files refer to (see above). NOTE THAT IF YOU USE THE SAME DIRECTORY FOR SOURCE AND DESTINATION YOUR OLD TRANSLATIONS WILL BE OVERWRITTEN AND YOU WILL NOT BE ABLE TO RECOVER THEM IN CASE OF ERROR!

You can also choose a filter file from the
Filters tab
to automatically
exclude some strings from the localization process. The filter window allows you to edit filter files. A general-purpose filter file, dorkafilter.dflt, containing patterns for strings which you usually don’t want to localize, is supplied with Dorkalize.

You should keep the default options for the tracking filter. This will allow Dorkalize to recover the localization status of the old strings: if Dorkalize had already patched those files, the strings you had decided not to localize will be automatically put among those that will not be localized. A filter or yourself can override this decision. If you do not want to take advantage of the tracking filter, set the tracking depth to zero.

Finally, if you want to generate some of the translations yourself, you should select which languages you want to generate translations for by adding them to the list in the
languages tab
.

Now you can choose the files you want to localize in the

source files tab
.
After you have done so, press the button labeled "Dorkalize" to parse the files.

Dorkalize will open the
string window
, which contains the strings which
survived the filtering process. To go on with the localization process, remove the strings you don’t want to translate double-clicking on them or selecting them and then pressing the "Remove" button. You will find the strings you removed in the
filtered string window
, which contains a list
similar to the one above. Double-click on any entry in this list to move it back to the initial list; alternatively you can select some entries and move them back with the "Add back" button.

At this point, you can patch your sources and generate the catalog descriptor and C header file by pushing the "Patch sources" button. Each patched source will be put into a subdirectory, called "Localized-source", of its original path.

If at least one language has been selected, Dorkalize will proceed recovering older translations from the source translation directory and opening the

translated strings window
. If you want to enter translations and generate
catalogs yourself, refer to the
quickstart for translators
to learn how to do
that.

1.8 Quickstart for translators

If you are a translator, you can use Dorkalize to generate, ↩
starting from a
catalog descriptor supplied by the programmer, a catalog for each language of your choice.

When you start Dorkalize, you will be presented with the
main window
. The
first time you start Dorkalize you should set up the
options
, in particular
you should tell Dorkalize where CatComp is found. This is required to generate the catalog files.

When you have set the global options, you should set the options for your current project. Select the
project options tab
in the
main window
to set

them up. You should specify the following items:

- The descriptor file path and name. This is the .cd file that contains the strings to be localized and that will be used later to generate the catalogs.

- The source directory for the translation files. If you have some old translation files (the files with a .ct extension used by CatComp) that you want to recover, put them in a subdirectory of the source directory that has the same name as the language they refer to. For example, if you have a previous .ct file containing the old Italian translations, and you specified "Dev:Workdir/" as source translation directory, you must put the file into "Dev:Workdir/italiano/". Dorkalize expects that the name of this file minus the .ct extension be the same as the name of the project minus the .dprj extension, if this is present.

- The destination directory for the translation and catalog files. Dorkalize will put the translation and catalog files it generates into subdirectories of this directory that have the same name as the languages the files refer to (see above). NOTE THAT IF YOU USE THE SAME DIRECTORY FOR SOURCE AND DESTINATION YOUR OLD TRANSLATIONS WILL BE OVERWRITTEN AND YOU WILL NOT BE ABLE TO RECOVER THEM IN CASE OF ERROR!

Finally you should select which languages you want to generate translations for by adding them to the list in the languages tab.

At this point you should save the project to avoid losing the settings when you quit Dorkalize.

Now press the "Recover translations" key. Dorkalize will open the translated strings window. In the rightmost column of the list you can enter the translations of the strings that appear in the central column. You can select the language you are working upon using the cycle gadget. When you're done you can choose to create the catalogs (Dorkalize will spawn CatComp itself) for the language selected or for all at once. You can also choose to create the translation files for CatComp and launch CatComp by yourself, if you wish to.

Note that you cannot save your work except by creating the translations or catalogs. In this case, the new strings will be in the .ct file found in the destination translation directory. You should move this file to the source translation directory to recover them next time you use Dorkalize.

See generating catalogs to learn how to use the catalog descriptor to generate catalogs without using Dorkalize.

1.9 Modifications to C source

First, all the files that contain strings you want to localize must include
the C header that defines the pragmas for the locale.library; for example,
for SAS/C this is:

```
    #include  <proto/locale.h>
```

Then, one of your files (which will most often be the one which contains the
main() function) must include the locale.library definitions header:

```
    #include  <libraries/locale.h>
```

and must declare the catalog variable as global:

```
    struct Catalog  *catalog;
```

NOTE THAT YOU HAVE TO NAME THIS VARIABLE JUST "catalog", NOTHING ELSE; THIS
IS BECAUSE DORKALIZE WILL REFER TO IT WITH THIS NAME.

Then you have to open locale.library and the catalog itself; for example, the
Dorkalize code that performs these actions is:

```
    if (LocaleBase = OpenLibrary("locale.library", 38))
    {
        catalog = OpenCatalog(NULL, "dorkalize.catalog",
                              TAG_DONE);
    }
```

Finally, all the files that will be localized, except the one above, must
contain a reference to the catalog variable:

```
    extern struct Catalog  *catalog;
```

1.10 Using CatComp without Dorkalize

You can use CatComp to generate a blank translation file from a catalog
descriptor with:

```
CatComp <catalog descriptor.cd> CTFILE=<blank translation file.ct>
```

You can refer to CatComp’s guide for a full explanation of how to edit this
to get a translation file. Finally, use

```
CatComp <catalog descriptor.cd> <translation file.ct>
   CATALOG=<catalog file.catalog> CFILE=<header file.h>
```

to generate all the files you need.

Note that you need to include the generated header file in your source and
define the constants CATCOMP_NUMBERS and CATCOMP_STRINGS before compiling
(at least this is the method we use).

1.11 Main window

The main window contains a register with four tabs.

Source files tab

Project options tab

Filters tab

Languages tab

Under the register file are two buttons. The "Dorkalize" button starts the parsing of the files and opens the string window. This is the only moment in which filtering is applied. The "Recover translations" button attempts to recover older translations if present and then opens the translated strings window.

1.12 Source files tab

The "Source files" tab contains the list of the source files to process. You can add files entering their name in the string gadget below or selecting them with the associated popup gadget (which supports multiple selection). You can remove any source file from the list by double-clicking on it.

1.13 Project options tab

The "Project options" tab contains:

- A checkmark labelled "Localize strings in defines". When selected, the strings found inside #define directives are by default localized, otherwise they are not. You can individually change this attribute for each one of them in the string window.

- A checkmark labelled "Use internal patcher". When selected, Dorkalize will patch the source files using its built-in patching routine, otherwise will invoke Localize. There is no real reason to uncheck it, unless you find a bug in the Dorkalize patcher (we found at least one in Localize, anyway), or you want to generate a catalog descriptor for each single file.

- A text gadget displaying the command line used to invoke Localize. You cannot alter its contents directly, but only acting upon the various options.

- A checkmark labelled "Merge catalogs". When selected it activates the MERGECATALOG option of Localize, which means that you will get only one catalog descriptor file for all of your source files. It is selected by default and should stay so unless you have any special requirements (e.g. the source files refer to more than one executable file). When selected, the descriptor file name is decided by the contents of the "Descriptor name" gadget.

- A string gadget and associated popup labelled "Descriptor name". If you checked the "Use internal patcher" checkmark or you selected the "Merge catalogs" option, you must insert here the path and name of the catalog descriptor (.cd) file you want to create.

- A string gadget and associated popup labelled "String output file". You must insert here the path and name of the patch file created by Dorkalize as an input to Localize.

- A string gadget and associated popup labelled "Working directory". Dorkalize will store the data needed for automatic string recovery here. Do not delete the .str file if you want to take advantage of this feature.

- A string gadget and associated popup labelled "Source translation directory". If you have some old translation files (the files with a .ct extension used by CatComp) that you want to recover, put them in a subdirectory of this directory that has the same name as the language they refer to. For example, if you have a previous .ct file containing the old Italian translations, and you specified "Dev:Workdir/" as source translation directory, you must put the file into "Dev:Workdir/italiano/". Dorkalize expects that the name of this file minus the .ct extension be the same as the name of the project minus the .dprj extension, if this is present.

- A string gadget and associated popup labelled "Destination translation directory". Dorkalize will put the translation and catalog files it generates into subdirectories of this directory that have the same name as the languages the files refer to (see above). NOTE THAT IF YOU USE THE SAME DIRECTORY FOR SOURCE AND DESTINATION YOUR OLD TRANSLATIONS WILL BE OVERWRITTEN AND YOU WILL NOT BE ABLE TO RECOVER THEM IN CASE OF ERROR!

1.14 Filters tab

The "Filters" tab contains two groups.

- The "Pattern filters" group contains a string gadget and associated popup labelled "Filter file". This gadget contains the path and name of the file containing the filters used to rule off strings while scanning the source files. You can view and edit its contents by pressing the "Show filters" button, which will open the filter window .

- The "Tracking filter" group configures the tracking filter. The tracking filter scans the data of the last localization you performed on that project and tries to determine, for each string, whether it was localized or not. If it was not localized it is by default filtered out, unless a pattern filter decides it must instead be localized. This is extremely useful to avoid having to select each time which strings you want to localize. Anyway it is not unmistakable and you should check if all and only the strings you want to localize are in the string list.

The tracking filter works by searching for patterns of strings that appear in the same order both in the old and new source files. The tracking depth is the number of consecutive strings that must appear in the same order to "lock" the filter. The higher this number, the more unlikely it is that the filter will lock erroneously, but the higher the chance to miss strings. A depth of 2 is a good compromise. A depth of 0 disables the tracking filter.

The "Track inside files only" checkmark determines whether the filter searches for patterns in the whole set of files (not checked) or in each file separately (checked). Leaving it checked will do no harm.

1.15 Languages tab

The languages tab contains the list of languages you want to generate catalogs for. To add a file, use the "Add" button: a list will open containing the supported languages. To remove a language from the list, select it and click the "Remove" button, or double-click it.

You can modify the supported languages editing the file "Dorkalize.lang", located in the Dorkalize main directory. Simply enter one language per line.

1.16 String window

This window contains a listview holding the strings that Dorkalize ↩ found in
the selected source files. Each line refers to a string. The listview has seven columns:

The "Source file" column contains the source file from which the string was extracted.

The "Pos" column contains a character which is d or D if the string was contained in a #define, S if the string was contained in the body of a function, and E otherwise. Strings marked with D or E will not be localized. Dorkalize will set the attribute for strings in #defines depending upon the status of the "Localize strings in defines" checkmark in the

project options tab
. Double-clicking on this column will toggle between the
d and D attributes.

The "Line", "Start" and "End" columns contain the line of the source file, the initial column and the final column where the string is found.

The "Msg #" column contains the number that Dorkalize assigns to each string. Identical strings in different places have the same number.

The "Text" column contains the text of the string.

Below the listview are six buttons. The "Remove" button will remove the selected strings from the list and put them into the filtered strings list. The "Add to filter" button adds the selected strings to the filter list, without removing it from the string list. You have to press again the "Dorkalize" button to have the new filters take effect. The "Show filtered" button opens the filtered string window. The "Show filters" button opens the filter window.

The "Create patch file" button will create a patch file containing all and only the strings in the list. The "Patch sources" button will create the same patch file and then start Localize or the internal patcher, depending on the setting you entered in the project options tab, to generate the localized C sources, the C header file and the catalog descriptor. See the description of the project options to learn how to configure the process. The localized C sources will be put into the "Localized-source" subdirectory of the directory containing the original source file. If you selected some languages in the languages tab in the main window, the translated strings window will open.

Double-click on any column of a line (except the "Pos" column for 'd' or 'D' strings) to remove the string from the list and insert it into the filtered strings list (see later). Double-click on the "Pos" column of a 'd' or 'D' string to toggle its status between 'd' and 'D'.

1.17 Filtered string window

This window contains a listview holding the strings that were filtered off the source files or removed later by the user from the list in the string window.

The "Source file" column contains the source file from which the string was extracted.

The "Pos" column contains a character which is d or D if the string was contained in a #define, S if the string was contained in the body of a function, and E otherwise.

The "Line", "Start" and "End" columns contain the line of the source file, the initial column and the final column where the string is found.

The "Msg #" column contains the number that Dorkalize assigns to each string. Identical strings in different lines have the same number.

The "Text" column contains the text of the string.

Below the listview are two buttons. The "Add back" button will remove the selected strings from the list and put them back into the strings list in the string window. The "Add to filter" button adds the current string to the filter list, without removing it from the filtered string list. You have to press again the "Dorkalize" button to have the new filters take effect.

Double-click on any column of a line to remove the string from the list and insert it back into the strings list in the string window.

1.18 Filter window

This window contains a listview holding the filters for your current project.
The listview has three columns. The "Filter" column contains the filters. The "Type" column controls whether the filter is a stopping or a passing one. The "Status" column controls whether the filter is active; if it contains "INACTIVE" the filter is ignored.

Each filter is an AmigaDOS standard pattern. Filtering works like this: Dorkalize checks every string it finds in the source files against each active filter in sequence, top to bottom of the list. When it finds a filter matching the string, if the filter is a stopping one the string will not be added to the list of strings to be localized, otherwise it will be added; in any case Dorkalize will go on with the next string in the source files. Matching is case-sensitive. If the string does not match any filter it will be added to the list.

From above derives that the filters appear in the list by descending priority, top to bottom. This means that, if you want for example to exclude all the strings starting with "dork", except "dorkalize", you have to insert "dorkalize" as a passing filter and, below it, "dork#?" as a stopping filter. You can refer as an example to the supplied file, dorkafilter.dflt, which contains some filters you should always keep active.

On the right of the listview are two buttons that you can use to move the active filter up and down the list. You can also move selected filters using drag’n’drop. Double-clicking on the Filter column removes a filter from the list, while double-clicking on the Type and Status column will toggle the contents of the clicked field. The active filter can be edited in the string gadget below the listview; the changes are entered by pressing the Enter key.

In the bottom part of the window are seven buttons. "New" inserts a new filter. "Toggle type" toggles the type of the selected filters between STOP and PASS. "Toggle status" toggles the status of the selected filters between ACTIVE and INACTIVE. "Remove" removes the selected filters. "Clear" empties the list. "Save" saves the list into the file specified in the filters tab

in the
main window
. "Save as" saves the list into a user-specified file,
which then is also set as the current filter file.

1.19 Translation window

This window contains a list with three columns and a row per message. The "Message number" column contains the message identifier. The "Original string" column contains the nontraslated message. The "Translation" column contains the translation for that string into the language selected by the cycle gadget below. If a previous translation is found from an older translation file, it will be shown here, otherwise this field will be empty. You can edit the translation for the active row in the string gadget just under the list; remember to press Enter to confirm your edits. The "Catalog version" string gadget contains the version of the catalog you are creating; if one is recovered from an older translation file it will be shown here.

The four buttons on the bottom are used to create the translation file (.ct) for use with CatComp, or directly the catalog (spawning CatComp), for the current language or for all the selected languages.

1.20 Option window

The option window is opened from the "Set options..." menu item in the "Options" menu. It contains:

- A string gadget with associated ASL popup gadget marked "Localize executable:". This must contain the location of the Localize executable, if you just want to use Localize instead of Dorkalize’s internal patcher.

- A string gadget with associated ASL popup gadget marked "CatComp executable:". This must contain the location of the CatComp executable.

- A set of radio buttons and a string gadget with associated popup labelled "Startup project". From here you can select if the next time you start Dorkalize the startup project will be none, the last project you saved, or a default project specified in the string gadget.

- Two buttons labelled "Use" and "Save", whose function is straightforward.

1.21 Menus

Dorkalize has two menus: the Project and the Options menu.

The Project menu contains the following items:

- New project (shortcut RightAmiga + N): Clears the source file list and filter file name and resets the project options to defaults.

- Load project (shortcut RightAmiga + L): Loads the files to localize, project options and filter file from a user-specified project file. The project files have by default extension .dprj.

- Save project (shortcut RightAmiga + S): Saves the project files, project options and filter file to the same file they were written last time. If this is the first time you save the project, you will be prompted for a file name.

- Save project as: Saves the project files, project options and filter file to a user-specified project file.

- About: Displays information about Dorkalize.

- About MUI: Displays information about MUI.

- Quit (shortcut RightAmiga + Q): Exits Dorkalize.

The Options menu contains only the "Set options" item, which opens the global

option window
.

1.22 Error messages

Error	Description
Out of memory	Memory allocation failed. Close some applications or buy more memory.
Can’t open file	Dorkalize cannot open the file specified. Check if the path is correct and the file does exist.
Can’t close file	Dorkalize cannot close the file specified. This should not normally occur. If it happens, maybe you are in trouble.
An error has occurred because	Unexpected error: maybe your Amiga has turned into a Macintosh.
Wrong file type	Dorkalize cannot recognize the file specified as a valid type. Check if you selected the right file or if you are using an incompatible version of Dorkalize.
Wrong file version	The file specified cannot be read by the current version of Dorkalize.
Can’t create directory	Dorkalize cannot create the specified directory. Check if you specified a valid path and a valid name for it.
Can’t run external program	Dorkalize cannot execute the specified external program. An error code is also reported. Check if you specified a correct path for the external program.

```
No file selected    Dorkalize requires that you select at least one file
                    before continuing.

No language         Dorkalize requires that you enter at least one language
selected            in the language list before continuing.

No project name     Dorkalize requires that you save your project and assign
specified           a name to it before continuing.
```

1.23 History

```
0.9     16/02/01    Added the tracking filter.
                    Added the internal patcher.
                    Added recovery of old translations.
                    Added built-in translation editing.
                    Added built-in support for multiple languages.
                    Added support for translators.
                    Added spawning of CatComp.

0.3     14/07/00    Added projects.
                    Added filtering.
                    Added confirmation requesters before quitting or saving.
                    Reorganized the GUI.
                    Removed some bugs in parsing functions.

0.2     12/05/00    Added selection of strings to localize.
                    Added menus.
                    Added spawning of Localize.
                    Added support for strings in #defines and outside functions.

0.1     26/08/99    First release. Replaces Localize parse function.
```

1.24 Distribution

This program is freeware. You can use and spread it any way you like. Please email us if you use it and you think there is something which does not work or could be improved.

NList.mcc and BetterString.mcc are property of the respective authors. Please read the included readme's for copyright information.

1.25 Note for translators

We have included the descriptor file "Dorkalize.cd" in the " ↩
Catalogs" dir of the
main archive, so you can translate Dorkalize in other languages. We obviously suggest to use Dorkalize to translate the .cd.
Contact us
us if you need support.

We would like to receive the .ct and .catalog files, so we can support other
languages in future releases.

1.26 Acknowledgements

```
                          This application uses


                        MUI - MagicUserInterface

                (c) Copyright 1992-97 by Stefan Stuntz


MUI is a system to generate and maintain graphical user interfaces. With
the  aid  of  a  preferences program, the user of an application has the
ability to customize the outfit according to his personal taste.

MUI is distributed as shareware. To obtain a complete package containing
lots of examples and more information about registration please look for
a  file  called  "muiXXusr.lha"  (XX means the latest version number) on
your local bulletin boards or on public domain disks.

          If you want to register directly, feel free to send


                         DM 30.-  or  US$ 20.-

                                  to

                             Stefan Stuntz
                        Eduard-Spranger-Straße 7
                             80935 München
                                GERMANY



             Support and online registration is available at

                          http://www.sasg.com/

-------------------------------------------------------------------------

NList © 1996-1998 Gilles Masson
    Les Balcons d'Antipolis
    Bloc C
    15, Traverse du Barri
    06560 VALBONNE
    FRANCE
    e-mail: masson@iutsoph.unice.fr

-------------------------------------------------------------------------

BetterString © Allan Odgaard
    Dagmarsgade 36
    DK-2200 Copenhagen
    email: Duff@DIKU.DK
    You can find the latest version of BetterString at:
    http://www.DIKU.dk/students/duff/
```

1.27 Contacts

You can contact the authors at the following addresses:

```
    FRANCESCO  BORGHESE
    VIA G. SPINEDI 39
    00015 MONTEROTONDO (ROMA)
    ITALY

    email: fraborg@tiscalinet.it

    ICQ UIN: 66104693

    http://members.tripod.com/BorgheseF/
```

Feel free to send comments and suggestions.