

SIEGFRIED LOCALE EDITOR

User Guide

Copyright by Siegfried Soft

**Siegfried Soft
Hauff-Weingärtner GbR
Reichenberger Str. 12
D-34246 Vellmar**

Introduction

What is the Siegfried Locale Editor?

The Siegfried Locale Editor makes it possible to create new "locale" files or edit existing files. "locale" files contain language data (text) for applications that are using the "sfliblocale.so" Library to localize their user interface. The usage of the library is free for all BeOS developers, this includes commercial developers as well! There is documentation for the usage of the library (sfliblocale developer guide).

The Siegfried Locale Editor can be used by developers which include the "sfliblocale.so" in their own projects, as well as by every other user desiring to expand the language capabilities of an application that is using the library (e.g. to include further language support to Siegfried Backup).

By using the editor it's easy to check whether an application supports localization by the "sfliblocale.so" library. If so, the editor has the ability to get the built-in text data (mostly the english text) from the application as a reference for translation.

The principle of localization

The kind of localization in the sfliblocale.so library is quite simple. Every text which should be localized is included in a table. The text is identified by the position in the table (the table row). The main function of the library returns a pointer to the text by using the row number as parameter. For every language that should be supported, such a table must be created. Corresponding to the row number the text will be translated to the appropriate language. Every table is saved as a single file. To get multi language support only the appropriate table ("locale" file) is loaded and used. By adding "locale" files, an application can be easily expanded for new languages without any additional development effort!

System requirements

The Siegfried Locale Editor is running on BeOS R5 or higher. Your computer should be equipped with a minimum of 32 MBytes RAM and 2 MBytes of free disk space.

Siegfried Locale Editor supports the x86 platform as well as the PPC.

Notice / Trademarks

In this manual, we mention several registered trademarks which are not marked as such in the body of the text. Hence you cannot conclude from the missing identification that the corresponding product name is free from rights of third parties:

- Be, BeOS is a trademark of the company Be Inc.

Warranty

Siegfried Soft has made every effort to insure the quality and reliability of this product. Siegfried Soft does not accept any liability for damage created by the use or misuse of this product.

Error reports, criticism, suggestions

If you should stumble over any error while using the Siegfried Locale Editor or if you should have any suggestions for improvement: please write to us! We are open to positive and negative criticism as well as to suggestions. We will try to incorporate your ideas in the Siegfried Locale Editor as far as possible. If you should find any error, please specify your computer configuration.

Please write to the following address:

Siegfried Soft
Hauff-Weingärtner GbR
Reichenbergerstraße 12
D-34246 Vellmar (Germany)

WEB: <http://www.siegfried-soft.de>
EMail: hauff@siegfried-soft.de

Are you aware of Siegfried Backup? If not, take this opportunity to correct this.

Siegfried Backup is a powerful and easy-to-use application to backup and restore all your software and data. It is the optimal tool for beginners and experts wishing to backup and restore any kind of data speedily and smoothly.



Siegfried Backup contains all the functions demanded of a top quality backup application. Furthermore, special attention has been paid to sophisticated graphic user guidance.

The idea behind the tool is to make work with Siegfried Backup as easy and efficient as possible for beginners as well as for experts. You will appreciate the manifold features offered, long after your initial steps with the tool. A large number of specific functions (such as filters, scheduler, compression, add-ons and many others) will turn your daily work with Siegfried Backup into a genuine pleasure.

Get a demo version of Siegfried Backup now: <http://www.siegfried-soft.de>

Installation

Scope of supply

The following files should be contained in the delivery of the Siegfried Locale Editor:

| Name | Folder | Comment |
|---|---|--|
| LocaleEditor | SiegfriedLocale | Main program |
| SiegfriedLocale-UserGuide-English.html | SiegfriedLocale/Documentation | User manual (english) |
| SiegfriedLocale-UserGuide-Deutsch.html | SiegfriedLocale/Documentation | User manual (german) |
| SiegfriedLocale-UserGuide-English.pdf | SiegfriedLocale/Documentation/pdf | User manual as PDF (english) |
| SiegfriedLocale-UserGuide-Deutsch.pdf | SiegfriedLocale/Documentation/pdf | User manual as PDF (german) |
| SiegfriedLocale-DeveloperGuide-English.html | SiegfriedLocale/Documentation | Developer manual (english) |
| SiegfriedLocale-DeveloperGuide-Deutsch.html | SiegfriedLocale/Documentation | Developer manual (german) |
| SiegfriedLocale-DeveloperGuide-English.pdf | SiegfriedLocale/Documentation/pdf | Developer manual as PDF (english) |
| SiegfriedLocale-DeveloperGuide-Deutsch.pdf | SiegfriedLocale/Documentation/pdf | Developer manual as PDF (german) |
| sfliblocale.so | SiegfriedLocale/lib | Localization library |
| Deutsch | SiegfriedLocale/locale | German language file for locale library |
| SFLocale.h | SiegfriedLocale/develop/include | Include for software development |
| sfliblocale.so | SiegfriedLocale/develop/lib/x86 | Dynamic link library for software development x86 platform |
| sfliblocale.so | SiegfriedLocale/develop/lib/ppc | Dynamic link library for software development PPC platform |
| HelloLocale_x86.proj | SiegfriedLocale/develop/lib/sample | Project file for x86 sample application |
| HelloLocale_ppc.proj | SiegfriedLocale/develop/lib/sample | Project file for PPC sample application |
| HelloLocale.cpp | SiegfriedLocale/develop/lib/sample | Main source file of the sample |
| HelloLocale.h | SiegfriedLocale/develop/lib/sample | Header file for application class |
| HelloLocale.rsrc | SiegfriedLocale/develop/lib/sample | Resource file for sample application |
| HelloLocaleWin.cpp | SiegfriedLocale/develop/lib/sample | Window class source |
| HelloLocaleWin.h | SiegfriedLocale/develop/lib/sample | Defintion of window class |
| SFTexts.h | SiegfriedLocale/develop/lib/sample | Default text for sample application |
| SFTextIDs.h | SiegfriedLocale/develop/lib/sample | Text IDs for sample application |
| sfliblocale.so | SiegfriedLocale/develop/lib/sample/lib | Siegfried Locale Library used by the sample application |
| Deutsch | SiegfriedLocale/develop/lib/sample/locale | German language file for sample application |

Installation

The Siegfried Locale Editor is delivered as a ZIP archive. To install the program simply double click the icon of the "SiegfriedLocale1.00_x86.zip" or "SiegfriedLocale1.00_ppc.zip" (platform dependent). The BeOS Expander starts and a destination directory can be selected. The Expander creates in the destination folder a subdirectory called "SiegfriedLocale" which contains all files of the Siegfried Locale Editor.

At the initial start of the Siegfried Locale Editor the necessary directories and files for internal administration will be created in the 'SiegfriedLocale' folder.

Deinstallation

To deinstall the Siegfried Locale Editor simply remove the folder 'SiegfriedLocale'.

Using the Siegfried Locale Editor

Start

To start the Siegfried Locale Editor double click the program icon. A double click on an existing "locale" file starts the editor and loads the file automatically.

It's also possible to start the editor from the terminal:

```
LocaleEditor <RETURN>
```

To work correctly the Siegfried Locale Editor needs the "sfliblocale.so" library at the local "lib" folder or "/boot/home/config/lib" folder. If the editor can't find the library during start up, an error message appears ("Could not open "LocaleEditor" (Missing library: sfliblocale.so)") and the editor terminates.

Structure of "locale" files

"locale" files contain mostly the text data of an application. But they include additional data:

- Name of the language
- Application ID
- Version

This information can be shown/edited above the edit area (fields "Language", "Version" and "Application ID". The name of the language (e.g. "Deutsch", "English") is the name that is returned by the language evaluation of a "locale" file. It's the native language name of the text data. The language name must not be the filename of a "locale" file. Theoretically a "locale" file for swedish can be named "Tomato.locale" or "Shuaheli.locale", but this isn't rich in meaning. Normally a "locale" is named for the native language that the file supports. If needed the application name can be added (eg. "SFImageFrancais.locale").

The application ID is a unique identifier for an application that is using "locale" files. The ID shows whether a "locale" file can be used by the application or not. Nothing brings up more strange effects than a "locale" file that isn't directed to the application ;-). The application ID is created by the developer of an application. All "locale" files of an application must have the same ID. Files with other IDs are ignored by the application. The developer is free to decide what kind of string is used for the application ID. It's a good idea if there is a reference to the application. For example, the Siegfried Locale Editor is using as application ID "siegfried localeeditor locale".

Note for developers: it's possible to use as ID the BApplication object signature of the application, because it's sfice unique.

The edit area

The edit area is the most important part of the Siegfried Locale Editor. All text to localize for an application will be managed here. The texts are organized in rows and separated by a <RETURN>. The order of the texts is forcing! An application identifies the texts with the row number of the text. The principle of the text management is a table. Every row of the table is a text for the application. If, for example, the application need the 10th. text, a ten as argument is delivered to the sfliblocale.so library.

Texts that are longer as the width of the editor window will be automatically wrapped. The wrapping doesn't include a <RETURN>! Every table row is used as floating text. Empty rows are ignored by the editor. Additional comments can be included. A comment line is marked by a semicolon " ; " at the start of the line. The end of a comment is marked by a <RETURN>. Comment lines are shown in the color red. **Important:** comment lines will be not saved to the "locale" file. Comment lines are always temporary to a session!

When an existing "locale" file is edited, the new text is always appended to the end of the table. Inserting new rows between existing text results in a break of backward compatibility and is definitely a kind of "SF_DONT_DO_THAT"! :-)

The edit area includes the standard edit functionality ("Cut/Copy/Paste/Undo"). But be aware while using the edit function. Unintentional removing or inserting of text between the rows can be result in an unexpected shifting of the rows. This brings strange results when using such a modified "locale" file. Wrong text will be printed.

Loading a reference

A reference assists a user by editing "locale" files. References are comment lines. A reference is created from an existing "locale" file or direct from an application which supports the "sfliblocale.so" library. Every text row is translated in a two line comment:

```
; #ID1:
; #ID1:
; First text line

; #ID2:
; Second text line

; #ID3:
; Third text line
:
:
:
; #IDn:
; n-th. text line
```

The first comment line shows the row number of the table and the second line includes the text of the table row. That all is for simplification to translate the language text. If a new "locale" file is created there only needs to be translated text inserted between the empty lines. If an existing "locale" file is to be edited it's easy to see which parts are to translate:

```
; #ID1:
; First text line
Erste Textzeile
; #ID2:
; Second text line
Zweite Textzeile
; #ID3:
; Third text line
Dritte Textzeile
:
:
:
; #IDn-1:
; (n-1)th text line

; #IDn:
; n-th text line
```

There are two ways to load a reference:

1. Loading an existing "locale" file as a reference. Every "locale" file can be loaded as a reference. To load a "locale" file as a reference use the "Load" button right of the reference line. A file requester opens where any given "locale" file can be selected.
2. Get the texts directly from an application which supports the "sfliblocale.so" library. Every application which supports the "sfliblocale.so" library correctly offer the possibility to get the build-in (english) texts by using the BeOS scripting. To get the build-in texts the Siegfried Locale Editor and the target application need to be started. The popup menu right of reference line shows every running application that supports the "sfliblocale.so" library. By selecting an application the Siegfried Locale Editor gets the complete text data as reference, the name of the language (normally english) and the application ID. The popup menu shows a minimum of one application, the Siegfried Locale Editor itself.

If there was a previously loaded reference it will be replaced by the new one. Existing text will not be deleted. The new reference is inserted between the text.

Creating a new "locale" file

A new "locale" file is created by using an existing "locale" file as reference, or the text data will be grabbed directly from the running application. In both cases the loaded text is shown as a reference comment. In the most cases the second way is the better one, to load the a reference by using the BeOS scripting. By this it's always sure that the loaded data includes all text of an application. If an existing "locale" file is loaded it's possible the file isn't up to date (because the file was delivered with an older version of the application).

To create a new "locale" file 5 steps are necessary:

1. Loading a reference by using the "Application" popup menu or the "Load" button of the reference line. The loaded data creates reference comments and stores the ID of the "locale" file to the application ID field.
2. Enter the name for the language at the field "Language". The name is always written in the native language for the new "locale" file. If, for example, a german "locale" file is to be created, the name to be inserted for the language is "Deutsch" or for a spanish file "Espanol". This name is used later by the application to show the available languages. Pay attention to the upper and lower case for the name. Simple rule: write the first character in upper case and the rest in lower case.
3. Set a version number at the "Version" field. The version should have the format "##.##" incl. trailing zeros (eg. "01.00").
4. Insert the translated text below the reference comments. If there is no translation for a text row, insert the english text. The order of the text is important. Don't change the order otherwise the application will produce unexpected text output results.
5. Save the new "locale" file. By using the "Save" button of the "Language" line a file requester opens. The filename can be set free, but it's recommended to use the language name as file name or the language name in addition with the application name. Where the "locale" is saved depends on the application. Normaly the application uses a folder named "locale" in the same directory where the application is found. This convention isn't binding, but it's recommended.

Edit an existing "locale" file

Editing an existing "locale" file is needed to correct typing errors or to add new text.

To edit a "locale" file use the "Load" button of the "Language" line. A file requester opens and the "locale" file can be selected. The editor loads the text of the file into the edit area. In addition the fields "Language", "Version" and "Application ID" will be set.

It's possible to load a reference after a "locale" file is loaded. To load a reference use the "Load" button or the "Application" popup menu of the reference line. The reference text will be inserted automatically between the text lines. It's all the same if the text or the reference is loaded first. In every case the data is inserted correctly.

To get the newest text, it's recommend to load the refernce by using "Application" popup menu. Scrolling to the end of the text area shows if new text have to add.

Menu

File

The "File" menu contains common functions and commands.

New

The function "new" clears the edit area and all input fields (file and reference name, version and the application ID). "New" is like a fresh start up of the editor.

Help

The documentation for the Siegfried Locale Editor is loaded using the "Help" menu. The documentation can also be viewed using the "<COMMAND> H" short cut.

The Siegfried Locale Editor then opens and shows the HTML documentation. The selected standard browser of the operating system is used as the help viewer.

Short cut: <COMMAND> H

About

This is the most import command of the Siegfried Locale Editor! ;-) The obligatory "About this famous application" window. The window contains the program version, our company and home page address.

Quit

"Quit" terminates the Siegfried Locale Editor. Edited text isn't saved automatically. Important: currently no save request appears if text is changed.

If "AutoPrefs" is set, all current settings of the Siegfried Locale Editor are save to a file (similar to the "Preference/Save" command). At the next start of the Siegfried Locale Editor the settings are loaded automatically.

Shortcut: <COMMAND> Q

Edit

The "Edit" Menu contains functions to edit the text area of the Siegfried Locale Editor. These are the standard funtions (Select, Cut, Copy, Paste, ...) which should be available by every kind of editor.

Undo

"Undo" is used to cancel (undo) the last edit function.

Shortcut: <COMMAND> Z

Cut

Copies the selected text to the clipboard and removes the text from the edit area.

Shortcut: <COMMAND> X

Copy

Copies the selected text to the clipboard . The clipboard is used from other applications, too. Therefore it's possible to use the clipboard to exchange texts with other programs.

Shortcut: <COMMAND> C

Paste

Insert a copied text area from the clipboard at the current cursor position.

Shortcut: <COMMAND> V

Select All

Marks the complete text.

Shortcut: <COMMAND> A

Preferences

The "Preferences" menu contains functions and settings to customize the Siegfried Locale Editor.

Language

The "Language" menu selects the language that is used by the Siegfried Locale Editor. Available languages are displayed in the submenu. The language currently used is selected with a checkmark.

After changing the language only newly created text appears in the new language (e.g. new open windows). All other text (e.g. the main window oder the menu) will remain in the old language. For this reason it is important to restart the Siegfried Locale Editor after selecting a new language.

Before leaving the Siegfried Locale Editor check to see that the settings for the new language are saved to the preferences (to save the preference use the menu "Preferences/Save"). Siegfried Locale Editor saves the settings automatically when exiting if "Autoprefs" is activated, which is the default.

The texts for the respective languages are saved in external files (folder "LocaleEditor/locale"). By simply adding a new language file Siegfried Locale Editor can easily adjust to a new language of a country.

Siegfried Locale Editor is fully functional without the language files. If there are no language files at the folder "LocaleEditor/locale", Siegfried Locale Editor uses the default language (English). If a text is missing in a language file because an older language file is in use, the default text will be used automatically.

Autoprefs

By using the option "AutoPrefs" (default: on), Siegfried Locale Editor automatically saves all settings when the program quits. If the option is activated, Siegfried Locale Editor writes all current settings to a disc when the program quits. The settings are written to the file "LocaleEditor.prefs". The file is loaded automatically when Siegfried Locale Editor starts.

The saving of the settings when the program quits has the same results as the use of the command "Save" of the "Preferences" menu.

Save

The "Save" command writes the current settings of all Siegfried Locale Editor windows to a disc. The data is saved to the file "SiegfriedLocale.prefs". At the next start of Siegfried Locale Editor the file is automatically loaded and the settings restored.
