Internationalization 4

This chapter describes how to internationalize your `gmf` applications.

Overview of Internationalization Concepts

Internationalization is the process of making software portable between languages or *locales*. An internationalized application runs in any locale without changes to the binary. Text strings and other locale-specific information is kept separate from application code in files which can be easily edited.

Localization is the process of adapting software for specific locales. It consists of translating the application's text strings and changing other locale-specific information for a locale. Internationalization is usually performed by the software writer; Localization is usually performed by experts familiar with the specific language or region,

`gmf` supports the `dgettext` method of internationalization; with the `-r` flag, `gmf` generates `bindtextdomain` for application initialization and `dgettext` for all strings.

Levels of Internationalization

There are currently four levels of internationalization. The requirements of each level are described in the sections below.

Level 1—Text and Codesets

Level 1-compliant software is "8-bit clean" and therefore can use the ISO 8859-1 (also called ISO Latin-1) codeset. The ASCII character set uses only 7 bits out of an 8-bit byte. The ISO Latin-1 codeset requires all 8 bits for each character.

Level 2—Formats and Collation

Many different formats are used throughout the world to represent date, time, currency, numbers, and units. Also, some alphabets have more letters than others and the sorting order may vary from one language to another. Level 2-compliant programs leave the format design and sorting order to the localizer in a particular country.

Level 3—Messages and Text Presentation

Text visible to the user on-screen must be easily translatable. This includes help text, error messages, property sheets, buttons, text on icons, and so forth. To assist localizers, text strings can be culled into a separate file, where they are translated. Because the text strings are sorted individually, level 3-compliant software does not contain compound messages—those created with separate `printf` statements, for example—because the separate parts of the message will not be kept together.

Level 4—Asian Language Support

Asian languages contain many characters (1500 to 15000). These cannot all be represented in eight bits and can be laborious to generate using keyboard characters. The EUC (Extended Unix Codeset) is a multi-byte character standard that can be used to represent Asian character sets. EUC does not support 8-bit codesets such as ISO Latin-1.

GMF Support for Internationalization

The current version of gmf supports Level 3 internationalization. gmf makes it easy to internationalize your application by writing out locale-specific resources to a resource file. In some cases, you may need to reposition widgets to accommodate different label lengths. The localizer only needs to edit this file to localize your application.

Generating Code for an Internationalized Application

Using GMF Command Line Options

To generate code for an internationalized application, use the -r option when you run gmf on the GIL or project file for the application:

- -r, which writes *all* resources into a resource file

When you run gmf on an individual file, the generated resource file is named <GIL_filename>.resource. When you run gmf on a project file, the generated resource file is named <project_filename>.resource.

For example, to generate code for a GIL file called myapp.G, you type:

```
% gmf -r myapp
```

This command generates a resource file called myapp.resource.

Generated Resource Files

The resource files gmf generates consist of resource specifications for each widget in the application. The specification for simple widgets has the following form:

```
*<WidgetName>.<ResourceName>: <Value>
```

The specification for composite widgets, such as TextEdit and ScrolledWindow, has the following form:

```
*<WidgetName>.<SubWidgetClass>.<ResourceName>: <Value>
```

To find the name for a widget, see "Getting the Right Widget Instance Name" on page 22.

Level-3 Resources

The sections below list the Level-3 Internationalization-specific resources.

Common Resources

The following are the Level-3 Internationalization-specific resources common to all widgets:

- XmNx
- XmNy
- XmNlabel
- XmNtitle
- There are other resources that are specific to widgets; see the `.resources` file generated when you run `gmf` with the `-r` flag.

Using XFILESEARCHPATH

The `XFILESEARCHPATH` environment variable in conjunction with the `LANG` environment variable helps applications automatically set up locale-specific resource files. The default value of this variable collapses to : `/usr/lib/X11/$LANG/app-defaults/<Class>`, where `Class` is the class of an application. In case of `gmf`-generated applications, the class-name is the name of the application with the first letter capitalized. Thus one would install the suitably localized resource file so that `XFILESEARCHPATH` points to it. Refer to the Xt Intrinsics documentation for more information on specifying resources and installing resource files.