Korean Solaris User's Guide

Sun Microsystems, Inc.
901 N. San Antonio Road
Palo Alto, CA 94303-4900
U.S.A.

Part No: 806–3484–10
March 2000

Adobe PostScript™

011025@2471

Contents

Preface

Korean Solaris User's Guide describes product behavior unique to the Korean Solaris™ operating environment and answers many questions commonly asked during initial experience with the software. This guide introduces the general appearance and properties of a variety of localized Desktop Tools™ and utilities offered with the Korean Common Desktop Environment (CDE) and OpenWindows™ environment.

Who Should Use This Book

This user's guide is for someone who wants to use the Korean features of Solaris software to manage files, calendars, e-mail, write or print Korean files, and so forth. Tools for these and many other applications run under Korean Solaris software. This guide helps you easily find, access, and get started with these tools. You should read this guide:

- If you have not used Korean Solaris software before
- For information on using product features
- If you need instructions for starting up your Korean Solaris operating environment (see especially Chapter 2, "Starting the Korean Solaris Software")

Before You Read This Book

Become familiar with the basics of the Solaris base release user documents, particularly the ones listed under “Related Books” on page xiii. This user’s guide focuses on using the Korean features of the Desktop Tools and other features of Korean Solaris software.

How This Book Is Organized

Each chapter of this guide addresses a different aspect of using Korean Solaris software. The chapters tell how to check your set up before you begin using the facilities of Korean Solaris software and give step-by-step instructions for using Korean facilities.

Chapter 1, “Introduction to Korean Solaris Software,” briefly describes general modifications made to Solaris software, including CDE, to internationalize and localize it for Korean.

Chapter 2, “Starting the Korean Solaris Software,” gives the step-by-step instructions you must follow to start your Solaris user environment. It also describes Korean Solaris-specific features you must use to turn Korean facilities OFF/ON by using dtlogin.

Chapter 3, “Using the `htt` Input Method Server,” introduces the startup, appearance, and use of `htt`.

Chapter 4, “Entering Korean Input,” describes different Korean-character entry modes and provides a step-by-step tutorial in their use. (Further information on customizing commands and other advanced user topics are covered in *Solaris Internationalization Guide for Developers* and *Korean Solaris System Administrator’s Guide*.)

Chapter 5, “Localized Applications,” describes uses of two desktop tools localized for Korean: `mailx`, `talk`, and tools to convert file codes.

Chapter 6, “Font Editor,” explains how to customize fonts used in your Korean Solaris applications.

Chapter 7, “Korean Printing Facilities,” discusses Korean Solaris support for line printers with built-in Korean fonts or using the Korean Solaris `xetops` and `xutops` filter packages.

Chapter 8, "Hanja Tool," describes using Hanja Tool to add, delete, or edit Hanja choices in Hangul–Hanja conversion mode. This viewer is the only tool you can use to read or edit the binary-format Hangul–Hanja dictionary.

Appendix A, "Open Windows Information," describes the special requirements of the OpenWindows environment.

Appendix B, "Binary Compatibility Package," discusses running compiled binary code of earlier SunOS™ 4.x/Solaris 1.x/Asian OpenWindows 2.x applications without recompilation.

Appendix C, "Running Networked Applications," discusses running localized applications that reside on another machine across your network.

Appendix D, "Mapping Korean Keyboard Function," discusses how to configure a Sun Korean keyboard to make selected key functions when you need them.

The Glossary is a list of words and phrases found in the Korean Solaris documentation set and their definitions.

Related Books

You should become familiar with the following basic documentation:

- *Solaris Introduction*
- *Solaris User's Guide*
- *Solaris 8 (SPARC Platform Edition) Asian Release Notes*
- *Solaris 8 (Intel Platform Edition) Asian Release Notes*

Advanced users may want to read *Solaris Advanced User's Guide*. Advanced users wanting to customize their system environment or the operations of their Sun tools will find pertinent information in *International Language Environments Guide* and *Korean Solaris System Administrator's Guide*. These books provide information on setting up, administering, programming, and customizing product features for advanced users, developers/programmers, and systems administrators.

What Typographic Changes Mean

The following table describes the typographic changes used in this book.

TABLE P–1 Typographic Conventions

Typeface or Symbol	Meaning	Example
`AaBbCc123`	The names of commands, files, and directories; on-screen computer output	Edit your `.login` file. Use `ls -a` to list all files. `machine_name% You have mail.`

`AaBbCc123`	What you type, contrasted with on-screen computer output	`machine_name%` **`su`**
		`Password:`

AaBbCc123	Command-line placeholder: replace with a real name or value	To delete a file, type `rm` *filename*.
AaBbCc123	Book titles, new words or terms, or words to be emphasized	Read Chapter 6 in *User's Guide*. These are called *class* options. You *must* be root to do this.

Shell Prompts in Command Examples

The following table shows the default system prompt and superuser prompt for the C shell, Bourne shell, and Korn shell.

TABLE P–2 Shell Prompts

Shell	Prompt
C shell prompt	`machine_name%`
C shell superuser prompt	`machine_name#`
Bourne shell and Korn shell prompt	`$`
Bourne shell and Korn shell superuser prompt	`#`

CHAPTER 1

Introduction to Korean Solaris Software

Korean Solaris software is a Korean localization of the Sun™ Solaris operating environment. Korean Solaris software includes the Korean Common Desktop Environment (CDE) for windowed applications that are built on Sun's Solaris software.

Design of Korean Solaris Software

Korean Solaris software is an extension of base Solaris software. Virtually all utilities and features of the U.S. and International Solaris standard releases are incorporated in Korean Solaris software. These products introduce Solaris input methods for the input and output of Korean. Application programs and CDE use the features of Korean CDE to communicate with users in Korean.

This Korean localization of Sun's internationalized CDE includes enhancements for handling appropriate linguistic and cultural conventions, which it provides to two broad working environments:

- A localized user environment, which includes localized desktop tools and window manager (`dtwm`) that communicate with users in Korean.
- A localized development environment, which programmers use to develop localized applications, with `Xlib` and Motif, which have been internationalized for this use. Programmer and developers should refer to *Solaris Internationalization Guide for Developers*.

Korean Graphical User Interface

This Korean Solaris release uses the Korean CDE Motif graphical user interface, which is similar in layout and design to the U.S. release of CDE. Korean CDE supports multibyte characters and Korean messages with Motif objects. Differences in character width and proportional spacing cause minor differences in the exact layout of some Motif objects.

All application windows that can take Korean input include a *status area* associated with their input window to show the current conversion mode. With an input conversion mode on, as Korean is being typed its entry point becomes a highlighted (reverse video) *preedit area* until the input is converted to Korean or special characters and committed. Some input modes also provide conversion choices among several Korean characters on menus.

Korean Input/Output

To accommodate the diversity of Korean, this Solaris software provides several different input methods for entering Korean characters. With these methods you can enter ASCII/English characters, Korean radicals, and Korean characters using an ASCII keyboard or a Korean keyboard.

Korean input at the keyboard is stored temporarily in an intermediate representation. The conversion manager program, with the help of user interaction, then transforms this intermediate representation into a displayed character string.

The Localized CDE Desktop

The following desktop tools are provided in this Solaris release. All can handle Korean input and output. A man page is provided for each.

Address Manager – Carries out remote operations and finds information about the systems and users on your network. Can speed up such tasks as sending email, logging in remotely, and setting appointments on someone else’s calendar.

Application Manager – Contains the applications available on your system. You can launch these applications through the Application Manager interface.

Audio Tool – Tool for recording, playing, editing, and controlling workstation audio parameters.

Calculator – Visual calculator for use with mouse or keyboard.

Calendar Manager – Manages business and social appointments; can use electronic mail to send automatic reminders.

Clock – Displays current analog or digital time.

Console – Standard Motif scrolling window terminal emulator.

File Finder – Tool for searching for a folder and subfolders that returns a list of files or folders that match your search criteria. You can also specify the size, owner, date modified, type and permissions in your search criteria.

File Manager – Graphical tool for accessing files and directories. Represents file types with varying colors and icons. Navigates through the file system with the mouse.

Front Panel – A centrally-located window containing controls for accessing applications and utilities, including the workspace switch. The Front Panel occupies all workspaces.

Help – On-line searchable help for CDE.

Icon Editor – Visual tool for editing icon appearance and creating new icons.

Image Tool – Interactive image viewer. Image Tool can be used to view the contents of file types such as GIF, TIFF, JPEG, PostScript, and others.

Mailer – Tool for handling electronic mail.

Performance Meter – Real-time system performance meter that can display a variety of data.

Print Manager – Graphical front-end to the print command. It supports *drag-and-drop* file transfer operations.

Process Manager – Tool for displaying and performing actions on the processes that are currently running on your workstation.

Snapshot – Tool to *snap* or capture picture of a window or region of a screen in a bitmap (raster file). Used for capturing screen image displays in this user's guide.

Text Editor – Visual text editor used in CDE tools such as the Mailer composition window.

Style Manager – Tool for setting workstation preferences, such as audio feedback from keyboard, mouse response, and so on.

Terminal – Standard Motif window terminal emulator. The window behaves like an ASCII character terminal for entry of UNIX® commands at a system shell prompt and other terminal operations.

CHAPTER **2**

Starting the Korean Solaris Software

The Korean Solaris operating environment must be specially set up for using Korean text facilities. This chapter describes the steps required to set up the Korean environment and to start Korean Solaris operation.

The Korean Solaris operating environment provides two window environments, CDE and the OpenWindows environment. CDE is a fully internationalized environment; it does not require most of the administration tasks that the OpenWindows environment requires to handle Korean. For information on starting up OpenWindows, see Appendix A, "Open Windows Information." The Korean Solaris product includes three locales:

- `C` – ASCII English environment
- `ko` – Korean extended UNIX code (EUC). This locale supports the KS C-5601–1992.3 standard. KS C 5601–1992.3 is based on ISO-2022; Extended UNIX Code is a reformatting of ISO-2022. You may find it more convenient to run this Korean locale if you use current Korean Solaris applications.
- `ko.UTF-8` – Korean Universal Multiple Octet Coded Character Set (UCS) Transmission Format. This locale supports the KS C-5700 standard, which was announced by the Korean government in December 1995. If you are a developer or advanced user and need access to a larger number of characters, Unicode, or KS C 5700 support, run this locale.
- `ko_KR.EUC` – This is the same as `ko` locale. A symbolic link.
- `ko_KR.UTF-8` – This is the same as `ko.UTF-8` locale. A symbolic link.
- `korean` – This is the same as `ko` locale. A symbolic link.

Note – OpenWindows does not support `ko.UTF-8, ko_KR.EUC, ko_KR.UTF-8` or `korean` locale.

Setting the Default Locale

You can change your default locale with the following procedure. In the OpenWindows environment, a setting of `ko.UTF-8` reverts to the `C` locale.

1. **Choose the language button on the** `dtlogin` **window.**
2. **Select the** `C`**,** `ko`**, or** `ko.UTF-8, ko_KR.EUC, ko_KR.UTF-8` **or** `korean` **locale.**

   Your new locale is in effect.

Using Aliases for Setting Locale

You can use aliases to change a terminal-emulation window between the Korean locales and ASCII/English locale from time to time without typing long command lines or editing your `.dtlogin` file and running `source` every time.

Setting up Locale Aliases for the Korn Shell

Setting up Locale Aliases for the C Shell

Changing Font Directories

The `Xsession` script (located in `/usr/dt/bin`) that comes with the Korean Solaris operating environment includes the following font path: `/usr/openwin/lib/locale/`*locale*`/X11/fonts`, where *locale* is either `ko` or `ko.UTF-8`. To add a different font directory path dynamically, type:

```
system% xset +fp font_directory-path
```

CHAPTER 3

Using the `htt` Input Method Server

The `htt` input method server handles Korean input for Korean Solaris software. The `htt` server receives keyboard input and makes the input available as Korean characters to Korean Solaris system applications. The `htt` server can serve any internationalized X Window application that uses X Windows Input Method (XIM) application program interfaces (API) to receive language input.

A new Internet Intranet Input Method Server (`iiim`) has also been implemented in Solaris 8 for all UTF-8 locales. It supports both European Local Input Methods using Compose key and Asian Remote Input Methods using IIIM Protocol.

This chapter explains the basic display features of the `htt` input server and the IIIM server. Normally, you do not need to change any `htt` settings, as discussed in this chapter, to operate any Korean Solaris application.

You need to be familiar with the input method terminology in Chapter 4, "Entering Korean Input," before reading this chapter. However, to use Solaris applications for Korean character input you do not need to read beyond the first section of this chapter.

The Korean Solaris `htt` server icon looks like this:

Input Method Server Basic Properties

The Korean Solaris operating environment starts the `htt` input method server automatically when you start up the Solaris operating environment in a locale that requires an input method. htt continues to run and service applications that start up and connect to it. If you need to restart htt, refer to "How and When htt is Started," on page 15.

So each Korean Solaris application that uses htt for Korean character input typically finds htt running when it starts up. To get htt's service for language input, htt should be running before an application is started. If an application does not find htt running when it starts up, that application may not be able to get the input service even if htt is started later.

Three htt Processes

The input method server comprises three related programs (htt_props, htt, htt_server) processes running together. One process controls the input method server properties, one process controls the population of the input method server (i.e., a watch dog process), and the third handles input methods of clients.

Usually, you don't need to know the details of these three processes. When htt is started, htt_props and htt_server are started automatically.

htt Property Manager

The htt input method server is started as an icon. Double click on the icon to start htt Property Manager to configure the behavior of htt.

Resetting and Terminating htt

The input method server property manager includes the following menus:

File Menu

Reset Input Manager – resets and restarts the htt input method server. This operation is not needed unless the input method server requires resetting (for example, when an application stops getting the language input). Selecting this operation destroys any intermediate data (such as preedit texts).

Exit – terminates `htt`. When you select this operation, `htt` Property Manager offers three options.

- Exit – terminates `htt` completely. If you need to restart `htt`, start it from a shell window:

  ```
  system% htt &
  ```

- Background – terminates `htt` Property Manager only. The `htt` icon and `htt` Property Manager window are no longer visible. However, the `htt` input method server still functions. If you need to start `htt` Property Manager again, start `htt_props` from a shell window.

  ```
  system% htt_props &
  ```

- Cancel – cancels the termination.

Help Menu

The `htt` Property Manager provides four categories of help messages to guide its operation:

- General – describes the operations in the general dialog
- Preedit Status – describes the operations in the Preedit/Status dialog
- Lookup Table – describes the operation in the Lookup table dialog
- About – describes the menu operations from the main Property Manager window

Using `htt` Command Line Options.

`htt`'s command line options can be referenced from its man page or by typing:

```
system% htt -help
```

Customizing the on/off Key from General Dialog

You can customize the key sequence to turn on/off locale specific input. To add a key sequence, perform the following steps:

1. **Press the Add button.**

   You will see the Add Key popup.

2. **Press the key to be added and OK.**

   The key pressed is added to the conversion key list.

   To remove any key from the list:

- **Select the key from the list and press Remove.**

Note – The customized key sequence will not take effect until you restart `htt`.

Customizing the Preedit/Status Window from the Preedit/Status Dialog

You can control the Preedit/Status window placement in this panel if your application specifies the root window style Preedit or Status.

- Selecting "Position on the screen" places the window in a fixed location of the workspace. This window is shared by multiple applications whose input style is root window style.
- Selecting "Position relative to the cursor" places the window close to the mouse cursor.
- Selecting "Attach to client input window" places the window near the application's input window.

In addition to the placement of the Preedit/Status window for the root window style, you can configure the behavior of the preedit string when it reaches to the end of a line. In the default, the preedit string will wrap around to the second line. But if you de-select "Wrap long lines in on-the-spot preedit," the preedit string will not exceed one line and it will scroll when reaching the end of a line.

Customizing the Lookup Window from the Lookup Table Dialog

You can control the appearance of the lookup window by choosing one of the four options listed:

- Client window status area – places the lookup choices in an area where the input status is shown on the application's window
- Preedit/status window – places the lookup choices in the window placed on the workspace, which is used by root window style applications
- Popup window near cursor – places the popup near to the current position of mouse
- Popup window near client input window – places the popup near the application's input window

You can also specify the maximum rows and columns for the lookup choices that can be displayed on a single popup. You can specify the labels used for lookup choices. The options are upper case or lower case alphabets or numbers.

How and When `htt` is Started

An application that uses the `htt` input method server to receive Korean characters must find `htt` running before it can receive such input. So if your default locale is the `C` locale (which does not require `htt`) and you then change from `C` to another locale in a single Terminal, for example, you must start `htt` with a command line like the following:

```
system% htt &
```

With the locale set to `ko` or `ko.UTF-8`, `htt` is started automatically when the Korean windowing environment starts up. In CDE, `htt` is started from a script, `/usr/dt/config/Xsession.d/0020.dtims`. This script is executed by `Xsession`, which runs at every initialization stage upon a user's login from `dtlogin`. This script ensures that `schtt` is started before other applications in CDE.

Input Method Server Basic Properties

The Korean Solaris operating environment starts the `htt` input method server automatically when you start up the Solaris operating environment in a locale that requires an input method. `htt` continues to run and service applications that start up and connect to it. If you need to restart `htt`, refer to "How and When `htt` is Started," on page 15.

So each Korean Solaris application that uses `htt` for Korean character input typically finds `htt` running when it starts up. To get `htt`'s service for language input, `htt` should be running before an application is started. If an application does not find `htt` running when it starts up, that application may not be able to get the input service even if `htt` is started later.

Using the `iiim` Server in `ko.UTF-8` and `ko_KR.UTF-8` Locales

The Internet Intranet Input Method Server (`iiim`) supports both European Local Input Methods using the Compose key and Asian Remote Input Methods that use IIIM Protocol. The default Language Engine is English/European which supports ASCII and some European Languages such as German and French.

To switch to other Language Engines, press CTRL+Space, as with other Asian Locales.

To switch to Korean Input Methods, click the left mouse button in the Status Area. The following Language Engine List appears:

- Cyrilllic
- Greek
- Thai
- Arabic
- Hebrew
- Unicode Hex
- Unicode Octal
- Lookup
- Korean

The following Language Engines appear if their corresponding locales are installed:

- Japanese — if `ja` locale is installed
- Simplified Chinese — if `zh.UTF-8` locale is installed.

- Traditional Chinese — if `zh_TW` locale is installed.

After the Korean Language Engine is selected, you can switch between different Korean Input Modes in the same way as with the `ko` locale and `ko.UTF-8`locale in Solaris 7. For more detailed information about each Korean Input Mode, see Chapter 4, "Entering Korean Input".

CHAPTER 4

Entering Korean Input

This section describes Korean Solaris mechanisms for typing Korean characters. All of the following character types can be entered:

- ASCII/English
- Hangul
- Hanja
- Special symbols

All of these character types can be entered in input regions of various application subwindows as follows:

- Terminal emulation (TTY) windows, such as a Terminal
- Text entry subwindows, such as those used by Text Editor or Mailer
- Control panel subwindows, such as those used by File Manager for typing a file name
- Other special use subwindows, such as pop-ups

Character Sets

Four types of coding conventions are currently supported in the Korean Solaris software:

- N-byte code - This single-byte code has each byte represent a consonant or vowel. These are combined together to build Hangul characters.
- Johap or Packed code - This two-byte code consists of a leading bit followed by three 5-bit fields. These three fields contain the codes or a leading consonant, followed by a vowel, followed by a final consonant (if any) for a Hangul character. This two-byte code is specified in Korean Industry Standard KS C 5601-1992–3.

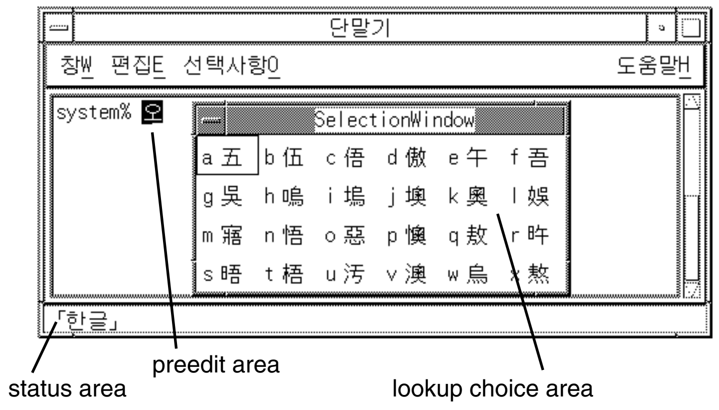
- Wansung code - This two-byte code is specified in Korean Industry Standard KS C 5601-1987 for Hangul, Hanja, and other characters. In the Korean Solaris software these KS C 5601-1987 characters are in EUC codeset 1.
- `ko.UTF-8` - Korean Universal Multiple Octet Coded Character Set (UCS) Transmission Format. `ko.UTF-8` supports all the characters of KS C 5601 and 11,172 characters from Johap, as well as all Korean-related Unicode 2.0 characters and fonts. `ko.UTF-8` supports the following subset of Unicode:
  - Basic Latin and Latin-i (190 characters) - Row 00 of BMP (Basic Multilingual Plan)
  - Symbolic characters - Row 20 to Row 27, and Row 32 of BMP including box (line) drawing characters that are defined in KS C 5601
  - Numerals that are defined in KS C 5601 (20 characters) - Row 21 and Row FF of BMP
  - Roman, Greek, Japanese, and Cyrillic alphabet characters that are defined in KS C 5601 (362 characters) - Row 02, Row 04, Row 30 and Row FF of BMP
  - Jamo (Hangul alphabet) characters (94 characters) - Row 31 of BMP
  - Pre-composed Hangul syllables (11,172 characters) - From Row AC to Row D7 of BMP
  - Hanja characters defined in KS C 5601 (4,888 characters) - From Row 4E to Row 9F and from Row F9 to Row FA of BMP

These four Korean code conventions Chinese code conventions at three levels of support:

- User commands support file transfers for existing files in different codes.
- Library functions support application development for existing codes.
- STREAMS modules support existing TTY devices using different codes.

Input Window Areas

Three separate areas of an application subwindow are used in entering characters. These areas are typically displayed, named, and used as follows:


단말기
창W 편집E 선택사항O 도움말H
system% 오
SelectionWindow
a 五 b 伍 c 俉 d 傲 e 午 f 吾
g 吳 h 嗚 i 塢 j 墺 k 奧 l 娛
m 寤 n 悟 o 惡 p 懊 q 敖 r 旿
s 晤 t 梧 u 汚 v 澳 w 烏 x 熬
「한글」
preedit area
status area
lookup choice area

- Preedit area—Highlighted (inverse video) entry display
- Status area—Lower left corner input conversion mode display
- Lookup choice area—Popup displaying multiple character or word choices

Preedit Area

The preedit area is a reverse video highlighted area that displays characters as they are entered or converted. It holds formations of characters before they are committed to the application, that is, before they are put in the text block being assembled for the application.

Status Area

The status area shows what the current input conversion mode is. Several modes are available, as discussed in following sections.

Lookup Choice Area

The lookup choice area is a popup that displays multiple Hanja or special character choices available for conversion of character(s) in the preedit area.

Entering and Converting Hangul/Hanja Characters

Each character type listed at the beginning of this chapter has its own input mode and input procedures, as described in following sections. The Control-key command combinations (initial default settings) for turning these modes and conversions on/off are listed in the table at the end of this chapter. The Back Space and Delete keys erase input (with peculiarities noted in the section "Backspacing and Deleting Characters" on page 31.)

ASCII/English Input Mode

The initial default setting is basic ASCII/English mode. Typing on the keyboard simply enters ASCII characters.

The status region in the lower left corner shows ASCII/English input mode:

Hangul Input Conversion Mode

The first step in entering any Korean character(s) is to turn on Hangul input conversion. Type Control-Space, or if you have a Hangul keyboard type the key labelled Hangul/English. (These keys toggle on/off Hangul input conversion.)

The status region then shows Hangul conversion on:

With Hangul input conversion on, typing appropriate sequences of keys composes Hangul characters in the preedit area on the screen.

Committing Hangul Input

The preceding preedit region contains five Hangul syllables that have not yet been *committed*. That is, they have not yet been added to the text block being assembled for the application.

These syllables could be committed to the application by typing Control-K. This nonprinting character is not committed in the input along with the syllables, for example:

Otherwise these syllables could be committed by typing the Space bar or a number, punctuation mark, or other printable character that is not part of a valid Hangul character. Unlike Control-K, a printable character (like a question mark) is committed in the input along with the Hangul syllables, for example:

Converting to Hanja

Converting Hangul input to Hanja can be done in several ways, including character by character or word by word. The following examples and instructions go through the available ways step by step.

Hanja Character Step Mode

1. **With Hangul input conversion on, type an appropriate sequence of keys to compose a Hangul character in the preedit area on the screen:**

2. **Type Control-N to convert it to a possible Hanja choice.**

   Typing Control-N again and again converts the Hangul character through the series of possible Hanja choices, one at a time (until the original Hangul reappears after all Hanjas have been displayed). Control-P converts back to the previous choice.

Committing a Hanja Choice

To choose and commit only the highlighted Hanja, type Control-K or any other nonprinting character except Control-N or Control-P. Then a new preedit area next to the just-committed Hanja is ready for the next Hangul input.

To commit the Hanja plus a space, number, punctuation mark, or other printable character, type the character's key. It and the highlighted Hanja are then both committed and a new preedit area next to the just-committed space, number, etc. is ready for the next Hangul input.

Hanja Character Lookup Choice Mode

1. **With Hangul input conversion on, typing an appropriate sequence of keys composes a Hangul character in the preedit area on the screen:**

2. **Typing Control-W or the Hanja key on the keyboard displays a lookup choice area that contains possible Hanja choices:**

3. **Typing Control-N displays the next lookup choice page if more choices are available. Control-P displays the previous page of choices.**

Committing or Rejecting Hanja Choices

The chosen Hanja (for example, choice `a`) is then committed, and a new preedit area is ready for the next Hangul input:

Hanja Word Conversion Modes

Hangul to Hanja conversion can be done also word by word just as it is done character by character (described in the preceding section). The steps for Hangul to Hanja word conversion are the same except conversion is not turned on until the preedit area contains a Hangul word.

Control-N (step mode) or Control-W (lookup choice mode) turns on Hanja conversion choices. The Hanja choices are a list of possible Hanja words plus a list of Hanja characters for only the last syllable of the Hangul word.

A display like the following appears after typing a Hangul word followed by Control-W:

The above list contains three Hanja word choices and several choices for converting only the last syllable of the Hangul word.

The display for some Hangul words contains only single character Hanja choices for the last Hangul syllable. This happens when the Hangul-to-Hanja dictionary contains no corresponding Hanja word:

Entering Special Symbol Characters

Many non-Hangul/Hanja characters that cannot be typed directly on the keyboard can nevertheless be entered in the Korean Solaris operating environment. Two input modes very similar to the input modes described in preceding sections are available: hex code input and lookup choice.

Hex Code Input Mode

Any character with a hexadecimal code listed in KS C 5601 can be entered directly as follows. The KS C 5700 standard is supported by UTF-8 hexidecimal code. (The `ko` locale is used in the following example.)

1. **Make sure Hangul input conversion is on:**

2. **Then toggle hex mode on by typing Control-X:**

The only keys that have any effect at this point are delete keys, letters a through f, numbers 0 through 9, and Control-X (which toggles hex mode off again).

1. **Type the first three keys of the hex code of the character, for example `a2dd`, in the preedit area.**

   The screen appears as follows after the first three numbers are typed:

2. **Type `d`.**

   The designated character is then displayed and automatically committed after the fourth number is typed:

Hex mode remains on until toggled off by Control-X.

Special Lookup Choice Mode

This mode is another way to enter non-Hangul/Hanja characters that cannot be typed directly on the keyboard. It works the same as “Hanja Character Lookup Choice Mode” on page 23:

- Typing Control-N or Control-P pages forward or back through the lookup choice area display
- Typing the letter of the choice makes/commits the choice and returns to Hangul input mode
- Typing Control-A at any time aborts the current lookup choice pages without choosing and returns to Hangul input conversion mode.

1. **With Hangul input conversion on, type Control-J to display the page of category choices:**

2. **Type the letter of a category (`a` in the following example) to display the first page of choices:**

   After Control-N has been typed nine times the following is displayed:

3. **Type `k` to choose/commit the displayed character and return to Hangul input mode:**

Backspacing and Deleting Characters

Backspace and Delete keys delete characters. But using one in the preedit area deletes only the last character (jamo) in the current syllable. This and following examples demonstrate Backspace/Delete operation.

1. **Note the Hangul input mode preedit area appearance before backspacing:**

2. **Enter one backspace.**
   The input appears like this:

3. **Type another backspace.**
   The input appears like this:

4. **Type another backspace.**
   The input appears like this:

5. **Type two backspaces.**
   At this point an entire syllable has been deleted.

Korean Solaris Function Keys Default Settings for Input Conversion Modes

Default Key	Function Description
Hangul Input Conversion Mode:	
Control-Space	Toggle Hangul input conversion mode on/off.
Control-K	Commit only the character(s) in the preedit area.

Korean Solaris Function Keys Default Settings for Input Conversion Modes	
Hanja Conversion, Step Mode:	(Hangul input conversion mode must be on.)
Control-N	Convert Hangul input to Hanja and display the first Hanja choice.
Control-N	Display the next Hanja choice.
Control-P	Display the previous choice.
Hanja Conversion, Lookup Mode:	(Hangul input conversion mode must be on.)
Control-W	Display the first group of Hanja conversion choices in the lookup choice area.
Control-N	Display the next group of choices.
Control-P	Display the previous group of choices.
Control-A	Abort conversion without making a conversion choice.
Special Symbol Input Lookup Mode:	(Hangul input conversion mode must be on.)
Control-J	Display the lookup choice area menu of special symbol categories.
Control-N	Display the next group of choices.
Control-P	Display the previous group of choices.
Control-A	Abort conversion without making a conversion choice.
Hex Input Mode:	(Hangul input conversion mode must be on.)
Control-X	Toggle hexadecimal input mode on/off.

CHAPTER **5**

Localized Applications

This chapter describes selected properties you need to use on two localized applications of the Korean Solaris operating environment. This chapter also provides lists of code conversion utilities.

These tools (and the commands to invoke them) include:

- `mailx` (`/usr/SUNWale/bin/mailx`)
- `talk` (`/usr/SUNWale/bin/talk`)

Using the `mailx` Utility with Korean Characters

The *encoding* variable in `.mailrc` does not affect Mailer behavior, but it does set `mailx` function. To send e-mail in formats other than 7-bit ASCII, such as to send (8-bit) Korean characters you must use `/usr/SUNWale/bin/mailx`. The *encoding* variable in `.mailrc` sets encoding formats for `mailx`.

International transmission conventions require that header information use only ASCII characters. So Korean characters should not be used in the header (including the Subject line) with Mailer or with `mailx`.

Note – The `/bin/mailx` application has not been localized and cannot send or receive Korean characters.

Using `talk` with Korean Characters

To use Korean characters with `talk`, the `/usr/SUNWale/bin/talk` application is required because `/bin/talk` is not localized.

The `xtobdf` Utility

Korean Solaris software provides this BDF font generator:

- `xtobdf` — Convert from font in X server to font in BDF

The `Sdtconvtool` Utility

Sdtconvtool is a graphic user interface utility that enables file conversion between various code sets. Its functionality is similar to `iconv`.

The following steps show how to convert a file encoded in UTF-8 to ko_KR-euc encoding:

1. **Select the code set of the file to be converted.**

   Click on the arrow button to the right of the "Source Code Set" label to reveal a list of available code sets in the system. Scroll through the list and select the code set of the file to be converted. In this case, select "UTF-8."

2. **Type the path of the file to be converted.**

   The path to the file can either be entered manually in the "Source File Path" area, or chosen by selecting the "Browse..." button and selecting the file name from the file selection box. In this case, enter or select `/tmp/ko_KR.UTF-8`.

   The "Clear" button to the right of the "Browse..." button can be used to erase the entered source or path.

3. **Select the code set to which the file will be converted.**

   Select the target file code set information from the pull-down menu to the right of the "Target Code Set:" label. In this case, select "ko_KR-euc."

4. **Type the path to which the converted file will be saved.**

   The path to the converted file can either be entered manually in the "Target File Path" area. In this case, enter or select `/tmp/ko-euc.txt`.

   The "Clear" button to the right of the "Browse..." button can be used to erase the entered target file path.

5. **Select "Start Conversion".**

   Pressing the "Start Conversion" button will perform the conversion with the given information.

   The "Clear All Fields" button erases the source and target file path names.

The `iconv` Utility

The `iconv` command converts the characters or sequences of characters in a file from one code set to another and writes the results to standard output. Korean Solaris software includes special filters for the `iconv` command.

If no conversion exists for a particular character, it is converted to the underscore "_" in the target codeset. The following options are supported:

- -f from code - Symbol of the input code set.
- -t to code - Symbol of the output code set.

The following code set conversion modules are supported in Korean Solaris software:

TABLE 5–1 Korean `iconv` Code Conversion Modules (`ko` locale)

Code	Symbol	Target Code	Symbol
Wansung	ko_KR-euc	Johap	ko_KR-johap92
Wansung	ko_KR-euc	Packed	ko_KR-johap
Wansung	ko_KR-euc	N-Byte	ko_KR-nbyte
Wansung	ko_KR-euc	ISO-2022-KR	ko_KR-iso2022-7
Johap	ko_KR-johap92	Wansung	ko_KR-euc
Packed	ko_KR-johap	Wansung	ko_KR-euc
N-Byte	ko_KR-nbyte	Wansung	ko_KR-euc
ISO-2022-KR	ko_KR-iso2022-7	Wansung	ko_KR-euc

The following modules perform character-based code conversion on the KS C 5700 character set. They convert KSC 5700 characters between Korean UTF-8, Completion code (Wansung), and Combination code (Johap).

TABLE 5–2 Common Korean `iconv` Code Conversion Modules (`ko` and `ko.UTF-8` locales)

Code	Symbol	Target Code	Symbol
UTF-8	ko_KR-UTF-8	Wansung	ko_KR-euc
UTF-8	ko_KR-UTF-8	Johap	ko_KR-johap92
UTF-8	ko_KR-UTF-8	Packed	ko_KR-johap
UTF-8	ko_KR-UTF-8	ISO-2022-KR	ko_KR-iso2022-7
Wansung	ko_KR-euc	UTF-8	ko_KR-UTF-8
Johap	ko_KR-johap92	UTF-8	ko_KR-UTF-8
Packed	ko_KR-johap	UTF-8	ko_KR-UTF-8
ISO-2022-KR	ko_KR-iso2022-7 UTF-8	UTF-8	ko_KR-UTF-8
UTF-8	ko_KR.UTF-8	Unified Hangul	ko_KR-cp949
Unified Hangul	ko_KR-cp949	UTF-8	ko_KR-UTF-8

In the following example, a KS C 5601 file (Korean EUC) is converted to KS C 5700 (`ko.UTF-8`):

```
system% iconv -f ko_KR-euc -t ko_KR-UTF-8 ko_euc_file > ko_UTF-8_file
```

For further information, see the *iconv(3)*, *iconv_ko.UTF-8(5)*, and *iconv_utf(5)* man pages. These utilities can be used for converting files for printing. See *Korean Solaris System Administrator's Guide* for more information.

CHAPTER 6

Font Editor

This chapter describes the PostScript fonts included in the Korean Solaris operating environment, what you need to use them, and how to edit them.

This chapter also describes how to work with Portable Compiled Format (PCF) fonts and edit existing characters or create and install new characters used by the Korean Solaris operating environment. You can then save your changes and use the edited font(s) on your local system. This process is a sequence of several steps:

1. Preparing a workspace for the font(s) you edit or create
2. Editing Bitmap Distribution Format (BDF) font file(s) with Font Editor
3. Converting BDF font file(s) to PCF format
4. Making the font(s) usable on your system

Note – `ko.UTF-8` messages are not supported by Font Editor. If you run Font Editor under the `ko.UTF-8` locale, the Font Editor interface will be in English.

Display PostScript System (DPS)

The Korean Solaris operating environment provides PostScript fonts in the Display PostScript System (DPS). This section describes what you need to use DPS in Korean Solaris software. For further details, see *Programming the Display PostScript System with X*, published by Adobe Systems.

Using Korean PostScript Fonts and DPS Facilities

The Korean Solaris operating environment DPS provides the fonts listed in the following table.

TABLE 6–1 Korean Solaris Operating Environment DPS Fonts

Font Name	Description
Kodig-Medium-COMB-H	Kodig-Medium font, 9/7 composite font encoding for horizontal display of Johap encoded Hangul and Roman text.
Kodig-Medium-COMB-V	Kodig-Medium font, 9/7 composite font encoding for vertical display of Johap encoded Hangul and Roman text.
Kodig-Medium	Kodig-Medium font, an alias of Kodig-Medium-EUC-H font; can be used like a Roman font.
Kodig-Medium-EUC-H	Kodig-Medium font, 9/7 composite font encoding for horizontal display of EUC text, can be used like a Roman font.
Kodig-Medium-EUC-V	Kodig-Medium font, 9/7 composite font encoding for vertical display of EUC text; can be used like a Roman font.
Kodig-Medium-H	Kodig-Medium font, 8/8 composite font encoding for horizontal display of shifted out ISO2022 text.
Kodig-Medium-V	Kodig-Medium font, 8/8 composite font encoding for vertical display of shifted out ISO2022 text.
Myeongjo-Medium-COMB-H	Myeongjo-Medium font, 9/7 composite font encoding for horizontal display of Johap encoded Hangul and Roman text.
Myeongjo-Medium-COMB-V	Myeongjo-Medium font, 9/7 composite font encoding for vertical display of Johap encoded Hangul and Roman text.
Myeongjo-Medium	Myeongjo-Medium font, an alias of Myeongjo-Medium-EUC-H font; can be used like a Roman font.
Myeongjo-Medium-EUC-H	Myeongjo-Medium font, 9/7 composite font encoding for horizontal display of EUC text; can be used like a Roman font.
Myeongjo-Medium-EUC-V	Myeongjo-Medium font, 9/7 composite font encoding for vertical display of EUC text; can be used like a Roman font.
Myeongjo-Medium-H	Myeongjo-Medium font, 8/8 composite font encoding for horizontal display of shifted out ISO2022 text.
Myeongjo-Medium-V	Myeongjo-Medium font, 8/8 composite font encoding for vertical display of shifted out ISO2022 text.

You can use the following Korean fonts just as you would use Roman fonts:

- Kodig-Medium-EUC-H

- Kodig-Medium-EUC-V
- Kodig-Medium
- Myeongjo-Medium-EUC-H
- Myeongjo-Medium-EUC-V
- Myeongjo-Medium

The following figure shows a sample of Kodig-Medium and Myeongjo-Medium text.

You can also use the following Korean fonts just like Roman fonts for an ISO2022 encoded Hangul string, that is, for a pure Hangul string between SO and SI characters with no intermediate ASCII (0x20) characters:

- Kodig-Medium-H
- Kodig-Medium-V

Creating Composite Korean Fonts

You can create composite fonts using one Roman font and one of the following Korean fonts:

- Kodig-Medium-COMB-H
- Kodig-Medium-COMB-V
- Myeongjo-Medium-COMB-H
- Myeongjo-Medium-COMB-V

For example, the following PostScript code defines a sample composite font, Times-Italic+Kodig-Medium, which uses Times-Italic for ASCII characters and Kodig-Medium horizontal font for Korean characters:

```
/Times-Italic+Kodig-Medium
13 dict begin
                /FontName 1 index def
                /FMapType 4 def
                /Encoding [ 0 1 ] def
                /WMode 0 def
                /FontType 0 def
                /FontMatrix [1.0 0.0 0.0 1.0 0.0 0.0] def
                /FDepVector [
                                /Times-Italic findfont
                                /Kodig-Medium-COMB-H findfont
                ] def
currentdict
end
definefont pop
```

Using Korean Fonts in DPS Programming

You can use Korean fonts just as you use Roman fonts in DPS wrap definitions. The following sample code creates the display which follows:

```
defineps PSWDisplayText(char *text)
            /pointSize 50 def
            /Helvetica pointSize selectfont
            (Hello World) stringwidth pop 2 div neg 0 moveto
            (Hello World) show

            /cpSize 40 def
            /Kodig-Medium-KO cpSize selectfont
            (text) stringwidth pop 2 div neg pointSize neg moveto
            (text) show
endps
```

You can tell `PSWDisplayText` (*Korean text*) in a C program to display the designated Korean text:.

Using the Font Editor

Setting Up in a Working Directory

1. **Create a new directory workspace for editing fonts, for example:**

```
system% mkdir /tmp/newfont
```

2. **Change your working directory to that directory:**

```
system% cd /tmp/newfont
```

Starting Up the Font Editor

Korean Solaris software provides the Font Editor for editing fonts.

- **Start up Font Editor:**

```
system% fontedit
```

A Font Editor window is displayed.

Font Editor does not operate on PCF format font files (file extension `.pcf`), which are used by the Solaris operating environment. This tool handles only fonts in BDF, a portable format defined by the MIT X Consortium.

Preparing a Font File

- **Get the font file you plan to edit in BDF format.**

Its encoding should start at 8481 (0x2121). For an example, consider that you have a BDF file like the following `myfont14.bdf`:

```
STARTFONT 2.1
COMMENT Sample Font
FONT Myfont-Medium14
SIZE 14 75 75
...
STARTCHAR C101
ENCODING 8481
...
```

Editing a BDF Font File

1. **Hold down the MENU mouse button on the Font Editor window's File button and select Load... on the pull-down menu.**
2. **Type the name of the BDF file you plan to edit, for example** `myfont14.bdf`**, in the Font Name field of the Load pop-up window.**
3. **Click SELECT on the Load button in the popup.**
   This loads the file.
4. **Click SELECT on the Font Editor main window Select button.**
   This displays the Select window, where you do the following steps.

5. **Type the high and low bytes of the font encoding code for the character you want to edit, for example 0x4751.**

   You can use the Next (right arrow) button or Previous (left arrow) button to increase or decrease the high or low byte.

6. **Set Code Length to MultiByte for this Korean font.**

   You would set Code Length to SingleByte for an ASCII/English font.

7. **Click on the Edit button (in the Fontedit: Select window) to display the glyph on the main canvas.**

   The specified glyph appears in the Font Editor window, resembling the following:

8. **Edit this glyph on the main canvas by turning pixels on/off:**
   - Click the SELECT mouse button to turn on a pixel.
   - Click the ADJUST mouse button to turn off a pixel.
   - Hold down the MENU mouse button for a menu of additional functions.

Note – The character glyph in the upper left corner of the main window shows the actual appearance of the glyph as you turn its pixels on and off.

---

9. **When you finish editing the glyph, click SELECT on the Store button.**

10. **Repeat the preceding steps 5 through 9 to edit each glyph you choose in this font.**

11. **After you finish editing the glyph(s) in this font, hold down MENU on the File button and select Save... to save the edited glyph(s) in the font file.**

Converting BDF to PCF Format

Before Solaris applications can use the modified BDF file, it must be converted to an PCF format file.

- **Use the** `bdftopcf` **command to make the BDF font file usable by the Korean Solaris operating environment by converting it to PCF format as follows:**

```
system% bdftopcf -o myfont14.pcf myfont14.bdf
```

For more information, see the *bdftopcf(1)* man page and the *mkfont(1)* man page.

Installing and Checking the Edited Font

1. **To add a new bitmap to the Solaris operating environment, put the** `.pcf` **font file in your font directory. You may compress the** `.pcf` **font file before moving it to your font directory:**

```
system% compress myfont14.pcf
```

2. **Run the following commands in your font directory.**

Note that the `.bdf` file should not be in the font directory.

```
system% cat >> fonts.alias
-new-myfont-medium-r-normal--16-140-75-75-c-140-ksc5601.1987-0
Myfont-Medium14
^D
system% mkfontdir
system% xset +fp 'pwd'
```

3. **You can view your font by entering:**

```
system% xfd -fn Myfont-Medium14
```

CHAPTER 7

Korean Printing Facilities

The Korean Solaris operating environment supports printing Korean output through the following two types of printing facilities:

- Line printer containing built-in Korean fonts
- PostScript-based printer

Note – Before you can print Korean text, a system administrator must set up your Korean printing support as described in *Korean Solaris System Administrator's Guide*.

You can use the `xetops`, `xutops` and `mp` utilities to print files containing Korean text on a PostScript printer.

- `xetops` is used with `ko` files
- `xutops` is used with `ko.UTF-8` files
- `m` is used with `ko.UTF-8` files.

These printing facilities can be used directly from a command line or from within Korean Solaris applications as discussed in the following sections.

Printing Korean Output from a Command Line

From a command line, you can print using any of the following ways:

- Directly to a line printer
- Using the `xetops` or `xutops` utilities to convert text to bitmapped graphics.

Printing with a Line Printer

- **To print an EUC file, use the following command:**

```
system% lp euc-filename
```

- **To print a Packed format file on a printer that supports this format, you can use the following command:**

```
system% lp -T PACK PACK-filename
```

- **To print a Johap format file on a printer that supports this format, you can use the following command:**

```
system% lp -T JOHAP JOHAP-filename
```

For more information on setting up the Packed or Johap filters, see *Korean Solaris System Administrator's Guide*.

- **To print a `ko.UTF-8` file to an EUC printer, type the following commands:**

The first line converts the file to an EUC file. The print out will be missing any characters that are not defined in EUC.

```
system% iconv -f ko_KR-UTF-8 -t ko_KR-euc ko.UTF-8_filename >euc-filename
system% lp euc-filename
```

- **To print a `ko.UTF-8` file to a Johap (KS C 5601-1992) printer, type the following commands:**

The first line converts the file to a Johap file.

```
system% iconv -f ko_KR-UTF-8 -t ko_KR-johap92 ko.UTF-8_filename >johap92-filename
system% lp johap92-filename
```

Printing with the `xetops` and `xutops` Utilities

The `xetops` and `xutops` utilities convert Korean text into a bitmapped graphics printed image. They allow you to print Korean characters using a PostScript-based printer.

A typical command line for printing a file named *filename* containing Korean characters, with or without ASCII/English characters, would be as follows:

```
system%  pr filename | xetops | lp
or
system%  pr filename | xutops | lp
```

Make *filename* the name of the file to print. This file may contain ASCII/English characters as well as Korean. Refer to the *xetops(1)* and *xutops(1)* man pages for more detailed information.

Printing with the `mp` Utility

A new and enhanced `mp(1)` print filter is available in the Solaris 8 environment that can print various input file formats including flat text files written in UTF-8. This filter uses TrueType and Type 1 scalable fonts and X11 bitmap fonts available on the Solaris operating system.

The output from the utility is standard PostScript, and can be sent to any PostScript printer.

Note – Starting with the next release of the Solaris environment, `zutops (10)` will be obsolete.

To print with the `mp` utility, type the following:

```
system% mp filename | lp
```

You can also use the utility as a filter, as the utility accepts `stdin` stream:

```
system% cat  filename | mp | lp
```

You can set the utility as a printing filter for a line printer. For example, the following command sequence tells the printer service `LP` that the printer `lp1` accepts only `mp` format files. This command line also installs the printer `lp1` on port `/dev/ttya`. See the `lpadmin (1m)` man page for more details.

```
system% lpadmin -p lp1 -v /dev/ttya -I MP
system% accept lp1
system% enable lp1
```

You can add the `lpfilter` utility for a filter by using the `lpfilter(1M)` command as follows:

```
system% lpfilter -f filtername -F pathname
```

The `lpfilter` command tells LP that a converter (in this case, xutops) is available through the filter description file named `pathname`. The pathname can be determined as follows:

```
Input types: simple
Output types: MP
Command: /usr/bin/mp
```

The filter converts the default type file input to PostScript output using `/usr/bin/mp`.

To print a UTF-8 text file, use the following command:

```
system% lp -T MP UTF-8-file
```

For more details on the mp(1) command, refer to the `mp(1)` man page.

CHAPTER 8

Hanja Tool

Hanja Tool enables users to expand the capabilities of the standard Korean Solaris operating environment Hangul–Hanja conversion mode by adding Hanja ideograms and managing available lookup choices for Hangul–Hanja conversion. It works only with words of more than one syllable.

Hanja Tool Features

Hanja Tool enables you to add (and delete) entries to your own Hangul–Hanja dictionary extension, thus expanding and controlling your Hanja choices in Hangul–Hanja conversion mode. The system-wide Hangul–Hanja dictionary is read-only and consequently not editable.

A Hanja Tool is a viewer for the Hangul–Hanja dictionary, which is in binary format and consequently not otherwise readable.

A Hanja Tool creates your own local Hangul–Hanja dictionary in `$HOME/.usrhjd`. Hanja Tool makes this an extension to the read-only, system-wide Hangul–Hanja dictionary, `/usr/lib/mle/ko/syshjd` or `/usr/lib/mle/ko.UTF-8/syshjd`. Each user's dictionary for Hangul–Hanja conversion mode comprises both the system-wide dictionary and the user's own extension, which work seamlessly together.

Using Hanja Tool

- **Enter the following command to create a Hanja Tool:**

```
system% sdthanja
```

This command displays a Hanja Tool window, like the following:

A Hanja Tool appears initially in ASCII input mode, as shown, in the status area. Before you can type Korean characters in Hanja Tool fields you must turn on Hangul input mode (Control-Space).

The Hanja Tool input fields, scrolling list areas, and buttons are labelled and used as follows:

Hangul – In this input field you type the Hangul word to search Hanja(s) in the Hangul–Hanja dictionaries.

System Dictionary – This scrolling-list area displays the system-wide Hangul–Hanja dictionary's entries. When you click on the Find button, this area displays the System Dictionary's Hanja entries associated with the Hangul entry committed in the Hangul: field. You can scroll through the available Hanja choices for that Hangul word or through the entire System Dictionary using the scroll bar on the right.

User Dictionary – This scrolling-list area displays the Hanja entries in the User Dictionary. When you click on the Find button, this area displays the User Dictionary's Hanja entries associated with the Hangul entry committed in the Hangul field. You can scroll through the available Hanja choices for that Hangul word or through your entire user dictionary using the scroll bar on the right.

Find – Click on this button to search the Hangul–Hanja dictionary (both parts) for Hanjas associated with the Hangul entry committed in the Hangul field.

Add – Click on this button to add a Hangul-Hanja conversion to the User Dictionary. Clicking on this button brings up the Add/Modify dialog box.

Modify – Click on this button to modify a Hangul-Hanja conversion in the User Dictionary. Clicking on this button brings up the Add/Modify dialog box only when the Hanja word from the User Dictionary is selected. Othewise, a user guidance message is displayed.

Delete – Click on this button to delete the selected Hanja word from the User Dictionary. A delete confirmation dialog box is displayed before starting a delete process.

Quit – Click on this button to quit the Hanja Tool.

Finding a Hanja

To search the two parts of the Hangul–Hanja dictionary you can use the scroll bars on the sides of the System Dictionary and User Dictionary scrolling list areas.

To find the Hanja(s) associated with a particular Hangul word, turn on Hangul input conversion and then proceed as follows:

1. **Type the Hangul word in the Hangul: field.**

   The entry appears reverse-video highlighted in the preedit area as you type.

2. **Commit the entry by pressing Control-K.**

   The following display appears:

3. **With the correct Hangul word committed in the Hangul: field, click on the Find button to command the Hanja Tool to search.**

   This produces a display like the following:

The line beneath the status area tells the number of the word in the User Dictionary and System Dictionary.

Adding a Hanja Word

To add a Hanja word to the Hangul–Hanja dictionary proceed as follows:

1. **Click on the Add button of the main screen to bring up the Add/Modify dialog box:**

2. **Type a new word or change the existing word in the Add/Modify dialog box, and commit (Control-K) the Hangul word to associate with the Hanja word you wish to add to your dictionary.**

3. **In the Hanja field, type the Hanja word associated with the Hangul word you wish to add to your dictionary.**

   Use the (character-by-character) methods described in "Converting to Hanja" on page 22.

4. **Click on the Add button.**

   A display like the following appears:

   The line beneath the status area tells you the word was added to the User Dictionary. (Only the User Dictionary is changed; the System Dictionary is read-only).

Modifying a Hanja Word

Modifying a Hanja word is similar to the process for adding a Hanja word (see "Adding a Hanja Word" on page 63).

- **Select a Hanja word from the User Dictionary and click on the Modify button.**

The remaining steps are exactly the same as adding a Hanja word.

Deleting a Hanja Word

To delete a Hanja from the User Dictionary, proceed as follows:

1. **In the Hangul: field type the Hangul word associated with the Hanja you seek in your User Dictionary.**
2. **Commit this Hangul input by pressing Control-K.**
3. **Click on the Find button.**

   This displays the Hanja in the User Dictionary scrolling list area. Otherwise scroll the list area to the Hanja you seek to delete.
4. **Click on that Hanja to select it.**
5. **Click on the Delete button to delete that Hanja from the User Dictionary.**

   A delete confirmation dialog box will be displayed:
6. **Click OK if you are sure you want to delete the Hanja. Otherwise, click Cancel.**

   The line beneath the status area tells you the word was deleted from the User Dictionary. (Only the User Dictionary is changed; the System Dictionary is read-only).

APPENDIX **A**

Open Windows Information

This appendix includes information specific to the OpenWindows environment.

Introduction to OpenWindows

Design of Korean Solaris Software

This Korean localization of Sun's internationalized OpenWindows environment includes enhancements for handling appropriate linguistic and cultural conventions, which it provides to two broad working environments:

- A localized user environment, which includes localized DeskSet tools and window manager (`olwm`) that communicate with users in Korean.
- A localized development environment, which programmers use to develop localized applications, with Xlib and the XView™ Toolkit, which have been internationalized for this use. Programmer/developers should refer to *Solaris Internationalization Guide for Developers*.

Expanded Workspace Properties Worksheet

The Localization category in the Workspace Properties worksheet lets you set the locale for applications to start up in. With this page you can set the Basic Setting, Display Language, Input Language, Numeric Format, and Time Format for new application windows from inside the Korean OpenWindows environment.

These settings take effect each time a local application starts up. Application windows are displayed in the currently set locale. After you change the locale, new application windows are displayed in the new locale, but existing application windows stay in the locale they originally started up in.

OpenWindows supports `ko` locale only. It does not support `ko_KR.EUC`, `ko.UTF-8`, or `ko_KR.UTF-8` language selections.

Korean OPEN LOOK DeskSet

The following DeskSet tools are provided in this Solaris release. All can handle Korean language input and output. A man page is provided for each.

Audio Tool – Tool for recording, playing, editing, and controlling workstation audio parameters.

Binder – Tool for defining which actions associate with which file type. This association can be set graphically.

Calculator – Visual calculator for use with mouse or keyboard.

Calendar Manager – Manages business and social appointments; can use electronic mail to send automatic reminders.

Clock – Displays current analog or digital time.

Command Tool – Standard OPEN LOOK scrolling window terminal emulator.

File Manager – Graphical tool for accessing files and directories. Represents file types with varying colors and icons. Navigates through the file system with the mouse.

Font Editor – Visual tool for editing font appearance and creating new fonts.

Icon Editor – Visual tool for editing icon appearance and creating new icons.

Image Tool – Interactive image viewer. Image Tool can be used to view the contents of file types such as GIF, TIFF, JPEG, PostScript, and others.

Mail Tool – Tool for handling electronic mail.

Performance Meter – Real-time system performance meter that can display a variety of data.

Print Tool – Graphical front-end to the print command. It supports OPEN LOOK *drag-and-drop* file transfer operations.

Shell Tool – Standard OPEN LOOK non-scrolling window terminal emulator. The window behaves like an ASCII character terminal for entry of UNIX commands at a system shell prompt and other terminal operations.

Snapshot – Tool to *snap* or capture picture of a window or region of a screen in a bitmap (raster file). Used for capturing screen image displays in this user's guide.

Tape Tool – Graphical tool for controlling the tape driver.

Text Editor – Visual text editor used in OpenWindows tools such as the Mail Tool composition window.

Starting Up OpenWindows

Checking Your User Environment

Before you log in, your system administrator should have set your required user environment variables and corresponding entries in the `.cshrc` file in your home directory. These system environment variables are essential to using Korean features.

`.cshrc` File

These system environment variables might not have been set. So the first time you log in, before you start up OpenWindows for the first time, check to make sure lines like the following are in the `.cshrc` file in your home directory:

```
setenv LANG ko
setenv OPENWINHOME /usr/openwin

set path=( /usr/SUNWale/bin $OPENWINHOME/bin $path )

...

if ($?USER != 0 && $?prompt != 0) then
   /bin/stty cs8 -istrip defeucw
endif
```

Make sure the `LANG` variable is set to `C` (ASCII) or `ko` (Korean) *before* the `if...endif` statement and that *no* prompt is set before the `if...endif`.

If these lines are not present or are different, contact your system administrator. If you are your own system administrator or an advanced user, refer to *Korean Solaris System Administrator's Guide* for further information on setting up your Korean Solaris system.

`.openwin-init` File

If you have a `.openwin-init` file in your home directory and might use the Korean character input facilities, check this file for the presence of an `htt` command, as described in "How and When `htt` is Started" on page 15.

`.xinitrc` File

If your system has a `.xinitrc` file in your home directory, check and make sure it contains at least the lines provided in Korean Solaris `$OPENWINHOME/lib/Xinitrc` file.

Starting the Korean OpenWindows Environment

After you verify that your `.cshrc` file has set your user environment correctly for Korean language operation, you are ready to start up your Korean OpenWindows environment as follows:

- **Type the following command at the system prompt to start up the Korean OpenWindows environment:**

```
system% openwin
```

Changing the Language Setting on the Workspace Properties Worksheet

The Korean OpenWindows Workspace Properties worksheet contains a localization field. To change the language setting for the next OpenWindows tools you start:

1. **Choose Properties... on the main window Workspace menu.**

2. **Choose the Locale category on the Category pull down menu.**

   The English form of the Workspace Properties worksheet is on the left and the Korean form is on the right:

You can set the display and input mechanisms of the Korean OpenWindows environment by using the Locale field in the Workspace Properties worksheet. You can switch between the U.S. and the Korean settings.

The Locale setting determines which characters (ASCII, Korean) appear in new tool windows when they start up. Changing the locale does not affect the appearance or operation of tool windows that were started up before the change.

Changing Your System Locale Setting

To change the Locale setting on the Workspace Properties worksheet (and in the `.OWdefaults` file as explained in *Korean Solaris System Administrator's Guide*):

OpenWindows and `htt`

In the OpenWindows environment, the `.openwin-init` file in the user's home directory is referenced if it exists. Therefore, if the user's `.openwin-init` file lacks a line to start `htt`, `htt` is not strarted when the OpenWindows environment is started up. (Refer to "Checking Your User Environment" on page 73.) For your applications' Korean input functions to operate as intended, the `.openwin-init` script must start `htt` before the script starts an application that uses `htt` for Korean character input. If `htt` is started after the application, only a root-window style input method server window can be used, as shown in "`htt` Displays Appearance" on page 10.

Changes to the `.openwin-init` File

The OpenWindows Workspace⇒Utilities⇒Save Workspace function writes or rewrites a user's `.openwin-init` file when it saves the current configuration of the workspace screen. So after each time you use the Save Workspace pulldown menu selection, or edit `.openwin-init`, check your `.openwin-init` file and make sure the `htt` command precedes any line that starts an application that takes Korean character input.

Such placement of this command ensures future correct connection to `htt` if the OpenWindows system is restarted later. A manual edit to `.openwin-init` will be overwritten the next time you use the Save Workspace function. You can save and use your edits by first exiting and then restarting the OpenWindows environment.

Printing in OpenWindows

You can use the Workspace Properties menu to set up the `xetops` print filter.

- **Type `cat $FILE | xetops | lp` in the Properties worksheet, as shown in the following figure:**

Customizing Your OpenWindows Workspace

Using Fonts

The localized language functions of Korean Solaris applications use font sets, or groips of fonts including both ASCII character fonts and non-ASCII Korean character fonts. These combinations of fonts are required for Korean display. They can be used, as font names are, in customizing your workspace as described in *Solaris User's Guide*.

Korean (ko) Font Lists

A Korean (ko) font list is composed of one English font representing ASCII characters in KS C 5636 or ISO8859-1, and one Korean font represeting characters in KS C 5601-1987-0.

The KS C 5636 and ISO8859-1 character sets are nearly identical. The diferences are that KS C 5636 uses only the code values from 0 to 127, and the backslash character (whose ISO8859-1 code value is 92) is replaced by the Korean currency symbol.

Korean Solaris provides some default font lists defined in the application defaults files in `/usr/dt/app-defaults/ko/*`. The following is an excerpt from one of the files, `Dtwm`:

```
Dtwm*icon*fontList: \
    -dt-interface-system-medium-r-normal-s*ksc*:
```

This portion of the file refers to a font list that contains two fonts, which are defined in `/usr/openwin/lib/locale/ko/X11/fonts/75dpi/fonts.alias`.

Korean (ko.UTF-8) Font Lists

A Korean (ko.UTF-8) font list is composed of one English font representing ASCII characters in KS C 5636 or ISO8859-1, and one Korean Johap font represeting codeset 1characters in KS C 5601-1992-3.

Korean Solaris provides some default font lists defined in the application defaults files in `/usr/dt/app-defaults/ko.UTF-8/*`. The following is an excerpt from one of the files, `Dtwm`:

```
Dtwm*icon*fontList: \
    -dt-interface-system-medium-r-normal-s*ksc*:
```

This portion of the file refers to a font list that contains two fonts, which are defined in `/usr/openwin/lib/locale/ko.UTF-8/X11/fonts/75dpi/fonts.alias`.

Starting Applications With a Specific Korean (ko or ko.UTF-8) Font List

When you start an Asian Solaris tool at the command line, you can also specify its fonts. Below is an example of a command line argument used to start a new Korean terminal with a specified font list:

```
system% dtterm -fn \
    -dt-interface system-medium-r-normal-s sans-14-120-75-75-p-60-ksc5636-0;\
    -dt-interface system-medium-r-normal-s sans-14-120-75-75-p-120-ksc5601.1987-0:
```

Note the two delimiters used in the font list. The ; delimiter is used to separate the font names except for the last font name, which ends with the : delimiter. (In the example above, ; follows the Korean font name.) Since there are spaces in long font names, the font list is enclosed in quotation marks.

▼ Specifying Font on Command Lines

A command line that starts a Korean OpenWindows application can specify the application's font. When the current locale is Korean, the command uses one of the defined font-set aliases instead (explained in the following section), for example:

```
system% cmdtool -font fontset_name &
```

But when the current locale is `C` the command uses a font name and cannot use a font-set alias. The following shows a command using the long name of an ASCII character font:

```
system% cmdtool \
    -font -misc-fixed-medium-r-normal--9-80-100-100-c-60-iso8859-1 &
```

Font Set Names

Korean Solaris software provides several font sets that combine two or more fonts so both English and Korean characters can be used together in one window. Some font sets comprise an English font plus a Korean font, both specified in KS C 5601. For easy use several of these font sets have simple names as follows:

- kodig12
- kodig14
- kodig16
- myeongjo14
- myeongjo20
- myeongjo24

  Each of these font sets is made up of two font files. The `$OPENWINHOME/lib/locale/ko/OW_FONT_SETS/OpenWindows.fs` file defines the full Korean Solaris font set.

  In addition, the Korean Solaris operating environment provides scalable and bitmap fonts in the following typefaces:

- Kodig
- Myeongjo
- Round Gothic
- Pilki

- Haeso
- Graphic

  Kodig and Myeongjo contain Korean characters in accordance with the KS C 5601 and KS C 5700 standards. These fonts are located in the `/usr/openwin/lib/locale/ko/X11/fonts` and `/usr/openwin/lib/locale/ko.UTF-8/X11/font` directories.

  Korean bitmap fonts are located in the `/usr/openwin/lib/locale/ko/X11/fonts/75dpi` and `/usr/openwin/lib/locale/ko.UTF-8/X11/font/75dpi` directories.

Scaling Applications Windows and Fonts

The `$OPENWINHOME/lib/locale/ko/OW_FONT_SETS/OpenWindows.fs` file also sets the following font size definitions for use in command lines:

- `small`=12 points
- `medium`=14 points (default size)
- `large`=16 points
- `extra_large`=20 points

  For example, the following command line starts up a Command Tool window that uses 16-point type and is scaled proportionally larger than the default:

  ```
  system% cmdtool -scale large &
  ```

APPENDIX **B**

Binary Compatibility Package

Applications compiled under Korean OpenWindows 2.*x* or Solaris 1.*x* or SunOS 4.*x* systems have different binary formats than the current Korean Solaris release. Older applications can nevertheless be run under the current Korean release without being recompiled by using its included binary compatibility package (BCP).

Note – `SUNWowbcp` must be included in your system configuration in order for you to run the following commands. See your system administrator for installation.

The following BCP command runs the compiled binary code of earlier SunOS4.*x* /Solaris 1.*x* /Korean OpenWindows 2.*x* applications without recompilation, although OpenWindows V2 Korean applications display no input server status region. As shown in the following examples, the command calls the application by its old name (*old_application_name*) and sets the basic locale, input language, and display language using the older version's specific locale name (*oldlocale*):

```
system% old_application_name -lc_basiclocale oldlocale -lc_inputlang
     oldlocale-lc_displaylang oldlocale
```

The following example shows the command for running the compiled binary code of an earlier version of the `textedit` application on a current Korean Solaris release system:

```
system% textedit -lc_displaylang korean -lc_basiclocale korean
    -lc_inputlang korean
```

Due to incompatibilities between Korean Solaris 2.x and 1.x applications, you cannot cut and paste Korean characters between them.

Korean Test Utilities

Every utility listed in this section is supported, but for this version of Solaris, you are encouraged to use the XPG4 internationalization APIs as described in *Solaris Internationalization Guide for Developers*.

The utilities in the following table test various aspects of the Korean (KS C 5601) national standard character set. Except Korean `isksc`, they also assume that the character being tested is part of the national standard character set.

The arguments for the functions in the following table must be a character in WC, `wchar_t`. For more information, see the *kctype(3x)* man page.

TABLE B–1 Korean Character Classification Functions

Utility	Description
`isksc`	Returns true if it is in the KS C 5601 character set.
`iskroman`	Returns true if it is a Roman character as defined by the KS C 5636 character set.
`iskromannum`	Returns true if it is a Roman numeral symbol in the KS C 5601 character set.
`isksymbol`	Returns true if it is a Latin symbol or special character in the KS C 5601 character set.
`iskparen`	Returns true if it is a right or left parenthesis in the KS C 5601 character set.
`isklatin`	Returns true if it is a Latin letter character in the KS C 5601 character set.
`iskletter`	Returns true if it is a Korean vowel or consonant in the KS C 5601 character set.
`iskline`	Returns true if it is a ruled line symbol in the KS C 5601 character set.
`iskunit`	Returns true if it is a unit character in the KS C 5601 character set.
`isksci`	Returns true if it is a scientific symbol in the KS C 5601 character set.
`iskgen`	Returns true if it is a graphic or general symbol in the KS C 5601 character set.
`iskgreek`	Returns true if it is a Greek character in the KS C 5601 character set.
`iskrussian`	Returns true if it is a Russian character in the KS C 5601 character set.
`iskuser`	Returns true if the character is in the user-defined area of the KS C 5601 character set.

TABLE B–1 Korean Character Classification Functions *(Continued)*

Utility	Description
`iskhanja`	Returns true if it is an ideogram in the KS C 5601 character set.
`iskhangul`	Returns true if it is a Hangul phonogram in the KS C 5601 character set.
`iskkata`	Returns true if it is a Japanese Katakana character in the KS C 5601 character set.
`iskhira`	Returns true if it is a Japanese Hiragana character in the KS C 5601 character set.

Asian-Specific Utilities

This section describes functions for wide character and string input and output, character classification, and conversion functions for the Korean character sets. Asian Solaris software implements a wide character library for handling Korean character codes according to industry standards.

Routines that have Korean language-specific dependency are in their own language-specific library, which is linked with the corresponding C compiler option. In Korean Solaris, `libkle` is linked with `-lkle`. Refer to the appropriate man page for more information.

Asian Solaris software defines WC as a constant-width, four-byte code. WC uses the ANSI C data type `wchar_t`, which Solaris software defines in `wchar.h` as follows:

```
typedef long wchar_h;
```

In Solaris software, `long` is four bytes.

Conversion Utilities

The conversion utilities described in this section are available, but you should use `iconv()` as a standard function.

Asian Solaris software provides facilities for various conversions, for example:

- Characters within a codeset, such as converting uppercase ASCII to lowercase.
- Between different conventions for national standard character sets, such as:
  - Between Combination and Completion code, both KS C 5601-1987 and KS C 5601-1992.

    - Between GB and EUC.
    - Between CNS 11643 code and Big5.
- Between code formats (such as EUC and WC).

Programs using the general multibyte conversion utilities should include the header files `widec.h` and `wctype.h`. Korean Solaris specific routines (such as `isk`*xxx*) are declared in `ko/xctype.h`.

Programs using the general multibyte conversion utilities should include three header files: `wctype.h`, `widec.h,` and `ko/xctype.h`.

Conversion Within a Codeset

The multibyte conversion functions are similar to the one-byte conversion functions `toupper()` and `tolower()`. These functions convert wide-characters to other wide characters. For more information on conversion routines, see the man pages for *wconv(3)* for all locales and *kconv(3)* for Korean.

Conversion for Korean Character Codes

The following routines perform character-based code conversion on the KS C 5601 character set. They convert characters in the set between Completion code (or EUC format) and Combination code (or Packed code). To use these routines, the library `kle` must be linked using the C compiler option `-lkle`. For more information, see the *kconv(3x)* man page.

TABLE B–2 Korean Code Conversion Functions

Function	Description
`comptopack ()`	Converts a character in Completion code to Combination (packed) code of KS C 5601-1987.
`packtocomp ()`	Converts a character in Combination (Packed) code of KS C 5601-1987 to Completion code.
`wasuntojohap ()`	Converts a character in Completion code to Combination (Packed) code of KS C 5601-1992.
`packtocomp ()`	Converts a character in Combination (Packed) code of KS C 5601-1992 to Completion code.

APPENDIX **C**

Running Networked Applications

You can run Korean localized applications on a remote machine as explained in Appendix C, “Networked Applications,” in *Solaris Advanced User's Guide*. That guide describes the advanced features of the OpenWindows environment that enable you to run applications that reside on another machine on your network.

Note – It is not possible to run networked applications as described in this appendix with the `ko.UTF-8` locale.

Note – Most users do not need to read this appendix. If you want to explore running networked applications, ask your system administrator about special applications possibly available on your network.

Applications running on Korean OpenWindows 2.*x* can be displayed remotely on an Korean OpenWindows 3.*x* system. Korean OpenWindows 3.*x* applications may not be able to be displayed remotely in an Korean OpenWindows 2.*x* environment because not all 3.*x* system fonts are available in 2.*x* environments.

Instructions for Running Networked Applications

You must follow the information and directions in Appendix C, “Networked Applications,” in *Solaris Advanced User's Guide*, which you should read thoroughly. You must additionally make the following adjustments to enable operation of the Korean features of the Korean OpenWindows environment.

Setting Required Environment Variables

To run a networked application on a remote machine you must set your environment variables correctly, as follows:

- The `DISPLAY` environment variable in your shell on the remote machine must be set to your local screen
- The `LANG` environment variable in your shell on the remote machine must be set to `ko`.
- If the OpenWindows libraries have not been installed in the standard `/usr/lib` or `/usr/local` shared library directories, you must set the `LD_LIBRARY_PATH` environment variable to the appropriate directory (`$OPENWINHOME/lib`)

Sample Command Sequences for Remote Operation

▼ How to Display Remote OpenWindows V2.x on a Local OpenWindows V3.x System

The following sequence of commands is an example of starting up a Shell Tool on a remote Korean Solaris 1.x (including Korean OpenWindows 2.x) machine. In this example, your local machine is running Solaris 2.x, including Korean OpenWindows 3.x, and the remote machine, where you start up the Shell Tool, has Korean Solaris 1.x, including Korean OpenWindows 2.x:

1. **Type the following command.**

```
local_machine% xhost +remote_machine
```

2. **Log in on the remote machine.**

```
local_machine% rlogin remote_machine
```

or

```
local_machine% telnet remote_machine
```

3. **Set the language locale on the remote machine.**

```
remote_machine% setenv LANG korean
```

4. **Set the remote machine to display on your local machine.**

```
remote_machine% setenv DISPLAY local_machine:0.0
```

5. **Set the OpenWindows environment home directory.**

   ```
   remote_machine% setenv OPENWINHOME /usr/openwin
   ```

6. **Set the path to the OpenWindows LD library.**

   ```
   remote_machine% setenv LD_LIBRARY_PATH $OPENWINHOME/lib
   ```

7. **Start up the OpenWindows application, for example Shell Tool:**

   ```
   remote_machine% $OPENWINHOME/bin/xview/shelltool -lc_basiclocale \oldlocale -lc_inputlang oldlocale -
   ```

   To run a different application, use that application's command in place of `shelltool`.

▼ How to Display Remote OpenWindows V3.x on a Local OpenWindows V2.x System

The following sequence of commands is an example of starting up a Shell Tool on a remote Korean Solaris 2.x (including Korean OpenWindows 3.x) machine from your local machine which is running Korean Solaris 1.x (including Korean OpenWindows 2.x). In this example, the remote machine, where you start up the Shell Tool, has only Korean OpenWindows 3.x.

1. **Make a directory on the Korean Solaris 2.x machine.**

   ```
   local_machine% mkdir /home/fonts
   ```

2. **Change directory.**

   ```
   local_machine% cd /home/fonts
   ```

3. **Create** `Compat.list` **and** `Families.list`**.**

   ```
   local_machine% cat > /home/fonts/Compat.list
   /-b&h-lucindatypewriter-medium-r-normal-asian-14-140-72-72-m-70\
   -iso8859-1 /hnggthr _FontDirectorySYN
   ^D
   local_machine% bldfamily
   ```

4. **Add directory to server font path.**

   ```
   local_machine% xset +fp /home/fonts
   ```

5. **Allow server access from the remote machine.**

   ```
   local_machine% xhost +remote_machine
   ```

6. **Log in on the remote machine.**

   ```
   local_machine% rlogin remote_machine
   ```

   or

   ```
   local_machine% telnet remote_machine
   ```

7. **Set the language locale on the remote machine.**

```
remote_machine% setenv LANG ko
```

8. **Set the remote machine to display on your local machine.**

```
remote_machine% setenv DISPLAY local_machine:0.0
```

9. **Set the OpenWindows environment home directory.**

```
remote_machine% setenv OPENWINHOME /usr/openwin
```

10. **Set the path to the Korean OpenWindows LD library.**

```
remote_machine% setenv LD_LIBRARY_PATH $OPENWINHOME/lib
```

11. **Start the `htt` X input method server.**

```
remote_machine% $OPENWINHOME/bin/htt &
```

12. **Start the OpenWindows application you want, for example, Shell Tool:**

```
remote_machine% $OPENWINHOME/bin/shelltool
```

In XView-based applications you do not need to give an explicit font set definition.

Mapping Korean Keyboard Functions

This appendix shows you how to configure a Sun Korean keyboard to make selected key functions when you need them.

Two keys are missing on the Korean Type-4 and Type-5 keyboards. They have no Compose or AltGraph keys.

You can use the `xmodmap` command to make any existing key function as a Compose or AltGraph key. The following example command sequence makes the right Meta key () function as the AltGraph key:

```
system% xmodmap -e "remove mod1 = Meta_R"
system% xmodmap -e "remove mod2 = Mode_switch"
system% xmodmap -e "add mod2 = Meta_R"
system% xmodmap -e "keysym Meta_R = Mode_switch"
```

The following command sequence maps the left Meta key () to the Compose key:

```
system% xmodmap -e "keysym Meta_L = Multi_key"
```

The `$OPENWINHOME/share/etc/keytables/Korea4.kt` and `$OPENWINHOME/share/etc/keytables/Korea5.kt` files and *xmodmap(1)* man page provide more information.

Glossary

ANSI	American National Standards Institute. ANSI proposes standard definitions for different computing languages. The most recent standard for the C language, prepared by the ANSI C X3J11 Committee, includes library functions for computing with multibyte characters for international usage, as well as a new data type, `wchar_t`, for dealing with four-byte characters. This standard is not completed, so it is referred to as the "proposed ANSI C standard," or ANSI C-X3J11.
ASCII	American Standard Code for Information Interchange. A seven bit code containing English upper and lowercase letters, punctuation, numbers and control codes. The eighth bit in each byte is used by different applications for parity checking, communication and message passing protocols, compacting data, or other purposes. Applications that are intended to be internationalized cannot utilize this bit if they are going to use multiple code sets or multibyte characters, and utilities that handle multiple code sets or multibyte characters.
Category	In the Korean Solaris documentation set, category is related to localization. A category is a portion of a country's language representation and cultural conventions. For instance, the date is often represented in the U.S. as *Month, Day*, *Year*; while in another country it might be *Day*, *Month*, *Year*. The date and time can be thought of as one category of a local language. Categories also refer to the program categories, the environment variables that are related to categories, and the ANSI localization tables for each category.
Character Set	A character set is defined as a set of elements used for the organization, control, or representation of data. Character sets may be composed of alphabets, ideograms, or other units. This may seem a bit open-ended, but character sets may contain other character sets, which makes the boundaries unclear. For example, the KS C 5601 character set contains English, Greek, Russian, and Japanese character sets, in

	addition to Hangul syllables (consonant and vowel combinations), Hanja ideograms (Chinese characters), and many other characters.
Code set	Also called a coded character set, this is a set of unambiguous rules that establishes a character set and the one-to-one relationship between each character in the character set and its bit representation. For example, the English character set, including punctuation and numbers, can be mapped to the ASCII code set in such a way that each character corresponds to only one bit code, and no bit code corresponds to more than one character.
Combination code	Another name for *Packed code* or *Johap code* described below.
Completion code	Also called Wangsung. Completion code is a pre-defined set of Korean character codes, which maps preselected Hangul, Hanja, special symbols, alphabets of other languages and so on into two-byte coding space. This representation is defined in KS C 5601 and used as EUC codeset 1 by the Korean Solaris operating environment.
EUC	Extended UNIX Code. Describes four code sets modelled on ISO-2022. Each code set can contain one or more different character sets, like the Hangul and Hanja character sets in KS C 5601. The four code sets are referred to as codesets 0, 1, 2, and 3, and in this text they are sometimes abbreviated as cs0, cs1, cs2, and cs3. Other internationalization efforts sometimes call these g0, g1, g2, and g3. Codeset 0 is also called the primary code set, and codesets 1, 2, and 3 are called the supplementary code sets. In the Korean and Chinese implementations of the EUC codes, the primary code set (cs0) contains ASCII and begins with a zero in the most significant bit.
Hangul	Hangul is the phonetic alphabet commonly used in Korea. Each character corresponds to a spoken syllable, usually a consonant-vowel pair or a consonant-vowel-consonant triad. KS C 5601 defines 2350 Hangul characters used in standard computing.
Hanja	Hanja characters are Korean ideograms, which came originally from ancient China (the word itself means Chinese character). They were adopted many centuries ago and have evolved somewhat different meanings in China and Korea. But because they are not phonetically based, Chinese and Korean Hanja have remained closer in meaning than have Italian, French, and Spanish, which evolved into separate languages over the same time span. The Korean Industry Standard defines the 4888 most frequently used Hanja characters in the KS C 5601 standard.
ISO	International Standards Organization. Composed of a number of professional societies and companies, this organization studies and makes recommendations on internationalization issues. ISO 2022 proposes and describes the Extended UNIX Codes. Other ISO

	proposals include the European 8-bit code and communication protocols for internationalization.
Johap code	Johap code is a Packed code (also called Combination code), which is defined in the KS C 5601-1992 document. Unlike the Packed code defined in KS C 5601-1987 or before, Johap code has a set of Hanja characters and special symbol characters.
KSC	Korean Industry Standard Codeset. This is the Korean analogue to ASCII. The KSC describes standards for computing in the Korean environment. KS C 5601 contains code assignments in Completion code for Hangul and Hanja characters, graphics and punctuation characters, two Japanese phonetic alphabets (Hiragana and Katakana), control codes, and several western alphabets (Roman, Russian, and Greek characters). This standard defines 2350 Hangul characters, 4888 Hanja characters, and 986 additional characters (for punctuation, foreign alphabets, numbers, graphics, and others). Each character is two bytes long, and does not utilize the highest or most significant bit of each byte. In other words, it uses the lower seven bits of each byte for character assignments.
Locale	A locale describes a language or cultural environment. Its setting affects the display or manipulation of language-dependent features. Korean Solaris software provides `C` for U.S.A, `ko` for Korean extended UNIX code, and `ko.UTF-8` for Korean Universal Multiple Octet Coded Character Set Transmission Format.
N-byte code	This coding system assigns each Korean alphabetic consonant or vowel a one-byte code. These are built up into Hangul syllabic characters with the Hangul automata.
Packed code	Packed code (also called Combination code) is a systematic method for coding Hangul syllabic characters in a two-byte code. Each 16-bit (two-byte) character contains a high or most-significant bit (1) and three 5-bit fields. These fields contain the codes for the beginning consonant (*x*), a middle vowel (*y*), and an optional ending consonant (*z*), as follows: *1xxxxxyyyyyzzzzz*. Hanja characters cannot be represented in Packed code, because many Hanja characters may be represented by one phonetic pronunciation. Packed code is defined in KS C 5601-1987 and earlier as a supplementary code set.
POSIX	Portable Operating System for Computer Environments. An IEEE standards group comprising seven committees that create documents for standardizing and internationalizing UNIX. POSIX document 1003.1 deals with the kernel and system calls. 1003.2 concerns the C-shell and standard libraries. The other five deal with real-time computing, communications and networking, and other issues.

UTF-8	Universal Multiple Octet Coded Character Set (UCS) Transmission Format. `ko-UTF-8` provides the Korean-related characters in this standard. UTF-8 is a representation of Unicode.
Unicode	The international character set and encoding developed by the Unicode Consortium.
Wide Character Code (WC)	A constant-width four-byte code, called WC in Asian Solaris documentation, for the internal representation of EUC codes using the new ANSI-C data type `wchar_t`. Although EUC does not specify limits on the size of the supplementary code sets (codeset 0 is always one byte), WC specifies a character as four bytes. Standardizing on four bytes takes up more memory space than necessary if the environment is primarily ASCII, but it also speeds processing time for strings of mixed characters; the 1000th character always begins at byte 4000 (and the 0th character starts at byte 0). This is useful for any type of indexing in applications.
X/Open	X/Open started as a consortium of international UNIX vendors from Europe, USA, and Asia. It is now one of the major standards organizations like POSIX and ANSI; source of the *X/Open System Interface Portability Guide*.

Index