Putting It All Together 4

Much of OLIT programming is figuring out how the various widgets work, and how to assemble them into a user interface. The best way to teach this is to show it in a real OLIT program. The program described in this chapter, `Translator.c`, translates a some common English expressions into several foreign languages. Examine the run-time application and the commented source code to get an idea of some of the issues involved in creating an OLIT application.

You may obtain `Translator.c` and the its helper files (`Makefile`, and `Resources_translator`) by sending mail to greg.kimura@Eng.sun.com with the subject line “OLIT Quick-Start Programs”. To compile and execute `Translator.c`, copy `Translator.c`, `Makefile`, and

`Resources_translator` into a work directory. Set the environmental variable `XENVIRONMENT` to `./Resources_translator`. Enter `make<return>` to make. Enter `Translator<return>` to run.[1]

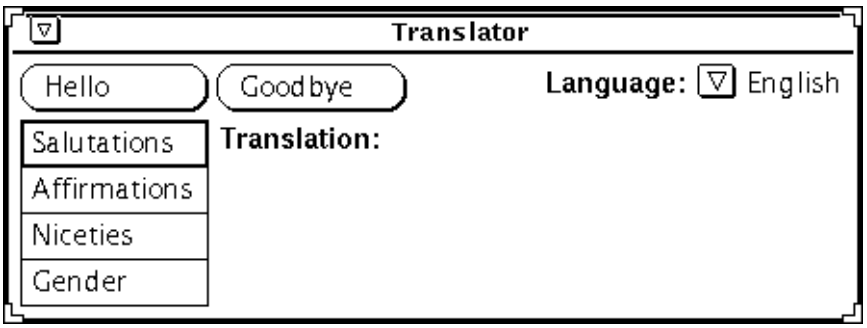

Figure 4-1 Translator

Translator translates some common English words into several different languages. Here's how to use it:

1. Selected a language by pulling down the abbreviated menu button and highlighting the desired language. The language will be displayed next to the abbreviated menu button.
2. Select the type of words you wish to translate by pressing the Salutations, Affirmations, Niceties, or Gender buttons.
3. Press one of the Oblong buttons for a translation of the desired word. The translated word will appear next to the label Translation:.

Table 4-1 lists the variables in `Translator.c`. Code Example 4-1 shows the resource file for `Translator.c`. Code Example 4-2 lists the source code.

1. Note that if you are not using OpenWindows 3.2 or Asian OpenWindows 3.1, you may have trouble compiling this program unless you remove the line which reads OlToolkitInitialize((XtPointer)NULL);

Table 4-1 Translator.c Variable List

Variable	Type	Description
abbrevmb	AbbrevMenuButton	Pulldown menu button for the Language buttons.
answer	StaticText	Foreign language translation of pressed button.
control_area	ControlArea	Manager widget.
goodbye_button	OblongButton	Right word button.
hello_button	OblongButton	Left word button.
holder	Exclusives	An Exclusives widget used to hold the RectButtons.
i	int	Counter.
israel[]	char	Israeli language button labels. Yiddish and Hebrew.
israeli_buttons	OblongButtons	Yiddish and Hebrew selection button widgets.
israeli_menu	MenuButton	Israeli MenuButton widget.
isrl_menu	MenuShell	,MenuShell subwidget of israeli_menu.
lang_button_cb()	void	Callback procedure called when a language (lang_button) is selected. Puts the name of the language next to the AbbreviatedMenuButton.
lang_buttons[]	char	Labels for the language selection buttons. English, Spanish, Japanese, German, French, Swahili
language_buttons[]	OblongButtons	English, Spanish, Japanese, German, French, Swahili selection button widgets.
lang_label	StaticText	Language: label to the left of the abbreviated menu button.
lang_menu	MenuShell	MenuShell subwidget of abbrevmb.
new_header_cb()	void	Callback procedure called when the hello and goodbye button are selected. Sets application header to the string in the Hello/Goodbye buttons.
opposites[]	char	Names of the word pairs. Salutations, Affirmations, Niceties, Gender.
opposites_cb()	void	Callback procedure called when an opposites button (Salutations, Affirmations, Niceties, and Gender) is selected. When button is selected, the corresponding word pair is displayed in the Hello and Goodbye buttons.

Table 4-1 Translator.c Variable List

Variable	Type	Description
Opposites[]	RectButtons	Opposites selection buttons. Salutations, Affirmations, Niceties, Gender.
opposites[]	char	Opposites selection button labels. Salutations, Affirmations, Niceties, Gender.
preview_lang	StaticText	Previews the selected language.
spacer_button	OblongButton	Used to create a blank space between the Goodbye button and "Language".
toplevel	TopLevelShell	Top level shell widget.
translation	StaticText	Static label. "Translations:"
translation_cb()	void	Callback executed when a word button is pressed. Matches word and language. Prints out the foreign language translation.

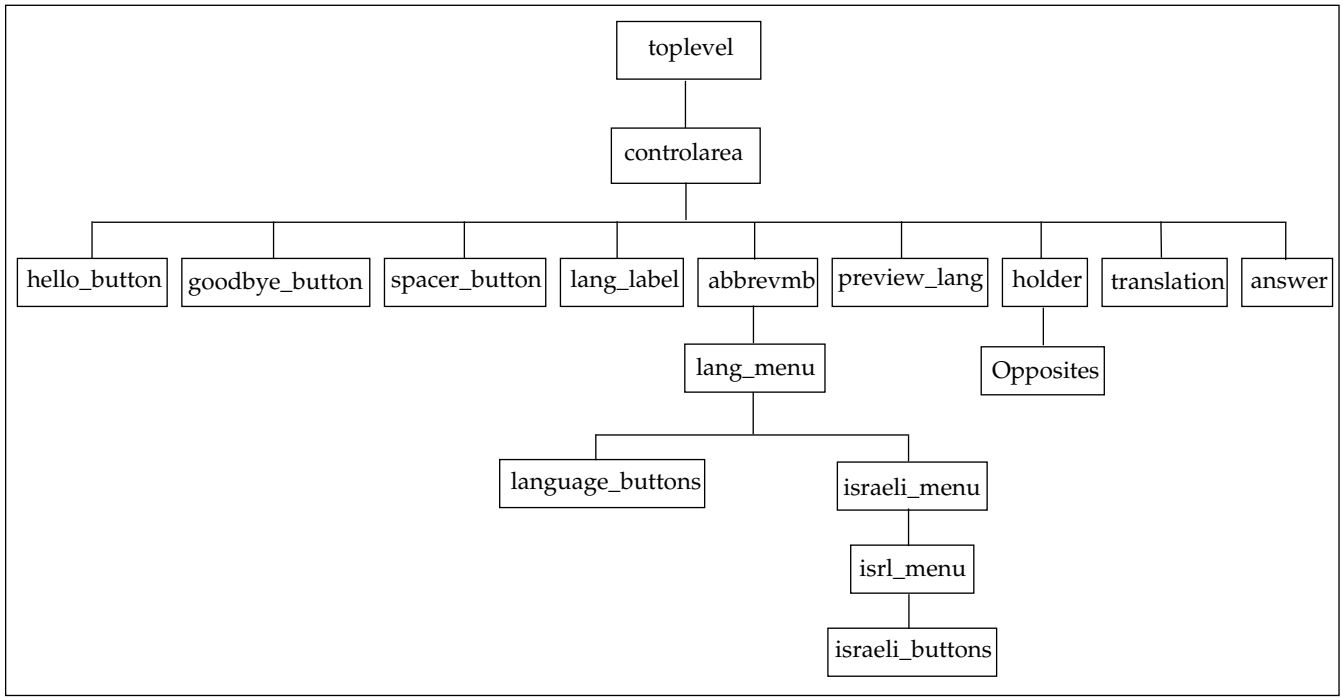

Figure 4-2 Translator.c Widget Tree Diagram

Code Example 4-1 Resource File for `Translator.c` (`Resources_translator`)

```
#######  Default Resource File  #######

hello.control_area.spacer_button.label: space
hello.control_area.spacer_button.mappedWhenManaged: FALSE
hello.control_area.lang_label.string: Language
hello.control_area.translation.string: Translation:
hello.control_area.preview_lang.string: English

*lang_label.Font: -adobe-helvetica-bold-r-normal--14-140-75-75-p-82-iso8859-1
*translation.Font: -adobe-helvetica-bold-r-normal--14-140-75-75-p-82-iso8859-1

hello.control_area.layoutType: fixedcols
hello.control_area.measure: 6

hello*holder.layoutType: fixedcols
hello*holder.measure: 1

*background:            grey85
*fontColor:             blue
*borderColor:           black
*inputFocusColor:       red
*traversalOn:           false
```

Code Example 4-2 `Translator.c`

```
/*  Translator.c program */

/* Required OLIT Headers */
#include <X11/Intrinsic.h>
#include <X11/StringDefs.h>
#include <X11/Shell.h>include
#include <Xol/OpenLook.h>

/* Specific Widget Headers */
#include <Xol/ControlAre.h>
#include <Xol/OblongButt.h>
#include <Xol/MenuButton.h>
#include <Xol/StaticText.h>
#include <Xol/AbbrevMenu.h>
#include <Xol/Exclusives.h>
```

Code Example 4-2 `Translator.c`

```
#include <Xol/RectButton.h>

/* The following variables are global because they are used in the callback  functions, which are outside
 * of main().*/

static Widget preview_lang, hello_button, goodbye_button, answer;
char *lang_buttons[] = { "English", "Spanish", "Japanese", "German",
                         "French", "Swahili" };

char *israel[] = {"Yiddish", "Hebrew"};
char *opposites[] = { "Salutations", "Affirmations","Niceties", "Gender"};

main(argc,argv)
    int     argc;
    char    *argv[];
{
    XtAppContext  app;
    Widget  toplevel, control_area,
        abbrevmb, language_buttons[XtNumber(lang_buttons)],
        israeli_buttons[XtNumber(israel)], lang_menu,
        israeli_menu, isrl_menu, lang_label, spacer_button, holder,
        Opposites[XtNumber(opposites)], translation;
    int i;
    void translation_cb();
    void new_header_cb();
    void lang_button_cb(), opposites_cb();

  /* Initialize the toolkit and create the toplevel shell.  OlSetDefaultTextFormat is for
   * internationalization and will only work with OpenWindows 3.2 or Asian OpenWindows
   * 3.1. If you are not using one of these versions, comment out this line.  */

    OlToolkitInitialize((XtPointer)NULL);
    toplevel = XtVaAppInitialize(&app, "hello",
        (XrmOptionDescList)NULL, 0,
        &argc, argv,
        (String *)NULL,
        NULL);

/* Create application's manager widget.  The ControlArea widget class lays out  children
 * widgets left to right in order of creation.  Thus, the first widget displays on the left, the
 * 2nd to the right of the 1st, etc.  If the layout is  specified as vertical, child widgets are
 * displayed top to bottom in order of  creation. */

    control_area = XtVaCreateManagedWidget( "control_area",
```

Code Example 4-2 `Translator.c`

```
        controlAreaWidgetClass, toplevel,
        NULL);

/* Create hello button widget. */

    hello_button = XtVaCreateManagedWidget( "hello_button",
        oblongButtonWidgetClass, control_area,
        XtNlabel, "Hello",
        NULL);

/* Attach callback procedures to the hello_button. */

    XtAddCallback(hello_button, XtNselect, translation_cb, NULL);
    XtAddCallback(hello_button, XtNselect, new_header_cb, toplevel);

/* Create goodbye button widget */

    goodbye_button = XtVaCreateManagedWidget( "button2",
        oblongButtonWidgetClass, control_area,
        XtNlabel, "Goodbye",
        NULL);

/* Attach callback procedures to the goodbye_button. */

    XtAddCallback(goodbye_button, XtNselect, translation_cb, NULL);
    XtAddCallback(goodbye_button, XtNselect, new_header_cb, toplevel);

/* Create spacer button widget. This button is used to put a blank space between the goodbye
 * button and "Language:" StaticText widget. This button takes up space  but is not visible.
 * We make the button invisible by setting the OblongButton resource mappedWhenManaged to
 *  FALSE in the resources file.  */

    spacer_button = XtVaCreateManagedWidget( "spacer_button",
        oblongButtonWidgetClass, control_area,
        NULL);

/*  Create "Language:" statictext widget.  */

    lang_label = XtVaCreateManagedWidget("lang_label", staticTextWidgetClass,
            control_area,
            XtNstring, "Language:",
            NULL);

/* Create AbbrevMenuButton widget. This widget allows us to pull down a menu with
```

Code Example 4-2 `Translator.c`

```
 * all the language selection buttons.  */

    abbrevmb = XtVaCreateManagedWidget("abbrevmb", abbrevMenuButtonWidgetClass,
                                       control_area, NULL);

/* The AbbrevMenuButton widget is a composite widget.  That is, other widgets,
 * called subwidgets, are attached to, and part of, this widget. An AbbrevMenuButton
 * widget consists of a triangular Abbreviated Menu Button, a Current Selection
 * Widget, and a MenuShell widget. In the line below, we get a pointer to abbrevmb's
 * MenuShell widget, and assign it to lang_menu. */

    XtVaGetValues(abbrevmb, XtNmenuPane, &lang_menu, NULL);

/* Create a language preview statictext widget that will preview the language
 * selected. */

    preview_lang = XtVaCreateManagedWidget("preview_lang", staticTextWidgetClass,
                           control_area, NULL);

/* Set abbrevmb's Current Selection Widget (resource XtNpreviewWidget) to the value
 * of preview_lang. */

    XtVaSetValues(abbrevmb, XtNpreviewWidget, preview_lang, NULL);

/* Create language selection buttons (OblongButton widgets). Put language labels on
 * the buttons. XtNumber() counts the number of members in an array.  Note that if no
 * button label resource (XtNlabel) is specified in the code or a resource file, the
 * widget name is used as the label.  Thus, all our language buttons are labeled
 * with their respective widget names.  */

    for(i=0;i<XtNumber(lang_buttons);i++) {
        language_buttons[i] = XtVaCreateManagedWidget(lang_buttons[i],
        oblongButtonWidgetClass, lang_menu, NULL);

/* Attach a callback such that when the language button is pressed, the selected
 * language is displayed next to the Abbreviated Menu Button. */

     XtAddCallback(language_buttons[i], XtNselect, lang_button_cb, lang_buttons[i]);
  }

/* Create an Israeli MenuButton widget. When this button is selected then dragged
 * to the right, two more language buttons, Yiddish and Hebrew, are displayed.  */

    israeli_menu = XtVaCreateManagedWidget("Israeli", menuButtonWidgetClass,
```

Code Example 4-2 `Translator.c`

```
                                    lang_menu, NULL);

/* The MenuButton widget is another composite widget.  In addition to the menu button,
 * it also consists of a triangular mark, and a MenuShell widget. The MenuShell is
 * accessed by selecting the button, then dragging the mouse to the right or down.
 * In the line below, we get a pointer to israeli_menu's MenuShell widget, and assign
 * it to isrl_menu. */

  XtVaGetValues(israeli_menu, XtNmenuPane, &isrl_menu, NULL);

/* Create two Israeli language OblongButtons, "Hebrew" and "Yiddish" in the MenuButton
 * widget. Attach a callback such that when the Hebrew or Yiddish buttons are pressed,
 * the selected language is displayed next to the Abbreviated Menu Button.   Again, if no
 * button label resource (XtNlabel) is specified in the code or a resource file, the
 * widget name is used as the label.  Thus, all our Israeli language buttons are labeled
 * with their respective widget names. */

    for(i=0;i<XtNumber(israel);i++) {
        israeli_buttons[i] = XtVaCreateManagedWidget(israel[i],
           oblongButtonWidgetClass,
           isrl_menu, NULL);
    XtAddCallback(israeli_buttons[i], XtNselect, lang_button_cb, israel[i]);
  }

/* Create an Exclusives Widget to contain our four RectButtons (Salutations,
 * Affirmations, Niceties, and Gender).  Buttons in the Exclusives widget are
 * exclusive.  That is, only one button can be selected in an exclusives widget. */

     holder = XtVaCreateManagedWidget("holder", exclusivesWidgetClass,
                          control_area, NULL);

/* Add 4 RectButton widgets to the exclusives widget.  Attach a callback such that
 * when a button is selected, the corresponding word pair is displayed in the
 * hello and goodbye buttons. Again, the button labels take on the names of the
 * buttons, so there is no need to explicitly set the label resource. */

    for(i=0;i<4;i++) {
       Opposites[i] = XtVaCreateManagedWidget(opposites[i], rectButtonWidgetClass,
                holder, NULL);
    XtAddCallback(Opposites[i], XtNselect, opposites_cb, NULL);
  }

/* Create a statictext widget called "Translation:"  */
```

Code Example 4-2 `Translator.c`

```
    translation = XtVaCreateManagedWidget("translation", staticTextWidgetClass,
            control_area, NULL);

/* Create a statictext widget that displays the foreign language translation of the
 * selected button. */

    answer = XtVaCreateManagedWidget("answer", staticTextWidgetClass,
            control_area,
            NULL);

/* realize the widgets and begin the main loop */

    XtRealizeWidget(toplevel);
    XtAppMainLoop(app);
}

/*  CALLBACKS!!! */

/*  Procedure called when the hello and goodbye button are selected. Sets application header
 *  to the string in the hello/goodbye buttons.   */

void new_header_cb(widget, client_data, call_data)
    Widget     widget;
    XtPointer  client_data, call_data;
{
    String label;
    XtVaGetValues(widget,
        XtNlabel, &label,
        NULL);
    XtVaSetValues(client_data,
        XtNtitle, label,
        NULL);
}

/* Procedure called when a language (lang_button) is selected. Puts the name of the
 * language next to the AbreviatedMenuButton. */

void lang_button_cb(widget, client_data, call_data)
   Widget      widget;
   XtPointer   client_data, call_data;
{
    XtVaSetValues(preview_lang, XtNstring, client_data, NULL);
}
```

Code Example 4-2 `Translator.c`

```
/* Procedure called when an opposites button (Salutations, Affirmations, Niceties, and Gender)
 * is selected. When a button is selected, the corresponding word pair is displayed in the
 * hello and goodbye buttons. */

void opposites_cb(widget, client_data, call_data)
    Widget     widget;
    XtPointer  client_data, call_data;
{
    char *word;
    XtVaGetValues(widget,
        XtNlabel, &word,
        NULL);

   if (strcmp(word, "Affirmations") == 0) {
      XtVaSetValues(hello_button,
        XtNlabel, "Yes",
        NULL);
      XtVaSetValues(goodbye_button,
        XtNlabel, "No",
        NULL);
    }
   else if (strcmp(word, "Salutations") == 0) {
      XtVaSetValues(hello_button,
        XtNlabel, "Hello",
        NULL);
      XtVaSetValues(goodbye_button,
        XtNlabel, "Goodbye",
        NULL);
    }
   else if (strcmp(word, "Niceties") == 0) {
      XtVaSetValues(hello_button,
        XtNlabel, "Please",
        NULL);
      XtVaSetValues(goodbye_button,
        XtNlabel, "Thank you",
        NULL);
    }
  else
{
      XtVaSetValues(hello_button,
        XtNlabel, "Sir",
        NULL);
      XtVaSetValues(goodbye_button,
        XtNlabel, "Madam",
```

Code Example 4-2 `Translator.c`

```
        NULL);
    }

}

/* Translations callback. Matches a word and language to print out the foreign language
 * translation. */

void translation_cb(widget, client_data, call_data)
     Widget        widget;
     XtPointer     client_data;
     XtPointer     call_data;
{
  char *word,*lang;
  XtVaGetValues(widget,
    XtNlabel, &word,
    NULL);
  XtVaGetValues(preview_lang,
    XtNstring, &lang,
    NULL);

 /*  English printout commands.  */

  if  ((strcmp(word, "Yes") == 0) && (strcmp(lang, "English")) == 0)
      XtVaSetValues(answer,
        XtNstring, "Yes",
        NULL);
  if  ((strcmp(word, "No") == 0) && (strcmp(lang, "English")) == 0)
      XtVaSetValues(answer, XtNstring, "No", NULL);
  if  ((strcmp(word, "Hello") == 0) && (strcmp(lang, "English")) == 0)
      XtVaSetValues(answer, XtNstring, "Hello!", NULL);
  if  ((strcmp(word, "Goodbye") == 0) && (strcmp(lang, "English")) == 0)
      XtVaSetValues(answer, XtNstring, "Goodbye!", NULL);
  if  ((strcmp(word, "Thank you") == 0) && (strcmp(lang, "English")) == 0)
      XtVaSetValues(answer, XtNstring, "Thank you", NULL);
  if  ((strcmp(word, "Please") == 0) && (strcmp(lang, "English")) == 0)
      XtVaSetValues(answer, XtNstring, "Please", NULL);
  if  ((strcmp(word, "Sir") == 0) && (strcmp(lang, "English")) == 0)
      XtVaSetValues(answer, XtNstring, "Sir", NULL);
  if  ((strcmp(word, "Madam") == 0) && (strcmp(lang, "English")) == 0)
      XtVaSetValues(answer, XtNstring, "Madam", NULL);

/*  Translation Tables.  When word and language button match is found,
 *  the English translation is printed on screen */
```

Code Example 4-2 `Translator.c`

```
/* Spanish translations. */

  if  ((strcmp(word, "Yes") == 0) && (strcmp(lang, "Spanish")) == 0)
      XtVaSetValues(answer, XtNstring, "Sí", NULL);
  if  ((strcmp(word, "No") == 0) && (strcmp(lang, "Spanish")) == 0)
      XtVaSetValues(answer, XtNstring, "No", NULL);
  if  ((strcmp(word, "Hello") == 0) && (strcmp(lang, "Spanish")) == 0)
      XtVaSetValues(answer, XtNstring, "Hola!", NULL);
  if  ((strcmp(word, "Goodbye") == 0) && (strcmp(lang, "Spanish")) == 0)
      XtVaSetValues(answer, XtNstring, "Adiós!", NULL);
  if  ((strcmp(word, "Thank you") == 0) && (strcmp(lang, "Spanish")) == 0)
      XtVaSetValues(answer, XtNstring, "Gracias", NULL);
  if  ((strcmp(word, "Please") == 0) && (strcmp(lang, "Spanish")) == 0)
      XtVaSetValues(answer, XtNstring, "Por Favor!", NULL);
  if  ((strcmp(word, "Sir") == 0) && (strcmp(lang, "Spanish")) == 0)
      XtVaSetValues(answer, XtNstring, "Señor", NULL);
  if  ((strcmp(word, "Madam") == 0) && (strcmp(lang, "Spanish")) == 0)
      XtVaSetValues(answer, XtNstring, "Señora", NULL);

/* Japanese translations. */

  if  ((strcmp(word, "Yes") == 0) && (strcmp(lang, "Japanese")) == 0)
      XtVaSetValues(answer, XtNstring, "Hai", NULL);
  if  ((strcmp(word, "No") == 0) && (strcmp(lang, "Japanese")) == 0)
      XtVaSetValues(answer, XtNstring, "iie", NULL);
  if  ((strcmp(word, "Hello") == 0) && (strcmp(lang, "Japanese")) == 0)
      XtVaSetValues(answer, XtNstring, "Moshi-moshi!", NULL);
  if  ((strcmp(word, "Goodbye") == 0) && (strcmp(lang, "Japanese")) == 0)
      XtVaSetValues(answer, XtNstring, "Sayonara", NULL);
  if  ((strcmp(word, "Thank you") == 0) && (strcmp(lang, "Japanese")) == 0)
      XtVaSetValues(answer, XtNstring, "Domo arrigato", NULL);
  if  ((strcmp(word, "Please") == 0) && (strcmp(lang, "Japanese")) == 0)
      XtVaSetValues(answer, XtNstring, "Dozo", NULL);
  if  ((strcmp(word, "Sir") == 0) && (strcmp(lang, "Japanese")) == 0)
      XtVaSetValues(answer, XtNstring, "-san appended to name, e.g., \
      Honorable Clinton-san", NULL);
  if  ((strcmp(word, "Madam") == 0) && (strcmp(lang, "Japanese")) == 0)
      XtVaSetValues(answer, XtNstring, "-san appended to name, e.g., \
      Honorable Hillery-san", NULL);

/* German translations. */

  if  ((strcmp(word, "Yes") == 0) && (strcmp(lang, "German")) == 0)
```

Code Example 4-2 `Translator.c`

```
        XtVaSetValues(answer, XtNstring, "Ja", NULL);
    if  ((strcmp(word, "No") == 0) && (strcmp(lang, "German")) == 0)
        XtVaSetValues(answer, XtNstring, "nein", NULL);
    if  ((strcmp(word, "Hello") == 0) && (strcmp(lang, "German")) == 0)
        XtVaSetValues(answer, XtNstring, "Hallo!", NULL);
    if  ((strcmp(word, "Goodbye") == 0) && (strcmp(lang, "German")) == 0)
        XtVaSetValues(answer, XtNstring, "Auf wiedersehen!", NULL);
    if  ((strcmp(word, "Thank you") == 0) && (strcmp(lang, "German")) == 0)
        XtVaSetValues(answer, XtNstring, "Dankeschön", NULL);
    if  ((strcmp(word, "Please") == 0) && (strcmp(lang, "German")) == 0)
        XtVaSetValues(answer, XtNstring, "Bitte", NULL);
    if  ((strcmp(word, "Sir") == 0) && (strcmp(lang, "German")) == 0)
        XtVaSetValues(answer, XtNstring, "Herr", NULL);
    if  ((strcmp(word, "Madam") == 0) && (strcmp(lang, "German")) == 0)
        XtVaSetValues(answer, XtNstring, "Frau", NULL);

/* French translations. */

    if  ((strcmp(word, "Yes") == 0) && (strcmp(lang, "French")) == 0)
        XtVaSetValues(answer, XtNstring, "Oui!", NULL);
    if  ((strcmp(word, "No") == 0) && (strcmp(lang, "French")) == 0)
        XtVaSetValues(answer, XtNstring, "Non!", NULL);
    if  ((strcmp(word, "Hello") == 0) && (strcmp(lang, "French")) == 0)
        XtVaSetValues(answer, XtNstring, "Allo!", NULL);
    if  ((strcmp(word, "Goodbye") == 0) && (strcmp(lang, "French")) == 0)
        XtVaSetValues(answer, XtNstring, "Adieu!", NULL);
    if  ((strcmp(word, "Thank you") == 0) && (strcmp(lang, "French")) == 0)
        XtVaSetValues(answer, XtNstring, "Merci!", NULL);
    if  ((strcmp(word, "Please") == 0) && (strcmp(lang, "French")) == 0)
        XtVaSetValues(answer, XtNstring, "S'il vous plâit", NULL);
    if  ((strcmp(word, "Sir") == 0) && (strcmp(lang, "French")) == 0)
        XtVaSetValues(answer, XtNstring, "Monsieur", NULL);
    if  ((strcmp(word, "Madam") == 0) && (strcmp(lang, "French")) == 0)
        XtVaSetValues(answer, XtNstring, "Madame", NULL);

/* Swahili translations. */

    if  ((strcmp(word, "Yes") == 0) && (strcmp(lang, "Swahili")) == 0)
        XtVaSetValues(answer, XtNstring, "Naam", NULL);
    if  ((strcmp(word, "No") == 0) && (strcmp(lang, "Swahili")) == 0)
        XtVaSetValues(answer, XtNstring, "Siyo", NULL);
    if  ((strcmp(word, "Hello") == 0) && (strcmp(lang, "Swahili")) == 0)
        XtVaSetValues(answer, XtNstring, "Jambo!", NULL);
    if  ((strcmp(word, "Goodbye") == 0) && (strcmp(lang, "Swahili")) == 0)
```

Code Example 4-2 `Translator.c`

```
        XtVaSetValues(answer, XtNstring, "Kwaheri!", NULL);
    if  ((strcmp(word, "Thank you") == 0) && (strcmp(lang, "Swahili")) == 0)
        XtVaSetValues(answer, XtNstring, "Asante", NULL);
    if  ((strcmp(word, "Please") == 0) && (strcmp(lang, "Swahili")) == 0)
        XtVaSetValues(answer, XtNstring, "Pendeza", NULL);
    if  ((strcmp(word, "Sir") == 0) && (strcmp(lang, "Swahili")) == 0)
        XtVaSetValues(answer, XtNstring, "Bwana", NULL);
    if  ((strcmp(word, "Madam") == 0) && (strcmp(lang, "Swahili")) == 0)
        XtVaSetValues(answer, XtNstring, "Bibi mkubwa", NULL);

/* Yiddish translations. */

    if  ((strcmp(word, "Yes") == 0) && (strcmp(lang, "Yiddish")) == 0)
        XtVaSetValues(answer, XtNstring, "Ya", NULL);
    if  ((strcmp(word, "No") == 0) && (strcmp(lang, "Yiddish")) == 0)
        XtVaSetValues(answer, XtNstring, "Neyn", NULL);
    if  ((strcmp(word, "Hello") == 0) && (strcmp(lang, "Yiddish")) == 0)
        XtVaSetValues(answer, XtNstring, "Sholom a ley-khem!", NULL);
    if  ((strcmp(word, "Goodbye") == 0) && (strcmp(lang, "Yiddish")) == 0)
        XtVaSetValues(answer, XtNstring, "A gut-n tog!", NULL);
    if  ((strcmp(word, "Thank you") == 0) && (strcmp(lang, "Yiddish")) == 0)
        XtVaSetValues(answer, XtNstring, "Zayt a-zoy gut", NULL);
    if  ((strcmp(word, "Please") == 0) && (strcmp(lang, "Yiddish")) == 0)
        XtVaSetValues(answer, XtNstring, "A dank", NULL);
    if  ((strcmp(word, "Sir") == 0) && (strcmp(lang, "Yiddish")) == 0)
        XtVaSetValues(answer, XtNstring, "Givar", NULL);
    if  ((strcmp(word, "Madam") == 0) && (strcmp(lang, "Yiddish")) == 0)
        XtVaSetValues(answer, XtNstring, "Givaret", NULL);

/* Hebrew translations. */

    if  ((strcmp(word, "Yes") == 0) && (strcmp(lang, "Hebrew")) == 0)
        XtVaSetValues(answer, XtNstring, "Ken", NULL);
    if  ((strcmp(word, "No") == 0) && (strcmp(lang, "Hebrew")) == 0)
        XtVaSetValues(answer, XtNstring, "Lo", NULL);
    if  ((strcmp(word, "Hello") == 0) && (strcmp(lang, "Hebrew")) == 0)
        XtVaSetValues(answer, XtNstring, "Shah-lom!", NULL);
    if  ((strcmp(word, "Goodbye") == 0) && (strcmp(lang, "Hebrew")) == 0)
        XtVaSetValues(answer, XtNstring, "Lehitra-ot!", NULL);
    if  ((strcmp(word, "Thank you") == 0) && (strcmp(lang, "Hebrew")) == 0)
        XtVaSetValues(answer, XtNstring, "Bevakasha", NULL);
    if  ((strcmp(word, "Please") == 0) && (strcmp(lang, "Hebrew")) == 0)
        XtVaSetValues(answer, XtNstring, "Toda raba", NULL);
    if  ((strcmp(word, "Sir") == 0) && (strcmp(lang, "Hebrew")) == 0)
```

Code Example 4-2 `Translator.c`

```
        XtVaSetValues(answer, XtNstring, "Adon", NULL);
    if  ((strcmp(word, "Madam") == 0) && (strcmp(lang, "Hebrew")) == 0)
        XtVaSetValues(answer, XtNstring, "Geveret", NULL);
}
```