```
/*-------------  SWEDISH.C -- ----------*/
#include <class.h>
#include <andrewos.h>
#include <proctbl.ih>
#include <swedish.eh>
#include <textv.ih>
#include <text.ih>
int (*textview_SelfInsertCmd)();
/* routines for entering characters */
void swedish_odsinsert(tv) struct textview *tv;
{   textview_SelfInsertCmd(tv,246);}
void swedish_odcinsert(tv) struct textview *tv;
{   textview_SelfInsertCmd(tv,214);}
void swedish_adsinsert(tv) struct textview *tv;
{   textview_SelfInsertCmd(tv,228);}
void swedish_adcinsert(tv) struct textview *tv;
{   textview_SelfInsertCmd(tv,196);}
void swedish_aosinsert(tv) struct textview *tv;
{   textview_SelfInsertCmd(tv,229);}
void swedish_aocinsert(tv) struct textview *tv;
{   textview_SelfInsertCmd(tv,197);}
/* return the position of the document to act upon */
static getSandL(t,tv,start,len)
struct text *t;
struct textview *tv;
long *start,*len;
{
   if((*len = textview_GetDotLength(tv)) >  0)
          *start = textview_GetDotPosition(tv) ;
   else {  /* process whole document */
          *start = 0;
          *len = text_GetLength(t);
   }
}
/* Conversion table */
static unsigned char conv[] = {
   '\\',   (unsigned char) 214,
   '{',    (unsigned char) 228,
   '}',    (unsigned char) 229,
   '[',    (unsigned char) 196,
   ']',    (unsigned char) 197,
   '|'    ,   (unsigned char) 246,
   0,0 };
/* convert to use iso characters */
```

```
void swedish_converttoATK(tv)
struct textview *tv;
{
    struct text *t;
    long start, len;
    register unsigned char c, *cp;
    long rep = 0;
    t = (struct text *) textview_GetDataObject(tv);
    getSandL(t,tv,&start,&len);
    for(;len; len--,start++){
            c = text_GetChar(t,start);
            for(cp = conv; *cp != 0; cp++){
                if(*cp++ == c){
                        text_AlwaysReplaceCharacters(t,start,1,cp,1);
                        rep++;
                }
            }
    }
    if(rep != 0)text_NotifyObservers(t,0);
}
/* convert to use local conventions */
void swedish_convertfromATK(tv)
struct textview *tv;
{
    struct text *t;
    long start, len;
    register unsigned char c, *cp;
    long rep = 0;
    t = (struct text *) textview_GetDataObject(tv);
    getSandL(t,tv,&start,&len);
    for(;len; len--,start++){
            c = text_GetChar(t,start);
            for(cp = conv + 1; *cp != 0; cp++){
                if(*cp++ == c){
                        text_AlwaysReplaceCharacters(t,start,1,cp - 2,1);
                        rep++;
                }
            }
    }
    if(rep != 0 )text_NotifyObservers(t,0);
}
/* provide conversions for printing */
char * swedish_converttotroff(i)
unsigned char i;
```

```
{
   static char buf[6];
   register unsigned char *cp;
   if(i == 214){/* special case backslash, since it has to be quoted */
         *buf = '\\'; buf[1] = '\\'; buf[2] = '\0';
         return buf;
   }
   buf[1] = '\0';
   for(cp = conv + 3; *cp != 0; cp++){
         if(*cp++ == i){
            *buf = *(cp - 2);
            return buf;
         }
   }
   sprintf(buf,"\\\\%3.3o",i);
   return buf;
}
boolean swedish__InitializeClass(ClassID)
struct classheader *ClassID;
{
   struct classinfo *textviewtype = class_Load("textview");
   struct proctable_Entry *pr;
   int res;
   /* import the textview routine for entering characters */
   if((pr = proctable_Lookup("textview-self-insert")) != NULL &&
proctable_Defined(pr) ){
         textview_SelfInsertCmd = proctable_GetFunction(pr) ;
   }
   /* export routines */
   proctable_DefineProc("swedish-o-diaeresis-small",swedish_odsinsert,
                           textviewtype,NULL,"insert small diaeresis o");
   proctable_DefineProc("swedish-o-diaeresis-cap",swedish_odcinsert,
                           textviewtype,NULL,"insert cap diaeresis o");
   proctable_DefineProc("swedish-a-diaeresis-small",swedish_adsinsert,
                           textviewtype,NULL,"insert small diaeresis a");
   proctable_DefineProc("swedish-a-diaeresis-cap",swedish_adcinsert,
                           textviewtype,NULL,"insert cap diaeresis a");
   proctable_DefineProc("swedish-a-overcircle-small",swedish_aosinsert,
                           textviewtype,NULL,"insert small overcircle a");
   proctable_DefineProc("swedish-a-overcircle-cap",swedish_aocinsert,
                           textviewtype,NULL,"insert cap overcircle a");
   proctable_DefineProc("swedish-convert-to-ATK",swedish_converttoATK,
                           textviewtype,NULL,"Convert to Iso characters");
   proctable_DefineProc("swedish-convert-from-ATK", swedish_convertfromATK,
```

```
                                textviewtype,NULL,"Convert from Iso characters");
    proctable_DefineProc("swedish-convert-to-troff", swedish_converttotroff,
                                &swedish_classinfo,NULL,"return troff string");

    return TRUE;
}
/*----------- isomsg.c -----------*/
........ program generate code deleted .............
#define BUTTONFILE environ_AndrewDir("/lib/arbiters/mb.lset")
int (*cfromiso)();
int (*ctoiso)();
static struct isomsg *FindSelf();
static struct view *NewWindow(filename,bflags)
char *filename;
int bflags;
{
    struct frame *newFrame;
    struct im *window;
    struct buffer *buffer;
    /* create a new buffer on the given file */
    if((buffer = buffer_GetBufferOnFile(filename,buffer_ReadOnly
                                        | buffer_MustExist)) == NULL){
        char buf[1300];
        sprintf(buf,"ERROR: can’t create %s",filename);
        message_DisplayString(NULL,0,buf);
        return NULL;
    }
    /* create a frame and window  to hold the buffer */
    if ((newFrame = frame_New()) != NULL)
        if ((window = im_Create(NULL)) != NULL) {
        /* insert the frame in the window , enable the menu commands, add the message
handler and put the frame in the buffer */
            im_SetView(window, newFrame);
            frame_SetCommandEnable(newFrame,TRUE);
            frame_PostDefaultHandler(newFrame, "message",
                                frame_WantHandler(newFrame, "message"));
            frame_SetBuffer(newFrame, buffer, TRUE);
        }
        else{/* window creation failed , destroy the frame */
            frame_Destroy(newFrame);
            newFrame = NULL;
        }
    /* return the created frame */
    return (struct view *) newFrame;
```

```
}
isomsg_Start(v,dat)
struct view *v;
long dat;
{
   struct isomsg *self;
   struct sendmessage *mes;
   if(!class_IsTypeByName(class_GetTypeName(v),"sendmessage")){
         /* Error, must be called from a sendmessage object */
         message_DisplayString(NULL,0,"Must be called from sendmessages");
         return;
   }
   /* keep a pointer to the passed in sendmessage view */
   mes = (struct sendmessage *) v;
   /* create a new 150x200 window containing the ADEW created buttons */
   im_SetGeometrySpec("150x200");
   v = NewWindow(BUTTONFILE,0);
   if(v == NULL) return;/* window creation failed */
   /* call FindSelf to associate a new isomsg controller object with the buttons in the
frame */
   if((self = FindSelf(v)) == NULL){
         message_DisplayString(NULL,0,"Can’t init source file");
         return;
   }
   /* store a pointer to the sendmessage view in the isomsg object*/
   self->mess = mes;
   /* if the sendmessage object is destroyed, we need to be informed, so that we can
delete our pointer to it */
   sendmessage_AddObserver(mes,self);
}
........ program generate code deleted .............
static void convert_to_isoCallBack(self,val,r1,r2)
struct isomsg *self;
struct value *val;
long r1,r2;
{
if(r2 == value_OBJECTDESTROYED) {
         if(self->convert_to_iso == val) self->convert_to_iso = NULL;
}
{
/* user code begins here for convert_to_isoCallBack */
if(ctoiso != NULL && self->mess != NULL && self->mess->BodyTextview !=
NULL){
   ctoiso(self->mess->BodyTextview);
```

```
}
/* user code ends here for convert_to_isoCallBack */
}
}
static void convert_from_isoCallBack(self,val,r1,r2)
struct isomsg *self;
struct value *val;
long r1,r2;
{
if(r2 == value_OBJECTDESTROYED) {
        if(self->convert_from_iso == val) self->convert_from_iso = NULL;
}
{
/* user code begins here for convert_from_isoCallBack */
if(cfromiso != NULL && self->mess != NULL && self->mess->BodyTextview !=
NULL){
   cfromiso(self->mess->BodyTextview);
}
/* user code ends here for convert_from_isoCallBack */
}
}
static void cvtCallBack(self,val,r1,r2)
struct isomsg *self;
struct value *val;
long r1,r2;
........ program generate code deleted .............
{
/* user code begins here for cvtCallBack */
struct textview *tv;
unsigned char st[1];
struct text *t;
if(self->mess){
   /* get pointers to the text and the textview that the sendmessageobject uses to store the
body of the message. */
   if((tv = self->mess->BodyTextview) == NULL) return;
   if((t = self->mess->BodyText) == NULL) return;
   /* since we stored the numeric value of the iso character as the second half of the
logical inset name,
          that number is passed as r1.*/
   *st = (unsigned char) r1;
   /* insert that string as the dot postion, notify the text’s observers that it has changed,
and advance the dot position */
   text_InsertCharacters(t,textview_GetDotPosition(tv),st,1);
   text_NotifyObservers(t,0);
```

```
    textview_SetDotPosition(tv,textview_GetDotPosition(tv) + 1);
}
/* user code ends here for cvtCallBack */
........ program generate code deleted .............
boolean isomsg__InitializeClass(ClassID)
struct classheader *ClassID;
{
struct classinfo *viewtype = class_Load("view");
firstisomsg = NULL;
proctable_DefineProc("isomsg-go",isomsg_go, viewtype,NULL,"isomsg go");
/* user code begins here for InitializeClass */
{
    struct proctable_Entry *pr;
    /* load the swedish class */
    class_Load("swedish");
    /* export the start routine so that it can be called from sendmessage */
    proctable_DefineProc("isomsg-start",isomsg_Start, viewtype,NULL,"isomsg start");
    cfromiso = ctoiso = NULL;
    /* import the swedish conversion routines */
    if((pr = proctable_Lookup("swedish-convert-from-ATK")) != NULL
            && proctable_Defined(pr) ){
            cfromiso = proctable_GetFunction(pr) ;
    }
    if((pr = proctable_Lookup("swedish-convert-to-ATK")) != NULL
            && proctable_Defined(pr) ){
            ctoiso = proctable_GetFunction(pr) ;
    }
}
/* user code ends here for InitializeClass */
return TRUE;
}
........ program generate code deleted .............
```