Differences Between Zinc 4.2 and OpenZinc 1.0

OpenZinc contains several differences from Zinc
Application Frameworks 4.2 on which it is based.

Most of these are simply to
accommodate newer compilers and C++ standards. The only changes in the public interface of the
library are; 1)the change of UI_WINDOW_OBJECT member UI_REGION true to UI_WINDOW_OBJECT member UI_REGION trueRegion to avoid a conflict with bool true and 2) the change of DEVICE_IMAGE DM_POSITION to DEVICE_IMAGE DM_POSITION_IMAGE. Other than that code written for Zinc 4.x should compile fine under OpenZinc.

Additionally, *.cpp and *.hpp files generated by OpenZinc Designer are slightly different than those generated under Zinc Designer 4.x. Some newer compilers
require an explicit cast of an objects New() function, thus the line

```
{ ID_BUTTON, ZIL_VOIDF(UIW_BUTTON::New), UIW_BUTTON::_className, 0 },
```

generated by Zinc 4.2 Designer becomes

```
{ ID_BUTTON, (UI_WINDOW_OBJECT*)ZIL_VOIDF(UIW_BUTTON::New),
UIW_BUTTON::_className, 0 },
```

under OpenZinc designer.

To modify older versions of these generated files they simply need to be opened and saved with
the OpenZinc Designer. Alternatively, these changes can be made by hand. This will allow the generated code to be compiled under
newer compilers (as well as older ones).

Other important changes include provision of code within header files to allow compilation under linux and a number of additional makefiles for new compilers.

An additional directory \bat has been added under the OpenZinc main directory. this directory contains a number of batch files to set up the compiling environment
for individual compilers. For example, the file bc5.bat sets up the environment for compilation
under Borland C++ 5. The file ow18w32.bat sets up the environment for OpenWatcom 1.8 for
compiling Windows 32 libraries and programs. These bat files are meant only as a skeleton and
should be modified according to the location of your compiler and base OpenZinc directories.

A number of other under the hood changes have been made
because many newer compilers require explicit casts of void pointers, because scoping rules involving "for" loops have changed and so forth. In general, these changes are transparent to the user and allow the code to compile both on legacy

compilers and newer compilers. Most of these can be examined by greping for jdh.

Linux Notes

The linux environment was not defined in the origonal 4.2 release,
so the following sections were added to ui_env.hpp for Linux

```
// following define added by jdh to allow compilation under linux
#if defined(__linux__)
#define ZIL_LINUX
#endif

//-------------------------------------------------------------------------
//-------------Linux---------------------------------------------------------
//---section added by jdh-----------------------------------------------------

#if defined(ZIL_LINUX)
#     define ZIL_POSIX
#     undef ZIL_WORD_SIZE
#     define ZIL_WORD_SIZE 32
#     undef ZIL_SIGNED_CHAR
#     define ZIL_LOAD_MOTIF
#endif
```

in file m_dsp.cpp source uses xor as an identifier. gcc objects to this.
Therefore xorInt has been substituted as in the code below.

```
//statement below altered by jdh, compiler objects to use of xor
//as identifier. xorInt has been substituted
//void UI_XT_DISPLAY::Ellipse(ZIL_SCREENID screenID, int x, int y, int
startAngle,
//    int endAngle, int xRadius, int yRadius,const UI_PALETTE *palette, int
fill,
//    int xor, const UI_REGION *clipRegion)
void UI_XT_DISPLAY::Ellipse(ZIL_SCREENID screenID, int x, int y, int startAngle,
      int endAngle, int xRadius, int yRadius,const UI_PALETTE *palette, int
fill,
      int xorInt, const UI_REGION *clipRegion)
{
      VirtualGet(screenID, x, y, x + xRadius + xRadius, y + yRadius + yRadius);

      GC gc = xGC;

//statement below altered by jdh, compiler objects to use of xor
//as identifier. xorInt has been substituted
//    if (xor)
      if (xorInt)
```

Also in file m_dsp.cpp, gcc requires modern scoping rules in "for" statements
for example;

```
//line below added by jdh to conform to current "for" scoping
      int i;
      for (i = 0; i < numPoints; i++)
```

In original source both UIW_VT_LIST and UIW_HZ_LIST do not space properly

under Linux. The spacing is totally fubar and screws up the event handling of objects
containg a UIW_XX_LIST. I suspect this may be a difference between Motif 1.2 and Motif 2.x but I can not prove this.
In the original code these objects are formed with the following Motif objects;

a Frame containing a ScrolledWindow containing a RowColumn

Modified object is formed with

a Frame containing a Form containing a ScrolledWindow containing a RowColumn

This is accomplished by

1. adding the following include

```
//include added by jdh to allow a form under linux
#include <Xm/Form.h>
```

2.altering the S_REGISTER_OBJECT case of the Event() function something like;

```
      case S_REGISTER_OBJECT:
            clipList.Destroy();
            ccode = UI_WINDOW_OBJECT::Event(event);
            if (FlagSet(woFlags, WOF_BORDER))
                  true -= 2;
            clipList.Add(new UI_REGION_ELEMENT(screenID, 0, 0,
                  true.right - true.left, true.bottom - true.top));

      // line below added by jdh to allow region to be available throughout
      UI_REGION region;


#if defined(ZIL_LINUX)

//under linux UIW_HZ_LIST needs to be enclosed in a frame and a form to
//function properly spacially

            //create frame
            region = true;
            nargs = 0;
            true.bottom -= 2;
            RegisterObject(xmFrameWidgetClass,
ZIL_NULLF(ZIL_MOTIF_CONVENIENCE_FUNCTION),
                  ccode, TRUE);
            true.right -= true.left;
            true.bottom -= true.top;
            true.left = 0;
            true.top = 0;
            nargs = 0;
            XtSetArg(args[nargs], XmNresizePolicy, XmRESIZE_NONE); nargs++;
            RegisterObject(xmFormWidgetClass,
ZIL_NULLF(ZIL_MOTIF_CONVENIENCE_FUNCTION),
                  ccode, TRUE, TRUE, screenID);

#endif
```

```
            // Create the Window
            if (hScroll)            // scroll bar?
            {
                  nargs = 0;
                  XtSetArg(args[nargs], XmNscrollingPolicy, XmAUTOMATIC);
nargs++;
                  XtSetArg(args[nargs], XmNscrollBarDisplayPolicy, XmSTATIC);
nargs++;
                  RegisterObject(ZIL_NULLP(_WidgetClassRec),
XmCreateScrolledWindow,
                        ccode, TRUE, TRUE, screenID);
#                 if (ZIL_MOTIF > 1001)
                  XtAddCallback(screenID, XmNtraverseObscuredCallback,
                              ScrollObscuredCallback, (XtPointer)this);
#                 endif

                  Widget vertScroll = ZIL_NULLP(_WidgetRec),
                        horizScroll = ZIL_NULLP(_WidgetRec);
                  nargs = 0;
                  XtSetArg(args[nargs], XmNhorizontalScrollBar, &horizScroll);
                        nargs++;
                  XtSetArg(args[nargs], XmNverticalScrollBar, &vertScroll);
                        nargs++;
                  XtGetValues(screenID, args, nargs);
                  hScroll->screenID = horizScroll;
                  XtVaSetValues(horizScroll, XmNtraversalOn, FALSE, NULL);
                  true.bottom -= 31;
                  XtUnmanageChild(vertScroll);
            }



// code below becomes unnecessary because we have already provided a frame

/*          if (FlagSet(woFlags, WOF_BORDER) && !(hScroll || vScroll) &&
                  ccode == S_CREATE)
            {
                  nargs = 0;
                  XtSetArg(args[nargs], XmNshadowType, XmSHADOW_IN); nargs++;
                  XtSetArg(args[nargs], XmNshadowThickness,
                        FlagSet(woFlags, WOF_BORDER) ? 2 : 0); nargs++;
                  RegisterObject(xmFrameWidgetClass,
                        ZIL_NULLF(ZIL_MOTIF_CONVENIENCE_FUNCTION), ccode, TRUE,
TRUE);
            }

            */

            // create the RowColumn for the window
            {
                  nargs = 0;
                  XtSetArg(args[nargs], XmNpacking, XmPACK_TIGHT); nargs++;
                  XtSetArg(args[nargs], XmNorientation, XmVERTICAL); nargs++;
                  XtSetArg(args[nargs], XmNnumColumns, true.bottom - true.top);
nargs++;
                  XtSetArg(args[nargs], XmNspacing, 0); nargs++;
                  XtSetArg(args[nargs], XmNmarginHeight, 0); nargs++;
```

```
                  XtSetArg(args[nargs], XmNmarginWidth, 0); nargs++;
                  XtSetArg(args[nargs], XmNentryAlignment,
XmALIGNMENT_BEGINNING); nargs++;
                  XtSetArg(args[nargs], XmNresizeHeight, FALSE); nargs++;
                  XtSetArg(args[nargs], XmNresizeWidth, FALSE); nargs++;
                  XtSetArg(args[nargs], XmNadjustLast, FALSE); nargs++;
                  RegisterObject(xmRowColumnWidgetClass,
                        ZIL_NULLF(ZIL_MOTIF_CONVENIENCE_FUNCTION), ccode, TRUE,
FALSE, screenID ? screenID : 0);
            }

            // Compute the support object regions.
            for (object = (UI_WINDOW_OBJECT *)support.First(); object;
                  object = object->Next())
            {
                  object->Event(event);
                  if (FlagSet(object->woFlags, WOF_NON_FIELD_REGION))
                        clipList.Split(screenID, object->true, FALSE);
            }

            // Fall thru.
```

NOTE: Alterations are slightly different for UIW_HZ_LIST and UIW_VT_LIST because UIW_VT_LIST is used in UIW_COMBO_BOX. See code for definative differences

In m_button.cpp __linux__ is added to the following define

```
// Some Motif 1.1s can't draw their bitmaps off the left side of the button
// if the bitmap is added after the button is created.  PIXMAP_BUG fixes this,
// with the one problem of using the parent's background color instead of the
// button's.
#if defined(_SUNOS4) || defined(_IBM_RS6000) || defined(__DVX__) ||
defined(__sgi) || defined(__DECCXX) || defined(__linux__)
#     define PIXMAP_BUG
#endif
```

In m_intl.cpp some thought still needs to be given to i18n contains note;

```
#elif defined(__linux__)
      //need to look into internationalization in linux
      //NOTE TO SELF jdh
```

In files m_notebook.cpp, m_print.cpp, m_sbar.cpp and m_win2.cpp minor changes where made
to conform to modern scoping of "for" statements see source

****

Designer Notes

To compile designer for linux several changes need to be made to the source code.

Casting references to CLASS_NAME::New() to (UI_WINDOW_OBJECT*)CLASS_NAME::New()

Casting references to funtions in the _userTable[] from

ZIL_VOIDF(FunctionName) to (EVENT_TYPE*)ZIL_VOIDF(FilenameCallback)

In file help.cpp and help_help.cpp changing block

```
// Metrowerks bug requires taking the address of the member function.
#if defined(__MWERKS__) || defined(__DECCXX)
#     define ZIL_PROCESS_REFERENCE(x) &ZAF_HELP_EDITOR::x
#else
#     define ZIL_PROCESS_REFERENCE(x) x
#endif
```

to

```
// Metrowerks bug requires taking the address of the member function.
#if defined(__MWERKS__) || defined(__DECCXX) || defined(__linux__)
#     define ZIL_PROCESS_REFERENCE(x) &ZAF_HELP_EDITOR::x
#else
#     define ZIL_PROCESS_REFERENCE(x) x
#endif
```

NOTE: In the make process for motif make will copy base files and add a suffix and use the derived file to compile. Therefore, if the project is made clean and compiled again the derived file will not have any of the changes previously made. If desired changes are to be permanent change in both base and derived files.

Examples;

```
help.cpp -> help_help.cpp
z_bnum.cpp -> z_bnum_i18n.cpp
z_border.cpp -> z_border_des.cpp       ect.
```

These changes will allow a clean compile an generate an executable that works. In the file /design/window/object.cpp the code from Zinc for the Subobject window will generate a segmentation fault. Specific changes that need to be made are:

```
            //subWindow = _subWindow[offset];
            //dirList = _dirList[offset];
            //objList = _objList[offset];
            //subList = _subList[offset];

//code above causes segmentation fault when switching to subobject
//window under Motif. Code segment below by jdh avoids this

            subWindow = new UIW_WINDOW("OBJ_SUBOBJECT", storage, file,
objectTable, userTable);
            dirList = (UIW_VT_LIST *)subWindow->Get("LIST_DIRECTORIES");
            objList = (UIW_VT_LIST *)subWindow->Get("LIST_OBJECTS");
            subList = (UIW_COMBO_BOX *)subWindow->Get("FIELD_ADD_OBJECT");
```

Interestingly enough the class ZAF_MESSAGE_PREFERENCES is not used in the designer at least in motif, win32 and win16 whether this is intentional or an oversight by Zinc is unknown. Located in /design/message/prefer.cpp

The file /design/storage/storage.cpp has been altered so that the generated .cpp file will cast entries in the userTable and compareTable from { 0, ZIL_VOIDF(buttonPush), _buttonPush, 0 } to
{ 0, (EVENT_TYPE*)ZIL_VOIDF(buttonPush), _buttonPush, 0 }
and objectTable entries from { ID_BIGNUM, ZIL_VOIDF(UIW_BIGNUM::New), UIW_BIGNUM::_className, 0 } to
{ ID_BIGNUM, (UI_WINDOW_OBJECT*)ZIL_VOIDF(UIW_BIGNUM::New), UIW_BIGNUM::_className, 0 } as required by gcc compiler.

UIW_MESSAGE_WINDOW had spacing issues for items in the list, they overlap. Location /design/message/message1.cpp. This is true of all Zinc code using WOS_OWNERDRAW. The message window has been made functional by using the routine button's drawing in function UIW_MESSAGE_ITEM::SetButtonTitle(void) rather than making the UIW_MESSAGE_ITEM ownerDraw and using UIW_MESSAGE_ITEM::DrawItem() and simply manipulating the button label to present the information in three columns. This change is included in platform specific #if defined(ZIL_LINUX) ect.To line up the columns vertically two proportional fonts are added to the application resources in design/main.cpp

A bug exists in the designer help system. After calling File|Open the help index will not show a complete listing from the window editor menu until another tool help editor ect. has been called. Appears to be an issue with Zinc code.

****
Notes for Open Watcom
****

IMPORTANT !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

To ensure library is correct with open Watcom under win32 the Watcom section of ui_env.hpp must be modified!!! Initially this section reads

```
// ---------------------------------------------------------------------------
// ----- WATCOM --------------------------------------------------------------
// ---------------------------------------------------------------------------

#if defined(__WATCOMC__)
#       undef ZIL_VOIDP
#       define ZIL_VOIDP(pointer)       ((void *)(pointer))
#       undef ZIL_SIGNED_CHAR
#       if defined(ZIL_UNICODE)
#               include <stddef.h>
#               define ICHAR_T wchar_t
#       endif
#       if defined(__OS2__)
#               define ZIL_OS2          20
#               undef ZIL_WORD_SIZE
#               define ZIL_WORD_SIZE    32
#               define ZIL_FARDATA
#       elif defined(__WINDOWS__) ||  defined(__NT__)
#               define ZIL_MSWINDOWS    WINVER
#               undef ZIL_MSWINDOWS_CTL3D
#               if defined (__NT__) || defined(WIN32)
```

```
#                       define ZIL_WINNT        __NT__
#                       undef ZIL_WORD_SIZE
#                       define ZIL_WORD_SIZE    32
#                       define ZIL_FARDATA
#               else
#                       define ZIL_FARDATA              far
#               endif
#               ifdef __DLL__
#                       undef ZIL_EXPORT_CLASS
#                       if defined(ZIL_WINNT)
#                               define ZIL_EXPORT_CLASS __export
#                       else
#                               define ZIL_EXPORT_CLASS _export
#                       endif
#               endif
#       elif defined(__DVX__)
#               define ZIL_LOAD_MOTIF   // Use ZIL_MOTIF not ZIL_LOAD_MOTIF in
programs.
#               define MSDOS
#               define ZIL_FARDATA
#       elif defined(__QNX__)
#               define ZIL_POSIX
#               define ZIL_LOAD_MOTIF   // Use ZIL_MOTIF not ZIL_LOAD_MOTIF in
programs.
#               define ZIL_FARDATA
                struct _XDisplay;
                struct _XrmHashBucketRec;
                struct _XPrivate;
#       else
#               define ZIL_MSDOS        20
#               undef ZIL_WORD_SIZE
#               define ZIL_WORD_SIZE    32
#               define ZIL_FARDATA
#       endif
#endif

It was modified to

// ---------------------------------------------------------------------------
// ----- WATCOM --------------------------------------------------------------
// ---------------------------------------------------------------------------

#if defined(__WATCOMC__)
#       undef ZIL_VOIDP
#       define ZIL_VOIDP(pointer)       ((void *)(pointer))
#       undef ZIL_SIGNED_CHAR
#       if defined(ZIL_UNICODE)
#               include <stddef.h>
#               define ICHAR_T wchar_t
#       endif
#       if defined(__OS2__)
#               define ZIL_OS2          20
#               undef ZIL_WORD_SIZE
#               define ZIL_WORD_SIZE    32
#               define ZIL_FARDATA
#       elif defined(__WINDOWS__) ||  defined(__NT__) || defined(WIN32)  // last
or added by jdh
```

```
#               define ZIL_MSWINDOWS    WINVER
#               undef ZIL_MSWINDOWS_CTL3D
#               if defined (__NT__) || defined(WIN32)    // last or added by jdh
#                   if defined(__NT__)                   // added by jdh
#                       define ZIL_WINNT        __NT__
#                   endif                                // added by jdh
#                   if defined(WIN32)                    // added by jdh
#                       define ZIL_WINNT       WIN32     // added by jdh
#                   endif                                // added by jdh
#                       undef ZIL_WORD_SIZE
#                       define ZIL_WORD_SIZE    32
#                       define ZIL_FARDATA
#               else
#                       define ZIL_FARDATA              far
#               endif
#               ifdef __DLL__
#                       undef ZIL_EXPORT_CLASS
#                       if defined(ZIL_WINNT)
#                               define ZIL_EXPORT_CLASS __export
#                       else
#                               define ZIL_EXPORT_CLASS _export
#                       endif
#               endif
#       elif defined(__DVX__)
#               define ZIL_LOAD_MOTIF   // Use ZIL_MOTIF not ZIL_LOAD_MOTIF in
programs.
#               define MSDOS
#               define ZIL_FARDATA
#       elif defined(__QNX__)
#               define ZIL_POSIX
#               define ZIL_LOAD_MOTIF   // Use ZIL_MOTIF not ZIL_LOAD_MOTIF in
programs.
#               define ZIL_FARDATA
                struct _XDisplay;
                struct _XrmHashBucketRec;
                struct _XPrivate;
#       else
#               define ZIL_MSDOS        20
#               undef ZIL_WORD_SIZE
#               define ZIL_WORD_SIZE    32
#               define ZIL_FARDATA
#       endif
#endif
```

The need to define ZIL_WINNT as WIN32 was only detected when making 9design.exe. Without this define there were unresolved references when trying to link 9design.exe
(design.exe will compile and link fine with this change

In files 9_table2.cpp and 9_win1.cpp the following line exists

```
ScrollEvent(UI_EVENT(S_SCROLLRANGE));
```

OpenWatcom requires the explicit cast

```
ScrollEvent((UI_EVENT&)UI_EVENT(S_SCROLLRANGE));
```

In 9_win2.cpp line 252

```
static initializedTime = FALSE;
```

needs to be changed to

```
static int initializedTime = FALSE;
```

Similarly on lines 1240 and 1241

```
        static registeredClass = 0;
        static mdiRegisteredClass = 0;
```

needs to be changed to

```
        static int registeredClass = 0;
        static int mdiRegisteredClass = 0;
```

in file d_wccdsp.cpp in lines 601 and 625

```
 for (i = 0; i < numPoints * 2; i += 2)
```

needs to be changed to

```
  for (int i = 0; i < numPoints * 2; i += 2)
```

because of change in "for" scoping rules

in file o_dsp.cpp in lines 91, 272 and 726

```
 for (i = 0; i < numPoints * 2; i += 2)
```

needs to be changed to

```
  for (int i = 0; i < numPoints * 2; i += 2)
```

because of change in "for" scoping rules

in file o_hlist.cpp line 57

UI_WINDOW_OBJECT *object needs to be declared outside of for loop because of change in "for" scoping rules

in file o_print.cpp in line 134

```
 for (i = 0; i < numPoints * 2; i += 2)
```

needs to be changed to

```
  for (int i = 0; i < numPoints * 2; i += 2)
```

because of change in "for" scoping rules

in line 705

```
  DevEscape(hdc, DEVESC_STARTDOC, ::strlen(message), (PSZ)message, 0, 0);
```

needs to be changed to

```
DevEscape(hdc, DEVESC_STARTDOC, ::strlen(message), (unsigned char
*)(PSZ)message, 0, 0); //explicit cast needed
```

in line 865

ZIL_ICHAR *space needs to be declared outside of for loop because of change in "for" scoping rules

In 0_win2.cpp line 230

```
static initializedTime = FALSE;
```

needs to be changed to

```
static int initializedTime = FALSE;
```

FILE W_IMAGE.CPP DOESN'T COMPILE!!!11

In file w_table2.cpp line 112

```
ScrollEvent(UI_EVENT(S_SCROLLRANGE));
```

OpenWatcom requires the explicit cast

```
ScrollEvent((UI_EVENT&)UI_EVENT(S_SCROLLRANGE));
```

In file w_win1.cpp lines 1311 and 1513

```
ScrollEvent(UI_EVENT(S_SCROLLRANGE));
```

OpenWatcom requires the explicit cast

```
ScrollEvent((UI_EVENT&)UI_EVENT(S_SCROLLRANGE));
```

In w_win2.cpp line 341

```
static initializedTime = FALSE;
```

needs to be changed to

```
static int initializedTime = FALSE;
```

Similarly on lines 1314 and 1315

```
        static registeredClass = 0;
        static mdiRegisteredClass = 0;
```

needs to be changed to

```
        static int registeredClass = 0;
        static int mdiRegisteredClass = 0;
```

In files 3_table2.cpp and 9_win1.cpp the following line exists

```
ScrollEvent(UI_EVENT(S_SCROLLRANGE));

OpenWatcom requires the explicit cast

ScrollEvent((UI_EVENT&)UI_EVENT(S_SCROLLRANGE));

In 3_win2.cpp line 289

static initializedTime = FALSE;

needs to be changed to

static int initializedTime = FALSE;

Similarly on lines 1274 and 1275

        static registeredClass = 0;
        static mdiRegisteredClass = 0;

needs to be changed to

        static int registeredClass = 0;
        static int mdiRegisteredClass = 0;

in file w_image.cpp the following lines are added

#if defined(__WATCOMC__)
    #undef  GlobalFreePtr
    #define     GlobalFreePtr(lp) (GlobalUnlockPtr(lp),
(BOOL)GlobalFree(GlobalPtrHandle(lp)))
#endif
and the line

    GlobalFreePtr(dib);

is changed to

#if defined(__WATCOMC__)
        GlobalFreePtr((unsigned)dib);
#else
         GlobalFreePtr(dib);
#endif

in d_error1.cpp lines 17 & 18

        const ERROR_OK = 9900;
        const ERROR_CANCEL = 9901;

needs to be changed to

        const int ERROR_OK = 9900;
        const int ERROR_CANCEL = 9901;

in d_icon.cpp line 149

        static initializedTime = FALSE;
needs to be changed to
```

```
        static int initializedTime = FALSE;
```

in file gfx.h line 886

```
typedef struct _csc { int text, bkgnd, quick_key, xor;
```

must be changed to

```
typedef struct _csc { int text, bkgnd, quick_key, xorInt;
```

because watcom compiler will not allow xor as a specifier

in d_sys.cpp line 11

```
        static initializedTime = FALSE;
```

needs to be changed to

```
        static int initializedTime = FALSE;
```

in file d_tdsp.cpp

xor is used extensively as an identifier. Watcom compiler does not allow this so all occurences have been replaced with xorInt

****
Digital Mars Notes
****

Dmc is the more current name for the sc compiler.
The compiling environment is controlled by the sc.ini file much like the turboc.cfg and tlink.cfg files.
To use the X32 dos extender copy the x32 lib files to the DM\lib directory.the compiler switch

NOTE:The compiler switch -bx must be removed from makefiles(DOS & win16) or an obsolete compiler will be called

In file gfx.h xor is used as an identifier, replaced by xorInt

In file Z_DSP.CPP in constructor the lines

```
        void z_gen_dummy(void);
        z_gen_dummy();
        extern void z_dsp_dummy(void);
        z_dsp_dummy();
```

cause the linker to complain for winnt and windows so I simply did

```
#if !defined(__SC__)
        void z_gen_dummy(void);
        z_gen_dummy();
        extern void z_dsp_dummy(void);
        z_dsp_dummy();
#endif
```

and it works OK

In dm.mak for the library use rc only with the test.rc file for windows compilaton

For 32 bit dos compilation the x32.lib needs to be explicitly added to the

D32_LIBS=d32_zil d32_gfx dm_32gfx x32

line

I have added a section to the makefile to allow for making a win32 version of the
lib under win32. (Pretty basic need). To do so needed to:
1. create a win32 section in the dm.mak file
2. Add -DZIL_WIN32 to the W32_CPP_OPTS= parameter. There is no obvious easy way
   to use predefined macros to do this.
3. created a w32_zil.rsp file to make the library

Program nvlist.exe and 9vlist.exe in the tutor directory need work

------ DESIGNER --

There is a name conflict in making the designer under winnt and win32.
Digital Mars uses the symbol DECIMAL in headers for winnt and win32.
In file p_i18n.dat in both windows LOC_CURRENCY and LOC_NUMBER there is a field named DECIMAL. This causes a conflict when compiled. Therefore, the name of the field in each window has been changed to DECIMAL_SYMBOL. This in turn changes the name in the generated file p_i18n.hpp. Finally, the symbol DECIMAL must be changed to DECIMAL_SYMBOL in files Z_CURR.CPP and Z_NUM.CPP to allow linking.

****
Symantec C++ 7.x Notes
****

In files 3_win1.cpp and 9_win1.cpp
in function ScrollEvent() the following problem exists

```
//UI_REGION client = clipList.First() ? clipList.First()->region : trueRegion;
// sc7 complains about line above and quits with an internal error
// the code segment below should be equivalent
    UI_REGION client;
    if (clipList.First() != NULL)
        client = clipList.First()->region;
    else
        client = trueRegion;
```

****
Microsoft Visual C++ 2010 Notes
****

The Microsoft Visual C++ 2010 compiler objects too the identifier DM_POSITION so in all occurances it has been replaced by DM_POSITION_IMAGE.

libc.lib has been replace by a family of libraries in Visual C++ 2010 so the library clibmt.lib has been substituted.

The library ctl3d32.lib is no longer supported or needed for visual C++ 2010 so it has been eliminated

Also, in various files in the designer the section reading

```
// Metrowerks bug requires taking the address of the member function.
#if defined(__MWERKS__) || defined(__DECCXX) || defined(__linux__)
#     define ZIL_PROCESS_REFERENCE(x) &ZAF_HELP_EDITOR::x
#else
#     define ZIL_PROCESS_REFERENCE(x) x
#endif
```

has been changed to

```
// Metrowerks bug requires taking the address of the member function.
#if defined(__MWERKS__) || defined(__DECCXX) || defined(__linux__) || (_MSC_VER
> 1500)
#     define ZIL_PROCESS_REFERENCE(x) &ZAF_HELP_EDITOR::x
#else
#     define ZIL_PROCESS_REFERENCE(x) x
#endif
```

```
*****************************************************************************
***
Borland C++ 3.1 Notes
*****************************************************************************
***

bc31note.txt
Borlandc 3.1 apparently runs out of memory when compiling the entire OpenZinc
library
in one go (at least from the command line in Windows 7).
So the library has been split in two to allow compilation without complaint.
These libraries have been named dos_zil.lib and dos_zil2.lib in DOS
and win_zil.lib and win_zil2.lib under 16 bit windows.
Dos_zilo.lib is small enough it does not need to be split.
```