１５．インストーラ

１５－１．概要

本章では、景観シミュレーター一式をセットアップするインストーラの動作、及びセットアップ一式を構築するための手順について解説する。

セットアップには、VisualC++開発環境に付属している簡単な InstallShield の機能を使用している。セットアップに際して、レジストリの設定などの高度な処理を行っていないため、Ver.2.03 のセットアップに使用した Ver.2.00 を用いて、必要な部分を修正するのみで現在に至っている。従って、素朴な機能を用いているのみである。

１５－２．セットアップの構成と動作

セットアップには、CD-ROM に必要なファイルを置き、ここから setup.exe を起動してセットアップを行う方法と、関係するファイル一式を自己解凍形式のファイル sim209.exe に束ねて WEB 配信し、クライアント側で適当なディレクトリ上に解凍した上でセットアップ作業を行う方法を用意した。

まず、CD-ROM で配布する場合、ディスク上には、リスト１に示したファイルを提供している。

リスト１５－１：セットアップのファイル構成

（標準のファイル）	
Setup.exe	セットアップを行う実行形式
Inst32I.ex	セットアップの実行に使用されるバイナリ
Setup.dll	同上
Uninst.exe	アンインストーラ
_isdel.exe	アンインストール
（景観シミュレーション・システムに固有のファイル）	
Setup.bmp	セットアップに際して表示するロゴ（イメージ）
Setup.ins	セットアップ中に表示されるテキスト等を格納したファイル
Setup.ini	アプリケーション名や必要とするディスクスペース等の定義ファイル
Setup.iss	システムのバージョンやライセンスを記述
（バージョンに固有のファイル）	
Data.z	セットアップするファイルを、ディレクトリ付きで圧縮したファイル
Setup.pkg	セットアップするファイルの一覧
_setup.lib	セットアップの手順などを定義したファイル
DISK1.ID	ID

バージョン・アップやデバッグにより更新される主な部分は、data.z と、これに関連した Setup.pkg である。

Setup.exe を実行すると、標準的なインストーラの画面が表示され、セットアップ先のディレクトリ指定など、必要なパラメータや選択肢を確認しながらセットアップ作業を進める。

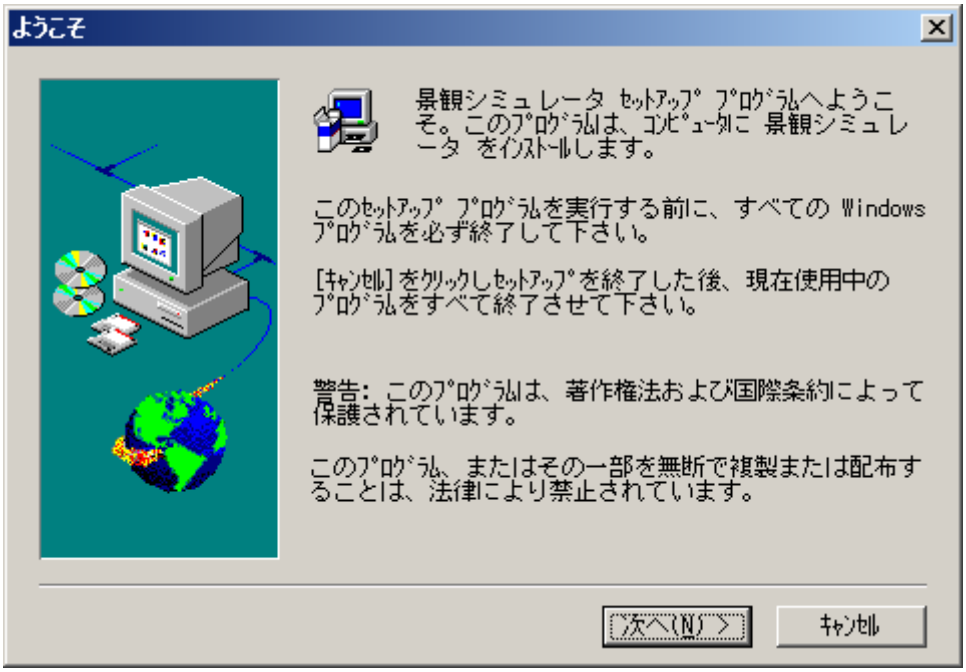

図１５－１：セットアップ画面１

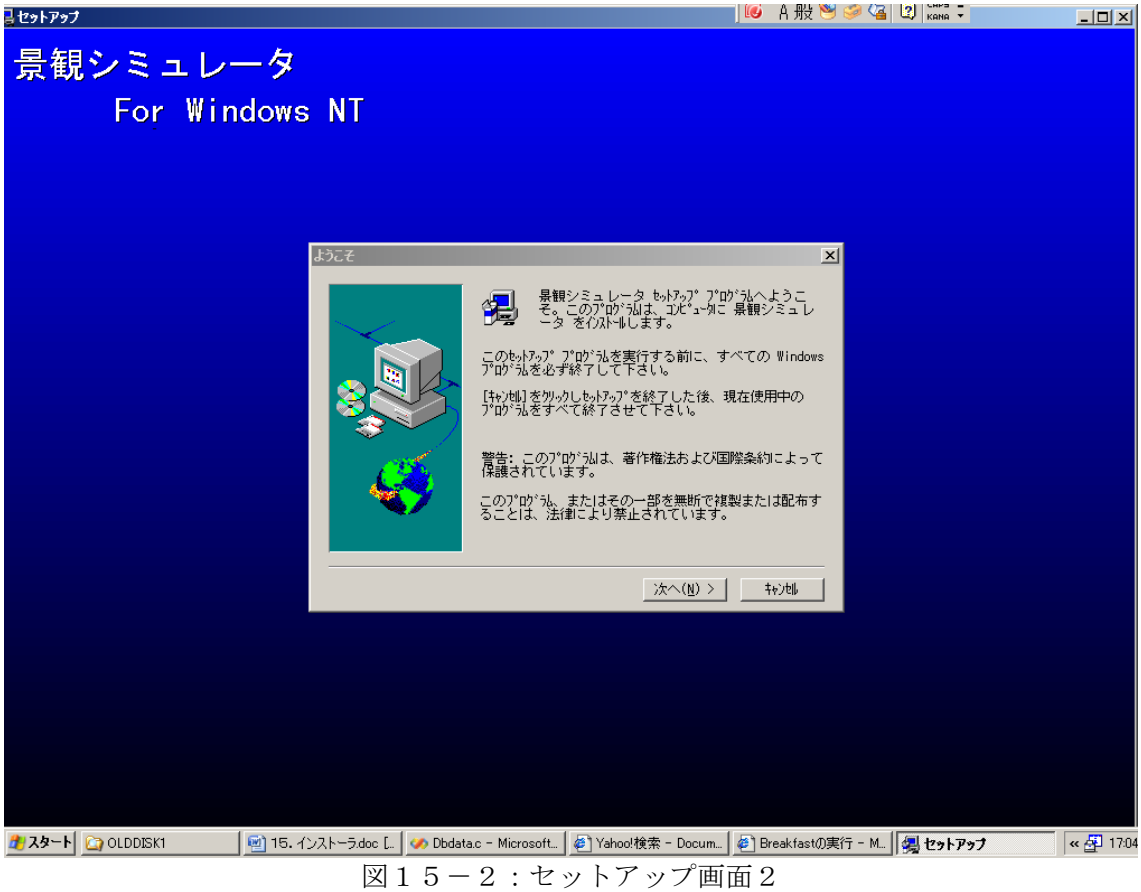

図１５－２：セットアップ画面２

セットアップがインストール先のシステムに適応して行う重要な処理は、環境変数 KSIM_ENV の設定(kdbms.set の所在をフルパスで指定する)と、kdbms.set の中の HOME_PATH エントリーを、ユーザーが指定したセットアップ先に書き換えることである。手動で必要なファイルをコピーした場合であっても、上記の２点の作業を行うことにより、インストーラを用いてセットアップを行った場合と同じ結果を得ることができる。

１５－３．セットアップの構築手順

新たなバージョンのセットアップを構築する場合、簡単には、ホームディレクトリ以下の各ファイルを用意した上で、以下の内容を有する BUILD.BAT を実行する。

リスト１５－２：セットアップ構築処理の全体を記述したバッチ・ファイル(build.bat)

```
rem @echo off
rem
rem ....Build components...
rem
setup¥compile setup¥setup.rul -g
copy setup¥setup.ins disk1
copy setup¥setup.ini disk1

if NOT "%1"=="" goto nopack

rem ====================================
rem  新規にアーカイブファイルを作成する
rem ====================================
del data.z

icomp data¥*.* data.z -i


:nopack

copy disk1.id disk1
cd setup
del setup.pkg
packlist setup.lst
copy setup.pkg ..¥disk1
cd ..
copy setup¥_setup.lib disk1

rem ================================================
rem  セットアップ起動時に表示されるビットマップをコピー
rem ================================================
copy setup¥setup.bmp                    disk1

rem ================================================
rem  タイトルとビルボードのビットマップを圧縮コピーする
rem ================================================
icomp setup¥title.bmp                                   disk1¥_setup.lib
icomp setup¥bbr*.bmp                                            disk1¥_setup.lib


move data.Z disk1
```

このバッチ・ファイルが実行している主な処理は、

①セットアップの中で行う処理を定義した setup.rul をコンパイルし、setup.exe の動作を定義する setup.ins 等のファイルを作成すること。

②セットアップするファイル（各種実行形式、データファイルを含むディレクトリ構成）

を圧縮し data.z を作成すること。

③表示する画像ファイル(title.bmp, bbr.bmp)を圧縮し、_setup.lib とすること

および、結果のファイルのコピー、移動等の補助的な処理である。

図１５－３：title.bmp

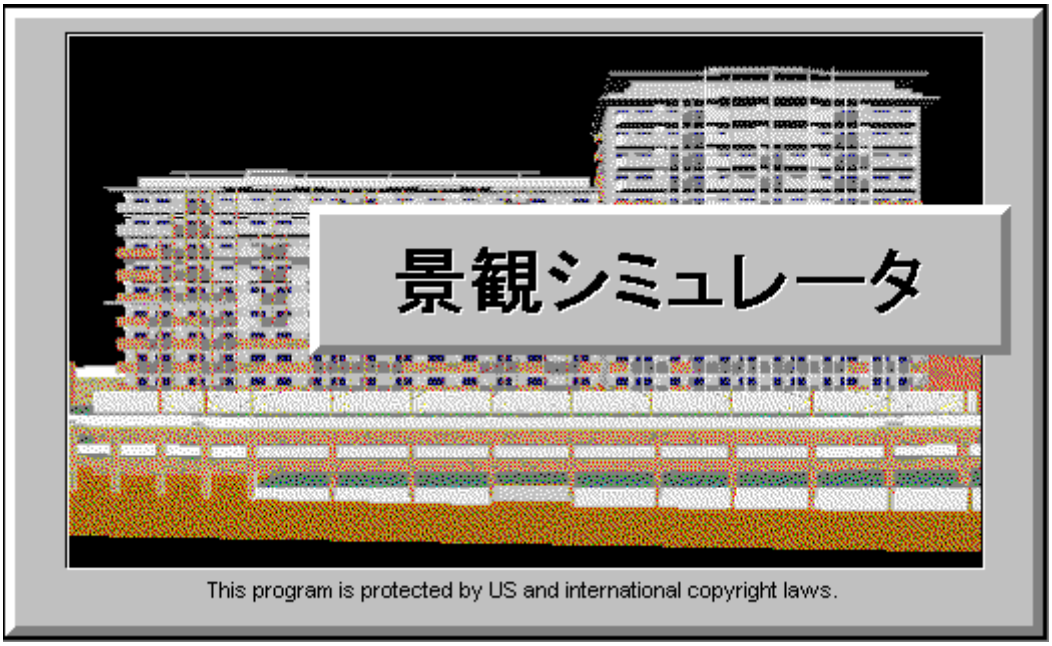

図１５－４：bbr.bmp, setup.bmp

セットアップ中に行われる処理を定義した、setup.rul は以下の内容を含んでいる。

リスト１５－３：setup.rul

```
/*---------------------------------------------------------------------*¥
 *
 * IIIIIII SSSSSS
 *    II SS          InstallShield (R)
 *    II SSSSSS      (c) 1990-1996, Stirling Technologies, Inc.
 *    II    SS           All Rights Reserved.
 * IIIIIII SSSSSS
 *
 *
 * This code is intended as a supplement to InstallShield documentation
 * and is provided AS IS.
 *
 *
 *    File Name:   SETUP.RUL
 *
 * Description:   InstallShield
 *         32-bit Template Two script.
 *
 *    Author:   InstallShield Corporation      Date:   1-10-96
 *
 *      Comments:   This template script performs a basic installation to a
```

```
  *         Windows 95 or Windows NT platform.    The installation
  *         includes components: Application Program Files, Sample and
  *         Template Files, Online Help Files, and Multimedia Tutorial
  *         Files.    With minor modifications, this template can be
  *         adapted to create new, customized installations.
  *
¥*---------------------------------------------------------------------------*/

// Constant declarations.
#define APP_NAME     "景観 シミュレータ"
#define PROGRAM_FOLDER_NAME   "景観 シミュレータ"
//#define APPBASE_PATH     "¥¥Program Files¥¥keikan¥¥"
#define APPBASE_PATH     "¥¥@keikan"
#define EXTRA_BLOCKSPACE    200000

#define COMPANY_NAME     "NILIM"
#define PRODUCT_NAME     "景観シミュレータ"
#define PRODUCT_VERSION     "2.09"
#define PRODUCT_KEY           "SIM.exe;貿易.exe;都市開発.exe"
#define DEINSTALL_KEY     "景観シミュレータ"
#define UNINSTALL_NAME     "景観シミュレータ"

// File component constant declarations.
#define ITEM_PROGRAMFILES   "プログラム　ファイル"
#define ITEM_ENVFILES     "環境設定　ファイル"
#define ITEM_DLLFILES     "ＤＬＬ　ファイル"

#define STR_DEFTAB       "                    "

declare

#include "sddialog.h" // Include script dialog definitions.

   // Global variable declarations.
   STRING svFolder, szMsg, szFileSet, szTitle;
   STRING svInstallDir;
   STRING svTarget, szProgram, szTemp, svName, svCompany, szComponentList;
   STRING svUninstLogFile, szAppPath, szRegKey, szAppSharedDir;
   BOOL bSpaceOk, bWinNT, bWin32S, bIsShellExplorer;
   NUMBER nResult, nType;
   NUMBER nRet;
   LIST listInfo;
   NUMBER nReturn;
   NUMBER nvLineNumber;
   STRING svLine;
   LIST listTopics; //20000406 DR.H.K.
   LIST listDetails; //20000406 DR.H.K.
   // Function declarations.
   prototype CheckRequirements();
   prototype ConstructInfoList( LIST );
   prototype CreateRegDBEntries();
   prototype DetermineComponentInfo( STRING );
   prototype EnoughSpace( STRING, STRING, NUMBER );
   prototype PerformFileTransfer( STRING );
   prototype SetupFileTransfer( STRING, STRING );
   prototype SetupFinish();
   prototype SetupScreen();

program

StartHere:
   //MessageBox("インストーラのデバッグ中",INFORMATION); //20000406 DR.H.K.
   Disable( BACKGROUND );

   // Set installation info., which is required for registry entries.
   InstallationInfo( COMPANY_NAME, PRODUCT_NAME, PRODUCT_VERSION, PRODUCT_KEY );
```

```
    // Set up the installation screen.
    SetupScreen();
    Enable( DIALOGCACHE );


     // Create a Welcome dialog.
     WelcomeDlg:
    Disable( BACKBUTTON );
    SdProductName( PRODUCT_NAME );
//    SdWelcome( //20000407 DR.H.K. 交換
    listTopics = ListCreate( STRINGLIST );
    listDetails = ListCreate( STRINGLIST );
    ListAddString( listTopics,
    "このプログラムは、" , AFTER );
    ListAddString( listDetails,
    "コンピュータに標記システムをインストールします。このセットアッププログラムを実行する前に、全ての
Windows プログラムを必ず終了して下さい。"
    , AFTER );
//    ListAddString( listTopics,
//    "[キャンセル]で抜けて", AFTER );
//    ListAddString( listDetails,
//    "現在使用中のプログラムをすべて終了させて下さい。", AFTER );
    ListAddString( listTopics,
    "            ◎ 一度[キャンセル]で抜けて準備して下さい。", AFTER );
    ListAddString( listDetails,
     "警告：このプログラムは、フリーウェアですが、"+
       "国土交通省が著作権を保持しており、"+
       "著作権法および国際条約によって保護されている公共財です。", AFTER);
    ListAddString( listTopics,
       "このプログラム、またはその一部を", AFTER );
    ListAddString( listDetails,
       "無断で翻訳、引用、複製または配布しても構いませんが、"+
       "何らの価値も付加しないデッドコピーを"+
       "メディア代以上の価格で販売することは制限しています。", AFTER );
    SdDisplayTopics("ようこそ",
    "国土交通省版・ネットワーク景観シミュレータ 2.05 のセットアップ"+
    "プログラムへようこそ。",
    listTopics,   listDetails, 0 );
    Enable( BACKBUTTON );

//インストール先のディレクトリを選択する


SelectInstallFolder:
//    svFolder = "C:" ^ APPBASE_PATH;
    svFolder = "C:" ^ APPBASE_PATH;
    nReturn = SdAskDestPath(PRODUCT_NAME + "のインストール先の選択",
              "セットアップは次のディレクトリに景観シミュレータを" +
              "インストールします。¥n¥n" +
              "このディレクトリへのインストールは［次へ］をクリック。¥n¥n" +
              "他のディレクトリへのインストールは［参照］をクリックしディレクトリを選択。¥n¥n" +
              "景観シミュレータをインストールしない場合は［キャンセル］をクリックして終了。",
              svFolder,
              STYLE_NORMAL);
    if( nReturn = NEXT ) then
      if( ExistsDir( svFolder ) = EXISTS ) then
//        SprintfBox( INFORMATION,
//              "すでに同名のディレクトリが存在します",
//              "ディレクトリ名 '%s' はすでに存在します¥n" +
//              "このディレクトリでよろしければＯＫを押してください¥n",
//              svFolder);
      else
        nReturn = SdConfirmNewDir( "ディレクトリを作成します(デバッグ中)",
                  svFolder,
                  STYLE_NORMAL);
        if( nReturn < 0 ) then
```

```
        SprintfBox( SEVERE, "エラー",
            "ディレクトリ '%s' の作成ができませんでした¥n" +
            "インストールは未完了です", svFolder);
        exit;
       elseif( nReturn = NO ) then
        MessageBox( "ディレクトリは作成しませんでした¥n" +
            "インストールは未完了です", SEVERE);
        exit;
       endif;
      endif;
      svInstallDir = svFolder;
    else
      goto WelcomeDlg;
    endif;


     RegisterUserName:
#if 0
    nResult = SdRegisterUser( "", "", svName, svCompany );
    if ( nResult = BACK ) then goto WelcomeDlg; endif;
#endif
    // Test target system for proper configuration.
    CheckRequirements();
    // win32s では動作不可なのでインストールしない
    if ( bWin32S ) then
        MessageBox( "Windows NT 以外のＯＳは現在サポートしていません", SEVERE );
        exit;
        //svTarget = TARGETDISK ^ APPBASE_PATH_WIN32S;
    else
        svTarget = svFolder;
    endif;
    szAppSharedDir = svTarget ^ "System";
    szComponentList = "FileComponents";

    // Get component information.
    DetermineComponentInfo( szComponentList );


     GetTargetLocAndSelection:
#if 0
    // Get type of installation (Typical, Compact, or Custom)
    // インストールのスタイルを選択する（標準、コンパクト、カスタム）
    nType = SdSetupType( "", "",
            svTarget,
            STYLE_NORMAL );

    switch (nType)
    case TYPICAL:
      ComponentSelectItem( szComponentList, ITEM_PROGRAMFILES, TRUE );
      ComponentSelectItem( szComponentList, ITEM_ENVFILES, TRUE );
      ComponentSelectItem( szComponentList, ITEM_DLLFILES, TRUE );
    case COMPACT:
      ComponentSelectItem( szComponentList, ITEM_PROGRAMFILES, TRUE );
      ComponentSelectItem( szComponentList, ITEM_ENVFILES, FALSE );
      ComponentSelectItem( szComponentList, ITEM_DLLFILES, FALSE );
    case CUSTOM:
      nResult = SdComponentDialog( "", "",
                svTarget,
                szComponentList );
      if ( nResult = BACK ) then
      goto GetTargetLocAndSelection;
      endif;
    case BACK:
      goto RegisterUserName;
    default:
      MessageBox( "SetupType - 不可能な選択肢です。", SEVERE );
      exit;
```

```
   endswitch;
#else
     ComponentSelectItem( szComponentList, ITEM_PROGRAMFILES, TRUE );
     ComponentSelectItem( szComponentList, ITEM_ENVFILES, TRUE );
     ComponentSelectItem( szComponentList, ITEM_DLLFILES, TRUE );
#endif

   szAppSharedDir = svTarget ^ "System";

   // Perform space check of target drive.
   if (EnoughSpace( svTarget,
          szComponentList,
          EXTRA_BLOCKSPACE ) = FALSE) goto GetTargetLocAndSelection;


  GetProgramFolderInfo:
   svFolder = PROGRAM_FOLDER_NAME;
   nResult = SdSelectFolder( "", "", svFolder );
   if ( nResult = BACK ) then
#if 0
     goto GetTargetLocAndSelection;
#else
     goto SelectInstallFolder;
#endif
   endif;


  ConfirmCopy:
   // Show SdStartCopy dialog to confirm file transfer operation.
   listInfo = ListCreate( STRINGLIST );
   ConstructInfoList( listInfo );
   szMsg = "セットアップには" + PRODUCT_NAME + "・ファイルのコピーを開始するに十分な情報があります。" +
      "設定を確認または変更する場合「戻る」をクリックします。現在の設定" +
      "のまま実行する場合は「次へ」をクリックしてファイルのコピーを開始します。";
   if ( SdStartCopy( "", szMsg, listInfo ) = BACK ) then
       ListDestroy( listInfo );
       goto GetProgramFolderInfo;
   endif;
   ListDestroy( listInfo );


    SetupRegAndUninstall:
   // Prepare InstallShield to record deinstallation information.
   DeinstallStart( svTarget, svUninstLogFile, DEINSTALL_KEY, 0 );
   RegDBSetItem( REGDB_UNINSTALL_NAME, UNINSTALL_NAME );

   // Set the App Paths key for the main program.
   szAppPath = svTarget ^ "ksim¥¥bin" + ";" + szAppSharedDir;
   RegDBSetItem( REGDB_APPPATH, szAppPath );
   szProgram = svTarget ^ "ksim¥¥bin¥¥sim.exe";
   RegDBSetItem( REGDB_APPPATH_DEFAULT, szProgram );


    SetupAndDecompFiles:
   szFileSet = "General";
   SetupFileTransfer( szComponentList, szFileSet );
   //MessageBox( "SetupFileTransfer : complete" , INFORMATION );
   // Set up progress indicator and information gauge.
   Enable( STATUS );
   Disable( DIALOGCACHE );
   //MessageBox( "Enable Progress Indicator : complete" , INFORMATION );
   // Transfer files to the target system.
   PerformFileTransfer( szFileSet );
   //MessageBox( "PerformFileTransfer : complete" , INFORMATION );


    SetRegistryEntries:
```

```
        SetStatusWindow( 92, "レジストリの設定中...." );
  CreateRegDBEntries( );

  // Set up progress indicator and information gauge.
  //Delay( 2 );
  Delay( 1 );
  //MessageBox( "ゲージが９０％で止っていても、気にする必要なし",INFORMATION); //20000406 DR.H.K.
  //Disable( STATUS );


   // Create program folder and icons.
   CreateProgramIcons:
  SetStatusWindow( 95, "グループとアイコンを作成中...." );

  // NT でシェルがエクスプローラでないなら
  if ( (bWinNT || bWin32S) && !bIsShellExplorer ) then
   //MessageBox( "NT でシェルがエクスプローラでない", INFORMATION ); //20000406 DR.H.K.
      //プログラムマネージャに新規グループを作成する
      AppCommand( PROGMAN, CMD_RESTORE );
      CreateProgramFolder( svFolder );
      ShowProgramFolder( svFolder, 0 );
      LongPathToShortPath( svTarget );
      Delay(1);
  else
   //MessageBox( "NT ではないか、シェルがエクスプローラである場合", INFORMATION ); //20000406 DR.H.K.
      //プログラムマネージャに新規グループを作成する
      AppCommand( PROGMAN, CMD_RESTORE );
      CreateProgramFolder( svFolder );
      ShowProgramFolder( svFolder, 0 );
      LongPathToShortPath( svTarget );
      Delay(1);
  endif;

  TARGETDIR = svTarget;

  if (ComponentIsItemSelected( szComponentList, ITEM_PROGRAMFILES )) then

      szProgram = TARGETDIR ^ "ksim¥¥bin¥¥sim.exe";
      // NT でシェルがエクスプローラでないなら
      if ((bWinNT || bWin32S) && !bIsShellExplorer ) then
     //アイコンを登録する
     //MessageBox("NT でシェルがエクスプローラでない", INFORMATION ); //20000406 DR.H.K.
     //MessageBox( APP_NAME ^ "アイコンを登録する", INFORMATION ); //20000406 DR.H.K.
     AddFolderIcon( svFolder, APP_NAME,
          szProgram,
          TARGETDIR ^ "ksim¥¥bin",
          "", 0, "", REPLACE );
//      Delay( 1 );
//      AddFolderIcon( svFolder, "メインメニュー",
//           "keikan.exe",
//           TARGETDIR ^ "ksim¥¥bin",
//           "", 0, "", REPLACE );
//      Delay( 1 );
//      AddFolderIcon( svFolder, "優良 景観 事例 検索",
//           "yuu.exe",
//           TARGETDIR ^ "ksim¥¥bin",
//           "", 0, "", REPLACE );
//      Delay( 1 );
//      AddFolderIcon( svFolder, "景観 材料 検索",
//           "zai.exe",
//           TARGETDIR ^ "ksim¥¥bin",
//           "", 0, "", REPLACE );
//      Delay( 1 );
//      AddFolderIcon( svFolder, "景観 構成 要素 検索",
//           "kou.exe",
//           TARGETDIR ^ "ksim¥¥bin",
//           "", 0, "", REPLACE );
```

```
//      Delay( 1 );
//      AddFolderIcon( svFolder, "市街地生成",
//            "都市開発.EXE",
//            TARGETDIR ^ "都市開発",
//            "", 0, "", REPLACE );
//      Delay( 1 );
//      AddFolderIcon( svFolder, "貿易",
//            "貿易.EXE",
//            TARGETDIR ^ "貿易 95",
//            "", 0, "", REPLACE );
      Delay( 1 );
      AddFolderIcon( svFolder, "お読み下さい",
            "NOTEPAD.EXE " + TARGETDIR ^ "README.TXT",
            TARGETDIR,
            "", 0, "", REPLACE );
      Delay( 1 );
       else
          //MessageBox("「NT でシェルがエクスプローラでない」ことはない", INFORMATION ); //20000406 DR.H.K.
          //MessageBox( szProgram, INFORMATION ); //20000406 DR.H.K.
          //MessageBox( TARGETDIR, INFORMATION );
          //LongPathToQuote( szProgram, TRUE ); //"" で囲む
       Delay( 1 );
       AddFolderIcon( svFolder, APP_NAME,
          szProgram,
          TARGETDIR ^ "ksim¥¥bin",
          "", 0, "", REPLACE );
//       Delay( 1 );
//       AddFolderIcon( svFolder, "優良景観事例検索",
//          TARGETDIR ^ "ksim¥¥bin¥¥yuu.exe",
//          TARGETDIR ^ "ksim¥¥bin",
//          "", 0, "", REPLACE );
//       Delay( 1 );
//       AddFolderIcon( svFolder, "景観構成要素検索",
//          TARGETDIR ^ "ksim¥¥bin¥¥kou.exe",
//          TARGETDIR ^ "ksim¥¥bin",
//          "", 0, "", REPLACE );
//       Delay( 1 );
//       AddFolderIcon( svFolder, "景観材料検索",
//          TARGETDIR ^ "ksim¥¥bin¥¥zai.exe",
//          TARGETDIR ^ "ksim¥¥bin",
//          "", 0, "", REPLACE );
//       Delay( 1 );
//       AddFolderIcon( svFolder, "景観データベース入力",
//          TARGETDIR ^ "ksim¥¥bin¥¥editor.exe",
//          TARGETDIR ^ "ksim¥¥bin",
//          "", 0, "", REPLACE );
      endif;
      Delay( 1 );
   endif;

   if ((bWinNT || bWin32S) && !bIsShellExplorer ) then

      // Global variable UNINST stores the name and location of the
      // uninstaller file.
      //MessageBox( "Global variable UNINST stores the name and location of the uninstaller file.",
INFORMATION ); //20000406 DR.H.K.
      szProgram = UNINST;
      LongPathToShortPath( szProgram );
      LongPathToShortPath( svUninstLogFile );
      szProgram = szProgram + " -f" + svUninstLogFile;
      AddFolderIcon( svFolder, "アンインストール", szProgram,
         WINDIR, "", 0, "", REPLACE );
      Delay( 1 );
   endif;


    // Announce setup complete and offer to read README file.
```

```
    CloseOfInstall:
      SetStatusWindow( 98 , "環境設定ファイルの調整中...." );
      //MessageBox("ディレクトリの作成",INFORMATION); //20000406 DR.H.K.
      CreateDir( svTarget ^ "¥¥ksim¥¥temp" );
      //MessageBox("環境ファイルの HOMEPATH の設定",INFORMATION); //20000406 DR.H.K.
      VarSave(SRCTARGETDIR);
      SRCDIR = svTarget ^ "ksim¥¥bin";
//      FileGrep("kdbms.set", "KEIKAN_HOMEPATH", svLine, nvLineNumber, RESTART); 970407 DR.H.K.
      FileGrep("kdbms.set", "HOME_PATH", svLine, nvLineNumber, RESTART);
      FileDeleteLine("kdbms.set", nvLineNumber, nvLineNumber);
      FileInsertLine("kdbms.set", "HOME_PATH = " + svTarget + ";", nvLineNumber, BEFORE);

   SetStatusWindow( 100, "インストールの完了。" );

   SetupFinish();

   szMsg = "セットアッププログラムにより環境設定を変更しました" +
     "変更を有効にするには再起動してください";
   if( RebootDialog("windows 再起動", szMsg, SYS_RESTART) = 0) then
    CommitSharedFiles(0);
   endif;


   exit;


/*---------------------------------------------------------------------------*¥
 *
 * Function:   SetupScreen
 *
 * Purpose:   This function will set up the screen look.   This includes
 *        colors, fonts, text to be displayed, etc.
 *
 *
 *     Input:
 *
 * Returns:
 *
 * Comments:
¥*---------------------------------------------------------------------------*/

function SetupScreen()
   string szBitmap;
   number nvDx, nvDy, nDxBillboard, nDyBillboard;
begin
   GetExtents( nvDx, nvDy );

   Enable( FULLWINDOWMODE );
   Enable( INDVFILESTATUS );
   Enable( BITMAP256COLORS );   //256 色のビットマップを使用するために必ず必要

   SetColor( BACKGROUND, BK_BLUE ); // Gradient blue background.
   SetTitle( "セットアップ", 0, BACKGROUNDCAPTION ); // Caption bar text.

   Enable( BACKGROUND );

   // Show the bitmap.
   Enable( BITMAPFADE );
   szBitmap = SUPPORTDIR ^ "TITLE.BMP"; //ビットマップファイル名の作成
   //ビットマップファイルの位置指定と表示
   PlaceBitmap(szBitmap , 1, 10, 10, UPPER_LEFT | BITMAPICON );
   Disable( BITMAPFADE );

end;


/*---------------------------------------------------------------------------*¥
```

```
  *
  * Function:   CheckRequirements
  *
  * Purpose:   This function will check all minimum requirements for the
  *         application being installed. If any fail, then the user
  *         is informed and the installation is terminated.
  *
  *
  *     Input:
  *
  * Returns:
  *
  * Comments:
¥*------------------------------------------------------------------------*/
function CheckRequirements()
number nvDx, nvDy;
number nvResult;
string svResult;
begin

   // Determine target system's operating system.
   GetSystemInfo( OS, nvResult, svResult );
   bWinNT = FALSE;
   bWin32S = FALSE;
   if (nvResult = IS_WINDOWSNT) then
       bWinNT = TRUE;

       // Check to see if NT is using EXPLORER Shell
       if( QueryShellMgr( svResult ) = 0 ) then
         if( StrCompare( svResult, "EXPLORER.EXE" ) = 0 ) then
           bIsShellExplorer = TRUE;
         endif;
       endif;

   elseif (nvResult = IS_WIN32S) then
       bWin32S = TRUE;
   endif;

   // Check screen resolution.
   GetExtents( nvDx, nvDy );
   if (nvDy < 480) then
       MessageBox( "このプログラムを実行するには VGA 以上の解像度が必要です。", WARNING );
       exit;
   endif;

end;

/*------------------------------------------------------------------------*¥
  *
  * Function:   SetupFileTransfer
  *
  * Purpose:   This function defines the file set based on the user's choices
  *         of components, then it performs the file set.
  *
  *
  *     Input:
  *
  * Returns:
  *
  * Comments:
¥*------------------------------------------------------------------------*/

function SetupFileTransfer( szComponentList, szFileSet )
begin

   // Define the file set.
   TARGETDIR = svTarget;
```

```
   FileSetBeginDefine( szFileSet );

      // Always include README files, located at the root of the
      // DATA.Z library file.
      SetStatusWindow( -1, "基本情報 ファイルをコピー中..." );
      TARGETDIR = svTarget;
      nRet = CompressGet( "data.z", "*.*", COMP_NORMAL );
     if( nRet < 0 ) then
     MessageBox("CompressGet failed(COMP_NORMAL) ", WARNING);
    endif; //20000523 DR.H.K.
      // If program files are selected, then add them to file set.
      if (ComponentIsItemSelected( szComponentList, ITEM_PROGRAMFILES )) then
      SetStatusWindow( -1, "ファイルをコピー中..." );
      TARGETDIR = svTarget;
      nRet = CompressGet( "data.z", "*.*", INCLUDE_SUBDIR );
     if( nRet < 0 ) then
      MessageBox("CompressGet failed(INCLUDE_SUBDIR)", WARNING);
          endif;
      endif;
   FileSetEndDefine( szFileSet );

end;


/*---------------------------------------------------------------------------*¥
 *
 * Function:   PerformFileTransfer
 *
 * Purpose:   This function will perform the file transfer and handle
 *       any error that may occur during the file transfer.
 *
 *
 *    Input:
 *
 * Returns:
 *
 * Comments:
¥*---------------------------------------------------------------------------*/

function PerformFileTransfer( szFileSet )

begin

   StatusUpdate( ON, 90 );

   // Perform the file set.
   nResult = FileSetPerformEz( szFileSet, 0 );

   switch (nResult)
   case FS_DONE: // Successful completion.

   case FS_CREATEDIR: // Create directory error.
      MessageBox( TARGETDIR + "の下にディレクトリを作成することができません。" + "."+
          "ディレクトリの書き込み属性を確認して下さい。", SEVERE );
      exit;
   // 20000405 DR.H.K. エラー発生のため、メッセージを追加。
   case 0:   MessageBox( "転送成功", SEVERE );
   case FS_TONEXTDISK: MessageBox( "次のディスク", SEVERE);
    exit;
   case FS_FILENOTINLIB:
    MessageBox( "FILENOTINLIB", SEVERE);
    exit;
   case FS_GENERROR:
    MessageBox( "GENERROR", SEVERE);
    exit;
   case FS_INCORRECTDISK:
    MessageBox( "INCORRECTDISK", SEVERE);
```

```
    exit;
  case FS_LAUNCHPROCESS:
    MessageBox( "LAUNCHPROCESS", SEVERE);
    exit;
  case FS_OPERROR: //圧縮前のソースのルートに DeIsL1.ins 等があると、このエラーが生じる場合がある。
    MessageBox( "OPERROR", SEVERE);
    exit;
  case FS_PACKAGING:
    MessageBox( "PACKAGING", SEVERE);
    exit;
  case FS_RESETREQUIRED:
    MessageBox( "RESETREQUIRED", SEVERE);
    exit;
  default: // Group all other errors under default label.
     NumToStr( szTemp, nResult );
     MessageBox( "一般的な転送エラー。"+
          "インストール先を確認して再度実行して下さい。"+
          "¥n¥n エラー番号:"+szTemp, SEVERE );

     exit;
  endswitch;
end;



/*---------------------------------------------------------------------------*¥
 *
 * Function:   DetermineComponentInfo
 *
 * Purpose:   This function will place all the separatly identifable
 *       file components into a component list. It will also
 *       set the size the files represent and will by default
 *       enable each to be selected.
 *
 *
 *    Input:
 *
 * Returns:
 *
 * Comments:
¥*---------------------------------------------------------------------------*/

function DetermineComponentInfo( szFileList )
number nvSize;
string svSize;
begin

  CompressInfo( "data.1", "ksim¥¥bin¥¥*.exe",
       COMP_INFO_ORIGSIZE | INCLUDE_SUBDIR,
       nvSize, svSize );
  ComponentAddItem( szFileList, ITEM_PROGRAMFILES, nvSize, TRUE );
  CompressInfo( "data.1", "kdb¥¥*.*",
       COMP_INFO_ORIGSIZE | INCLUDE_SUBDIR,
       nvSize, svSize );
  ComponentAddItem( szFileList, ITEM_PROGRAMFILES, nvSize, TRUE );


  CompressInfo( "data.1", "ksim¥¥bin¥¥*.set",
       COMP_INFO_ORIGSIZE | INCLUDE_SUBDIR,
       nvSize, svSize );
  ComponentAddItem( szFileList, ITEM_ENVFILES, nvSize, TRUE );

  CompressInfo( "data.1", "ksim¥¥bin¥¥*.DLL",
       COMP_INFO_ORIGSIZE | INCLUDE_SUBDIR,
       nvSize, svSize );
  ComponentAddItem( szFileList, ITEM_DLLFILES, nvSize, TRUE );
```

```
end;

/*---------------------------------------------------------------------------*¥
 *
 * Function:   EnoughSpace
 *
 * Purpose:   This function will determine if enough space exists on
 *        the target drive based on the selection in thge component
 *        list.
 *
 *
 *     Input:
 *
 * Returns:
 *
 * Comments:
¥*---------------------------------------------------------------------------*/
function   EnoughSpace( svTarget, szComponentList, nExtraSpace )
LIST list;
number nTotal, nvSize, nFreeSpace, nResult;
string svComponent;
begin

   // Get total size of all selected components.
   list     = ListCreate( STRINGLIST );
   nTotal = nExtraSpace;
   ComponentListItems( szComponentList, list );

   nResult = ListGetFirstString( list, svComponent );

   while (nResult = 0)
       ComponentGetItemSize( szComponentList, svComponent, nvSize );
       if (ComponentIsItemSelected( szComponentList, svComponent )) then
      nTotal = nTotal + nvSize;
       endif;
       nResult = ListGetNextString( list, svComponent );

   endwhile;

   ListDestroy( list );

   // Determine if target disk has enough space.
   nFreeSpace = GetDiskSpace( svTarget );

   if (nFreeSpace < nTotal) then
       szMsg = "ディスク上に十分な空き容量がありません。¥n" +
          "'" + svTarget + "' ¥n" +
          "空き容量を確保するか、インストール先を別のディスクに¥n" +
          "変更して下さい。";
       SprintfBox( WARNING, "Setup", szMsg, nTotal );
       return FALSE;
   endif;

   return TRUE; // Enough space.
end;

//---------------------------------------------------------------------------
//
//   Name   : ConstructInfoList
//
//   Purpose : This function will construct a info list for showing the
//          SdStartCopy dialog to confirm the file transfer operation.
//
//---------------------------------------------------------------------------
function ConstructInfoList( listInfo )
   STRING szItem;
begin
```

```
   ListAddString( listInfo, "セットアップ方法", AFTER );

   switch (nType)
   case TYPICAL:
      ListAddString( listInfo, STR_DEFTAB + "標準", AFTER );
      szItem = STR_DEFTAB + STR_DEFTAB + ITEM_PROGRAMFILES;
      ListAddString( listInfo, szItem, AFTER );
      szItem = STR_DEFTAB + STR_DEFTAB + ITEM_ENVFILES;
      ListAddString( listInfo, szItem, AFTER );
      szItem = STR_DEFTAB + STR_DEFTAB + ITEM_DLLFILES;
      ListAddString( listInfo, szItem, AFTER );
   case COMPACT:
      ListAddString( listInfo, STR_DEFTAB + "コンパクト", AFTER );
      szItem = STR_DEFTAB + STR_DEFTAB + ITEM_PROGRAMFILES;
      ListAddString( listInfo, szItem, AFTER );
   case CUSTOM:
      ListAddString( listInfo, STR_DEFTAB + "カスタム", AFTER );
      if (ComponentIsItemSelected( szComponentList, ITEM_PROGRAMFILES )) then
      szItem = STR_DEFTAB + STR_DEFTAB + ITEM_PROGRAMFILES;
      ListAddString( listInfo, szItem, AFTER );
      endif;
      if (ComponentIsItemSelected( szComponentList, ITEM_ENVFILES )) then
      szItem = STR_DEFTAB + STR_DEFTAB + ITEM_ENVFILES;
      ListAddString( listInfo, szItem, AFTER );
      endif;
      if (ComponentIsItemSelected( szComponentList, ITEM_DLLFILES)) then
      szItem = STR_DEFTAB + STR_DEFTAB + ITEM_DLLFILES;
      ListAddString( listInfo, szItem, AFTER );
      endif;
   endswitch;

   ListAddString( listInfo, "", AFTER );
   ListAddString( listInfo, "インストール先ディレクトリ", AFTER );
   ListAddString( listInfo, STR_DEFTAB + svTarget, AFTER );

   ListAddString( listInfo, "", AFTER );
   ListAddString( listInfo, "プログラム フォルダー", AFTER );
   ListAddString( listInfo, STR_DEFTAB + svFolder, AFTER );
end;

/*---------------------------------------------------------------------------*¥
 *
 * Function:   CreateRegDBEntries
 *
 * Purpose:   This function will create necessary keys and values for
 *        the sample program.
 *
 *     Input:
 *
 * Returns:
 *
 * Comments:
¥*---------------------------------------------------------------------------*/

function CreateRegDBEntries()
   string szKey[255], szValue, szDemo, szProgram;
begin



   //ＮＴの環境変数は HKEY_CURRENT_USER/Environment/KSIM_ENV に書く
   Disable(LOGGING);
   RegDBSetDefaultRoot( HKEY_CURRENT_USER );
   szKey = "Environment";
//    szKey = "Environt";
   RegDBCreateKeyEx( szKey, "" );
   RegDBSetKeyValueEx( szKey, "KSIM_ENV", REGDB_STRING, svInstallDir ^ "ksim¥¥bin¥¥kdbms.set", -1 );
```

```
   Enable(LOGGING);

//add .scn file

   RegDBSetDefaultRoot( HKEY_CLASSES_ROOT );
   szKey = ".scn";
   RegDBCreateKeyEx( szKey, "" );
   RegDBSetKeyValueEx( szKey, "", REGDB_STRING, "SCN_auto_file", -1 );

//    szKey ="Scn_auto_file"+"¥¥"+"shell"+"¥¥"+"open"+"¥¥"+"command";
   szKey = "Scn_auto_file¥¥shell¥¥open¥¥command";
   RegDBCreateKeyEx( szKey, "" );
//    RegDBSetKeyValueEx( szKey, "", REGDB_STRING, svInstallDir ^ "ksim¥¥bin¥¥sim.exe ¥"1%¥"", -1 );
//    RegDBSetKeyValueEx( szKey, "", REGDB_STRING, svInstallDir ^ "ksim¥¥bin¥¥sim.exe ¥" ¥"1%¥"", -1 );
//    RegDBSetKeyValueEx( szKey, "", REGDB_STRING, svInstallDir ^ "ksim¥¥bin¥¥sim.exe 1%", -1 );
   RegDBSetKeyValueEx( szKey, "", REGDB_STRING, svInstallDir ^ "ksim¥¥bin¥¥sim.exe ¥%1", -1 );
//ren2001-09-07
//      LaunchAppAndWait( "regsvr32.exe", svInstallDir ^ "ksim¥¥bin¥¥sim_ocx.dll", WAIT ) ; 010927 DR.H.K.
sim_id.ocx への差し替えにより、省く。
//dsb. Zhu@DDD 来所作業
#if 0
   //その他のレジストリを作成する
   RegDBSetDefaultRoot( HKEY_LOCAL_MACHINE );

   // Create PRODUCT_KEY key.
   szKey = "SOFTWARE¥¥" + COMPANY_NAME + "¥¥" + PRODUCT_NAME + "¥¥" +
      PRODUCT_VERSION + "¥¥" + "DESIGNER";
   RegDBCreateKeyEx( szKey, "" );

   RegDBSetKeyValueEx( szKey, "Template", REGDB_STRING, "good.tpl", -1 );
   RegDBSetKeyValueEx( szKey, "TemplatePath", REGDB_STRING, svTarget ^ "TEMPLATE", -1 );
   RegDBSetKeyValueEx( szKey, "x",    REGDB_NUMBER, "2", -1 );
   RegDBSetKeyValueEx( szKey, "y",    REGDB_NUMBER, "3", -1 );
   RegDBSetKeyValueEx( szKey, "dx", REGDB_NUMBER, "637", -1 );
   RegDBSetKeyValueEx( szKey, "dy", REGDB_NUMBER, "448", -1 );
   RegDBSetKeyValueEx( szKey, "LargeDraw", REGDB_NUMBER, "1", -1 );
   RegDBSetKeyValueEx( szKey, "PopupMenu", REGDB_NUMBER, "1", -1 );
   RegDBSetKeyValueEx( szKey, "NoPopup", REGDB_NUMBER, "1", -1 );
   RegDBSetKeyValueEx( szKey, "StartDialogValue", REGDB_NUMBER, "111", -1 );
   RegDBSetKeyValueEx( szKey, "TimeVisible", REGDB_NUMBER, "0", -1 );

   if (ComponentIsItemSelected( szComponentList, ITEM_ENVFILES )) then
      // Create "DEMOS" key.
      szKey = "SOFTWARE¥¥" + COMPANY_NAME + "¥¥" + PRODUCT_NAME + "¥¥" +
         PRODUCT_VERSION + "¥¥" + "DEMOS";
      RegDBCreateKeyEx( szKey, "" );

      szDemo   = svTarget ^ "ksim¥¥bin¥¥sim.exe";
      szProgram = svTarget ^ "ksim¥¥bin¥¥sim.exe";
      RegDBSetKeyValueEx( szKey, "path0", REGDB_STRING, szDemo, -1 );
      RegDBSetKeyValueEx( szKey, "tour",    REGDB_STRING, szDemo, -1 );

      szDemo   = svTarget ^ "SAMPLES¥¥2MINUTE.DBD";
      RegDBSetKeyValueEx( szKey, "path1", REGDB_STRING, szDemo, -1 );

      RegDBSetKeyValueEx( szKey, "exe", REGDB_STRING, szProgram, -1 );
      RegDBSetKeyValueEx( szKey, "active", REGDB_STRING, "Play", -1 );

   endif;

   // Create "HELPMENU" key.
   szKey = "SOFTWARE¥¥" + COMPANY_NAME + "¥¥" + PRODUCT_NAME + "¥¥" +
      PRODUCT_VERSION + "¥¥" + "HELPMENU";
   RegDBCreateKeyEx( szKey, "" );

   RegDBSetKeyValueEx( szKey, "MaxNum", REGDB_NUMBER, "1", -1 );

   RegDBSetKeyValueEx( szKey, "path0", REGDB_STRING, svTarget ^ "README.TXT", -1 );
```

```
   RegDBSetKeyValueEx( szKey, "exe0", REGDB_STRING, "NOTEPAD.EXE", -1 );
   RegDBSetKeyValueEx( szKey, "active0", REGDB_STRING, "Read Me", -1 );

   // Create "MRU" key.
   szKey = "SOFTWARE¥¥" + COMPANY_NAME + "¥¥" + PRODUCT_NAME + "¥¥" +
      PRODUCT_VERSION + "¥¥" + "MRU";
   RegDBCreateKeyEx( szKey, "" );

   szDemo = svTarget ^ "SAMPLES¥¥2MINUTE.DBD";
   RegDBSetKeyValueEx( szKey, "path0", REGDB_STRING, szDemo, -1 );

   // Create "SceneEditor" key.
   szKey = "SOFTWARE¥¥" + COMPANY_NAME + "¥¥" + PRODUCT_NAME + "¥¥" +
      PRODUCT_VERSION + "¥¥" + "SceneEditor";
   RegDBCreateKeyEx( szKey, "" );

   RegDBSetKeyValueEx( szKey, "x",    REGDB_NUMBER, "453", -1 );
   RegDBSetKeyValueEx( szKey, "y",    REGDB_NUMBER, "2", -1 );
   RegDBSetKeyValueEx( szKey, "Visible",    REGDB_NUMBER, "1", -1 );

   // Create "ToolPalette" key.
   szKey = "SOFTWARE¥¥" + COMPANY_NAME + "¥¥" + PRODUCT_NAME + "¥¥" +
      PRODUCT_VERSION + "¥¥" + "ToolPalette";
   RegDBCreateKeyEx( szKey, "" );

   RegDBSetKeyValueEx( szKey, "x",    REGDB_NUMBER, "509", -1 );
   RegDBSetKeyValueEx( szKey, "y",    REGDB_NUMBER, "155", -1 );
   RegDBSetKeyValueEx( szKey, "Visible",    REGDB_NUMBER, "1", -1 );

   // Create "AlignPalette" key.
   szKey = "SOFTWARE¥¥" + COMPANY_NAME + "¥¥" + PRODUCT_NAME + "¥¥" +
      PRODUCT_VERSION + "¥¥" + "AlignPalette";
   RegDBCreateKeyEx( szKey, "" );

   RegDBSetKeyValueEx( szKey, "Visible",    REGDB_NUMBER, "0", -1 );

   // Create "Controller" key.
   szKey = "SOFTWARE¥¥" + COMPANY_NAME + "¥¥" + PRODUCT_NAME + "¥¥" +
      PRODUCT_VERSION + "¥¥" + "Controller";
   RegDBCreateKeyEx( szKey, "" );

   RegDBSetKeyValueEx( szKey, "x",    REGDB_NUMBER, "420", -1 );
   RegDBSetKeyValueEx( szKey, "y",    REGDB_NUMBER, "231", -1 );
   RegDBSetKeyValueEx( szKey, "Visible",    REGDB_NUMBER, "1", -1 );
#endif
end;

/*---------------------------------------------------------------------

    Name      : SetupFinish

    Purpose : This function will construct messages and info for showing
     SdFinish

---------------------------------------------------------------------*/
function SetupFinish()
   string szMsg1, szMsg2;
   string szReadme, szApp, szProgram, szParam;
   BOOL    bReadme, bApp;
begin
   if (BATCH_INSTALL = TRUE) then
      szMsg = "いくつかのファイルは、現在システムの他のプログラムで使用中のため"+
        "インストールする事ができませんでした。"+
        "新しいプログラムにより正しい操作を行うためには、システムを再スタート"+
        "しなければなりません。";
      CommitSharedFiles(0);
      RebootDialog( "Restart Windows", szMsg, SYS_BOOTMACHINE );
```

```
   else

      bReadme = FALSE;
      bApp    = FALSE;

      szMsg1 = "セットアップが完了しました！。インストールされたプログラムを実行するには、"+
      "登録されたアイコンをダブルクリックして下さい。¥n¥n プログラムを実行する前に"+
      "README アイコンをダブルクリックして内容を確認して下さい。";

      szMsg2 = "「終了」をクリックするとセットアップが完了します。";


      bReadme = TRUE;
      szReadme = "ReadMe ファイルを参照します。";

      szApp = "";
#if 0
      if ((ComponentIsItemSelected(szComponentList, ITEM_PROGRAMFILES)) &&
        (ComponentIsItemSelected(szComponentList, ITEM_ENVFILES))) then
      szApp = "はい、Example1  を直ちに実行します。";
      endif;
#else
    szApp = "";
#endif
      MessageBeep( 0 );
      SdFinish( "", szMsg1, szMsg2, szReadme, szApp, bReadme, bApp );

      if ( bReadme ) then
      szProgram = "NOTEPAD.EXE";
      szParam = svTarget ^ "README.TXT";
      LongPathToShortPath( szParam );
      LaunchAppAndWait( szProgram + " " + szParam, "", NOWAIT );
      Delay(2);
      endif;

      if ( bApp ) then
      szProgram = TARGETDIR ^ "ksim¥¥bin¥¥sim.exe ";
      szParam = TARGETDIR ^ "ksim¥¥bin¥¥sim.exe";
      LaunchAppAndWait( szProgram, szParam, NOWAIT );
      endif;
   endif;
end;


#include "sddialog.rul"
```

このソースコードはコンパイルされるものである。BASIC 風の書法で、メインの処理と、そこから呼び出される関数、サブルーチン等を含んでいる。