# webMethods JIS:

## Getting Started

Version 9.0

November 2009
(originally released January 2005)

This document applies to webMethods JIS Version 9.0 and to all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Copyright © 1992–2009 Software AG, Darmstadt, Germany and/or Software AG USA, Inc., Reston, VA, United States of America, and/or their suppliers. All rights reserved.

The name Software AG, webMethods, and all Software AG product names are either trademarks or registered trademarks of Software AG and/or Software AG USA, Inc. Other company and product names mentioned herein may be trademarks of their respective owners.

**Document ID: JIS-GS-90-20091130**

# Table of Contents

# List of Figures

# List of Tables

# List of Examples

# About this Guide

Only webMethods JIS products provide the benefits of both host-based computing and client/server computing – through the JIS KnowledgeBase and the Automated Conversion Environment (ACE). Together they form the powerful core of a family of graphical generation products that can quickly and easily produce Java classes and XHTML pages for mainframe and midrange applications.

If you are working with webMethods JIS products for the first time, this book will get you started. All the information you need in order to begin a conversion can be found right here. All the chapters are introduced by summary pages. You can use a summary page to find out if a chapter contains the topics you are looking for or for some background information on the topics you are about to learn.

**Chapter 1 -** "**Developing Skills in ACE**" shows you how to install ACE on your system and how to use the help tools that ship with ACE. The concept of wizards is also introduced here.

In **Chapter 2 -** "**Application Foundations**" you learn how to run ACE for the first time. You learn about Applications, screen images, Subapplications and libraries. The last section of this chapter introduces KnowledgeBase concepts and techniques.

In **Chapter 3 -** "**Workflow**" you learn about the ACE workflow. You are provided with a handy flow chart illustrating each step in a conversion project. In this chapter you establish the ACE development environment, create an Application, screen images and Subapplications. The chapter closes with lessons about the development views in ACE and lessons about the runtime.

**Chapter 4 -** "**Runtime**" explains how to set up the runtime environment and how to run Applications. In this chapter you gain insight as to what the end user sees during a runtime. You are introduced to the concepts of testing runtimes, communication between PC's and hosts, and working with emulators.

**Chapter A -** "**JIS DDS Compiler**" describes the automatic and manual installation and operation procedures for the JIS DDS Compiler. You must have the JIS DDS Compiler installed so that ACE can read iSeries files.

The following table maps the new terminology, which is now used in the product, with the old terminology, which is still used in the documentation.

## Documentation Set

webMethods JIS is supplied with the manuals shown below. The documentation is delivered in Adobe Acrobat Reader Portable Document Format (PDF). No hardcopy documentation is provided, but you can print the PDF files on your local printer.

**Table 1. webMethods JIS documentation set**

| This book... | Contains... |
|---|---|
| ***webMethods JIS: Getting Started with the Automated Conversion Environment*** | Startup information and an introduction to the Automated Conversion Environment (ACE). |
| ***webMethods JIS: Basic User's Guide*** | Full explanations of the ACE Views and how to use them |
| ***webMethods JIS: Advanced Topics*** | Explanations of advanced features that give your application extra functionality. |
| ***webMethods JIS: KnowledgeBase User's Guide*** | In-depth information about the way the ACE KnowledgeBase is designed and how to work with it. |
| ***webMethods JIS: Java Client User's Guide*** | Information for migrating your host application to Java. |
| ***webMethods JIS: XHTML Client User's Guide*** | Information for migrating your host application to an XHTML web application. |

## Document Conventions

The following conventions are used throughout this manual.

Table 2. Documentation conventions

| Convention | Description |
|---|---|
| Click | Position the mouse pointer on the control and quickly press and release the left mouse button **once**. (Unless the right mouse button is explicitly specified, you should click the left mouse button.) |
| Double-click | Position the mouse pointer on the control and quickly press and release the left mouse button **twice**. (Unless the right mouse button is explicitly specified, you should double-click the left mouse button.) |
| UPPERCASE | Uppercase letters are used for the names of files. For example, a panel file with the name Menu, will be expressed as MENU.PNL**.** |
| *italics* | Names of applications, programs, menus, dialog boxes, and libraries. |
| **Bold** | Menu options, and items, dialog boxes and items to be selected from a dialog box. The names of pull-down menus. |
| ***Bold Italics*** | Pattern definitions, representation definitions, message definitions, method names, layout names, section names, selection definitions, function definitions. |
| **BOLD + UPPERCASE** | Keyboard shortcuts: Press the **SHIFT** key. Press **CTRL + Z**. |

## Viewing the Documentation Online

You can also access the latest version of the documentation for Software AG products at http://documentation.softwareag.com/. As new versions become available, the documentation on this web site will be updated and the previous versions will be migrated to the Software AGdocumentation web site at http://servline24.softwareag.com/public/. If you have a maintenance contract, you can view all versions of documentation on this web site. You will find instructions for registering and obtaining a userid and password on the documentation web site.

# Chapter 1. Developing Skills in ACE

In this chapter you learn about concepts involved in conversions. Depending on your particular background, the concepts and terminology introduced here may seem familiar to you. The chapter begins by showing you how to install webMethods JIS (JIS) on your system. With JIS installed you can begin learning about the Automated Conversion Environment (ACE) by using the online tutorials. The tutorials are the best way to get started using JIS and it is recommended you invest some time making the most of them.

This chapter describes:

- Installation
- Setup Procedure On the PC
- Setup Procedure on the iSeries
- The ACE Tutorial
- Wizards

Before getting started you should install JIS on your computer system. There are certain system requirements JIS demands in order to run effectively. Once installed, launching JIS is done from the Windows *Start* menu or by double clicking on a desktop icon.

Online learning is an integral part of making webMethods JIS work for you. Many of the best features ACE has to offer are explained in the comprehensive online learning resources.

Step-by-step wizards take you from the beginning to the end of many operations introduced in this chapter. Wizards make difficult tasks simple by prompting you for responses that lead you to completing a task quickly.

## Installation

To run ACE efficiently, the development system should have:

- CPU 200 Mhz or higher.
- Video board and monitor SVGA 800x600 or higher.
- Minimum 64MB RAM (128MB is recommended).
- A minimum of 200MB free space on your hard disk for ACE installation.
- An additional 300MB are required to convert a 1,000 screen host application.

  This estimation can vary in accordance to the amount of information carried by the screens and does not include space necessary to hold DDS, BMS or MFS source files.

- A mouse or other pointing device.
- Host Communication: TCP/IP.
  Where this is not available, the following is needed:
  - iSeries applications - a Router.
  - Mainframe applications - an Emulator.
- The full version of the Wise installation program is strongly recommended.

## Setup Procedure On the PC

To install ACE from CD-ROM:

1 With the PC turned off, insert the copy protection key in the PC's printer port.
2 Turn on the PC. Place the CD into the PC's drive, and run *setup* from the drive's root directory. The *Setup* wizard is automatically launched.
3 Follow the instructions in the *Setup* wizard.

**Note:** Certain webMethods JIS features require the Borland Database Engine (BDE). The BDE installation accompanies webMethods JIS's installation.

### Installing Multiple Versions

You can install more than one version of ACE. If you have a previous version of ACE installed on your machine, the *Setup* wizard gives you the choice of either upgrading your existing version or making a separate installation for the new version.

If you choose to upgrade, you will not be able to reinstall the older version unless you first uninstall the upgraded version of ACE.

## Setup Procedure on the iSeries

The JIS DDS Compiler processes DDS files into the DDO files that are used by ACE. The first stage in using ACE with DDS is to install the Compiler. Full installation instructions are given in Appendix A of this book. Once installed you copy the compiler output to your PC, and process it in the ACE Converter.

Figure 1 illustrates the flow of the DDS files from the iSeries to the PC. The JIS DDS Compiler processes the DDS source files into DDO files and places them in a shared folder on the iSeries. Then you copy the DDO files to a storage directory on the PC. This storage directory should be external to the ACE file structure. Keep a record of where the DDO files are on the PC. You will need to supply this information when you create screen images from the DDO files.

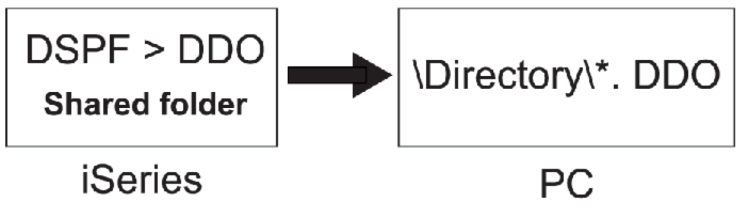


**Figure 1. DDS files from iSeries to PC**

Screen images are the objects that ACE uses to generate the graphical interface.

### JIS DDS Compiler Operating Instructions

For a complete description of the DDS Compiler's features and details on how to use them, see "JIS DDS Compiler" on page 99.

## The ACE Tutorial

The tutorial contains three animated online presentations and two cue card sessions. Each presentation takes approximately ten minutes to complete. The cue card sessions follow the animated presentations and incorporate what you learned in the presentations. Each lesson builds on the information gained in the previous one. Therefore, it is recommended you complete the lessons in the order in which they are presented to you on the tutorial's *Welcome* splash screen.

To launch the tutorial, in the *Help* menu choose *Tutorial*. The title and scope of each lesson in the tutorial is outlined below.

**Table 3. ACE tutorial parts (Sheet 1 of 2)**

| Section | Description |
|---|---|
| **General Concepts** | An online presentation that provides an overview of the ACE workflow without going into the details of each step. |

**Table 3. ACE tutorial parts (Sheet 2 of 2)**

| Section | Description |
|---|---|
| **Working with Views** | An online presentation that explains what tasks can be performed in each view. There are 8 views in ACE and in this presentation you learn about all of them. In each view a different aspect of the host screen or the generated GUI is emphasized. |
| **First Steps in ACE** | Using cue cards you work through examples, employing what you learned in the previous two online presentations. |
| **KnowledgeBase Principles** | An online presentation covering KnowledgeBase terminology and principles. |
| **Tuning the KnowledgeBase** | Using cue cards you work in ACE using the KnowledgeBase. Examples demonstrate how to modify the KnowledgeBase. These examples are based on the topics learned in the previous presentation. |

## Wizards

ACE uses wizards to guide you from the beginning to the end of the conversion process. Wizards also assist you with certain advanced features.

Wizards help you to:

- Create new Applications
- Create, edit and maintain screen images for your Application
- Create your Application's Subapplications
- Create tabs on a Subapplication
- Create an executable file from your Application
- Run your Application's executable file
- Create a stand-alone installation of your Application

The wizards contain extensive help on the options available, including variable text that explains the effect of the currently selected option.

# Chapter 2. Application Foundations

Here you learn what forms the backbone of all webMethods JIS applications. The topics in this chapter lay the groundwork which exists before you begin to modify the GUI look of host screens. This chapter presents these topics in theory only. In the following chapter you use wizards to create some of the items which are covered here.

This chapter includes:

- Running ACE For the First Time
- What is an Application?
- What is a Screen Image?
- What is a Subapplication?
- What is a Library?
- Packaging Applications
- The KnowledgeBase and How it Works
- Pattern Definitions

Now that JIS is installed on your computer you will be able to launch the program. Getting the organizational structure of your Application right from the start can save you time in the long run. You learn about the structure of an Application from the top down. The JIS Application is at the top of the organizational structure of a converted host application. The library is under the Application level in the organizational structure of a converted host application. The screen image is a picture of the host screen without a GUI. ACE analyzes a screen image before converting it into a GUI. Once the conversion takes place the screen image is called a GUI window. The Subapplication is where local modifications are made to customize the GUI look.

The final topic of this chapter contains information about the KnowledgeBase**.** The KnowledgeBase is the tool which allows you to fully customize JIS Applications. In this chapter you learn how the KnowledgeBase works and some KnowledgeBase terminology.

## Running ACE For the First Time

To run ACE either:

- Double click the ACE icon

  -OR-
- Select the ACE folder from the *Start* menu.

The product's splash screen is displayed for several seconds and then you are presented with the conversion environment. To get acquainted with ACE it is recommended that you work through the online training tools.

The first time you enter ACE you will have no Applications defined. You must create a new Application. To create a new Application, from the File menu select New > New Application. Follow the instructions in the New Application wizard.

Close open Applications using the Close Application option in the File menu.

**Note:** You can use the online training tools without having an Application open.

Subsequently, when you enter ACE you may open an existing Application or create additional new Applications.

## What is an Application?

The ACE engine generates a graphical equivalent for each host screen belonging to a host application. The collection of all these graphical equivalents form an Application. The Application is tested, converted into an executable and loaded onto the customer's server machine. When creating an Application you set certain parameters that are applied to each screen contained in the Application. Establishing the Application is the first step in the conversion process.

## Application Properties

Application properties are set step-by-step in the *New Application* wizard. The table below shows what information the wizard asks for when you create an Application.

**Table 4. Application properties and their descriptions (Sheet 1 of 2)**

| Property | Description |
|---|---|
| **Application Name** | The eventual filename of the executable GUI. The name cannot be `AFTER, BEFORE, CLASS, GLOBAL, GLOBALS, INIT, METHODS, NULL_OBJ, PARMS, RT, SUPER, TRUE_OBJ,` or `VARS.` |

**Table 4. Application properties and their descriptions (Sheet 2 of 2)**

| Property | Description |
|---|---|
| **Language** | The language of the host application and the language that the graphical interface will use for dialog boxes, menus and the like.<br><br>webMethods JIS currently supports the following language groups and languages:<br><br>*Western European languages:* English (US and UK), German, French, Spanish, Italian, Dutch, Swedish, Portuguese, Swiss-German.<br><br>*Western European languages*: Czech, Polish, Hungarian, Romanian, Slovenian, Slovak, Croatian, Albanian.<br><br>*Cyrillic languages:* Russian, Bulgarian, Ukrainian, Belorussian, Macedonian, Serbian.<br><br>*Other languages:* Greek, Turkish, Chinese (simplified), Japanese.<br><br>For instruction on translating certain keyboard keys, see also the section about translating accelerator keys in *webMethods JIS: Advanced Topics*. |
| **Default Close Key** | Specifies the default key for exiting a Subapplication/host screen. |
| **List Scrolling Keys** | Specifies the default keys that will scroll host application lists in runtime. You will be able to override this choice on a per screen basis when you create Subapplications. |
| **Resolution** | Specifies the display resolution of the runtime. |
| **With Toolbar** | Specifies whether or not a toolbar will be displayed in runtime. |

**Note:** The Application properties cannot be changed once the Application is created.

## What is a Screen Image?

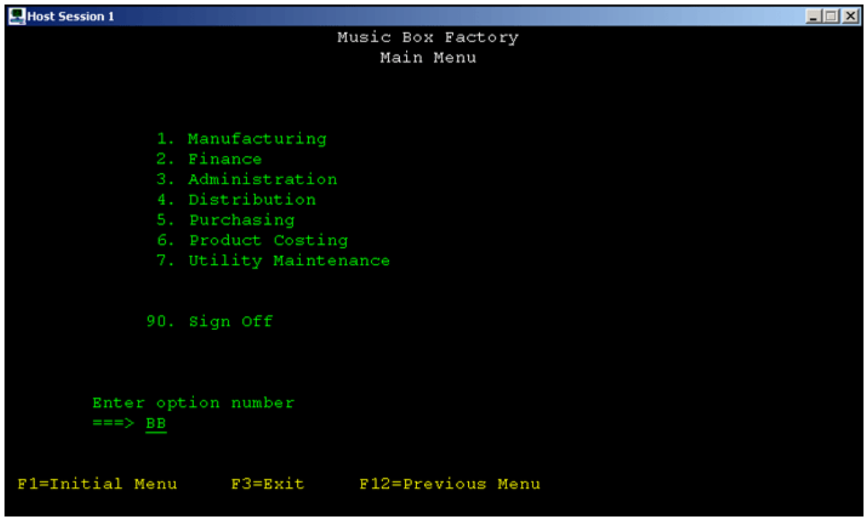
A screen image is a picture of a host screen. ACE cannot work directly on the host, so for each host screen you must create a screen image file on your development platform. A screen image can be a screen capture—a simple snapshot of the host screen, or the screen image can be an ACE-readable form of a host Screen Definition File: Mainframe BMS/MFS or iSeries DDS.

Screen images are created step-by-step in the *Create Screen Images* wizard. The wizard includes complete explanations of how to configure the screen images. You can create more than one screen image from a single host screen or SDF.

A GUI screen is automatically generated from each host screen. The automation is accomplished by the KnowledgeBase. The KnowledgeBase identifies the information present on a host screen and generates an appropriate graphical control to represent this information on the GUI.

## How a Screen Image Appears

A screen image looks like the original host screen.



```
                                Music Box Factory
                                    Main Menu


               1. Manufacturing
               2. Finance
               3. Administration
               4. Distribution
               5. Purchasing
               6. Product Costing
               7. Utility Maintenance


              90. Sign Off


         Enter option number
         ===> BB

 F1=Initial Menu     F3=Exit     F12=Previous Menu
```

**Figure 2. Screen image**

The above image is taken from a “green on black” screen.

ACE takes this screen and changes it into a GUI.

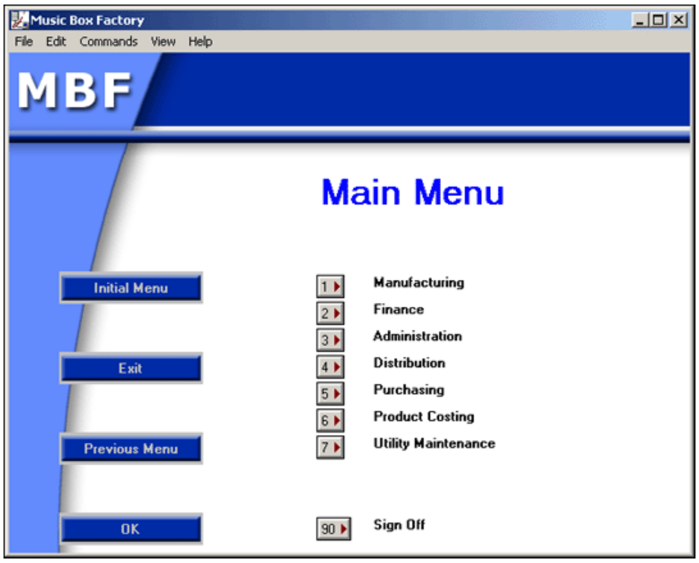


Figure 3. A GUI window

## What is a Subapplication?

A Subapplication is the graphical equivalent (GUI) generated from a host screen image. A single screen image can generate only one Subapplication. However, you can create more than one screen image from a single Screen Definition File. You can also capture and save a single screen many times.

Therefore, you can create more than one Subapplication for each host screen, whether from captures or SDFs:

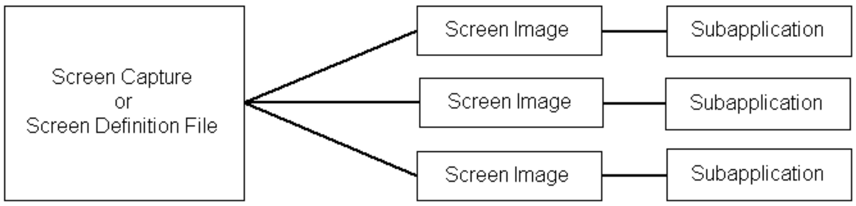


Figure 4. Multiple subapplication creation from a single host screen

## How a Subapplication Appears

A Subapplication can be viewed in various screen and graphical views:

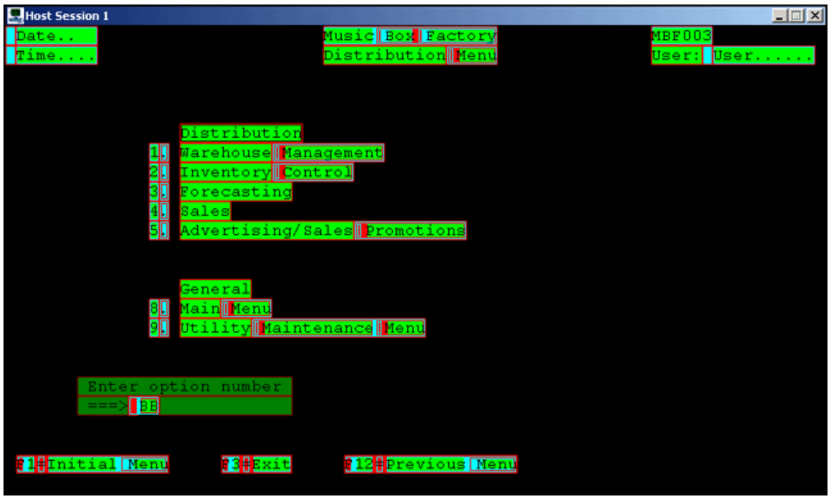
**Table 5. Screen and Window views**

| View Type | Views |
|---|---|
| **Screen views** | • Host View<br>• Layout View<br>• Analysis View<br>• Runtime Field Information View<br>• Runtime Screen Identification View |
| **Graphical views** | • Design View<br>• Test View |

Each of these views emphasizes a different aspect of the host screen or the GUI generated from it.

### Subapplications In Analysis View

How a Subapplication appears in Analysis View:

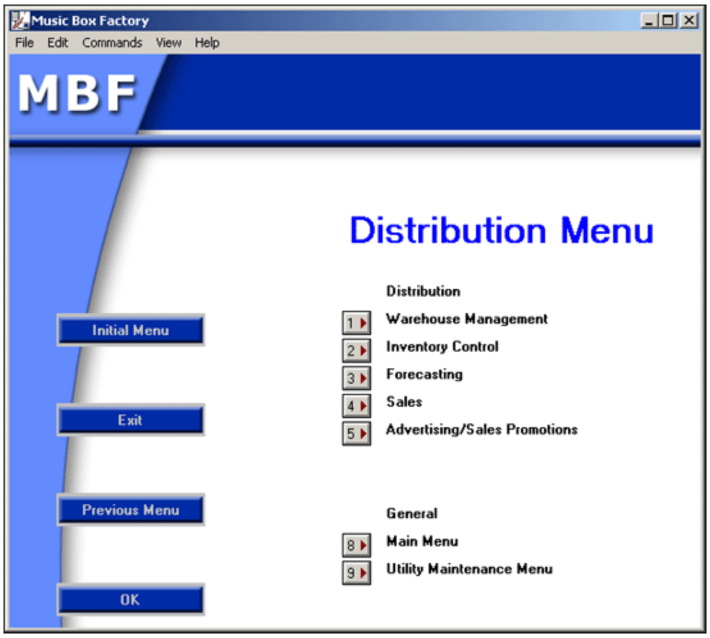


```
Date..                             Music Box Factory                 MBF003
Time....                           Distribution Menu                 User: User......


                   Distribution
                1. Warehouse Management
                2. Inventory Control
                3. Forecasting
                4. Sales
                5. Advertising/Sales Promotions


                   General
                8. Main Menu
                9. Utility Maintenance Menu


     Enter option number
     ===> BB


F1=Initial Menu     F3=Exit     F12=Previous Menu
```

Figure 5. Subapplication in Analysis view

## Subapplications In Test View

How a Subapplication appears in Test View:



Distribution Menu

Distribution
1 Warehouse Management
2 Inventory Control
3 Forecasting
4 Sales
5 Advertising/Sales Promotions

General
8 Main Menu
9 Utility Maintenance Menu

**Figure 6. Subapplication in Test view**

## The Relationship of Subapplications to Applications

A Subapplication is the graphical equivalent of a single host screen. Together, the Subapplications form an Application. An Application is the graphical equivalent of the entire host application.

## What is a Library?

A library is a number of Subapplications grouped together, with the group having a name. An Application can have many libraries, with each library containing many Subapplications. Libraries are therefore an intermediate level of organization between Applications and Subapplications:

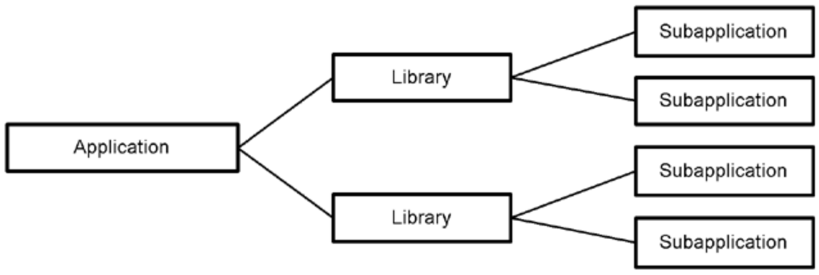


**Figure 7. Library structure**

Libraries are logically equivalent to Applications: You can set global properties for a library without affecting the properties of the other libraries. Each new library has its own KnowledgeBase by default. These default settings can be changed by adding a new section to an INI file located under the Applications directory. The syntax of the new section is as follows:

```
[Converter]
NewLibraryCreationType=
```

(Where "0" zero denotes sharing the KnowledgeBase with the Application. Where "1" one denotes a library that uses a KnowledgeBase separate from the Application.)

## Using Libraries

Libraries are most useful when your host application's screens divide up naturally into categories, either because the screens belong to different Screen Model Types, or because a number of screens have similar characteristics. Libraries also allow different developers to work simultaneously without each having to worry about how their KnowledgeBase modifications will affect the other developer's work.

## Benefits of Using a Library

Using libraries gives you the following advantages:

- *DOS File Handling* - Reduces the amount of files to handle per directory, by classifying them into libraries.
- *Product Management* - Enables you to send updates for your runtime Application which are specific to that particular library. For example, if a correction is made in one of the libraries, then only that specific library update can be sent. If there are Subapplications which have changed, then only the library which contains those particular Subapplications could be shipped. For example, suppose you have an Application with 8,000 screens, and 100 of

those screens were changed by the company. You could create a new library for only those corrected screens.

- *Classification* **-** Enables better organization and maintenance of the Application. Dividing the Application into libraries assists in better management and distribution. For example, you can create a library for Screen Model Type 2 host screens and a library for Screen Model Type 5 host screens.
- *Team Work* - Enables several developers to work on the same Application, where each developer works on his own library.
- *KnowledgeBase Explosion* **-** Prevents KnowledgeBase explosion, by making KnowledgeBases specific per library. This reduces the size of the overall KnowledgeBase file, making it easier to manage and maintain.

## Creating a Library

To create a new library, perform the following:

1. Open an Application.
2. From the *File* menu select *New* > *New Library*. The *New Library* wizard opens.
3. Follow the wizard steps. In these steps you must provide the following:
   - The Screen Model Type of the host screens represented by this library's Subapplications. You can choose *Model 2 (24x80), Model 3(32x80), Model 4 (43x80)* or *Model 5 (27x132)*. The choice you make stipulates that all Subapplications in this library should be of your chosen Screen Model Type.
   - The library name.

If you look at your directory structure you will see that under the APPLS directory, a new sub-directory is included and named <LIBRARY_NAME.LIB>. The new subdirectory contains the library's directory structure, and stores all the information about the library. Although a library is associated with a specific Application, the library does not reside in the Application's directory. Inside each library directory lies a file called library.ion that contains specific information for that library.

**Note:** Ensure that each library has a unique name. An error message is generated when a duplicate name is entered in the *Name* field for new Applications and libraries.

## Calling up a Library

The left hand side drop-down box of the ACE toolbar contains the names of the current Application and all of the Application's libraries:

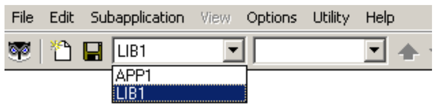


**Figure 8. Library drop-down in ACE**

From the list, select the library you wish to open.

## Deleting a Library

To delete a library:

1 From the *File* menu, select *Delete* > *Library*. The following dialog box appears.

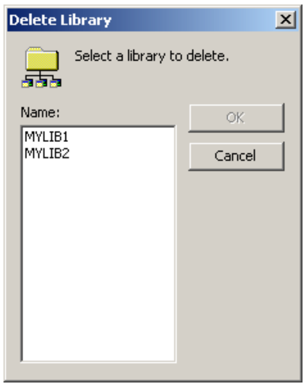


**Figure 9. Delete Library dialog box**

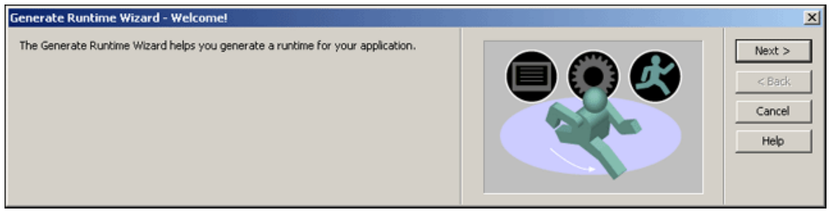
2 Select the library you wish to delete and click *OK*.

## Compiling a Library

The first time you compile a library (Generate Runtime) you must compile it with all its Subapplications. This is configured in either the Application's or the library's *Generate Runtime* wizard. After compiling the library with all its Subapplications, you can re-compile the library with only the Subapplications you choose. You can run and test libraries as if they were stand-alone Applications.

To compile a library, open a library and select *Generate Runtime* from the *File* menu. ACE will prompt you with the *Generate Runtime* wizard. Read the text located on the left side of the wizard for an explanation of the choices you are required to make. To continue, click *Next* while reading the dialogs for further information. At the end, the wizard provides a *Finish* button.

Clicking *Finish* begins the compilation using the parameters you have set in the wizard



The Generate Runtime Wizard helps you generate a runtime for your application.

**Figure 10. Generate Runtime wizard for Libraries**

In your compiled ACE Application, you can choose to run just the Application, all the libraries, or some of the libraries which are related to the Application. When compiling your Application it is advisable to first compile the libraries one by one independently. Only after this do you compile them together as an Application.

## Packaging Applications

Packed files (packages) are “archives” used for distributing and storing ACE Applications. Packed Applications contain one or more files, compressed to save space. Applications can be packaged in ACE on one computer and unpacked on other computers. Packaged Applications save time and space, and make transferring e-mail attachments faster.

Consider packaging an Application when the following situations are applicable:

- When several developers are working on different parts of the same Application in a multiple-developer environment
- When sending part, or all, of an Application to customer support
- When creating archives for backup purposes

The need for Application/library transfer generally arises when the database files on the machine of a library developer require additions or fine tuning. The library developer then transfers the library to the central developer. The central developer modifies the Application-level elements as needed, and transfers the updated development environment to the library developer.

This ensures constant updating of the development environment and efficient work assignment amongst developers.

**Note:** The Pack/Unpack utility requires the installation of Borland's Database Engine (BDE). BDE is supplied on the installation CD and is installed after ACE. If the BDE is not installed on the computer and you run the Pack/ Unpack wizard, no error message is issued, but the Pack/Unpack process will fail.

## Contents of a Package

You can choose to pack an Application, a library, or a subset of Subapplications from an Application or library.

The package can contain:

- For each Application or library, all the files in its directory unless only some of the Subapplications were selected.
- All KnowledgeBase files used by the Application or library, even those located outside the Application or library directory.
- All.INI files used by the Application or library. This also includes the relevant database information.

You can also include other files that are related to the Application or library in the package. These include:

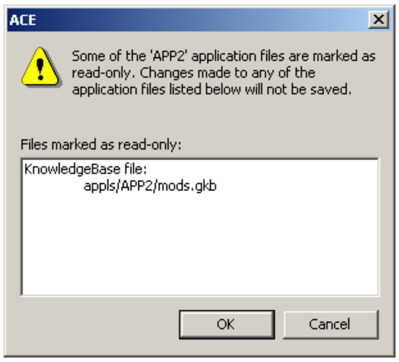
- Visual Basic and/or Java files.
- Bitmaps, even when located outside the Application or library directory.
- Files in the input directories: DDS, SDF, and screens.
- Files in the output directories: Temporary Generate Runtime, Runtime, and Runtime Installation.
- Other files on your computer.

## Supporting Multi-Developer Environments

In a multi-developer environment that uses proprietary version control or a third party version control, ACE recognizes the read-only status and generates appropriate messages.

### When the KnowledgeBase File is Read-Only

Opening an Application when a KnowledgeBase file, especially mods.gkb, is marked read-only generates the following message:


ACE

Some of the 'APP2' application files are marked as read-only. Changes made to any of the application files listed below will not be saved.

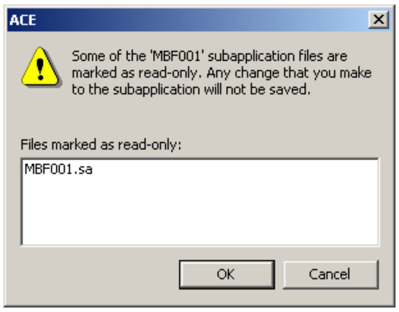
Files marked as read-only:

```
KnowledgeBase file:
        appls/APP2/mods.gkb
```

Figure 11. Warning for opening application read-only files

All Save functions in the KnowledgeBase are disabled.

Opening a Subapplication where some of its files are marked read-only generates the following message:


ACE

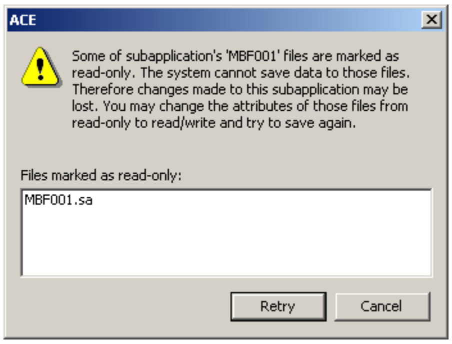
Some of the 'MBF001' subapplication files are marked as read-only. Any change that you make to the subapplication will not be saved.

Files marked as read-only:

```
MBF001.sa
```

Figure 12. Warning for opening subapplication read-only files

Attempting to save a Subapplication that is marked read-only generates the following message:


ACE

Some of subapplication's 'MBF001' files are marked as read-only. The system cannot save data to those files. Therefore changes made to this subapplication may be lost. You may change the attributes of those files from read-only to read/write and try to save again.

Files marked as read-only:

MBF001.sa

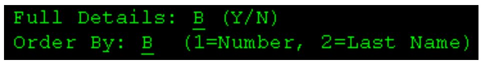
**Figure 13. Warning for saving subapplication read-only files**

The Subapplication is not saved.

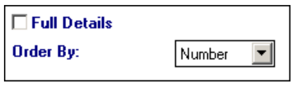
## The KnowledgeBase and How it Works

The KnowledgeBase is a set of rules for reading and understanding host screens. The rules are based on the fact that host applications present information in standard ways.

The KnowledgeBase is both flexible enough to account for wide variation within the standard ways that host screens present information and robust enough to correctly discern between similar character combinations and generate the most appropriate control:


```
Full Details: B (Y/N)
Order By: B  (1=Number, 2=Last Name)
```

*Host Screen*



*Graphical Equivalent*

You may override the KnowledgeBase default choices. Modifications you can make include changing control colors, text fonts, the size of controls, and the arrangement of the controls on the GUI. You may add and delete controls, as well as change the type of control itself.

The KnowledgeBase provides standard menus to which you can add additional menu items, or you can create completely new menus. You may also make your Application easier to navigate by grouping controls into folders on the window.

You can override the KnowledgeBase locally or globally. A local override affects the properties of the one control you've chosen to modify. A global override can change the default properties of all controls of a given type. You can also globally change how the KnowledgeBase understands host screen information. When you are satisfied with your GUI, you compile an executable and package it as an installation which you distribute to runtime users. During the compilation process you may choose to generate the GUI as a web-based XHTML executable or as a Java executable including Java classes.

## Pattern Definitions

The KnowledgeBase rules are a means of dividing up all the host screen characters into groups that contain information. These rules are called *Pattern Definitions*. Each Pattern Definition specifies a sequence of characters or, usually, a number of possible sequences. When ACE identifies a host screen character sequence that matches the criteria of a pattern definition, ACE generates the corresponding GUI element.

The Pattern Definition ***FKey*** is designed to recognize host character sequences that tell you the effect of pushing a function key.

Such sequences could be:

- F3=Exit
- pf12: Cancel
- Cmd18 - Print Screen

These character sequences have similar structure.

- Each sequence starts with a group of letters that tells you that the sequence is concerned with a function key: `F, pf, Cmd`.
- Then each sequence contains a number that identifies *which* function key is being described: `3, 12, 18`.
- The last part of each sequence is text that describes the effect of pushing the function key.
- Between the identification number and the text is a terminator. The terminator may be one of the following characters: "`=`","`:`", "`-`".

The Pattern Definition *FKey* identifies any character sequence that starts with `F` or `pf` or `cmd`, followed by a number, a termination sequence, and finally text. When *FKey* recognizes this sequence, ACE generates a push button on the GUI with descriptive text written on the button face. The instruction that a mouse click on the button should signal the host as if the corresponding function key was pressed is automatically included in the application.

### Pattern Definition Name

Pattern Definitions are identified by a unique name. The example above discussed the pattern definition named *FKey*.

### Hierarchy of Pattern Definitions

A Pattern Definition such as *FKey* is made up out of smaller Pattern Definitions. For example, *FKey* contains the Pattern Definition *FKeyPrefix* which recognizes character sequences such as `F`, `PF`, `Pf` and `CMD`. *FKey* also contains the Pattern Definition *FKeyDescription* which recognizes character sequences that make up descriptive text. These Pattern Definitions are in turn made up of still smaller Pattern Definitions.

The hierarchy of Pattern Definitions allows the KnowledgeBase to be concise and still accurately decide whether a given character sequence has the correct structure.

## Pattern Definition Type

The Pattern Definition type defines the spatial relation between the smaller Pattern Definitions that make up a larger Pattern Definition. The type defines such characteristics as whether the components of a pattern appear horizontally or vertically on the screen, or whether the components are a group of various components, or an iteration of the same component several times. A complete discussion of all the Pattern Definition types can be found in *webMethods JIS: Basic User's Guide*. Only the two lowest level Pattern Definitions are described here.

### String

String is a specific sequence of characters that appears on a screen. In contrast to the Character Set type, which defines any one character of the characters that belong to the set, String is a unique sequence of characters, that does not change such as "PASSWORD" or "USERID".

A pattern that is of String type must match the Pattern Definition in content and in length.

## Character Set

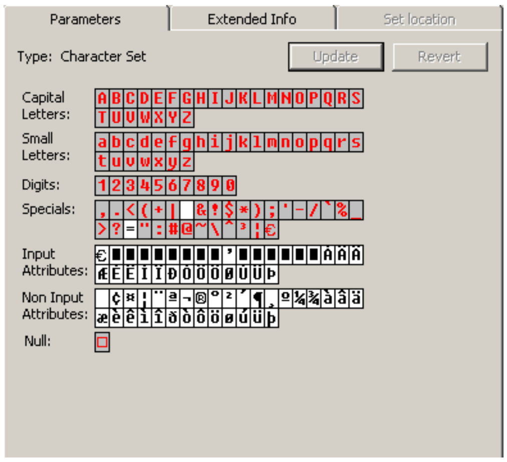
Character Set is a sub-set of all the characters. The list of all the characters includes the following:

- Capital Letters
- Lower-case Letters
- Digits
- Specials
- Input Attributes
- NonInput Attributes
- Null

A Pattern Definition of the Character Set type, matches a pattern one character long, that is one of the characters defined by the set.

### Example 1. FKeyChar character set

*FKeyChar* is a Pattern Definition of Character Set type. This Character Set includes all the capital letter and lower-case letter characters. This Pattern Definition can be used in defining other compound Pattern Definitions.



Type: Character Set

Capital Letters: A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Small Letters: a b c d e f g h i j k l m n o p q r s t u v w x y z

Digits: 1 2 3 4 5 6 7 8 9 0

Specials: , . < ( + | & ! $ * ) ; ' - / ` % _ > ? = " : # @ ~ \ ^ ³ ¦ €

Input Attributes: € █ █ █ █ █ █ █ █ ’ █ █ █ █ █ █ À Á Ä Æ È Ë Ì Ï Ð Ò Ó Ö Ø Ù Ü Þ

Non Input Attributes: ¢ ¤ ¦ ¨ ª ¬ ® ° ² ´ ¶ ¸ º ¼ ¾ à â ä æ è ê ì î ð ò ô ö ø ú ü þ

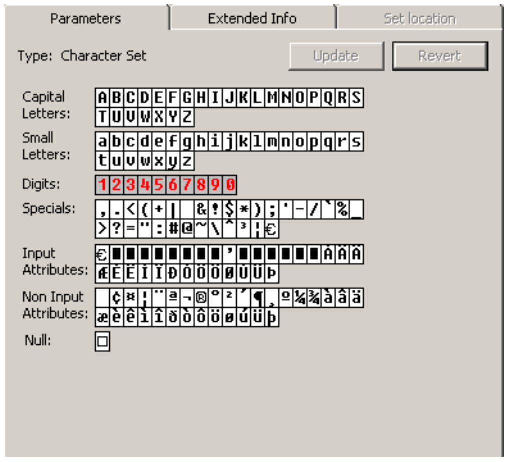
Null: □

Figure 14. FKeyChar character set

For instance, an ***FKeyWord*** Pattern Definition is an iteration of the ***FKeyChar*** Pattern Definition, that matches command names such as Cancel, Enter, etc.

**Example 2. Digit character set**

*Digit* is a Pattern Definition of Character Set type. This Character Set includes all the digits: 0-9. This Pattern Definition can be used in defining other Compound Pattern Definitions.



Type: Character Set

Figure 15. Digit character set

For instance, a *Digit1to2* Pattern Definition is a combination of two ***Digit*** Pattern Definitions, that match command key patterns such as (F)01, (PF)12, (CMD)24, etc.

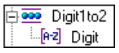


Figure 16. Digit1to2 pattern definition

## Compound Pattern Definition Types

The Compound Pattern Definition types are combinations of the two basic Pattern Definition types (character and string). Compound Pattern Definitions fulfill an important function, as they allow ACE to obtain the most comprehensive information regarding the patterns that compose a screen.

The following are the Compound Pattern Definitions and their symbols. The symbols are presented as an identification aid when creating and editing Pattern Definitions and Representation Definitions in the KnowledgeBase.

**Table 6. Compound pattern definitions and their symbols (Sheet 1 of 2)**

| Symbol | Pattern Type |
|---|---|
| ●▲■ | Dynamic Group |
| ●●● | Dynamic Iteration |
| ▲●■ | Horizontal Group |
| ●●● | Horizontal Iteration |
| ☰ | List |
| ☰ | List Column |
| ☰ | List with Parameters |
| ▲●■✓ | One Of |
| ▣ | Popup Border |

**Table 6. Compound pattern definitions and their symbols (Sheet 2 of 2)**

| Symbol | Pattern Type |
|---|---|
| ▲●■ | Scattered Group |
| ▲●■ | Vertical Group |
| ●●● | Vertical Iteration |

# Chapter 3.  Workflow

The ACE workflow comprises the steps which move a host application from its native green screen to GUI. This chapter builds on the previous chapter by showing you how to use some of the features you learned about. It also introduces the workflow concepts. Understanding the ACE workflow helps you plan ahead, saving you time and avoiding mistakes.

This chapter describes:

- The ACE workflow
- Establishing an Application
- Establishing the environment and getting down to work
- Creating and processing screen images
- Creating and processing Subapplications
- Learning about the ACE views
- Creating a runtime

So that you become familiar with the ACE interface, this chapter delves into the functions of each of the ACE views. You were introduced to the eight views in the online tutorial. The views are explained in more detail here. Each view contains dedicated menus and functions concerning the conversion process. You learn the purpose of each view plus what you see and what you can do in the views.

This chapter also discusses how to create a runtime. For more detailed information about runtime options, see Chapter 4 - "Runtime" on page 85.

## The Prototype

The flow chart on the following page represents the ACE workflow. The first step in the flowchart is to create the GUI design. Creating a GUI design sets standards for the GUI look and behavior. Next you should install ACE on your computer. After ACE is installed you create a prototype Application, create screen images and create Subapplications for this prototype.

The prototype Application should contain at least one of each type of screen layout found in the Application. Choose the representative screens for the prototype carefully. Carefully selecting these screens lifts the pace of production once the Application conversion takes place. This is because no further modifications to the KnowledgeBase will be necessary after the prototype is completed.

Tune the KnowledgeBase to recognize patterns in these representative screens. Generate the runtime and test the runtime for functionality. You should not be concerned with the GUI look in the prototype stages. Check that host patterns are being recognized by the KnowledgeBase and everything functions as designed. If controls or other items are not arranged properly at this time, fix them during the Application's conversion.

## The Final Application

After you test and debug the prototype Application start to create the final Application. The first steps are to create the Application, screen images and Subapplications.

Tuning the KnowledgeBase at this stage should not be necessary since it was done in the prototype Application. In creating the prototype you did not work on the GUI of the Subapplications but here you do.

Ensure that the GUI look of all Subapplications matches the design criteria laid out in the beginning of the project. Generate Runtime.

Test and configure the runtime environment to match that of your end users. After testing, create the runtime installation and install the ACE Application on the customer base.

## The Conversion Workflow

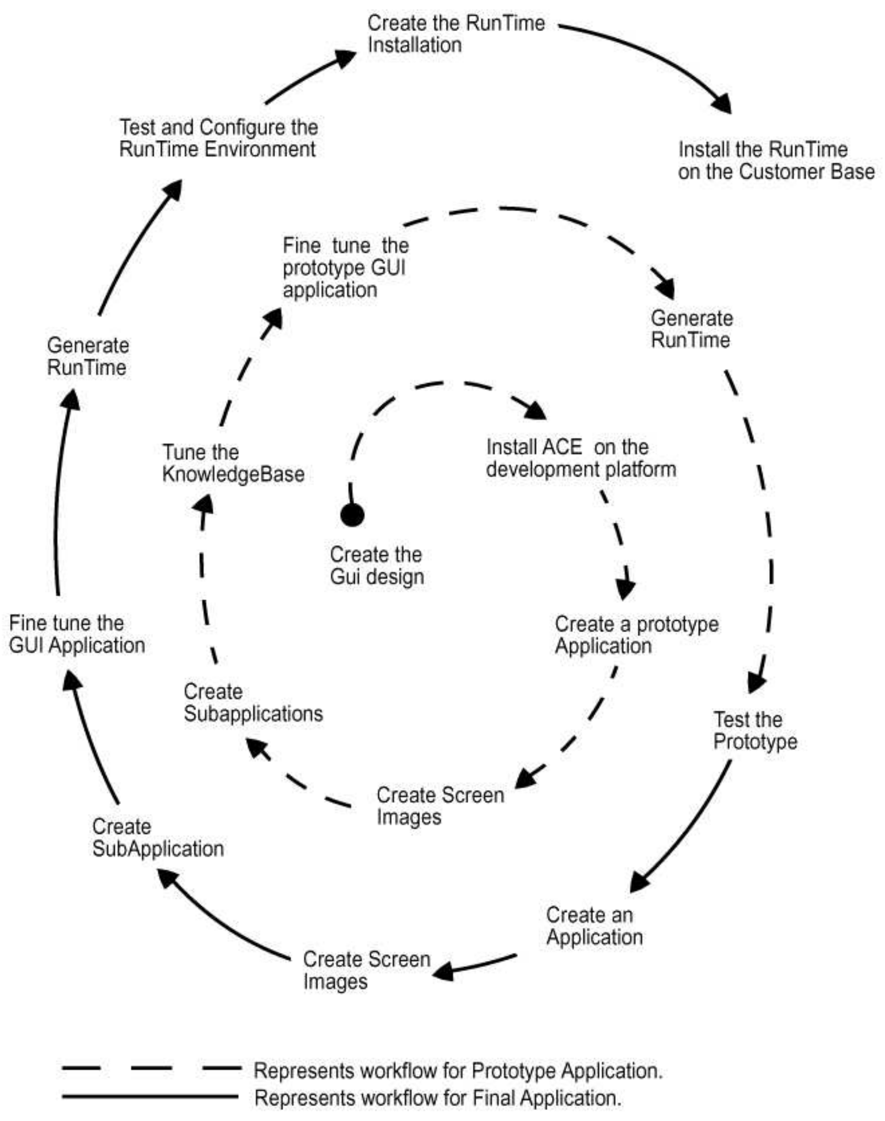

Create the RunTime Installation
Install the RunTime on the Customer Base
Test and Configure the RunTime Environment
Fine tune the prototype GUI application
Generate RunTime
Generate RunTime
Tune the KnowledgeBase
Install ACE on the development platform
Create the Gui design
Fine tune the GUI Application
Create a prototype Application
Create Subapplications
Test the Prototype
Create Screen Images
Create SubApplication
Create an Application
Create Screen Images
Represents workflow for Prototype Application.
Represents workflow for Final Application.

**Figure 17. The conversion workflow**

The GUI has the look and function of contemporary *Windows* applications, while maintaining the full functionality of the host application.

Wizards guide you in creating Applications, screen images, Subapplications and runtimes. The Application and library arrangement must be well thought out.

## Establishing an Application

Each time you start ACE, you must establish an Application to work in. Establish an Application either by creating a new Application or opening an existing one.

To create a new Application:

- From within ACE select *File* > *New* > *New Application*.
- The *Create New Application* wizard appears. Follow the instructions in the wizard.

To open an existing Application:

- From within ACE select *File* > *Open* > *Open Application*.
- Choose an Application from the *Open Application* selection box.

Once you have established your working Application, you can do any of the following things:

- Transfer data from the host to your development platform and create new screen images or maintain existing screen images.
- Edit existing screen images by combining a screen image with a screen capture.
- Create a new Subapplication from an existing screen image. New Subapplications are automatically converted and then presented for further fine tuning.
- Open an existing Subapplication. The Subapplication is presented for fine tuning.
- Create and run an executable composed of any existing Subapplications of the Application.

## Establishing the Environment

Before you can generate your GUI you must convert information about the host application into a form that ACE can read and then make the information available to ACE.

These two procedures are performed in different ways, and in different order, depending on the type of host and whether you are using screen definition files or screen captures.

The workflow for screen definition files on the iSeries is:

1. Compile DDS files into DDO files on the iSeries using the *JIS DDS Compiler*.
2. Transfer the DDO files to your development platform.

3. Create screen images from the DDO files using the *Create Screen Images* wizard.
4. Use the *Maintain Screen Images* wizard also to resume work on a DDO file.

The workflow for mainframe screen definition files is:

1. Transfer the BMS/MFS SDF source files to your development platform as text files.
2. Create screen images from the source files using the *Create Screen Images* wizard.
3. To resume work on a screen definition file use the *Maintain Screen Images* wizard.

The workflow for mainframe and iSeries screen captures is:

1. Establish the infrastructure for connecting to the host.
2. Use the *Create Screen Images* wizard to go online to the host, capture screens and create screen images.
3. To resume work on a screen capture file use the *Maintain Screen Images* wizard.

## Creating Screen Images

Before you begin you must install your product - the ACE - on your development platform. Each of the ACE suites includes ACE.

Within ACE:

1. *Create a new Application.* The new Application is the structure corresponding to the host application. The Create New Application wizard takes you through this process step by step.
   Whenever you work in ACE you must specify the Application you are working in by either creating a new Application—generally once per host application—or opening an existing Application. You may also choose to work within a library by either creating a library or opening one of the Application's existing libraries.
2. *Create screen images.* This involves either transferring host files to the PC and/or capturing the screens of the host application in a live connection. The information on what to do with host files, target directories on the development platform, and host connection parameters for screen captures is provided step-by-step in the Create Screen Images wizard.

When the Create Screen Images wizard finishes each image, it prompts your next action. You may:

- Process the screen image you have just created by creating a Subapplication.
- Return to ACE and create additional screen images—or perform other ACE activities.

## Processing the Screen Images

Creating a Subapplication for each host screen is the first step. A Subapplication forms the framework for the graphic equivalent of a host screen. Creating Subapplications is driven by the *New Subapplication* wizard. A newly created Subapplication automatically includes a fully functional GUI.

## Fine Tuning

The automatic GUI uses various rules, contained in the ACE KnowledgeBase, to perform the conversion. You may wish to override these rules by changing colors, the arrangement of controls, the choice of controls, adding controls that give the Application additional functionality and performing other modifications.

In certain situations you may need to change the screen images underlying your Subapplications. ACE allows you to make these changes while preserving as much as possible of your fine tuning.

- You may need to edit existing screen images, when you wish to combine the image with information from a screen capture. This situation often occurs when header information on the host screen is not static text but output fields.
- Editing screen images is performed in the *Edit Screen Images* wizard.
- You may need to maintain screen images to reflect changes in the host applications screens. Screen image maintenance is performed in the *Maintain Screen Images* wizard.

To fine tune your GUI:

1. Select a Subapplication from the *Subapplication* combo box.
2. Within the various views, modify the Subapplication and test your modifications. ACE provides several levels of help.

## Editing Screen Images Using the Screen Image Editor

The *Screen Image Editor* provides a tool for editing or creating pnl (screen image) files that are then used to combine screens or for file testing.

The *Screen Image Editor* is generally used to solve unique situations when encountering screen identification failures during runtime testing.

Runtime identification anomalies can also be solved by changes made to *.ini files. Changes made to the *.ini files are globally applied to all Subapplications in a given Application. Changes made using the *Screen Image Editor* are applied only to the edited pnl file and its Subapplication.

Examples for using the Screen Image Editor:

- Use the *Screen Image Editor* to create independent screen images and save under new names. *Example*: Create an empty screen for use as a principal Subapplication when working with Many-to-One.
- Change the controls in table columns. *Example:* Change an adjustable edit field to a combo box or a check box by changing the attributes.

The Screen Image Editor is used for the following activities:

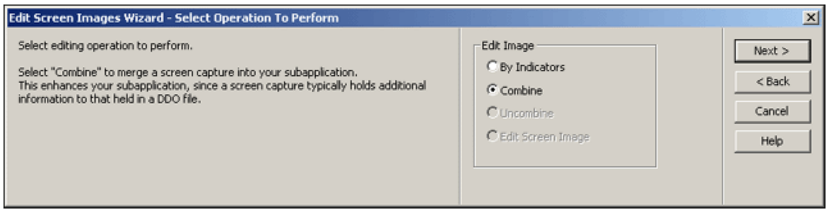
- Changing Text.
- Changing the foreground and/or the background color.
- Changing attributes for the attribute properties and/or the field type.
- Changing the default cursor location on the host screen.

## Accessing the Screen Image Editor

The **S***creen Image Editor* is used to edit pnl files. You can access the *Screen Image Editor* via the *Host* menu or via the *Utilities* menu.

### Via the Host Menu

The Screen Image Editor is accessed through the *Select Operations to Perform* step in the *Edit Screen Images* wizard through the *Host* menu.


Edit Screen Images Wizard - Select Operation To Perform

Select editing operation to perform.

Select "Combine" to merge a screen capture into your subapplication.
This enhances your subapplication, since a screen capture typically holds additional information to that held in a DDO file.

Edit Image: By Indicators [✓ Combine] Uncombine Edit Screen Image

**Figure 18. Accessing the screen image editor through the wizard**

The *Edit Screen Image* option is only available when the source is a captured screen. The screen must be open in ACE.

Pressing *Next* in the wizard step shown above opens the *Screen Image Editor*.

After editing, the pnl file is saved and the corrected image is automatically applied to the Subapplication.

### Via the Utility Menu

The *Screen Image Editor* is also available in the *Utility* menu.

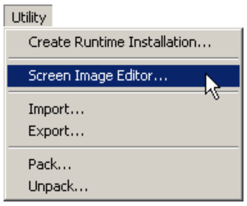
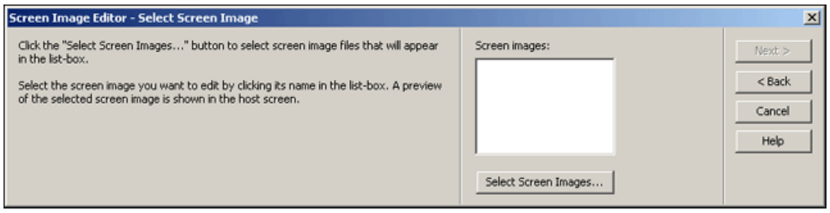


**Figure 19. Accessing the screen image editor through the Utility menu**

Here the *Screen Image Editor* can be applied to all pnl files regardless of their source. The *Select Screen Image* step in the *Screen Image Editor* allows selecting any pnl file appearing in the appropriate directories.

Since the editing is not carried out on screen images, the modifications are not applied to existing Subapplications.


Screen Image Editor - Select Screen Image

Click the "Select Screen Images..." button to select screen image files that will appear in the list-box.

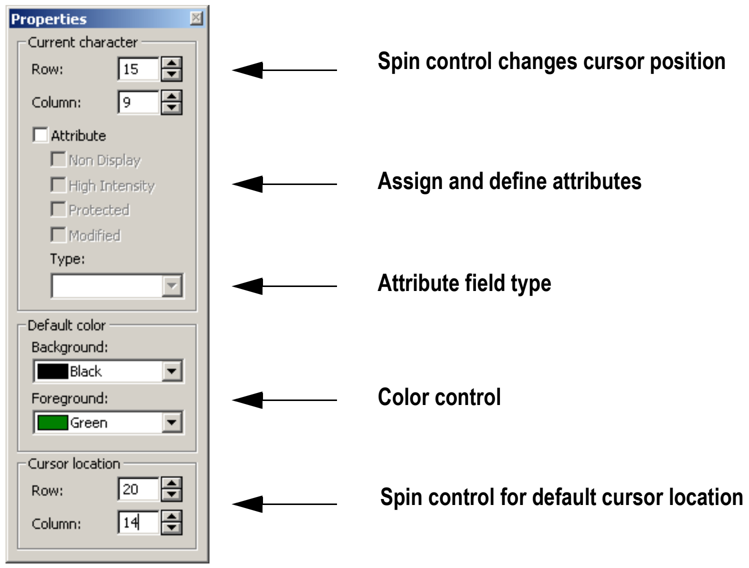
Select the screen image you want to edit by clicking its name in the list-box. A preview of the selected screen image is shown in the host screen.

**Figure 20. New Screen Image wizard - select a file and press Next**

In the wizard step shown above, select a file and press *Next* to open the *Screen Image Editor*.

## Using the Screen Image Editor

The *Screen Image Editor Properties* dialog box is automatically displayed after navigating to the *Edit Screen Image* step in the *Screen Image Editor* wizard.



Properties

Current character
Row: 15
Column: 9
Attribute
Non Display
High Intensity
Protected
Modified
Type:

Default color
Background: Black
Foreground: Green

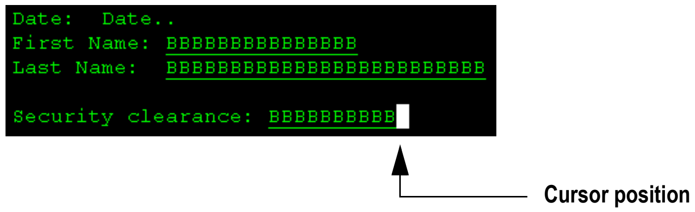
Cursor location
Row: 20
Column: 14

Spin control changes cursor position
Assign and define attributes
Attribute field type
Color control
Spin control for default cursor location

Figure 21. The screen image editor interface

### Moving the Cursor

When the *Screen Image Editor* is open, the cursor blinks in the Host screen. The editable place is at the cursor position.



```
Date:   Date..
First Name:  BBBBBBBBBBBBBBB
Last Name:   BBBBBBBBBBBBBBBBBBBBBBBBB

Security clearance:  BBBBBBBBBB
```

Figure 22. The editable area on the host screen

The cursor can be moved from its current position by:

- Pointing and left clicking on a new position.
- Using the *Current Character* spin control in the *Properties* dialog box.
- Using the keyboard.

### Keyboard functionality:

Table 7. Keys and their descriptions

| Key | Description |
|---|---|
| **Arrow keys** | Move the cursor right/left and up/down. When reaching the end of a line the cursor jumps to the beginning of the next line. When reaching the end of the screen the cursor jumps to the first position on the screen. |
| **End/Home** | End jumps the cursor to the end of the line (last column).<br><br>Home jumps the cursor to the beginning of the line (first column). |
| **PageUp/ PageDown** | The cursor jumps to the first or last line. |
| **Backspace** | Deletes the previous character. |
| **Tab** | Jump the cursor to the next editable field. |

### RMB Functionality

When right clicking on the host session, this popup is displayed:

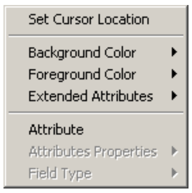


Figure 23. Right-click menu on the host session

Changes set using the RMB popup take effect at the current cursor position.

### Changing the Default Cursor Location

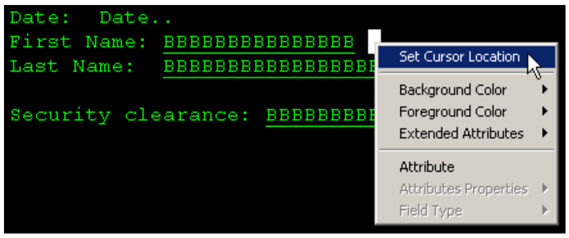
Sometimes the cursor is not present in popups captured in runtime. The cursor helps identify the popup border. In this case, it is necessary to set the default cursor location.

Using the *Properties* dialog box:

1. Select the new default location using the *Cursor Location* spin controls.
2. Press *Save* in the *Edit Screen Image* wizard step. The new location is visible only after exiting and re-entering the *Screen Image Editor*.

Using the RMB popup:

1. Move the cursor to the new position by using the arrow keys or by pointing and clicking.
2. With the pointer on the cursor, right click.
3. Click *Set Cursor Location* in the shortcut menu (see figure below). The *Cursor Location* spin control is updated and displays the new location.



```
Date:  Date..
First Name: BBBBBBBBBBBBBBB
Last Name:  BBBBBBBBBBBBBBBBB

Security clearance: BBBBBBBBB
```

**Figure 24. Set Cursor Location**

### Changing Colors

Changing colors is generally used for messaging. By default, messages appear in white on a yellow background.

Using the *Properties* dialog box:

1. Point and click or use the arrow keys to put the cursor on the target position.
2. Open the Background/Foreground color combo box and select the colors. The text typed at the cursor position is displayed in the selected colors.

Using the RMB popup:

1. Move the cursor to the target position by using the arrow keys or pointing and clicking.
2. With the pointer on the cursor, right click.
3. Click *Background Color* or *Foreground Color* in the shortcut menu. A list of color selections is displayed (see figure below).
4. Select the desired color. The new color appears on the host screen.

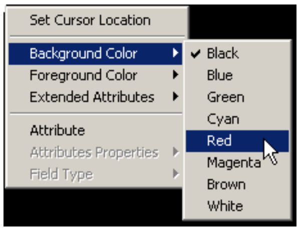


**Figure 25. Selecting the desired color**

**Note:** Color options are disabled when the cursor is positioned on an attribute.

Using LMB and frames:

1. Position the pointer at the desired position or next to the target text.
2. Left click and drag a frame around the area or text.
3. Right click in the frame. A shortcut menu containing *Paint Frame* is displayed.
4. Click *Paint Frame*. The *Paint Frame* dialog box is displayed:

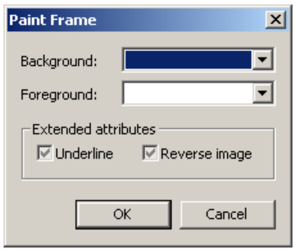


**Figure 26. The Paint Frame dialog box**

5. Change background/foreground colors using the color combo boxes. Extended Attributes are also available in this dialog box.

## Adding or Removing Extended Attributes

Extended Attributes include underline and reverse image of text characters.

1. Place the cursor on the character or on any position.
2. With the pointer on the cursor, right click.
3. From the shortcut menu, select *Extended Attributes*. A submenu opens.
4. Click *Underline* and/or *Reverse Image*. The underline appears on the host screen. Reverse Image is not visible.

Extended Attributes are also available in the *Paint Frame* dialog box. See the preceding section for more details.

**Note:** Extended Attributes are not available when the cursor is positioned on an attribute.

## Adding, Editing, and Deleting Attributes

An example of editing attributes is when the captured panel contains output, but input attributes may be necessary.

Using the *Properties* dialog box to add Attributes:

1. Move the cursor to the target position by using the arrow keys or pointing and clicking.
2. In the *Properties* dialog box click in the *Attribute* checkbox at the cursor position and select the Attribute property. The default is unprotected alphanumeric type.
3. Select the Attribute type from the *Type* combo box.

Using the RMB shortcut menu to add Attributes:

1. Move the cursor to the target position by using the arrow keys or pointing and clicking.
2. Put the pointer on the cursor and right click.
3. Click on *Attribute* in the shortcut menu. The Attribute item in the shortcut menu is checked and the checkbox in the *Properties* dialog box is checked. The default is alphanumeric type with no attribute properties selected.

Using the *Properties* dialog box to edit or delete Attributes:

1. Move the cursor and stand on the target Attribute.
2. To edit an attribute, click on the *Attribute* property or select the Attribute type from the *Type* combo box.

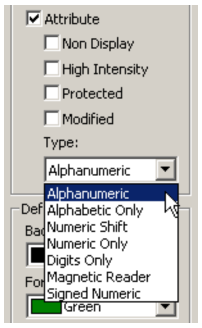


**Figure 27. Editing attributes**

3. To delete an Attribute, stand on the Attribute and uncheck the *Attribute* check box.

Using the RMB shortcut menu to edit Attributes or delete an Attribute:

1. Move the cursor and stand on the target Attribute.
2. To edit an Attribute, put the pointer on the cursor and right click. All attribute options are enabled. Standing on *Attribute Properties* or *Field Type* displays the options in cascaded menus. Click on the desired option.
3. To delete an Attribute, click on *Attribute*. The selected attribute is deleted from the host screen. The *Properties* dialog box is updated.

### Adding and Deleting Text

Text can be added by directly typing at the cursor position. Typing is always in overwrite mode. The text color and the background color can be altered. Adding text might be used to edit dynamic options or leaders on mainframe screens.

To delete text, place the cursor to the right of the character and press *Backspace*. Deleting text might be used to delete a message that appeared on the screen when it was captured during runtime.

## Processing Subapplications

Subapplications are processed from previously created screen images. Subapplication processing is wizard-driven. When completing all the steps in the *Subapplication* wizard the GUI is automatically generated.

Subapplications can be fine tuned in Design View for local modifications, and through the KnowledgeBase for global modifications.

To create a Subapplication:

- Select *New* from the *Subapplication* menu.
  -OR-
- Click on the *New Subapplication* icon.
  -OR-
- Click *Yes* when prompted to create a new Subapplication directly after completing the *Create Screen Image* wizard.

  The *New Subapplication* wizard opens. The wizard asks you for information as described in the next sections.

## Screen Types in the New Subapplication Wizard

The *New Subapplication* wizard classifies screens into three types:

- *Nothing*. A screen that is not a menu or a list.
- *Menu*. A screen that contains a set of options and a mechanism for choosing one of these options.
- *List*. Automatically recognized by the wizard.

For all screen types the wizard asks you for the following information:

**Table 8. Wizard steps for all screen types (Sheet 1 of 2)**

| Step | Description |
|---|---|
| **Select Screen Image Source Type** | There are three options. Choose the option appropriate to your screen image source:<br>• Captured<br>• DDS<br>• Other SDF |
| **Select Screen Image** | From the *Source File* list select the DDO file.<br><br>From the *Screen Image* list select the screen image.<br><br>If you like, you can select multiple source files and multiple screen images, to create multiple subapplications in a single execution of the New Subapplication wizard. The screen layout and window layout that you choose in the wizard is applied to all of the screens. See the section “Generating Multiple Subapplications Simultaneously” on page 60.<br><br>This “multiple screen image” processing does not handle menus or tables. Those screen images for which you want the generated subapplication to contain a menu or a table must be processed individually.<br><br>Note that in the Select Screen Image panel you can also create screen images. |

**Table 8. Wizard steps for all screen types (Sheet 2 of 2)**

| Step | Description |
|---|---|
| **Display Window as Popup** | When the *Popup Window* check box is cleared, the runtime behavior is that the resultant GUI replaces the GUI of the previous screen.<br><br>When the *Popup Window* check box is set, the runtime behavior is that the resultant GUI is displayed on top of the GUI of the previous screen. |
| **Specify Subapplication Properties** | *Regular***.** The GUI is a stand-alone window that is always displayed in response to a host write/read statement.<br><br>*Never Display Window, Conditionally Display Window.* There will either be no GUI or the GUI is sometimes not displayed. In these cases, you write ACE methods that act in place of the user. See the chapter entitled "Skipping Windows" in *webMethods JIS: Advanced Topics*.<br><br>*Dependent, Principal.* The GUIs from two or more host write/read statements are combined. See the chapter entitled "Many-to-One" in *webMethods JIS: Advanced Topics*. |
| **Select Screen Layout** | Select the screen layout appropriate for the screen image. |
| **Select Window Layout** | Assign a window layout to the Subapplication by selecting a window layout from the list. |
| **Add Menu or List to Subapplication** | From the *Add?* combo box choose:<br><br>*Nothing.* The screen contains various input/output fields, none of which explicitly control navigation on the host.<br><br>*Menu.* Configure menus. The menu option chosen generally controls host navigation. See below.<br><br>*List.* Configure lists. See below. |
| **Subapplication Description** | Optionally enter free text to describe or identify the Subapplication. |

## Menus in the New Subapplication Wizard

The menus in Table 9 reside in the New Subapplication Wizard.

**Table 9. Menus**

| Menu | Description |
|---|---|
| **Specify Menu Type**<br><br>**Menus only** | *Code Selection.* The user specifies the option chosen by entering a particular value into an edit field.<br><br>*Cursor Selection*. The user specifies the option chosen by placing the cursor on the text describing the option and pressing *Enter/Return*. |
| **In code selection menus - Mark Menu Selection Input Field** | Draw a rectangle around the text and option input field where users enter the menu selection value. |
| **In code selection menus - Mark Code Blocks** | Draw rectangles around the screen text that indicates the values that can be entered in the code selection input field. The rectangle should not include the text that describes the effect of each option value. The rectangle *should* include any terminator character used to separate the "value" text from the "description" text.<br><br>When the values cannot be covered by a single rectangle, the *Mark More Blocks* button allows you to draw additional rectangles. |
| **In cursor selection menus - Mark Menu Blocks** | Draw rectangles around the screen text that describes the option.<br><br>When the values cannot be covered by a single rectangle, the *Mark More Blocks* button allows you to draw additional rectangles. |

## Lists in the New Subapplication Wizard

When the screen image contains a list, the *New Subapplication* wizard automatically recognizes it and prompts you to identify some details:

**Table 10. List options (Sheet 1 of 2)**

| Option | Description |
|---|---|
| **Specify List Rolling Keys for Subapplication** | Specify the keys that page up and down through lists.<br><br>*Note*: The default was specified in the *New Application* wizard. |
| **Specify List Type** | The wizard chooses the list type automatically, but you can override the choice.<br><br>*Standard*. Each list record is contained in a single screen line and there is only one record per line.<br><br>*Folded*. List records extend over two or more screen lines<br><br>*Repeated Columns*. There are two or more list records per screen line. |
| **Mark the List Body Size** | This is done automatically by the wizard, but you can change the marking rectangle's dimensions. The rectangle should enclose the entire screen area occupied by list records. |
| **Specify if List Has a Selection Column** | Set the check box if the list contains a column of input fields where values indicating row operations can be entered. |
| **In folded lists Specify Number of Folded Lines** | This is done automatically by the wizard, but you can override the choice. The spin box displays the number of screen lines occupied by a single list record. |

**Table 10. List options (Sheet 2 of 2)**

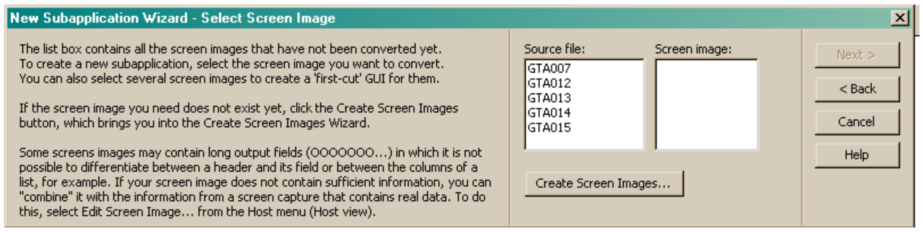
| Option | Description |
|---|---|
| **Mark the List Headers Are** | The header type is selected automatically by the wizard, but you can override the choice.<br><br>From the combo box choose a header type according to whether headers appear above the list, internally, or some combination of both. Note that not all possibilities are available for all list types.<br><br>Mark a rectangle that encloses only headers that appear above the list. Mark the rectangle to include the entire screen area occupied by these headers. |

**Note:** Subapplications can be renamed using the **Save As** option under the **Subapplication** menu.

## Generating Multiple Subapplications Simultaneously

In a single execution of the New Subapplication wizard, you can generate a single subapplication, or multiple subapplications simultaneously. This section describes the generation of multiple subapplications simultaneously.

**1** The wizard lists the source files that contain screen images that have not yet been converted to subapplications.



New Subapplication Wizard - Select Screen Image

The list box contains all the screen images that have not been converted yet.
To create a new subapplication, select the screen image you want to convert.
You can also select several screen images to create a 'first-cut' GUI for them.

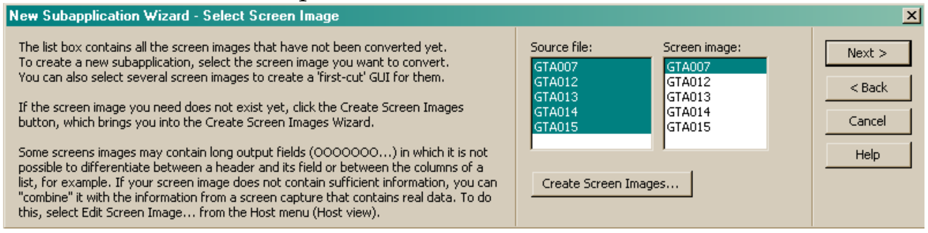
If the screen image you need does not exist yet, click the Create Screen Images button, which brings you into the Create Screen Images Wizard.

Some screens images may contain long output fields (OOOOOOO...) in which it is not possible to differentiate between a header and its field or between the columns of a list, for example. If your screen image does not contain sufficient information, you can "combine" it with the information from a screen capture that contains real data. To do this, select Edit Screen Image... from the Host menu (Host view).

Source file: GTA007, GTA012, GTA013, GTA014, GTA015

Screen image:

You can select multiple source files,



New Subapplication Wizard - Select Screen Image

The list box contains all the screen images that have not been converted yet.
To create a new subapplication, select the screen image you want to convert.
You can also select several screen images to create a 'first-cut' GUI for them.

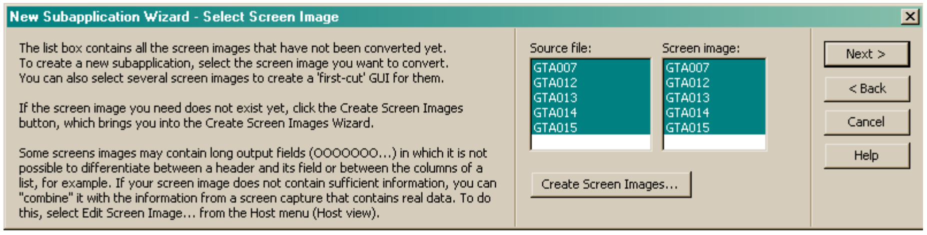
If the screen image you need does not exist yet, click the Create Screen Images button, which brings you into the Create Screen Images Wizard.

Some screens images may contain long output fields (OOOOOOO...) in which it is not possible to differentiate between a header and its field or between the columns of a list, for example. If your screen image does not contain sufficient information, you can "combine" it with the information from a screen capture that contains real data. To do this, select Edit Screen Image... from the Host menu (Host view).

Source file: GTA007, GTA012, GTA013, GTA014, GTA015 (all selected)

Screen image: GTA007 (selected), GTA012, GTA013, GTA014, GTA015

and then multiple screen images. Then click Next.



New Subapplication Wizard - Select Screen Image

The list box contains all the screen images that have not been converted yet.
To create a new subapplication, select the screen image you want to convert.
You can also select several screen images to create a 'first-cut' GUI for them.

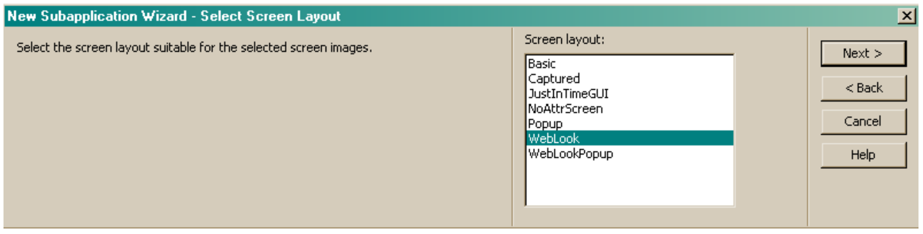
If the screen image you need does not exist yet, click the Create Screen Images button, which brings you into the Create Screen Images Wizard.

Some screens images may contain long output fields (OOOOOOO...) in which it is not possible to differentiate between a header and its field or between the columns of a list, for example. If your screen image does not contain sufficient information, you can "combine" it with the information from a screen capture that contains real data. To do this, select Edit Screen Image... from the Host menu (Host view).

Source file: GTA007, GTA012, GTA013, GTA014, GTA015 (all selected)

Screen image: GTA007, GTA012, GTA013, GTA014, GTA015 (all selected)

**2** Select the screen layout best suited for your particular host screen images . The layout selected is applied to all of the selected screen images.

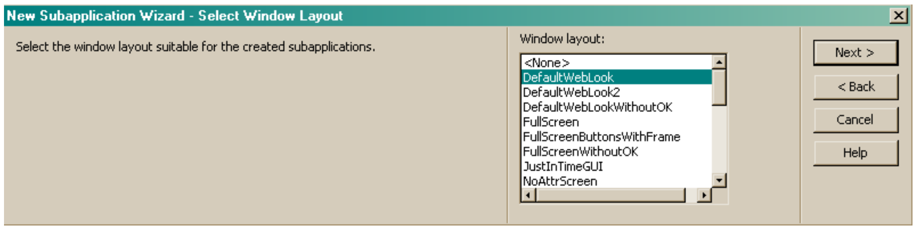


New Subapplication Wizard - Select Screen Layout

Select the screen layout suitable for the selected screen images.

Screen layout: Basic, Captured, JustInTimeGUI, NoAttrScreen, Popup, WebLook (selected), WebLookPopup

3 The wizard prompts you to choose a window layout. This defines the "look" of your subapplication screen. Here, too, the window layout you choose is applied to all of the new subapplications.

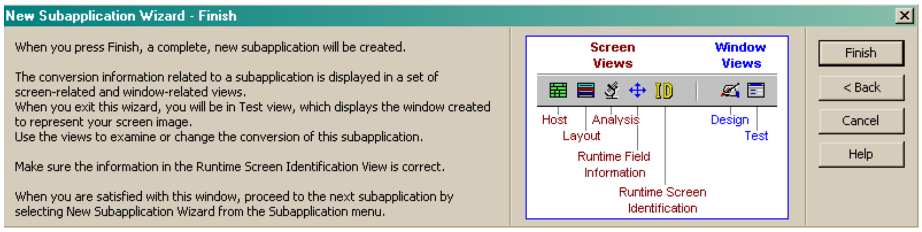

New Subapplication Wizard - Select Window Layout

Select the window layout suitable for the created subapplications.

Window layout:

```
<None>
DefaultWebLook
DefaultWebLook2
DefaultWebLookWithoutOK
FullScreen
FullScreenButtonsWithFrame
FullScreenWithoutOK
JustInTimeGUI
NoAttrScreen
```

4 Click Finish to complete the process.


New Subapplication Wizard - Finish

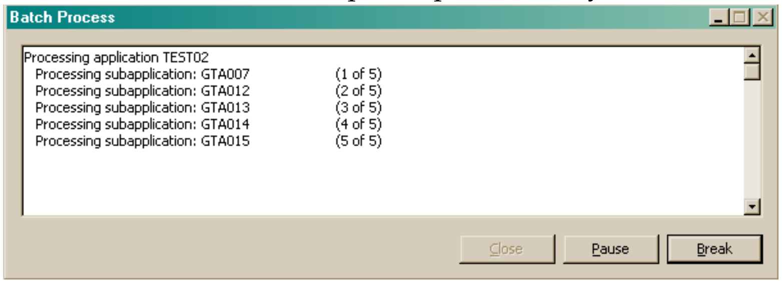
When you press Finish, a complete, new subapplication will be created.

The conversion information related to a subapplication is displayed in a set of screen-related and window-related views.
When you exit this wizard, you will be in Test view, which displays the window created to represent your screen image.
Use the views to examine or change the conversion of this subapplication.

Make sure the information in the Runtime Screen Identification View is correct.

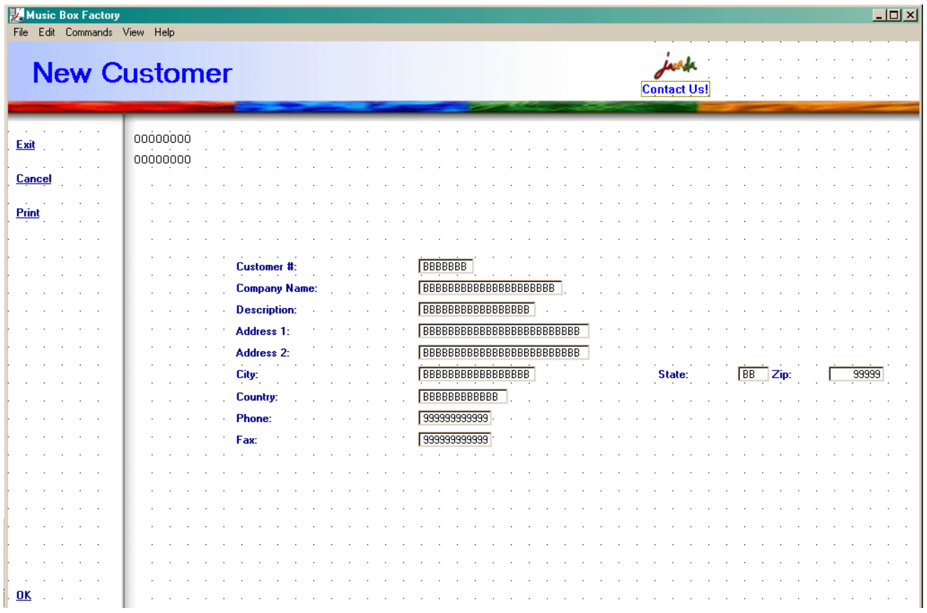
When you are satisfied with this window, proceed to the next subapplication by selecting New Subapplication Wizard from the Subapplication menu.

A small console window opens up to inform you of the wizard's progress.


Batch Process

```
Processing application TEST02
  Processing subapplication: GTA007          (1 of 5)
  Processing subapplication: GTA012          (2 of 5)
  Processing subapplication: GTA013          (3 of 5)
  Processing subapplication: GTA014          (4 of 5)
  Processing subapplication: GTA015          (5 of 5)
```

Here is an example of a generated subapplication window. Of course, you can use ACE to manually modify and customize the appearance of the generated subapplication window



New Customer

00000000
00000000

| | |
|---|---|
| Customer #: | BBBBBBB |
| Company Name: | BBBBBBBBBBBBBBBBBBBB |
| Description: | BBBBBBBBBBBBBBBBB |
| Address 1: | BBBBBBBBBBBBBBBBBBBBBBBBB |
| Address 2: | BBBBBBBBBBBBBBBBBBBBBBBBB |
| City: | BBBBBBBBBBBBBBBBB |
| State: | BB |
| Zip: | 99999 |
| Country: | BBBBBBBBBBBB |
| Phone: | 999999999999 |
| Fax: | 999999999999 |

## Logging for Multiple Subapplication Generation

As part of the process of generating multiple applications simultaneously, a log file is produced. The log file is named `NewSubAppls.log`, and is created in the ACE root directory.The log file includes information on any screens where lists have been identified, on screens that may be menu screens, and on screens that appear to be pop-up screens. These particular cases are noted in the log file because they require manual intervention. For example, you would want to delete the subapplications generated for screens with menus or lists, and recreate them individually with the New Subapplication wizard.

## Deleting Menu and List Sections

When assigning sections to lists or menus in the *New Subapplication wizard*, right clicking on a selected section opens a shortcut menu with a single entry - *Remove Section from Screen*. Click to delete the section.

**Note:** The menu or list sections can be deleted only within the step where they were created. Example: To delete the List Body section you must be in the wizard's *Mark the List Body Size* step.

## Deleting Subapplications

To delete a Subapplication:

1. Go to the *Subapplication* menu.
2. Choose *Delete*.
3. Double click a Subapplication
   OR
   Highlight a Subapplication and click *OK*.
4. Click *Yes* to confirm or *Cancel* to abort.

## Tuning the GUI in the Views

Fine tune the automatic GUI using the tools in the different ACE views. Following is a brief description of each view.

**Table 11. ACE views (Sheet 1 of 2)**

| View | Description |
|---|---|
| **Host View** | Edit host screen images. |
| **Layout View** | Create layouts by specifying which KnowledgeBase rules should be applied to different areas of the screen. New layouts are available when you process the next screen image into a Subapplication. |
| **Analysis View** | View how the ACE KnowledgeBase rules interpret the text on the screen image. Analysis View provides a convenient point for changing the KnowledgeBase rules themselves. In Analysis View you may also change how ACE understands the different parts of a host screen list**.** |
| **Runtime Field Information View** | Account for fields whose runtime location and dimension can vary. |
| **Runtime Screen Identification View** | Change, if necessary, the markings that uniquely identify the host screen during runtime. |

**Table 11. ACE views (Sheet 2 of 2)**

| View | Description |
|---|---|
| **Design View** | Manipulate the size, color and arrangement of GUI controls, and change the function performed when a control is triggered in various ways. Similarly, you can add controls that do not originate from host screen information and give them functionality. |
| **Test View** | Examine the size, color and arrangement of GUI controls as well as how the controls respond to being triggered. Check whether the control representations correspond to the host screen elements. Perform certain specialized design functions related to tables. |

The next sections describe each of these views in somewhat greater detail. Full details, including the operating instructions, are contained in *webMethods JIS: Basic User's Guide*.

## Host View

In Host View you see the original screen image. You can examine the contents of the screen image and verify that they are correct.

It is useful to open Host View when viewing the window in Design View, so that you can ensure that the window represents the screen correctly.

If the screen image is not correct, you can edit it. The editing options enable you to correct the way the SDF (Screen Definition Files) information is used to create the screen image. You can also use an option named "combine" to combine a screen image created from SDF with a screen capture. This is necessary when the SDF screen image contains many output fields, and it is difficult to differentiate between the various fields. For example, a list may be displayed as one block of outputs, in which it is not possible to differentiate between the columns. In a screen capture, however, the list contains real data. By combining the screen capture with the SDF screen image, the information in the resulting screen image is more complete.

## What You See in Host View

The elements in Table 12 can be seen in Host View.

**Table 12. Elements that can be seen in Host View**

| Element | Description |
|---|---|
| **Screen Image** | The original screen image. |
| **Attributes** | The screen image is displayed by default with attributes. The attributes help you understand the contents of the screen image, by defining the type of a field and delimiting its size. |
| **OOOOOOs, BBBBBBs, IIIIIIIIIs** | In screen images created from some form of Screen Definition File (such as DDS for iSeries and BMS or MFS for mainframe), these characters represent different field types. |
| **OOOOOOs** | Output |
| **IIIIIIIIIIIIIIIs** | Input |
| **BBBBBBBs** | Both (Input and Output). |

**Note:** BBBBBBs are more commonly used than IIIIIIIIIs.

## What You Can Do in Host View

In Host View, you can:

- Edit an incorrect screen image. Select *Edit Screen Images* from the *Host* menu.
- Display or hide attributes in screen views. It is recommended to work with attributes displayed.

  To toggle the display of attributes, from the *Options* menu within any screen view, select *Host Screen Options*.

## Layout View

ACE decides how to represent different patterns in your screen by applying KnowledgeBase definitions to them. According to the information you provided in the *New Subapplication* wizard, the screen is divided into the most appropriate sections. Each section contains a set of KnowledgeBase definitions suitable for the part of the screen it covers. Using these definitions, the patterns are automatically recognized and given the most appropriate GUI representation.

In Layout View you can override ACE's choice of sections by applying a section of your choosing to any part of the host screen.

- Some sections are designed to be applied to a specific host screen pattern.

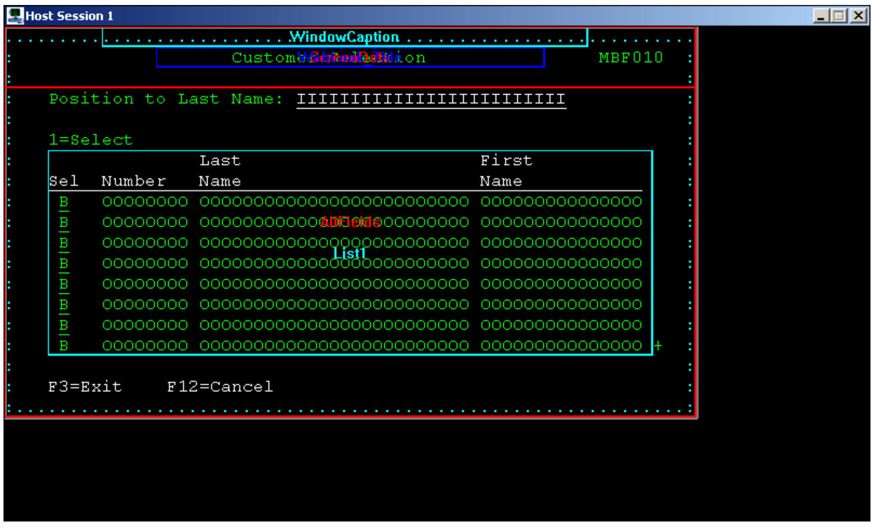
  For example, when an input field accepts only Y or N as input, you can change its control from an edit box into a check box by applying the *CheckBoxYorN* section to the screen pattern.
- Some sections are designed to be applied to an entire area of the screen.

  For example, the display of functions keys in your screen may be dynamic—different each time the screen appears. To handle such a case, apply the *DynamicFKeys* section to the entire area in which function keys may appear. Make sure that the version of the screen you have contains at least one function key inside the area covered by the *DynamicFKeys* section.

In Layout View you can also define new sections and group them together into layouts. These layouts are then available to you when you turn other screen images into Subapplications. Thus, when you have many similar screens, you can avoid fine tuning each screen's automatic GUI individually. Instead, you create a layout with all the fine tuning built in and then apply this layout to the appropriate screens in the *New Subapplication* wizard.

## What You See in Layout View

Layout View displays the host screen with the sections marked as colored rectangles.



```
                  Custom[illegible]on                    MBF010

  Position to Last Name: IIIIIIIIIIIIIIIIIIIIIIIII

  1=Select
                Last                          First
  Sel   Number  Name                          Name
   B   00000000 0000000000000000000000000 000000000000000
   B   00000000 0000000000000000000000000 000000000000000
   B   00000000 0000000000000000000000000 000000000000000
   B   00000000 0000000000000000000000000 000000000000000
   B   00000000 0000000000000000000000000 000000000000000
   B   00000000 0000000000000000000000000 000000000000000
   B   00000000 0000000000000000000000000 000000000000000
   B   00000000 0000000000000000000000000 000000000000000 +

  F3=Exit     F12=Cancel
```

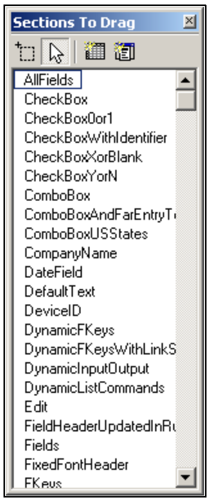
Figure 28.  Layout View

The colors have no significance other than making each section visually distinct. The name of the section appears in the center of the rectangle.

Sections may overlap on the screen or be entirely contained in sections applied to larger areas. Clicking within the border of a section selects it for Layout View operations.

## The Sections Panel

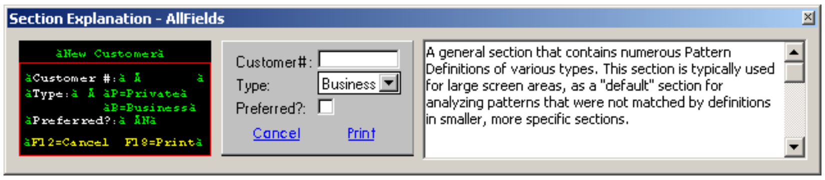
The *Sections* panel lists the available sections, and is your tool for applying them to the screen.


Sections To Drag

AllFields
CheckBox
CheckBox0or1
CheckBoxWithIdentifier
CheckBoxXorBlank
CheckBoxYorN
ComboBox
ComboBoxAndFarEntryT
ComboBoxUSStates
CompanyName
DateField
DefaultText
DeviceID
DynamicFKeys
DynamicFKeysWithLinkS
DynamicInputOutput
DynamicListCommands
Edit
FieldHeaderUpdatedInRu
Fields
FixedFontHeader
FKeys

**Figure 29. Sections panel**

The *Section Explanation* panel displays an explanation of the currently selected *Sections* panel section.


Section Explanation - AllFields

```
     àNew Customerà
àCustomer #:à Ã          à
àType:à  Ã  àP=Privateà
            àB=Businessà
àPreferred?:à  ÃNà
àF12=Cancel   F18=Printà
```

Customer#:
Type: Business
Preferred?:
Cancel Print

A general section that contains numerous Pattern Definitions of various types. This section is typically used for large screen areas, as a "default" section for analyzing patterns that were not matched by definitions in smaller, more specific sections.

**Figure 30. Sections Explanations panel**

The information includes a textual description and two pictures. The left hand picture shows a host screen pattern that would be recognized by the selected section. The picture indicates how you would place the section on such a host screen, including whether or not attributes should be included.

The right hand picture displays the GUI controls that would result from applying the selected section as in the left hand picture.

It is highly recommended that you browse through all the sections and view their explanations in order to have an idea of what options are available.

## What You Can Do in Layout View

In Layout View you can:

- Add, delete, move and resize sections on the screen image.

- Create new sections and decide which KnowledgeBase Pattern Definitions (rules) they contain.
- Save the arrangement of sections as a Layout.

A detailed description of the KnowledgeBase Pattern Definitions, what it means to put a Pattern Definition into a section, what it means to apply a section to an area of the screen, as well as the detailed operating instructions for all these tasks are found in *webMethods JIS: Basic User's Guide*.

## Analysis View

In Analysis View you see how the KnowledgeBase understood the host screen. A group of host screen characters that were recognized by a KnowledgeBase rule receive a color overlay. Thus, Analysis View provides a convenient entry point for modifying the KnowledgeBase rules, since the effect of your modifications are immediately visible as changes in the color overlays.

Analysis View also provides a visual tool for refinement of list analysis. In Analysis View you can manipulate the identification of host screen list headers, list columns and header-column links.

### What You See in Analysis View

In Analysis View, you see:

- If your screen image contains a list, you see yellow lines representing links between list headers and list columns.
- On the screen image, you see groups of host screen characters colored according to the KnowledgeBase rule that recognized them.
- The *Pattern Definition Properties* panel which displays information about the analyzed screen patterns.

To view which KnowledgeBase rule recognized a colored group of host screen characters, place the mouse cursor on top of the colored area. The *Pattern Definition Properties* panel displays the name of the KnowledgeBase rule that recognized the characters within the colored area, as well as the location and dimension of the area.

### What You Can Do in Analysis View

In Analysis View you can:

- Refine the analysis of lists.
- Enter the KnowledgeBase at a specific point.

### Refining List Analysis

You refine ACE's analysis of host lists in Analysis View. When ACE recognizes a host list, it marks part of the screen as being the body of the list and (optionally) part of the screen as being the list headers.

- ACE divides the list body area into individual list columns.
- ACE divides the list header area into individual column headers.
- ACE links each header to a column.

In Analysis View you may override each of these three aspects of ACE's analysis. Explanations are given in the *webMethods JIS: Basic User's Guide*.

### Entering the KnowledgeBase From Analysis View

You can enter the KnowledgeBase from any view, but it is convenient to do so from Analysis View. In Analysis View you can double click a character on the screen image, and the KnowledgeBase opens to the Pattern Definition that recognized that character.

## Runtime Field Information View

The Runtime Field Information View is where ACE handles the possibility that the runtime screen does not exactly match the screen image. Specifically, in Analysis View, when a group of host screen characters is recognized by a KnowledgeBase Pattern Definition, the character group's location on the screen and the dimension of the character group are recorded. This information is used during runtime as part of the identification process.

During runtime, the location and dimension of a character group may not match what is on the screen image. For instance, if the character group contains an output numeric field, the dimension of the group may vary according to the number of digits in the numeric field. ACE automatically marks character groups of variable dimension and location. In Runtime Field Information View you can change whether or not a character group is considered variable and the dimension and location parameters of the variable character groups.

## What You See in Runtime Field Information View

The screen image with character groups overlaid with red, light blue, dark blue or without overlay:

- Character groups displayed without overlay are definitely of fixed dimension and location. You cannot designate them as being variable.

- Character groups with red or light blue overlays have been identified as of fixed dimension, but you can designate them as variable. You can also designate them as definitely fixed.
- Character groups with dark blue overlays have been identified as variable, but you can designate them as fixed. You can also designate them as definitely fixed.

### The Decomposition Definition Properties Dialog Box

The *Decomposition Definition Properties* dialog box displays information on which KnowledgeBase pattern recognized a character group, the group's location and dimension and the tolerances for each, whether or not the character group is fixed, and tools for modifying these parameters.

### What You Can Do in Runtime Field Information View

In Runtime Field Information View you can change the parameters of any character group that has a colored overlay.

You can:

- Remove the overlay and designate the group as definitely fixed.
- Change the group's designation from fixed to variable and from variable to fixed.
- Change the group's location and dimension tolerances.

Full operating instructions are contained in *webMethods JIS: Basic User's Guide*.

## Runtime Screen Identification View

In Runtime Screen Identification View, ACE provides the information necessary to uniquely identify a host screen in runtime. When a screen is misidentified, you can correct the identification markings in this view.

### What You See in Runtime Screen Identification View

You see the screen image with different regions colored blue, red, yellow or green. These colors reflect the information necessary for the runtime screen identification process.

The meaning of each color:

- Areas that do not change in runtime are referred to as Fixed and are colored blue. Header text is usually colored blue.
- Areas that do change in runtime are referred to as Variable and are colored red. Input fields are colored red.
- The area in which messages appear during runtime is colored yellow.
- An area marked as a fingerprint is colored green. A fingerprint region must contain only fixed characters. A fingerprint should be unique to that screen—no other screen's fingerprint region should have the same combination of dimension, location and contained characters.

The identification process is based on the Fixed areas in the host screen. When an area that is marked in blue changes in runtime the identification fails.

**Note:** Therefore, it is very important that no variable characters be colored blue.

If you are not sure whether a certain area of the screen changes in runtime - mark it in red.

Note that all attributes are colored red because they may change in runtime, depending on the screen's contents or the host connection type.

**Note:** A screen does not receive a fingerprint automatically and you are not required to supply one.

## The Runtime Behavior Toolbox

The *Runtime Behavior* toolbox contains four buttons for overriding the automatic identification markings generated by ACE.

These are the options:

**Table 13. Buttons in the Runtime Behavior Toolbox (Sheet 1 of 2)**

| Icon | Description |
|---|---|
| 🟦 | Fixed (blue) |
| 🟥 | Variable (red) |

**Table 13. Buttons in the Runtime Behavior Toolbox (Sheet 2 of 2)**

| Icon | Description |
|---|---|
|  | Messages (yellow) |
|  | Fingerprint (green) |

## What You Can Do in Runtime Screen Identification View

When working with the runtime, occasionally a screen might not be recognized. When this happens, a default mechanism, named "Just-in-Time GUI", provides a GUI presentation for the screen. If this mechanism is turned off, an unrecognized screen will "bleed-through". This means that the screen will be displayed in its original host appearance.

A screen may be entirely new and bleed-through occurs simply because there is no Subapplication for the screen. In this case create a new Subapplication for the screen. Otherwise you need to correct the runtime identification.

To customize the identification information:

1. Select the desired character type, Fixed, Variable, Message Line, or Fingerprint from the toolbox.
2. Holding down the left mouse button, drag it across the area in the screen you wish to color differently. A frame indicates the marked area. When you release the mouse button, the framed area is colored according to the type you selected.

To cancel all your markings, from the *Runtime Screen Identification* menu select *Restart Screen Identification*. You can also cancel an incorrect color marking by coloring over it.

**Note:** If your Subapplication is a Host Popup, and the screen image displays the popup in the same location it appears in runtime, make sure you mark the entire area that is external to the popup as Variable (red), to enable the popup to be identified at runtime.

To determine why a screen was misidentified:

1. Capture the misidentified screen in runtime—from the *Application* menu, select *Emulator* > *Save Host Screen Image*. The *Save Panel As* dialog box opens.
2. Provide a name for the host screen capture and click *OK*.

3 In ACE, open the Subapplication of the misidentified screen in Runtime Screen Identification View.

4 From the *Runtime Screen Identification* menu select *Compare to Captured Screen Image*.

5 In the *Select Next Screen* dialog box select the *.PNL file that you saved in Step 2 and click *OK*. The inconsistencies between the screen image and the captured screen are color coded pink.

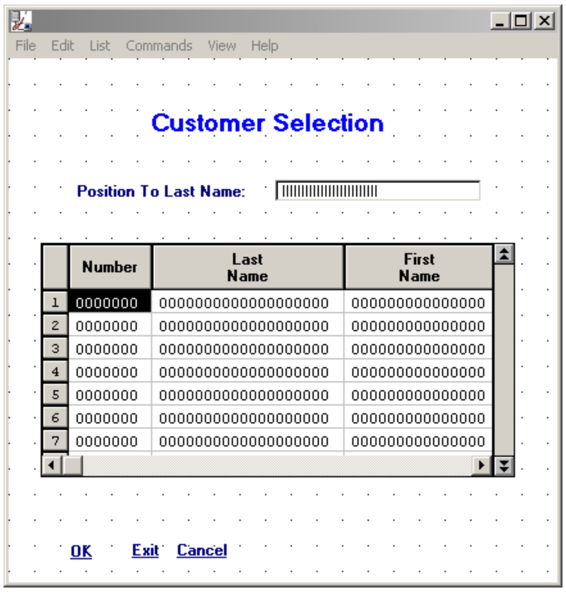
6 Use the pink areas to determine the reason why the screen was misidentified in runtime:

- If the host screen has been changed, select the *Maintain Screen Images* wizard from the *File* menu.
- If the screen image is incorrect and needs to be edited, from the *Utility* menu, select the *Screen Image Editor*.
- Otherwise, the Runtime Screen Identification information is not marked correctly. Correct the identification by marking the pink areas as Variable (red).

## Design View

In Design View you change the appearance of the window to suit your standards. You can arrange the controls on the window, modify control properties, delete controls and add controls to the window.

You also create and modify the methods that give functionality to these controls. Controls added in Design View have no functionality until you link them to a method.



| | Number | Last Name | First Name |
|---|---|---|---|
| 1 | 0000000 | 00000000000000000000 | 000000000000000 |
| 2 | 0000000 | 00000000000000000000 | 000000000000000 |
| 3 | 0000000 | 00000000000000000000 | 000000000000000 |
| 4 | 0000000 | 00000000000000000000 | 000000000000000 |
| 5 | 0000000 | 00000000000000000000 | 000000000000000 |
| 6 | 0000000 | 00000000000000000000 | 000000000000000 |
| 7 | 0000000 | 00000000000000000000 | 000000000000000 |

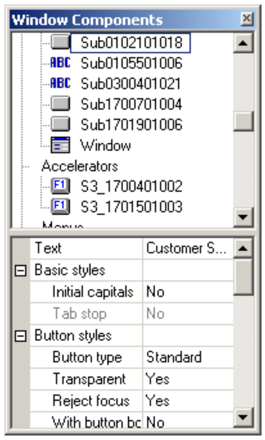
**Figure 31. Design View**

## What You See in Design View

Design View displays the current state of the GUI in a window. By default the GUI is displayed with a dot grid. To turn off the grid, from the *View* menu select *Customize* > *Show Grid*.

Design View is an extensive window editor that enables you to perform various functions on one or more controls.

You also see the *Window Components* dialog box and the *Definitions Palette*.



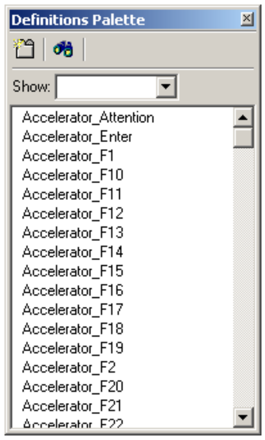
Window Components

Sub0102101018
Sub0105501006
Sub0300401021
Sub1700701004
Sub1701901006
Window
Accelerators
S3_1700401002
S3_1701501003

| | |
|---|---|
| Text | Customer S... |
| Basic styles | |
| Initial capitals | No |
| Tab stop | No |
| Button styles | |
| Button type | Standard |
| Transparent | Yes |
| Reject focus | Yes |
| With button bc | No |

Figure 32. Window Components dialog box

The *Window Components* dialog box contains a complete list of the current window's components. In addition, it displays information on the leading components currently selected: the type of the component; the name of the Pattern Definition that created it; the text displayed on the component; the component's dimensions and its position on the window.



Definitions Palette

Show:

Accelerator_Attention
Accelerator_Enter
Accelerator_F1
Accelerator_F10
Accelerator_F11
Accelerator_F12
Accelerator_F13
Accelerator_F14
Accelerator_F15
Accelerator_F16
Accelerator_F17
Accelerator_F18
Accelerator_F19
Accelerator_F2
Accelerator_F20
Accelerator_F21

Figure 33. Definitions palette

The *Representation Palette* contains a list of available Representation Definitions. It enables adding user controls to the window. These are additional controls that are not derived from the host screen. The *Add Tabs* button invokes the *Create Tab* wizard for creating tabs in the GUI window.

### What You Can Do in Design View

In Design View you can arrange the controls on the window, modify control properties, delete controls and add controls to the window. You can create new methods, modify existing methods, and link methods to controls.

## Test View

In Test View you see the way the window and all of its components will look in runtime. You can also see a description of the functionality of items such as push buttons and menu items when the buttons are clicked. The purpose of Test View is to make sure you are satisfied with the window.

To verify that the window represents the screen image in the most suitable way, it is recommended to compare the window shown in Test View to the screen image displayed in Host View. ACE enables you to do so by simultaneously activating a screen-related view together with a window-related view. Activate the Host View and compare all the screen image components to the GUI controls that represent them. This will help you determine whether you need to make any changes to the Subapplication.

### What You See in Test View

Test View displays the GUI as it will look in runtime.

You can see, for example:

- The values in a combo box.
- The values on tabs.
- A description of the functionality attached to controls.

### What You Can Do in Test View

In Test View you can:

- View attached functionality.

- Click a push button, click a menu item, or double click a table to display information on the runtime functionality of the control.
- Change Table Column order.
- Change Table Column width.

To change Table Column order:

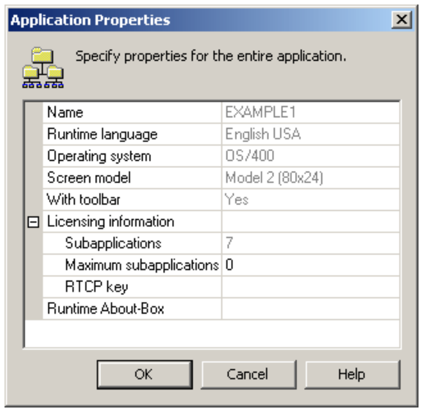
1. Select the column you wish to move by clicking on its heading button.
2. Hold down the *SHIFT* key and drag the heading button of the selected column to its new position.

To change Table Column width:

1. Place the cursor over the vertical line separating the heading button of the column you want to modify from the heading button to its right. The cursor becomes a two headed arrow.
2. Hold down the left mouse button and drag the line to a new location.

## Application Properties

After you have selected an Application you can view the Application properties in the *File* menu. From the *File* menu, select *Application Properties*.



Specify properties for the entire application.

| | |
|---|---|
| Name | EXAMPLE1 |
| Runtime language | English USA |
| Operating system | OS/400 |
| Screen model | Model 2 (80x24) |
| With toolbar | Yes |
| ⊟ Licensing information | |
| Subapplications | 7 |
| Maximum subapplications | 0 |
| RTCP key | |
| Runtime About-Box | |

**Figure 34. Application Properties dialog box**

The *Application Properties* dialog box displays the properties which you specified when you opened the Application.

You also have an opportunity to edit your Application's “About” screen and this is also where you will enter your license key.

**Note:** You do not have to enter this information when you begin developing an Application. This information can be entered after you have created a group of Subapplications.

## Runtime About Box

Windows programs frequently display information such as the product name, version number and copyright date. The Windows convention for displaying this information is in an About box that is accessed from the Application's *Help* menu. *About* is usually the last option in the *Help* menu.

ACE enables you to edit the contents that will appear in the runtime Application's *About* box. In the *Application Properties* dialog box, click *Runtime About-Box* to call up a dialog box displaying the way the *About* box will appear in your runtime Application. The fields with the white backgrounds are fully editable.

## Creating a Runtime

In the Application's *Generate Runtime* wizard you can choose to compile:

- Any or all of the Subapplications in the Application level.
- Any or all of the Subapplications in the Application level and all the Subapplications in the libraries
- Any or all of the Subapplications in the Application level and a subset of the Subapplications in the libraries.

In order to compile Subapplications in the Application level and a subset of the Subapplications in the libraries you must first perform a partial compilation on each of the relevant libraries. Next, in the Application's *Generate Runtime* wizard, include the library but do not recompile it. If you choose to recompile the library at this stage, all the Subapplications in the library will be included in the Application's runtime.

- To compile a library, in particular in order to select a subset of its Subapplications, open that library in ACE and perform a Generate Runtime.
- To compile an Application, with or without its libraries, open the Application in ACE and perform a Generate Runtime.

To generate runtime:

1. In ACE go to the appropriate level, Application or library, as explained above.
2. From the *File* menu choose *Generate Runtime*. The *Generate Runtime* wizard appears.

3 Progress through the wizard configuring the runtime as needed. See the following sections for more details.

4 Click *Finish*. The ACE Compiler starts.

**Note:** When the Compiler finishes the compilation, click *Done*. If the Application contains uncompiled libraries, the *Select Subapplication to Process* step is skipped and the *All* option is chosen by default

## The Application Generate Runtime Wizard

In the *Generate Runtime* wizard you make the following choices:

**Table 14. Options in the Generate Runtime wizard**

| Option | Description |
|---|---|
| **Runtime Type** | You can choose to create Java classes, or XHTML forms. |
| **Libraries** | You choose which libraries to include in the runtime and which libraries to compile. See “Generating Runtime with Libraries” on page 81 for more details. |
| **Subapplications** | You may include all of the Subapplications or any selection of the Application/library’s Subapplications. You may also create and save selection groups for future use. See “Generating Runtime Using Subsets” on page 79 for more details. |

## Generating Runtime Using Subsets

You can create an executable from a subset of all the Subapplications in an Application. This is very useful for testing a specific group of Subapplications in runtime.

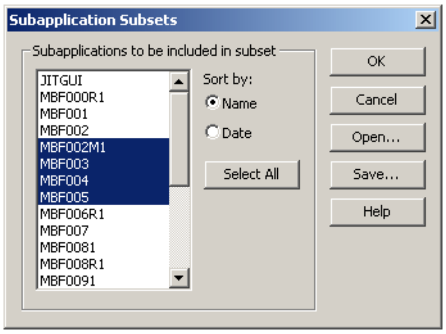
Generating runtime using subsets is a time-saving feature. If you are creating a runtime version for a subset of Subapplications, where some of them have been modified, you can save time by performing the Generate Runtime only on those that were modified.

## Selecting the Subapplications

To select Subapplications:

1 In the *Subapplications to Include* step of the *Generate Runtime* wizard choose *Subset* and click the *Select* button.

The *Subapplication Subsets* dialog box is displayed:



JITGUI
MBF000R1
MBF001
MBF002
MBF002M1
MBF003
MBF004
MBF005
MBF006R1
MBF007
MBF0081
MBF008R1
MBF0091

**Figure 35. Subapplication Subsets dialog box**

2 Make your selection.

- Select a single Subapplication by clicking its name in the list.

- Select several Subapplications by holding down the *Ctrl* key and clicking each in turn. Clicking a selected Subapplication deselects it.

- Select a sequence of Subapplications by holding down the *Shift* key and clicking the first and last Subapplication in the sequence. Choose a *Sort by* option to change the sequence order.

- Click *Select All* to choose all the Subapplications. The button text changes to *UnSelect All*.

- Select a set of Subapplications previously saved as a group. Click the *Open* button. A list of groups previously saved for this Application is displayed. This may be a group previously saved in the *Subapplication Subsets* dialog box or a group saved by a Query operation in ACE, as described in the *Query* section in *webMethods JIS: Basic User's Guide*. Click the name of a group to select it and then click *Ok*.

Focus returns to the *Subapplication Subsets* dialog box where you may further refine the selection in the group by selecting and unselecting Subapplications.

3 To save the selection as a subset, click *Save*.

The *Save Subset As* dialog box opens:

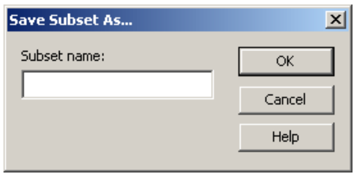


**Figure 36. Save Subset As dialog box**

4 In the edit field enter a name for the saved group and click *OK*.
5 Click *OK*.

### Partial Subsets in the Converter INI File

The parameters defined in the *Generate Runtime* wizard are recorded in two groups in the `SPECIFIC.INI` file: [MakeExe Parms] and [Partial MakeExe Groups]. If you wish, you can change the parameters of your partial group by editing the entries in this group of this INI file.

[MakeExe Parms] The first time you generate a partial runtime your system creates this group title in your `SPECIFIC.INI` file.

[Partial MakeExe Groups] When you save a partial Generate Runtime subset, your system will create this group title in your generate a partial runtime file.

## Generating Runtime with Libraries

You can generate a runtime from the Application level that includes all, some, or none of the libraries defined under the Application. This allows you to save compile time when you wish to test changes in a small number of libraries and to create Applications with different capabilities for different customers.

### Selecting the Libraries

In the *Select Libraries to Include in Runtime* step of the *Generate Runtime* wizard, select the libraries to be compiled from the *Libraries to include* list. This step only appears in Applications that have at least one library.

The *Libraries to include* list is a multi-selection tool:

- To select more than one library hold down the *Ctrl* key while selecting.
- To select a range of libraries, hold down the *Shift* key while selecting the last library in the range.
- Click the *Select All* button to select all the libraries.
- Click the *Unselect All* button to clear the entire selection.

### Setting the Library List

You can limit which libraries are available in runtime by editing the *Runtime library list*. Generally, you will want the libraries you have compiled in the Application to actually be available in runtime. The *Overwrite Library List* step contains two options; *Overwrite* and *Leave as is*. Note that this step does not appear if the set of libraries you chose to compile is the same as the already existing library list, or if this is the first time you are compiling the Application.

Note also that this step appears if the order in the library list is different from the order in the *Libraries to include* list of the previous step.

### Overwrite

Select the *Overwrite* radio button if you want all of the compiled libraries to be available in runtime in the order that they appear in the *Libraries to include* list of the previous step.

### Leave As Is

The order of the libraries in the library list may not serve your purposes in terms of the priority of different versions of the same Subapplication located in different libraries. You can edit the library list to change this order. In that case, you do not want to overwrite the library list, but simply leave it as is.

For details on editing the library list see *webMethods JIS: Advanced Topics*.

**Note:** If you want to include a partial library in an Application's runtime, you must first compile the library on its own, and then include it in an Application level compilation.

## Selecting Libraries to Recompile

In the *Select Libraries to Recompile* step you choose which libraries need to be recompiled and which you wish to incorporate as they were previously compiled.

- Click the *None* radio button to incorporate all libraries as previously compiled.
- Click the *Selected Libraries* button to choose libraries to recompile.

  The list is a multi-selection tool:

  - To select more than one library hold down the *Ctrl* key while selecting.

  - To select a range of libraries, hold down the *Shift* key while selecting the last library in the range.

  - Click the *Select All* button to select all the libraries.

  - Click the *Unselect All* button to clear the entire selection.

Any library that has never been compiled is automatically compiled, even if you choose *None*. Such a library does not appear in the list box.

## The Runtime Files

When you create a runtime installation with libraries, the runtime directory includes:

- A *.GSP file for each library, as well as the general *.GSP file.
- The *.A files.

ACE places these files in the Application's directory during compilation.

- Your ACE Application and its Subapplications and libraries are not by themselves an executable program. You make your Application into a program by creating the executable file. Remember that creating an executable is performed in the *Generate Runtime* wizard.
- After you have created your executable you need to set the executable's runtime behavior. You invoke the *Run Application* wizard to enter the runtime environment—where you set your Application's runtime behavior.
- The runtime environment you enter via the *Run Application* wizard can be either a simulation or a true online session connected to the host. You can use an online session to test the executable's online behavior.
- Once you have created an executable program and you are satisfied with its performance you can package it for distribution to end users by creating an installation. You are encouraged to test the installation on a PC that is separate from your development platform.

Packaging the executable for distribution is performed in the *Create Runtime Installation* wizard.

## Adding Last Minute Files

The ACE installation allows you to replace or add last minute files to your installation, without having to remake the installation. You add this file to the last diskette of your installation, together with a file called FILES.LST which contains a line for each file you are replacing/adding. Each line contains the name of the added file, the directory path relative to your development installation directory and the name you want the file to have in the installation. These three are separated by spaces.

### Example 3. Adding files to your installation

Suppose you want to replace the file `C:\ACE\APPLS\<MYAPPL>\STUFF.OLD` with the file `STUFF.NEW` after you have made up your installation. You add the files `STUFF.NEW` and `FILES.LST` to the last diskette of your installation. The file `FILES.LST` contains the single line:

```
        STUFF.NEW APPLS\<MYAPPL> STUFF.OLD
```

# Chapter 4. Runtime

Your runtime Application is the end result of your conversion process and it is the product that will be distributed to the end users. The runtime Application operates online with the host application. For this to function, you must open a line of communication between the runtime executable and the host. You must also set various runtime properties.

This chapter describes:

- Setting the Runtime Environment
- Runtime Options
- Running the Application
- Additional Runtime Settings

Your converted Application runs under a runtime environment. When you invoke the *Run Application* wizard from ACE you enter the runtime environment but the Application itself is not yet running. In this chapter you learn how to set up the runtime environment. Once configured, a connection to the host must be established and then the runtime can be launched. This chapter also gives you information to keep in mind before the user runs the Application for the first time.

Testing the runtime is also covered here. Testing the runtime properly is important as it uncovers, among other things, screens that may be misidentified during runtime. Finally the last sections cover PC to host communication and Emulator set up.

## Setting the Runtime Environment

The behavior of the runtime environment is independent of a specific Application. Runtime behavior can be reset each time the runtime environment is entered - without touching the Application executable.

When you run an Application for the first time, the runtime environment is created with certain default values. You may, if you wish, change these values. The new values are automatically preserved between runtime sessions. A brief outline is provided below.

## Runtime Options

You can configure a large number of display options for your Application. These options are configured from the *JIS Administrator* and include:

- Starting the Application running automatically.
- Making the Application window maximum size.
- The relationship of the Application's window to the runtime environment's window.
- Navigation options for Lists.
- How dynamic controls are displayed.
- How input prohibited fields are displayed.
- You can also lock some of these options from your end user.

## Running the Application

This section deals with running the application.

### The First Time

The first time you enter the runtime environment you must follow a two step process to actually start the Application executable running.

These steps are:

- Choosing an emulator.
- Starting the Application running.

To choose an emulator:

1. Start the JIS Server.
   Do this from your Windows *Start* menu > *Programs* > *JIS*> *JIS Server*.
2. Start the JIS Administrator.
   Do this from your Windows *Start* menu > *Programs* > *JIS*> *JIS Administrator*.
3. Click the *Runtime Configuration* tab.
   The Runtime Configuration interface is displayed.
4. In the *Category* pull-down listbox, select *Emulator Type*.
5. Select an emulator type from the *Emulator Type* combo box.

To start the Application running:

1 Start your web server.
2 From within ACE, go to the *File* menu and choose *Run Application*.

## The Next Time You Run the Application

The runtime environment saves your choice of emulator between sessions. If you wish to use the same emulator as in the previous session, you do not need to choose an emulator type.

## What Your End User Sees

Your end user will typically call up the Application by clicking its desktop icon. This puts the user into the runtime environment.

If you have configured the Application to start automatically, the runtime environment opens a host session and the user is presented with the initial screen of a host session. If the Application does not start running automatically, the user will need to start it manually from the runtime environment's *Application* menu.

Simply navigating to your Application from within a "green-on-black" PC terminal emulation session will not present your Application in GUI form. The GUI only appears when the host session is initiated by running your converted GUI Application or another Application that has been grouped with it using the cluster feature.

If you are a third party supplier and not an in-house developer, then you may not have included the initial screens of the end user's host session in your GUI Application. In this case, the user will see "green-on-black" screens if you have chosen the bleed through option, or an on-the-fly GUI if you have chosen to use Just-in-Time GUI.

When the Application is running, the user will have ready access to the runtime menus if you have chosen to display the Application window as an overlapping window. If you have chosen to display the Application as a child window, either maximized or normal, then the runtime menus are found in the *Application* submenu of the runtime environment's *Control* menu. You open a window's **Control** menu by clicking the button or picture in the extreme upper left hand corner of the window.

The end user will have no access at all to certain runtime options if you have chosen not to display the developer's dialog.

## Testing the Runtime

Certain problems that occur during the conversion can only be detected through functional testing. To perform functional testing on your runtime Application you should work online with the host.

Check your Application for each of the following:

- Functionality.
- Misidentified screens.
- Data flow.

## Functionality

To test a window for functionality make sure that:

- All controls have functionality attached to them.
- Each control executes the functionality assigned to it. Example: Verify that clicking a certain button in the GUI window presses the correct key on the host.

## Misidentified Screens

When ACE encounters an inconsistency between the host screen and the screen image, the runtime does not display the corresponding GUI. Either a simple GUI is presented or the host screen bleeds through depending on whether or not the Just-in-Time GUI feature was used.

To track down the reason for the unidentified screens:

**Table 15. Possible reasons for unidentified screens (Sheet 1 of 2)**

| Reason | Description |
|---|---|
| **An unconverted screen** | Create a new Subapplication. |
| **A screen that changed on the host after conversion** | Take the updated screen image through the *Maintain Screen Image* wizard and then to Runtime Screen Identification View to update identification. |

**Table 15. Possible reasons for unidentified screens (Sheet 2 of 2)**

| Reason | Description |
|---|---|
| **An incorrect screen image** | Correct the screen image, take it through the *Edit Screen Image* wizard and check where identification has failed in the Runtime Screen Identification View. |
| **Incorrect coloring in Runtime Screen Identification** | Identify the fields that are colored incorrectly, by using the *Compare to Captured Screen* utility, and then change their coloring. |

## Data Flow

Verify that all the data presented in the window is correct. If information is not being updated from the screen to the window make sure that the *Update text in runtime from screen to window* check box is set on the control's *Component > Manager* tab.

## A Word About Communication

Long before people thought about Client/Server applications, mainframe and iSeries computers were run through terminals. Because this is the conventional configuration, your host computer always expects to be talking to a terminal. ACE comes with two types of facilities that either enable or emulate communication between your host and your PC.

### Types of Communication

ACE can work either live for actually operating the host application or via a simulated emulator called the File emulator.

Live communication with the host can be established using `TCP/IP` or non `TCP/IP` means.

The File emulator is used for testing an Application offline.

### The File Emulator

ACE contains a proprietary emulator, called the File emulator, for testing an Application offline. The File emulator is used to simulate online operation with a live emulator. Use the File emulator to check Application screens while offline from the host.

### Live Communication

All of the systems using the runtime Application communicate live with the host computer. The host application runs invisibly while the end users see the converted Application only. The converted Application is completely synchronized with the host application. When data is entered on the GUI, it is transferred to the host.

### Direct Runtime/Host Communication using TCP/IP

ACE comes with proprietary `TN5250` or `TN3270` protocols for establishing direct communication between the runtime Application and the host.

TN5250 implementation acts as an iSeries terminal emulator which connects to the iSeries using TCP/IP. TN3270 performs the same function but is used to connect to a Mainframe. By using TCP/IP there is no need for a router to connect to the host.

### Runtime/Host Communication without TCP/IP

For iSeries Applications, using the JIS Internal 5250 WSF for working online with the iSeries is recommended. If you choose to use a commercial emulator of any type, check with webMethods JIS Technical Support to determine its compatibility with ACE.

For communicating with a Mainframe use any commercial emulator and router that is compatible with ACE.

### Using an Emulator

When not using a TCP/IP connection to the host or the Internal 5250 WSF you must install a software product called a Terminal Emulator (“Emulator”) on your PC. When this software is installed it makes your PC look like a terminal to the host computer. Thus you can run your host application from the PC.

You must install an emulator on both your development system and your target systems. Your development system must have emulator software so that you can capture your host application and test your converted Application. Your target systems require an emulator to run your host in the background.

ACE supports a wide variety of commercial emulators. ACE is also supplied with its own proprietary emulator. This section describes the emulators supported by ACE.

The following diagram illustrates the logical connection between the host application, the terminal emulator, and ACE. Whereas a terminal communicates directly with the host computer, the PC communicates to the host via a terminal emulator. ACE links and synchronizes between the GUI Application and the emulator.

When not using `TCP/IP` or the `Internal 5250 WSF`, ACE requires an emulator which operates under Windows, and `HLLAPI`. ACE supports a variety of live emulators. Consult your JIS service office for the most up-to-date list of supported emulators.

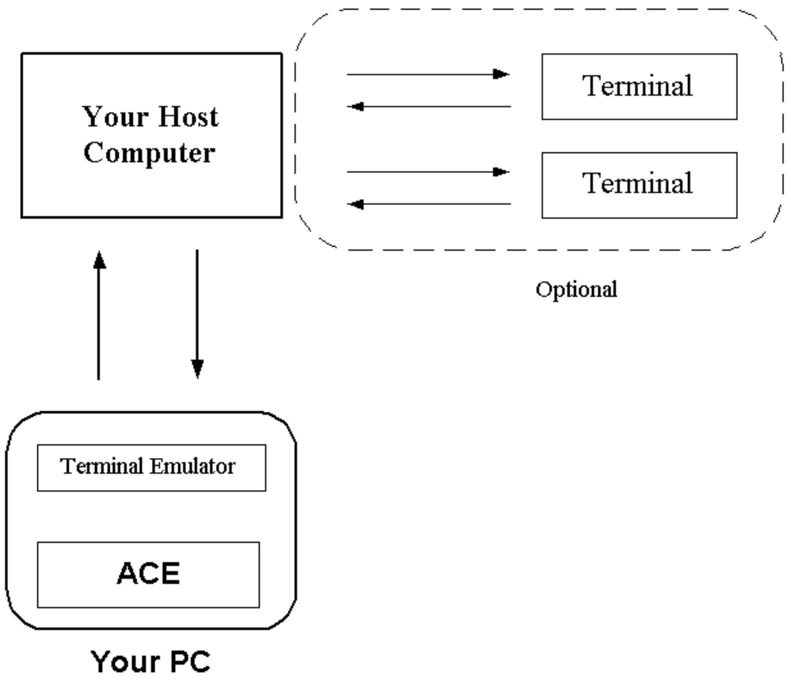


**Figure 37. Using an emulator**

## Additional Runtime Settings

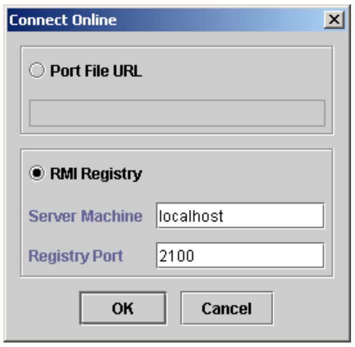
Additional runtime settings can be set in the JIS Administrator utility.

### Working Online

To work online:

1. Make sure that the server is running and open the *JIS Administrator*.
2. From the *Connect* menu, select *Connect*.

   The *Connect Online* dialog box opens:

   Server Machine: localhost
   Registry Port: 2100

   **Figure 38. Connect Online dialog box**

3. Set the JIS Server's *Port File URL* or *RMI Registry* by entering the necessary information in the appropriate fields.
4. Click *OK*.

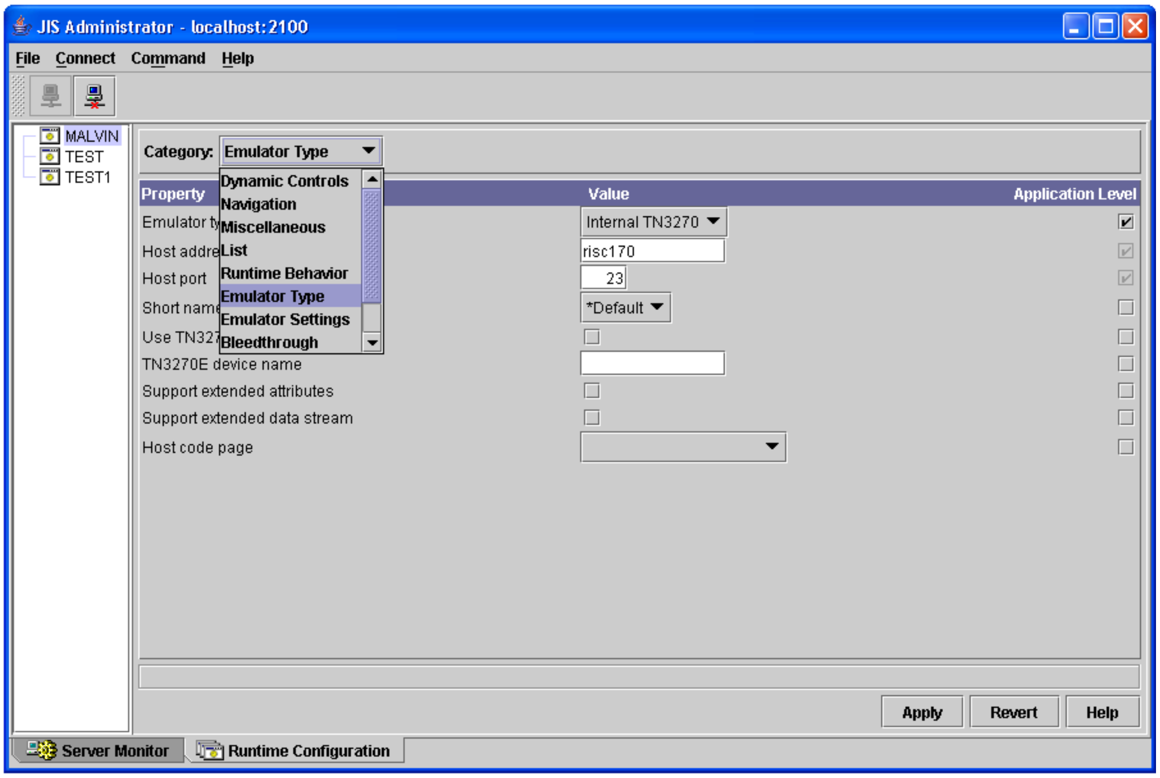
### Working With Emulators

You select and customize emulators in the JIS Administrator interface.

To select an emulator:

1. In the *JIS Administrator*, click the *Runtime Configuration* tab.

   The Runtime Configuration interface is displayed.
2. From the *Category* listbox, select *Emulator Type*.



Category: Emulator Type

| Property | Value | Application Level |
|---|---|---|
| Emulator ty | Internal TN3270 | ☑ |
| Host addre | risc170 | ☑ |
| Host port | 23 | ☑ |
| Short name | *Default | ☐ |
| Use TN327 | ☐ | ☐ |
| TN3270E device name | | ☐ |
| Support extended attributes | ☐ | ☐ |
| Support extended data stream | ☐ | ☐ |
| Host code page | | ☐ |

**Figure 39. Selecting the emulator type**

3 Select an emulator type from the *Emulator Type* combo box. Choose one of the following emulators:

- File
- Internal TN3270
- Internal TN5250

4 Set the emulator type properties. See the following section for full details regarding emulator type properties.

## Setting the Emulator Type Properties

When choosing an emulator, you can customize its properties. These customizations are different for each emulator type.

The following table lists the properties for each emulator type:

### File Emulator

The File Emulator settings are:

**Table 16. File Emulator settings**

| Property | Description |
|---|---|
| **First panel to display** | The first screen to be displayed out of the pool of panels. The advance mechanism of the panels is determined in the Panels.ini file. |

### Internal TN3270

The Internal TN3270 emulator settings are:

**Table 17. Internal TN3270 settings (Sheet 1 of 2)**

| Property | Description |
|---|---|
| **Host address** | Contains the name or the IP address of the host mainframe. |
| **Host port** | Contains the port number for the Telnet 3270 protocol. The port number is usually **23**. |
| **Short name** | Can usually be left as default. Some systems may require multiple sessions to be named **A**, **B**, etc. |
| **Use TN3270E if available** | Enables the use of the extended protocol of the 3270E emulator for printing emulation purposes. |
| **TN3270E device name** | When the *Use TN3270E if available* parameter is checked, specifies the device name for the printing session. |

**Table 17. Internal TN3270 settings (Sheet 2 of 2)**

| Property | Description |
|---|---|
| **Support extended attributes** | Used for compatibility between the converted screens and the current runtime. Clear checkbox if screens without extended attributes were converted. |
| **Support extended data stream** | Used for compatibility between the converted screens and the current runtime. Clear checkbox if screens without extended data stream were converted. |
| **Host code page** | The number and name of the `EBCDIC` code page on the host. This is used for choosing the `EBCDIC/ANSI` conversion table and during negotiation with the host. |

## Internal TN5250

The Internal TN5250 emulator settings are:

**Table 18. Internal TN5250 settings**

| Property | Description |
|---|---|
| **Host address** | Contains the name or the IP address of the host iSeries. |
| **Host port** | Contains the port number for the Telnet 5250 protocol. The port number is usually **`23`**. |
| **Short name** | Can usually be left as default. Some systems may require multiple sessions to be named **A**, **B**, etc. |
| **LU name** | Specify the device (LU) name for the session. Verify that the iSeries operating system supports external LU name recognition. |
| **ASCII/EBCDIC conversion table** | Choose the host application language. |

## Emulator Settings

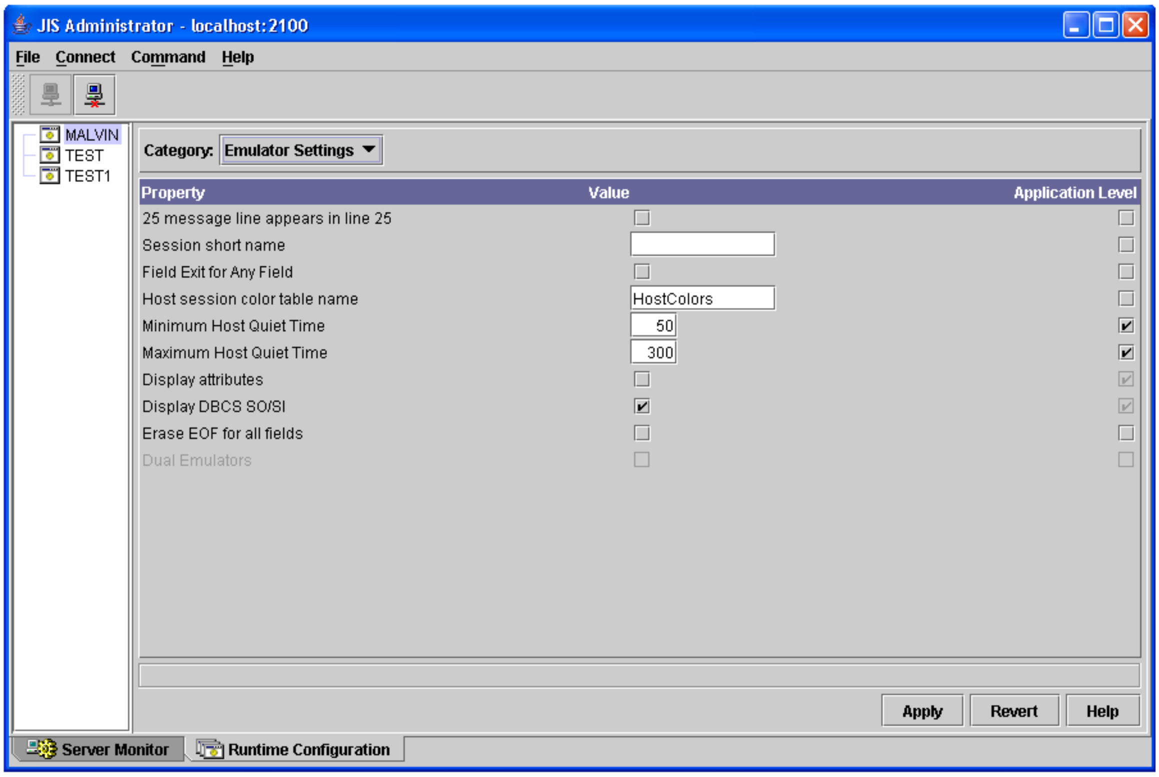
To change emulator settings:

1 In the *JIS Administrator (JAM),* click the *Runtime Configuration* tab.

The Runtime Configuration interface is displayed.

2 From the *Category* listbox, select *Emulator Settings*.

3 The *Emulator Settings* dialog box is displayed:



Category: Emulator Settings

| Property | Value | Application Level |
|---|---|---|
| 25 message line appears in line 25 | ☐ | ☐ |
| Session short name | | ☐ |
| Field Exit for Any Field | ☐ | ☐ |
| Host session color table name | HostColors | ☐ |
| Minimum Host Quiet Time | 50 | ☑ |
| Maximum Host Quiet Time | 300 | ☑ |
| Display attributes | ☐ | ☑ |
| Display DBCS SO/SI | ☑ | ☑ |
| Erase EOF for all fields | ☐ | ☐ |
| Dual Emulators | ☐ | ☐ |

**Figure 40. Emulator Settings dialog box**

4 Customize the emulator settings.

Table 19 lists and explains the Emulator Settings properties:

**Table 19. Emulator Settings properties (Sheet 1 of 2)**

| Property | Description |
|---|---|
| **25 message line appears in line 25** | Directs the emulator to retrieve messages from line 25 of the host screen. |

**Table 19. Emulator Settings properties (Sheet 2 of 2)**

| Property | Description |
|---|---|
| **Session short name** | Can usually be left as default. Some systems may require multiple sessions to be named **A**, **B**, etc. |
| **Field Exit for Any Field** | Automatic Field Exit for any field (not only numeric). |
| **Host session color table name** | Enters a color table to use for host sessions to override the green or black colors. |
| **Minimum Host Quiet Time** | Minimum amount of time (in milliseconds) the host screen must remain unchanged for the screen to be recognized. 0 - do not wait. |
| **Maximum Host Quiet Time** | Maximum amount of time (in milliseconds) to wait for the host screen to stabilize. |
| **Display attributes** | Option to display attributes in the host session. |
| **Display DBCS SO/SI** | When checked, the host screen displays Shift-Out/ Shift-In characters. When unchecked, these characters are replaced by blanks. |
| **Erase EOF for all fields** | Erases the remainder of a field using the 'Erase to End of Field' key instead of using blanks. |
| **Dual Emulators** | Determines whether JIS will work in dual emulator mode: GDS emulator and Telnet emulator. Note that this parameter is enabled only if you chose the GDS or the TN5250 emulators for *Emulator Type*. |

# Appendix A. JIS DDS Compiler

This section only applies to iSeries applications. The webMethods JIS software development kit is supplied with a CD-ROM containing the JIS DDS Compiler. The Compiler is installed on the iSeries.

This chapter describes:

- The installation procedure for the JIS DDS Compiler
- Using and operating the JIS DDS Compiler

The JIS DDS Compiler processes the source files into DDO files and saves them in a shared folder. This chapter describes the process for installing and operating the JIS DDS Compiler. You also learn how to select the files to be compiled and select the mode of operation.

## The JIS DDS Compiler

The JIS DDS Compiler processes DDS files into the DDO files that are used by ACE. The first stage in using ACE with DDS is to install the Compiler and then compile the DDS files. Then you copy the compiler output to your PC, and process it in ACE.

The following diagram illustrates the flow of the DDS files from the iSeries to the PC. The JIS DDS Compiler processes the DDS source files into DDO files and places them in a shared folder on the iSeries. Then you copy the DDO files to a directory on the PC. Keep a record of where the DDO files are on the PC. You will need to supply this information when you create screen images from the DDO files.

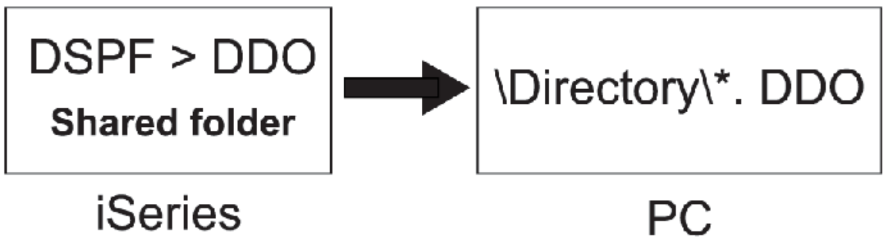


**Figure 41. JIS DDS compiler**

Screen images are the objects that ACE uses to generate the graphical interface.

## Preparations for Installation

The JIS DDS Compiler can be installed on the iSeries either as part of the general webMethods JIS installation or separately after the JIS Server has been installed. In this section we describe the procedure for the combined installation.

If you choose not to install the JIS iSeries components when installing JIS, you can install them at a later date by running the DDS Compiler setup located on the installation disk in the `\Utilities\JIS iSeries Components` directory.

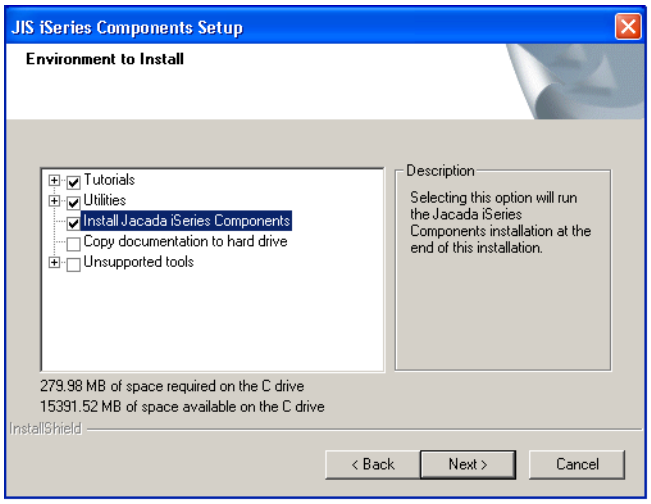
Before installing the DDS Compiler, several things should be determined and prepared:

- Your iSeries IP address.
- A valid user ID and password for the iSeries.
- An FTP connection to the iSeries.

## Installation Procedures

To install the JIS DDS Compiler together with JIS Server:

1 In the JIS Server *Setup* wizard, in the *Installation Options* step, select *Install JIS iSeries Components*.



JIS iSeries Components Setup

Environment to Install

- [✓] Tutorials
- [✓] Utilities
- [✓] Install Jacada iSeries Components
- [ ] Copy documentation to hard drive
- [ ] Unsupported tools

Description

Selecting this option will run the Jacada iSeries Components installation at the end of this installation.

279.98 MB of space required on the C drive
15391.52 MB of space available on the C drive

**Figure 42. webMethods JIS setup wizard**

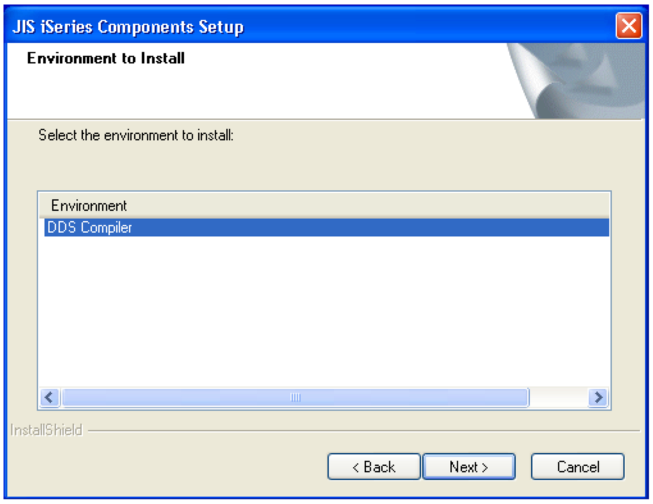
2 Click *Next* and continue through the wizard. At the end of the JIS installation process the *JIS iSeries Components Setup* wizard opens.

3 Specify the iSeries IP address or name, Username and Password. The setup wizard searches the host system for previously installed webMethods JIS components.

4 If a previous installation exists, you are asked whether you want to install a new copy or if you want to upgrade an existing library.

5 Enter the Program Objects Library and Common Work Objects Library names. The Program Objects Library must either be empty or not exist on the target system. If you specify a library that is not empty, you are asked to either delete the library or specify a new name.

6 Select the environments to install.


JIS iSeries Components Setup

Environment to Install

Select the environment to install:

| Environment |
|---|
| DDS Compiler |

**Figure 43. Select the environments to install**

7 Click *Finish* to complete the installation.

## Installing the JIS DDS Compiler Manually

In the event that you cannot install the JIS DDS Compiler using the installation wizard, you can install the JIS DDS Compiler manually on the iSeries.

To install the JIS DDS Compiler manually on the iSeries you will need access to a PC as well as the iSeries. The installation procedure includes steps on both machines.

Before installing the JIS DDS Compiler on the iSeries make sure you have:

- The webMethods JIS installation CD.
- A PC with an iSeries capable FTP client.
- The IP address of the iSeries.

## On the PC

Open an FTP connection to the iSeries and transfer the relevant JIS DDS Compiler objects from the PC to the iSeries.

1. Go to the DDS Compiler objects `jacprd0005` and `jacprd0000` located on the installation CD at:
   ```
   \Utilities\JIS iSeries Components\JIS
   ```
2. Open an MSDOS session.
3. In the MSDOS session open an FTP connection to the iSeries. Type:
   ```
   ftp <iSeries_host_ip_address>
   ```
   For example: `ftp 123.45.678.9`
4. Provide your username and password.
5. In the FTP session, type the following commands:
   ```
   bin
   cd QGPL
   lcd Utilities\JIS iSeries Components\JIS
   quot rcmd crtsavf QGPL/jacprd0005
   put jacprd0005 QGPL/jacprd0005
   quot rcmd crtsavf QGPL/jacprd0000
   put jacprd0000 QGPL/jacprd0000
   ```

## On the iSeries

On the iSeries create a library to contain the DDS Compiler files, empty the contents of the save file into this library, and then delete the save file.

1. Issue the command `CRTLIB JACDDSCOM` (this name can be changed). This creates a library called `JACDDSCOM`.
2. Issue the command:
   ```
   RSTOBJ OBJ(*ALL) SAVLIB (JACPRD0005) DEV(*SAVF) SAVF(QGPL/
   JACPRD0005) MBROPT(*ALL) ALWOBJDIF(*ALL) RSTLIB(JACDDSCOM)
   ```
   This restores the JIS DDS Compiler objects into the `JACDDSCOM` library.
3. Issue the command:
   ```
   DLTF FILE(QGPL/JACPRD0005)
   ```
   This deletes the jacprd0005 save file.
4. Repeat steps 2 and 3 using the file `jacprd0000`.

## Using the JIS DDS Compiler

Perform the following steps to compile your DDS files on the iSeries using the JIS DDS Compiler:

1. Install the JIS DDS Compiler.

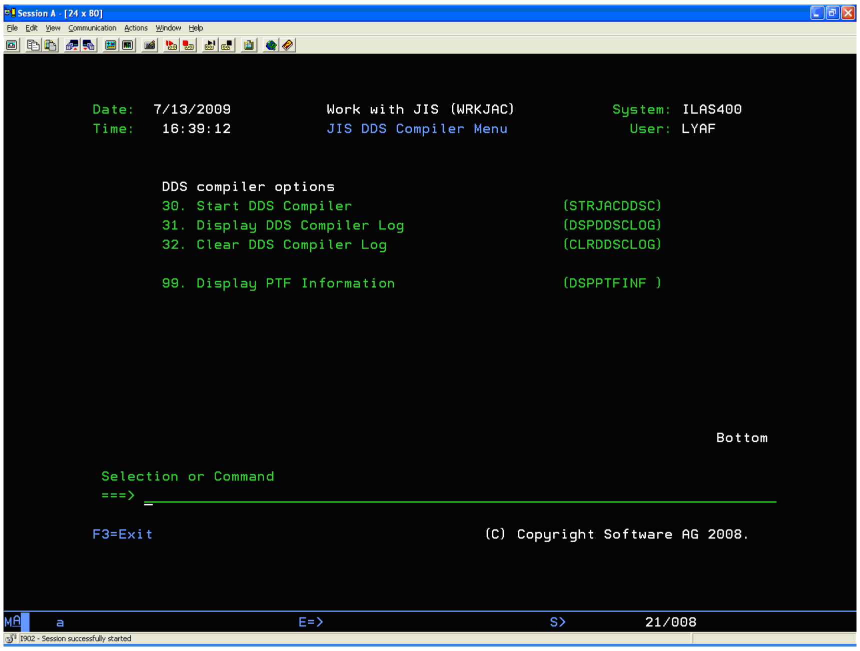
   See "Installation Procedures" on page 100.
2. Call up the Compiler on the iSeries.
3. Specify the DDS members to be compiled and their location.
4. Select the mode of operation. DDS files can be compiled interactively or can be submitted to a batch.

The output of the JIS DDS Compiler is placed in a shared folder. You can either create a new shared folder on the iSeries or access an already existing shared folder.

**Note:** You must verify that you are enrolled in the system distribution directory and that you have the authority to access the folders. Otherwise, the system will not allow you access to the shared folders.

## Operating the JIS DDS Compiler

The *Work With JIS* screen is the main menu for working with the DDS Compiler. Type `WRKJAC<ENTER>` from any screen to get to the *Work With JIS* screen:



```
 Date:   7/13/2009          Work with JIS (WRKJAC)           System: ILAS400
 Time:    16:39:12          JIS DDS Compiler Menu              User: LYAF


         DDS compiler options
         30. Start DDS Compiler                             (STRJACDDSC)
         31. Display DDS Compiler Log                       (DSPDDSCLOG)
         32. Clear DDS Compiler Log                         (CLRDDSCLOG)

         99. Display PTF Information                        (DSPPTFINF )








                                                                        Bottom

  Selection or Command
  ===> _

 F3=Exit                                      (C) Copyright Software AG 2008.
```

Session A - [24 x 80]
File Edit View Communication Actions Window Help
MA a E=> S> 21/008
I902 - Session successfully started

**Figure 44. The Work With JIS screen**

The *Work With JIS* screen provides the following options:

- *Start JIS DDS Compiler*: This calls up the DDS compiler, sending you to the *Start JIS DDS Compiler* screen.

  You can also do this by typing `STRJACDDSC` from any screen.
- *Display DDS Compiler Log*: This displays the contents of the log file, by sending you to the *Select Log File* screen, where you select the log file to display. In the *Select Log File* screen you can look for errors in the log file.

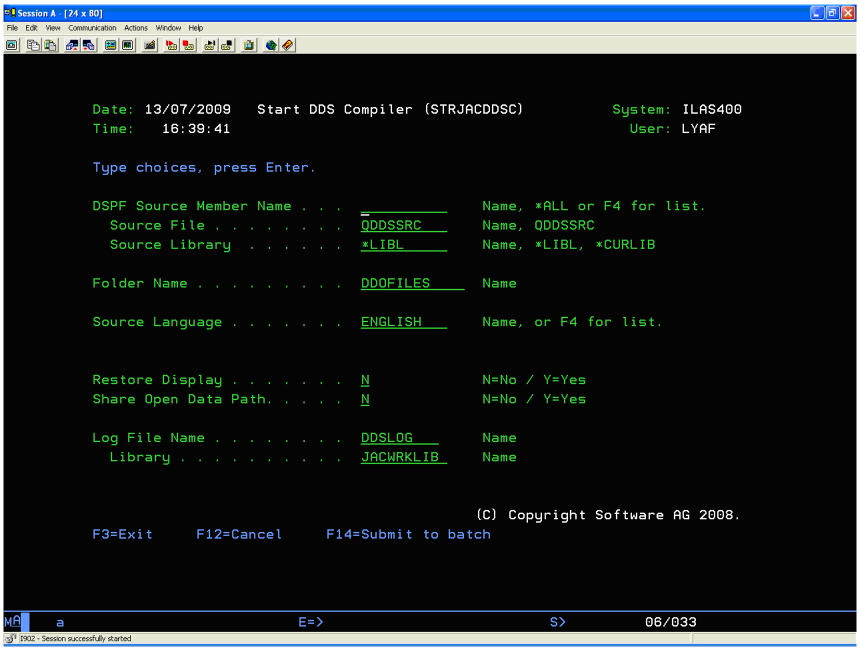
  You can also do this by typing `DSPDDSCLOG` from any screen.
- *Clear DDS Compiler Log*: This clears the log file.

  You can also do this by typing `CLRDDSCLOG` from any screen.
- *Display PTF Information*: This displays information on the current ACE version.

  You can also do this by typing `DSPPTFINF` from any screen.

## Selecting the Files to be Compiled

Use the *Start JIS DDS Compiler* screen to enter the names and locations of the DDS files to be compiled. The *Start JIS DDS Compiler* screen is displayed as follows:



```
Date: 13/07/2009   Start DDS Compiler (STRJACDDSC)          System: ILAS400
Time:    16:39:41                                             User: LYAF

Type choices, press Enter.

DSPF Source Member Name . . .  __________   Name, *ALL or F4 for list.
  Source File . . . . . . . .  QDDSSRC      Name, QDDSSRC
  Source Library  . . . . . .  *LIBL        Name, *LIBL, *CURLIB

Folder Name . . . . . . . . .  DDOFILES     Name

Source Language . . . . . . .  ENGLISH      Name, or F4 for list.


Restore Display . . . . . . .  N            N=No / Y=Yes
Share Open Data Path. . . . .  N            N=No / Y=Yes

Log File Name . . . . . . . .  DDSLOG       Name
  Library . . . . . . . . . .  JACWRKLIB    Name


                                          (C) Copyright Software AG 2008.
F3=Exit     F12=Cancel     F14=Submit to batch

MA     a                          E=>                          S>          06/033
I902 - Session successfully started
```

Figure 45.  Start JIS DDS Compiler screen

Enter information in the following fields on this screen:

Table 20.  Fields in the Start JIS DDS Compiler screen (Sheet 1 of 3)

| Field | Enter Information |
|---|---|
| **DSPF Source Member Name** | The name of the display member to be compiled. Enter one of three possibilities in this field.<br>• The specific name of the DDS member to be compiled.<br>• `*ALL` - all members located in the file specified in the second parameter will be compiled.<br>• F4 - call up a popup screen containing a list of display members (see “Selecting DDS Members from a List” on page 108). |

**Table 20. Fields in the Start JIS DDS Compiler screen (Sheet 2 of 3)**

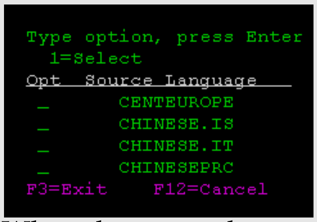
| Field | Enter Information |
|---|---|
| **Source File** | The name of the file in your iSeries application that contains the DDS sources. The default name `QDDSSRC` is entered in this field when you call up the Compiler. Change this name if your source file has another name. |
| **Source Library** | `*LIBL` is the default parameter. This parameter initiates a search of all the libraries in the user library list until the source file is found.<br><br>`*CURLIB` initiates a search of the current library. You can also supply the name of the library that contains the DDS source file. |
| **Folder Name** | The name of the shared folder where the compiled DDS files will reside. The PC system running the ACE Converter accesses the files from the shared folder.<br><br>If the designated shared folder already exists on your system, enter the folder name. You can create a shared folder from this screen by entering a new name in the field. The Compiler creates a shared folder with this name. |
| **Source Language** | The language of your application. English is the default language.<br><br>Press `F4` for a list of supported languages. The languages are displayed in the following popup window. Press *Page Down* to view additional supported languages.<br><br><br>`Type option, press Enter`<br>`  1=Select`<br>`Opt  Source Language`<br>` _       CENTEUROPE`<br>` _       CHINESE.IS`<br>` _       CHINESE.IT`<br>` _       CHINESEPRC`<br>`F3=Exit    F12=Cancel`<br><br>When the source language is Japanese the *Start JIS DDS Compiler* screen includes the *Single Byte Language* field where you must enter the name of a Single Byte Language. |
| **Single Byte Language** | The name of the Single Byte Language used when Japanese is the source language. Enter `F4` in the field to view a list of supported languages |

**Table 20. Fields in the Start JIS DDS Compiler screen (Sheet 3 of 3)**

| Field | Enter Information |
|---|---|
| **Restore Display** | Specifies whether data being shown at a display device by this display file is saved at the time the file is suspended (made temporarily inactive). In this way, a different display file can be used to show different data on the same device.<br><br>`N=No`. The data being shown by this file is not saved when the file is suspended.<br><br>`Y=Yes`. The data being shown when the file is suspended is saved. In this way it can be restored to the display of the device when the file is used again. |
| **Share Open Data Path** | Specifies whether the Open Data Path (ODP) is shared with other programs in the same routing step.<br><br>`N=No`. The ODP is not shared with other programs in the routing step. A new ODP is created and used every time a program opens the file.<br><br>`Y=Yes`. The same ODP is shared with each program in the job which also specifies a YES when it opens the file. |
| **Log File Name** | The name of the log file in which information on the compiled DDS files will be stored. The default name is `DDSLOG`. |
| **Library** | The library, in which the log file resides. The default library is the library that you specified during the setup procedure. |

**Note:** If you are not enrolled in the system distribution directory or you do not have authority to access the shared folders, the system will terminate the process.

## Select the Mode of Operation

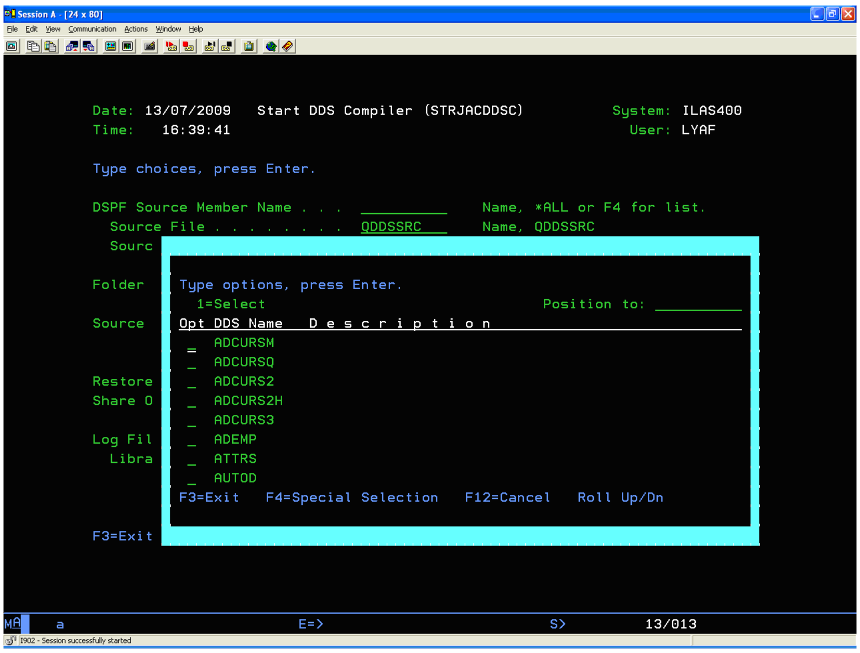
When you have entered the information in the fields, press one of the following:

| | |
|---|---|
| **Press Enter** | To compile all DDS members interactively. |
| **Press F14** | To run the compilation in a batch. Running the compilation in a batch frees the screen for other work. |

## Selecting DDS Members from a List

The JIS DDS Compiler enables you to select single or multiple members from a list of DDS members in the file that you have specified. To call up a list of DDS members, press F4 when the cursor is on the *DSPF Source Member Name* input field.

The system displays the list of DDS members in the location that you have specified in a popup window, as follows:



```
          Date: 13/07/2009   Start DDS Compiler (STRJACDDSC)          System: ILAS400
          Time:   16:39:41                                              User: LYAF

          Type choices, press Enter.

          DSPF Source Member Name . . .  __________   Name, *ALL or F4 for list.
            Source File . . . . . . . .  QDDSSRC      Name, QDDSSRC
            Sourc

          Folder   Type options, press Enter.
                     1=Select                                 Position to: __________
          Source   Opt DDS Name   D e s c r i p t i o n
                    _   ADCURSM
                    _   ADCURSQ
          Restore   _   ADCURS2
          Share O   _   ADCURS2H
                    _   ADCURS3
          Log Fil   _   ADEMP
            Libra   _   ATTRS
                    _   AUTOD
                   F3=Exit   F4=Special Selection   F12=Cancel   Roll Up/Dn

          F3=Exit

MA   a                                E=>                          S>         13/013
I902 - Session successfully started
```

Figure 46.  List of DDS members

Type a DDS file name in the *Position to* field in order to position the file at the top of the list. By entering a single character in the *Position to* field, the first file name starting with that character is positioned at the top of the list. If there are no file names that begin with that specific character, then the file name beginning with the next closest alphabetic character is positioned at the top of the list. Select an item from the list by entering the number *1* next to the item.

After you have chosen the items from the list press one of the following:

| | |
|---|---|
| **Enter** | To confirm your selections. The string *List is displayed in the *DSPF Source Member Name* input field. |
| **F12** | To cancel the request for the popup window and return to the main Compiler screen. |
| **F13** | To exit the entire Compiler application. |

## Compiling DDS Members by Direct Entry

If you wish to compile a single DDS member without being prompted by the *Start JIS DDS Compiler* screen, type `STRJACDDSC` and press `F4` (instead of `Enter`).

This opens the *Start JIS DDS Compiler* screen without the prompt options

## The DDS Log File

For each compiled DDS a record is written to the `DDSLOG` file in the default library. When the reference field cannot be resolved, the message `***Error1***` appears in place of the document name, and a detailed explanation appears in the `ERRMSG` field.

# Index