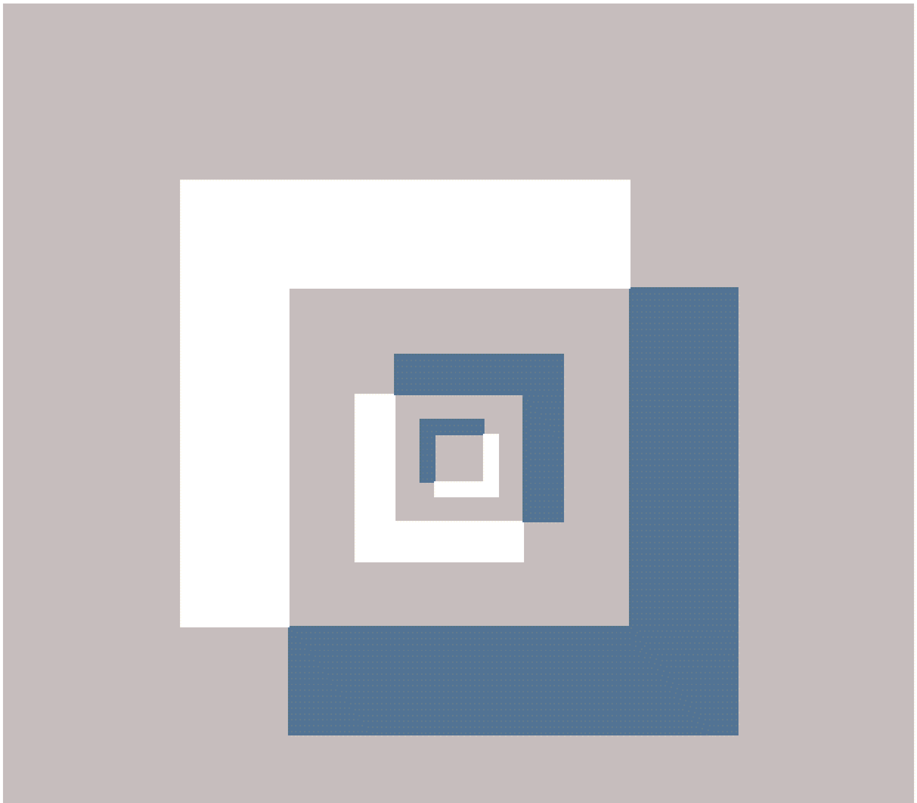
Issue 4b, March 2002



Micro Focus

Net Express®

Getting Started

20020423093343

Table of Contents

Part 1: Welcome to Net Express!

1 Overview 13

1.1 What's in Net Express? 13

1.2 Additional Software 16

2 Finding Information 17

3 Licensing and Deployment 21

3.1 Licensing 21

3.2 When Do You Require an Application Server License? 21

3.3 Deploying Your Application 22

3.4 Licensing of Additional Software 22

3.4.1 ODBC Drivers 23

3.4.2 Data Table Grid Control 23

3.4.3 Other Third-party Components 23

4 Start Here for the Tutorials 25

4.1 Overview 25

4.2 Terminology and Conventions 26

4.3 Tips 27

4.4 Tutorials Map 28

Part 2: Basic Tutorials

5 Using Net Express 33

5.1 Overview 33

5.1.1 Projects and Project Folders 33

5.1.2 Demonstration Applications 34

5.2 Sample Session 35
5.2.1 Starting the IDE 35
5.2.2 Loading a Project 37
5.2.3 Building a Project 38
5.2.4 Running the Application 39
5.2.5 Debugging an Application 41
5.2.6 Setting IDE Options 42
5.2.7 Ending Animation 43
5.2.8 Using Context Help 43

5.3 Before Continuing 44

6 Maintaining and Creating Data Files 45

6.1 Overview 45

6.2 Preparation 45

6.3 Sample Session 46
6.3.1 Converting a File 46
6.3.2 Viewing a Data File Unformatted 47
6.3.3 Creating a Record Layout File 49
6.3.4 Creating the Default Record Layout 50
6.3.5 Creating the Conditional Record Layouts. 51
6.3.6 Saving the Record Layout File 52
6.3.7 Viewing a Data File Formatted 52
6.3.8 Looking Up Information about a Record Layout File 53
6.3.9 Viewing Files in Hexadecimal 54
6.3.10 Printing a Data File 54
6.3.11 Editing Multiple Data Files 55
6.3.12 Creating a New Data File. 56
6.3.13 Converting a File's Character Set 57
6.3.14 Viewing the EBCDIC File 58

6.4 Before Continuing 59

Part 3: Web Development Tutorials

7 Introduction to Web Applications 63

7.1 Solo 64

7.2 Server-side Programs 64

7.2.1 Structure 64

7.2.2 Symmetric and Asymmetric 65

7.3 Before Continuing 66

8 Creating a Web Application 67

8.1 Overview 67

8.2 Preparation 68

8.3 Sample Session 68

8.3.1 Creating the Project and Form 68

8.3.2 Your IDE at this Point 70

8.3.3 Adding Controls to the Form 71

8.3.4 Creating the Server-side Program 74

8.3.5 Files Created 75

8.3.6 Testing the Form 75

8.4 Before Continuing 77

9 Completing and Running Your Web Application . . 79

9.1 Overview 79

9.2 Preparation 79

9.3 Sample Session 80

9.3.1 Editing the COBOL Program. 80

9.3.2 Building the Application 81

9.3.3 Running the Application 82

9.3.4 Debugging the Application 83

9.4 Before Continuing 84

10 Creating a Web Application from a COBOL Application. 85

10.1 Overview 85

10.2 Preparation 86

10.3 Sample Session 86

10.3.1 Creating the Project, Forms and Program 86

10.3.2 Adding Field Properties 88

10.3.3 Files Created 90

10.3.4 Building the Application 91

10.3.5 Running the Application 91

10.4 Before Continuing 92

11 Creating a Web Database Application 93

11.1 Overview 93

11.1.1 Terminology 94

11.2 Preparation 94

11.3 Sample Session 94

11.3.1 Creating the Project, Forms, and Programs 95

11.3.2 Files Created 98

11.3.3 Building the Application 99

11.3.4 Running the Application 99

11.3.5 Navigating the Database 100

11.3.6 Updating the Database 101

11.3.7 Filtering the Database 102

11.4 Before Continuing 103

Part 4: Windows GUI Development Tutorials

12 Creating a Windows GUI Application 107

12.1 Overview 107

12.1.1 Creating a Screenset 108

12.2 Preparation 108

12.3 Sample Session 108

12.3.1 Creating the Project and Screenset 109

12.3.2 Your Screen at this Point 110

12.3.3 Defining Data for the Screenset 112

12.3.4 Adding Controls to the Screenset 114

12.3.5 Adding Dialog to the Screenset 117

12.3.5.1 Global Dialog 118

12.3.5.2 Local Dialog 119

12.3.6 Files Created 120

12.3.7 Building the Application 121

12.3.8 Testing the Screenset 121

12.4 Before Continuing 123

13 Completing and Running Your Windows GUI Application 125

13.1 Overview 125
13.1.1 Structure of an Associated Program 125

13.2 Preparation 126

13.3 Sample Session 126
13.3.1 Editing the COBOL Program. 126
13.3.2 Building the Application 127
13.3.3 Running the Application 127

13.4 Before Continuing 128

Part 5: UNIX Option Tutorials

14 Deploying an Application on UNIX 131

14.1 Overview 131

14.2 Preparation 132

14.3 Sample Session 132
14.3.1 Creating a Project 133
14.3.2 Building the Application 133
14.3.3 Using Your PC as a UNIX Terminal 134
14.3.4 Setting Up the Options for Publishing 135
14.3.5 Publishing the Application. 139
14.3.6 Files Created 139
14.3.7 Running the Application 140

14.4 Before Continuing 140

Part 6: SQL Option Tutorials

15 DB2 Applications (SQL Option). 145

15.1 Overview 145

15.2 Preparation 146

15.3 Sample Session 146
15.3.1 Creating a Project 146
15.3.2 Adding Files to the Project. 147
15.3.3 Building the Project 147

15.3.4 Starting the XDB Server 147
15.3.5 Running the Application 148
15.3.6 Debugging the Application 148
15.3.7 Closing the XDB Server 149
15.3.8 Setup access to mainframe through XDB Link 149

15.4 Before Continuing 150

16 Maintaining a DB2 Database 151

16.1 Overview 151

16.2 Preparation 152

16.3 Sample Session 152
16.3.1 Start the SQL Wizard 153
16.3.2 Create a New Table 153
16.3.3 Defining the Primary Key 154
16.3.4 Define a Secondary Index 156
16.3.5 Viewing the SQL 156
16.3.6 Finishing Creating the Table 157
16.3.7 Adding Data to the Table 157
16.3.8 Creating a Query 158
16.3.9 Running the Query 159

16.4 Before Continuing 159

Part 7: Appendices

A Windows Tips 163

A.1 Standard Buttons on a Window 163

A.2 Changing Folder 164

A.3 Manipulating Folders 165

A.4 Tree Views 165

A.5 Showing/Hiding File Extensions 165

A.6 Column Headings in a Container 166

A.7 Selecting Items in a List 167

A.8 Dropdown Lists 168

A.9 Popup Menus 169

A.10 Cut and Paste 169

A.11 Drag and Drop 170

A.12 Resizing and Docking Windows 170

B More Features 173

B.1 Descriptions of Features 173
B.1.1 List of Features 173
B.1.1.1 Btrieve Interface 173
B.1.1.2 Callable File Handler 173
B.1.1.3 Callable Rebuild. 174
B.1.1.4 Callable Sort 174
B.1.1.5 Cbllink 174
B.1.1.6 Client/Server Bindings 174
B.1.1.7 COBOL System Library Routines 174
B.1.1.8 Command Prompt 175
B.1.1.9 Common Gateway Interface (CGI) Support 175
B.1.1.10 Embedded HTML 175
B.1.1.11 Fileshare 175
B.1.1.12 Integrated Preprocessor Support 175
B.1.1.13 Microsoft Transaction Server Support 176
B.1.1.14 Multithreading 176
B.1.1.15 National Language Support 176
B.1.1.16 Object-oriented COBOL Syntax 176
B.1.1.17 Object Request Broker (ORB) 176
B.1.1.18 ODBC Drivers 177
B.1.1.19 OLE Automation Support 177
B.1.1.20 OpenESQL 178
B.1.1.21 OpenESQL Assistant 178
B.1.1.22 PVCS 178
B.1.1.23 Rebuild 178
B.1.1.24 Sort Utility 178
B.1.1.25 Source Code Control System Support 178
B.1.1.26 WebSync link 179
B.1.1.27 Win32 Native API programming 179

B.2 Mainly for Migration. 179
B.2.1 List of Features 179
B.2.1.1 COBSQL 179
B.2.1.2 Dialog Editor. 180
B.2.1.3 On-line Help System 180
B.2.1.4 Panels 180
B.2.1.5 Screen and Keyboard Handler (Character Mode). . . 180
B.2.1.6 Screen and Keyboard Configuration Tools 180
B.2.1.7 Screen Section. 181
B.2.1.8 Windowing Support 181

C Web Applications 183

D Configuring Your Web Server. 185

Part 1: Welcome to Net Express!

This part contains the following chapters:

- Chapter 1, “Overview”
- Chapter 2, “Finding Information”
- Chapter 3, “Licensing and Deployment”
- Chapter 4, “Start Here for the Tutorials”

1 Overview

Congratulations on your choice of Net Express, the most productive tool for developing distributed applications. Net Express is a complete, integrated development environment for Internet applications. You can create Web forms, and generate a complete working program to handle them. Then you simply add your business logic to the program. You can also generate a complete Web application from an existing application.

Net Express has all the tools and facilities to create, develop, build and test Web applications. It even includes a personal Web server.

You can also use Net Express for developing distributed applications that have a Windows user interface.

Net Express also enables you to use new component-based technologies like Microsoft's COM/DCOM and ActiveX, and Sun Microsystems' Java Beans to build enterprise components.

In this book, we will generally simply say "Windows" as an abbreviation for Windows 95, Windows 98, Windows 2000, and/or Windows NT 4.0.

Apart from this ***Getting Started*** book, all the documentation is online only. You get to it from the Windows **Start** menu, or from Net Express's **Help** menu. This ***Getting Started*** book is both online and in printed book form. All necessary instructions for installing Net Express are displayed by the installation utility, Setup.

1.1 What's in Net Express?

Let's start with a brief description of the major parts of Net Express. Once you're familiar with these you will be ready to start using

Net Express. The chapters *Start Here for the Tutorials* onward are a set of tutorials that takes you through all the major parts:

- The Development Environment

  The main window, with pulldown menus, in which you edit, compile and debug your code, and via which you access most of the Net Express tools. This is often referred to as the IDE, for Integrated Development Environment.

- Project

  A file detailing all the files in your application, and how they should be compiled and linked. A project is very easy to create, and enables you to compile and link the entire application with a single mouse-click. You should create a project for every application, even the simplest.

- Data Tools

  A set of tools that enable you to convert, browse, edit and create data files used by an application. You can examine data files to see how an application has updated them, or create and edit a file to provide test data for an application. Files can be in any COBOL format and you can view them at both the record and field levels.

- HTML Page Wizard

  The tool you use for creating the user interfaces for your Web applications. You specify your Web forms, and it generates the necessary files.

- Internet Application Wizard

  The tool you use to create Web applications. It does virtually all the writing of HTML pages and coding of COBOL to handle these pages for you. You can create an application in three ways:

  - from an HTML form you've created using HTML Page Wizard
  - from the Linkage Section of an existing COBOL program
  - from the structure of an existing database (you can use it with any database for which an ODBC driver is available)

- Form Designer

  A WYSIWYG form editor, which you use to design forms for your Web-based applications.

- Solo

  Web server software to run on your own computer. You use it for testing your Web applications.

- Windows GUI Application Wizard

  The tool you use to create Windows-based applications. You use this to create a screenset, in which details of windows and dialog boxes are stored, and to generate a COBOL program to handle them.

- Dialog System

  The tool you use to design windows and dialog boxes for your Windows-based applications. You can also write procedures for handling events.

- PVCS

  A popular Source Code Control System (SCCS). It gives multiple users controlled access to source code stored centrally on a server. It stores changes in the form of deltas so that it can re-create successive versions of a file. Net Express includes most of the commonly used functionality of PVCS.

- UNIX Option

  The tool you use to deploy Net Express applications to a UNIX server. It also provides tools to allow the complete development of UNIX applications using Net Express.

- SQL Option

  SQL Option enables you to include a DB2 application in a Net Express project. You can also maintain a DB2 database using an interactive tool, the SQL Wizard. You can compile your DB2 application, edit it, and debug it in a DB2 environment on your PC. You can also compile, edit and debug your DB2 application against the mainframe itself.

1.2 Additional Software

Some non-Micro Focus software is supplied under license on the Net Express CD-ROM or an accompanying CD-ROM.

- Microsoft Internet Explorer

  Microsoft Internet Explorer is a popular Web browser. You need it because Form Designer uses some of its features when you are designing your form, and because you need a Web browser to read the Net Express documentation and to test your Web applications. If you do not already have it, or do not have the version Form Designer needs, Setup offers to install it.

- Win32 Software Development Kit

  The Win32 Software Development Kit (SDK) is a set of tools and system libraries useful when writing Windows applications. Install it using **setup.exe** in the root of the Win32 SDK CD-ROM that accompanies the Net Express CD-ROM.

- Adobe Acrobat Reader

  Adobe Acrobat Reader enables you to read online books in the format supported by Adobe Acrobat. The Net Express online books are provided in this format as well as in the form of Web pages - see the chapter *Finding Information* for details.

- Server Control Program (SCP)

  The Server Control Program (SCP) is server software for use on UNIX. You must install it on your UNIX machine if you want to use the UNIX Option of Net Express. See the online book ***UNIX Option User's Guide*** for details.

- Samba

  Samba enables you to see drives on a UNIX server from your PC. You install it on the UNIX machine. See the online book ***UNIX Option User's Guide*** for details.

2 Finding Information

Apart from this ***Getting Started*** book, all the documentation is online only. You get to it from the Windows **Start** menu, or from Net Express's **Help** menu. This ***Getting Started*** book is both online and in printed book form.

Much of the documentation is in the form of Windows help. The rest is in the form of online books written as Web pages. You get to these from the Windows **Start** menu or from the Net Express help - when you click a book's button in the help, the book is displayed in your Web browser.

When you're looking for information, we recommend trying the different kinds of documentation in the following order:

1 See if there's a tooltip.

If you want information about a field or control, try leaving the mouse pointer on it for a moment. A brief description of its purpose may be displayed, either beside it or at the bottom of the Net Express window.

2 Try context help.

If you want information about a menu item or control, click the Context Help button ? on the toolbar, then click the menu item or control.

3 Look in the help, using the help index.

If Net Express is running, click **Help Topics** on the Help menu. Otherwise, from the **Start** menu, click **Programs**, then **Micro Focus Net Express**, then **Help**. Then click the **Index** tab and enter a relevant word. The index will automatically position at the entry, if one exists. Alternatively, scroll through the index to find what you want.

In some cases the help index takes you to an online book via a help topic. If this happens, click **Index** in the Table of Contents of the online book, click the first letter of the phrase you're looking for, then scroll the index to find the phrase.

4. Look in the online books, using the Master Index.

   From the **Start** menu, click **Programs**, then **Micro Focus Net Express**, then **Bookshelf**. Then in the left-hand pane, click **Index**. Click the first letter of the word you want at the top of the right-hand pane, then scroll through the index to find the word.

5. Look in the help using Find.

   Return to Help Topics, click the **Find** tab, and enter the word you're looking for. If this is the first time you've used **Find** since installing, the system will take a little while to create the dictionary.

   There is no equivalent to Find for the online books.

6. Look in the online books using Search.

   From the **Start** menu, click **Programs**, then **Micro Focus Net Express**, then **Bookshelf**. Then in the left-hand pane, click **Search**. You can choose whether to use keyword highlighting, or navigation controls, or both. Enter the word you're looking for. Scroll through the list and select an entry by clicking on it.

7. If you still haven't found the information you want, repeat Steps 3 through 6 using some other relevant word.

You will find it useful to select the **Contents** tab in Help Topics and spend a few minutes looking through the Contents to see what the help covers. Similarly, select each book in turn from the Bookshelf and look through their contents to see what they cover.

Here are some further tips:

- The online help is mainly for quick help. To read about a subject in depth, look first in the appropriate online book, and if you don't find it look in the help. The books are all accessible from the help Contents.There are separate indexes for the books, the Net Express Help, the Readme Help, and the Help for individual options such as Dialog System. The index for the books is called the Master Index because it covers all the books. The books have individual indexes as well.

- You can print out the online books from your Web browser. Each Web page is one chapter; so if your Browser's print function prints one page, each print operation prints one chapter. To be certain of printing the text rather than the Contents list, click first in the frame containing the text.

- The Net Express CD-ROM contains the Adobe Acrobat Reader and an Adobe Acrobat file for each book, from which you can print the whole book in one operation. You are authorized to print one copy of each book for your own personal use.
- Links in the online books are either to other chapters or books, or to the online glossary. To return, use your Web browser's **Back** button. You may have to click it twice to get the correct Contents list back.
- The online books have the Contents in the left-hand frame and the text in the right-hand frame. On some browsers, when you click a link in the Contents, you can get the text in a full frame by dragging the mouse a little before releasing the button. If you click a link in the text you get the Contents of the target chapter in a full frame.
- Two online books are especially worth looking at after this one. The ***Solutions Guide*** guides you through developing certain frequently required types of application. The ***Migration Cookbook*** gives advice on migrating from other COBOL systems to Net Express.
- To get a popup menu with functions appropriate to an item, right-click the item. These functions often include **What's This?**.
- To get the syntax for a COBOL reserved word visible in a text window in the IDE, put the cursor anywhere within the word, then press **F1**. Alternatively, right-click on the word and click **Help** on the popup menu.
- If you have any problem installing or running Net Express, look in the Readme Help to see if it's a known problem. From your Windows **Start** menu, click **Programs**, then **Micro Focus Net Express**, then **Readme**. You can also get to the Readme Help via the contents of the Net Express Help. Information in the Readme Help is listed in the index of the Readme Help, not in the index of the Net Express Help.
- For support, updates, and information on Net Express, use WebSync, a facility in Net Express to access a Micro Focus Web site. To start WebSync, click **WebSync** on the **Help** menu.
- To find Micro Focus's contact addresses and numbers, see the Readme Help.

3 Licensing and Deployment

This chapter explains the licensing implications of deploying your applications for use in a production environment, and provides pointers to information about the technical aspects of deployment.

3.1 Licensing

Your Micro Focus Net Express license allows you to develop and execute applications for your own use. If you wish to distribute any applications created or modified using Net Express then you must purchase a Micro Focus Application Server License for each user. A Net Express Application Server license allows deployment of Net Express functionality to a Windows or Web platform.

Application Server is our name for the software components that provide a run-time environment for your applications. It is provided with Net Express, and is also available separately.

If you are deploying your application on UNIX you will require a Micro Focus Application Server License for Server Express or Object COBOL Developer Suite. Please refer to your Micro Focus Server Express or Object COBOL license agreement.

3.2 When Do You Require an Application Server License?

You need to purchase an Application Server license before any application created or modified using Net Express is executed in a production environment, whether the application is for internal use or resale. Please contact your Micro Focus sales representative for details of the Application Server licenses available.

The only circumstances in which you do not require at least one Application Server license are:

- Where the application user already has a Net Express Development license and is executing the application on the same machine as that used for the Net Express development system
- Where the user already has a single license for Application Server for Net Express for another application that is executed on the same machine

For further details concerning licensing conditions please refer to your End-user Software License Agreement.

3.3 Deploying Your Application

The way in which you build an application for deployment is usually different from the way you build it for development and debugging. For further details of the different methods of building your application please see the Net Express online help. (Click **Help Topics** on the **Help** menu. Then, on the **Contents** tab, click **Development Environment**, **Building an Application**).

For information on deploying your application to a production environment, see the Net Express online help. (Click **Help Topics** on the **Help** menu. Then, on the **Contents** tab, click **Deploying your Application**.

There is also specific guidance on building and deploying Internet applications in the chapter *Deploying Your Application* in your ***Internet Applications*** book. You may not make Application Server files available for download from the Internet.

3.4 Licensing of Additional Software

Micro Focus Net Express includes additional products as part of the development system, some of which are owned by other vendors. If you wish to distribute these additional components as part of your application then a separate additional license may be required according to license agreements of those products.

3.4.1 ODBC Drivers

The Net Express development system includes the DataDirect Technologies ODBC Drivers (DataDirect Technologies was formerly part of MERANT). If you wish to distribute any DataDirect Technologies ODBC Drivers with your application you need to obtain a separate license; please contact your Micro Focus sales representative.

If you are deploying with other ODBC drivers, please contact the appropriate vendor for licensing terms and conditions.

3.4.2 Data Table Grid Control

Net Express includes Data Table, an ActiveX grid control developed by Infragistics (formerly ProtoView). If you wish to develop an application using this control and distribute it you need to obtain a separate license from Infragistics.

3.4.3 Other Third-party Components

Other third party software such as the Microsoft SDK is included with Net Express. For further details please refer to the *Third-Party Software Terms and Conditions* section of your End-user Software License Agreement and contact the appropriate vendor.

4 Start Here for the Tutorials

Welcome to the "Getting Started" tutorials of Net Express!

4.1 Overview

If you've come to this book from the ***Start Here!*** screen in the help, you've probably come straight to this chapter, skipping the earlier chapters. This is the last chapter in Part I. From this chapter, you can go straight into the other Parts, which consist of tutorials. There's one Part for each major area of Net Express. Each Part contains one or more chapters, each of which is a tutorial session.

(You should read the earlier chapters in Part I at some point. They give an overview of what's in Net Express, and lots of practical advice on finding information in the documentation.)

These sessions assume you're familiar with COBOL and with the operating system you're using - Windows 95, Windows 98, Windows 2000, or Windows NT 4.0 - but that you've never seen Net Express or any other Micro Focus software before.

It's essential that you read the following before you start.

4.2 Terminology and Conventions

Windows

In this book, we generally say "Windows" as an abbreviation for Windows 95, Windows 98, Windows 2000, and/or Windows NT 4.0.

Version numbers

Any version numbers in the text on menu items are omitted in the examples.

Mouse buttons

We assume your main mouse button is the left-hand one. If you've configured your mouse to reverse the buttons, please swap the words "left" and "right", as applied to mouse buttons, throughout.

Clicking

The word "click" means a single click of your main (left-hand) mouse button. If you are to double-click or right-click, the instructions say so.

Case

In general in these sample sessions you need not worry about using exactly the combination shown of upper and lower case. Windows is not case sensitive. The few places where case is important are indicated.

Directory and folder

In Windows, the terms directory and folder are used interchangeably. You *enter* a directory, and a directory has *subdirectories*. Equivalently, you *open* a folder, and a folder *contains* other folders. In the Net Express documentation, both types of terminology are used. In this book we mostly use the term "folder".

Drive

In writing the names of folders within Net Express, we will write ***d:*** for the drive letter. Replace this by the letter of the drive where you installed Net Express.

4.3 Tips

Using this book

Your screen may get crowded during these sessions - you'll often have several windows open. If you're displaying this book online in your Web browser, it may frequently be covered by other windows. You may prefer to work from the printed book. However, the online version has many links to help you move quickly through the book. A good compromise is to print out each chapter from your browser, and keep the browser open in a small or minimized window, enlarging it when you need to use the links.

Returning after a link

If you've clicked a link to another chapter, and you want to get back to where you came from, click your browser's **Back** button twice to get back both the text and the Table of Contents.

Cross-references

In these sessions, cross-references you might want to look at immediately are written as links. Cross-references suggesting places to look later are not links.

Windows techniques

Some Windows techniques that are commonly used are described briefly in the appendix *Windows Tips*.

Reloading projects

Successive sessions sometimes use the same project. If you close Net Express between sessions, you will find it useful to know the ways you can reload a project.

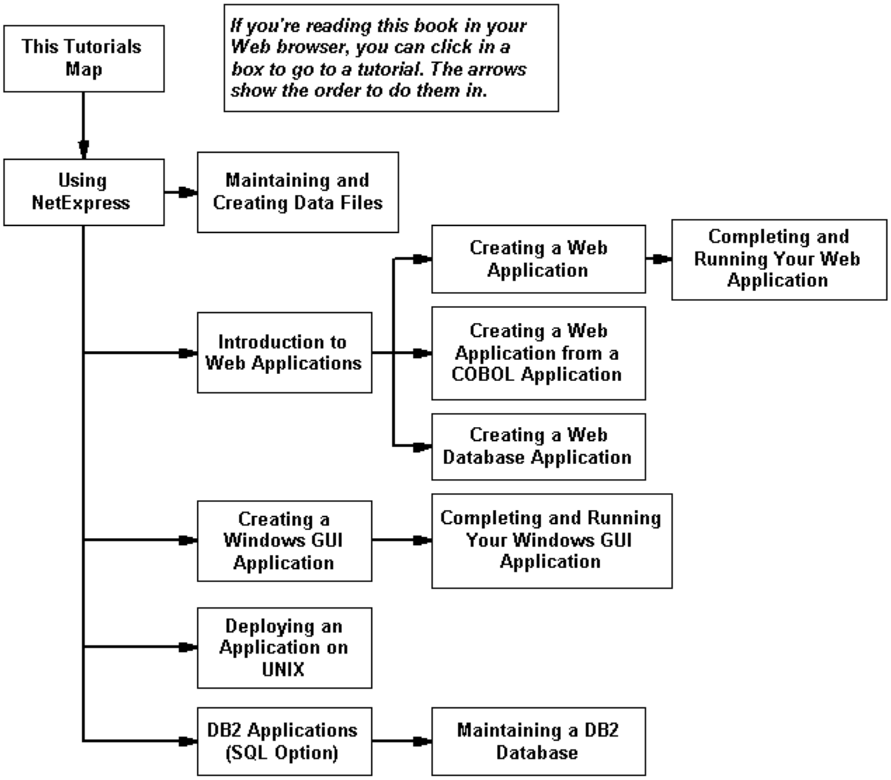
Optionally, when Net Express starts, it loads whatever project was open when you closed Mainframe Express. To make sure this option is on, click **Customize** on the **Options** menu, then click the **Workspace** tab and ensure there is a check mark by **Reload files and last project on startup**. Click **OK** to close the dialog box.

Other ways of opening an existing project are to use **Open** or **Recent Projects**, which are both on the **File** menu.

4.4 Tutorials Map

Each session is one chapter. In the online version of this book, you can get to a chapter by clicking its box below. It’s essential that you do the first session before any of the others. After that, the following tree diagram shows their interdependence - if two tutorials are joined by an arrow you must do the first one before the second.

These "Getting Started" tutorials are intended to give you a quick introduction to the areas they cover, by taking you through some of the most common tasks. Full information on all these areas is in the Help and the online books, as described in the chapter *Finding Information*. The Help and online books are all accessible from your Windows **Start** menu.


This Tutorials Map
If you're reading this book in your Web browser, you can click in a box to go to a tutorial. The arrows show the order to do them in.
Using NetExpress
Maintaining and Creating Data Files
Creating a Web Application
Completing and Running Your Web Application
Introduction to Web Applications
Creating a Web Application from a COBOL Application
Creating a Web Database Application
Creating a Windows GUI Application
Completing and Running Your Windows GUI Application
Deploying an Application on UNIX
DB2 Applications (SQL Option)
Maintaining a DB2 Database

Part 2: Basic Tutorials

This part contains the following chapters:

- Chapter 5, “Using Net Express”
- Chapter 6, “Maintaining and Creating Data Files”

5 Using Net Express

The Integrated Development Environment (IDE) is where you compile, edit and debug all your applications.

This is the first session of the tutorial in the ***Getting Started*** book. You need to work through this before you do any of the others. You need to have read the chapter *Start Here for the Tutorials* before doing this one.

5.1 Overview

The IDE integrates all the tools you need for editing, compiling and debugging (known as "animating") COBOL applications. It contains extensive on-line documentation, both for the tools and for the COBOL language.

The easiest way to learn about the IDE is to complete a few simple tasks, so this session takes you through loading a project, compiling it, running it, and animating (debugging) it - that is, stepping through the source code.

5.1.1 Projects and Project Folders

A project is a file detailing all the files in your application, and how they should be compiled. A project is very easy to create, and makes compiling extremely quick and easy. You should create a project for every application, even the simplest. On disk, a project is recognizable by the extension **.app** at the end of its name. However, you never need to look at a project directly because, like most things in Net Express, you create and maintain it entirely using the IDE.

The folder where you keep the project for an application is called its project folder. You could keep the application’s other files anywhere, since the project contains pointers to them, but it’s usually convenient to keep them all in the project folder.

When you install Net Express, Setup creates, within the system folders containing Net Express, a folder called by default **Net Express\Base\Workarea**. This folder is intended as your work area, and we suggest that you put all your project folders within it. Because the first part of the directory path can be user-defined, we do not spell it out in full in this book: you need to be aware, therefore, that **Net Express** is not the top level folder.

5.1.2 Demonstration Applications

Net Express includes many demonstration applications. Some are used in the tutorials in this ***Getting Started*** book. Others are used in more advanced tutorials accessed via the online help. Most are in folders within **Net Express\Base\Demo**, though a few for optional extensions are in other folders - for example the demos for Dialog System are in folders within **Net Express\DialogSystem\Demo**.

If, when you installed, you specified a different name for the folder **Net Express**, remember to use it in these sessions.

If you get partway through a tutorial and decide you want to start again, you can return the supplied files of demo to their initial state by clicking the demo’s **Initialize** button in the full list of tutorials in the Net Express Help. You must have the CD-ROM or network connection from which you installed Net Express available. To get to the list, click **Start -> Programs -> Micro Focus Net Express -> Help**, and then in the Help Contents double-click **Start Here! -> Tutorials, introductions and demonstrations**, then double-click the subject area you want.

This does not delete files or folders created during the demo, but reinstalls the original versions of the supplied files.

Note: If you try to initialize a tutorial demo when a wizard is running in the IDE, you can get a protection violation. Before copying or opening demo files in the IDE, you should complete or cancel any wizard you have active.

The demo application used in this session is called Locking. A project file is supplied with it, and in this session you load the project and get the application running.

5.2 Sample Session

In this sample session you:

- Start the IDE
- Load a project
- Build a project
- Run COBOL code
- Animate COBOL code
- Set IDE options
- End animation

5.2.1 Starting the IDE

If you've entered the IDE by clicking the **Run** button at the end of installing, you may still have visible the main screen of the installation utility - the one with the **Install**, **Installation Notes**, and **Exit** buttons - though it may be hidden behind the IDE. You can close it by clicking its ☒ button, now or at any time - it won't interfere with this session

If not, then start the IDE as follows (this is how you will always start the IDE in future):

1. From the Windows **Start** menu, click **Programs**, then **Micro Focus Net Express**, then **Net Express**.

   You may get a dialog box entitled Micro Focus Protection System, warning you that your license expires in a few days. In these sessions we will ignore this warning, but later you should load Net Express again and click the **Help** button on this screen for details of how to get a full license.

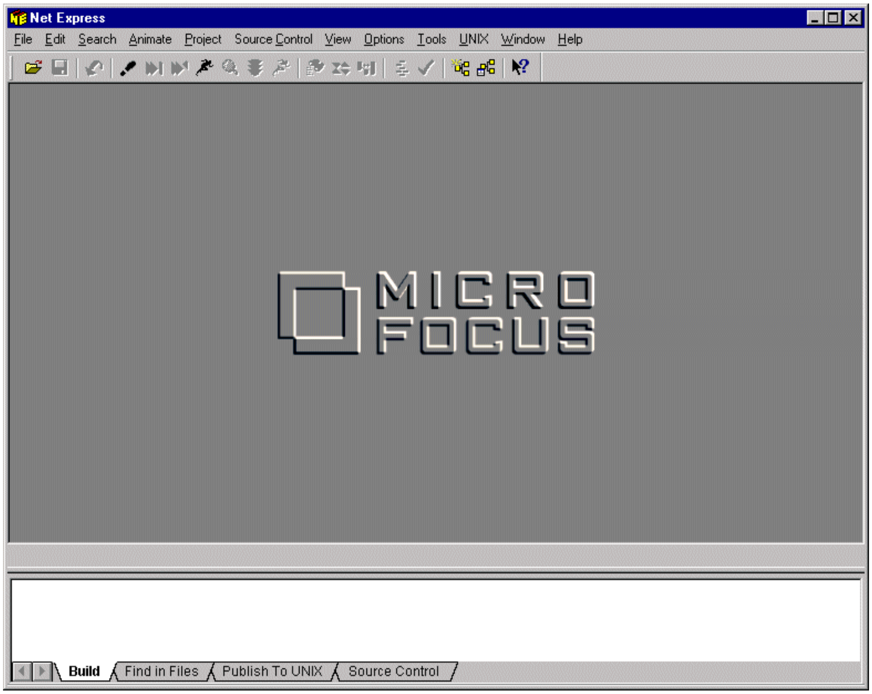
2 If you got the license warning, click **OK** to ignore it.

When you first load the IDE you get a Welcome screen as well as the IDE window. It has a check box you can set to show whether you want to get it every time. It enables you to choose whether to look at the Help file, or go straight into Net Express.

3 Click **Continue** on the Welcome screen.

The Welcome screen closes. You now have the IDE on your screen. It is shown in Figure 5-1.

Figure 5-1. The Integrated Development Environment (IDE)


Net Express
File Edit Search Animate Project Source Control View Options Tools UNIX Window Help
MICRO FOCUS
Build
Find in Files
Publish To UNIX
Source Control

The large pane is where various windows such as project windows and editing windows will be opened. The pane below it is the Output window, where messages from the IDE and compiler are displayed. It has several tabs, of which the one you will use most is marked **Build**. When the **Build** tab is highlighted (that is, white), this window displays messages to show the progress of a build.

You can change the size and shape of the IDE by dragging the edges and corners. You can detach some panes and position them elsewhere in the IDE or separately on your screen - this is called docking or undocking them. See the appendix *Windows Tips* if you want to do this.

In doing the sessions in this book, be aware that if anyone has used the product on your computer before you, they might have moved panes from their standard positions.

5.2.2 Loading a Project

To load the Locking project:

1 Click **Open** on the **File** menu.

This opens the Open dialog box. If this is the first time you have started Net Express since installing, this displays the contents of **Net Express\Base\Demo**. If it doesn’t, change it to that directory.

2 Go into subdirectory **Locking**, select the project **locking.app** and click **Open**. (The extension **.app** and the icon are standard for a project file. If your Windows settings are such that extensions aren’t shown, you can identify this file by its icon.)

The IDE opens a project window, displaying the files in the project. The left-hand pane is a tree view showing which files are created from which. The right-hand pane is a list of the files. You can drag the borders of the window; for example, you might want to make the right-hand pane wider to see more details of the files.

3 Right-click in the right-hand pane to get a popup menu. Ensure **Show only source files** on this menu has no checkmark by it. If it has, click it to make this pane show all files, not just files that can be compiled.

The **.cbl** files are the COBOL source files.

4 Double-click on any **.cbl** file.

This opens a text window showing you the source code

5 Click **Close** on the **File** menu.

This closes the text window, leaving the project window open. Alternatively you can close a window by clicking the Close icon ☒ at its top right-hand corner.

The **.int** files are executable files in intermediate code format (a Micro Focus executable format).

The IDE can also produce industry-standard **.exe** and **.dll** files, but **.int** code has the advantage of not needing linking - this makes it very quick to recompile individual programs when you are debugging a new application.

Files don’t have to exist yet to be shown in the project. A project is analogous to a makefile or batch file that will guide the build. Since you haven’t built the project yet, none of the **.int** files exist.

5.2.3 Building a Project

Building a project means compiling the files to an executable format. The format of the files created depends on the type of build and the options selected for the project type. Each project has two standard build types available, Debug and Release, and you can create your own build types as well.

To build the project:

1 Click **Rebuild** on the **Project** menu, or click ☰ on the toolbar.

This rebuilds all the files that have changed since the last time the project was built. Since you haven’t built the project before, all the files are rebuilt. (The **Rebuild All** function on the same menu rebuilds all files in the project, whether they need it or not.)

This one function carries out the entire build. The correct compiler or translator is automatically called for each source file. In Net Express, the term "compilation" is generally used for any compilation, translation, conversion or preprocessing. Also, the term "source file" is used for any file that is the input to such a compilation.

Notice the ☰ button on the toolbar. The same symbol appears by **Rebuild** on the **Project** menu. Many commonly used functions on the menus have equivalent buttons on the toolbar. It’s often quicker to use these. In these sessions we’ll sometimes use the menus and sometimes the toolbar. If you leave the mouse pointer over a button on the toolbar for a moment, a brief explanation of that button appears. These explanations are called tooltips.

Messages in the Output window keep you informed about the progress of the build. The build is over when you see the message "Rebuild complete".

5.2.4 Running the Application

You use the **Animate** menu for both running and debugging. To run the application without debugging:

1 Click **Run** on the **Animate** menu.

The Start Animating dialog box appears. You use it to specify where execution starts.

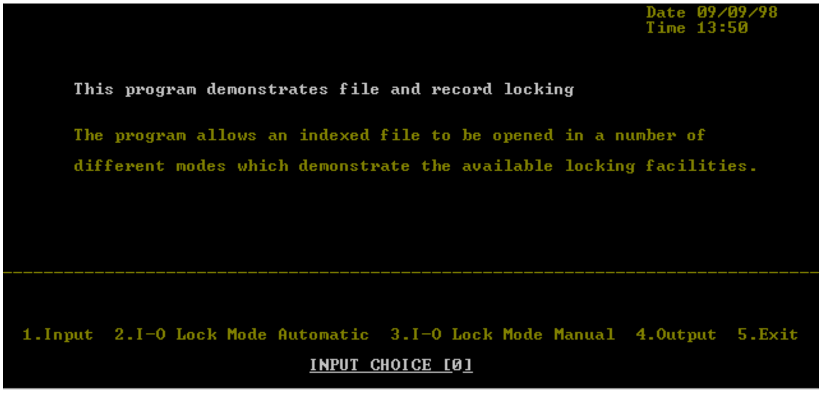
This dialog box appears for both running and debugging. In the IDE, running and debugging use the same underlying mechanism. In running an application within the IDE you are really using the debugging engine with the debugging features switched off. Consequently, running is controlled via the **Animate** menu, and observations we make about debugging will generally apply to running as well.

The 🏃 button on the toolbar is equivalent to Run on the Debug menu.

2 Click **OK** to confirm that you want to start the Locking program.

This is a character-based application, so a new window appears at the bottom of the IDE. This window always appears for an application that does character displays. It is called the Application Output window, or character window. It is shown in Figure 5-2. You might have to drag the borders within the IDE to make this window bigger so you can see the whole display. If it is not big enough to show the whole display, there is a slider at its right-hand side.

Figure 5-2. The Application Output Window



```
                                                          Date 09/09/98
                                                          Time 13:50


      This program demonstrates file and record locking

      The program allows an indexed file to be opened in a number of

      different modes which demonstrate the available locking facilities.





--------------------------------------------------------------------------


 1.Input  2.I-O Lock Mode Automatic  3.I-O Lock Mode Manual  4.Output  5.Exit
                           INPUT CHOICE [0]
```

In this session we are just showing how to build and run an application, so we will not use this application in any detail.

3 Click in the Application Output window to select it, then type "5" and press **Enter**.

"5" is the code that tells this application to finish. The Application Output window remains so that you can see the application's final display.

You can check it has finished by clicking the **Animate** menu. The **Start Animating** and **Run** functions are not grayed out, as they are while an application is running. Click elsewhere on your screen to close the menu again.

(This Locking application is also used in one of the more advanced tutorials accessed via the online help, to demonstrate different ways of locking a file.)

4 Right-click anywhere in the Application Output window, and click **Hide** on the popup menu.

This closes the Application Output window.

5.2.5 Debugging an Application

The IDE gives you an intuitive and graphical way to trace the execution of your code. It provides an extensive set of debugging (known as "animating") facilities. It is useful if the application isn’t doing what you expect, or if you want to get familiar with an unfamiliar application. We will animate this application to see a few of these features:

1 Click **Start Animating** on the **Animate** menu.

The Start Animating dialog box appears.

2 Click **OK** to confirm that you want to debug the Locking program.

The IDE opens a text window showing the source code for **locking.cbl**. The first statement is highlighted, ready for execution.

3 Click **Step** on the **Animate** menu, or click ✎ on the toolbar.

Step executes the next statement. This statement displays spaces on the text console, so it opens the Application Output window for this application. Windows might be moved about to make room for the Application Output window. You might have to drag borders within the IDE so you can see the source code again.

4 Step the next statement as well (DISPLAY LOCKING01-00).

5 Click **Run Thru** on the **Animate** menu.

Run Thru executes all the code for a PERFORM or CALL statement in a single step.

6 Right-click on the statement EVALUATE CHOICE (second statement in the paragraph RE-ENTER-CHOICE) and click **Run to Cursor** on the popup menu. (You might have to make the text window bigger to see more of the source.)

Run to Cursor gives you a quick way of executing to a particular statement without setting a breakpoint. Execution does not reach the EVALUATE statement yet though - part of the code you have executed is an ACCEPT statement, and the application pauses there waiting for input.

7 Click in the Application Output window to select it, then type "5" and press **Enter**.

The statement EVALUATE CHOICE is highlighted ready for execution.

8 Double-click on any reference to the data item CHOICE to see its value.

This opens the Examine List, where you can see that its value is 5. From this dialog box you could create an individual monitor window for a data item, or change its value, or add it to the Watch list (which opens a window at the bottom of the screen).

9 Click the ☒ button at the top right-hand corner of the Examine List.

5.2.6 Setting IDE Options

Let's look at some ways you can configure the behavior of the IDE:

1 Click **Animate** on the **Options** menu.

2 On the dialog box, make sure the checkbox by **Show tool tips to examine data** contains a checkmark (click in it if it hasn't), and click **OK**.

3 Put the mouse pointer on the data item CHOICE, without clicking, and wait a moment.

With this option set, putting the mouse pointer on a data item displays the item's value. This is quicker than using the Examine List, but you may not want this to happen every time you put the mouse on a data item. It's up to you whether you leave this option set or not.

4 Put the mouse pointer on any line in the source and, with the button held down, drag it a little way along the line.

The whole line is selected. You often use this when editing. However, you may not want the whole line selected.

5 Click anywhere in the source to deselect the line.

6 Click **Edit** on the **Options** menu.

7 On the dialog box, click the **Blocks/Clipboard** tab, then make sure the checkbox by **Select column blocks when dragging, except when in the prefix area** contains a checkmark (click in it if it hasn't), and click **OK**.

8 Put the mouse pointer on any line in the source and, with the mouse button held down, drag it a little way along the line.

Only the part of the line you dragged the mouse along is selected. Again, it's up to you whether you leave this option set or not.

There are many other options which you can set. We suggest you explore them when you are more familiar with using the IDE.

5.2.7 Ending Animation

1 Click 🏃 to run the rest of the application.

Execution halts on the STOP RUN statement at the end of the application.

2 Click **OK** on the "Stop Run encountered" message and then click **Stop Animating** on the **Animate** menu.

5.2.8 Using Context Help

Let's take a moment here to look at one way you can get help whenever you're using Net Express.

1 Click ↖? on the toolbar, and then click anywhere in the project window.

A popup appears telling you what the project window is for. Generally, whenever you see the Context Help button, normally ? or ↖?, you can use it to get quick context-sensitive help in this way. Of course, many screens also have a **Help** menu or **Help** button which takes you into the main help.

2 Press any key or click the mouse anywhere to make the popup disappear.

5.3 Before Continuing

Close the project, by clicking in it to select it and clicking **Close** on the **File** menu, or by clicking its ☒ button. Closing a project window closes all its dependent windows - in this case, the text window displaying the source.

You can't close the standard windows like the Application Output window using **Close** on the **File** menu, and they do not have ☒ buttons (unless you've undocked them). Instead, right-click on the Application Output window and click **Hide** on the popup menu. Alternatively, click **Dockable Windows** on the **View** menu, then click by **Application Output** on the dialog box to delete the check mark, and click **Close** to close the dialog box. You hide these windows rather than close them - if you open them again, they are still displaying what was there before.

If you're planning to go straight on to another session, you can keep Net Express open. Otherwise, either click **Exit** on the **File** menu, or click the IDE's ☒ button.

Return to the *Tutorials Map* in the chapter *Start Here for the Tutorials* and choose which session to go on to next, depending on your interests.

6 Maintaining and Creating Data Files

You use the Data Tools in Net Express to convert, browse, edit and create data files used by an application.

You need to have read the chapter *Start Here for the Tutorials* and worked through the first session, *Using Net Express*, before you do this session.

6.1 Overview

Using the Data Tools, you can examine data files to see how an application has updated them, or create and edit a file to provide test data for an application. Files can be in any COBOL format and you can view them at both the record and field levels. You can convert between formats and character sets.

The demo application used in this session consists of a data file and the COBOL source program that maintains it. The data file is a variable length sequential file, containing details of staff in three record types: employee, manager and executive. In this session you convert it to a variable length indexed sequential file, and then view it in various ways, formatted and unformatted, using Data File Editor.

6.2 Preparation

If you have closed Net Express, open it as before. If any project window or other windows are open, close them.

To save you creating a project, we have supplied one - click **Open** on the **File** menu, and open **Net Express\Base\Demo\Dtoldemo\dtoldemo.app**.

You'll see the **.cbl** and **.dat** files in the project window.

6.3 Sample Session

In this session you:

- *Convert a file*
- *View a data file unformatted*
- *Create a record layout file*
- *Create the default record layout*
- *Create the conditional record layouts*
- *Save the record layout file*
- *View a data file formatted*
- *View files in hexadecimal*
- *Print a data file*
- *Edit multiple data files*
- *Create a new data file*
- *Converting a File's Character Set*
- *Viewing the EBCDIC File*

6.3.1 Converting a File

To make an indexed sequential copy of the supplied sequential file:

1. Click **Data Tools** on the **Tools** menu, then click **Convert**.
2. To supply the input file details, click on the **Browse** button and use the **Open** dialog box to select the file **datavseq.dat** (the dialog box should open at the right folder, **Net Express\Base\Demo\Dtoldemo**).

   The details for the the input file and the output file are initialized with the information read from the file header for **datavseq.dat**.

3 In the lower part of the dialog box, enter the following details for the output file (leave the remaining fields unchanged):

Filename	**Net Express\Base\Demo\Dtoldemo\staff.dat**
Format	Micro Focus
Organization	Indexed Sequential

4 Click **Define Keys**, then click **Insert Key.**

The field to be used as the key is at the beginning of the record, and is seven bytes long.

5 Set **Key Offset** to **0** and **Key Length** to **7**, then click **OK.**

6 Click **Convert**.

7 Click **OK** on the message saying the data file conversion is complete, then click **OK** on the message reporting the results of the conversion.

The Data File Convert dialog box reappears so you can convert more files.

8 Click **Cancel**.

9 Look in the project window. Net Express has added the new file **staff.dat** to the project automatically.

6.3.2 Viewing a Data File Unformatted

To view the file you just created:

1 From the project window, double click on the file **staff.dat**.

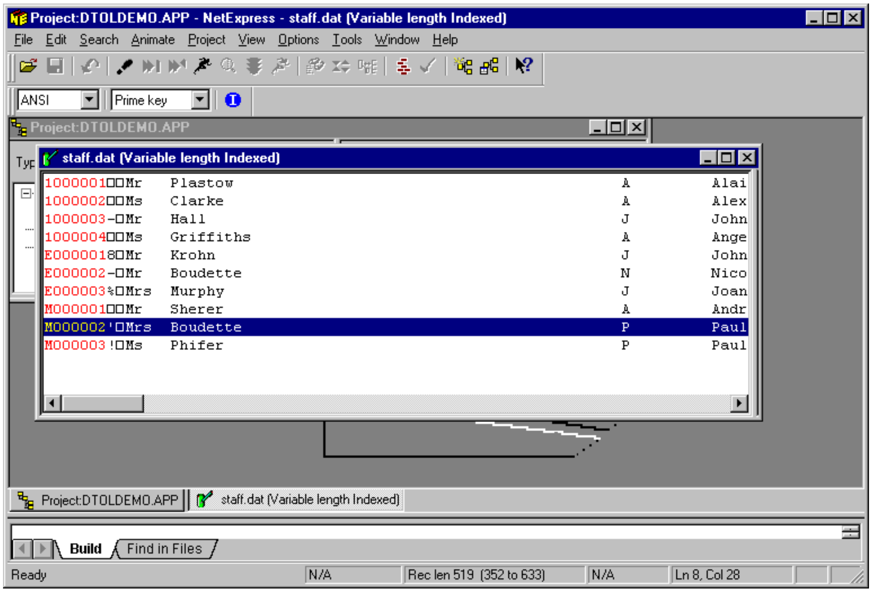
A message box appears, explaining that all edits to indexed and relative file types are applied immediately.

2 To prevent this message appearing again, click in the check box to add a check mark against the prompt **Do not show this message again**.

3 Click **OK**.

A Data File Editor window appears displaying the contents of **staff.dat**, as shown in Figure 6-1.

Figure 6-1. The IDE with a Data File Editor Window


Project:DTOLDEMO.APP - NetExpress - staff.dat (Variable length Indexed)
File Edit Search Animate Project View Options Tools Window Help
ANSI
Prime key
Project:DTOLDEMO.APP
Typ
staff.dat (Variable length Indexed)
Project:DTOLDEMO.APP
staff.dat (Variable length Indexed)
Build
Find in Files
Ready
N/A
Rec len 519 (352 to 633)
N/A
Ln 8, Col 28

```
1000001□□Mr    Plastow                                   A         Alai
1000002□□Ms    Clarke                                    A         Alex
1000003-□Mr    Hall                                      J         John
1000004□□Ms    Griffiths                                 A         Ange
E0000018□Mr    Krohn                                     J         John
E000002-□Mr    Boudette                                  N         Nico
E000003%□Mrs   Murphy                                    J         Joan
M000001□□Mr    Sherer                                    A         Andr
M000002'□Mrs   Boudette                                  P         Paul
M000003!□Ms    Phifer                                    P         Paul
```

Each record is shown as one line. Because Data File Editor does not know the lengths of the fields in the record, each line is simply a continuous line of text - the records are unformatted. Non-character fields such as COMP items generally display as non-standard characters or, if they don’t correspond to any printable character, as □.

You can edit the file by typing over the existing data. There are many editing facilities available. For example, the **Data Tools** option on the **Search** menu offers four ways of searching a data file:

- **Data File Find and Replace**
- **Find On Indexed Key** This option is available for indexed files only. Depending on whether the Data File Editor window is open and, if so, the current key of reference, this option name can also read as **Find on prime key** and **Find on alternate key number *n***.
- **Locate Field** Use this option to find a field within the current record. This option is only available when you have a formatted view available for the data file. To obtain a formatted view involves

loading a record layout file which we shall do later on in this tutorial.

The status line at the bottom of the IDE shows various statistics, depending on the file details:

- Relative record number (relative files only)
- Length of the selected record, followed by (minimum length, maximum length) if variable length.
- Number of records (sequential files only)
- Cursor position - line number and column position, both starting at 0.

The Data File Editor window may have a right-hand pane visible, displaying "A record layout file must be loaded before using this window". We'll see in a later section what this is for.

You can see the file's properties in full:

1. Right-click in the Data File Editor window.
2. Click **File Information** on the popup menu.

   The File Information window displays information such as the file format and organization.
3. To view more information about the file, click on the **Keys** and **General** tabs.
4. When you have finished, click on **Cancel** to close the File Information window.
5. Leave the Data File Editor window open to use later in this tutorial.

6.3.3 Creating a Record Layout File

Data File Editor can display file contents formatted, if it knows the layout of the fields. You create a record layout file from the Data Division of a COBOL program that uses the file. You must have compiled the program to create the necessary information.

Our data file **staff.dat** has three record types: employee, manager, and executive. You have to tell Data File Editor the layout of each type of record, and how to identify the type. You define one type to be the default. The others are called conditional.

To create a record layout file:

1. Select the project window.
2. Right-click **dfdstaff.cbl** in the left-hand pane of the project window, and click **Compile** on the popup menu.
3. Right-click **dfdstaff.cbl** again and click **Create Record Layout** on the popup menu.

   A Record Layout Editor window opens. This window displays a tree view of the data division of **dfdstaff.cbl**.

6.3.4 Creating the Default Record Layout

To create the default record layout:

1. Expand the **FD MF-FILE** entry by clicking its "+".
2. Right-click **01 EMPLOYEE-REC**, then click **New Record Layout** on the popup menu.

   Default Layout should be set automatically as you have not yet defined a default layout.
3. If it is not, click **Default Layout** to set **EMPLOYEE-REC** as the default record layout.
4. Click **Next**, then click **Finish**.

   A folder for **EMPLOYEE-REC-DEFAULT** appears in the right-hand pane.

6.3.5 Creating the Conditional Record Layouts

To create a conditional record layout:

1 Right-click **01 MANAGER-REC**, then click **New Record Layout** on the popup menu.

Conditional Layout is set automatically as you have already defined a default layout.

2 Click **Next**, then click **Finish**, to add **MANAGER-REC** as a conditional layout.

A folder for **MANAGER-REC** appears in the right-hand pane.

You now need to specify the field and condition that identify the record type.

3 In the right-hand pane, expand the **MANAGER-REC** folder by clicking its "+", then expand the **01 MANAGER-REC** entry by clicking its "+", then expand the **02 MN-CODE** entry by clicking its "+".

4 In the right-hand pane, right-click **03 MN-POSITION**, then click **Properties** on the popup menu.

The **Field Properties** dialog box appears.

5 Set **Condition** to **IS = TO**, by selecting it from the pulldown list.

6 Type **M** (upper case) in the left-hand **Condition** field.

M indicates a Manager record type. The two characters below the dotted line are the hexadecimal equivalent, arranged vertically. They should be "4" and "D", since M in ANSI is hex 4D.

7 Click **OK**.

A small red **IF** appears by **03 MN-POSITION** in the right-hand pane.

8 Create a conditional record layout for the Executive record type, by repeating this section replacing **MANAGER** by **EXECUTIVE**, **MN-POSITION** by **EX-POSITION**, and **M** by **E**.

6.3.6 Saving the Record Layout File

To save the record layout file:

1. Click **Save** on the File menu.

   The Save As dialog box appears. The folder defaults to **Net Express\Base\Demo\Dtoldemo**.

   The filename defaults to **dfdstaff.str**, which is the COBOL program name with a **.str** extension.

   When Data File Editor loads a data file, it looks in the same folder for a record layout file with the same name as the data file but extension **.str**. Our data file is named **staff.dat**. (Remember, we used the Data File Converter to convert **datavseq.dat** to **staff.dat**.)

2. Change the **File Name** field for the record layout file to **staff.str**. Then click **Save**.

3. Close the Record Layout Editor window by clicking **Close** on the **File** menu.

6.3.7 Viewing a Data File Formatted

When you open a data file, if a record layout file with the same name (apart from the extension) is present in the folder, that record layout file is used automatically. To apply a record layout file to a data file that is already open, or whose name does not match that of the record layout file, you must load it explicitly. This is called creating an association. Once you've created an association, that record layout file will be always used for that data file even if a record layout file with a name matching the data file is also present.

To load a record layout file:

1. Click in the Data File Editor window that you left open earlier, to bring it to the front.

2. Click **Data Tools** on the **File** menu, then click **Load Record Layouts**.

3 Select **staff.str** and click **Open**.

A new right-hand pane appears. The left-hand pane contains the unformatted view of multiple records, one per line, that you had before.

In the right-hand pane, Data File Editor has applied the formatting information from the record layout file. This pane shows a formatted view of a single record. The record layout name is at the top of the pane. At the top right are two navigation controls: the small up-arrow takes you to the previous record and the small down-arrow takes you to the next record.

You can edit a record by clicking in the **Value** column and typing over the existing data. The editing facilities mentioned earlier for the unformatted view are available in the formatted view too.

4 Click in the left-hand pane, and move the cursor over the records in the unformatted view

Notice how the formatted view alters, matching the record types as you move through the file.

6.3.8 Looking Up Information about a Record Layout File

You can find out the record layout file associated with a data file and review the conditions that you set up for the different conditional layouts:

1 Right-click in the Data File Editor window, then click **File Information** on the popup menu.

The File Information window opens. Because a record layout file is now associated with the data file **staff.dat**, the File Information window has a new tab **Record Layouts**.

2 Click the **Record Layouts** tab.

This window displays information about the record layout file **staff.str** associated with **staff.dat**. The top of the window displays the name and location of the record layout file.

3 Under **Layout Name**, click on **MANAGER-REC**.

The Conditions box underneath shows that, for Manager records, the field **MN-POSITION** equals **M**.

4 Under **Layout Name**, click on **EXECUTIVE-REC**.

The Conditions box shows that, for Executive records, the field **EX-POSITION** equals **E**.

5 Click **Cancel** to close the File Information window.

6.3.9 Viewing Files in Hexadecimal

You can use Data File Editor to view and edit your data in hexadecimal:

1 Right-click in the Data File Editor window (either pane) and click **Show Hex** on the popup menu.

New panes appear at the bottom of the window. The left-hand pane shows the selected record from the top left-hand pane, with the hex beneath. For elementary field items only, the right-hand pane shows the selected field from the top right-hand pane, with the hex beneath. Otherwise, the right hand pane displays a text description such as **Group item**.

Each pair of hex digits is arranged vertically. For example, when you edit ANSI data, an **M** is shown in hex as "4" with a "D" beneath. If you can't see both lines of hex, make the window bigger by dragging its bottom edge downward.

2 Right-click within the Data File Editor window and click **Show Hex** on the popup menu.

The check mark by this menu item disappears, and the hex panes disappear.

6.3.10 Printing a Data File

You can print a data file from Data File Editor using the Print function, or use the Print Preview function to display it on screen as it would appear when printed. You can choose the formatted or unformatted view. You can choose whether to print the current record, all records or partial records.

To preview a print of the unformatted view of a data file:

1. Click in the left-hand pane of the Data File Editor window, then click **Print Setup** on the **File** menu.

2. Set **Orientation** to **Landscape**, then click **OK**.

3. Select **Print Preview** on the **File** menu.

4. Check the checkbox beside **Header Text** and specify the heading text as **Print of Data File STAFF.DAT**.

5. Put check marks by:

   - Page numbers
   - Ruler
   - Record Numbers
   - Hexadecimal Values

   and select:

   - All Records
   - Whole Record

6. Click **Preview**.

   A Preview window appears. You can use the buttons on the toolbar to zoom in and out, move around the window, and print.

7. Click **Close** on the buttonbar of the Preview window.

6.3.11 Editing Multiple Data Files

As with many of the editors in Net Express, you can use Data File Editor to edit several files at once, choosing various ways to position their windows within the IDE.

To edit two data files, putting their windows one above the other:

1. From the project window, double click on the file **datavseq.dat**.

   A new Data File Editor window opens for **datavseq.dat**. Only the unformatted pane appears, as no record layout is associated with this file.

2. Minimize the project window by clicking its 🗕 button.

3. Click **Tile Horizontally** on the **Window** menu.

   You can now see the Data File Editor windows for both **datavseq.dat** and **staff.dat**

4. Toggle between the windows by clicking alternatively on the tags for **datavseq.dat** and **staff.dat** near the bottom of the Net Express window.

5. Close both Data File Editor windows. Restore the project window to its normal size (click its 🗗 button).

6.3.12 Creating a New Data File

To create a fixed length sequential file:

1. Select **New** on the **File** menu, then select **Data File** and click **OK**.

   The **Create file** dialog box appears.

2. Specify **Net Express\Base\Demo\Dtoldemo\newfile.dat** in the **Filename** field, and **20** in the **Maximum Length** field, accept the defaults for the other fields, and click **Create**.

   Since a fixed length sequential does not have a file header, Net Express prompts you to save the file header details you have just entered in a profile file. Creating a profile saves you having to enter the file header details next time you open the data file.

3. Click **Yes**.

   Net Express saves the profile in a profile file in the same directory as the data file. Profile files have the same filename as the data file with a **.pro** extension. Therefore, the profile file for this new data file is named **newfile.pro**.

   A Data File Editor window appears. It displays "File Empty".

Let's add and delete a few records.

4 Right-click in the Data File Editor window, then click **Insert Record After** on the popup menu.

The cursor is positioned at the start of the file.

5 Type **abc**.

6 Right-click in the Data File Editor window, then click **Repeat Record** on the popup menu.

A copy of the above record is added to the file.

7 Right-click in the Data File Editor window, then click **Delete Record** on the popup menu, and click **Yes** on the Delete warning message box.

The new record is deleted from the file and the cursor is positioned on the previous record.

8 Click **Save** on the **File** menu.

9 Close the Data File Editor window

6.3.13 Converting a File's Character Set

To make an EBCDIC copy of the supplied sequential file:

1 Click **Data Tools** on the **Tools** menu, then click **Convert**.

2 To supply the input file details, click on the **Browse** button and use the **Open** dialog box to select the file **datavseq.dat**.

The details for the the input file and the output file are initialized with the information read from the file header for **datavseq.dat**.

3 Click in the **Convert character set** check box to put a check mark there.

The character set for the output file, shown in the lower half of the dialog box, changes from ANSI to EBCDIC.

4 Click in the **Records contain non-text data items** check box.

The converter needs to know where the non-text data items (items such as COMP fields) are, so as not to convert them. So you need to specify a record layout file. You have already created one describing **datavseq.dat**.

5 Click **Select layout for conversion**. Use the **Browse** button on the **Select layout for conversion** dialog box to select **staff.str**, and click **OK**.

6 In the lower half of the dialog box, enter **ebcvseq.dat** as the name of the new file, then click **Convert**.

7 Click **OK** on the message saying the conversion is complete, then click **Cancel**.

The new file **ebcvseq.dat** has been added to the project.

6.3.14 Viewing the EBCDIC File

When Data File Editor is open, the IDE has an extra toolbar, which includes a dropdown list to switch the Data File Editor window between ANSI and EBCDIC. To display the EBCDIC file you just created:

1 Double-click **ebcvseq.dat** in the project window.

Garbage is displayed because Data File Editor is expecting ANSI.

2 On the toolbar, use the dropdown list to change the character set from ANSI to EBCDIC.

The file is displayed correctly.

This field on the toolbar affects only the current display in the Data File Editor window.

To look at the file in formatted view, you cannot use the record layout file you created earlier because the comparisons to determine record type would use ANSI. You need to change Data File Editor's default character set to EBCDIC and create a new record layout file.

3 Click **Data Tools** on the **Options** menu, then click the **General** tab.

4 Using the dropdown list, change the **Character Set** field to EBCDIC. Click **OK**.

This makes Data File Editor read and write all files in EBCDIC. You can now if you wish create and use a new record layout file, in the same way as you did earlier in the sections *Creating a Record Layout File* through *Viewing a Data File Formatted*.

5 When you've finished, close the Data File Editor window.

6 Click **Data Tools** on the **Options** menu, then change the character set back to ANSI, and click OK.

If you normally use the ANSI character set, you need to keep this option normally set to ANSI.

6.4 Before Continuing

Close the project. If you want to take a break before going on to the next session, you can close Net Express.

Return to the *Tutorials Map* in the chapter *Start Here for the Tutorials* and choose which session to go on to next, depending on your interests.

Part 3: Web Development Tutorials

This part contains the following chapters:

- Chapter 7, "Introduction to Web Applications"
- Chapter 8, "Creating a Web Application"
- Chapter 9, "Completing and Running Your Web Application"
- Chapter 10, "Creating a Web Application from a COBOL Application"
- Chapter 11, "Creating a Web Database Application"

Getting Started

7 Introduction to Web Applications

This chapter introduces basic concepts relevant to all the tutorials on creating Web applications.

You need to have read the chapter *Start Here for the Tutorials* and worked through the first session, *Using Net Express*, before you read this chapter.

Be sure you understand the basics of the World Wide Web before continuing. They're explained in the appendix *Web Applications*. Many terms used in this chapter and the sessions in this Part are explained there.

In Net Express, you can create a Web application in three ways:

- From scratch - you need to create HTML forms to display in users' browsers, together with a server-side program to accept data, process it, and display the results.
- From an existing COBOL program - you need to create HTML forms, together with a server-side program to accept data, pass it to the existing program to process, and display the results.
- From an existing database - you need to create HTML forms, together with a server-side program to accept data, look up information in the database, and display the results.

For all three cases you use Internet Application Wizard. In the first case you first use HTML Page Wizard to design the form yourself. After that, the three ways of using Internet Application Wizard are very similar, in that you give it the one existing item - form, program, or database - and it generates the rest of the application for you.

Forms and form-handling code generated in this way are based on standard templates supplied with Net Express. You can edit the forms and form-handling code later if you want to customize them to your own exact requirements.

The sessions in this Part take you through these three ways of using Internet Application Wizard. You can use any Web browser to test your applications in these sessions.

7.1 Solo

You can use any Web server software to test your applications. In these sessions we use Solo, the Web server software included in Net Express. It is designed especially for testing Internet applications developed using Net Express. It is not designed for use as a production Web server. Solo needs no configuration or setup. (If you want to see how it is automatically configured, see the appendix *Configuring Your Web Server*.)

7.2 Server-side Programs

In a server-side program created using Net Express, the input/output of the form is done by two Micro Focus extensions to COBOL: an extended ACCEPT statement, and an EXEC HTML statement (known as Embedded HTML) which outputs HTML written within the statement. The form has to include a "Submit" button; when the end-user clicks this, the program runs on the Web server. The ACCEPT statement inputs data from the form.

Micro Focus COBOL also has an extended DISPLAY statement, but the EXEC HTML statement is more powerful.

7.2.1 Structure

A server-side program typically consists of little more than:

1 The ACCEPT statement to input the data from the input form.

2 Code to move the data from the data items associated with the input form to your work areas.

3 The business logic you've inserted.

4 Code to move the results from your work areas to the data items associated with the form.

5 The EXEC HTML statement(s) to create the output form(s).

Notice that:

- There's no main loop. Each time the end-user submits the input form, the program runs again.
- The ACCEPT statement doesn't cause the form to be read; nor does it wait for it. The Web browser sends the filled-in form to the program when the end-user clicks the Submit button. The ACCEPT takes the data if it's there waiting; otherwise it does nothing.
- The EXEC HTML statement doesn't output the form, it just creates it. The Web server sends the form to the browser when the program finishes.

7.2.2 Symmetric and Asymmetric

An application can use the same form for both input and output, or use separate forms.

An application with only one form is called symmetric. Typically, it is started by the end-user running the executable. You'd probably create a Web page with a link to the executable. When the end-user clicks this, the program runs once, displaying the form on the browser. The end-user fills in the form, clicks the Submit button, and the program runs again to process the data and display the results. The form is then ready for the end-user to use again.

An application with separate input and output forms is called asymmetric. Typically, it is started by the end-user loading the input form. You'd probably create a Web page with a link to this form. When the end-user clicks this, the form is loaded. The end-user fills in the form, clicks the Submit button, and the program runs to process the data and display the output form with the results. To enter the next query, the end-user loads the input form again. (Though if the output form contains all the input fields that the input form does, the user can get away with using it for input rather than bothering to reload the input form - the program won’t know the difference, as it knows only that it’s receiving data from a browser.)

If an application uses more than one input form, you create a separate server-side program for each one.

7.3 Before Continuing

Return to the *Tutorials Map* in the chapter *Start Here for the Tutorials* and choose which session to go on to next, depending on your interests.

8 Creating a Web Application

In this and the next session you create a Web application.

In this session, you use HTML Page Wizard and Form Designer to create the user interface and then use Internet Application Wizard to create the program to process the data. You use the IDE, your Web browser, and the Web server software, Solo, included in Net Express, to run the application to see the appearance of the interface.

In the next session, *Completing and Running Your Web Application*, you add the business logic to the program and run the complete application.

You need to have read the chapter *Start Here for the Tutorials*, worked through the first session *Using Net Express*, and read the chapter *Introduction to Web Applications*, before you do this session.

8.1 Overview

You use HTML Page Wizard to create forms for display on a Web browser. These are the user interface to your application. From within this Wizard, you enter Form Designer, where you design and edit their layout.

You use Internet Application Wizard to generate a server-side COBOL program to handle the forms.

With an HTML form you can specify the order of the controls, but the exact layout is normally determined by the browser that displays it. However, Form Designer gives you facilities to specify the layout exactly.

The server-side program is central to a Web application. It accepts data from an input form, and outputs data in one or more output forms. A special case is where there’s just one form, used for both input and output. This is the kind of application we create in this session.

8.2 Preparation

If you have closed Net Express, open it as before. If any project window or other windows are open, close them.

8.3 Sample Session

In this sample session you:

1. Create the project and form
2. Add controls to the form
3. Create a server-side program
4. Test the form

8.3.1 Creating the Project and Form

You could start by creating a project in the way described in the chapter *Using Net Express*, but if you go straight ahead and create the form Net Express will give you an opportunity to create the project. To create a form:

1. Click **New** on the **File** menu, then select **HTML Page** on the New dialog box and click **OK**.
2. Click **OK** on the message asking if you want to create a project.
3. Make sure **HTML Project** is selected. Enter **Goform** as the name of the project, and **Net Express\Base\Demo\Goform** as the folder to contain the project, then click **Create**.
4. Click **Yes** when asked whether or not you want to create this directory. (If this session has been run previously, you will instead be asked whether to overwrite the existing project. Click **Yes**.)

   The project window for the project appears. So does the first page of the HTML Page Wizard.

5 Click **Positional Form.htm** (the extension **.htm** may not be shown, depending on settings) and click **Next**.

Normally, when you design an HTML form, you can specify only the order of the objects you put on it. Their exact positions are determined by the browser used to view the form. However, in Net Express you can specify that part of your form is a *positional form*. Objects within this will be placed exactly where you put them. By clicking **Positional Form.htm** here you choose that the form will contain one positional form, covering most of its area.

Remember that in Web terminology a Web page that contains fields for a user to fill in is called a form. Do not confuse this with the positional form within it.

6 Overwrite the default HTML filename with **mypage**, leave **Cross-platform** selected, and click **Next**.

This specifies that your HTML form will be called **mypage.htm**. The normal extension for HTML files is **.htm**.

Cross-platform specifies the method Net Express uses to ensure objects in your positional form are positioned as you wish. Cross-platform forms use HTML tables for this. In Dynamic HTML, style attributes are used. HTML Tables are supported by more browsers. Dynamic HTML offers more exact positioning, but is only supported by Internet Explorer 4.0 and later. Remember your forms will be seen by users with different browsers.

7 If you get a message box saying **mypage.htm** or **mypage.mff** exists already, click **Yes** to overwrite it.

You might get one or both of these messages if this session has been run previously. Otherwise the next dialog box appears straight away.

The next dialog box shows a summary of what you have selected.

8 Click **Finish**.

The HTML Wizard generates the form, and finishes. The names of the generated files are added to the project, then the project is scanned for dependencies.

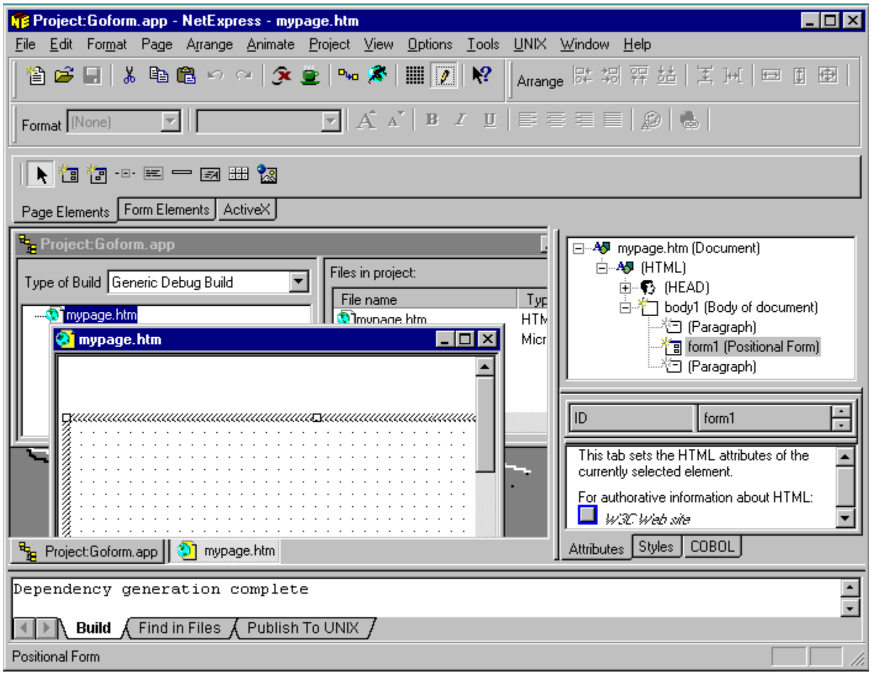
Then the IDE opens a form design window, recognizable by its dotted grid background, and three related windows. An extra toolbar appears, with three tabs. It is called the object toolbar, and contains objects you might want to add to a form. Extra menus **Page** and **Arrange** appear on the menu bar.

This also starts the Web server software, Solo, included in Net Express; you may notice the Solo icon appearing in the Windows taskbar tray. You can ignore it for now - Form Designer needs it running in the background.

8.3.2 Your IDE at this Point

Your IDE now looks something like Figure 8-1.

Figure 8-1. The IDE with Form Designer's Windows Open



Format (None)

Type of Build: Generic Debug Build

ID: form1

This tab sets the HTML attributes of the currently selected element.

For authorative information about HTML:

```
Dependency generation complete
```

The four new windows are:

- Form design window

  The window with the dotted grid background is where you design your form. The form **mypage.htm** that you just created is loaded ready for you to design it. Notice that most of its area is taken up by a rectangle shown as highlighted. This is the positional form.

- Control tree window

  The window at the top right shows you all the controls on your form. You can rename controls by right-clicking on them and selecting **Rename** from the context menu.

- Property list window

  The window below the control tree window shows the properties for the object currently selected in the form design window. Property names are listed down the left, and the corresponding values down the right. You can edit any property by clicking in its value field.

- Help window

  The window at the bottom right displays help for the property list window. If you click any field in the property list window, help for that field appears in the help window.

You may want to drag the form design window aside for a moment to see how the project has changed. You may want to move windows around your IDE or resize them so you can see them all clearly.

8.3.3 Adding Controls to the Form

To design your form:

1. Click the **Form Elements** tab on the object toolbar to bring it in front of the other tabs.
2. Click inside the positional form - the dotted rectangle in the form design window - to ensure it is selected, so objects you create will be placed in it.

3 Click the Text Input button abl on the object toolbar, then click in the form design window. (Leaving the mouse pointer on a toolbar button for a moment displays a tooltip, telling you the button's name.) Delete the initial text **Text Input** that appears in the field.

This creates an entry field - a field where your end-user can enter or display data - at the place where you clicked. You can drag it around with the mouse after you have placed it, by putting the mouse pointer in its border and holding down the mouse button. Put it towards the right-hand side of the window.

Notice the Name property in the property list window (you may have to click in the list and use the arrow keys to go up or down the list to see it). It is **input1**. This will be the data-name for this field in the COBOL program you generate later.

4 Click elsewhere in the dotted rectangle to select it again, so further objects you create will be placed in it.

5 Similarly, click the Text button Aa and then click to the left of the entry field, to create a label for the entry field. Make sure it doesn't overlap the entry field - if it does, move or resize the fields by dragging their borders.

6 In the label field, overtype the default caption **New text** with **Enter name**. (Don't press **Enter**, as you'll start a new line.)

7 Select both the caption and the entry field, by placing your mouse pointer above and to the left of the caption and then dragging it to the bottom right of the entry field. You may have to make the form designer window bigger first so you can see both.

A rectangle appears for a moment, and then both fields are highlighted.

8 Click **Align** on the **Arrange** menu, then click **Top**.

This lines up the two fields.

9 Select the dotted rectangle again, then add a second entry field and label as above, and set the caption for the label to "Greeting".

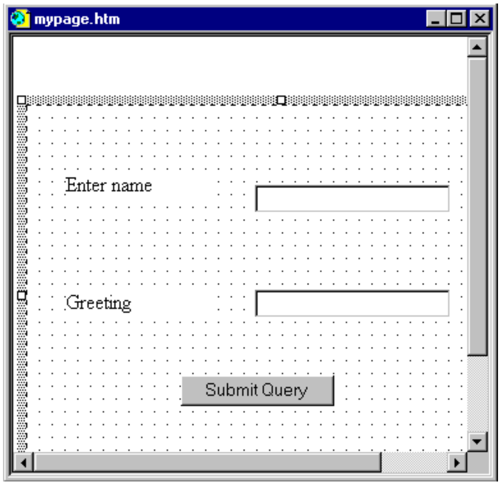
Notice the Name property in the Property List for this second entry field is **input2**.

10 If you wish, you can line up the two captions with one another and the two entry fields with one another. In each case, select the two fields (ensure the rectangle entirely encloses both fields, or they will not be selected), and click **Align** on the **Arrange** menu, and then click **Left**.

11 Select the dotted rectangle again, then click the Submit button Subt on the toolbar, then click in the form to place the button. Put it somewhere toward the bottom of the window.

You should now have a form that looks something like Figure 8-2.

Figure 8-2. The Form You've Created in the Form Design Window



12 Click **Save** on the **File** menu to save your form.

13 Click the ☒ button of the form design window.

The form design window and its associated windows all close. You don't strictly need to close them here - you could keep them open through the next section - but closing them prevents the screen getting too crowded.

8.3.4 Creating the Server-side Program

You use Internet Application Wizard to create the server-side program:

1 Click **New** on the **File** menu, then select **Internet Application** on the New dialog box and click **OK**.

The first page of Internet Application Wizard appears. On this page you choose which of the three methods of creating an application, as described in the chapter *Introduction to Web Applications*, you wish to use.

2 Select **Server Program (from Net Express created HTML)**.

In the lower part of the dialog box, **HTML Form(s)** is automatically selected.

3 Click **Next**.

The page that now appears enables you to give a name to your server-side program, and specify the input form and one or more output forms that it is to use.

4 Enter **myprog.cbl** in the Filename field.

This is the filename that will be given to your server-side program.

5 In both the Input File/Form field and the Output Files field, make sure **mypage** is selected, that is, highlighted.

Unless the project folder has been used previously, **mypage** will be the only form present. The Input File/Form field is automatically set to **mypage**.

A server-side program has one input form, and one or more output forms. A program like this one, which has just one form for both input and output, is called a symmetric server-side program.

6 Click **Next**.

This displays a dialog box showing a summary of what you have selected.

7 Click **Finish**.

The Internet Application Wizard generates the program, and finishes. The names of the generated files are added to the project, then the project is scanned for dependencies. Internet Application Wizard then closes.

8.3.5 Files Created

You have now created your user interface. The Wizards and Form Designer have created the following files for you and added them to the project (you may have to expand the tree view in the project window by clicking the "+" signs to see them all):

- **mypage.htm** is the form in HTML.
- **myprog.cbl** is the server-side program.

The following file is shown in the right-hand pane only:

- **mypage.mff** contains additional information needed for generating the server-side program.

It has also added to the project the following object files, created when you build the project:

- **myprog.obj** is the compiled server-side program.
- **myprog.exe** is the executable file.

It has also created several copyfiles (extensions **.cp***). Their purpose is described in comments in the **.cbl** file.

Although the filenames are shown above in lower case, some might appear in upper case in the project. Filenames are not case sensitive.

If you ever need to update the form, you simply double-click on **mypage.htm** in the project window, and Form Designer opens. When you save the updated form, the copyfiles are regenerated, so you should not edit them directly.You can however edit the **.cbl** program that the Wizard generated.

8.3.6 Testing the Form

If you’re reading this book in your Web browser, you should start a new instance of your browser (use the **New** function on its **File** menu) so that you can still see the book during this section.

To see how your form looks in your Web browser:

1 Double-click on **mypage.htm** in the project window.

This opens the form for editing again. We are not going to make any more changes - we have opened the form for editing just to make the **Page** and **Arrange** menus appear on the menu bar again.

2 Click **Preview** on the **Page** menu.

This starts your Web browser, if it’s not already running, and loads your form. It would also start Solo, if Solo were not already running.

This is explained in more detail in the next session, *Completing and Running Your Web Application*. For now you are just checking to see how the form looks in your browser.

You’ll notice that the two entry fields contain ":f-Input1" and ":f-Input2". You can ignore these; they are placeholders for the data that will be displayed when your server-side program displays the form.

3 Click the form design window’s ☒ button.

If you want, you can now see your server-side program running, although you haven’t yet added the business logic.

4 Click ▤ on the toolbar.

Net Express builds the **.exe** file.

5 Click **Run** on Net Express’s **Animate** menu.

6 Click **OK** on the Start Animating dialog box.

Your server-side program now runs and displays the form in your browser. You’ll notice that the placeholders are no longer visible. The IDE is automatically minimized.

7 Enter your name in the first field and click **Submit Query**.

The program runs and redisplays the form, but nothing is displayed in the second field as you have not added any business logic yet. We will do this in the next session, *Completing and Running Your Web Application*.

8 Restore the minimized IDE, and click **Stop Animating** on Net Express’s **Animate** menu.

8.4 Before Continuing

If you opened a second instance of your Web browser to display the form, close it.

In the next session, *Completing and Running Your Web Application*, you get this application working fully. If you're planning to go onto that session right away, you can keep the project open. If you want to take a break first, you can close the project, and open it again at the start of the next session. Or you can close Net Express entirely, with or without closing the project first. You can also close Solo, by right-clicking on the Solo icon 🌐 in the Windows taskbar tray and clicking **Exit** on the popup menu.

9 Completing and Running Your Web Application

In the previous session you used HTML Wizard and Form Designer to design a form, and Internet Application Wizard to generate a COBOL program. You now edit the COBOL program generated and add your business logic. You then run and debug the completed application, using your Web browser and the supplied Web server software, Solo. You run both on your own machine.

You need to have read the chapter *Start Here for the Tutorials*, worked through the first session, *Using Net Express*, read the chapter *Introduction to Web Applications*, and worked through the previous session, *Creating a Web Application*, before you do this session.

9.1 Overview

Internet Application Wizard generates a COBOL source program in a **.cbl** file, containing code to input and output the form. You then edit this **.cbl** file to add your business logic to process the data from the form and create data to be output. Of course, you can simply add CALL statements to call other programs, and so keep your business logic completely separate from this form-handling program.

9.2 Preparation

If you have closed Net Express, open it as before. If you have closed the project, open it as described in the chapter *Start Here for the Tutorials*. If you use **Open**, the project to open is **Goform** in the directory **Net Express\Base\Demo\Goform**. By default the Open dialog box shows files of type **.app**, **.cbl**, and **.cpy**, so the file **Goform** should be visible.

If you have any Web server software running on your computer, other than Solo, close it. Otherwise you might have problems running Solo.

It doesn't matter whether you initially have your Web browser open.

If you're reading this book in your Web browser, you should start a new instance of your browser (use the **New** function on its **File** menu) so that you can still see the book during this chapter.

9.3 Sample Session

In this sample session you:

1. Edit the COBOL Program
2. Build the application
3. Start Solo
4. Run the Application
5. Debug the Application

9.3.1 Editing the COBOL Program

You could use any text editor to add your business logic to the COBOL program. We will use the IDE.

1. Double-click **myprog.cbl** in the project window.

   A text window opens with the source of the program. You may need to resize the window to see the source clearly. Look at **Hide All Copyfiles** on the **View** menu and ensure that the button-like icon beside it is shown depressed (if it isn't, click it).

   Take a few minutes to look through the Procedure Division and read the comments to see what it does.

2 Click **Hide All Copyfiles** on the **View** menu.

This turns off "Hide All Copyfiles" and so expands them. The copyfiles were generated by Internet Application Wizard together with the **.cbl** file.

(Turning off "Hide All Copyfiles" affects only COPY statements that have previously been expanded using **Copyfile** on the File menu, or that were in the source when the program was previously compiled. The program was compiled when the project was built in the section *Testing the Form* in the previous session, so it expands them all.)

Take a moment to look through the expanded copyfiles.

The Convert-Input Section performs the Input-Conversion Section, which moves data from the data items associated with the form to the program's own work areas. Similarly, the Mypage-Cvt Section moves data to the data items associated with the form. Comments in the source tell you what is in each copyfile.

Remember that this program uses the same form for both input and output. The two entry fields are called F-INPUT1 and F-INPUT2. You can see that on input the data entered in the two entry fields will be moved to INPUT1 and INPUT2, and on output the contents of INPUT1 and INPUT2 will be moved to the two entry fields.

3 In the Process-Business-Logic Section, move the cursor to the blank line above the EXIT statement and press **Enter** to create an empty line. Move the cursor to column 12, above the "e" of EXIT, and insert:

```
string "Hello " Input1 delimited by size into Input2
```

In a real application you would probably put a call here to a subprogram to process data from the form and calculate or look up data to display.

4 Close the text window. Click **Yes** on the message asking if you want to save the file.

9.3.2 Building the Application

To build the application:

1 Click **Rebuild All** on the **Project** menu.

Net Express saves and compiles the program, and builds the executable file. Wait until the message "Rebuild Complete" appears in the Output window before continuing.

9.3.3 Running the Application

To run the application:

1 Click **Run** on Net Express's **Animate** menu.

Because this application uses only one form, the Start Animating dialog box shows the executable file as the place to start execution.

2 Click **OK** on the Start Animating dialog box.

This starts Solo and your Web browser, and then runs your program, just as if an end-user had clicked a link to the executable. Your program displays its form. The IDE is automatically minimized.

If you have any problems getting Solo to run, restore the minimized IDE and click **Help Topics** on Net Express's **Help** menu, then look up "Solo Troubleshooter" in the help index.

3 In your Web browser, click in the **Enter Name** field, type your name and click **Submit Query**.

The program runs. It puts your name after the "Hello " in the **Greeting** field, then redisplays the form in your Web browser.

4 Click in the **Enter name** field, replace your name with another name and click **Submit Query**.

The program runs again, displaying "Hello " followed by the new name.

5 Restore the minimized IDE, and click **Stop Animating** on Net Express's **Animate** menu.

Note: When you run a GUI application, you might not be able to see its window because it is hidden behind the Net Express window. Click the application's button on the taskbar to bring it to the foreground.

9.3.4 Debugging the Application

To debug the application:

1. Click **Start Animating** on Net Express's **Animate** menu.

2. Click **OK** on the Start Animating dialog box.

   Your Web browser starts, and the source of **myprog.cbl** appears in the IDE, ready for debugging. Typically you want to run through the program once to display the form before you really start debugging.

3. Click **Run** on Net Express's **Animate** menu.

   The IDE is minimized, then the program runs without animation and displays its form in your Web browser. The form is in its initial state (it replaces the one left from the previous section, which still has the output data in it).

   Run is the function used for running an application within the IDE. It is also used while debugging, to execute up to a breakpoint without animation. When debugging a server-side program, it treats reentering the program at the beginning as a breakpoint.

4. In your Web browser, click in the **Enter name** field, type your name and click **Submit Query**.

   The minimized IDE is restored, with the source of **myprog.cbl** ready for debugging again.

5. Use on the toolbar to step through **myprog.cbl** until you are about to execute the STOP RUN statement.

   The program is not very long - stepping through it will probably take you less than a minute. The Embedded HTML (the EXEC HTML statement) may take a few seconds to execute.

6. Step the STOP RUN statement.

   The IDE is minimized and the completed Web form is displayed, with the **Greeting** field updated.

7. Restore the minimized IDE, and click **Stop Animating** on Net Express's **Animate** menu.

9.4 Before Continuing

Close the project.

Close Solo, by right-clicking on the Solo icon in the taskbar tray and clicking **Exit** on the popup menu.

If you opened a second instance of your Web browser to display the forms, close it.

If you're planning to go straight on to another session, you can keep Net Express open.

Return to the *Tutorials Map* in the chapter *Start Here for the Tutorials* and choose which session to go on to next, depending on your interests.

10 Creating a Web Application from a COBOL Application

You use Internet Application Wizard to convert an existing COBOL application into a Web application.

You need to have read the chapter *Start Here for the Tutorials*, worked through the first session *Using Net Express*, and read the chapter *Introduction to Web Applications*, before you do this session.

10.1 Overview

You use Internet Application Wizard to generate forms and a COBOL program to handle them, from the Linkage Section of an existing COBOL program. The form-handling program and the original program together are your server-side program.

The existing COBOL program could be a legacy program, or you could have written it as the first step in creating this new application.

The existing program needs to be a subprogram, in which the data for the user interface is defined in the Linkage Section. Typically, it is a file-handling or database-handling program, and in the existing application the screen handling was done by another program which called this program, passing it user queries and getting back answers. In the new Web-based application the generated forms and program take the place of that original screen-handling program.

Internet Application Wizard is not suitable for converting a Windows GUI application into a Web application. The chapter *Introduction* in the online book ***Migration Cookbook*** discusses this task.

In this session you start with an existing COBOL subprogram, **acctopen.cbl**, which accesses account data held in a COBOL indexed file. When your converted application runs, your end-user sees an HTML form with one field. The end-user types in an account number and clicks **Submit Query**, and is returned a form showing the name of the sales representative handling the account and details of the most recent order on that account. The application looks this up in the data files **acctsasc.vsm** and **acctsasc.idx** in **Net Express\Base\Demo\Formx**.

10.2 Preparation

If you have closed Net Express, open it as before. If any project window or other windows are open, close them.

It doesn't matter whether you initially have your Web browser and/or Solo running.

If you're reading this book in your Web browser, you should start a new instance of your browser (use the **New** function on its **File** menu) so that you can still see the book during this chapter.

10.3 Sample Session

In this sample session you:

1. Create the project, forms, and program
2. Add field properties
3. Build the application
4. Run the application

10.3.1 Creating the Project, Forms and Program

You could start by creating a project in the way described in the chapter *Using Net Express*, but if you go straight ahead and create the application Net Express will give you an opportunity to create the project. To create the application:

1 Click **New** on the **File** menu, then select **Internet Application** on the New dialog box and click **OK**.

  The first page of the Internet Application Wizard appears. On this page you choose which of the three methods of creating an application, as described in the chapter *Introduction to Web Applications*, you wish to use.

2 Select **Full Application (HTML Client and Server Program)**.

  In the lower part of the dialog box, **COBOL Source File** is selected by default.

3 Click **Next**.

4 Click **Yes** on the message asking if you want to create a project.

5 Make sure **Empty Project** is selected (it's the default, so it should be). Enter **Account** as the name of the project, and **Net Express\Base\Demo\Formx** as the folder to contain the project, then click **Create**.

  This folder was created when Net Express was installed, and the COBOL program **acctopen.cbl** supplied for this tutorial was installed in it. If this session has been run previously, you will be asked whether to overwrite the existing project. Click **Yes**.

  The page that now appears enables you to specify the existing program you are generating the form and form-handling program from.

6 Set the **Source Program** field by clicking **Browse** and double-clicking **acctopen.cbl**, then click **Next**.

7 Click **Separate Input/Output Forms**, and accept the default **Cross-platform (table output)**, then click **Next**.

  Cross-platform specifies the method Net Express uses to ensure objects in your positional form are positioned as you wish. Cross-platform forms use HTML tables for this. In Dynamic HTML, style attributes are used. HTML Tables are supported by more browsers. Dynamic HTML offers more exact positioning, but is only supported by Internet Explorer 4.0 and later. Remember your forms will be seen by users with different browsers.

The page that now appears shows the files that will be created. Two are the HTML forms for entering data and displaying results; the other is the COBOL program to handle the forms and call Acctopen.

8 Click **Next**.

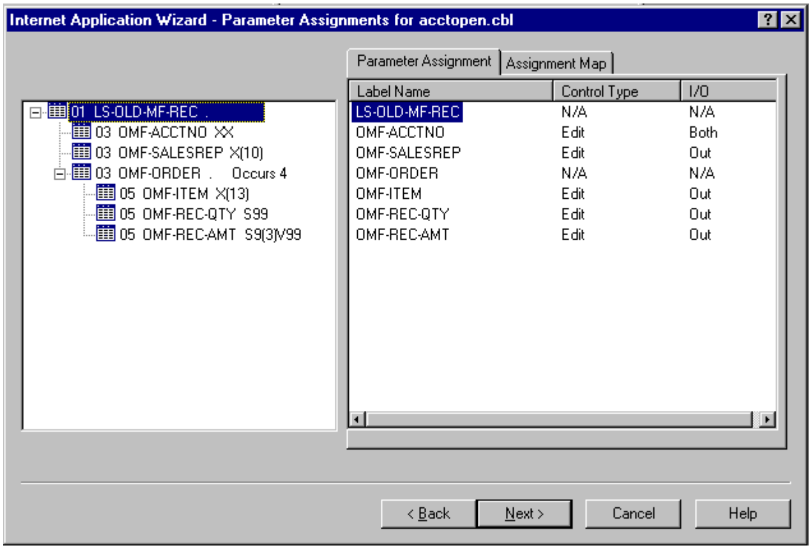
10.3.2 Adding Field Properties

The page that now appears shows the data items in your original program, and on the right-hand side the corresponding fields on the forms that will be generated. Initially only the top-level item is shown.

1 Click "+" by **LS-OLD-MF-REC**, and then click the "+" by **OMF-ORDER**.

The tree view expands to show all the items subordinate to **ls-old-mf-rec**, as shown in Figure 10-1.

Figure 10-1. Data items and their Corresponding Fields


Internet Application Wizard - Parameter Assignments for acctopen.cbl
Parameter Assignment
Assignment Map
< Back
Next >
Cancel
Help

```
01 LS-OLD-MF-REC .
   03 OMF-ACCTNO XX
   03 OMF-SALESREP X(10)
   03 OMF-ORDER .    Occurs 4
      05 OMF-ITEM X(13)
      05 OMF-REC-QTY S99
      05 OMF-REC-AMT S9(3)V99
```

Label Name	Control Type	I/O
LS-OLD-MF-REC	N/A	N/A
OMF-ACCTNO	Edit	Both
OMF-SALESREP	Edit	Out
OMF-ORDER	N/A	N/A
OMF-ITEM	Edit	Out
OMF-REC-QTY	Edit	Out
OMF-REC-AMT	Edit	Out

The left-hand pane shows the data items declared in the Linkage Section of **acctopen.cbl**. The right-hand pane shows the corresponding fields that will be on the forms that Internet Application Wizard will generate. The **Label Name** column shows the labels that will appear by the fields. These have been set to the data-names from the declarations. You now replace these by more user-friendly labels, and set other properties.

2 In the right-hand pane, click **OMF-ACCTNO** in the **Label Name** field and type **Account** instead. Press **Enter**.

There's no need to change the entry under **Control Type**, because the default, **Edit**, is what we want. It specifies that this data item will be represented by a text entry field on the form.

There's no need to change the entry under **I/O**, because the default, **Both**, is what we want. It specifies that this data item both receives input from and sends data to the application's form(s). Since this is an asymmetric application (having separate input and output forms), this means the corresponding field will appear on both the input and the output forms.

3 In the right-hand pane, click **OMF-SALESREP** in the **Label Name** field and type **Representative** instead. Press **Enter**.

There's no need to change the entry under **Control Type**, because the default, **Edit**, is what we want.

4 Click the **I/O** field for **OMF-SALESREP** (now **Representative**), and select **Out** from the dropdown list.

This specifies that this data item sends data to the application's form(s). Since this is an asymmetric application, this means the corresponding field will appear only on the output form.

5 Right-click **OMF-ORDER** in the left-hand pane, then click **Out** on the popup menu.

Setting a property of a group field in this way affects its subordinate items. The **I/O** column in the right-hand pane changes to show this property on these items.

6 Click **Next**.

The page that now appears shows a summary of what you have selected.

7 Click **Finish**.

The Internet Application Wizard generates the forms and program, and finishes. The names of the generated files are added to the project, then the project is scanned for dependencies. Internet Application Wizard then closes.

10.3.3 Files Created

You have now created your user interface. Internet Application Wizard has created the following files for you and added them to the project (you may have to expand the tree view in the project view by clicking the "+" signs to see them all):

- **acctopen_input.htm** is the input form.
- **acctopen_output.htm** is the output form.
- **acctopen_server.cbl** when compiled and linked with **acctopen.cbl** is the server-side program.

The following files are shown in the right-hand pane only:

- **acctopen_input.mff** contains additional information needed for generating the server-side program.
- **acctopen_output.mff** contains additional information needed for generating the server-side program.

It has also added your original program **acctopen.cbl** to the project.

It has also added to the project the following object files, created when you build the project:

- **acctopen_server.obj** is the compiled form-handling program.
- **acctopen_server.exe** is the executable file of the server-side program.
- **acctopen.obj** is the compiled original program

It has also created several copyfiles (extensions **.cp***). Their purpose is described in comments in the **.cbl** file.

Although the filenames are shown above in lower case, some might appear in upper case in the project. Filenames are not case sensitive.

If you ever want to update the forms, you simply double-click on **acctopen_input.htm** or **acctopen_output.htm** in the project window, and Form Designer opens. When you save the updated form, the copyfiles are regenerated, so you should not edit them directly.You can however edit the **.cbl** program that the Wizard generated.

10.3.4 Building the Application

To build the application:

1 Click **Rebuild All** on the **Project** menu.

Net Express compiles the programs, and builds the executable file. Wait until the message "Rebuild Complete" appears in the Output window before continuing.

10.3.5 Running the Application

To run the application:

1 Click **Run** on Net Express's **Animate** menu.

Because this application uses separate input and output forms, the Start Animating dialog box shows the input form as the place to start execution.

2 Click **OK** on the Start Animating dialog box.

This starts Solo and your Web browser, and then loads your input form **acctopen_input.htm** into the browser, just as if an end-user had clicked a link to it.

If you have any problems getting Solo to run, click **Help Topics** on Net Express's **Help** menu, then look up "Solo Troubleshooter" in the help index.

3 In your Web browser, click in the **Account Number** field, type "A1", and click **Submit Form**. (This application is case-sensitive, so type "A1", not "a1".)

The IDE is automatically minimized, and the program runs. It puts the name of the sales representative for account "A1" and details of the most recent order on that account into the output form, and displays the form in your Web browser.

4. Try other numbers between "A1" and "A9" in the **Account Number** field.

   The file has records for accounts from A1 to A9. The existing program **acctopen.cbl** includes code such that if you try some other number, such as B1, you get "Not found" in the **Representative** field.

5. Restore the minimized IDE, and click **Stop Animating** on Net Express's **Animate** menu.

10.4 Before Continuing

Close the project.

Close Solo, by right-clicking on the Solo icon 🌐 in the taskbar tray and clicking **Exit** on the popup menu.

If you opened a second instance of your Web browser to display the forms, close it.

If you're planning to go straight on to another session, you can keep Net Express open.

Return to the *Tutorials Map* in the chapter *Start Here for the Tutorials* and choose which session to go on to next, depending on your interests.

11 Creating a Web Database Application

You use Internet Application Wizard to create a Web interface to an existing database.

You need to have read the chapter *Start Here for the Tutorials*, worked through the first session *Using Net Express*, and read the chapter *Introduction to Web Applications*, before you do this session.

11.1 Overview

You use Internet Application Wizard to create forms and server-side programs so you can access an existing database via the Web.

The database must have been created with a database system for which Open Database Connectivity (ODBC) support is available. This includes most major databases, such as Oracle, Sybase, Informix, DB2, Microsoft SQL (pronounced "sequel") Server, and Microsoft Access.

Using Internet Application Wizard you don't need to do any form design or programming yourself. The Wizard looks at the database structure and creates the forms and server-side programs for you. You just set some options about what kinds of operation are allowed.

The application generated can both query and - if you choose this option - update the database. You can still use your original database tool to query and update the database too.

Net Express includes several sample databases, including a Microsoft Access database containing details of a wholesaler's customers. The customers are retail outlets. In this session, you create a Web application to query and update this database. The other database we will look at is an XDB one. Where appropriate in the following sections, we tell you what to enter if you want to use the XDB database rather than the Microsoft Access one.

The database is accessed via the data source name (DSN) **NetExpress Sample2**, which represents the sample Microsoft Access database **\Net Express\Base\Demo\SMPLDATA\access\sample.mdb**. (For details of ODBC and how DSN's are assigned, see the help for your ODBC Administrator, accessible via the Control Panel on your Windows desktop.) The XDB DSN is **NetExpress XDB Sample1**.

11.1.1 Terminology

In this session, when talking about the database we will keep to database terminology and use the terms *row* and *column*. We will use the corresponding terms *record* and *field* when data is displayed in the forms on the Web browser.

11.2 Preparation

If you have closed Net Express, open it as before. If any project window or text windows are open, close them.

It doesn't matter whether you initially have your Web browser and/or Solo running.

If you're reading this book in your Web browser, you should start a new instance of your browser (use the **New** function on its **File** menu) so that you can still see the book during this chapter.

11.3 Sample Session

In this sample session you:

1. Create the project, forms, and programs
2. Build the application
3. Run the application

4 Navigate the database

5 Update the database

6 Filter the database

11.3.1 Creating the Project, Forms, and Programs

You could start by creating a project in the way described in the chapter *Using Net Express*, but if you go straight ahead and create the application Net Express will give you an opportunity to create the project. To create the application:

1 Click **New** on the **File** menu, then select **Internet Application** on the New dialog box and click **OK**.

The first page of the Internet Application Wizard appears. On this page you choose which of the three methods of creating an application, as described in the chapter *Introduction to Web Applications*, you wish to use.

2 Select **Full Application (HTML Client and Server Program)**, and in the lower part of the dialog box select **SQL Database**.

3 Click **Next**.

4 Click **Yes** on the message asking if you want to create a project.

5 Make sure **Empty Project** is selected (it's the default, so it should be). Enter **Daw** as the name of the project, and **Net Express\Base\Demo\Daw** as the folder to contain the project, then click **Create**.

6 Click **Yes** when asked whether or not you want to create this directory. (If this session has been run previously, you will instead be asked whether to overwrite the existing project. Click **Yes**.)

7 Enter **Retail** as the basename for all the files to be generated, and **Retail Outlets** as the title to appear on all the forms, then click **Next**.

The page that now appears lists the data sources on your computer. At the right are a number of tabs. Make sure the **Query** tab is at the front. (It should be, as it's the default.)

8 Double-click **NetExpress Sample2** (or **NetExpress XDB Sample1** for XDB).

The list expands to show the tables in this database.

9 Double-click **Customer**.

The list expands to show the columns in the table called Customer.

A SQL (pronounced "sequel") SELECT statement is generated and displayed on the right. SQL is a language for querying databases. Net Express includes a feature called OpenESQL, a compiler preprocessor that enables SQL to be included in COBOL source. You don't need to know SQL to use the Internet Application Wizard, as the Wizard generates it for you.

The statement is not yet complete. It will be completed in the next step, and then used as the basis for the SQL statements included in the COBOL source program that the Internet Application Wizard generates.

10 Right-click **Customer** in the tree view and click **Select All Columns** on the popup menu.

Checkmarks appear by all the column names, and the statement in the Query field is completed. This statement will cause all columns from the Customer table to be displayed.

Although you are creating this query for inclusion in a COBOL program, you can run it now, interactively, as follows.

11 Click **Run Query** (near the top left of the Internet Application Wizard dialog box).

The **Results** tab comes to the front, displaying all the columns from all the rows in the Customer table.

The SQL statements being generated for inclusion in the source program are equivalent to this one, but they also handle selection criteria that can be specified at run time by the user, to select which rows - that is, which customers' details - will be displayed. These statements will cause data from the table to be read into the program's Working-Storage Section, from where other COBOL statements will move them to the Web form.

This dialog box is also used by another facility in Net Express, the OpenESQL Assistant. You use this facility to create SQL queries either to run interactively or to include in a COBOL source program that you write yourself.

To learn how to create more complex queries than the one we've just created, after finishing this book you should read the tutorial on the OpenESQL Assistant in the online book ***Database Access***.

12 Click **Next**.

On the page that now appears you specify what Web forms you want your application to have.

Cross-platform specifies the method Net Express uses to ensure objects in your positional form are positioned as you wish. Cross-platform forms use HTML tables for this. In Dynamic HTML, style attributes are used. HTML Tables are supported by more browsers. Dynamic HTML offers more exact positioning, but is only supported by Internet Explorer 4.0 and later. Remember your forms will be seen by users with different browsers. We will accept this default.

There are two standard forms. The single-record view (also known as simply "record view") form displays the data from one row of the table; in our case, one customer's details. The table view form displays the data from several rows, in a tabular layout. We will accept the default, which is to have both forms.

13 Click **Next**.

On the page that now appears you specify whether your user can only query the database, or can update it too. We will accept the default, which is to let the user update it.

14 Click **Next**.

The page that now appears shows a summary of what you have selected.

15 Click **Finish**.

The Internet Application Wizard generates the forms and programs, and finishes. The names of the generated files are added to the project, then the project is scanned for dependencies. Internet Application Wizard then closes.

11.3.2 Files Created

You have now created your user interface. The Internet Application Wizard has created the following files for you and added them to the project (you may have to expand the tree view in the project view by clicking the "+" signs to see them all):

- **retailform.htm** is the record view form, in HTML.
- **retailform.cbl** is the server-side program for this form.
- Plus a similar set of files, all with the base name **retaillist**, for the table view form.

The following files are shown in the right-hand pane only:

- **retailform.mff** contains additional information needed for generating the server-side program.
- **retaillist.mff** contains additional information needed for generating the server-side program.

It has also added to the project the following object files, created when you build the project:

- **retailform.obj** is the compiled server-side program
- **retailform.exe** is the executable file.
- Plus a similar set of files, all with the base name **retaillist**, for the table view form.

It has also created several copyfiles (extensions **.cp***). Their purpose is described in comments in the **.cbl** file.

Although the filenames are shown above in lower case, some might appear in upper case in the project. Filenames are not case sensitive.

If you ever want to update the forms, you simply double-click on **retailform.htm** or **retaillist.htm** in the project window, and Form Designer opens. When you save the updated form, the copyfiles are regenerated, so you should not edit them directly. You can however edit the **.cbl** programs that the Wizard generated.

11.3.3 Building the Application

To build the application:

1 Click **Rebuild All** on the **Project** menu.

Net Express saves and compiles the programs, and builds the executable files. Wait until the message "Rebuild Complete" appears in the Output window before continuing.

11.3.4 Running the Application

To run the application:

1 Click **Run** on Net Express's **Animate** menu.

2 Click **OK** on the Start Animating dialog box.

This starts Solo and your Web browser, and then runs the program Retailform. The program displays its form, the single-record view form **retailform.htm**. The IDE is automatically minimized.

Initially, the form shows the customer details from the first row in the table, the one with CustID = ALWAO.

Depending on the font size in your Web browser, you may not always see all five characters of the Customer ID in the record view form. For example, ALWAO may appear as ALWA, or MERRG as MERR. You will see the full Customer ID if you scroll the CustID field sideways by clicking in it and using the left and right arrow keys.

3 Click **Table View** at the top right of the form.

The table view form **retaillist.htm** is loaded into the browser. It shows the first ten customers in the table, ALWAO through CONSH.

4 Click **Record View** at the top right of the form.

The single-record view form **retailform.htm** is loaded again.

11.3.5 Navigating the Database

Both the forms have standard controls for moving backward and forward through the table. The record view form **retailform.htm** also has functions for updating and querying the table. Some of these are optional when you are creating the forms - we accepted all the defaults to have them all included.

In this section we'll try out some of these standard controls. Make sure you still have the single-record view form displayed. The Order By field has defaulted to CustID. This causes rows from the table to appear in the order of CustID.

You may need to make your Web browser full screen to see the displayed forms properly.

1 Click > at the top of the form.

The form shows the next customer in the table, the one with CustID = ANDRC.

2 Click **Table View**.

The table view form is loaded, showing ten customers starting from ANDRC.

3 Click >>.

The form shows the end of the table, that is, the last ten customers TOPOF through WITAE.

4 Click >.

The form is redisplayed, but it has not changed. In table view, if you try to go beyond the end of the table, you don't move.

5 Click <<.

The form shows the beginning of the table, that is, the first ten customers ALWAO through CONSH.

6 Click >.

The form shows the next ten customers EASTC through HANOP.

7 Click **Record View**.

The record view is loaded, showing the customer EASTC.

8 Click >>.

The form shows the last customer in the table, WITAE.

9 Click >.

The form shows the first customer in the table, ALWAO. In record view, if you try to go beyond the end of the table, you go back to the start of the table.

10 Type "BE" in the CustID field, replacing ALWAO which is currently there, and click **Query**.

The form shows the first customer whose CustID is alphabetically greater than or equal to what you put in the CustID field. It is customer BERGS.

11 Click Table View.

The table view is loaded, showing ten customers starting from BERGS

12 Click **EASTC** in the first column of customer EASTC.

The single-record form appears showing customer EASTC.

11.3.6 Updating the Database

In creating the application, we accepted the default, which is to let the user update the database. You use the single-record view form to insert, delete, or update records.

1 In the Address field, after "35 King George" add "Street"; then click **Update record** at the bottom of the form.

The record is redisplayed.

2 Click **Table View**.

The table view is loaded again, and you can see that the database has been updated with the change you made.

11.3.7 Filtering the Database

You can use the fields at the top of the record view form to filter out records you don’t want to see, and to specify which field you want records ordered on.

To display only customers in Canada, displaying them in order of city:

1 Click **Record View**.

The record view form appears. You could now click **Clear Screen** to clear the customer data fields before setting the filter, but it’s not necessary - only the fields you specify will be looked at anyway.

2 Using the drop-down lists, set the Order By field to **City**, the Filter Type field to **=** (the default), and the Filter By field to **Country**.

3 Type **Canada** in the Country field, replacing what is currently there, and click **<<**.

The form shows the first (in order of city) customer in the table who is in Canada. This is MERRG, in East Vancouver.

The Filter By field has automatically changed to "(existing)". This ensures that in the next few steps (until you change this field) the set of rows already selected will be used - these steps just change the position in the table, they don’t set a new filter.

4 Type **T** in the City field, replacing what is currently there, and click **Query**.

The form shows the first (in city order) Canadian customer in a city whose name is greater than or equal to "T". This is BOTTM, in Tsawassen.

5 Click **Table View**.

The table view form is loaded. It now displays, in city order, Canadian customers from the one in Tsawassen onward. There are three of them.

Now let’s set a new filter. To do this you will change the Filter By field from "(existing)" to a field-name again.

6 Click **Record View**.

7 Change the Filter By field to **Country**, then type **UK** in the Country field and click <<.

The form shows the first (in city order) UK customer, ISLAT in Hedge End.

8 Click **Table View**.

The table view form is loaded. It now displays, in city order, the first ten UK customers.

Now let's get the full table back, and get it in order of Customer ID again.

9 Click **Record View**.

10 Change the Order By field to **CustID** and the Filter By field to **(none)**, and click <<.

The form is redisplayed showing the first (in order of Customer ID) customer in the table.

11 Click **Table View**.

The table view form is loaded showing the first ten customers. You've got the full table displayed again, in order of Customer ID.

12 Restore the minimized IDE, and click **Stop Animating** on Net Express's **Animate** menu.

11.4 Before Continuing

Close the project.

Close Solo, by right-clicking on the Solo icon 🌐 in the taskbar tray and clicking **Exit** on the popup menu.

If you opened a second instance of your Web browser to display the forms, close it.

If you're planning to go straight on to another session, you can keep Net Express open.

Return to the *Tutorials Map* in the chapter *Start Here for the Tutorials* and choose which session to go on to next, depending on your interests.

Part 4: Windows GUI Development Tutorials

This part contains the following chapters:

- Chapter 12, “Creating a Windows GUI Application”
- Chapter 13, “Completing and Running Your Windows GUI Application”

12 Creating a Windows GUI Application

In this and the next session you create a Windows graphical application.

In this session, you use Windows GUI Application Wizard and Dialog System to create the user interface and the program to process the data. You use the IDE to run the application to see the appearance of the interface.

In the next session, *Completing and Running Your Windows GUI Application*, you add the business logic to the program and run the complete application.

You need to have read the chapter *Start Here for the Tutorials* and worked through the first session, *Using Net Express*, before you do this session.

You need to have installed Dialog System to do this session.

12.1 Overview

You use Windows GUI Application Wizard to create windows and dialog boxes for display on a Windows system. These are the user interface to your application. From within this Wizard, you enter Dialog System, where you design and edit their layout. The set of windows and dialog boxes you design is called a screenset. Dialog System creates a screenset file defining the windows and dialog boxes you create.

You also use Dialog System to generate a COBOL program to handle the screenset. This is called the associated program.

12.1.1 Creating a Screenset

There are four steps in creating a screenset:

1. Define the windows and dialog boxes, and their properties such as status bars and toolbars.
2. Define the data items that will hold data to be passed between the user interface and the COBOL program.
3. Define the controls (such as entry fields and pushbuttons) to appear on the windows and dialog boxes.
4. Write dialog to define what will happen when the user does something at the interface, such as clicking a pushbutton.

You write the dialog mentioned in Step 4 in Dialog System's own scripting language.

Generally, the Dialog System documentation uses the term *graphical objects*, or just *objects*, when talking about controls.

12.2 Preparation

If you have closed Net Express, open it as before. If any project window or other windows are open, close them.

12.3 Sample Session

In this sample session you:

1. Create the project and screenset
2. Define data items for the screenset
3. Add controls to the screenset
4. Add dialog to the screenset

5 Build the application

6 Test the screenset

12.3.1 Creating the Project and Screenset

You could start by creating a project in the way described in the chapter *Using Net Express*, but if you go straight ahead and create the screenset Net Express will give you an opportunity to create the project. To create the screenset:

1 Click **New** on the **File** menu, then select **Dialog System Screenset** on the New dialog box, and click **Yes**.

2 Click **OK** on the message asking if you want to create a project.

3 Select **Windows GUI Project**. Enter **Welcome** as the name of the project, and **Net Express\Base\Demo\Welcome** as the folder to contain the project, then click **Create**.

4 Click **Yes** when asked whether or not you want to create this directory. (If this session has been run previously, you will instead be asked whether to overwrite the existing project. Click **Yes**.)

The project window for the project appears. So does the first page of Windows GUI Application Wizard.

5 Overwrite the default screenset name **NewSet.gs** with **Welcome.gs** (leave the path unchanged) and click **Next**. (If this session has been run previously, you will be asked whether to overwrite the existing **Welcome.gs**. Click **Yes**.)

6 Select **Yes** to create a Multiple Document Interface (MDI) application, enter **2** as the number of MDI children, and click **Next**.

A Single Document Interface (SDI) consists of just one window. An MDI consists of a number of windows and/or dialog boxes, with one window as their parent.

7 Select **Status Bar** and **Main window Toolbar**, and click **Next**.

This specifies that the main window is to have a status bar and toolbar, and also a menu bar - the toolbar and menu bar are created together.

8 Click **Next** twice.

The two dialog boxes you are skipping enable the use of Dialog System Extensions, and enable you to modify run-time behavior. We will accept the defaults.

9 Ensure **Generate skeleton COBOL program** is checked and **Net Express\Base\Demo\Welcome\Welcome.cbl** is the name of the COBOL program to generate, and click **Next**.

The next dialog box shows a summary of what you have selected.

10 Click **Finish**.

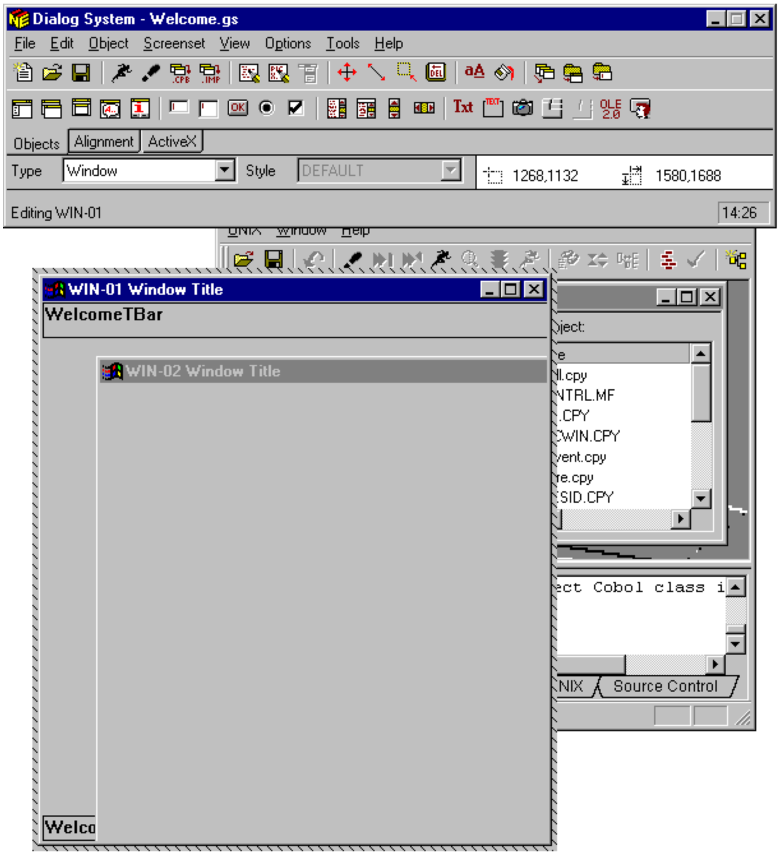
The Windows GUI Application Wizard generates the screenset and program, and finishes. The names of the generated files are added to the project, then the project is scanned for dependencies.

Then Dialog System is loaded.

12.3.2 Your Screen at this Point

You screen now looks something like Figure 12-1. The positions of the three individual windows may vary. You may want to move them around your screen so you can see them all clearly.

Figure 12-1. The Screen with Dialog System Loaded



Type Window
Style DEFAULT
1268,1132
1580,1688
Editing WIN-01
14:26

The two new windows are:

- WIN-01

  The window titled "WIN-01 Window Title" is where you will design your screenset. This window WIN-01 is the main window of your screenset. A child window, WIN-02, is shown within it because we specified it as an MDI.

- Dialog System's own window

  This consists of menu bar, toolbars and status bar, with the title "Dialog System - Welcome.gs" above and "Editing WIN-01" below. Initially it may overlap WIN-01 on your screen.

You may want to bring the project window to the foreground (click in it) to see how the project has changed. Then click in Dialog System and WIN-01 to bring them to the foreground again.

12.3.3 Defining Data for the Screenset

On-screen objects that contain data - objects such as entry fields - need to be associated with data items. Data is passed between these data items and the on-screen objects.

In this tutorial we define the data items now and the objects in a later section. It's not essential to do it this way round. As you'll see later, you can test the appearance and behavior of your screenset without data. However, it's generally best to define your data first, so that you have a clear idea of what information has to be passed to, and obtained from, the user. You can change the appearance of the screenset without changing any data definitions.

The data items you define are called the *data block*. When the generated program calls Dialog System's run-time support module, DSGRUN, it passes the data block as a parameter. This is how data is passed between the COBOL program and the screenset.

To define data items, you use the Dialog System window. Ignore WIN-01 for now.

Dialog System provides two modes of entering data field definitions. In Prompted mode you are prompted with lists of choices and dialog boxes to complete. This helps if you are not familiar with the syntax of data declarations in Dialog System. In unprompted mode (also called editor mode) you type the declarations directly. We will use unprompted mode.

1 Click **Data block** on Dialog System's **Screenset** menu.

  The Data Definition window appears. Specialized edit windows like this are used throughout Dialog System for entering different kinds of information. It has its own menu, which we will be using in the next few steps.

You can see some data declarations already present. These were automatically generated when the screenset was created. They are used in the maintenance of the MDI, toolbar, menu bar, and status bar, and you can ignore them.

2 Ensure that **Prompted mode** on the Data Definition window's **Options** menu has no checkmark by it. (If it has, click it.)

3 Click in the Data Definition window to select it, then page down to the end of the existing items, and press **Enter**.

4 Type: **I-NAME X 12.0** and press **Enter**.

The line is automatically formatted. You can even type in lower case, and it will be changed automatically to upper case. You have now defined your first data item, I-NAME. The X means it is alphanumeric, and the 12.0 means it is 12 characters long with no decimal spaces. The ".0" must be present, even though the field is alphanumeric; however, you can omit it, and it will be automatically inserted.

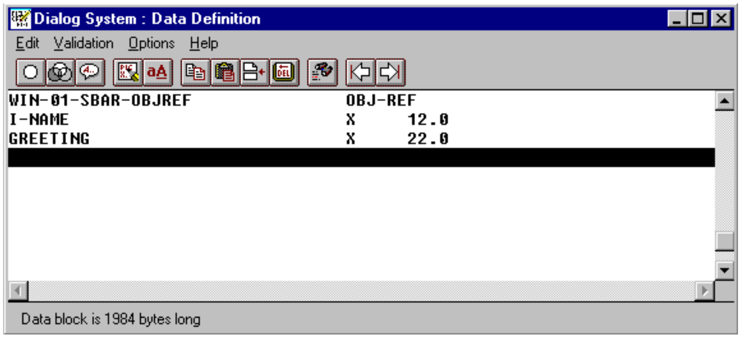
Because these data items will be accessed from the generated COBOL program as well as from Dialog System, you have to avoid COBOL reserved words (no warning is given if you use one). This is why we've called it I-NAME rather than NAME.

5 Type: **GREETING X 22.0** and press **Enter**.

This is your second data item, GREETING. It is alphanumeric and 22 characters long.

The Data Definition window should now look like Figure 12-2.

Figure 12-2. Data Definition Window with Data Declarations



```
WIN-01-SBAR-OBJREF                 OBJ-REF
I-NAME                             X      12.0
GREETING                           X      22.0
```

Data block is 1984 bytes long

6 Click **Exit data definition** on the Data Definition windows's **Edit** menu.

Although the data definitions disappear from the screen, they are still loaded in memory. They will not be saved to disk till you do a **Save** later to save the screenset.

When Dialog System generated the screenset and the COBOL program earlier, it generated a copyfile (**.cpb**) containing a definition of the data block, in COBOL. This copyfile is used in the Working-Storage Section of the generated program, which is where the data is actually held. Now that you have changed the data block you must regenerate this copyfile.

7 Click **Generate** on Dialog System's **File** menu, then click **Data block copyfile**, accept the default **Welcome.cpb** as the name of the copyfile, and click **Save**. Click **Yes** to confirm the existing copyfile is to be overwritten.

Always remember to regenerate this copyfile whenever you change the data block.

Before going on to the next section, let's save the screenset to safeguard our work so far.

8 Click **Save** on Dialog System's **File** menu.

12.3.4 Adding Controls to the Screenset

Next you specify the appearance of the user interface. We have already added a status bar, toolbar and menu bar, when we created the screenset, but we now need to add text and entry fields. Controls such as these are generally referred to as "objects" in Dialog System.

The window now visible on the screen, WIN-01, is the parent, or main window, of your screenset. The rectangular outline at the top, marked **Welcome TBar**, indicates the presence of a menu bar and tool bar, and the one at the bottom, marked **Welcome SBar**, indicates the presence of a status bar.

In an MDI, the parent window is only a frame for the child windows; you cannot put objects on it. The child windows are what the user actually uses. We have created an MDI with two child windows to show how it's done, but we will use only one of the child windows.

You add objects by using the **Object** menu or the object toolbar. The object toolbar is quick and easy for mouse users, whereas the menu can also by used by keyboard users. The two are completely equivalent. This session uses the object toolbar.

Dialog System gives all the windows the same initial size. Let’s start by resizing them so you can see the whole of WIN-02 within WIN-01.

1. Move WIN-01 up your screen, then drag its bottom right-hand corner to make it bigger, so you can see the bottom right-hand corner of WIN-02. Then drag that to make WIN-02 smaller, so it fits easily between the two rectangles marked **Welcome TBar** and **Welcome SBar**. Then change WIN-01 back to its original size - just big enough to enclose these two rectangles.

2. Click anywhere within WIN-02 to ensure it’s selected, and click **Properties** on Dialog System’s **Edit** menu. (Alternatively you can right-click in WIN-02, and click **Properties** on the popup menu, or simply double-click in WIN-02.)

   The Window Properties dialog box is displayed for you to edit the properties of WIN-02.

3. Overtype the default title **WIN-02 Window Title** with **Welcome**. Accept the rest of the default settings and click **OK**.

4. Click the Entry Field button on the object toolbar, then click in the WIN-02 window.

   This creates an entry field - a field where your end-user can enter or display data - at the place where you clicked. You can drag it around with the mouse after you have placed it. Put it towards the right-hand side of the window.

5. Double-click on the entry field.

   The Entry Field Properties dialog box appears for you to define the properties of the entry field.

6. Overtype the default name **EF1** with **I-NAME-DISP**.

   This gives the name I-NAME-DISP to the entry field.

7. Click **Master field**.

   The Master Fields dialog box is displayed. This lists the fields in the Data Block, so you can link the appropriate data field with the entry field.

8 In the Master Fields dialog box, page down to the field **I-NAME**, click it, and then click **OK**.

The "Master field" field now displays I-NAME, and the "Picture string" field is automatically set to X(12). This means the entry field I-NAME-DISP will be 12 characters long, alphanumeric, and will display whatever data is in the data item I-NAME.

9 Accept the default choices for the remaining properties, and click **OK**.

10 Click the Text button Txt on the object toolbar, and then click to the left of the entry field, to create a label for the entry field.

11 Double-click on the label to display the Text Properties dialog box.

12 Overtype the default text **Text** with **Enter name**, and click **OK**.

13 In the same way, add a second entry field and label. Use "GREETING-DISP" as the name of the field, and select GREETING as its master field. Set the label to "Greeting".

14 Click the OK button OK on the Object toolbar, then click in the WIN-02 window to place the button. Put it somewhere toward the bottom of the window.

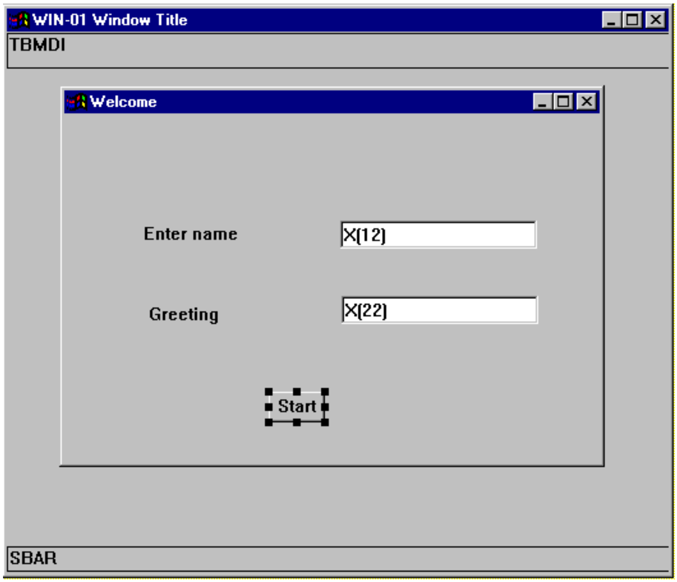
15 Double-click on the button.

The Pushbutton Properties dialog box appears for you to define the properties of the button.

16 Overtype the default name **PB1** with **START-BUTTON**, and overtype the default text **~Pushbutton** with **Start**. Accept the default choices for the remaining properties, and click **OK**.

You should now have a screenset that looks something like Figure 12-3.

Figure 12-3. Your Screenset with the Entry Fields, Text Fields and Pushbutton



Before going on to the next section, let's save the screenset to safeguard our work so far.

17 Click **Save** on Dialog System's **File** menu.

12.3.5 Adding Dialog to the Screenset

So far you have defined the appearance of your window, and defined data items to hold data received from (or for display on) the fields on the window. But when the user uses the window - for example, clicks the **Start** button - nothing will happen. You define the effect of user actions by writing *dialog script*, generally referred to as just *dialog*.

User actions, such as pressing a key or clicking a mouse button, generate *events*. You write procedures to be executed in response to each event.

You can attach dialog to a particular object - a control or a window - or make it global. This is called the *scope* of that dialog. Dialog attached to an object is called local. When an event happens on an object, Dialog System looks first for dialog attached to that object. If there is none, Dialog System looks for dialog attached to the window that the object is on. If there is none, Dialog System looks for global dialog.

We need to write dialog so that when the user clicks the **Start** button, DSGRUN returns to the calling program. Of course we want the data that the user has entered to be moved into the fields in the data block first, but this happens automatically because of the master field associations you defined earlier - you don't need to write dialog to do this.

As with defining data, Dialog System provides a choice of prompted mode or unprompted mode. We will use unprompted mode.

12.3.5.1 Global Dialog

In this example we don't need to write any global dialog, but it will be useful to look at some that has been generated automatically.

1 Click **Global dialog** on Dialog System's **Screenset** menu.

   The Dialog Definition window appears. You can see some dialog already present. This was automatically generated when the screenset was created, to implement the run-time behavior defaults selected in the Wizard.

   The first procedure consists of the event ESC and the two functions MOVE and RETC. This specifies that on the event ESC (generated when the user presses **Esc**), DSGRUN will move 1 to the data item EXIT-FLAG and then return to the calling program.

   Because this dialog is global, this procedure will be executed whenever **Esc** is pressed, regardless of what object currently has focus - unless that object, or the window it is on, has its own dialog for ESC.

   The second procedure specifies the same effect for the event CLOSED-WINDOW (generated when the user closes a window).

   The rest of this dialog maintains the MDI, menu bar, toolbar and status bar, and you can ignore it.

A procedure in Dialog System is called a dialog table. It consists of an event or procedure-name followed by one or more functions that will be performed if the event occurs or the procedure-name is explicitly performed from another procedure. A complete list of Dialog System events and functions can be found in the Help.

The order in which procedures appear does not matter - unlike in COBOL, control does not fall through from the end of one procedure into the next. At the end of a procedure, DSGRUN simply stops and looks for the next event (unless the last function was RETC, which returns to the calling program).

2 Click **Exit dialog defn** on the Dialog Definition window's **Edit** menu.

12.3.5.2 Local Dialog

Now let's write the dialog to make the **Start** button work. Since this dialog applies only to this one object, we will make it local dialog, attached to this object.

1 On your window Welcome, right-click on the **Start** button and click **Dialog** on the popup menu.

The Dialog Definition window appears. Dialog System has started the dialog off for you by inserting the name of the event you're most likely to want to write dialog for on this type of object. BUTTON-SELECTED (generated when the user clicks the button) is indeed what we want.

The highlight is already positioned on the line below the event, ready for you to type.

2 Type *** The Start button returns to the calling program**, then press **Enter**.

Anything after an asterisk on a line is treated as a comment.

3 Type **RETC** and press **Enter**.

The line is automatically formatted. You can even type in lower case, and it will be changed automatically to upper case. Dialog System recognizes RETC as the name of a function, and positions it indented.

4 Type **REFRESH-OBJECT GREETING-DISP** and press **Enter**.

The dialog should now be as follows:

```
 BUTTON-SELECTED
* The Start button returns to the calling program
    RETC
    REFRESH-OBJECT GREETING-DISP
```

RETC makes DSGRUN return to the COBOL program. When DSGRUN is next called, it continues from where it left off. Often RETC is the last function in a procedure, and so DSGRUN, when it is reentered, finds the procedure finished and simply waits for the next event. However, in this case we want DSGRUN to do something more when it's reentered - we need GREETING-DISP updated with the updated data that the COBOL program will have put in its master field GREETING. The REFRESH-OBJECT function does this.

The syntax of functions is given in the topic *Dialog Statements: Functions* in the Dialog System Help.

5 Click **Exit dialog defn** on the Dialog Definition window's **Edit** menu.

6 Click **Save** on Dialog System's **File** menu.

12.3.6 Files Created

You have now created your user interface. The Wizard and Dialog System have created the following files for you and added them to the project (you may have to expand the tree view in the project window by clicking the "+" signs to see them):

- **welcome.gs** is the screenset
- **welcome.cbl** is the associated program
- **welcome.cpb** is a copyfile defining the data block in COBOL
- **welcomeTBar.cbl** is a program to handle the toolbar and menu
- **welcomeSBar.cbl** is a program to handle the status bar

It has also added to the project the following object files, created when you build the project:

- **welcome.int** is the compiled associated program.

- **welcomeTBar.int** is the compiled toolbar/menu program.
- **welcomeSBar.int** is the compiled status bar program.

It has also created several copyfiles (extensions **.cp***).

Although the filenames are shown above in lower case, some might appear in upper case in the project. Filenames are not case sensitive.

If you ever need to update the screenset, you simply double-click on **Welcome.gs** in the project window, and Dialog System opens. When you save the updated screenset, the copyfiles are regenerated, so you should not edit them directly.You can however edit the **.cbl** program that the Wizard generated.

12.3.7 Building the Application

To build the application:

1 Click **Rebuild All** on the **Project** menu.

Net Express compiles the programs, and builds the executable file. Wait until the message "Rebuild Complete" appears in the Output window before continuing.

12.3.8 Testing the Screenset

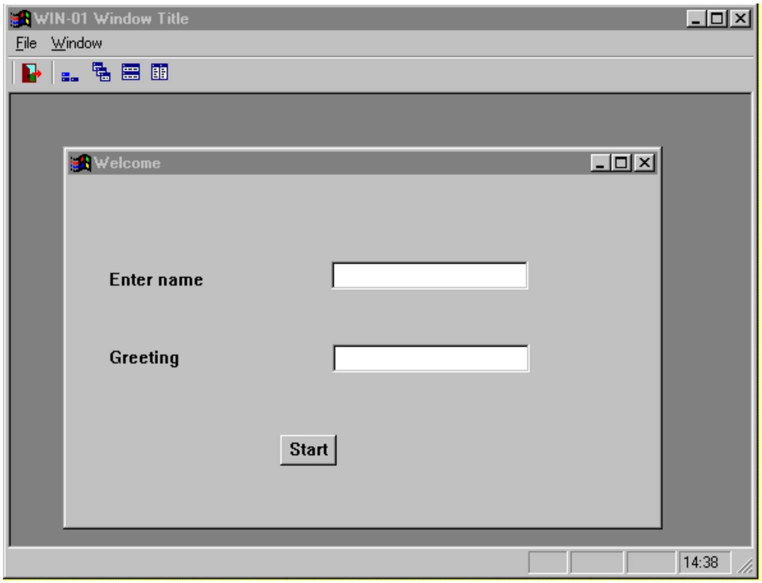
You can run the screenset within Dialog System, without the associated program. This enables you to evaluate and test its behavior, then alter it, if needed, by simply returning to the screenset definition. This is useful for prototyping your interface.

You had to build before doing this, as the toolbar/menu and status bar programs are executed during this testing.

1 Click **Run** on Dialog System's **File** menu.

The Dialog System window and the WIN-01 window disappear, and WIN-01 is displayed as it will appear at run time. Notice that it has a toolbar, menu bar and status bar, as you specified in the Wizard. WIN-02 is visible within it.

Figure 12-4. The Screenset as Displayed by the Run Function



The toolbar and menu bar created by the Wizard are created by generated control programs using the Net Express Class Library. For details, and to see how to amend the Wizard-created toolbar and menu bar, see the ***Dialog System User's Guide***. When you create menus yourself on your windows, you have the additional option of using Dialog System's own integrated system for creating menus. See the chapter *Windows and Objects* in the ***Dialog System User's Guide*** for details of this.

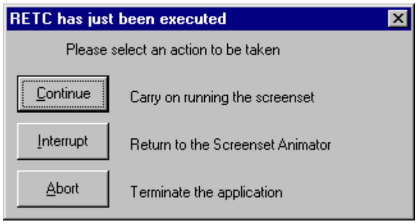
A small dialog box also appears at the bottom right of your screen, with **Debug** and **Abort** buttons. When you use this screenset testing facility in Dialog System, the screenset is run under control of the Screenset Animator. This is a debugger for dialog, similar in concept to the COBOL debugging features in the IDE. The dialog box enables you to do the following:

- Debug - display the dialog so you can use Screenset Animator's features such as single-stepping.
- Abort - stop testing the screenset and return to Dialog System.

2 Click the **Start** button you defined.

The dialog box shown in Figure 12-5 appears.

Figure 12-5. RETC dialog box



RETC has just been executed

Please select an action to be taken

Carry on running the screenset

Return to the Screenset Animator

Terminate the application

When you clicked **Start**, the dialog you wrote for this button was executed, and as you'll remember this includes a RETC command. Whenever a RETC is executed, Screenset Animator changes to this dialog box and you must choose what to do next:

- Continue - continue testing the screenset without animation.
- Interrupt - display the dialog so you can use Screenset Animator's features such as single-stepping.
- Abort - stop testing the screenset and return to Dialog System.

If ever you want to debug dialog when you are running a screenset within an application, there is a parameter to DSGRUN to enable the Screenset Animator. The Screenset Animator is documented in the Dialog System Help.

3 Click **Abort**.

4 Click **Exit** on Dialog System's **File** menu.

12.4 Before Continuing

In the next session, *Completing and Running Your Windows GUI Application*, you get this application working fully. If you're planning to go onto that session right away, you can keep the project open. If you want to take a break first, you can close the project, and open it again at the start of the next session. Or you can close Net Express entirely, with or without closing the project first.

13 Completing and Running Your Windows GUI Application

In the previous session you used Windows GUI Application Wizard and Dialog System to design a screenset and generate a COBOL program. You now edit the COBOL program generated and add your business logic. You then run the completed application, using the IDE.

You need to have read the chapter *Start Here for the Tutorials*, worked through the first session, *Using Net Express*, and worked through the previous session, *Creating a Windows GUI Application*, before you do this session.

You need to have installed Dialog System to do this session.

13.1 Overview

Windows GUI Application Wizard generates a COBOL source program in a **.cbl** file, containing code to input and output the screenset. You then edit this **.cbl** file to add your business logic to process the data from the screenset and create data to be output. Of course, you can simply add CALL statements to call other programs, and so keep your business logic completely separate from this screenset-handling program.

13.1.1 Structure of an Associated Program

An associated program typically has a main loop which does the following:

1. Calls DSGRUN
2. Tests the status codes returned from DSGRUN

3 Performs the business logic you've inserted

4 Repeats to call DSGRUN again to display the output and get more input

The program generated for you by Windows GUI Application Wizard has this structure. For another example, look later at the Customer demo in the directory **Net Express\DialogSystem\Demo\Customer**.

13.2 Preparation

If you have closed Net Express, open it as before. If you have closed the project, open it as described in the chapter *Start Here for the Tutorials*. If you use **Open**, the project to open is **Welcome** in the directory **Net Express\Base\Demo\Welcome**. By default the Open dialog box shows files of type **.app**, **.cbl**, and **.cpy**, so the file **Welcome** should be visible.

13.3 Sample Session

In this sample session you:

1. Edit the COBOL Program
2. Build the application
3. Run the application

13.3.1 Editing the COBOL Program

You could use any text editor to add your business logic to the COBOL program. We will use the IDE.

1. Double-click **Welcome.cbl** in the project window.

   A text window opens with the source of the program. You may need to resize the window to see the source clearly. Look at **Hide All**

Copyfiles on the **View** menu and ensure that the button-like icon beside it is shown depressed (if it isn't, click it).

Take a few minutes to look through the program to see what it does.

2 Put the cursor on the line **COPY "Welcome.CPB"** in the Working-Storage, click **Copyfile** on the **File** menu, then click **Show**.

The file **welcome.cpb** is the data block. You'll see your data items I-NAME and GREETING at the end of it.

The other copyfile, **ds-cntrl.mf**, is called the control block. It is supplied with Net Express and contains standard definitions needed by all Dialog System applications.

3 In the Program-Body Section, insert the following line above the period and below the statement **PERFORM Call-Dialog-System**:

```
string "Hello " I-Name delimited by size into Greeting
```

In a real application you would probably put a call here to a subprogram to process data from the screenset and calculate or look up data to display. You could use an EVALUATE statement performing actions depending on what is returned from DSGRUN. This would typically include a WHEN EXIT-FLAG-TRUE CONTINUE branch.

13.3.2 Building the Application

To build the application:

1 Click **Rebuild All** on the **Project** menu.

Net Express saves and compiles the program, and rebuilds the executable file. Wait until the message "Rebuild Complete" appears in the Output window before continuing.

13.3.3 Running the Application

To run the application:

1 Click **Run** on Net Express's **Animate** menu.

The Start Animating dialog box appears.

2 Click **OK** on the Start Animating dialog box.

The program runs, and displays the screenset.

3 Click in the **Enter name** field, type your name and click **Start**. Because you defined I-NAME-DISP to be 12 characters long, there is a limit of 12 characters.

The program puts your name after the "Hello " in the **Greeting** field, then redisplays the screenset.

4 Click in the **Enter name** field and delete your name, then type another name and click **Start**.

The program goes round the main loop again, displaying "Hello " followed by the new name.

5 Press **Esc** to close the window and end the program.

13.4 Before Continuing

Close the project.

If you’re planning to go straight on to another session, you can keep Net Express open.

Return to the *Tutorials Map* in the chapter *Start Here for the Tutorials* and choose which session to go on to next, depending on your interests.

Part 5: UNIX Option Tutorials

This part contains the following chapters:

- Chapter 14, “Deploying an Application on UNIX”

14 Deploying an Application on UNIX

You use the UNIX Option to deploy applications developed on Net Express to UNIX systems.

You need to have read the chapter *Start Here for the Tutorials* and worked through the first session, *Using Net Express*, before you do this session.

You need to have installed UNIX Option to do this session.

14.1 Overview

You use the UNIX Option to deploy an application that you have created on Net Express to a UNIX system that has Micro Focus COBOL for UNIX V3.1 or later installed.

In this sample session we start with an existing COBOL application. When you deploy the application on the UNIX system using the UNIX Option, the application is rebuilt on UNIX and you run it there. This sample application consists of just one COBOL program, which simply creates a one-record output file.

To see how to deploy a Web application on UNIX, after finishing this book you should see the online book ***Internet Applications***.

You cannot deploy a Windows GUI application, such as is created using Dialog System, on UNIX. However, the UNIX Option includes a character version of Dialog System for creating character user interfaces; this is described in the ***Dialog System Character Mode User's Guide***.

14.2 Preparation

Before you can run this session you must have the Server Control Program (SCP) installed on your UNIX system. It is needed for the **Publish** function of UNIX Option to work. For instructions on installing it, see the appendix *Installing SCP and Samba* in the ***UNIX Option User's Guide***.

You must have a connection to your UNIX system. Provide a login on the UNIX system, and configure the **.rhosts** file to enable transparent login without requiring a password. See your UNIX documentation for details on how to do this.

If you have closed Net Express, open it as before. If any project window or text windows are open, close them.

14.3 Sample Session

In this sample session you:

1. Create a project
2. Build the application and check it's portable
3. Use your PC as a UNIX terminal (optional)
4. Set up the options for publishing
5. Publish the application on the UNIX system
6. Run the application

14.3.1 Creating a Project

To create a project for your application:

1 Click **New** on the **File** menu, then select **Project** on the New dialog box and click **OK**.

2 Click **Project from an existing application** on the New Project dialog box.

3 Enter **Unixo** as the name of the project , and **Net Express\Base\Demo\Unixo** as the folder to contain the project, then click **Create**.

4 If this tutorial has been run previously, you will be asked whether to overwrite the existing project. Click **Yes**.

5 Click **Add Files**.

6 Select **Unixo.cbl** and click **Open**.

7 Click **Next**.

8 Click **Next** on the next dialog box.

9 Click **Finish** to create the project.

The project window for the project appears, and the project is scanned for dependencies. It currently contains just the COBOL program you are going to use to build your application.

14.3.2 Building the Application

When building this application, we need to set an option so we will be warned if it contains any features not portable to UNIX.

1 Click **Properties** on the **Project** menu.

2 In the **Project Directives** field, before the semicolon at the end, type **WARNINGS"2"** (leave a space after **ENSUITE"3"**). Then click **OK**.

3 Click **Rebuild** on the **Project** menu to compile the program.

A message warning that the intermediate code is not portable is displayed in the Output window. You may need to make the Output window bigger to see the message. You need to edit the file to change the non-portable syntax:

4 Double-click the error message in the Output window.

An edit window opens and the line containing the problem is marked by a red cross in the margin.

5 Edit the line so that OUT FILE is replaced by OUTFILE.

This line was flagged by the compiler because the filename contained a space. A filename that contains spaces can be used on Windows, but not on UNIX.

6 Click **Rebuild** on the **Project** menu.

Your program should now compile without warnings, and you can deploy the application to the UNIX system.

7 Close the text window displaying **Unixo.cbl**.

14.3.3 Using Your PC as a UNIX Terminal

Net Express includes terminal emulation software, PowerTerm, so you can use your PC as a UNIX terminal. While running this tutorial, you may find it convenient to use PowerTerm to give you a separate window open on your PC as a terminal onto your UNIX host system. If you don't intend to do this, either because you have other terminal emulation software or because you are using a separate UNIX terminal, you can skip this section.

1 Click **Terminal** on the **UNIX** menu.

2 Click **Connect** on PowerTerm's **Communication** menu.

3 On the Connect dialog box, ensure the Session Type is set to a type that is compatible with the network you're using. Enter the name of your UNIX host machine in the Host Name field. Then click **Connect**.

4 Log in to your UNIX system as usual.

14.3.4 Setting Up the Options for Publishing

Deployment is known as *publishing* in the UNIX Option. It consists of copying the files to the correct directories on the UNIX system and building the application there. Accessible from the Net Express menus is a tool, Publisher, that handles this process.

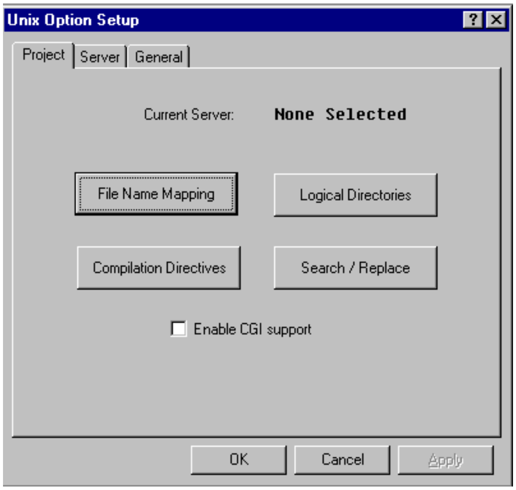
First you provide Publisher with details of the publish operation required:

1 Click **Setup** on the **Unix** menu.

2 Click **OK** on the Welcome screen. You should have already made the preparations it tells you about, when you were doing the section *Preparation* above.

This displays the form shown in Figure 14-1.

Figure 14-1. The Setup Dialog Box

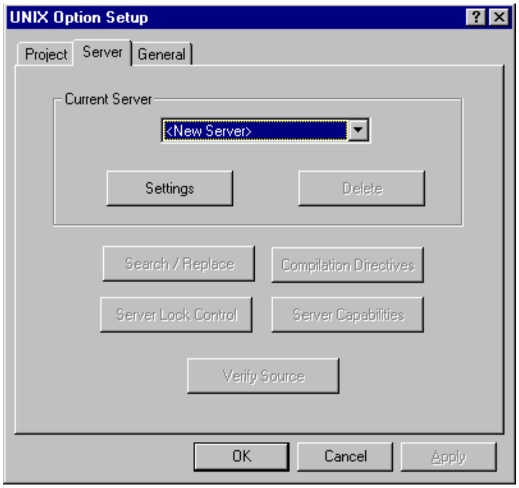


Current Server: None Selected

☐ Enable CGI support

3 Click the Server tab. This displays the form shown in Figure 14-2:

Figure 14-2. The Server Dialog Box



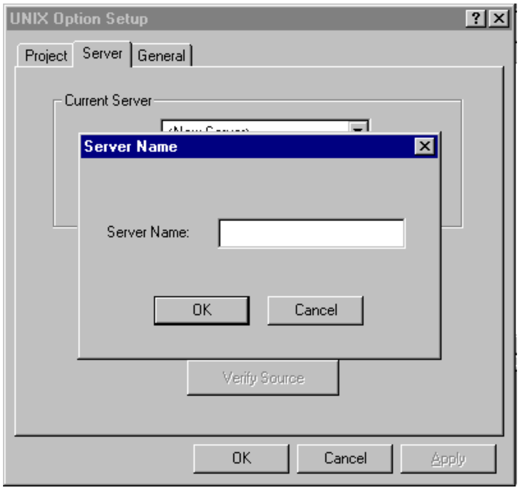
Current Server

<New Server>

The list box displays any servers that have currently been defined. If you select one of these, then any changes you make using the buttons in this tab are made to the selected server. The entry **New Server** enables you to define a new server. If there are no servers currently configured, **New Server** is selected by default and all of the buttons on the dialog except **Settings** are deactivated (see figure above).

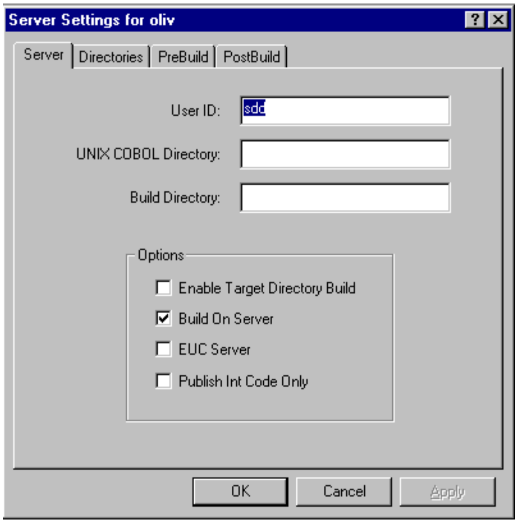
4 Click **Settings**. A dialog is displayed that enables you to specify a new server name:

Figure 14-3. Specifying a Server



5 Enter a valid Server name, and press **OK**. The following dialog box is then displayed:

Figure 14-4. Server Settings



Server Settings for oliv

User ID: sdd

UNIX COBOL Directory:

Build Directory:

Options

☐ Enable Target Directory Build

☑ Build On Server

☐ EUC Server

☐ Publish Int Code Only

6 In the **User ID** field, enter the user id you use to log onto your UNIX system.

7 In the UNIX COBOL Directory field, enter the name of the COBOL system directory on your UNIX system. You can find out this directory by entering **echo $COBDIR** on your UNIX system. If you leave this field blank it defaults to **/usr/lib/cobol**.

8 Decide the directory on your UNIX system to which you want to publish your application, and enter its name in the Build Directory field.

9 Click **OK**.

10 Click **OK** again.

14.3.5 Publishing the Application

To publish the application:

1 Ensure the directory you named in the Build Directory field in the Setup dialog box exists on your UNIX system.

2 Click **Publish** on Net Express's **UNIX** menu.

This builds the project on the PC, creates a file called **Makefile** to do the same thing on UNIX, copies the project files to the UNIX system and runs **Makefile** there.

If you have any problem, check your setup - use **Setup** on the **UNIX** menu again - to ensure you specified the server name, UNIX login name, and directory names correctly, and try again.

If this doesn't solve the problem, see the chapter *Tips and Troubleshooting* in the ***UNIX Option User's Guide***

3 On your UNIX machine, use any listing utility to look at the log file **Make.log**.

If the log file shows any errors, or if any of the files listed in the next section *Files Created* are not present in your UNIX build directory, check your setup and try again.

Publish publishes only files that have changed since the previous time you published the project. **Publish All** publishes all the files unconditionally. If you run this tutorial a second time, do not first delete the files from your build directory on UNIX. **Publish** will not know they have gone, and will not replace them. Alternatively, if you do delete the files from your UNIX directory, use **Publish All**.

14.3.6 Files Created

UNIX Option has now created the following file for you in your project directory on the PC and copied it into your build directory on UNIX (it has not added it to the project, as it is not used on the PC):

- **Makefile** is the script file to guide the build on UNIX.

It has also copied the following file from your project directory on the PC into your build directory on UNIX:

- **Unixo.cbl** is your COBOL program

The build directory also contains the following files created by the build process on UNIX:

- **Make.log** is a log showing the results of the build on UNIX

Several files created for use during the build process are also in the build directory on UNIX.

Note that filenames on UNIX are case sensitive.

14.3.7 Running the Application

To run the application

1. On your UNIX system, run the application in whatever way you normally run your UNIX COBOL applications; for example:

   ```
   cobrun Unixo.int
   ```

2. List your build directory (for, example, use an **ls** command), and check that file **OUTPUT** exists. Use any listing utility to check that it has just one record, containing the words "Example output".

   All that **Unixo.cbl** does is create this file, so if this file exists the application has been built and run correctly.

14.4 Before Continuing

Close the project.

If you’re using PowerTerm, logoff from UNIX in your usual way and click **Exit** on PowerTerm’s **File** menu.

If you’re planning to go straight on to another session, you can keep Net Express open.

Return to the *Tutorials Map* in the chapter *Start Here for the Tutorials* and choose which session to go on to next, depending on your interests.

Part 6: SQL Option Tutorials

This part contains the following chapters:

- Chapter 15, “DB2 Applications (SQL Option)”
- Chapter 16, “Maintaining a DB2 Database”

15 DB2 Applications (SQL Option)

Use SQL Option to include a DB2 application in a Net Express project.You can also maintain a DB2 database using an interactive tool, the SQL Wizard. You can compile, edit and debug your DB2 application in a DB2 environment on your PC. In addition, you can also compile, edit and debug your DB2 application against the mainframe itself, using the XDB Link technology.

You need to have installed SQL Option to do this session. We assume you are familiar with DB2 on a mainframe.

You also need to have read the chapter *Start Here for the Tutorials* and worked through the first session, *Using Net Express*, before you do this session.

15.1 Overview

The full name of this option is SQL Option for DB2. We will generally refer to it as SQL Option. It is a relational database that supports the debugging and testing of DB2 applications.

This session takes you through maintaining and running a DB2 application on your PC. It also shows how you can setup your PC to access the mainframe with the XDB Link technology, and run your DB2 application directly to the mainframe.

The demo application used in this session is a simple DB2 application that you might have downloaded from a mainframe. In this session, you get it working on your PC.

15.2 Preparation

If you have closed Net Express, open it as before. If any project window or other windows are open, close them.

15.3 Sample Session

In this session you:

- *Create a project*
- *Add files to the project*
- *Build the Project*
- *Start the SQL server*
- *Run the application*
- *Debug the application*
- *Close the SQL server*
- *Setup access to mainframe through XDB Link*

15.3.1 Creating a Project

To create the project:

1. Click **New** on the **File** menu, then select **Project** on the New dialog box and click **OK**.
2. Highlight Empty Project entry
3. Enter **sqldemo** as the name of the project
4. Click **Browse** and select a suitable folder to contain the project: for example **Net Express\mfsql\demo**
5. Click **Create**

6 Click **Properties** on the **Project** menu

7 Click **SQL Directives**; on the ESQL Preprocessor drop-down menu, select **XDB** and click **OK**

8 After Project directives are displayed, click **OK**

15.3.2 Adding Files to the Project

To add your files to the project:

1 Click **Add Files to Project**on the **Project** menu.

2 In the **Add Files** dialog box, open the folder **Net Express\mfsql\demo**. Ensure the **Files of Type** field is set to **All Files**.

3 Select **test1.cbl**, then click **Add**. The files are added to the project.

15.3.3 Building the Project

To build the project:

1 Click **Rebuild All**on the **Project** menu.

As you saw in the chapter *Using Net Express*, the correct compiler is automatically called for the source files - in this case, for the COBOL files. The SQL preprocessor is called by the COBOL Compiler for each COBOL file in the ECM environment.

The build finishes with "Rebuild complete".

15.3.4 Starting the XDB Server

To start the XDB Server on Windows NT:

1 Go to Control Panel, and double click **Micro Focus XDB Server for NX**

2 Click **Start** if the XDB Server is not running.

To start the XDB Server on Windows 95:

1 Click **SQL for DB2** on the **Tools** menu.

2 Click **Start Server**.

15.3.5 Running the Application

To run the application:

1 Click **test1.cbl** in the right-hand pane of the project window to select it.

2 Click **Run** on the **Animate** menu.

3 Click **OK**.

The application runs. The Application Output window appears and displays the screen output from the application. This application simply inserts an employee record into a database, fetches the surname and displays it, and then deletes the record. It displays messages showing its progress.

You can leave the Application Output window visible, as the next section uses it as well.

15.3.6 Debugging the Application

To debug the application:

1 Ensure **test1.cbl** is still selected, then click **Start Animating** on the **Animate** menu.

You see the same things happening on your screen as when running the application, but in addition a source view window appears, showing the source of **test1.cbl**. The application has not yet displayed its first screen in the Application Output window, because execution is paused on the first executable statement of the COBOL application. (This is the second statement in the Procedure Division, because the first, **EXEC SQL DECLARE...CURSOR**, is executed at compile time.

Notice that the source you see is the original source and not the source after preprocessing.

2 Click a few times.

This demonstrates that you single-step a DB2 application in the same way as any other. All the same debugging features are available for a DB2 application as for any other application. We will not debug this application further, but will simply complete the run.

3 Click .

The application completes without debugging, and exits. As before, the Application Output window shows the messages displayed by the application.

15.3.7 Closing the XDB Server

When you have finished, you shut down the XDB Server:

1 Click **SQL for DB2** on the **Tools** menu.

2 Click **Stop Server**.

15.3.8 Setup access to mainframe through XDB Link

To connect to the mainframe directly, without using the XDB server:

1 Use the Gateway Profile utility to log information about the mainframe DB2 location, as follows:

a Click **SQL for DB2** on the **Options** menu

b Click **XDB LINK**

c Login to local XDB server as user **INSTALL** with no password

d Click **REGISTER**

e Consult mainframe DBA and **HELP** information to fill out relevant fields.

2. Use the Options utility to define the location and protocol you wish to use, as follows:

   a. Click **SQL for DB2** on the **Options** menu

   b. Click **CLIENT**

   c. Click **Security** tab and enable **Client Security**

   d. Click **Connect** tab and pick **DRDA** as Communcation Protocol

   e. Ensure **Connect Location** and **System Location** are the value specified in **GPROF** utility

15.4 Before Continuing

Close the source view window and hide the Application Output window.

The Sqldemo project is used in the next session, *Maintaining a DB2 Database*. If you're planning to go onto that session, you can keep the project open.

If you want to take a break before going on to the next session, you can close Net Express.

Return to the *Tutorials Map* in the chapter *Start Here for the Tutorials* and choose which session to go on to next, depending on your interests.

16 Maintaining a DB2 Database

You can use the SQL Wizard to maintain a DB2 database. You can do things such as adding and updating tables and defining queries.

You need to have installed SQL Option to do this session. We assume you are familiar with SQL and with DB2 on a mainframe.

You also need to have read the chapter *Start Here for the Tutorials* and worked through the first session, *Using Net Express*, and the session *DB2 Applications (SQL Option)*, before you do this session.

16.1 Overview

This session shows you how to use the SQL Wizard to define and maintain tables and queries in an existing DB2 database.

In this session you use the Wizard to add a Pensioner table to the supplied database called Tutorial, and to create and run a query on this table.

Although the SQL Wizard and the sample database used in this session are not related to any project, you must have a project loaded to enable the relevant IDE menu functions. It must be a project where the **SQL** check box was checked. We will use the project created in the chapter *DB2 Applications (SQL Option)*.

16.2 Preparation

This demo uses the project you created and built in the chapter *DB2 Applications (SQL Option)*.

1. If you have closed Net Express, open it as before. Close any project windows and any other windows that are open.
2. Open the project **sqldemo** in any of the ways described in the chapter *Start Here for the Tutorials*.
3. Start the SQL Server as described in the section *Starting the XDB Server* in the chapter *DB2 Applications (SQL Option)*.
4. After verifying that the XDB Server is running, please close this window.

16.3 Sample Session

In this session you:

- *Start the SQL Wizard*
- *Create a new table*
- *Define the primary key*
- *Define a secondary index*
- *View the SQL*
- *Finish creating the table*
- *Add data to the table*
- *Create a query*
- *Run the query*

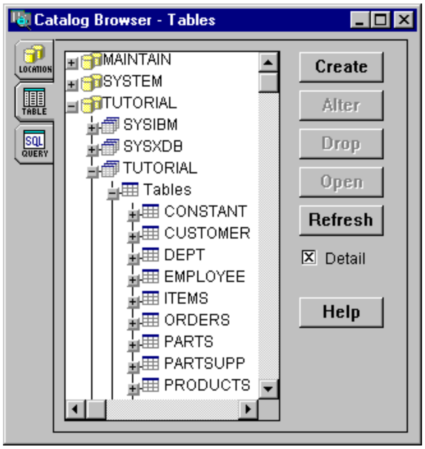
16.3.1 Start the SQL Wizard

To start the SQL Wizard:

1 Click **SQL for DB2** on the **Tools** menu, then click **SQL Wizard**.

The SQL Wizard appears, containing a Catalog Browser dialog box as shown in Figure 16-1.

Figure 16-1. The Catalog Browser Dialog Box.



16.3.2 Create a New Table

To create a table:

1 Click the **TABLE** tab to ensure it’s selected.

2 Click **Create**.

3 Enter **PENSIONERS** in the Table Name field.

The bottom part of the dialog box is for defining the fields to appear in the table. We will define three fields.

4 Click in the empty field under the column heading **Field Name** and enter the following:

Field Name	PensionerId
Data Type	Char
Length	5
Not Null	Unique

5 Press **Tab** until a new new field appears below the first, and enter the following:

Field Name	PensionerAddress
Data Type	Varchar
Length	100
Not Null	True

6 Press **Tab** again to add another new field, and enter the following:

Field Name	PensionerAge
Data Type	Varchar
Length	3
Not Null	False

16.3.3 Defining the Primary Key

To define the primary key for the table:

1 Click **Create**, just to the right of the **Indexes and Keys** field.

The Create Index dialog box appears.

2 Type **PensionerId** in the Index Name field.

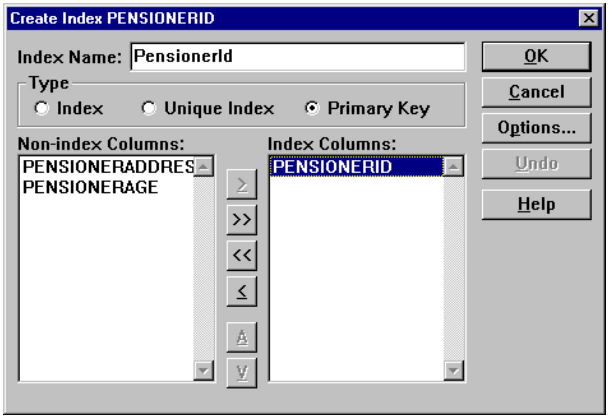
3 Click **PensionerId** in the Non-index Columns listbox and click >

This moves this field across to the list of Index Columns.

4 Click **Primary Key** under **Type**.

You should now have the dialog box shown in Figure 16-2.

Figure 16-2. The Create Index Dialog Box for the Primary Key



Create Index PENSIONERID

Index Name: PensionerId

Type
○ Index ○ Unique Index ● Primary Key

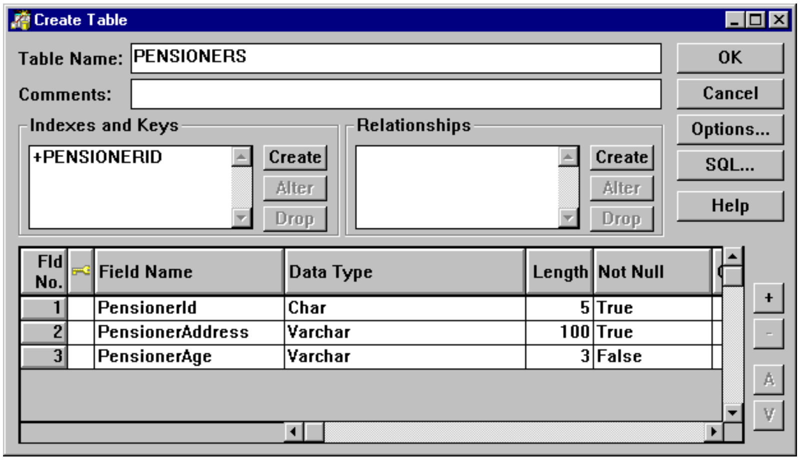
Non-index Columns:
PENSIONERADDRES
PENSIONERAGE

Index Columns:
PENSIONERID

5 Click **OK**.

This adds the primary key **+PensionerId** to the list of **Indexes and Keys**. You should now have the dialog box shown in Figure 16-3.

Figure 16-3. The Create Table Dialog Box



Create Table

Table Name: PENSIONERS

Comments:

Indexes and Keys
+PENSIONERID

Relationships

Fld No.		Field Name	Data Type	Length	Not Null
1		PensionerId	Char	5	True
2		PensionerAddress	Varchar	100	True
3		PensionerAge	Varchar	3	False

16.3.4 Define a Secondary Index

To define a secondary index for the table:

1 Click **Create**, just to the right of the **Indexes and Keys** field.

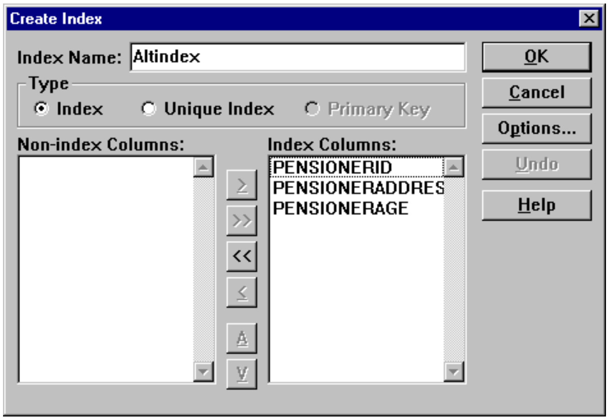
The Create Index dialog box appears again.

2 Type **Altindex** in the Index Name field.

3 Click >> to add all the fields as Index Columns.

You should now have the dialog box shown in Figure 16-4.

Figure 16-4. The Create Index Dialog Box for the Secondary Index



Index Name: Altindex

Type: ● Index ○ Unique Index ○ Primary Key

Index Columns:
PENSIONERID
PENSIONERADDRES
PENSIONERAGE

4 Click **OK**.

This adds the index **+AltIndex** to the list of **Indexes and Keys**.

16.3.5 Viewing the SQL

To view the SQL which will be used to create this new table:

1 Click **SQL...**.

2 Close this view window.

16.3.6 Finishing Creating the Table

To finish creating the table:

1 Click **OK**.

2 Click **Refresh**.

The new table called PENSIONERS has been added to the list of tables in the database called Tutorial. You may have to pull the slider down to see it.

3 Expand the table details by clicking the "+" alongside the table name PENSIONERS. Expand the **Columns** and **Indexes** in the same way.

16.3.7 Adding Data to the Table

To store some data in this table:

1 Click **PENSIONERS** in the Catalog Browser, then click **Open**.

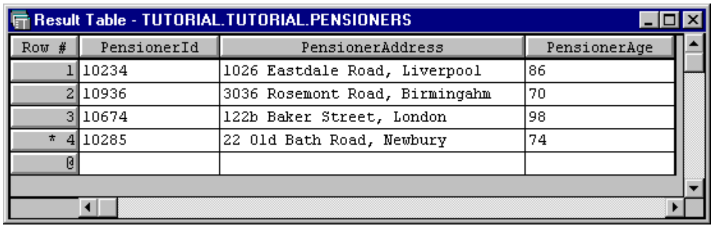
The Result Table window appears. This table displays the records in the table, but there are none at present.

2 Click **Allow Editing** on the **Record** menu.

This enables you to enter records into the table.

3 Enter a few record details as shown in Figure 16-5.

Figure 16-5. The Pensioners Result Table



Result Table - TUTORIAL.TUTORIAL.PENSIONERS

Row #	PensionerId	PensionerAddress	PensionerAge
1	10234	1026 Eastdale Road, Liverpool	86
2	10936	3036 Rosemont Road, Birmingahm	70
3	10674	122b Baker Street, London	98
* 4	10285	22 Old Bath Road, Newbury	74
@			

4 Close the Result Table window.

16.3.8 Creating a Query

To create a query against this table:

1 Click the **QUERY** tab to ensure it's selected.

2 Click **New**.

The Add Table dialog box appears.

3 Click **PENSIONERS** in the Table Name field, and click **Add**.

The **PENSIONERS(P1)** window appears.

4 Click **Done**.

5 On the **PENSIONERS(P1)** window, click **PensionerId** and **PensionerAddress** and see how these are added to the list.

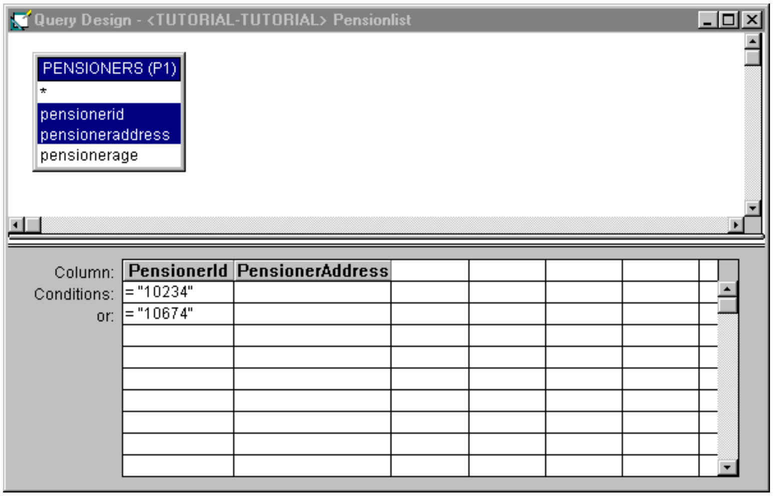
6 Enter **10234** in the **Conditions** field directly below **PensionerId** and press **Tab**.

Notice how your entry changes

7 Enter **10674** in the **or** field directly below **PensionerId** and press **Tab**.

The produces the window shown in Figure 16-6.

Figure 16-6. The Query Design window.



Column:	PensionerId	PensionerAddress
Conditions:	="10234"	
or:	="10674"	

8 Click **Save As** on the **File** menu, enter **Pensionlist** as the Query Name, and click **OK**.

9 Close the **Query Design** Window.

10 Click **Refresh**.

The new query called Pensionlist has been added to the list of queries in the database called Tutorial.

16.3.9 Running the Query

To run the query:

1 Click **Pensionlist** in the list, then click **Batch**.

The result table appears, showing records that match the criteria you specified in the query.

2 Close the Result Table window.

16.4 Before Continuing

Close the SQL Wizard and the SQL server.

Close the project. If you want to take a break before going on to the next session, you can close Net Express.

Return to the *Tutorials Map* in the chapter *Start Here for the Tutorials* and choose which session to go on to next, depending on your interests.

Part 7: Appendices

This part contains the following chapters:

- Appendix A , “Windows Tips”
- Appendix B , “More Features”
- Appendix C , “Web Applications”
- Appendix D , “Configuring Your Web Server”

A Windows Tips

In this documentation we assume you have a basic knowledge of how to use your Windows operating system. This appendix briefly describes some techniques that are commonly used in the operating system interface and in applications.

A.1 Standard Buttons on a Window

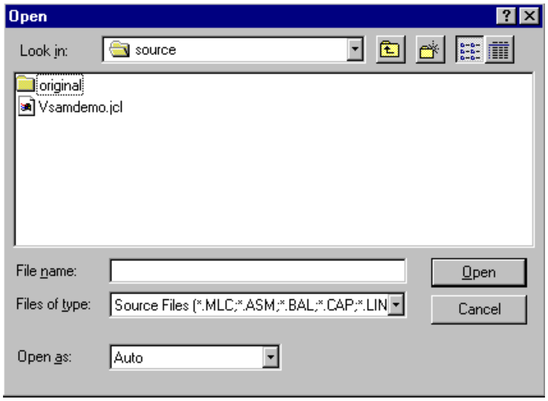
Most windows have up to three buttons at the top right-hand corner, on the window’s title bar. Buttons that can appear here are:

×	Close	Closes the window and whatever is running in it.
_	Minimize	Reduces the window to simply an entry on the task bar at the bottom of your screen. This saves spaces on the screen - the window and whatever is running in it are still there.
□	Maximize	Expands the window to fill the screen. Other things on the screen are still there, but hidden behind this window.
🗗	Restore	Reduces a window from full screen to occupy only part of the screen.
?	Context Help	Changes the mouse pointer to a question mark. Clicking anywhere with this pointer brings up brief help on the item you’ve clicked on, instead of executing the function normally invoked by that item.

A.2 Changing Folder

Often when you need to select a file, you are presented with a dialog box showing all the files and folders that are in a particular folder. The dialog box is said to be open at this folder. Examples are the commonly used **Open** dialog box, shown in Figure A-1, and the **Add Files to Project** dialog box.

Figure A-1. The Open dialog box



Look in: source

original
Vsamdemo.jcl

File name:

Files of type: Source Files (*.MLC;*.ASM;*.BAL;*.CAP;*.LIN

Open as: Auto

Some or all of the following methods may have been implemented for moving around the hierarchy of folders:

- To open a folder that is within the open folder, double-click it.
- To move to the drive or folder containing the open folder, click 📂.
- To move to a drive or folder, type its path in the Filename field and click whatever button normally executes the function (for example **Open** or **Add**). If you end the path with the name of a file, this will execute the function.
- To see a full list of the hierarchy of drives and folders, click the down arrow ▾ to the right of the **Look In** field, which displays the name of the open folder. You can then change folder by clicking the folder you want. Merely moving the mouse pointer changes the selection, so to close this list without changing folder, either check the highlight hasn't moved or click somewhere outside the list.

A.3 Manipulating Folders

In the kind of dialog box described in the section *Changing Folder*, you can usually carry out functions like creating folders, and renaming and deleting folders and files.

This means you can do these things from within whatever software you happen to be using, which is sometimes more convenient than going into another Windows session. For example, if you're using the IDE, and you want to create a folder, click **Open** on the IDE's **File** menu and do it via the Open dialog box. Then simply click **Cancel** to close the dialog box.

- To create a new folder within the open folder, click 📁.
- For other functions, right-click the name of a folder within the open folder and use the popup menu.

A.4 Tree Views

Hierarchical data, for example a list of directories and their subdirectories, is often shown in a tree view. This is a list of the top level, with a "+" on the left of each item. Click the "+" to expand the entry to show the items immediately subordinate to it. The "+" then changes to a "-", which you click to compress ("close") that part of the tree again. Or click an item and type "*" (asterisk) on your numeric keypad, to show all items subordinate to the item.

A good example is Windows Explorer (right-click **My Computer** on your desktop, and then click **Explore**), where the left-hand pane is a tree view showing the drives and folders on your computer.

A.5 Showing/Hiding File Extensions

In the lists of files shown in (for example) the **Open** dialog box or Windows Explorer, file extensions that Windows recognizes may not be shown. Instead, the file type may be indicated by an icon or a description. If you want to change your Windows settings so that these the extensions are shown, you can do so via Windows Explorer, as follows:

1 Right-click **My Computer** on your desktop, and then click **Explore**.

2 Click **Folder Options** on the **View** menu.

3 Click the **View** tab.

4 Make sure **Hide file extensions for known file types** has no check mark - click in the check box if necessary to delete it.

5 Click **OK**.

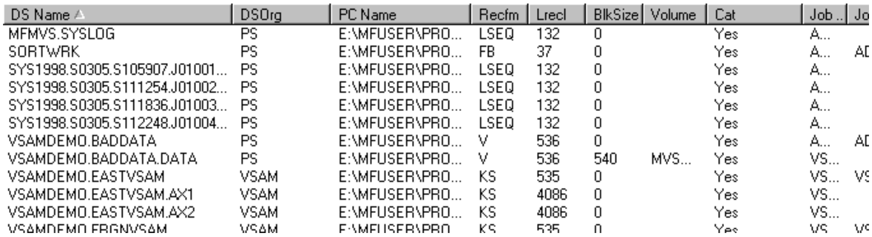
You can now close Windows Explorer.

If you have a dialog box open while you are doing this, you may have to close it and reopen it before you see the change.

A.6 Column Headings in a Container

Sometimes you see a list with several columns, with column headings on a gray background. In Windows such a list is called a *container*. Figure A-2 shows part of such a list.

Figure A-2. Part of a Container



DS Name	DSOrg	PC Name	Recfm	Lrecl	BlkSize	Volume	Cat	Job ..	Jo
MFMVS.SYSLOG	PS	E:\MFUSER\PRO...	LSEQ	132	0		Yes	A...	
SORTWRK	PS	E:\MFUSER\PRO...	FB	37	0		Yes	A...	AD
SYS1998.S0305.S105907.J01001...	PS	E:\MFUSER\PRO...	LSEQ	132	0		Yes	A...	
SYS1998.S0305.S111254.J01002...	PS	E:\MFUSER\PRO...	LSEQ	132	0		Yes	A...	
SYS1998.S0305.S111836.J01003...	PS	E:\MFUSER\PRO...	LSEQ	132	0		Yes	A...	
SYS1998.S0305.S112248.J01004...	PS	E:\MFUSER\PRO...	LSEQ	132	0		Yes	A...	
VSAMDEMO.BADDATA	PS	E:\MFUSER\PRO...	V	536	0		Yes	A...	AD
VSAMDEMO.BADDATA.DATA	PS	E:\MFUSER\PRO...	V	536	540	MVS...	Yes	VS...	
VSAMDEMO.EASTVSAM	VSAM	E:\MFUSER\PRO...	KS	535	0		Yes	VS...	VS
VSAMDEMO.EASTVSAM.AX1	VSAM	E:\MFUSER\PRO...	KS	4086	0		Yes	VS...	
VSAMDEMO.EASTVSAM.AX2	VSAM	E:\MFUSER\PRO...	KS	4086	0		Yes	VS...	
VSAMDEMO.FRGNVSAM	VSAM	E:\MFUSER\PRO	KS	535	0		Yes	VS	VS

Some or all of the following methods may have been implemented for rearranging the layout:

- To make a column wider or narrower, click the line separating its column heading from the one to its right, and drag it to left or right. Note that columns can be reduced to nothing by doing this - if a column you expect is not present, drag the existing columns - you may uncover the missing one.
- To adjust a column automatically to the width of its widest entry, double-click the line separating its column heading from the one to its right.
- To see a list of columns in the table, right-click a column heading. Click entries in this list to include or exclude columns in the display.
- To rearrange the order of columns, click a column heading and drag it onto another. The column will move to be immediately to the right of the one you dragged it onto.
- To sort on a particular column, click that column's heading.
- To reverse the order of entries, click the column heading of the field that the list is already sorted on. Small arrows by the column heading may indicate whether the list is in forward △ or reverse ▽ order.

A.7 Selecting Items in a List

Often you're presented with a list, and you need to select one or more items in the list before clicking a button to perform some action on the selected item or items. Examples are listboxes, the commonly used **Open** dialog box, shown in Figure A-1 above, and the **Add Files to Project** dialog box.

To select a single item you simply left-click on it, but in many cases you can also select multiple items. Some or all of the following methods may have been implemented:

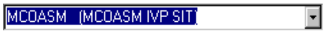
- Hold **Ctrl** while clicking multiple individual items. If you click any without holding **Ctrl**, you deselect all you've selected so far.

- Click one item, then hold down **Shift** while clicking a second. This selects both and all in between.
- Click the down arrow while holding **Shift**. This selects successive entries.
- Type **A** while holding **Ctrl**. This selects the whole list.

A.8 Dropdown Lists

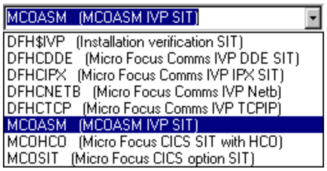
Sometimes you see a field with a down-arrow ▾ on the right. The arrow indicates a dropdown list, listing possible values you could enter into the field. Figure A-3 illustrates a field with a dropdown list, and Figure A-4 shows the same field with its dropdown list open

Figure A-3. A Field with a Dropdown List



MCOASM (MCOASM IVP SIT)

Figure A-4. The Same Field with its Dropdown List Open



MCOASM (MCOASM IVP SIT)

DFH$IVP (Installation verification SIT)
DFHCDDE (Micro Focus Comms IVP DDE SIT)
DFHCIPX (Micro Focus Comms IVP IPX SIT)
DFHCNETB (Micro Focus Comms IVP Netb)
DFHCTCP (Micro Focus Comms IVP TCPIP)
MCOASM (MCOASM IVP SIT)
MCOHCO (Micro Focus CICS SIT with HCO)
MCOSIT (Micro Focus CICS option SIT)

Some or all of the following methods may have been implemented for entering a value into the field:

- Simply type the value you want.
- Click the ▾, then in the dropdown list click the value you want.
- In the field, type the first letter of the value you want. The first value in the list that starts with that letter will be inserted in the field. In some implementations, you can type further letters to

identify more closely the value you want; in other implementations, any subsequent letter you type replaces the earlier one and is taken as the first letter of the required value.

A.9 Popup Menus

Often if you right-click on a window or object on the screen, a menu appears showing functions that are appropriate for that window or object. This is called a popup menu or context menu.

A.10 Cut and Paste

In many applications, you can move an item as follows:

1. Highlight the item with your mouse. If it’s a filename, a single click normally does this. If it’s text, drag your mouse button along it to highlight the part you want.
2. If you want to copy the text, press **Ctrl+C**. If you want to move it, deleting it from its present position, press **Ctrl+X**.
3. Highlight the place where you want to put the item. If you want to insert the item, click there. If you want to replace existing text, drag your mouse pointer along the text to be replaced to highlight it.
4. Click **Ctrl+V**.

Ctrl+C, **Ctrl+X**, and **Ctrl+V** are the shortcut keys for functions **Copy**, **Cut**, and **Paste** usually available on menus . It’s usually quicker to use the shortcut keys. A few applications may use different shortcut keys for these functions.

You may find you can cut and paste text from the online version of this ***Getting Started*** book into the IDE. This can save typing when the book gives a path to enter into a field.

A.11 Drag and Drop

In many applications, you can move or copy an item by "dragging and dropping" it, or its icon in a list or tree view. You put your mouse pointer on it, then hold down the left mouse button while you move your pointer to the place you want to put the item. This is called "dragging" the item, and "dropping" it at the place where you release the button.

This might copy the item, giving you a copy in the new location but leaving the original where it was; or move it, deleting it from the original location. It depends on the application.

Often you can transfer an item between two lists or tree views in this way. A good example is Windows Explorer, where you can even have several copies of the same tree view, by opening Windows Explorer several times. You can drag a file to its new location in another copy of the tree view, if the whole tree view is too big for you to see both the source and target locations on your screen in one tree view.

Often you can operate on a group of items. You select them as described in the section *Selecting Items in a List* above, and then press down the left mouse button on any one of them and drag. Be careful not to release the button till your mouse pointer is on the target location, or you'll deselect them all.

A.12 Resizing and Docking Windows

The ways you can change the shape and size of windows, and reposition the panes within a complex window, are best shown through a short tutorial using the IDE.

1 Place your mouse pointer on the gray border at the top of the Output window.

   The mouse pointer changes shape depending on where it is. It's not possible to illustrate the shape, as the shapes used depend on your Windows settings. On this border it has its standard shape (often an arrow). If it's lower, within the Output window itself, or higher, where the border of the Output window meets the border of the main pane above, it may be different.

2 Hold down your left mouse button, and drag it up and to the right.

An outline representing the Output window moves with the pointer. As the pointer crosses the right-hand border of the IDE, this outline changes shape.

Notice that the tip "Hold **Ctrl** to prevent docking" appears at the bottom of the IDE.

3 Release the mouse button.

The Output window attaches itself to the right-hand border. This is called docking.

4 Place your mouse pointer on the line dividing the left-hand border of the Output window from the border of the main pane, and drag it to the left till it’s a convenient width.

Thus you can widen the pane in the usual way.

5 Place your mouse pointer on the gray border at the left of the Output window, and drag it to the right until it is outside the IDE.

This makes the Output window into a separate window outside the IDE. It is said to be undocked, or floating.

6 Put your mouse pointer on the title bar of the Output window, and drag it back to its original position so that it docks there.

B More Features

This appendix lists some features that may be of interest in addition to the major features described in the chapter *Overview*. It is not a complete list of the features in Net Express.

Features included mainly to help you migrate from other COBOL systems are in a second list at the end of this appendix.

Additional software supplied to complement Net Express is described in the chapter *Additional Software*.

B.1 Descriptions of Features

Features include both development tools and features for use in your applications.

B.1.1 List of Features

Features are listed alphabetically.

B.1.1.1 Btrieve Interface

The interface to Novell's Btrieve file-handling system enables you to use COBOL file I/O syntax to access Btrieve files. You can use Btrieve files and files in the format native to Net Express in the same program.

B.1.1.2 Callable File Handler

The Callable File Handler is an interface to Net Express's file handler enabling you to call it from your program using CALL statements. This gives you low-level control over files of all COBOL organizations, so you can write sophisticated file and database handling programs. You can also use it to access COBOL format files from other languages, such as C or Assembler.

B.1.1.3 Callable Rebuild

Callable Rebuild is an interface to the Rebuild utility so you can call it from your program using CALL statements (see the entry for *Rebuild*).

B.1.1.4 Callable Sort

The Callable Sort Module is a stand-alone sort routine which enables you to sort and reorder data files. It is faster than the default COBOL sort mechanism. The call interface provides greater flexibility in sorting data and enables you to substitute alternative sort modules.

B.1.1.5 Cbllink

Cbllink is a high-level command-line interface to the system linker. It is normally called automatically for you by your application's project.

B.1.1.6 Client/Server Bindings

Client/Server Bindings provides a standard mechanism for communicating over networks using a variety of protocols. A program written to work with it will work with any of the protocols it supports, simply by interchanging modules.

B.1.1.7 COBOL System Library Routines

The COBOL system library routines are a set of routines you can call from your program, providing many operations not available in the COBOL language itself.

B.1.1.8 Command Prompt

The Net Express Command Prompt is a command-line session with the Net Express environment set up. You can run some Net Express functions from the command prompt. From the Windows **Start** menu, click **Programs**, then **Micro Focus Net Express**, then **Net Express Command Prompt**.

B.1.1.9 Common Gateway Interface (CGI) Support

CGI Support enables you to run programs that conform to the CGI, ISAPI, or NSAPI standard, so that they can be invoked from a remote Web browser to run under the control of Web server software.

B.1.1.10 Embedded HTML

Embedded HTML is a COBOL extension enabling you to write HTML in the Procedure Division of your program so as to create and display a Web page.

B.1.1.11 Fileshare

Fileshare provides rapid network I/O by compressing I/O requests into packets and sending them across network lines, so that the file processing is done on the server that contains the files. It can also link several files into a logical database. You can set up a recovery log when accessing that database, providing a high level of data integrity. It has transaction logging which enables your user to delay writing changes to files until all the information is complete. You can secure these changes with a COMMIT or cancel them with a ROLLBACK.

B.1.1.12 Integrated Preprocessor Support

Integrated Preprocessor Support is an extension to the Compiler. It enables the Compiler to invoke a user-defined language processor to convert non-COBOL syntax to COBOL syntax. When debugging, you see the original source, as it is before statements are altered by the preprocessor.

For more information on writing and using preprocessors, click **Help Topics** on the **Help** menu. With the Net Express online help displayed, click the **Index** tab and type **preprocessor**, then click **Display** and select one of the available topics.

B.1.1.13 Microsoft Transaction Server Support

Microsoft Transaction Server is a component-based transaction processing system for developing, deploying and managing Internet and intranet applications. You can create Transaction Server components in COBOL using Net Express.

B.1.1.14 Multithreading

Multithreading enables you to set off several concurrent paths of execution in the same application.

B.1.1.15 National Language Support

National Language Support (NLS) enables your program to adapt itself automatically at run time to the character set, currency symbol, and editing symbols appropriate to your user's country. It ensures correct collation and folding of national (for example, accented) characters, and provides library routines to fetch messages in the appropriate national language from a message file.

B.1.1.16 Object-oriented COBOL Syntax

Object-oriented syntax is a set of extensions to COBOL so that you can use object-oriented (OO) programming. It's supported by the OO Class Library, a set of predefined functions for use from OO programs.

B.1.1.17 Object Request Broker (ORB)

An Object Request Broker (ORB) is the middleware that establishes the client/server relationship between components in a distributed computing environment. When a client calls on the services of a distributed component, the ORB intercepts and processes the call. The client does not need to be aware of where the component is located, the programming language it was written in, the operating system it is running on, or any other aspect that is not a part of the component's interface.

B.1.1.18 ODBC Drivers

The ODBC drivers enable applications to interface to databases that conform to the Open Database Connectivity standard (ODBC). Some of Net Express's demonstration programs use them. Applications created using the Data Access Wizard also use them.

The driver for Microsoft Access databases is supplied by Microsoft. The others are supplied by DataDirect Technologies, formerly part of MERANT.

Note: You are not authorized to sublicense or distribute the ODBC drivers contained in Net Express to any third parties. For further information about licensing the ODBC drivers see the section *ODBC Drivers* in the chapter *Licensing and Deployment*.

B.1.1.19 OLE Automation Support

Object Linking and Embedding (OLE) is a feature of Microsoft Windows which enables one application to load and send messages to any other application registered with the operating system as an OLE object. OLE automation support is a feature of Net Express which enables you to send messages to OLE objects from Net Express programs and classes.

You use the Class Wizard to generate skeleton Object COBOL classes, including classes that provide OLE automation or Microsoft Transaction Server (MTS) support to an application. It also generates any other required files for OLE, such as trigger files, registry files and type library files.

You use the Method Wizard to add skeleton methods to Object COBOL classes. It gives you the opportunity to enter the parameter and return types, whether the method supports MTS, and it adds any type library information required for OLE classes.

B.1.1.20 OpenESQL

OpenESQL is a preprocessor provided with Net Express which enables you to access ODBC data sources from a COBOL program by embedding SQL statements in your source code.

B.1.1.21 OpenESQL Assistant

The OpenESQL Assistant is a wizard that generate SQL statements for inclusion in your COBOL program.

B.1.1.22 PVCS

PVCS is a popular Source Code Control System (SCCS). It gives multiple users controlled access to source code stored centrally on a server. It stores changes in the form of deltas so that it can re-create successive versions of a file.

B.1.1.23 Rebuild

The Rebuild utility is a tool for maintaining indexed files. You can use it to improve performance by reorganizing keys and data, to give a new key structure to an existing file, to recreate a corrupted index, and to convert sequential and relative files to indexed format. It uses 64-bit addressing, so it can handle big files.

B.1.1.24 Sort Utility

The Sort utility gives you a file-sorting facility which you can call from the operating system prompt. It uses the Callable Sort Module.

B.1.1.25 Source Code Control System Support

The Source Code Control System (SCCS) support enables you to access an SCCS via the menus of the Net Express IDE when you have a project open. SCCS's that it can be used with are PVCS, MS Visual SourceSafe and VisualAge TeamConnection. A version of PVCS is included in Net Express.

B.1.1.26 WebSync link

WebSync is a link within the Net Express help to access a Micro Focus Web server. You can use it to get support, updates, and information on Net Express.

B.1.1.27 Win32 Native API programming

You can call Windows 95 or Windows NT operating system routines if you want very low-level control.

B.2 Mainly for Migration

The following additional features are included mainly to help migration to Net Express. The online book ***Migration Cookbook*** recommends alternatives to use for new applications.

B.2.1 List of Features

Features are listed alphabetically.

B.2.1.1 COBSQL

COBSQL is an integrated pre-processor designed to work with COBOL precompilers supplied by relational database vendors. It is intended for use with Oracle Pro*COBOL Version 1.8 and Sybase Open Client Embedded SQL/COBOL Version 11.1. You should use COBSQL if you are already using either of these precompilers with an earlier version of a Micro Focus COBOL product and want to migrate your application(s) to Net Express, or if you are creating applications that will be deployed on UNIX platforms and need to access either Oracle or Sybase relational databases.

For any other type of embedded SQL application development, we recommend that you use OpenESQL.

B.2.1.2 Dialog Editor

Dialog Editor is a screen painter for creating windows and dialog boxes. The same tool is in Micro Focus's Visual Object COBOL V1.0. Applications created using that system can be run and further developed using the Dialog Editor in Net Express.

B.2.1.3 On-line Help System

The On-line Help system (Hyhelp and Ohbld) is for creating character-mode on-line documentation and displaying it on the screen. It has facilities for extensive navigation around the information. Your programs can call it to present help with a minimal amount of programming. It can produce both character and graphical on-line documentation, and native documentation on Windows.

B.2.1.4 Panels

Panels is an application programming interface (API) that your application can call to use windowing on a character-mode screen. It provides overlapping windows, separate or synchronized updating of text and attributes, scrolling of text in a window, and popup or pulldown menus.

B.2.1.5 Screen and Keyboard Handler (Character Mode)

The Screen and Keyboard Handler, often known as Adis, is a module that provides run-time support for the Enhanced Accept/Display feature, which gives full-screen character mode ACCEPT and DISPLAY verbs using a Screen Section in the Data Division. (Don't confuse this with the extended ACCEPT and DISPLAY for handling Web pages.)

B.2.1.6 Screen and Keyboard Configuration Tools

The screen and keyboard configuration tools (Adiscf and Keybcf) are two utilities for configuring character-mode screen and keyboard handling to your own environment and needs, for example for compatibility with other COBOL systems.

B.2.1.7 Screen Section

The Screen Section is a section in the Data Division where you can define forms for display on a character-mode screen.

B.2.1.8 Windowing Support

Windowing Support consists of COBOL syntax which enables you to draw character-mode lines and boxes on the screen and create virtual windows on a physical screen. The syntax also allows underlying displays to be kept and restored. It is compatible with ACU COBOL.

C Web Applications

Net Express can be used for developing many kinds of COBOL application, from the most traditional to the most modern, but it is particularly suited to *client/server* applications, in which most data is kept on a central computer (the server) and accessed from many other computers (the clients). In such an application, queries are entered at a client, and software running locally forwards them to the server where software interrogates or updates the data and returns a response.

Nowadays most of the world’s computer networks are connected together to form the *Internet*. A major use of the Internet is the *World Wide Web*, often called simply the *Web*. Computers known as *Web servers* store files known as *Web pages*, which can contain text and graphics, and run software to make them accessible across the Internet. Any computer running suitable software, called a *Web browser*, can access them. A Web page with fields for the user to fill in is called a *Web form*. Web server software can run programs (called *server-side* programs, or sometimes *CGI programs* after the most popular standard, Common Gateway Interface) to process data, and a Web browser can display forms sent to it by a Web server.

This is one example of a client/server application, and Net Express is very suitable for designing Web business forms and server-side programs.

A Web page includes formatting commands in a language called *HTML* (HyperText Markup Language). If you’re creating a Web page by hand you include these commands, but if you use Net Express they are included for you.

There are several popular Web browsers on the market, and a Web page may look slightly different in different browsers.

Many companies and other organizations have their own *intranet*, with Web servers accessible only from computers within the organization. An application meant for an intranet works exactly like one intended for the World Wide Web, and people tend to use the term *Web application* for either.

Also, many organizations have a more traditional network. A file on a network server can be accessed from a network client as if the disk on the server were a disk on the client machine. Net Express can also be used to create traditional client/server applications. Since the Internet is not involved, you use Windows-type graphical objects like windows and dialog boxes, instead of Web pages. We call these *Windows GUI applications* to distinguish them from Web applications. (You could of course use character-mode text input and output if you wished.)

Web forms, windows and dialog boxes can have objects on them such as pushbuttons, entry-fields, checkboxes, and so on. These are called *controls*.

If you want a longer and more detailed introduction to the World Wide Web, see the chapter *Introduction to the World-Wide Web* in the online book ***Internet Applications***.

D Configuring Your Web Server

You need Web server software to test your Web applications. We recommend you use Solo, which is supplied with Net Express. It needs no configuration - it is configured automatically as shown below, and if you use it you can regard this appendix as background information. If you use any other Web server software, you must configure it yourself as shown below.

You need two Web shares set up on your Web server, pointing into the project directory where your application is. One points to the project directory itself, and one to a subdirectory known as the project build directory. When Solo starts, it automatically sets up these two Web shares, pointing into the directory for the project most recently loaded in Net Express. If you use any other Web server, you must set them up yourself.

The project build directory is created automatically when you create a project. As you've already seen, when tools such as Form Designer create files they store them automatically in the appropriate directory.

The two Web shares are:

- **/COBOL/**

  The project's source directory. It has Read access permission. This is where you keep the application's HTML pages. For example, for the Goform project created in the chapter *Creating a Web Application*, it is **\Program Files\Micro Focus\Net Express\Base\Demo\Goform**

- **/cgi-bin/**

  The project's build directory. It has Read and Execute access permissions. This is where you keep the application's server-side executable files. For our Goform project, it is **c:\Program Files\Micro Focus\Net Express\Base\Demo\Goform\DEBUG**

Index

Symbols

- sign
    in tree view 165
+ sign
    in tree view 165

Numerics

64-bit file addressing 178

A

ACCEPT/DISPLAY
    extended 64
Accessing a database
    what to use 14
Acrobat 18
Add Files to Project
    dialog box 164
Adding controls 114
Adding controls to a form 71
Additional software 16
    licensing 22
Adis 180
Adobe Acrobat 18
Animating 33, 41
    basic tutorial 33
    examining data 42
    what to use 14
Animating a Web application 83
Animation 39
Animation commands
    Run Thru 41
    Run to Cursor 41
    Step 41
ANSI/EBCDIC
    converting 57
    Data File Editor 58
.app file 33
Application
    deploying 22
Application for database access
    creating 95
Application Output window
    hide not close 44
    hiding 44
Application Server
    license 21
Arranging
    columns in list 166
Associated master field 115
Associated program
    basic tutorial 125
    editing 126
Asterisk
    to expand tree 51
Asymmetric 65

B

Books 17
    finding your way around 17

Browser for the Web
    what to use 16
Btrieve interface 173
Building a project 38
    basic tutorial 33
Building an application 81
Button
    adding to a form 71
    close 163
    maximum 163
    minimize 163
    on window 163
    restore 163

C

Callable File Handler 173
Callable Rebuild 174
Callable Sort 174
Case
    in tutorial 26
Cbllink 174
CGI program 183
CGI Support 175
Changing
    folder 164
Character set
    converting 57
    Data File Editor 58
Character-mode 184
Checking portability
    in UNIX Option 133
Class Browser 176
Class Library 176
Class Wizard 177
Clicking
    in tutorial 26
Clicking a cross-reference 27
Client 183
Client/server 183
Client/Server Bindings 174
Close button 163
Closing
    a project 44
    a window 44
    can’t close Application Output window 44
    Net Express 44
    Output window 44
COBOL Syntax
    getting help 19
COBOL System Library Routines 174
COBSQL 179
Column 94
Columns in list
    rearranging 166
Command prompt 175
Comments
    in Dialog System 119
Common Gateway Interface 175
Communication across networks 174
Compiling
    what to use 14
Conditional record layout 51
Configuring a Web server 185
Configuring the IDE 42
Contacting Micro Focus
    addresses and numbers 19
Container
    rearranging 166
Contents List
    returning 27
Context help 17, 43
Context help button 163
Context menu 169
Controls 184
    adding 114
Converter
    tutorial 45
Converting
    character set 57
    data file 46
Converting existing COBOL
    basic tutorial 85
    what to use 14

Copyfiles
  displaying 80
Copying
  by cut and paste 169
  by drag and drop 170
Creating
  conditional record layout 51
  data file 56
  default record layout 50
  file or folder 165
  record layout file 49
Creating a Form
  in Form Express 86
Creating a form 68
Creating a GUI application
  basic tutorial 107
Creating a project 68, 109, 133
Creating a screenset 109
Creating a Web application
  basic tutorial 63
  introduction 67
Creating a Web database application 95
  basic tutorial 93
Cross-reference 27
Ctrl+C 169
Ctrl+V 169
Ctrl+X 169
Cut and Paste 169

D

Data
  adding in SQL Wizard 157
  examining when debugging 42
Data block 112
Data file
  converting 46
  creating 56
  editing multiple 55
  printing 54
  viewing formatted 52
  viewing in hexadecimal 54
  viewing unformatted 47
Data File Converter
  tutorial 45
Data File Editor
  character set 58
Data File editor
  tutorial 45
Data files
  editing 45
Data source 94
Data Source Name 94
Data Tools
  brief description 14
  introduction 45
  tutorial 45
Database
  filtering 102
  navigating 100
  querying 102
  updating 101
  what to use 14
Database application
  basic tutorial 93
Database sample.mdb 94
Datatype column 88
DB2 145
  debugging 148
  running application 148
Debugging 33, 41
  basic tutorial 33
  DB2 application 148
  examining data 42
  Run Thru 41
  Run to Cursor 41
  Step 41
  what to use 14
Debugging a Web application 83
Default record layout 50
Defining objects 114
Deltas 15
Demo application
  Locking 40
Demonstration applications 34

Deploying a UNIX application
 what to use 15
Deploying on UNIX
 basic tutorial 131
Deployment 22
Designing a form 71
Development Environment
 basic tutorial 33
 brief description 14
 starting 35
Dialog box 67
 Add Files to Project 164
 Open 164
Dialog Editor 180
Dialog in Dialog System
 basic tutorial 117
Dialog System
 basic tutorial 107
 brief description 15
 generated files 120
Directory
 in tutorial 26
Displaying
 EBCDIC file 58
Docking
 window 170
Documentation
 finding your way around 17
Drag and Drop 170
Drive
 in tutorial 26
Dropdown list 168
DSN 94
Dtoldemo 45

E

EBCDIC file
 displaying 58
EBCDIC/ANSI
 converting 57
 Data File Editor 58
Editing
 basic tutorial 33
 data in SQL Wizard 157
 multiple data files 55
 what to use 14
Editing a server-side program 80
Editing an associated program 126
Embedded HTML 64, 175
Examining data 42
EXEC HTML 64
Existing COBOL
 basic tutorial 85
 what to use 14
Explorer
 Internet 16
Extended ACCEPT/DISPLAY 64
Extension
 showing and hiding 165

F

File
 converting data file 46
 creating 165
 record layout 49
File extension
 showing and hiding 165
File maintenance
 what to use 14
Files generated
 by Dialog System 120
 by Form Designer 75
 by Form Express 90
 by Internet Application Wizard 98
 by UNIX Option 139
Files in project
 nonexistent files 38
Fileshare 175
Filtering a database 102
Finding information 17
Floating
 window 170

Folder
- changing 164
- creating 165
- in tutorial 26

Folder hierarchy 164
Folder Options 165
Form
- adding controls 71
- creating 68
- creating in Form Express 86
- testing 74, 75

Form Designer
- basic tutorial 63
- brief description 15
- generated files 75

Form Express
- basic tutorial 85
- generated files 90

Form processing
- basic tutorial 79

Full screen 163

G

Getting Started book
- using 27

Getting Started tutorials 25
Global dialog 118
GUI
- what to use 15

GUI application
- basic tutorial 107

H

Headings of columns
- rearranging 166

Help 17
- finding your way around 17
- for item on screen 19
- on COBOL syntax 19

Hiding
- Application Output window 44
- file extensions 165

Hierarchy of folders 164
Highlighting
- list items 167

Hotlink 27
How to find information 17
How to use this book 27
HTML 175, 183
HTML form 67
HTML Label field 89
HTML Page Wizard 68
- brief description 14

Hyper Text Markup Language 183

I

IDE
- basic tutorial 33
- brief description 14
- starting 35
- with Dialog System loaded 110
- with Form Designer loaded 70

IDE options
- setting 42

Index
- creating 154

Indexed file
- maintenance 178

Indexes
- of books and help 18

Information
- finding 17

.int file
- in projects 38

Integrated Development Environment
- basic tutorial 33
- brief description 14
- starting 35

Integrated Preprocessor Support 175

Interface for database via Web
  basic tutorial 93
Interface for GUI
  basic tutorial 107
Interface for Web
  basic tutorial 63
  introduction 67
Internationalization 176
Internet 183
Internet Application Wizard 93, 95
  basic tutorial 93
  brief description 14
  generated files 98
Internet Explorer
  brief description 16
Intranet 183
Introduction
  Internet Application Wizard 67

K

Key
  primary 154

L

Language Support 176
Legacy COBOL
  basic tutorial 85
  what to use 14
Library routine 174
License
  deployed applications 21
License agreement
  ODBC 177
Link 27
Linking
  what to use 14
Linking to a Web application 65
Links
  in books 19
List
  dropdown 168
  multicolumn 166
  selecting items 167
Loading a project 37
Loading project
  in tutorial 27
Local dialog 118
Locking files on network 175
Locking program 40
Low-level file-handling 173

M

Main window
  Form Express 88
Mainframe access
  XDB link 149
Maintaining files
  what to use 14
Master field 115
Maximize button 163
Menu
  popup and context 169
Method Wizard 177
mfuser 33
Micro Focus
  addresses and numbers 19
  via Web 19
Microsoft Access
  ODBC driver 177
Microsoft Transaction Server Support 176
Migration Cookbook
  brief description 19
Minimize button 163
Minus sign
  in tree view 165
Mouse button
  in tutorial 26

Mouse pointer
- shape 170

Moving
- by cut and paste 169
- by drag and drop 170
- columns in list 166

Moving around folders 164

Multicolumn list
- rearranging 166

Multi-threading 176

N

National Language Support 176

Navigating a database 100

Net Express
- basic tutorial 33
- closing 44
- features 13
- license 21

Net Express applications
- explained 183

Net Express command prompt 175

Network 184

Network and UNIX 16

Network server 184

Nonexistent files in project 38

O

Object toolbar 69, 114

Object-oriented COBOL syntax 176

Objects
- defining 114

ODBC 93

ODBC drivers
- brief description 177

ODBC license agreement 177

OLE Automation Support 177

Online books 17

Online Help System 180

OO CLass Browser 176

OO Class Library 176

Open
- dialog box 164

Open Database Connectivity 93, 177

OpenESQL 178

OpenESQL Assistant 178

Options
- setting in IDE 42

P

Page
- returning 27

Panels 180

Picture string 116

Plus sign
- in tree view 165

Popup help 43

Popup menu 19, 169

Portability
- in UNIX Option 133

Preprocessor 175
- integrated 175

Primary key
- defining 154

Printed books 18

Printing
- data file 54

Problems
- getting help 19

Project 33
- brief description 14
- building 38
- closing 44
- creating 68, 109, 133
- dtoldemo 45
- loading 37
- sqldemo 146, 152

Project folder 33

Projects 178
    basic tutorial 33
projects 33
Properties
    of data file 49
Prototyping 121
Publishing a UNIX application 139
PVCS 15

Q

Query
    creating in SQL Wizard 158
    running 159
Querying a database 102

R

Rearranging
    columns in list 166
Rebuild 38, 178
Rebuild All 38
Rebuilding a project
    basic tutorial 33
Record layout file
    creating 49
    look up information 53
    saving 52
Reference 27
Release Notes 19
Reloading Project
    in tutorial 27
Resizing
    columns in list 166
    window 170
Restore button 163
Returning
    to Web page 27
Revealing
    file extensions 165
Right-click 169
Row 94
Run Thru 41
Run to Cursor 41
Running a UNIX application 140
Running a Web application 82, 91
Running a Web database application 99
Running Windows GUI application 127
Running within the IDE 39

S

Samba 16
Sample program
    Locking 40
Sample programs 34
Sample.mdb database 94
Saving
    record layout file 52
SCCS 15
SCCS support 178
SCP 16
Screen
    with Dialog System loaded 110
    with Form Designer loaded 70
Screen and Keyboard Configuration Tools 180
Screen and Keyboard Handler 180
Screen Section 181
Screenset
    creating 109
Screenset processing
    basic tutorial 125
Searching for Information 17
Secondary index
    defining 156
Selecting
    list items 167
Server 183
    network 184
    Web 183
Server Control Program 16

Server-side program
- basic tutorial 79
- editing 80
- structure 64

Setting IDE options 42
Sharing files on network 175
Showing
- file extensions 165

Sizing
- columns in list 166
- window 170

Solo 64
- basic tutorial 79
- brief description 15

Solutions Guide
- brief description 19

Sort 178
Sorting 174
Source code control
- what to use 15

Source Code Control System 15
Source Code Control System support 178
SQL
- server 149
- viewing 156

SQL Option
- creating a query 158
- creating a table 153
- debugging 148
- running application 148
- starting server 147
- tutorial 145

SQL Wizard
- tutorial 151

Sqldemo project 146, 152
Standard button
- on window 163

Starting a Web application 65
Starting the Development Environment 35
Starting the IDE 35
Starting the Integrated Development Environment 35
Step 41
Submit button 64
Support
- via Web 19

Symettric 65
Syntax
- getting help 19

T

Table
- in SQL Option 153

Table of Contents
- returning 27

Task bar 163
Technical support
- via Web 19

Testing
- basic tutorial 79

Testing a form 74, 75
Testing a Web application
- what to use 15

Text
- adding to a form 71

Text properties 116
Textbox
- adding to a form 71
- definition 71

Tool tips 38
Toolbar
- in IDE 38
- object 69

Tree view 165
Tutorial
- converting existing COBOL 85
- debugging a Web application 79
- deploying on UNIX 131
- dialog in Dialog System 117
- Dialog System 107
- editing 33
- Form Designer 63
- Form Express 85
- GUI application 107
- Integrated Development Environment 33

- Internet Application Wizard 93
- migrating to the Web 85
- running a Web application 79
- server-side program 79
- Solo 79
- SQL Option 145
- SQL Wizard 151
- testing a Web application 79
- UNIX Option 131
- Web application 63
- Web database application 93
- Windows GUI program 125

Tutorials
- Getting Started 25

Tutorials map 28

U

UNIX application
- publishing 139
- running 140

UNIX deployment
- basic tutorial 131
- what to use 15

UNIX network
- what to use 16

UNIX Option
- brief description 15
- generated files 139

UNIX option
- basic tutorial 131

UNIX portability
- in UNIX Option 133

UNIX server
- what to use 16

Updating a database 101

User interface for database via Web
- basic tutorial 93

User interface for GUI
- basic tutorial 107

User interface for Web
- basic tutorial 63
- introduction 67

User project location mfuser 33

V

Version number
- in tutorial 26

Viewing
- columns in list 166
- data file formatted 52
- data file in hexadecimal 54
- data file properties 49
- data file unformatted 47
- file extensions 165
- folders 164

W

Web application
- basic tutorial 63
- creating 67
- debugging 83
- running 82, 91
- starting 65
- what to use 14

Web applications
- explained 183

Web browser 183
- what to use 16

Web database application
- basic tutorial 93
- running 99

Web form 183

Web page 183
- returning 27

Web program
- basic tutorial 79

Web server 64, 183
- basic tutorial 79
- configuring 185

Web server Solo
- brief description 15

Web shares 185
WebSync 19, 179
Win32 native GUI 179
Win32 Software Development Kit
- brief description 16

Window 67
- border 170
- closing 44
- docking 170
- floating 170
- resizing 170

Window buttons 163
Windowing Support 181
Windows 26
Windows 95 26
Windows 98 26
Windows Explorer 165
Windows GUI
- what to use 15

Windows GUI application
- running 127

Windows GUI program
- basic tutorial 125

Windows NT 26
Wizard
- SQL 153

World Wide Web 183

X

XDB link
- mainframe access 149

XDB Server
- starting 147
- stopping 149