

Introduction to Object REXX Programming

Course Summary

Description

This course is designed to enable the student to create, debug and modify programs in IBM's Object REXX programming language. The course is primarily designed for execution in a Windows environment (Windows XP/2000/NT or Windows 9x/ME), although it can also be hosted in a Linux or UNIX (AIX/Solaris) or OS/2 Warp 4.0 environment.

A principle advantage of Object REXX is the ease of creating new programs that utilize existing vendor-supplied and/or site-written classes. Rapid design and development techniques will be taught that result in clean, maintainable code and reusable class libraries.

Objectives

At the completion of this course, the student will:

- Understand the primary programming constructs and classes supplied with Object REXX and how to combine these to form effective Object REXX programs.
- Have knowledge of many advanced techniques; both those reviewed in class and additional sample programs.
- Understand the concept and practice of defensive programming, handling and recovering from errors

Topics

- Introduction to Object REXX
- Quick review of classic REXX
- Object REXX Functions
- Introduction to objects
- Using the built-in classes
- Building your own classes
- Coding graphical user interfaces using the OODialog class Library
- Using the Windows System Class Library
- TCP/IP Socket programming in REXX

Audience

This course is intended for programmers, analysts and project leaders.

Prerequisites

Students should have attended ProTech's Multiplatform REXX Programming course or equivalent knowledge of the REXX language keywords, functions and syntax.

Duration

Five days

Introduction to Object REXX Programming

Course Outline

- 1) **Introduction to Object REXX**
 - a) Built-in Functions and Methods
 - b) REXX, Object REXX, NetRexx
 - c) Object REXX
 - d) Acquiring, Installing Object REXX
 - e) Running REXX in Windows 9x etc.
 - f) Running a Win 9x/NT REXX exec:
 - g) Using the Object REXX Workbench
 - h) Understanding the Trace Bar
 - i) Introduction to Object Orientation
 - j) Object REXX Supplied Classes
 - k) Object REXX Windows Classes
 - l) Variables: Strings & Numbers
 - m) Overcoming Traditional REXX Problems
 - n) Quick Glimpse at Object REXX Pgm
 - o) New REXX Keywords
 - p) Enhanced REXX Keywords
 - q) Object REXX for Windows Books
- 2) **Quick Review of Classic REXX**
 - a) REXX Syntax
 - b) REXX Clause Types
 - c) REXX Keywords
 - d) Expressions
 - e) Functions and Subroutines
 - f) Simple and Stem variables
 - g) The REXX Stack
 - h) Command Interface
 - i) File I/O
 - j) Error handling
- 3) **Object REXX Functions**
 - a) Built-in Functions: Beep
 - b) Built-in Functions: Directory
 - c) Built-in Functions: Filespec
 - d) Built-in Functions: Var
 - e) REXX API Functions
 - f) REXX Function Packages
 - g) Understanding REXXUTIL
 - h) REXXUTIL Character UI Functions
 - i) REXXUTIL Disk & File Functions
 - j) Optional Lab: Finding a Program
 - k) Optional Lab: Tallying disk information
 - l) REXXUTIL Misc. Functions
 - m) REXXUTIL REXX Variable Functions
 - n) REXXUTIL Windows API Functions
 - o) REXXUTIL Printer Access Functions
 - p) REXXUTIL - SYSINI()
 - q) REXXUTIL - SYSINI1 Sample Output
 - r) REXXUTIL REXX Macro Functions
 - s) Understanding Unicode
 - t) Unicode Conversion Functions
 - u) Windows Encryption Functions
 - v) REXXUTIL Windows Semaphore Functions
 - w) REXXUTIL OS/2 Workplace Shell Functions
 - x) REXXUTIL Misc. OS/2 Functions
- 4) **Introduction to Objects**
 - a) Object Orientation - Why do it?
 - b) Object-Oriented Terminology
 - c) Objects
 - d) Data and Function Encapsulation
 - e) Classes and Inheritance
 - f) Directives
 - g) Introducing ~ (the "Twiddle")
 - h) Defining Classes w/ Directives
 - i) Defining Classes w/ Messages
 - j) Object REXX Class Types
 - k) More Object Oriented Buzzwords
 - l) Private vs. Public Methods & Classes
 - m) Polymorphism
- 5) **Using the Built-in Classes**
 - a) Supplied Classes
 - b) Understanding the Built-in Classes
 - c) Understanding Collection Classes
 - d) Using Objects
 - e) Understanding the Table Class
 - f) Table Class Methods
 - g) Understanding the Set Class
 - h) Set Class Methods
 - i) Using the SET class
 - j) Understanding the Built-in Classes
 - k) Understanding the Bag Class
 - l) Bag Class Methods
 - m) Lab: Using the Bag Class

Due to the nature of this material, this document refers to numerous hardware and software products by their trade names. References to other companies and their products are for informational purposes only, and all trademarks are the properties of their respective companies. It is not the intent of ProTech Professional Technical Services, Inc. to use any of these names generically

Introduction to Object REXX Programming

Course Outline (cont'd)

- 6) **Building your own Classes**
 - a) Creating Classes
 - b) Directives
 - c) How are Directives Processed?
 - d) ::REQUIRES Directive
 - e) ::CLASS Directive
 - f) ::METHOD Directive
 - g) ::ROUTINE Directive
 - h) Defining a STACK Class
 - i) Lab: Extending the Stack Class
 - j) Lab: Making Stack.cls
 - k) Two Common Methods
 - l) Example: INIT & UNINIT Methods
 - m) Special Object REXX Variables
 - n) Special and Built-in Objects
 - o) The .environment Objects
 - p) .Local Environment Objects
 - q) Sharing Data Between 2 Classes
 - r) The .methods Object
- 7) **Introduction to the OODialog Classes**
 - a) Why Should We Write Dialogs?
 - b) What we can build:
 - c) Using Standard Dialogs
 - d) Standard Dialog: TimedMessage
 - e) Standard Dialog: TimedMessage
 - f) Standard Dialog: InputBox
 - g) Standard Dialog: MultiInputBox
 - h) Standard Dialog: CheckList
 - i) Optional Lab: OODialog Functions
 - j) Building a Generic Dialog
 - k) The Buttons() Generic Function
 - l) Calling Buttons() Function
 - m) Inside Buttons()
 - n) Optional Lab: Using BUTTONS()
 - o) Building dialogs w/ Resource Workshop
 - p) Lab: Layout a List Box
 - q) Lab: Enhanced List Box Management
 - r) Optional Lab: Layout File Browser
 - s) Optional Lab: Implement File Browser
 - t) Optional Lab: File Browser w/ Menu
 - u) Optional Lab: Enhanced File Browser
- 8) **The Windows System Class Library**
 - a) Windows System Library Classes
 - b) WindowsManager Class Methods
 - c) WindowsManager Class Caveats
 - d) WindowObject Class Overview
 - e) WindowObject Information Methods
 - f) WindowObject Move/Size Methods
 - g) Optional topic: CA-Automation Point
 - h) Coexisting w/ CA-Automation Point
 - i) Managing the AP Desktop
 - j) Understanding Windows Focus
 - k) WindowObject Focus Methods
 - l) Focus Methods Example
 - m) WindowObject Send Methods
 - n) SendSysCommand Method Details
 - o) Windows Key Mnemonics
 - p) WindowObject Menu Methods
 - q) Misc WindowObject Methods
 - r) MenuObject Class Methods
 - s) MenuObject Class Example
 - t) Lab: MenuObject Class
 - u) Optional Lab: MenuObject Class
 - v) WindowsEventLog Class Overview
 - w) WindowsEventLog Methods
 - x) WindowsEventLog Class Example
 - y) WindowsClipboard Class Overview
 - z) WindowsClipBoard Methods
 - aa) WindowsClipboard Class Example
- 9) **TCP/IP Socket Programming in REXX**
 - a) TCP/IP Networking Review
 - i) TCP/IP Application Functionality
 - ii) Two Similar Packet Delivery Systems
 - iii) Packet Routing
 - iv) Network Physical Layer, IP Layer
 - v) OS/390 TCP/IP Sockets
 - vi) Well Known UDP & TCP Ports
 - vii) TCP/IP Services
 - viii) TCP/IP Applications
 - ix) Telnet Access
 - x) TCP/IP Applications: rsh, rexec
 - xi) FTP (File Transfer Protocol)
 - xii) Socket Programming Overview
 - xiii) Basic Socket Send/Receive Functions

Due to the nature of this material, this document refers to numerous hardware and software products by their trade names. References to other companies and their products are for informational purposes only, and all trademarks are the properties of their respective companies. It is not the intent of ProTech Professional Technical Services, Inc. to use any of these names generically

- xiv) Resolver, Port number
- b) Basic Socket Connection Functions
- c) Sample TCP/IP Socket Application