ooRexx Documentation 4.2

Open Object Rexx™

RxFtp Class Library Reference

W. David Ashley

Rony G. Flatscher

Mark Hessling

Rick McGuire

Lee Peedin

Oliver Sims

Jon Wolfers

ooRexx Documentation 4.2 Open Object Rexx™
RxFtp Class Library Reference
Edition 1

Author	W. David Ashley
Author	Rony G. Flatscher
Author	Mark Hessling
Author	Rick McGuire
Author	Lee Peedin
Author	Oliver Sims
Author	Jon Wolfers

Preface

This book describes the Open Object Rexx RxFtp Class Library and its methods.

This book is intended for people who plan to develop applications using Rexx and FTP. Its users range from the novice, who might have experience in some programming language but no Rexx or FTP experience, to the experienced application developer, who might have had some experience with Object Rexx and FTP.

This book is a reference rather than a tutorial. It assumes you are already familiar with object-oriented programming concepts.

Descriptions include the use and syntax of the language and explain how the language processor "interprets" the language as a program is running.

1. Document Conventions

This manual uses several conventions to highlight certain words and phrases and draw attention to specific pieces of information.

In PDF and paper editions, this manual uses typefaces drawn from the *Liberation Fonts*[1] set. The Liberation Fonts set is also used in HTML editions if the set is installed on your system. If not, alternative but equivalent typefaces are displayed. Note: Red Hat Enterprise Linux 5 and later includes the Liberation Fonts set by default.

1.1. Typographic Conventions

Four typographic conventions are used to call attention to specific words and phrases. These conventions, and the circumstances they apply to, are as follows.

`Mono-spaced Bold`

Used to highlight system input, including shell commands, file names and paths. Also used to highlight keycaps and key combinations. For example:

> To see the contents of the file **`my_next_bestselling_novel`** in your current working directory, enter the **`cat my_next_bestselling_novel`** command at the shell prompt and press **`Enter`** to execute the command.

The above includes a file name, a shell command and a keycap, all presented in mono-spaced bold and all distinguishable thanks to context.

Key combinations can be distinguished from keycaps by the hyphen connecting each part of a key combination. For example:

> Press **`Enter`** to execute the command.
>
> Press **`Ctrl`**+**`Alt`**+**`F2`** to switch to the first virtual terminal. Press **`Ctrl`**+**`Alt`**+**`F1`** to return to your X-Windows session.

The first paragraph highlights the particular keycap to press. The second highlights two key combinations (each a set of three keycaps with each set pressed simultaneously).

[1] https://fedorahosted.org/liberation-fonts/

If source code is discussed, class names, methods, functions, variable names and returned values mentioned within a paragraph will be presented as above, in **mono-spaced bold**. For example:

> File-related classes include **filesystem** for file systems, **file** for files, and **dir** for directories. Each class has its own associated set of permissions.

Proportional Bold

This denotes words or phrases encountered on a system, including application names; dialog box text; labeled buttons; check-box and radio button labels; menu titles and sub-menu titles. For example:

> Choose **System** → **Preferences** → **Mouse** from the main menu bar to launch **Mouse Preferences**. In the **Buttons** tab, click the **Left-handed mouse** check box and click **Close** to switch the primary mouse button from the left to the right (making the mouse suitable for use in the left hand).
>
> To insert a special character into a **gedit** file, choose **Applications** → **Accessories** → **Character Map** from the main menu bar. Next, choose **Search** → **Find…** from the **Character Map** menu bar, type the name of the character in the **Search** field and click **Next**. The character you sought will be highlighted in the **Character Table**. Double-click this highlighted character to place it in the **Text to copy** field and then click the **Copy** button. Now switch back to your document and choose **Edit** → **Paste** from the **gedit** menu bar.

The above text includes application names; system-wide menu names and items; application-specific menu names; and buttons and text found within a GUI interface, all presented in proportional bold and all distinguishable by context.

Mono-spaced Bold Italic or ***Proportional Bold Italic***

Whether mono-spaced bold or proportional bold, the addition of italics indicates replaceable or variable text. Italics denotes text you do not input literally or displayed text that changes depending on circumstance. For example:

> To connect to a remote machine using ssh, type **ssh *username*@*domain.name*** at a shell prompt. If the remote machine is **example.com** and your username on that machine is john, type **ssh john@example.com**.
>
> The **mount -o remount *file-system*** command remounts the named file system. For example, to remount the **/home** file system, the command is **mount -o remount /home**.
>
> To see the version of a currently installed package, use the **rpm -q *package*** command. It will return a result as follows: ***package-version-release***.

Note the words in bold italics above — username, domain.name, file-system, package, version and release. Each word is a placeholder, either for text you enter when issuing a command or for text displayed by the system.

Aside from standard usage for presenting the title of a work, italics denotes the first use of a new and important term. For example:

> Publican is a *DocBook* publishing system.

1.2. Pull-quote Conventions

Terminal output and source code listings are set off visually from the surrounding text.

Output sent to a terminal is set in **mono-spaced roman** and presented thus:

```
books        Desktop   documentation  drafts  mss    photos   stuff  svn
books_tests  Desktop1  downloads      images  notes  scripts  svgs
```

Source-code listings are also set in **mono-spaced roman** but add syntax highlighting as follows:

```
package org.jboss.book.jca.ex1;

import javax.naming.InitialContext;

public class ExClient
{
   public static void main(String args[])
       throws Exception
   {
      InitialContext iniCtx = new InitialContext();
      Object         ref    = iniCtx.lookup("EchoBean");
      EchoHome       home   = (EchoHome) ref;
      Echo           echo   = home.create();

      System.out.println("Created Echo");

      System.out.println("Echo.echo('Hello') = " + echo.echo("Hello"));
   }
}
```

1.3. Notes and Warnings

Finally, we use three visual styles to draw attention to information that might otherwise be overlooked.

Note

Notes are tips, shortcuts or alternative approaches to the task at hand. Ignoring a note should have no negative consequences, but you might miss out on a trick that makes your life easier.

Important

Important boxes detail things that are easily missed: configuration changes that only apply to the current session, or services that need restarting before an update will apply. Ignoring a box labeled 'Important' will not cause data loss but may cause irritation and frustration.

Warning

Warnings should not be ignored. Ignoring warnings will most likely cause data loss.

2. How to Read the Syntax Diagrams

Throughout this book, syntax is described using the structure defined below.

- Read the syntax diagrams from left to right, from top to bottom, following the path of the line.

  The **>>---** symbol indicates the beginning of a statement.

  The **--->** symbol indicates that the statement syntax is continued on the next line.

  The **>---** symbol indicates that a statement is continued from the previous line.

  The **---><** symbol indicates the end of a statement.

  Diagrams of syntactical units other than complete statements start with the **>---** symbol and end with the **--->** symbol.

- Required items appear on the horizontal line (the main path).

```
>>-STATEMENT--required_item-----------------------------------><
```

- Optional items appear below the main path.

```
>>-STATEMENT--+---------------+--------------------------------><
              +-optional_item-+
```

- If you can choose from two or more items, they appear vertically, in a stack. If you must choose one of the items, one item of the stack appears on the main path.

```
>>-STATEMENT--+-required_choice1-+----------------------------><
              +-required_choice2-+
```

- If choosing one of the items is optional, the entire stack appears below the main path.

```
>>-STATEMENT--+-----------------+----------------------------><
              +-optional_choice1-+
              +-optional_choice2-+
```

- If one of the items is the default, it appears above the main path and the remaining choices are shown below.

```
              +-default_choice--+
>>-STATEMENT--+-----------------+----------------------------><
              +-optional_choice-+
```

```
              +-optional_choice-+
```

- An arrow returning to the left above the main line indicates an item that can be repeated.

```
              +-----------------+
              V                 |
>>-STATEMENT----repeatable_item-+------------------------------><
```

  A repeat arrow above a stack indicates that you can repeat the items in the stack.

- A set of vertical bars around an item indicates that the item is a fragment, a part of the syntax diagram that appears in greater detail below the main diagram.

```
>>-STATEMENT--| fragment |-------------------------------------><
```

  fragment:

```
|--expansion_provides_greater_detail---------------------------|
```

- Keywords appear in uppercase (for example, **PARM1**). They must be spelled exactly as shown but you can type them in upper, lower, or mixed case. Variables appear in all lowercase letters (for example, **parmx**). They represent user-supplied names or values.
- If punctuation marks, parentheses, arithmetic operators, or such symbols are shown, you must enter them as part of the syntax.

The following example shows how the syntax is described:

```
        +-,------+
        V        |
>>-MAX(----number-+--)----------------------------------------><
```

3. Getting Help and Submitting Feedback

The Open Object Rexx Project has a number of methods to obtain help and submit feedback for ooRexx and the extension packages that are part of ooRexx. These methods, in no particular order of preference, are listed below.

3.1. The Open Object Rexx SourceForge Site

The *Open Object Rexx Project*[2] utilizes *SourceForge*[3] to house the *ooRexx Project*[4] source repositories, mailing lists and other project features. Over time it has become apparent that the Developer and User mailing lists are better tools for carrying on discussions concerning ooRexx and that the Forums provided by SourceForge are cumbersome to use. The ooRexx user is most likely to get timely replies from one of the mailing lists.

Here is a list of some of the most useful facilities provided by SourceForge.

The Developer Mailing List
: You can subscribe to the oorexx-devel mailing list at *ooRexx Mailing List Subscriptions*[5] page. This list is for discussing ooRexx project development activities and future interpreter enhancements. It also supports a historical archive of past messages.

The Users Mailing List
: You can subscribe to the oorexx-users mailing list at *ooRexx Mailing List Subscriptions*[6] page. This list is for discussing using ooRexx. It also supports a historical archive of past messages.

The Announcements Mailing List
: You can subscribe to the oorexx-announce mailing list at *ooRexx Mailing List Subscriptions*[7] page. This list is only used to announce significant ooRexx project events.

The Bug Mailing List
: You can subscribe to the oorexx-bugs mailing list at *ooRexx Mailing List Subscriptions*[8] page. This list is only used for monitoring changes to the ooRexx bug tracking system.

Bug Reports
: You can create a bug report at *ooRexx Bug Report*[9] page. Please try to provide as much information in the bug report as possible so that the developers can determine the problem as quickly as possible. Sample programs that can reproduce your problem will make it easier to debug reported problems.

Documentation Feedback
: You can submit feedback for, or report errors in, the documentation at *ooRexx Documentation Report*[10] page. Please try to provide as much information in a documentation report as possible. In addition to listing the document and section the report concerns, direct quotes of the text will help the developers locate the text in the source code for the document. (Section numbers are generated when the document is produced and are not available in the source code itself.) Suggestions as to how to reword or fix the existing text should also be included.

Request For Enhancement
: You can suggest ooRexx features at the *ooRexx Feature Requests*[11] page.

[2] http://www.oorexx.org/
[3] http://sourceforge.net/
[4] http://sourceforge.net/projects/oorexx
[5] http://sourceforge.net/mail/?group_id=119701
[6] http://sourceforge.net/mail/?group_id=119701
[7] http://sourceforge.net/mail/?group_id=119701
[8] http://sourceforge.net/mail/?group_id=119701
[9] http://sourceforge.net/tracker/?group_id=119701&atid=684730
[10] http://sourceforge.net/tracker/?group_id=119701&atid=1001880
[11] http://sourceforge.net/tracker/?group_id=119701&atid=684733

Patch Reports

If you create an enhancement patch for ooRexx please post the patch using the *ooRexx Patch Report*[12] page. Please provide as much information in the patch report as possible so that the developers can evaluate the enhancement as quickly as possible.

Please do not post bug fix patches here, instead you should open a bug report and attach the patch to it.

The ooRexx Forums

The ooRexx project maintains a set of forums that anyone may contribute to or monitor. They are located on the *ooRexx Forums*[13] page. There are currently three forums available: Help, Developers and Open Discussion. In addition, you can monitor the forums via email.

3.2. The Rexx Language Association Mailing List

The *Rexx Language Association*[14] maintains a mailing list for its members. This mailing list is only available to RexxLA members thus you will need to join RexxLA in order to get on the list. The dues for RexxLA membership are small and are charged on a yearly basis. For details on joining RexxLA please refer to the *RexxLA Home Page*[15] or the *RexxLA Membership Application*[16] page.

3.3. comp.lang.rexx Newsgroup

The *comp.lang.rexx*[17] newsgroup is a good place to obtain help from many individuals within the Rexx community. You can obtain help on Open Object Rexx or on any number of other Rexx interpreters and tools.

4. Related Information

See also: *Open Object Rexx: Reference*

[12] http://sourceforge.net/tracker/?group_id=119701&atid=684732
[13] http://sourceforge.net/forum/?group_id=119701
[14] http://www.rexxla.org/
[15] http://rexxla.org/
[16] http://www.rexxla.org/rexxla/join.html
[17] http://groups.google.com/group/comp.lang.rexx/topics?hl=en

What is RxFtp?

RxFtp is an Object Rexx Class library providing access to the TCP/IP FTP interface based on RFC 959.

It is assumed that you are familiar with the basic FTP functionality and can reference those specific to the system. For more information, refer to the book *Internetworking with TCP/IP, Volume I: Principles, Protocols and Architecture* by Douglas Comer (Prentice Hall PTR).

The RxFtp package requires the RxSock external function package supplied with Open Object Rexx.

Installation

The RxFtp package is contained in the file *rxftp.cls*. This file must be placed in a directory listed in your PATH. To get access to the class and methods in the RxFtp package you must include a ::REQUIRES directive in any Object Rexx script that uses the class:

```
     .
     .
     .
::requires "rxftp.cls"
```

You instantiate an instance of the class using the normal Object Rexx conventions.

```
myftpobj = .rxftp~new
```

Each instance of the class completely encapsulates a connection to an FTP server. This means that you can instantiate multiples instance of the class and have multiple open connections to FTP servers simultaneously.

2.1. Notes on Using RxFtp

1. The FtpProxy method does not exist in the RxFtp class.
2. Firewalls can prevent many of the RxFtp methods from working properly. Active and passive mode transfers can both be restricted by the use of firewalls. This especially applies to active mode transfers when the client is running a firewall locally (like WindowsXP SP2 or any modern Linux distribution).
3. Each instance of the RxFtp class you instantiate has its own connection to an FTP server. Thus you can instantiate multiple instance of this class and connect to multiple FTP servers simultaneously. Also, each instance can be reused - the FtpLogoff method reinitializes the instance and a following FtpSetUser method can establish a new connection to an FTP server.

RxFtp Method Reference

The following sections describe how the individual methods available in RxFtp are invoked from the Rexx programming environment:

3.1. FtpAppend

The FtpAppend method appends a file to another file on the FTP server. If the target file on the server does not exist it is created.

Syntax:

```
>>--myftpobj~FtpAppend(--localfilename, remotefilename--+----------+--)--><
                                                       +--, mode--+
```

Arguments:

localfilename
: The filename and optional path where the source file is stored.

remotefilename
: The filename and optional path of the target file for the append operation on the FTP server. If the path is not given the current directory on the ftp server will be used. If the file does not exist it will be created.

mode
: The file transfer mode, either 'ASCII' or 'BINARY'. If the argument is not given the current mode will be used.

Return Values:

A value of 0 indicates successful execution of the method. The value -1 indicates an error. You can get the specific error code by inspection of the method FtpErrno. Possible values:

FTPCOMMAND
: The internal FTP command or the arguments to the method are in error.

3.2. FtpChDir

The FtpChDir method changes the current directory on the FTP server. If this command fails the directory on the server remains unchanged.

Syntax:

```
>>-- myftpobj~FtpChDir(newdir)-----------------------------------------><
```

Arguments:

newdir
: The directory name and optional path to the directory to be made the current directory.

Return Values:

A value of 0 indicates successful execution of the method. The value -1 indicates an error. You can get the specific error code by inspection of the method FtpErrno. Possible values:

FTPCOMMAND
: The internal FTP command or the arguments to the method are in error.

3.3. FtpDelete

The FtpDelete method deletes the specified file on the FTP server.

Syntax:

```
>>--myftpobj~FtpDelete(filename)-----------------------------------------><
```

Arguments:

filename
: The filename and optional path to the file to be deleted. If the path is not given the file will be deleted from the current directory on the ftp server.

Return Values:

A value of 0 indicates successful execution of the method. The value -1 indicates an error. You can get the specific error code by inspection of the method FtpErrno. Possible values:

FTPCOMMAND
: The internal FTP command or the arguments to the method are in error.

3.4. FtpDir

The FtpDir method returns a listing (in long format) of a directory on the server. The directory listing is placed in the attribute *Response* as an Object Rexx array of lines. The array will be empty if this method encounters an error.

The format of the returned directory listing depends on the operating system running on the FTP server machine.

Syntax:

```
>>-- myftpobj~FtpDir(--+-----------+--)---------------------------------><
                       +--pattern--+
```

Arguments:

pattern
: The filename pattern to use to filter the directory listing. You can use the standard '*' and '?' wildcard characters in the pattern. This argument is optional. If this argument is not specified the default pattern './*' will be used.

Return Values:

A value of 0 indicates successful execution of the method. The value -1 indicates an error. You can get the specific error code by inspection of the method FtpErrno. Possible values:

FTPCOMMAND
: The internal FTP command or the arguments to the method are in error.

3.5. FtpGet

The FtpGet method gets a single file from the FTP server.

Syntax:

```
>>-- myftpobj~FtpGet(localfilename, remotefilename--+----------+--)------><
                                                   +--, mode--+
```

Arguments:

localfilename
: The filename and optional path where the retrieved file will be stored. If the path is not given the file will be placed in the user's current directory.

remotefilename
: The filename and optional path to the file to be retrieved. If the path is not given the file will be retrieved from the current directory on the ftp server.

mode
: The file transfer mode, either 'ASCII' or 'BINARY'. If the argument is not given the current mode will be used.

Return Values:

A value of 0 indicates successful execution of the method. The value -1 indicates an error. You can get the specific error code by inspection of the method FtpErrno. Possible values:

FTPCOMMAND
: The internal FTP command or the arguments to the method are in error.

3.6. FtpGetMode

The FtpSetType method returns the file transfer mode on the FTP server.

Syntax:

```
>>-- myftpobj~FtpGetMode()-----------------------------------------------><
```

Arguments:

None.

Return Values:

Returns one of the following strings.

'ACTIVE'
: The current transfer mode is active.

'PASSIVE'
The current transfer mode is passive.

3.7. FtpGetType

The FtpSetType method returns the file transfer type on the FTP server.

Syntax:

```
>>-- myftpobj~FtpGetType()-----------------------------------------------><
```

Arguments:

None.

Return Values:

Returns one of the following strings.

'ASCII'
The current type is ASCII.

'BINARY'
The current type is binary.

3.8. FtpLogoff

The FtpLogoff method logs off the FTP server and closes the connection. It also reinitializes the instance of the class so it can be used again.

Syntax:

```
>>-- myftpobj~FtpLogoff()-----------------------------------------------><
```

Arguments:

None.

Return Values:

A value of 0 indicates successful execution of the method. The value -1 indicates an error. You can get the specific error code by inspection of the method FtpErrno. Possible values:

FTPCOMMAND
The internal FTP command or the arguments to the method are in error.

3.9. FtpLs

The FtpLs method returns a listing (in short format) of a directory on the server. The directory listing is placed in the attribute *Response* as an Object Rexx array of lines. The array will be empty if this method encounters an error.

Each member of the array containing the directory listing usually only contains the name of a file in the specified directory. But the format may be different in some cases.

Syntax:

```
>>-- myftpobj~FtpLs(--+-----------+--)-----------------------------------><
                     +--pattern--+
```

Arguments:

pattern
: The filename pattern to use to filter the directory listing. You can use the standard '*' and '?' wildcard characters in the pattern. This argument is optional. If this argument is not specified the default pattern './*' will be used.

Return Values:

A value of 0 indicates successful execution of the method. The value -1 indicates an error. You can get the specific error code by inspection of the method FtpErrno. Possible values:

FTPCOMMAND
: The internal FTP command or the arguments to the method are in error.

3.10. FtpMkDir

The FtpMkDir method creates a new subdirectory on the FTP server.

Syntax:

```
>>-- myftpobj~FtpMkDir(newdir)-------------------------------------------><
```

Arguments:

newdir
: The name and optional path to the new directory.

Return Values:

A value of 0 indicates successful execution of the method. The value -1 indicates an error. You can get the specific error code by inspection of the method FtpErrno. Possible values:

FTPCOMMAND
: The internal FTP command or the arguments to the method are in error.

3.11. FtpPut

The FtpPut method sends a single file to the FTP server.

Syntax:

```
>>-- myftpobj~FtpPut(localfilename, remotefilename--+----------+--)------><
                                                   +--, mode--+
```

Arguments:

localfilename
: The filename and optional path where the source file is stored.

remotefilename
: The filename and optional path used to store the file on the FTP server. If the path is not given the file will be stored in the current directory on the ftp server.

mode
: The file transfer mode, either 'ASCII' or 'BINARY'. If the argument is not given the current mode will be used.

Return Values:

A value of 0 indicates successful execution of the method. The value -1 indicates an error. You can get the specific error code by inspection of the method FtpErrno. Possible values:

FTPCOMMAND
: The internal FTP command or the arguments to the method are in error.

3.12. FtpPutUnique

The FtpPut method sends a single file to the FTP server. If the file exists on the server it is not replaced and a new name is assigned to the file sent from the client.

Syntax:

```
>>-- myftpobj~FtpPutUnique(localfilename, remotefilename--+----------+--)--><
                                                         +--, mode--+
```

Arguments:

localfilename
: The filename and optional path where the source file is stored.

remotefilename
: The filename and optional path used to store the file on the FTP server. If the filename exists then a new filename is used. If the path is not given the file will be stored in the current directory on the ftp server.

mode
: The file transfer mode, either 'ASCII' or 'BINARY'. If the argument is not given the current mode will be used.

Return Values:

A value of 0 indicates successful execution of the method. The value -1 indicates an error. You can get the specific error code by inspection of the method FtpErrno. Possible values:

FTPCOMMAND
: The internal FTP command or the arguments to the method are in error.

3.13. FtpPwd

The FtpPwd method returns the current directory name and path on the FTP server. The information is placed in the attribute Response as an Object Rexx array of lines. However, this method always returns a single line to the array if it succeeds. The array will be empty if this method encounters an error.

Syntax:

```
>>-- myftpobj~FtpPwd()-------------------------------------------------><
```

Arguments:

None.

Return Values:

A value of 0 indicates successful execution of the method. The value -1 indicates an error. You can get the specific error code by inspection of the method FtpErrno. Possible values:

FTPCOMMAND
: The internal FTP command or the arguments to the method are in error.

3.14. FtpQuote

The FtpQuote method sends a command to the FTP server.

Syntax:

```
>>-- myftpobj~FtpQuote(cmd)-----------------------------------------><
```

Arguments:

cmd
: The command to be executed on the server.

Return Values:

A value of 0 indicates successful execution of the method. The value -1 indicates an error. You can get the specific error code by inspection of the method FtpErrno. Possible values:

FTPCOMMAND
The internal FTP command or the arguments to the method are in error.

3.15. FtpRename

The FtpRename method renames an existing file on the FTP server. On Unix FTP server this command can also move a file from one subdirectory to another with either a new or its old name.

Syntax:

```
>>-- myftpobj~FtpRename(oldfile, newfile)------------------------------><
```

Arguments:

oldfile
The name and optional path of the file to be renamed.

newfile
The new name and optional path of the file.

Return Values:

A value of 0 indicates successful execution of the method. The value -1 indicates an error. You can get the specific error code by inspection of the method FtpErrno. Possible values:

FTPCOMMAND
The internal FTP command or the arguments to the method are in error.

3.16. FtpRmDir

The FtpRmDir method removes a subdirectory on the FTP server.

Syntax:

```
>>-- myftpobj~FtpRmDir(newdir)-----------------------------------------><
```

Arguments:

newdir
The name and optional path to the directory to be removed.

Return Values:

A value of 0 indicates successful execution of the method. The value -1 indicates an error. You can get the specific error code by inspection of the method FtpErrno. Possible values:

FTPCOMMAND
The internal FTP command or the arguments to the method are in error.

3.17. FtpSetMode

The FtpSetMode method changes the passive/active mode used for file tranfers to/from the FTP server.

Syntax:

```
>>-- myftpobj~FtpSetMode(--+--"PASSIVE"--+--)---------------------------><
                           +--"ACTIVE"---+
```

Arguments:

'PASSIVE'
: Sets passive mode for file transfers. Only the first letter of the argument is significant and it is not case-sensitive.

'ACTIVE'
: Sets active mode for file transfers. Only the first letter of the argument is significant and it is not case-sensitive.

Return Values:

A value of 0 indicates successful execution of the method. The value -1 indicates an error. You can get the specific error code by inspection of the method FtpErrno. Possible values:

FTPCOMMAND
: The internal FTP command or the arguments to the method are in error.

If an error is returned the mode remains unchanged.

3.18. FtpSetType

The FtpSetType method changes the file transfer type on the FTP server.

Syntax:

```
>>-- myftpobj~FtpSetType(--+--"ASCII"---+--)----------------------------><
                           +--"BINARY"--+
```

Arguments:

'ASCII'
: Changes the file transfer type on the FTP server to ASCII. Only the first letter of the argument is significant and it is not case-sensitive.

'BINARY'
: Changes the file transfer mode on the FTP server to binary. Only the first letter of the argument is significant and it is not case-sensitive.

Return Values:

A value of 0 indicates successful execution of the method. The value -1 indicates an error. You can get the specific error code by inspection of the method FtpErrno. Possible values:

FTPCOMMAND
: The internal FTP command or the arguments to the method are in error.

If an error is returned the type remains unchanged.

3.19. FtpSetUser

The FtpSetUser method creates a session with an FTP server and logs the user on.

Syntax:

```
>>-- myftpobj~FtpSetUser(host, user--+--------------------------+--)------><
                                    +--, password--+----------+-+
                                                   +--, acct--+
```

Arguments:

host
: The host name or TCP/IP address of the FTP server.

user
: The user name for the server.

password
: (optional) The user password.

acct
: (optional) The accounting information for the user.

Return Values:

A value of 0 indicates successful execution of the method. The value -1 indicates an error. You can get the specific error code by inspection of the method FtpErrno. Possible values:

FTPCOMMAND
: The internal FTP command or the arguments to the method are in error.

FTPHOST
: Unknown *host* name or server not responding.

FTPSOCKET
: RxSock function error.

FTPCONNECT
: Server not responding or not available.

FTPLOGIN
: *user*, *password* or *acct* information is invalid.

3.20. FtpSite

The FtpSite method sends a Site command to the FTP server.

Syntax:

```
>>--myftpobj~FtpSite(cmd)-----------------------------------------------><
```

Arguments:

cmd
: The Site command to be executed on the server.

Return Values:

A value of 0 indicates successful execution of the method. The value -1 indicates an error. You can get the specific error code by inspection of the method FtpErrno. Possible values:

FTPCOMMAND
: The internal FTP command or the arguments to the method are in error.

3.21. FtpSys

The FtpSys method returns the operating system description information from FTP server. The information is placed in the attribute Response as an Object Rexx array of lines. However, this method always returns a single line to the array if it succeeds. The array will be empty if this method encounters an error.

Syntax:

```
>>-- myftpobj~FtpSys()-------------------------------------------------><
```

Arguments:

None.

Return Values:

A value of 0 indicates successful execution of the method. The value -1 indicates an error. You can get the specific error code by inspection of the method FtpErrno. Possible values:

FTPCOMMAND
: The internal FTP command or the arguments to the method are in error.

3.22. FtpTrace

The FtpTrace method starts the tracing of FTP commands. The initial state is off. If tracing is on the FTP commands sent to the server and their responses will be displayed.

Syntax:

```
>>-- myftpobj~FtpTrace()-------------------------------------------------><
```

Arguments:

None.

Return Values:

A value of 0 indicates successful execution of the method. The value -1 indicates an error. You can get the specific error code by inspection of the method FtpErrno. Possible values:

FTPCOMMAND
: The internal FTP command or the arguments to the method are in error.

3.23. FtpTraceLog

The FtpTraceLog method causes all FTP commands and responses in the current trace buffer to be written to a log file.

Syntax:

```
>>--myftpobj~FtpTraceLog(filename--+---------------+--)------------------><
                                  +--, "REPLACE"--+
```

Arguments:

filename
: The filename and optional path to the log file.

'REPLACE'
: If this argument is given then if the log file exists it will be replaced. The default is to append the trace output to the log file. Only the first letter of the argument is significant.

Return Values:

A value of 0 indicates successful execution of the method. The value -1 indicates an error. You can get the specific error code by inspection of the method FtpErrno. Possible values:

FTPCOMMAND
: The internal FTP command or the arguments to the method are in error.

3.24. FtpTraceLogoff

The FtpTraceLogoff method stops the tracing of FTP commands.

Syntax:

```
>>-- myftpobj~FtpTraceLogoff()-----------------------------------------><
```

Arguments:

None.

Return Values:

A value of 0 indicates successful execution of the method. The value -1 indicates an error. You can get the specific error code by inspection of the method FtpErrno. Possible values:

FTPCOMMAND
: The internal FTP command or the arguments to the method are in error.

3.25. FtpVersion

The FtpVersion method returns the version number of the RxFtp class in the form of *x.x.x*.

Syntax:

```
>>--myftpobj~FtpVersion()------------------------------------------><
```

Arguments:

None.

RxFtp Additional Method Attributes

The following sections describe the additional method attributes available in the RxFtp class.

- BufSize
- CmdResponse
- CR_Remove
- Debug
- FtpErrno
- Response

4.1. BufSize

The BufSize method attribute contains the size of the buffer used by RxFtp methods. The default is 4096 and you can change this to any positive integer value.

Increasing this value may speed your FTP transactions, but making it too large may slow them down. Experimentation may be called for.

Example:

The following will display the BufSize and reset it to 8192.

```
say myftpobj~bufSize
myftpobj~bufSize = 8192
```

4.2. CmdResponse

The CmdResponse method attribute is an Object Rexx array which contains the commands sent to the FTP server and the responses received from the server. Items are continually added to this array for each method invoked during the session with the server.

This array is a complete record of the command/response activity between the client and the server.

Example:

The following will display the CmdResponse array.

```
do i = 1 to myftpobj~cmdresponse~items
   say myftpobj~cmdresponse[i]
end
```

4.3. CR_Remove

The CR_Remove method attribute is always in one of two states. If it is set to .true then carriage return characters ('0D'x) are removed from ASCII downloads when the class is running on a Unix platform. If it is set to .false then CR bytes are retained in the locally stored file.

This attribute is ignored for non-Unix machines.

The default value is .true.

Example:

The following will disable the removal of CR bytes.

```
myftpobj~cr_remove = .false
```

4.4. Debug

The Debug method attribute is always in one of two states. If it is set to .true then debug messages will be sent to STDOUT. If it is set to .false then all debugging messages will be suppressed.

The default value is .false.

Example:

The following will enable the debugging messages.

```
myftpobj~debug = .true
```

4.5. FtpErrno

The FtpErrno method attribute contains extended information when a method return an error. It is a simple string value or a zero-length string when the method is successful.

These are the possible values for FtpErrno.

FTPCOMMAND
: The internal FTP command or the arguments to the method are in error.

FTPHOST
: Unknown *host* name or server not responding.

FTPSOCKET
: RxSock function error.

FTPCONNECT
: Server not responding or not available.

FTPLOGIN
: Invalid login information was supplied.

Example:

The following will display the FtpErrno.

```
if myftpobj~ftperrno <> "" then say myftpobj~ftperrno
```

4.6. Response

The Response method attribute is always in one of two states. If it is set to .nil then the command did not generate a response. In all other cases the Response is an Object Rexx array object with each array entry containing a single line of the response.

A typical value for Response would be the lines from the server directory listing in response to invoking the FtpDir method.

Example:

The following will display the Response array.

```
if myftpobj~response <> .nil then do i = 1 to myftpobj~response~items
   say myftpobj~response[i]
   end
```

Appendix A. Sample RxFtp Program

The following is a sample program using REXX FTP Class Library methods.

```
/*==============================================================*/
/* Basic RxFTP sample to send a file with cmd/reply logging     */
/*==============================================================*/
/* Define the variables for: */
server = "127.0.0.1" /* IP address or server name */
userid = "remote_user_ID"
passwd = "password_of_remote_user"
trclog = "logfile.txt" /* Trace log file name */
retc = 0 /* Set return code to 0 */

myftp = .rxftp~new()

/* Start tracing FTP commands and logging of replies */
retc = myftp~FtpTrace()

/* Trace log file will be replaced, not appended to. */
retc = myftp~FtpTraceLog(trclog, "R")
If retc = 0 then Say " Replies will be written to log file: "trclog"."
Else Say " No writing to log file: "trclog" possible."

/* Define remote host and user to be used during the session */
retc = myftp~FtpSetUser(server, userid, passwd)
If retc = 0 then Say " Connection established."
Else Call Terminate " *** Connection failed."

/* Transfer an ASCII file to the remote ftp server */
retc = myftp~FtpPut("sample.rex", "sample.put", "ASCII")
If retc = 0 then Call Terminate " File has been sent. "
Else Call Terminate " *** File has NOT been sent."

/* Terminate the file transfer */
Terminate:
Parse Arg Message
Say Message
Say "FtpErrNo =" myftp~ftperrno

retc = myftp~FtpLogoff()

retc = myftp~FtpTraceLogoff()
retc = myftp~FtpTrace()

exit retc

::requires "rxftp.cls"
```

Appendix B. Notices

Any reference to a non-open source product, program, or service is not intended to state or imply that only non-open source product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any Rexx Language Association (RexxLA) intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-open source product, program, or service.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-open source products was obtained from the suppliers of those products, their published announcements or other publicly available sources. RexxLA has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-RexxLA packages. Questions on the capabilities of non-RexxLA packages should be addressed to the suppliers of those products.

All statements regarding RexxLA's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

B.1. Trademarks

Open Object Rexx™ and ooRexx™ are trademarks of the Rexx Language Association.

The following terms are trademarks of the IBM Corporation in the United States, other countries, or both:

1-2-3
AIX
IBM
Lotus
OS/2
S/390
VisualAge

AMD is a trademark of Advance Micro Devices, Inc.

Intel, Intel Inside (logos), MMX and Pentium are trademarks of Intel Corporation in the United States, other countries, or both.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the Unites States, other countries, or both.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

B.2. Source Code For This Document

The source code for this document is available under the terms of the Common Public License v1.0 which accompanies this distribution and is available in the appendix *Appendix C, Common Public License Version 1.0*. The source code itself is available at **`http://sourceforge.net/project/showfiles.php?group_id=119701`**.

The source code for this document is maintained in DocBook SGML/XML format.

Appendix C. Common Public License Version 1.0

THE ACCOMPANYING PROGRAM IS PROVIDED UNDER THE TERMS OF THIS COMMON PUBLIC LICENSE ("AGREEMENT"). ANY USE, REPRODUCTION OR DISTRIBUTION OF THE PROGRAM CONSTITUTES RECIPIENT'S ACCEPTANCE OF THIS AGREEMENT.

C.1. Definitions

"Contribution" means:

1. in the case of the initial Contributor, the initial code and documentation distributed under this Agreement, and

2. in the case of each subsequent Contributor:
   a. changes to the Program, and

   b. additions to the Program;

where such changes and/or additions to the Program originate from and are distributed by that particular Contributor. A Contribution 'originates' from a Contributor if it was added to the Program by such Contributor itself or anyone acting on such Contributor's behalf. Contributions do not include additions to the Program which: (i) are separate modules of software distributed in conjunction with the Program under their own license agreement, and (ii) are not derivative works of the Program.

"Contributor" means any person or entity that distributes the Program.

"Licensed Patents " mean patent claims licensable by a Contributor which are necessarily infringed by the use or sale of its Contribution alone or when combined with the Program.

"Program" means the Contributions distributed in accordance with this Agreement.

"Recipient" means anyone who receives the Program under this Agreement, including all Contributors.

C.2. Grant of Rights

1. Subject to the terms of this Agreement, each Contributor hereby grants Recipient a non-exclusive, worldwide, royalty-free copyright license to reproduce, prepare derivative works of, publicly display, publicly perform, distribute and sublicense the Contribution of such Contributor, if any, and such derivative works, in source code and object code form.

2. Subject to the terms of this Agreement, each Contributor hereby grants Recipient a non-exclusive, worldwide, royalty-free patent license under Licensed Patents to make, use, sell, offer to sell, import and otherwise transfer the Contribution of such Contributor, if any, in source code and object code form. This patent license shall apply to the combination of the Contribution and the Program if, at the time the Contribution is added by the Contributor, such addition of the Contribution causes such combination to be covered by the Licensed Patents. The patent license shall not apply to any other combinations which include the Contribution. No hardware per se is licensed hereunder.

3. Recipient understands that although each Contributor grants the licenses to its Contributions set forth herein, no assurances are provided by any Contributor that the Program does not infringe the patent or other intellectual property rights of any other entity. Each Contributor disclaims any liability to Recipient for claims brought by any other entity based on infringement

of intellectual property rights or otherwise. As a condition to exercising the rights and licenses granted hereunder, each Recipient hereby assumes sole responsibility to secure any other intellectual property rights needed, if any. For example, if a third party patent license is required to allow Recipient to distribute the Program, it is Recipient's responsibility to acquire that license before distributing the Program.

4. Each Contributor represents that to its knowledge it has sufficient copyright rights in its Contribution, if any, to grant the copyright license set forth in this Agreement.

C.3. Requirements

A Contributor may choose to distribute the Program in object code form under its own license agreement, provided that:

1. it complies with the terms and conditions of this Agreement; and

2. its license agreement:

    a. effectively disclaims on behalf of all Contributors all warranties and conditions, express and implied, including warranties or conditions of title and non-infringement, and implied warranties or conditions of merchantability and fitness for a particular purpose;

    b. effectively excludes on behalf of all Contributors all liability for damages, including direct, indirect, special, incidental and consequential damages, such as lost profits;

    c. states that any provisions which differ from this Agreement are offered by that Contributor alone and not by any other party; and

    d. states that source code for the Program is available from such Contributor, and informs licensees how to obtain it in a reasonable manner on or through a medium customarily used for software exchange.

When the Program is made available in source code form:

1. it must be made available under this Agreement; and

2. a copy of this Agreement must be included with each copy of the Program.

Contributors may not remove or alter any copyright notices contained within the Program.

Each Contributor must identify itself as the originator of its Contribution, if any, in a manner that reasonably allows subsequent Recipients to identify the originator of the Contribution.

C.4. Commercial Distribution

Commercial distributors of software may accept certain responsibilities with respect to end users, business partners and the like. While this license is intended to facilitate the commercial use of the Program, the Contributor who includes the Program in a commercial product offering should do so in a manner which does not create potential liability for other Contributors. Therefore, if a Contributor includes the Program in a commercial product offering, such Contributor ("Commercial Contributor") hereby agrees to defend and indemnify every other Contributor ("Indemnified Contributor") against any losses, damages and costs (collectively "Losses") arising from claims, lawsuits and other legal actions brought by a third party against the Indemnified Contributor to the extent caused by the acts or omissions of such Commercial Contributor in connection with its distribution of the Program in a commercial product offering. The obligations in this section do not apply to any claims or Losses relating to any actual or alleged intellectual property infringement. In order to qualify, an Indemnified

Contributor must: a) promptly notify the Commercial Contributor in writing of such claim, and b) allow the Commercial Contributor to control, and cooperate with the Commercial Contributor in, the defense and any related settlement negotiations. The Indemnified Contributor may participate in any such claim at its own expense.

For example, a Contributor might include the Program in a commercial product offering, Product X. That Contributor is then a Commercial Contributor. If that Commercial Contributor then makes performance claims, or offers warranties related to Product X, those performance claims and warranties are such Commercial Contributor's responsibility alone. Under this section, the Commercial Contributor would have to defend claims against the other Contributors related to those performance claims and warranties, and if a court requires any other Contributor to pay any damages as a result, the Commercial Contributor must pay those damages.

C.5. No Warranty

EXCEPT AS EXPRESSLY SET FORTH IN THIS AGREEMENT, THE PROGRAM IS PROVIDED ON AN "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, EITHER EXPRESS OR IMPLIED INCLUDING, WITHOUT LIMITATION, ANY WARRANTIES OR CONDITIONS OF TITLE, NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Each Recipient is solely responsible for determining the appropriateness of using and distributing the Program and assumes all risks associated with its exercise of rights under this Agreement, including but not limited to the risks and costs of program errors, compliance with applicable laws, damage to or loss of data, programs or equipment, and unavailability or interruption of operations.

C.6. Disclaimer of Liability

EXCEPT AS EXPRESSLY SET FORTH IN THIS AGREEMENT, NEITHER RECIPIENT NOR ANY CONTRIBUTORS SHALL HAVE ANY LIABILITY FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING WITHOUT LIMITATION LOST PROFITS), HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OR DISTRIBUTION OF THE PROGRAM OR THE EXERCISE OF ANY RIGHTS GRANTED HEREUNDER, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

C.7. General

If any provision of this Agreement is invalid or unenforceable under applicable law, it shall not affect the validity or enforceability of the remainder of the terms of this Agreement, and without further action by the parties hereto, such provision shall be reformed to the minimum extent necessary to make such provision valid and enforceable.

If Recipient institutes patent litigation against a Contributor with respect to a patent applicable to software (including a cross-claim or counterclaim in a lawsuit), then any patent licenses granted by that Contributor to such Recipient under this Agreement shall terminate as of the date such litigation is filed. In addition, if Recipient institutes patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Program itself (excluding combinations of the Program with other software or hardware) infringes such Recipient's patent(s), then such Recipient's rights granted under Section 2(b) shall terminate as of the date such litigation is filed.

All Recipient's rights under this Agreement shall terminate if it fails to comply with any of the material terms or conditions of this Agreement and does not cure such failure in a reasonable period of time after becoming aware of such noncompliance. If all Recipient's rights under this Agreement terminate, Recipient agrees to cease use and distribution of the Program as soon as reasonably practicable.

However, Recipient's obligations under this Agreement and any licenses granted by Recipient relating to the Program shall continue and survive.

Everyone is permitted to copy and distribute copies of this Agreement, but in order to avoid inconsistency the Agreement is copyrighted and may only be modified in the following manner. The Agreement Steward reserves the right to publish new versions (including revisions) of this Agreement from time to time. No one other than the Agreement Steward has the right to modify this Agreement. IBM is the initial Agreement Steward. IBM may assign the responsibility to serve as the Agreement Steward to a suitable separate entity. Each new version of the Agreement will be given a distinguishing version number. The Program (including Contributions) may always be distributed subject to the version of the Agreement under which it was received. In addition, after a new version of the Agreement is published, Contributor may elect to distribute the Program (including its Contributions) under the new version. Except as expressly stated in Sections 2(a) and 2(b) above, Recipient receives no rights or licenses to the intellectual property of any Contributor under this Agreement, whether expressly, by implication, estoppel or otherwise. All rights in the Program not expressly granted under this Agreement are reserved.

This Agreement is governed by the laws of the State of New York and the intellectual property laws of the United States of America. No party to this Agreement will bring a legal action under this Agreement more than one year after the cause of action arose. Each party waives its rights to a jury trial in any resulting litigation.

Appendix D. Revision History

Revision 0-0 Tue Aug 7 2012 David Ashley

Initial creation of book by publican

Index

Symbols

A

B

C

D

E

F

G

I

L

M

N

O

P

R

S

T

U

W