

Remote Execution ActiveX Control

**Remote Execution
ActiveX Control
for Microsoft® Windows™**

Version 5.2

**Copyright © 1995 - 2003 by Distinct® Corporation
All rights reserved**

Distinct Corporation

3315 Almaden Expressway
San Jose, CA 95118 USA

Phone: +1 408-445-3270

Fax: +1 408-445-3274

Email: sales@distinct.com

WWW: <http://www.distinct.com>

Disclaimer

Distinct Corporation makes no warranties as to the contents of this documentation and specifically disclaims any implied warranties of merchantability or fitness for any particular purpose. Information in this manual is subject to change without notice and does not represent a commitment on the part of Distinct Corporation. The Software described in this manual is furnished under a license agreement and may be used or copied only in accordance with the terms of that agreement.

Copyright Notice

© 1996 - 2003 by Distinct Corporation. All rights reserved.

No part of this publication may be reproduced, transmitted or translated into any language by any means without the express written permission of Distinct Corporation.

Trademarks

Distinct is a registered trademark and Distinct Remote Execution is a trademark of Distinct Corporation. Windows is a registered trademark of Microsoft Corporation. Other product names are trademarks or registered trademarks of their respective owners.

Last updated June 5th, 2003

Published in the United States of America

Table of Contents

Table of Contents.....	3
1 Overview of the RLIB ActiveX.....	5
1.1 Introduction.....	5
1.2 Usage of the Rexec, Rlogin and rsh ActiveX.....	5
1.3 When you must be a Trusted Host.....	6
1.4 Property Summary for Rexec, Rlogin and Rsh.....	6
1.5 Event Summary for Rexec, Rlogin and Rsh.....	7
1.6 Method Summary for Rexec, Rlogin and Rsh.....	7
1.7 D_RLIB.TXT.....	8
2 Properties for Rexec, Rlogin and Rsh.....	9
2.1 Action.....	9
2.2 BinaryData.....	10
2.3 CmdLinePrompt.....	11
2.4 Command.....	12
2.5 ErrorMessage.....	13
2.6 Host.....	14
2.7 Password.....	15
2.8 PasswordPrompt.....	16
2.9 Protocol.....	17
2.10 Send.....	18
2.11 User.....	19
2.12 UseVariant.....	20
3 Events for Rexec, Rlogin and Rsh.....	21
3.1 OnClose.....	21
3.2 OnConnect.....	22
3.3 OnError.....	23
3.4 OnReceive.....	25
3.5 OnReceiveB.....	26
4 Methods for Rexec, Rlogin and Rsh.....	27
4.1 Abort.....	27
4.2 Disconnect.....	28
4.3 ReceiveB.....	29
4.4 Rexec.....	30
4.5 Rlogin.....	31
4.6 Rsh.....	32
4.7 SendB.....	33
Registry Entries.....	35
Registry Entries for RLIB.....	35
REXEC registry entries.....	35
RLOGIN registry entries.....	35
RSH registry entries.....	36
RCOPY registry entries.....	36

1 Overview of the RLIB ActiveX

1.1 Introduction

The Distinct Remote Execution ActiveX control allows you to integrate remote shell (*rsh*), remote login (*rlogin*) and remote execution (*Rexec*) capabilities into your applications. *rsh* passes a command to a remote host for execution. Both the standard output and errors from that execution are returned to the local host. *rlogin* provides interactive access to remote hosts in a similar way that telnet does. *rexec* is similar in function to *rsh* but requires a user login and password to be sent to the remote host for verification. The Remote Execution ActiveX control is useful for fast development of applications that control or use a remote UNIX server. For example, an application may remotely start UNIX scripts and act on the results.

1.2 Usage of the Rexec, Rlogin and rsh ActiveX

See the section entitled "Using Distinct ActiveX controls in various environments" on how to add the control to your project.

After placing a Remote Execution ActiveX control into a form, some properties can be set at design time. Although property settings default to their most common values, some properties may need to be changed at design time. The PasswordPrompt property defaults to "word:" and the CmdLinePrompt property defaults to "%". These two properties should be preset at design time, although they can also be changed at run time before a session is established. Please check the reference pages of these properties for more details.

The Host property is usually set at run time right before a session is established. This allows the application to request the host name or internet address of the remote server from the user. If an application will always connect to the same host, then the Host property can also be set at design time to minimize user interaction.

The User and Password properties are required to log into the server. The remote user name is specified in the User property and the password is specified in the Password property. The Command property is set to the initial command that is to be executed on the remote server once the connection is established and the user is logged in. The Protocol property identifies the protocol to be used (*rexec*, *rlogin* or *rsh*).

The Action and Send properties can only be accessed at run time. To establish a remote session, the value of the Action property must be set to ACTION_CONNECT (or the Rlogin, Rsh or Rexec method can be used). If the connection can be established, then the OnConnect event will occur before the next line of code is reached. Once the user is logged into the remote server, the command specified by the Command property is executed on the server. The Send property can be accessed to send commands for execution on the remote server once the session is established.

Once a connection is no longer needed, the Action property must be set to ACTION_DISCONNECT (or call the Disconnect method). After the session is disconnected, the OnClose event will occur before the next line of code is reached. Applications must disconnect all connected sessions before terminating.

1.3 When you must be a Trusted Host

In order to make a connection using RSH, REXEC or RCopy you must register the PC running your application (that uses the Remote Execution or Remote Copy ActiveX control) as a trusted host on the Unix machine that you are trying to connect to.

Here is an example of how to make your PC a Trusted Host:

Assuming you are trying to connect to a UNIX system called unix1.mydomain.com (192.0.0.1) and your PC is called mypc.mydomain.com (192.0.0.3) you must update the following files to be able to use REXEC, RSH or RCopy successfully:

Add the following entries in /etc/hosts file on the Unix system called unix1.mydomain.com:

```
192.0.0.3 mypc mypc.mydomain.com
```

Add the following entries in the /etc/hosts.equiv file:

```
mypc
mypc.mydomain.com
```

Updating the /etc/hosts file allows a reference of the remote host by name. In some cases this file may NOT be read at all. If this is your case, you will have to update the appropriate hosts database file on that system. For example if the host uses a name server you will have to modify the name server's database to achieve the desired result.

If on the other hand you need to run RCopy between two UNIX systems, you will need to do all of the above plus: When running the RCopy command, both UNIX systems must have the other system defined in the hosts file and hosts.equiv files. So if your second system is called unix2.mydomain.com, add the following entry in /etc/hosts file

```
192.0.0.1 unix1 unix1.mydomain.com
192.0.0.3 mypc mypc.mydomain.com
```

Add the following entries in the /etc/hosts.equiv file:

```
unix1
unix1.mydomain.com
mypc
mypc.mydomain.com
```

In this case you will also need to add the information for unix2.mydomain.com to the hosts and hosts.equiv files on the unix1.mydomain.com system.

1.4 Property Summary for Rexec, Rlogin and Rsh

Action

Connect, disconnect or abort a remote session

BinaryData

Contains the binary data received following a call to ReceiveB (C#)

CmdLinePrompt

Characters identifying system prompt

Command

Command to be executed on remote machine

ErrorMessage

Error message returned by remote server

Host

Name of host or dotted decimal internet address

Password

Password used during login process

PasswordPrompt

Characters identifying password prompt

Protocol

Protocol used to establish connection

Send

Send buffer

User

User name used during login process

UseVariant

Receive data as binary or ascii

[Back to Top](#)

1.5 Event Summary for Rexec, Rlogin and Rsh

OnClose

Connection has been closed

OnConnect

Connection has been established

OnError

Local error has occurred

OnReceive

More data has been received

OnReceiveB

More data has been received and is ready to be retrieved as binary

[Back to Top](#)

1.6 Method Summary for Rexec, Rlogin and Rsh

Abort

Abort current session

Disconnect

Close connection to server

ReceiveB

Reads binary data

Rexec

Remote execution

Rlogin

Remote login

Rsh

Remote shell

SendB

Send binary data

[Back to Top](#)

1.7 D_RLIB.TXT

The following provides a complete listing of the D_RLIB.TXT definition file. If your application uses more than one Distinct ActiveX control in the same form, then some definitions will conflict. For example, the FTP Client ActiveX control includes the definition

```
Global Const ACTION_DISCONNECT = 3
```

in the D_FTP.TXT file and the Telnet ActiveX control includes the definition

```
Global Const ACTION_DISCONNECT = 2
```

in the D_TNET.TXT file. To avoid this conflict, you must rename at least one of the constants (for example, FTP_ACTION_DISCONNECT or TNET_ACTION_DISCONNECT).

```
' Remote Execution ActiveX Control
' (C) Copyright 1995 - 1998 by Distinct Corporation
' All rights reserved

' actions
Global Const ACTION_NONE = 0
Global Const ACTION_CONNECT = 1
Global Const ACTION_DISCONNECT = 2
Global Const ACTION_ABORT = 3

' protocols
Global Const PROTOCOL_REXEC = 0
Global Const PROTOCOL_RLOGIN = 1
Global Const PROTOCOL_RSH = 2

' error codes
Global Const ERR_CANNOT_CONNECT = 1
Global Const ERR_HOST_NOT_DEFINED = 2
Global Const ERR_USER_NOT_DEFINED = 3
Global Const ERR_CHANGE_PROTOCOL = 4
Global Const ERR_NEED_PASSWORD = 5
Global Const ERR_IN_ACTION = 6
Global Const ERR_CONNECT_TO_SEND = 7
Global Const ERR_CANNOT_SEND = 8
Global Const ERR_UNABLE_TO_LOAD = 9 Back to Top
```

2. Properties for Rexec, Rlogin and Rsh

2.1 Action

Summary

Connect, disconnect or abort a remote session.

Description

The Action property controls the connection state of the Remote Execution ActiveX control. A session can be established, closed or aborted by assigning one of the following values to this property.

Value	Meaning
ACTION_CONNECT	Establish session.
ACTION_DISCONNECT	Close session.
ACTION_ABORT	Abort session.

This property can be changed at run time only.

Before setting the Action property to ACTION_CONNECT, the following properties must be initialized. The Host property must be set to the name or internet address (in dotted decimal notation) of the remote server. The Protocol property must be set to the protocol to be used for establishing the connection. The User and Password properties must contain a valid user name and password to complete the login process. The Command property must contain the command to be executed on the remote machine once a connection has been established.

If the connection can be established, then the OnConnect event will be fired. If the connection cannot be established, then the OnError event will be fired. These events will occur before the next statement (i.e. the statement following the assignment of ACTION_CONNECT to the Action property) is executed. The application should set a flag in the OnConnect and OnError events, so that it can determine if the session has been established or not. In addition, the application may want to display an error message in the OnError event to inform the user that the connection has not been established. Please check the reference page of the OnError event for a complete listing of error codes.

Once a connection is no longer needed, the session can be terminated by setting the Action property to ACTION_DISCONNECT. An application must close all connections it has created before it quits. A connection can also be closed by setting the Action property to ACTION_ABORT. This action resets and closes the connection without properly closing down and should not be called under normal circumstances.

The Rsh, Rlogin, Rexec, Disconnect and Abort methods accomplish the same as the above actions. Please check the reference pages of these methods for more detailed information on their usage. There is no default value for this property.

Example

```

RUtils.Host = "speedy"
RUtils.User = "santa"
RUtils.Password = "north pole"
RUtils.Protocol = PROTOCOL_REXEC
RUtils.Command = "ls -al"
RUtils.Action = ACTION_CONNECT

```

2.2 BinaryData

Summary

Contains the binary data received following a call to the ReceiveB method.

Description

After each call to the ReceiveB method in a C# application this property needs to be read to access binary data.

Example

```
Private void Rlib_OnReceive (object sender, System.EventArgs e)
{
    int len;
    int bytes;
    Object arrData = new byte[bytes];

    Len = ax Rlib1.ReceiveB (arrData);
    If (len > 0)
    {
        object pBuf = new Byte[len];
        pBuf = ax Rlib1.BinaryData;
        byte []RcvByte = (byte [])pBuf;
        .....
        .....
    }
}
```

2.3 CmdLinePrompt

Summary

Characters identifying system prompt.

Description

The CmdLinePrompt property specifies the command line prompt for which the ActiveX control must wait before executing a command using the RLOGIN protocol. The ActiveX control will wait for this system prompt from the remote system before sending the command to be executed. Once the system prompt is received by the ActiveX control, the control will execute the command specified in the Command property on the remote machine.

The default value of this property is "%". This should be suitable for most applications, but any other value is legal. If an application connects to a host or multiple hosts that send different system prompts, then all these prompts must be assigned to the CmdLinePrompt property to ensure proper performance. Multiple system prompts for different hosts may be separated by commas, for example, "%,>,#". Because the prompt is case sensitive, the first character of each word should be skipped as it may be sent in upper or in lower case.

This property can be changed at design time and at run time before a connection has been established. The default value for this property is "%".

Example

```
RUtils.CmdLinePrompt = "%,>,#"
```

[Back to Property Summary](#)

2.4 Command

Summary

Command to be executed on remote machine.

Description

The Command property specifies the command that should be executed on the remote system as soon as the connection is established. This property must initially be set before setting the Action property to ACTION_CONNECT.

Usually, this property is only set before a connection has been established. Once connected, the Send property may also be used to send further data or commands to the remote server. The REXEC protocol requires that the Command property contain the command to be executed on the remote server because the connection is closed by the server once the initial command is executed. The RLOGIN protocol, however, allows the application to use the Send property to send further remote commands.

This property can be set at design time or at run time before and sometimes even after a connection is established. There is no default value for this property.

Example

```
RUtils.Host = "speedy.distinct.com"  
RUtils.User = "santa"  
RUtils.Password = "north pole"  
RUtils.Protocol = PROTOCOL_RSH  
RUtils.Command = "ls -al"  
RUtils.Action = ACTION_CONNECT
```

[Back to Property Summary](#)

2.5 ErrorMessage

Summary

Error message returned by remote server.

Description

The ErrorMessage property contains the response sent by the remote server if a connection could not be established or if the command could not be properly executed for some other reason. This property will contain the exact response from the server explaining the error that occurred.

If setting the Action property to ACTION_CONNECT returns an error, then the ErrorMessage property can be used to retrieve the exact error message sent by the host. This message will describe the reason for the failure of ACTION_CONNECT.

This property can only be read at run time. There is no default value for this property.

Example

```
MsgBox RUtils.ErrorMessage, 64, "Sample Program"
```

[Back to Property Summary](#)

2.6 Host

Summary

Name of host or dotted decimal internet address.

Description

The Host property specifies the name or internet address of a remote server. This property must be set before a session can be established. There are three possible ways of specifying a remote server.

Machine Name

An application only needs to specify the name of the remote server if the server is located on the same network as the local PC or if the internet address of the server is defined in the local host table. If the remote server is not on the local network, then the underlying protocol will route the traffic through a gateway. If the remote server is not defined in the local host table, then the underlying protocol will contact the domain server to resolve the internet address of the server.

Machine and Domain Name

An application needs to specify the machine name and the domain name if the remote server is not located on the same network as the local PC. Fully domain qualified machine names are written from left to right in ascending order (for example, *speedy.distinct.com*). If both machine and domain names are specified, then the underlying protocol will contact the domain server to resolve the internet address of the server.

Internet Address

Sometimes the user knows only the internet address of the remote server that he or she wants to use. In this case, the internet address can be entered in what is known as the dotted decimal notation (for example, *127.43.101.12*). If the remote server identified by this address is not on the local network, then the underlying protocol will route the traffic through a gateway.

This property can be changed at design time and at run time before a connection has been established. There is no default value for this property.

Example

```
RUtils.Host = "127.43.101.12"  
RUtils.User = "santa"  
RUtils.Password = "north pole"  
RUtils.Protocol = PROTOCOL_RLOGIN  
RUtils.Action = ACTION_CONNECT
```

[Back to Property Summary](#)

2.7 Password

Summary

Password used during login process.

Description

The Password property is used during the login process to the server. Most remote servers require both a user name and a correct password before a session can be established. Before connecting to a server, the application must therefore also set the User property to the correct user name.

For security reasons, both the user name and the password are usually obtained from the user just before a connection is established. If security is not an important issue and the application will always connect to the same remote server with the same user name and password, then these two properties can be set at design time.

This property can be changed at design time and at run time before a connection has been established. There is no default value for this property.

Example

```
RUtils.Host = "speedy"  
RUtils.User = "santa"  
RUtils.Password = "north pole"  
RUtils.Protocol = PROTOCOL_REXEC  
RUtils.Command = "ls -al"  
RUtils.Action = ACTION_CONNECT
```

[Back to Property Summary](#)

2.8 PasswordPrompt

Summary

Characters identifying password prompt.

Description

The PasswordPrompt property specifies the password prompt for which the ActiveX control must wait before sending the password during a login procedure using the RLOGIN protocol. The ActiveX control will wait for this password prompt to be issued by the remote system before sending the password specified by the Password property. If no password is specified, then the application can still send the password to the remote system by assigning it to the Send property.

The default value of this property is "word:". This should be suitable for most applications, but any other value is legal. If an application connects to a host or multiple hosts that send different prompts during the login process, then all these prompts must be assigned to the PasswordPrompt property to ensure proper performance. Multiple prompts for different hosts may be separated by commas, for example, "word:;,sername:;,ogin:". Because the prompt is case sensitive, the first character of each word should be skipped as it may be sent in upper or in lower case.

This property can be changed at design time and at run time before a connection has been established. The default value for this property is "word:".

Example

```
RUtils.PasswordPrompt = "word:"
```

[Back to Property Summary](#)

2.9 Protocol

Summary

Protocol used to establish connection.

Description

The Protocol property specifies the protocol that the control should use to establish a connection. This property can be set to one of the following values.

Value	Meaning
PROTOCOL_REXEC	Remote execution.
PROTOCOL_RLOGIN	Remote login.
PROTOCOL_RSH	Remote shell.

The REXEC protocol can be used to execute a single command on a remote machine. The application must provide the user name and password for login verification. The command is executed on the remote machine and the result is sent to the application. The REXEC protocol terminates as soon as it executes the remote command.

The RLOGIN protocol connects your virtual terminal on the local host to the remote host system. Once the user is logged in, the connection is terminated only when the application executes a disconnect command on the remote machine or sets the Action property to ACTION_DISCONNECT.

The RSH protocol connects to the remote host and executes the specified command. The remote user name is required to execute the command. The RSH protocol normally terminates as soon as the remote command is executed.

This property can be changed at design time and at run time before a connection has been established.

Example

```
RUtills.Host = "speedy.distinct.com"  
RUtills.User = "santa"  
RUtills.Password = "north pole"  
RUtills.Protocol = PROTOCOL_RSH  
RUtills.Action = ACTION_CONNECT
```

[Back to Property Summary](#)

2.10 Send

Summary

Send buffer.

Description

The Send property is used to send a command to be executed on the remote server or to transfer data to the remote server. The command or data is sent by assigning a string to this property. Care should be taken not to exceed the buffer and transport capabilities of the underlying protocol stack. In general, no more than 512 bytes of data should be assigned to this property at any one time. Most TCP/IP stacks are able to transfer multiple 512 byte chunks of data in quick succession.

Before a session is established, the application must set the Command property to the first command to be executed. Once the connection has been established and the initial command has been executed, some protocols allow the application to send additional commands to the server. Until the connection is closed by either side, additional commands may be sent to the server by assigning them to the Send property.

This property can only be written to at run time while a connection is established. There is no default value for this property.

Example

```
RUtils.Send = "This is a test"
```

[Back to Property Summary](#)

2.11 User

Summary

User name used during login process.

Description

The User property is used during the login process to the server. Most remote servers require both a user name and a correct password before a session can be established. Before connecting to a server, the application must also set the Password property to the correct password.

For security reasons, both the user name and the password are usually obtained from the user just before a connection is established. If security is not an important issue and the application will always connect to the same remote server with the same user name and password, then these two properties can be set at design time.

This property can be changed at design time and at run time before a connection has been established. There is no default value for this property.

Example

```
RUtils.Host = "127.43.101.12"  
RUtils.User = "santa"  
RUtils.Password = "north pole"  
RUtils.Protocol = PROTOCOL_RLOGIN  
RUtils.Action = ACTION_CONNECT
```

[Back to Property Summary](#)

2.12 UseVariant

Summary

Receive binary or ascii data.

Description

The UseVariant property is used to specify whether data is to be received or sent in binary or ASCII form. If this property is set to True, the OnReceiveB event will be fired when data arrives; otherwise, the OnReceive event is fired.

This property should be set before any data arrives This property can be read at any time. The default value for this property is False.

Example

```
RUtils.UseVariant = True
```

[Back to Property Summary](#)

3 Events for Rexec, Rlogin and Rsh

3.1 OnClose

Summary

Connection has been closed.

Description

The OnClose event occurs usually in response to setting the Action property to ACTION_DISCONNECT or ACTION_ABORT (or in response to the Disconnect or Abort methods). In most cases, the remote server may close a connection (for example REXEC). This will also trigger an OnClose event. In this case, the application must still set the Action property to ACTION_DISCONNECT (or use the Disconnect method) to free up all the resources allocated for the connection. However, this should be done with care during the OnClose event because it might result in an infinite loop.

If this event occurs in response to setting the Action property to ACTION_DISCONNECT (or in response to the Disconnect method), it will occur before the statement following the assignment of ACTION_DISCONNECT (or the call to the Disconnect method) to the Action property is reached.

Normally, an application should simply set a flag in response to this event. Then, this flag can be checked directly after the ACTION_DISCONNECT action (or the Disconnect method) to make sure that the connection was actually terminated.

Example

```
Sub RUtils_OnClose ()  
  If Connected = True Then  
    Connected = False  
    RUtils.Action = ACTION_DISCONNECT  
  End If  
End Sub
```

[Back to Event Summary](#)

3.2 OnConnect

Summary

Connection has been established.

Description

The OnConnect event occurs in response to setting the Action property to ACTION_CONNECT (or in response to the Rexec, Rlogin, or Rsh method). This event will occur before the statement following the assignment of ACTION_CONNECT to the Action property (or the call to the Rexec, Rlogin, or Rsh method) is reached.

Normally, an application should simply set a flag in response to this event. Then, this flag can be checked directly after the ACTION_CONNECT action (or the Rexec, Rlogin, or Rsh method) to make sure that the connection was actually established.

If the connection could not be established, then the OnError event will be called instead of the OnConnect event.

While handling the OnConnect event, an application should not perform tasks which have the potential of requiring a lot of time to complete, such as generating a message box.

Example

```
Sub RUtils_OnConnect ()  
    Connected = True  
End Sub
```

[Back to Event Summary](#)

3.3 OnError

Summary

Local error has occurred.

Description

The OnError event occurs when a property is accessed in an illegal way or when a connection with the remote server cannot be established. The table below lists all possible error codes delivered by this event.

Value	Meaning
ERR_CANNOT_CONNECT	Unable to connect to remote host.
ERR_HOST_NOT_DEFINED	Host name must be defined before connecting.
ERR_USER_NOT_DEFINED	User name must be defined before connecting.
ERR_CHANGE_PROTOCOL	Cannot change protocol while connected.
ERR_NEED_PASSWORD	Password required for login.
ERR_IN_ACTION	Another action is in progress.
ERR_CONNECT_TO_SEND	Must be connected before accessing send buffer.
ERR_CANNOT_SEND	Unable to send command/data to remote host.

The following describes each error in more detail.

ERR_CANNOT_CONNECT

The remote server is unreachable. The Protocol, Host, User or Password properties may be set incorrectly or the host may be down.

ERR_HOST_NOT_DEFINED

The Host property must be set before attempting to establish a connection.

ERR_USER_NOT_DEFINED

The User property must be set before attempting to establish a connection.

ERR_CHANGE_PROTOCOL

An attempt was made to change the Protocol property while connected.

ERR_NEED_PASSWORD

Password required to complete login. During the RLOGIN process, if the remote system requests a password and the Password property is not set, then this error will occur. The application can still send the password and the command using the Send property once the connection is established.

ERR_IN_ACTION

Another action is already in progress.

ERR_CONNECT_TO_SEND

An attempt was made to access the Send property without being connected.

ERR_CANNOT_SEND

Unable to send the command or data to the remote server.

Example

```
Sub RUtils_OnError (ErrorCode As Integer)
  If ErrorCode = ERR_CANNOT_CONNECT Then
    MsgBox "Unable to connect to remote host", 64, "Sample Program"
```

End If
End Sub

[Back to Event Summary](#)

3.4 OnReceive

Summary

More data has been received.

Description

The OnReceive event occurs whenever the underlying protocol stack receives one or more bytes of additional data from the remote server. This event actually delivers the data to the application.

The OnReceive event occurs in response to a connection request or the execution of a command on the remote server. In either case, the data delivered will contain the response from the remote server. One or more OnReceive events may occur to deliver the complete response.

While handling the OnReceive event, an application should not perform tasks which have the potential of requiring a lot of time to complete, such as generating a message box. A substantial delay could cause the application to not receive subsequent OnReceive events.

Example

```
Sub RUtils_OnReceive (Buffer As String)
    Dim Message As String

    Message = Buffer
End Sub
```

[Back to Event Summary](#)

3.5 OnReceiveB

Summary

More data has been received and is ready to be retrieved as binary.

Description

The OnReceiveB event is fired only if the UseVariant property is set to True. If the UseVariant property is False, the OnReceive event will be fired instead.

The OnReceiveB event occurs whenever the underlying protocol stack receives one or more bytes of additional data from the remote server. The OnReceiveB event does not actually deliver the data to the application but instead expects that the application will call the ReceiveB method to obtain all data waiting in the buffer. The *Bytes* argument to the OnReceiveB event indicates how many bytes of data are available to be retrieved.

The OnReceiveB event occurs in response to a connection request or the execution of a command on the remote server. In either case, the data retrieved with the ReceiveB method will contain the response from the remote server. One or more OnReceiveB events may occur to retrieve the complete response.

While handling the OnReceiveB event, an application should not perform tasks which have the potential of requiring a lot of time to complete, such as generating a message box. A substantial delay could cause the application to not receive subsequent OnReceiveB events.

Example

```
Sub RUtils_OnReceiveB (Bytes As Long)
    Dim Buffer(1 To Bytes) As Byte
    Dim Siz As Long
    Dim i As Integer

    Siz = RUtils.ReceiveB (Buffer, Bytes)
    If Siz > 0 Then
        Open "c:\abc.exe" For Binary Access Write As #1
        For i = 1 To Siz
            Put #1, i, Buffer (x)
        Next i
        Close #1
    Else
        MsgBox "Cannot receive binary data", 64, "Sample Program"
    End If
End Sub
```

[Back to Event Summary](#)

4. Methods for Rexec, Rlogin and Rsh

4.1 Abort

Summary

Abort current session.

Syntax

Boolean Abort ()

Description

The Abort method aborts a session to the remote server. This method resets and closes the connection without properly closing down and should not be called under normal circumstances.

The Abort method takes no parameters and returns a boolean. If the connection is successfully reset and closed, then the method returns True; otherwise, it returns False. The application should ensure that the method was successfully executed by checking the return value.

Calling this method is equivalent to setting the Action property to ACTION_ABORT.

Example

```
Result = RUtils.Abort ()  
If Result = False Then  
    MsgBox "Unable to abort", 64, "Sample Program"  
End If
```

[Back to Method Summary](#)

4.2 Disconnect

Summary

Closes connection to server.

Syntax

Boolean Disconnect ()

Description

Once a connection is no longer needed, the session can be terminated by calling the Disconnect method. An application must close all connections it has created before it quits.

The Disconnect method takes no parameters and returns a boolean. If a connection is successfully terminated, then the method returns True; otherwise, it returns False. The application should ensure that the method was successfully executed by checking the return value.

Calling this method is equivalent to setting the Action property to ACTION_DISCONNECT.

Example

```
Result = RUtils.Disconnect ()  
If Result = False Then  
    MsgBox "Unable to disconnect from server", 64, "Sample Program"  
End If
```

[Back to Method Summary](#)

4.3 ReceiveB

Summary

Retrieve binary data.

Syntax

Long ReceiveB (*Buffer*, *Bytes*)

Buffer Variant

Bytes Long

Description

The ReceiveB method retrieves the binary data and passes to the application.

The ReceiveB method takes a buffer (*Buffer*) and the number of bytes to read (*Bytes*) as its parameters and returns the actual number of bytes retrieved. The *Buffer* parameter should be an array of bytes data type. The *Bytes* parameter specifies how many bytes should be retrieved.

If there are less data to be read than the requested *Bytes*, only the available data are read. This is reflected in the return value. If the returned value is less than the requested *Bytes*, that means there are less data available than requested. The application should also check to make sure what is the actual number of bytes read.

Example

```
Length = RUtils.ReceiveB (Buffer, Bytes)
```

[Back to Method Summary](#)

4.4 Rexec

Summary

Remote execution.

Syntax

Boolean Rexec (*Host*, *User*, *Password*, *Command*)

<i>Host</i>	String
<i>User</i>	String
<i>Password</i>	String
<i>Command</i>	String

Description

The Rexec method is used to execute a single command on a remote machine. First, a connection to the remote machine is established. Then, the command is executed on the remote machine and the result is sent to the application. The method terminates as soon as it executes the remote command.

The Rexec method takes a host name (*Host*), a user name (*User*), a password (*Password*), and the command to execute (*Command*) as its parameters and returns a boolean. The host name must be set to the name or internet address (in dotted decimal notation) of the remote server. The user name and password must be valid to complete the login process. The command must contain the command to be executed on the remote machine once a connection has been established. If the remote command can be successfully executed, then the method returns True; otherwise, it returns False. The application should ensure that the method was successfully executed by checking the return value.

If the connection can be established, then the OnConnect event will be fired. If the connection cannot be established, then the OnError event will be fired. These events will occur before the next statement (i.e. the statement following the call to the Rexec method) is executed. The application should set a flag in the OnConnect and OnError events, so that it can determine if the session has been established or not. In addition, the application may want to display an error message in the OnError event to inform the user that the connection has not been established. Please check the reference page of the OnError event for a complete listing of error codes.

Calling this method is equivalent to setting the Action property to ACTION_CONNECT with the Protocol property set to PROTOCOL_REXEC.

Example

```
Result = RUtils.Rexec ("speedy.distinct.com", "santa", "north pole", "ls -la")
If Result = False Then
    MsgBox "Remote execution failed", 64, "Sample Program"
End If
```

[Back to Method Summary](#)

4.5 Rlogin

Summary

Remote login.

Syntax

Boolean Rlogin (*Host, User, Password, Command*)

<i>Host</i>	String
<i>User</i>	String
<i>Password</i>	String
<i>Command</i>	String

Description

The Rlogin method connects your virtual terminal on the local host to the remote host system. Once the user is logged in, the connection is terminated only when the application executes a disconnect command on the remote machine, sets the Action property to ACTION_DISCONNECT, or calls the Disconnect method.

The Rlogin method takes a host name (*Host*), a user name (*User*), a password (*Password*), and the command to execute (*Command*) as its parameters and returns a boolean. The host name must be set to the name or internet address (in dotted decimal notation) of the remote server. The user name and password must be valid to complete the login process. The command must contain the command to be executed on the remote machine once a connection has been established. If the remote command can be successfully executed, then the method returns True; otherwise, it returns False. The application should ensure that the method was successfully executed by checking the return value.

If the connection can be established, then the OnConnect event will be fired. If the connection cannot be established, then the OnError event will be fired. These events will occur before the next statement (i.e. the statement following the call to the Rlogin method) is executed. The application should set a flag in the OnConnect and OnError events, so that it can determine if the session has been established or not. In addition, the application may want to display an error message in the OnError event to inform the user that the connection has not been established. Please check the reference page of the OnError event for a complete listing of error codes.

Calling this method is equivalent to setting the Action property to ACTION_CONNECT with the Protocol property set to PROTOCOL_RLOGIN.

Example

```
Result = RUtils.Rlogin ("speedy.distinct.com", "santa", "north pole", "ls -la")
If Result = False Then
    MsgBox "Remote login failed", 64, "Sample Program"
End If
```

[Back to Method Summary](#)

4.6 Rsh

Summary

Remote shell.

Syntax

Boolean Rsh (*Host*, *User*, *Command*)

<i>Host</i>	String
<i>User</i>	String
<i>Command</i>	String

Description

The Rsh method connects to the remote host and executes the specified command. The remote user name is required to execute the command. The remote shell command normally terminates as soon as the remote command is executed.

The Rsh method takes a host name (*Host*), a user name (*User*), and the command to execute (*Command*) as its parameters and returns a boolean. The host name must be set to the name or internet address (in dotted decimal notation) of the remote server. The user name must be valid to complete the login process. The command must contain the command to be executed on the remote machine once a connection has been established. If the remote command can be successfully executed, then the method returns True; otherwise, it returns False. The application should ensure that the method was successfully executed by checking the return value.

If the connection can be established, then the OnConnect event will be fired. If the connection cannot be established, then the OnError event will be fired. These events will occur before the next statement (i.e. the statement following the call to the Rsh method) is executed. The application should set a flag in the OnConnect and OnError events, so that it can determine if the session has been established or not. In addition, the application may want to display an error message in the OnError event to inform the user that the connection has not been established. Please check the reference page of the OnError event for a complete listing of error codes.

Calling this method is equivalent to setting the Action property to ACTION_CONNECT with the Protocol property set to PROTOCOL_RSH.

Example

```
Result = RUtils.Rsh ("speedy.distinct.com", "santa", "ls -la")
If Result = False Then
    MsgBox "Remote shell failed", 64, "Sample Program"
End If
```

[Back to Method Summary](#)

4.7 SendB

Summary

Send binary data.

Syntax

Boolean SendB (*Buffer*, *Bytes*)

Buffer Variant

Bytes Long

Description

The SendB method is used to transfer binary data to the remote server. Care should be taken not to exceed the buffer and transport capabilities of the underlying protocol stack. In general, no more than 512 bytes of data should be assigned to this property at any one time. Most TCP/IP stacks are able to transfer multiple 512 byte chunks of data in quick succession.

The SendB method takes a variant buffer (*Buffer*) and a long byte (*Bytes*) as its parameters and returns a boolean. The variant buffer has to be set to the data to be sent. The bytes parameter has to be set to the number of bytes of data (or the size of the buffer parameter) to be sent. If the operation is successful, then the method returns True; otherwise, it returns False. The application should ensure that the method was successfully executed by checking the return value.

This method can only be called at run time while a connection is established.

Example

```
Dim Buffer() As Byte
Dim Bytes As Long
Dim i As Integer

Open "c:\abc.exe" For Binary Access Read As #1
Bytes = FileLen("c:\abc.exe")
For i = 1 To Bytes
    Get #1, , Buffer(i)
Next i
Result = RUtils.SendB(Buffer, Bytes)
If Result = False Then
    MsgBox "Cannot send binary data", 64, "Sample Program"
End If
Close #1
```

[Back to Method Summary](#)

Registry Entries for Rexec, Rlogin and Rsh

Registry Entries

Registry Entries for RLIB

Various parameters used to control the behavior of the Distinct RLIB library are stored under the following registry path.

HKEY_LOCAL_MACHINE\SOFTWARE\Distinct\DLLS\RLIB32

The following entries define the settings that can be modified through registry entries for each of the remote commands.

REXEC registry entries

RexecTimeout REG_DWORD 1-999

Specifies the default timeout in seconds used while connecting to an rexec server. The default value is 20 seconds.

RexecLog REG_DWORD 1-5

Specifies the queue size. The default value is 5.

RexecSendBuf REG_DWORD 1024-32000

Specifies the size of the send buffer. The default value is 32000 bytes.

RexecReceiveBuf REG_DWORD 1024-32000

Specifies the size of the receive buffer. The default value is 32000 bytes.

RLOGIN registry entries

RloginTimeout REG_DWORD 1-999

Specifies the default timeout in seconds used while connecting to an rlogin server. The default value is 20 seconds.

RloginSendBuf REG_DWORD 1024-32000

Specifies the size of the send buffer. The default value is 4096 bytes.

RloginReceiveBuf REG_DWORD 1024-32000

Specifies the size of the receive buffer. The default value is 4096 bytes.

RSH registry entries

RshTimeout REG_DWORD 1-999

Specifies the default timeout in seconds used while connecting to an rsh server. The default value is 20 seconds.

RshLog REG_DWORD 1-5

Specifies the queue size. The default value is 5.

RshSendBuf REG_DWORD 1024-32000

Specifies the size of the send buffer. The default value is 4096 bytes.

RshReceiveBuf REG_DWORD 1024-32000

Specifies the size of the receive buffer. The default value is 4096 bytes.

RCOPY registry entries

RcpTimeout REG_DWORD 1-999

Specifies the default timeout in seconds used while connecting to an rcp server. The default value is 20 seconds.

RcpErrorBuffer REG_DWORD 128-32000

Specifies the size of the error buffer. The default value is 2048 bytes.

RcpMaxErrors REG_DWORD 1-999

Specifies the maximum number of errors that are allowed. If the number of errors is greater than this value, then the function will terminate. The default value is 50.