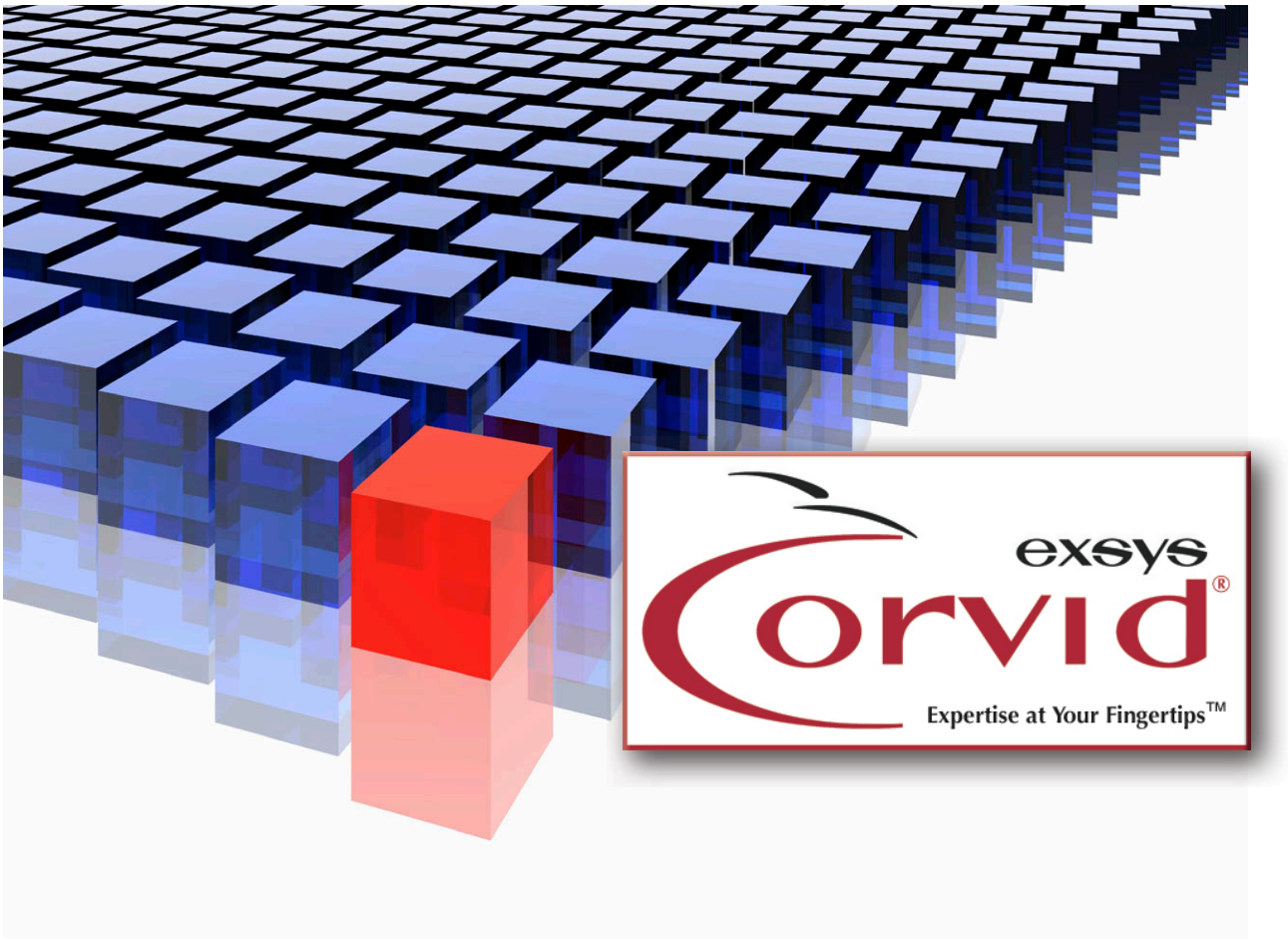


Exsys Corvid v5.3 New Features Overview

Exsys Corvid v5.3 has several new features that will make system development easier for all users. The 3 most important are:

- Exsys has a new, completely rewritten Corvid manual reflecting over 11 years of teaching Corvid and helping developers understand its advanced features. The new manual makes it much easier to learn, understand and use Corvid both for beginners and advanced users.

- A new Trace feature in the Applet Runtime allows examining the details of the variables, rules and Logic Blocks in a system as it is running. Whenever a system asks a question or displays a screen, the detailed state of the system can be reviewed.

- A new look for systems run with either the default screen for the Corvid Applet Runtime or using the Corvid Servlet Runtime with default templates.

In addition, there are several other enhancements that make development faster and easier.

1. The New Corvid Manual

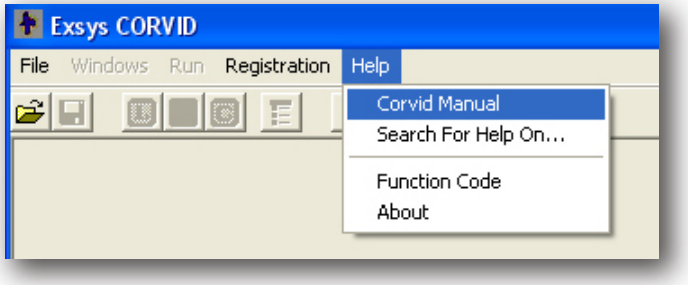
The new Corvid Manual is a complete rewrite of the entire manual. The previous manual had been evolving over the past 11 years, but due to updates and changes in the program, was not always organized in the best way. To remedy this, the entire manual has been rewritten and restructured to reflect the many years of teaching Corvid and helping developers understand how to use its many features.

The new manual makes it far easier to develop systems with Corvid. Chapters now not only explain the Corvid controls, but provide step by step instructions on implementing the functionality often used by most developers. Any areas where we have found developers typically having problems or getting confused are clarified in detail.

The manual is also structured to make it easier for new developers. It is designed so that beginning chapters explain the fundamentals that all developers should know, without trying to explain advanced functions. These advanced functions, while very useful later, may be confusing to beginning users. Later chapters cover advanced functions in a way that makes it easier to use them and actually implement advanced features in a system.

Even experienced Corvid developers will almost certainly learn new capabilities and techniques by reading the manual. If there is a portion of Corvid that has been confusing, read the chapter on it. If you are new to Corvid, or have not used it in a while, start from the beginning and you will be soon be building systems.

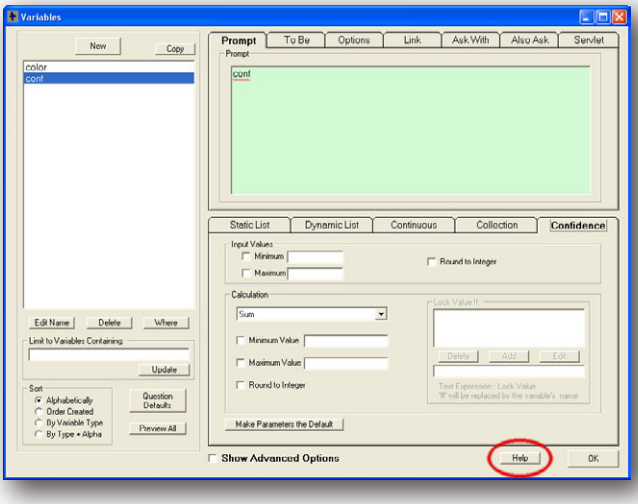
The full manual can be displayed by selecting the “Corvid Manual” menu item under the “Help” menu.

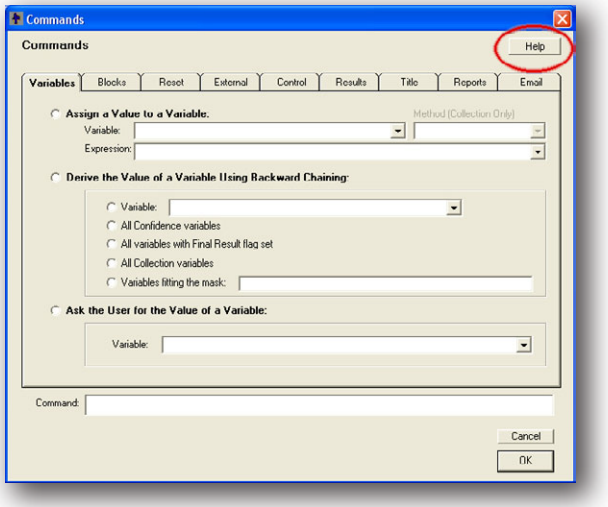


The manual is a PDF document and it will be displayed with Adobe Acrobat Reader. (If Acrobat Reader is not installed on your PC, it can be downloaded from **www.adobe.com** for free.)

Adobe Acrobat makes it easy to search the manual for any commands or functions. In addition, most complex Corvid windows have a “Help” button on them. Clicking the “Help” button will open the manual in Acrobat Reader to the section that explains the specific window. This context sensitive help makes it easy to get an explanation of how to use any of Corvid’s many windows.

Typically the PDF version of the manual is far easier to search and use than a printed version. PDFs are easy to search and with Acrobat Reader 10, PDFs can even be annotated. Due to this, and to reduce the use of natural resources, printed versions of the manual are only available by request. If you feel that a printed version of the manual is needed, please contact info@exsys.com.





2. Running Systems with TRACE

Corvid provides a way to run systems in conjunction with a new “Trace Applet” that allows examining the status of variables, rules, Logic Blocks and the goal stack, along with a detailed history of the session.

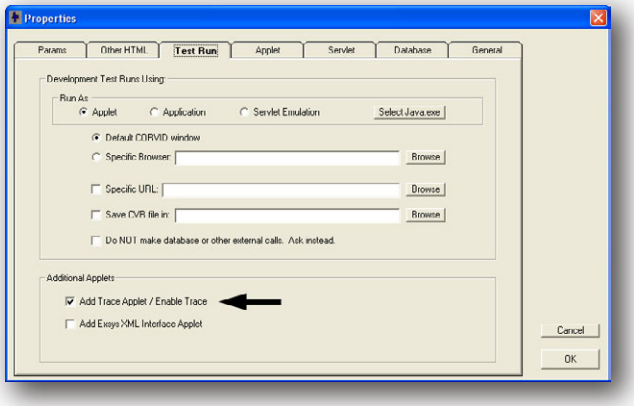
To use the Trace Applet, the system must be running with the Corvid Applet Runtime. The Servlet Runtime and running as an Application only support the portion of trace that is the history of the session - not the interactive examination of variables and rules.

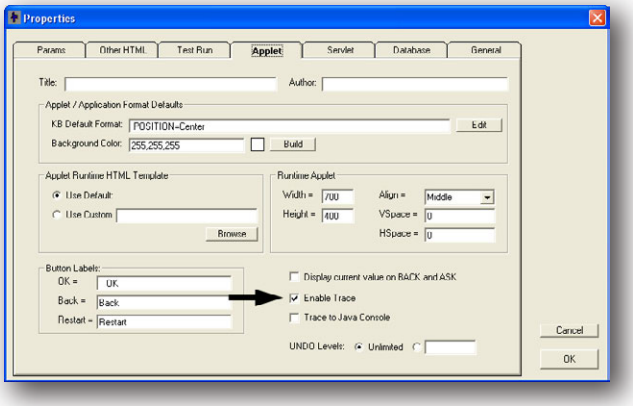
Activating Trace

To activate the Trace Applet, open the Properties window and either:

1. Open the “Test Run” tab and select “Add Trace Applet / Enable Trace”.
2. Open the “Applet” tab and select “Enable Trace”.

These two options are identical and selecting / unselecting one of the checkboxes will select / unselect the other automatically.

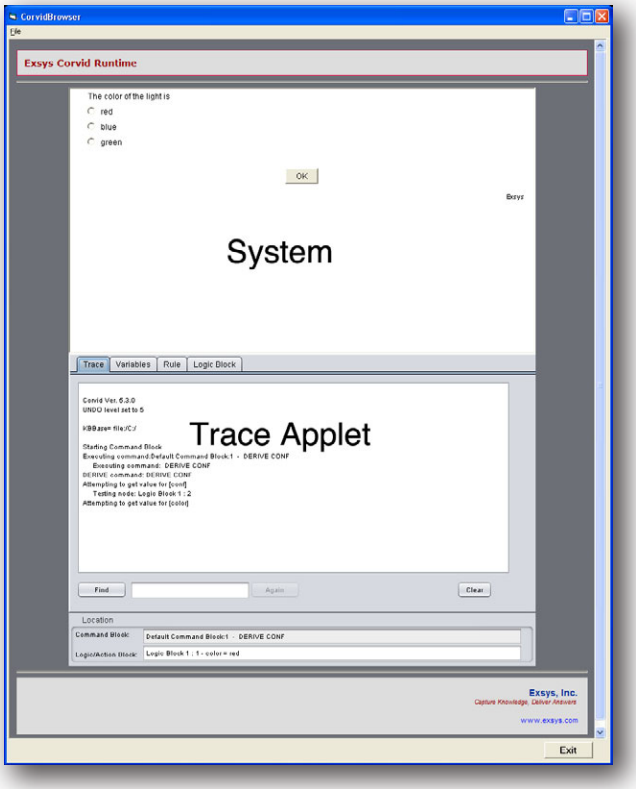




Running the System with Trace

When the system is running, the Trace Applet will automatically be added, and communicate with the system being run.

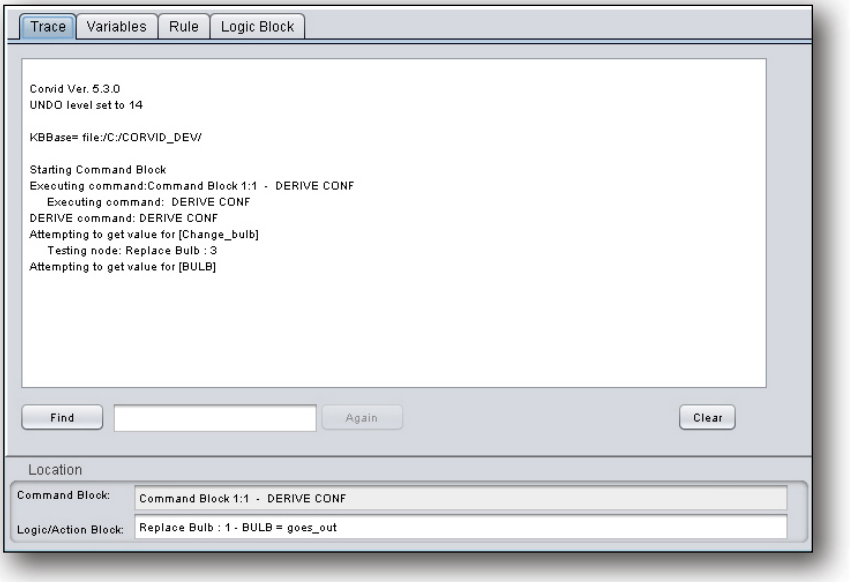
> **NOTE:** For the trace, the system should be run with either the default Corvid template or a modified template that uses the “CORVID_RUNTIME_APPLET” marker rather than hard coding the apple tag. Corvid will automatically add the trace applet along with the applet running the system to replace the “CORVID_RUNTIME_APPLET”.



Trace Applet Tabs

The Trace Applet has 4 tabs at the top that allow examining various information about the session. In addition the bottom of the window always displays the current Command Block command and current Logic Block line being executed. In some cases, where the Command Block command does not require use of a Logic Block (such as an ASK or RESULTS command), the Logic Block line may be blank.

Trace Tab



The Trace tab displays a history of the session.

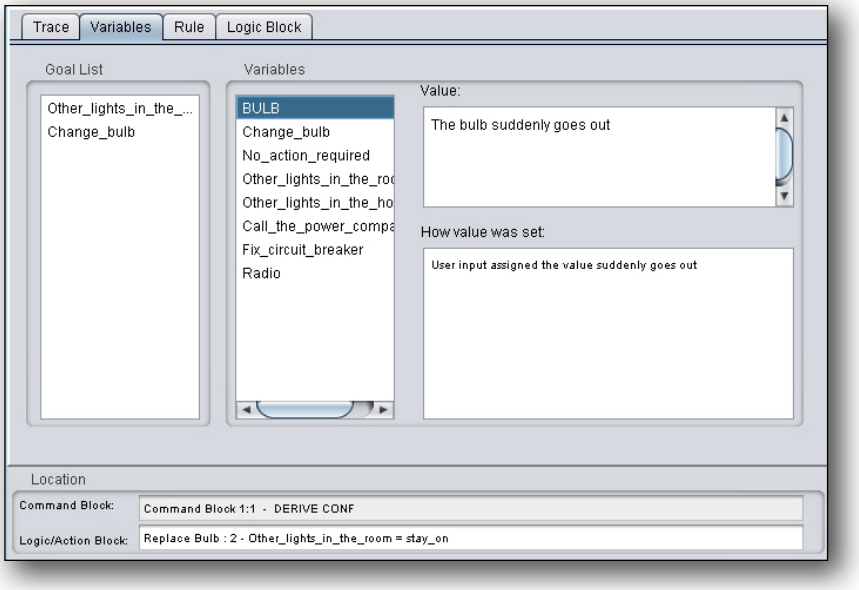
This is the same trace data as was displayed in earlier versions of Corvid (ver 5.2 and before), and which is displayed when trace is used with the Servlet Runtime or console window.

The trace displays the steps that the system executes during the session.

The trace history can be searched for any text. This can be used to search for any actions that were taken for a particular variable by searching on the variable's name. Enter the text to search for and click "Find". To search again, click the "Again" button.

Since the trace history can get quite long, especially for MetaBlock systems, the "Clear" button allows clearing the existing text. The subsequent trace information will be written to the empty window. This can make searching for specific actions easier in MetaBlock systems.

Variables Tab



The Variables Tab allows examining both the current backward chaining Goal List and the value of any variable in the system.

Variables

The "Variables" list in the center of the window displays all the variables in the system. To see the current value of any variable, just click on it. The current value will be displayed in the "Value" window and an explanation of how the value was set will be displayed in the "How value was set" window. This can include the value coming from the end user, external sources, assignments or rules in the system.

When the value comes from a rule, the IF/THEN syntax of the rule is changed to SINCE/THEN since the rule will have had the IF conditions determined to be TRUE and will have fired. The SINCE/THEN syntax is easier to read. Also, even if the rule had multiple THEN assignments, only the one for the specific variable selected will be included.

The explanation of the how the value was set is the same as the .HOW property for the variable. (When running with TRACE, the data needed for .HOW is automatically maintained.)

The Variables tab makes it easy to examine the immediate value and status of any variable in the system.

Goal List

The Goal List is displayed on the left side of the window. This is used for backward chaining systems and shows the order of the Goal variables being used for the immediate backward chaining. The bottom goal is the actual variable needed in the top level system logic - typically set by a Command Block command such as DERIVE. To set the value for this Goal variable, the next highest variable in the list was needed. If that variable could be derived from other rules, the next variable above that was needed, etc. up to the top variable in the Goal List which is the actual one being asked or derived. The Goal List changes dynamically as rules fire and variable values are set.

In a backward chaining system, being able to examine the Goal List provides insight to how the rules are being used. If a variable is being asked unexpectedly, the Goal List may indicate why that variable is needed, and how it will be used. If a system is being run in forward chaining, or executing a command that does not require backward chaining (such as an ASK command), the Goal List may only have a single variable or be empty.

To check on the value and status of any variable in the Goal List, click on it. This will select it in the Variable list and display the current value and how it was set.

Rules Tab

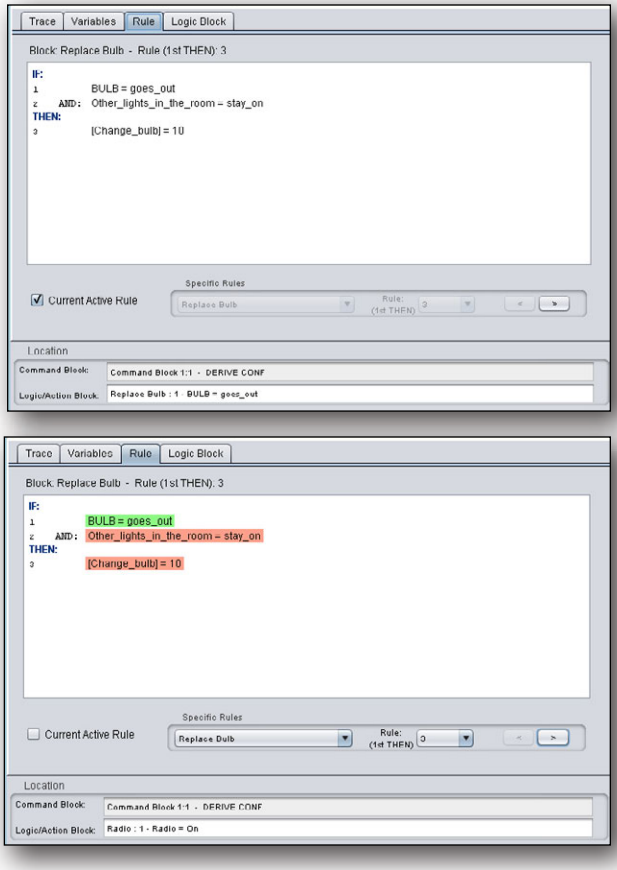
The Rules tab displays the current status of any rule in the system with the conditions color coded to indicate if the they are True, False or Unknown.

An IF condition that has not yet been determined to be either True or False will be displayed without a highlight color.

If an IF condition has been determined to be True, it will be highlighted in green. If it has been determined to be false, it will be highlighted in light red.

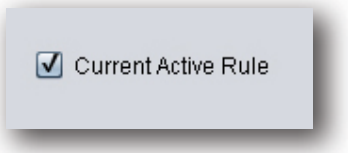
When the IF conditions for a rule are all True, the THEN conditions will be highlighted in green indicating that the assignments in those conditions have or will be made. Checking the “Logic Block” line at the bottom of the Trace Applet window will show if the system is currently making assignments from that rule’s THEN conditions. (Normally, unless the assignments require asking for user input, all of the THEN conditions highlighted in green will have been made.)

If the set of IF conditions are False, the THEN conditions will be highlighted in light red. These assignments will NOT be made since the IF part was False.

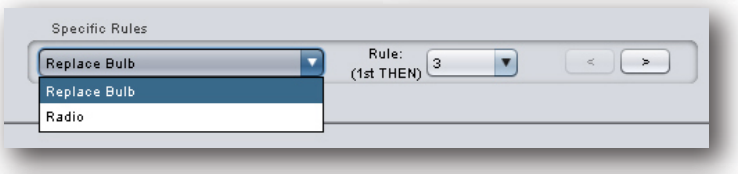


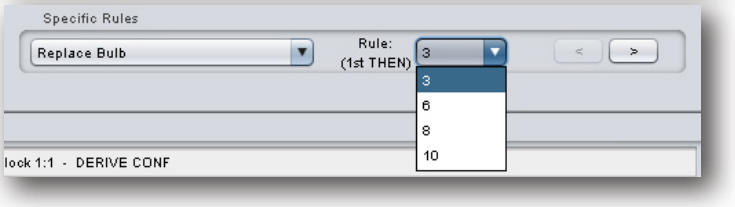
Current Active Rule

If the “Current Active Rule” checkbox is selected, the Rule tab will always display the rule that is currently being tested (IF condition) or used to make assignments (THEN condition). This rule will change dynamically as the system runs.



To examine the status of a specific rule, uncheck the “Current Active Rule” checkbox and the “Specific Rule” options will be enabled.





These allow selecting any rule in any Logic Block for examination. To select a rule, first select the Logic Block it comes from. Then select the rule in that Logic Block. The rules in a Logic Block are defined by the line number of the first THEN node for the rule. So if a Logic block has a rule where the first THEN node is on line 3 and another rule where the first THEN node is on line 6, the Rule drop down will display lines 3 and 6 for the rules.

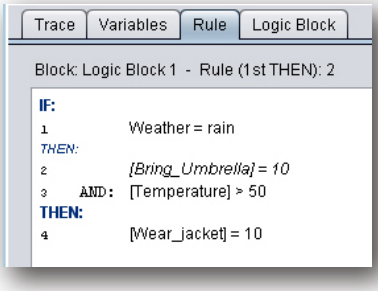
Once in a Logic Block, another way to quickly step through the rules in that block is to click the “<“ and “>” buttons which will move to the previous or next rule in the block. This allows quick examination of multiple rules in a system.

Once a specific rule is selected, the Rule tab display will only show that rule until another rule is chosen or the “Current Active Rule” checkbox is selected.

Selecting a specific rule allows watching that rule as the system runs to check its status during the session.

IF / THEN / IF Rules

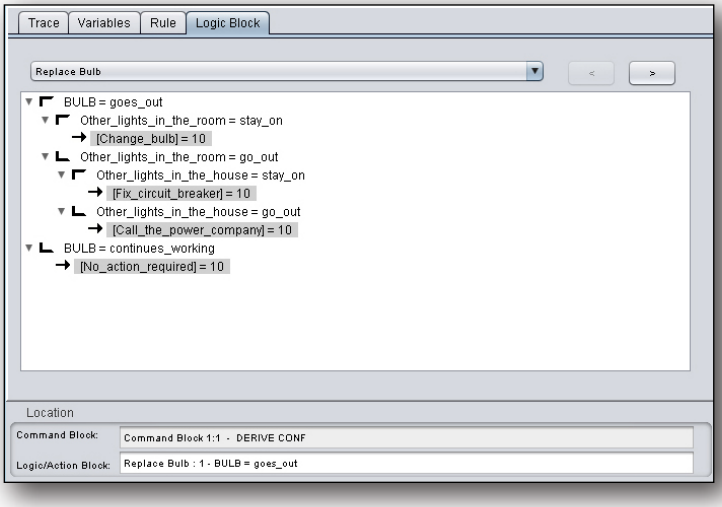
If a system uses IF/THEN/IF rules (IF nodes added under a THEN node), the rule display will reflect this. The Trace Applet “Rule” tab will display a rule for the first THEN node, and subsequent rules for the additional THEN nodes in the rule. The first THEN will be displayed in the standard way. The portions of the rule associated with the IF nodes following the first THEN node will include the first THEN node in italic and will a small “THEN” next to it.



Logic Block Tab

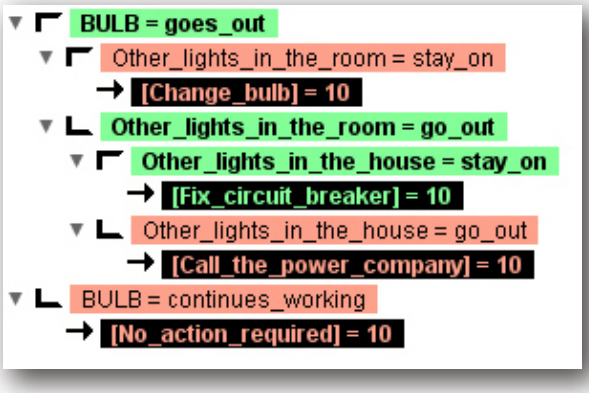
The most powerful way to examine the status of a system as it is running, is in the Logic Block tab. This displays the actual Logic Blocks in a system highlighted to indicate what nodes are True, False and Unknown.

Logic blocks can be selected from the drop down list or by clicking the “<“ and “>” buttons to select the previous / next Logic Block in the system.



Nodes are marked with the same bracket and arrow icons as are used in the Logic Block development environment. IF nodes that have not yet been determined to be True or False are not highlighted. THEN nodes that have not had the IF associated IF nodes determined to the True/False are highlighted with a light gray to make it easier to differentiate IF and THEN nodes.

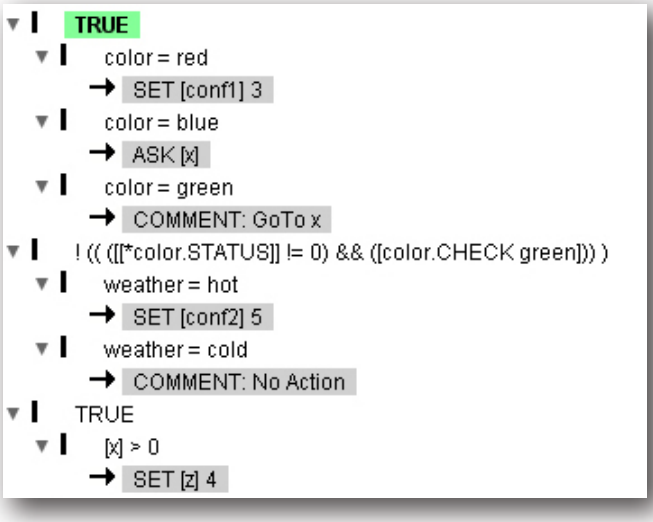
If nodes that have been determined to be true are highlighted in green and IF nodes that have been determined to be false are highlighted in light red. (Same as in the Rules tab.) THEN nodes that associated with IF nodes determined to be True are displayed in green on black and THEN nodes that associated with False IF nodes are displayed in light red on black. (The reverse text on black always indicates a THEN node).



Selecting a Logic Block during a run allows watching the status of the node change as the system executes. Details of individual sections can be examined in more detail in the Rule and Variable tabs.

Action Block Rules

If a system uses Action Blocks, at runtime these are converted to equivalent Logic Blocks that are processed by the runtime program. The Logic Blocks created are displayed in the Trace. While these are logically equivalent to the Action Blocks, they look quite different in Logic Block form. The individual lines are converted into separate rules in the Logic Block, and “Go To” statements are converted into fairly complex expressions that cause some of the rules to be skipped when a “Go To” command is executed.



It can be more difficult to follow the trace with Action Blocks since there is not an exact correlation with a system Logic Block. When tracing Action Block systems, the “Variables” tab is the most useful, though since the IF conditions correlate with the Action Block, it is still possible to use the “Rule” and “Logic Block” tabs too. Fortunately, Action Block logic is generally simple and straightforward, and Trace is generally not needed to follow the execution of Action Blocks.

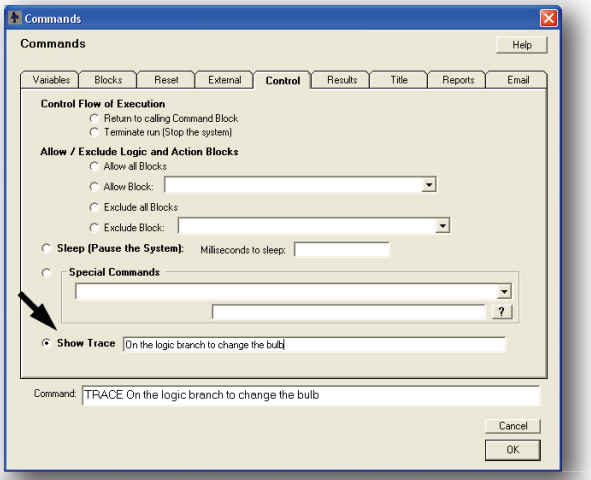
The TRACE command can be added to the “Action” portion of an Action Block to pause the system at that point and allow examination of the value of the variables.

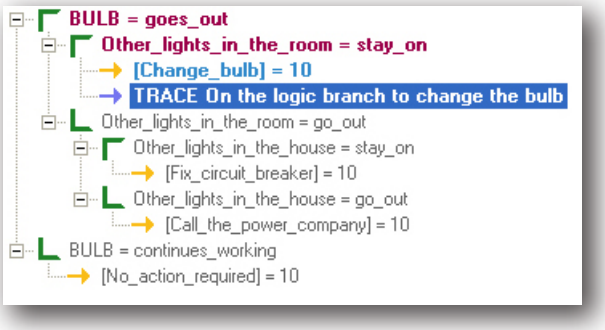
The TRACE Command

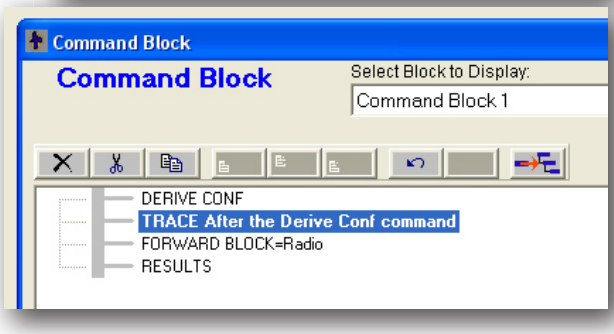
The Trace Applet allows examining the value of variables and the state of the rules and Logic Blocks in a system at a particular point during a run. However, the Trace Applet is constantly interacting with the Corvid Runtime Applet during a session receiving updates on the status. Because of this, the Trace Applet can only be used when the system pauses to either ask a question or display a results screen. Normally this is not a problem, since it is at these points that the developer needs most to check on the value of variables and rules. However, for some systems it is desirable to be able to pause the session so that the Trace Applet can be used to examine the current state.

This can be done with the TRACE command. The TRACE command is a standard Corvid command that can be added in the THEN part of rules or in a Command Block. It is built from the “Control” tab on the Corvid Command Builder. It allows a string to be added which will be displayed when the TRACE command is executed. This can be used to identify the specific TRACE command and indicate where in the system the session is.

The TRACE command can be added in the THEN part of rules (Logic Blocks), or in Command Blocks.







When the TRACE command is executed either due to the rule firing or the Command Block command being executed, Corvid will display “TRACE” followed by the string associated with the command and wait for the use to click the OK button. At this point the variables, rules and Logic Blocks can be examined with the Trace Applet.

There can be as many TRACE commands as needed in a system. The TRACE command has no meaning and is ignored if the system is run without the Trace Applet, so the TRACE commands do NOT need to be removed from a system when it is fielded - they will just be ignored.

Security Issues with TRACE

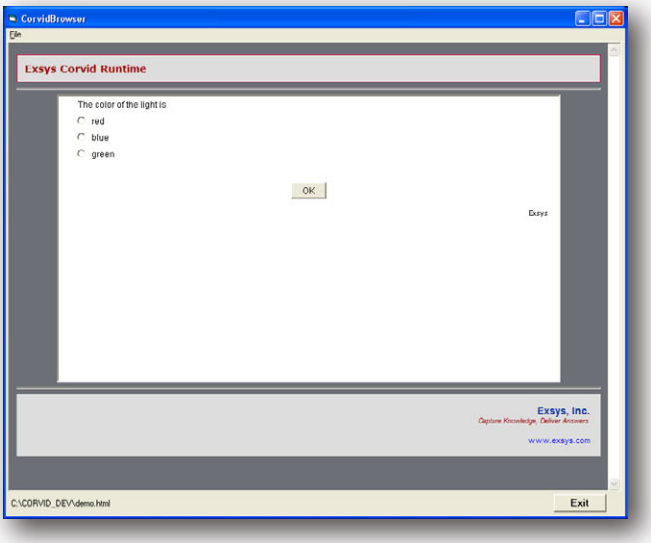
Since the new Trace Applet allows the user to examine the internal structure of a system it could reveal more information about the internal working, logic and structure of a system than some developers may want. To prevent this, the Trace Applet will ONLY work with a special CVR_TRACE file that Corvid generates when running in the Trace mode. This CVR_TRACE file contains special data and encoding required by the Trace Applet. The Trace Applet cannot run with the normal CVR file.

The CVR_TRACE file should be controlled the same as the CVD file. It should not be put on the server with the CVR file or made available to anyone that you would not want to have the system CVD file that allows editing. In practice, the CVR_TRACE does not allow editing the way the CVD file would, but it does allow examining the internal structure of the rules and Logic Blocks in a system.

Trace Applet Window Logic Block Refresh

The Trace Applet displays the Logic Blocks using a Java object. This updates the status of the nodes in the Logic Block quite rapidly and in some browsers, this can cause the text of the nodes to occasionally not be drawn correctly. This seems to be a Java / browser issue and has only been seen on Safari. If the Logic Block display appears garbled, click on the “Logic Block” tab to refresh it. This will redraw the display and should solve any display problems.

3. New Default Templates



Systems run with the Corvid Applet Runtime using the default templates now have a new appearance. The previous basic blue screen has been replaced. The system will be run in the white applet window with a new gray border and design.

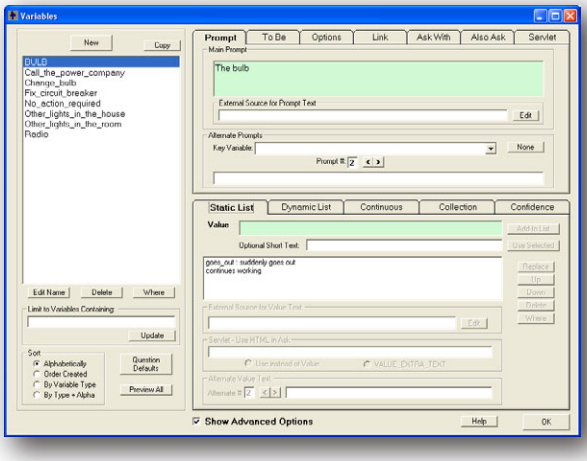
As before, this default look can be changed to any design that is preferred by specifying a system HTML page.

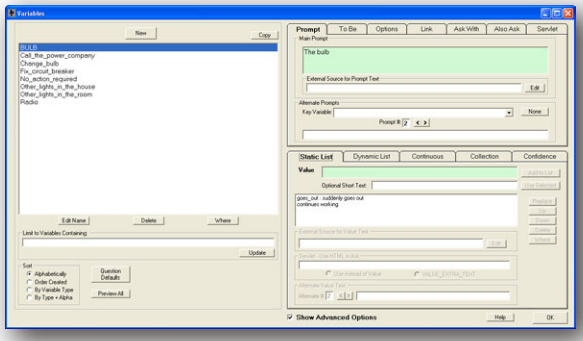
The default template can also be edited by modifying the “Corvid_Run_Template.html file installed in “Program Files/Exsys/Corvid”.

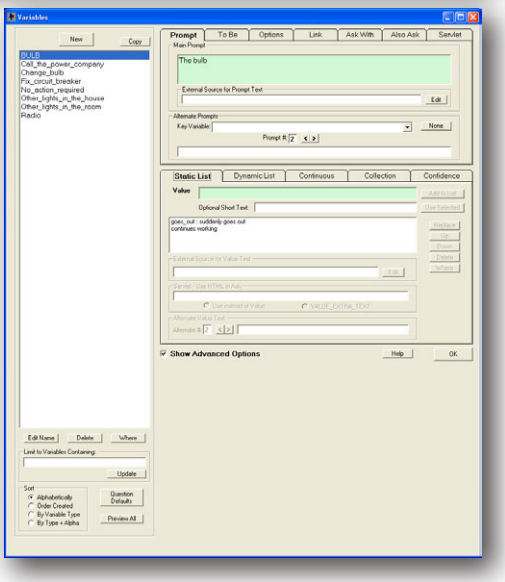
The default templates for the Corvid Servlet Runtime have also been modified to with similar look-and-feel.

4. Resizable Variables Window

Previously the Corvid “Variables” window could not be resized. This presented several problems. While it would fit and could be run on 1024x768 screen, it was a tight fit. Also, if a system used very long variable names these would be truncated and could be difficult to read. The Variables window now can be resized both to make it smaller to fit better on 1024x768 screens, and to allow it to better display long variable names.



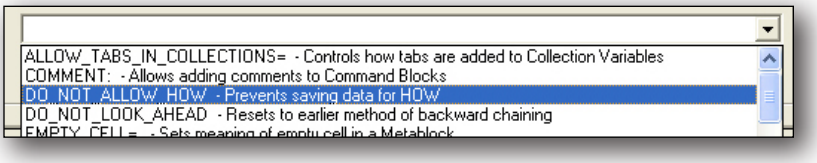




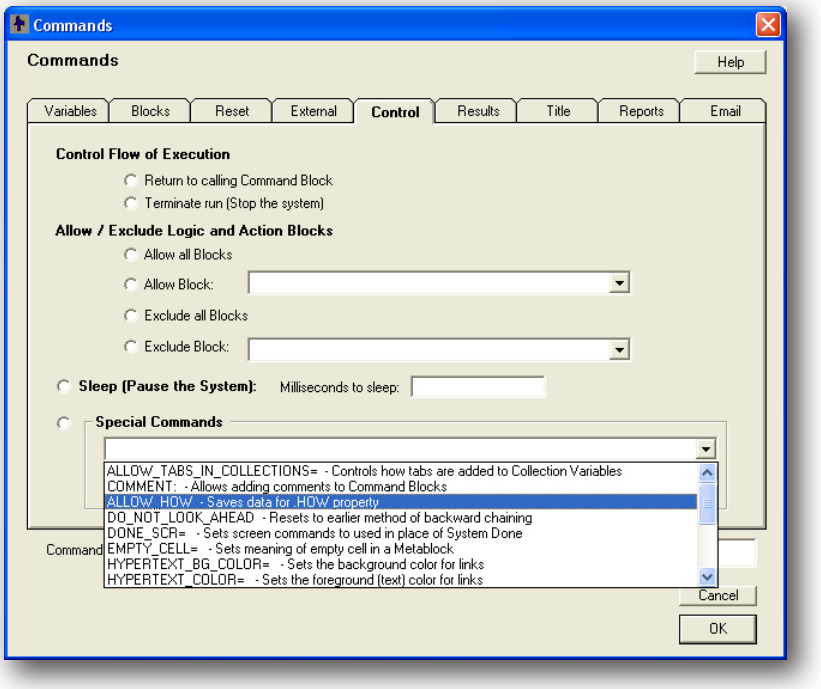
Users with larger screens can also now make the window higher so that more variables can be displayed in the list, or made wider so that variables with long names can be displayed without truncating.

5. Change in .HOW Defaults

The .HOW property displays the history of how the value for a variable was set. This can be displayed like any other variable, but is generally used for tracing how a system ran. The .HOW property takes up significant resources for large systems, especially MetaBlock systems, since it must keep track of any action that set the value for the variable. Because of this, earlier versions of Exsys Corvid had a command to disable the .HOW property and not take up the resources needed by it. This was done by setting the DO_NOT_ALLOW_HOW command in the "Special" commands.



With the new Trace Applet, the use of .HOW in a system is greatly reduced since the .HOW value can be examined for any variable in the Trace Applet "Variables" tab. Because of this the default for .HOW has now been changed to disabled. If a system needs the .HOW property it MUST add the new command ALLOW_HOW in the "Special Commands" section. When running with the Trace Applet the ALLOW_HOW is automatically set and does not need to be added.



This new default will reduce system resources for the vast majority of systems that do not need or use the .HOW property, but did not have it explicitly disabled.

6. New Java Requirements and New Java Compiler

The new ExsysCorvid.jar file with Corvid v5.3 requires Java v1.5. This is higher than the previous Java requirement of v1.2. However, Java v1.5 was released in 2004 and most browsers now automatically prompt users to update their Java version. Since virtually all users will have installed a new operating system or browser since 2004, it is unlikely that they will not have Java 1.5 or higher and there should be no problem running the new Corvid Applet Runtime.

We have also moved all our Java development from JBuilder to Oracle's NetBeans tools. This provides the most complete and definitive Java environment. There should be no change in functionality due to this switch, and many systems have been tested to make sure they function exactly the same. However, as with any such change, you should test your systems to make sure that they perform and appear exactly the same. There may be slight differences in the layout managers between JBuilder and NetBeans. While we did not find any difference in test system appearance, there is always a possibility that a system's control layout will be right on a threshold that will appear different. Please test systems before fielding them with the v5.3 ExsysCorvid.jar. If any differences are found, please let us know.

7. What's Next

The manual rewrite was a major project that took considerable time and resources. A simple review and edit became a full rewrite. Currently we are working on an Apple Mac expert system development tool. This will have a somewhat different look-and-feel from Corvid due to the nature of the underlying Mac OSX operating system. It will only implement a subset of the Corvid feature set, leaving out many advanced functions, but will be an extremely easy to use tool. Systems built on the Mac tool will be upwardly compatible with Exsys Corvid.

On the Windows side, the next version of Corvid will make it much easier to build systems delivered with the Corvid Servlet Runtime. It will be easy to build, test and design systems for Servlet delivery. Corvid Servlet Runtime systems have many advantages for fielded systems since screens are designed with HTML, CSS, JavaScript, etc. Since they only send HTML forms to the end user, the end user's machine does not require Java support - allowing systems to run on iPhones, iPads and other mobile devices. It also eliminates the Java requirement for PC browsers, particularly Internet Explorer, that provide poor support for Java applets. Servlets also provide better system security and easier integration with server resources.

Corvid already allows building systems for Servlet Runtime delivery, but it is not as easy as building systems for Applet Runtime delivery. We will be changing that to make it easier to take advantage of all the features of Servlets, will the same ease as applets.

Look for both of these early next year.

Exsys, Inc.
6301 Indian School Rd. NE, Suite 700
Albuquerque, NM 87110 U.S.A.
Tel: (505) 888-9494, Fax: (505) 888-9509

info@exsys.com
www.exsys.com

For support questions contact: support@exsys.com