6 Automatic Testing

About this Section

In this section you will learn about CableEye's Automatic Testing Capabilities. Separated in two main subsections, Macros is intended for simple automation applications, but yet powerful enough for most of them, and Javascript, intended for complex situations, where more control is needed.

You will learn how to create your own programs, edit and customize working flows to automate your electrical testing requirements.

You will also learn how to use accessories like our CB35 Relay Board, to control external devices and integrate CableEye to a bigger complete test station.

Follow this index to jump quickly to a topic of interest:

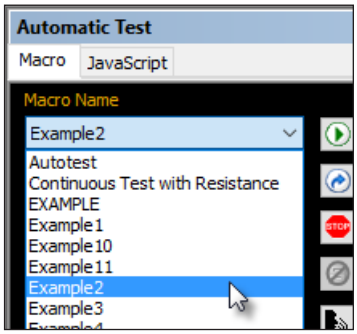
6.1 Macros

Production testing requires only that a PASS/FAIL result be obtained. You may set up CableEye so that the operator needs only to press the TEST pushbutton on the tester and check the PASS and FAIL LEDs. Using the mouse or keyboard, or reading the screen, becomes unnecessary. A Macro lets you do this by performing a sequence of steps automatically. Each step represents a basic operation that you could perform manually, if desired, such as "Test Cable", or "Print Difference List". Other operations valid only in a Macro let you issue warning tones, create loops, or post messages for the operator to see.

The following is a quick description of how macros work. More details about creating macros start at page 6-8, with several examples to follow. In the meantime before getting to the examples, you can have your software open and try to familiarize with the different screens and buttons.

The test engineer would initially create a Macro to achieve the desired operation. To begin testing, the operator loads the Macro and presses the TEST button as shown in the following steps:

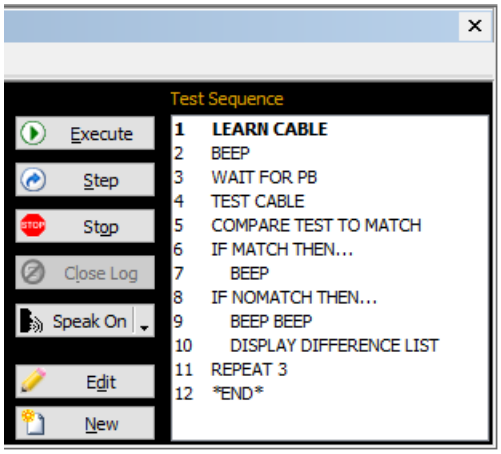
Click the **AUTO** button to open the "Automatic Test" window. Then choose the Macro to load from the menu. You may create any number of Macros for different jobs.



You will find several pre-saved macro examples in your CableEye software, which will be explained in detail in this section.

The Macro specifies the test operations you need for a particular job. Easily create Macros by just choosing commands from a menu (described later). You may modify the Macro at any time with the "Edit" function, and save the final Macro on disk for future use.

Press the TEST pushbutton on the tester, or click the "Execute" button on the screen, to begin testing. Results are shown on bright LED lamps.

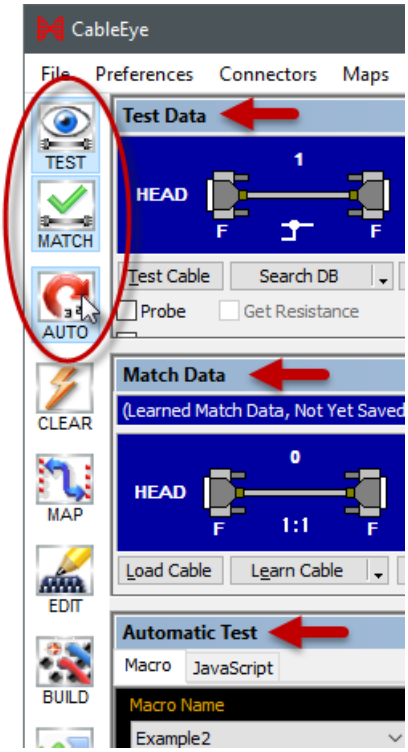


If the cable matches, a large green box with a yellow checkmark is shown. If the cable fails, the macro Displays the Difference List as shown on line 10 and a big red box with a black X is displayed as well.

Generally the macro will Repeat automatically to test the next cable, unless you only want to test one particular and specific cable or perform another function only once.

Macros can perform many functions and you can have as many lines as you need in your macro script to accomplish them. All macro instructions will be covered in this section and Section 7 - Data Logging.

6.1.1 The Macro Window



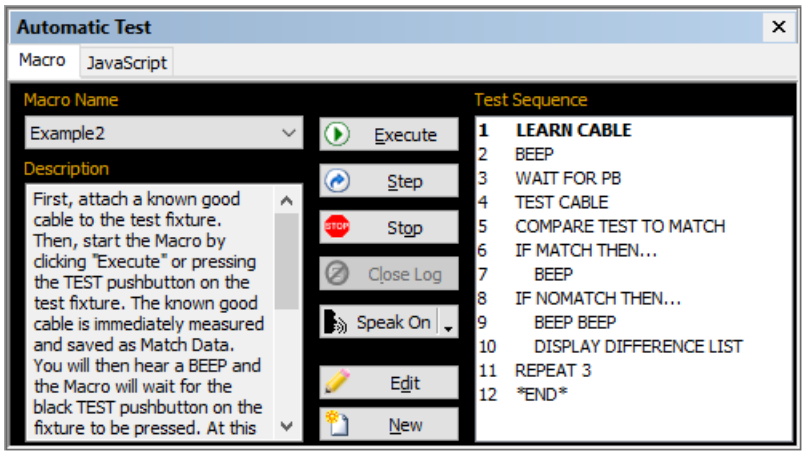
A special window provides all the controls you need to create, edit, and execute Macros. Click the **AUTO** button to turn this window on, and click it again or click the Close box to turn it off.

When the *Automatic Test* window appears, it will share the entire screen with either the Test Data window, Match Data window or both, depending on which windows were open when you clicked the AUTO button. In the example in the right, the 3 windows are displayed. Normally these windows have the same size, but you can re-size them by dragging the edges of the window.

It is a good idea to hide the Test Data and Match Data windows (by clicking TEST and MATCH) to have more space when creating your macro. Although you can scroll down your script if it has more lines that it can fit on the screen, sometimes you might prefer to have the extra space to display most of your program at once.

The Automatic Test Window displays three tabs, Macro, which is being explained in this section 6.1, JavaScript explained on Section 6.2 and Macro Preferences.

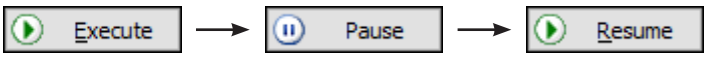
The Automatic Test window appears as seen on the right when open. The text box under "Macro Name" shows the currently selected Macro; click the Down-Arrow Tab to see the entire selection of available Macros and choose the one you need.



The Description field is an editable text block. Add comments freely. Any text you enter here is saved automatically with the Macro for future reference.

Execute – Click the "Execute" button, or press the TEST pushbutton on the tester to start the Macro you see shown in the Test Sequence text block. The instructions execute automatically, one after the other, until an specific instruction that stops or pauses the macro is reached. The WAIT FOR PB ("Wait for Pushbutton") instruction for example, pauses the macro until you press the TEST pushbutton on the tester or click the "Resume" button (read below) on the screen, allowing time to disconnect one cable and connect the next.

The "Execute" button changes to "Pause" after you begin. Click "Pause" to temporarily



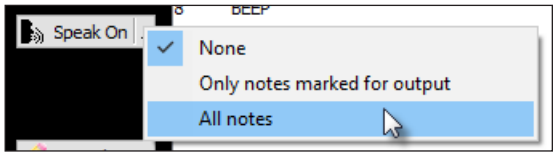
stop the Macro. Once paused, the button changes to "Resume" which you may click when ready to continue testing. *During a pause, you may manually test an individual cable, load data from the database, or print a report.* Clicking "Resume" continues the execution where you left off rather than starting over from the beginning. Note that if you turn off the Macro mode during a Pause, the next instruction address and loop counter will be cleared.

Step – Click the "Step" button instead of "Execute" to process instructions one at a time. When one instruction finishes, the software always pauses and will wait until you press "Step" again before continuing. In this way, you may study the effect each instruction has before going further, an especially useful tool when debugging a new Macro. Once you are convinced that your Macro performs as intended, click "Execute" to continue at full speed.

Stop - Click the "Stop" button to end the execution of a macro. This will clear your loop counter and return to the first instruction.

Close Log – Use the "Close Log" button to manually close a log file that had been opened within a Macro. To learn more about Log Files, read Section 7 - Data Logging.

Speak On – Several macro instructions display text for the user to read. This text can also be spoken by the software through your computer speakers. Clicking this button allows you to select which notes will be spoken, if any.



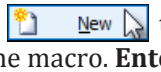
Edit - Click the "Edit" button to edit an existing macro. Choose the macro that you want to edit from the *Macro Name* list, and then click the *Edit* button. When you are finished editing and save your changes, the altered Macro will replace the original one.

New - Click the "New" button to open the edit window with a completely blank slate. From this screen, you may create a new Macro of your choice. When finished editing, click "Save" to write the new file to disk.

6.1.2 Creating and Editing a Macro

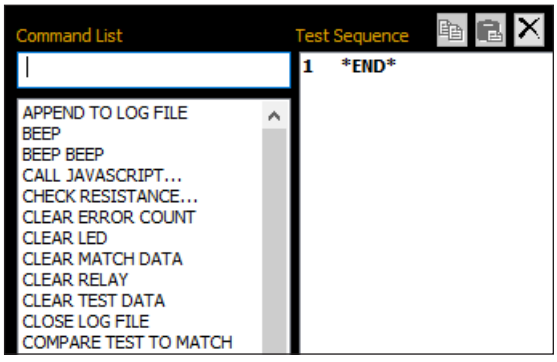
Several example Macros come with the CableEye software to illustrate some of the ways you can employ automatic testing. These examples are ready to use and cover many typical situations, however you can edit these examples or create your own if you have to do something differently than we have done with these examples. This section explains how to create and edit a Macro. If possible, it would be helpful if you had CableEye operating while you read this so you could work through the examples as we discuss them.

The "Edit" and "New" buttons located at the bottom of the Macro window activate the Macro editing function. They remain dimmed during Macro execution and become active only when execution stops.

1. **Click** the **New** button 

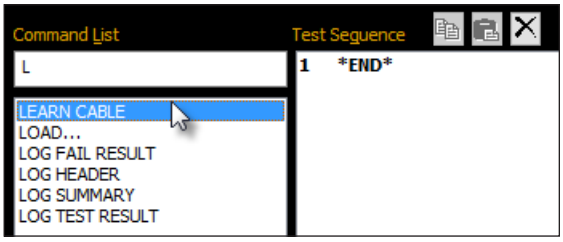
to start entering a new Macro. A pop up window will appear asking you for the name of the macro. **Enter** any desired **name** and **click OK** when ready.

You see a list of permissible Macro commands in alphabetical order on the left and your actual Macro sequence on the right. You will select instructions from the command list one at a time to assemble the Macro sequence you need.

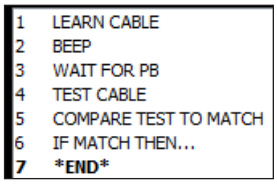


Scroll down the command list to review the instructions. Note that most of the operations you've previously used when testing individual cables, like "Test Cable", appear in the list. You will also see special commands useful only in a Macro, like "If Match Then . . ."

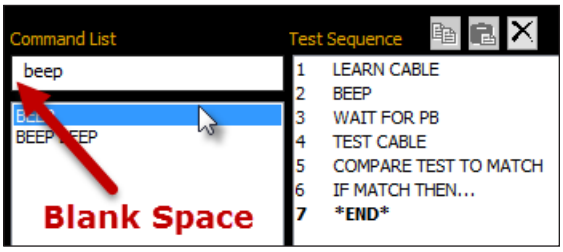
2. **Click in the text entry box** under **Command List**. Then, **type** the letter "**L**" to see all commands that begin with that letter. **Double-click on LEARN CABLE** to add it to the test sequence. Note that you may either scroll to a command or continue typing its name until no other selections are shown.



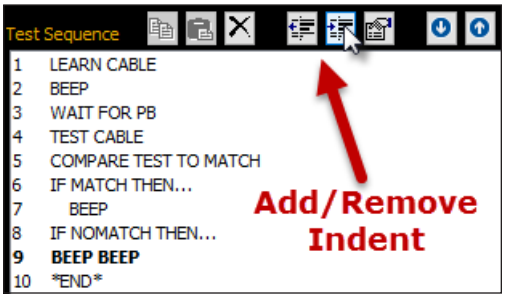
3. Continue selecting Macro commands and adding them to the list as shown in the right. You should have six macro instructions plus END as shown at the right.



4. The next instruction we need is BEEP, shown selected on the right, but not yet entered in the list. Because we want to hear a tone (beep) only if the cable passes, we must make the BEEP instruction dependent on the IF MATCH THEN... being true. To do this, you must indent the BEEP instruction in the macro. To indent an instruction, **type a space in the command list before selecting it!** as shown on the right. Now, **Double click on BEEP** and it will become an indented instruction, executing only when the IF MATCH THEN... statement is true.

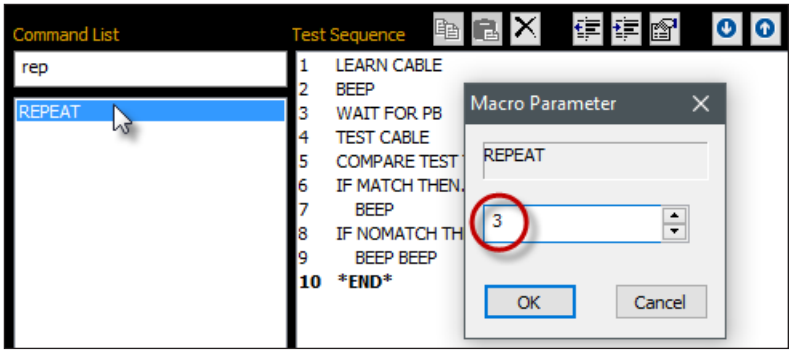


5. Enter additional instructions so the Macro appears as shown on the right. Here, we have made an intentional mistake by not indenting the BEEP BEEP instruction. To correct this, **click on the BEEP BEEP instruction** to make it bold. Now you may **click the Add Indent button** shown on the right. You should now see a blank space at the beginning of the BEEP BEEP instruction. You can also remove an Indent by clicking the Remove Indent button.



6. The last instruction in this Macro creates a loop to line 3. This will bring us back to the WAIT instruction and allows the test sequence to repeat executing after you load the next cable.

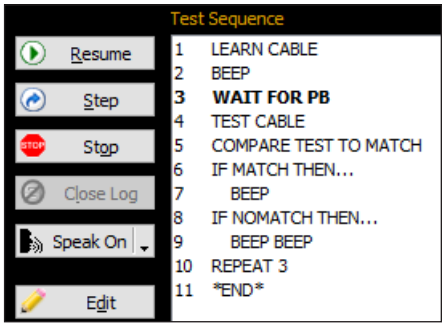
   Insert the **REPEAT** instruction in the macro. A small window appears, at which time you should **type in the line number to which you will jump**.



You may jump forward instead of backward by entering a line number greater than that of the REPEAT instruction. If the destination instruction has not yet been entered, you may need to enter a temporary value and go back to change it later.

Save your work **by clicking the "Save" button**. If you pressed "Cancel", all changes you made would be discarded and nothing saved to disk. Click the "Print" button, just to the left of "Save", to print hard copy of this Macro on your report printer.

7. To test the Macro, first **connect a cable** of your choice to the tester. Then **click Execute** to begin. When execution starts, the message bar at the bottom of the screen will turn red and show the current Test Count and Error Count. You will also note that the instruction currently executing will appear bold.

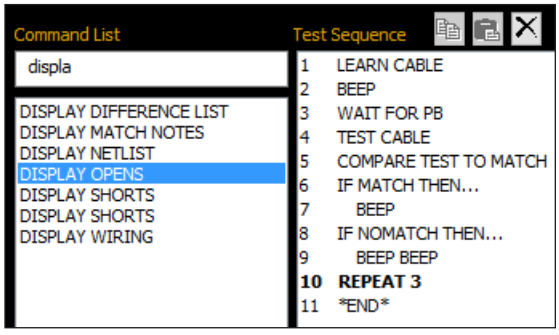


In this Macro, CableEye will learn the attached cable as "good", sound a tone, and then wait for you to press the TEST pushbutton on the tester. When ready, press the button to run through a typical test cycle. To force an error, loosen the cable slightly at one end to create a few open circuits, and press TEST again. Note that a different tone sounds for the error condition. Click "Stop" to stop execution.

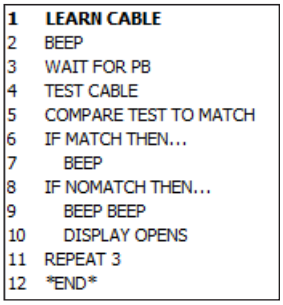
Important: The Test Count value normally starts at "1" and increments automatically when the REPEAT instruction executes.

8. **Stop the Macro**, and start again, but this time **click Step** instead of Execute. In this case, each instruction waits for you to click "Step" again before executing. You will find this very helpful when debuging Macros.

9. **To add a new instruction** to the Macro, **click Edit** to return to the Macro edit window. Let's display the open circuits if an error occurs. To do this, click on the instruction just below where you want to insert the new instruction. The instruction selected will be highlighted in bold. In this case, **select REPEAT 3**, because we want to insert the new instruction above it. Then, from the Command List box, **type a space** to create an indent, type "display opens" and **double click DISPLAY OPENS.**



10 – Save the edited Macro. The final result should appear as shown at the right.

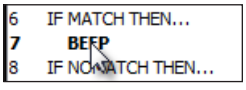


11 – Run the modified Macro and check it by learning the cable you used before, and then loosening it in the tester to cause an open circuit. This time, when you test it, you should see the Open Circuits displayed in the Test Data Window.

6.1.3 Macro Editing Summary

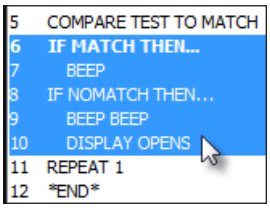
The following paragraphs summarize important facts about Macro editing:

Insertion Point: The bold instruction in the list indicates the insertion point. New instructions will be inserted above this. Change the insertion point by clicking on a different instruction while in the Edit mode.



Deleting Instructions: Click on the instruction you wish to delete and press the DELETE key on the keyboard while in the Edit mode. You can also click the button in the toolbar.

Inserting Instructions: Place the insertion point and select the desired instruction from the list while in the Edit mode.



Copy and Paste Macro Instructions: You can copy and paste macro instructions by making a selection holding the SHIFT key on the keyboard and clicking on the instructions that you want to copy. Once you have your selection, press the copy button to copy the instructions to the clipboard. Finally, select the insertion point and click the paste button.

Conditional Instructions: Indented instructions are conditional and will be executed only if the previous "IF"-type instruction is true. Create an indented instruction by first typing a space in the Command List's text box, and then choosing the instruction while in the Edit mode. You may have as many indented instructions as you wish following an IF statement. You can also click the Add Indent and Remove Indent buttons if the instructions are already inserted in the list.

6.1.4 Macro Instructions and Examples

Eighty-four Macro instructions appear in the command list at the time of this writing. We typically add new instructions as the software advances, so you may find more than this number in the software you receive. The command list you see in Edit mode presents these instructions in alphabetical order, although all are easily accessed by typing the first few letters of the instruction, and then scrolling into the shortened list to pick what you need.

Reference List of Macro Instructions showing the page where the instruction is explained:

In this section, we explain the Macro instructions based on examples. Each example will introduce one or more Macro instructions (highlighted in blue in the script) that will be explained after the example. These examples are installed in your software for you to try them.

EXAMPLE 1: BASIC TEST - Before starting you should mount a cable on the tester!

```
1   LEARN CABLE
2   BEEP
3   WAIT FOR PB
4   TEST CABLE
5   COMPARE TEST TO MATCH
6   IF MATCH THEN...
7      BEEP
8   IF NO MATCH THEN...
9      BEEP BEEP
10     DISPLAY DIFFERENCE LIST
11  REPEAT 3
```

New instructions used in this example:

LEARN CABLE – measures a cable and copies data into the Match Data buffer. Operates exactly like the "Learn Cable" button.

BEEP - sounds Window's Asterisk tone through the computer speakers. Use this to announce successful completion of a test, or to cue the Operator to mount another cable.

WAIT FOR PB (wait for pushbutton) – pauses the macro indefinitely until you press the TEST pushbutton on the tester, press the optional footswitch, or click the "Resume" button on the screen. You can use this instruction to allow time for the operator to remove a cable that has been tested and mount a new one before allowing execution to continue.

TEST CABLE – measures a cable and copies data into the Test Data buffer. Operates exactly like the "Test Cable" button.

COMPARE TEST TO MATCH – compares Test Data to Match Data. If Test and Match Data are exactly equal in wiring, connector type, and gender, the match condition is set (meaning that there are no faults). Anything other than exact equality sets the condition to nomatch.

IF MATCH THEN . . . – executes indented instructions that follow, only if the match condition is true. Should the match condition be false, all the following indented instructions will be skipped, and execution will resume in the next unindented instruction.

IF NOMATCH THEN . . . – executes indented commands that follow, only if the match condition is false. This command functions just like IF MATCH THEN . . . , except that the match condition is nomatch.

BEEP BEEP – sounds Window's critical stop tone through the sound card on your computer. If you do not have a sound card installed, you should hear two successive beep tones on the computer's built-in speaker. Use this to warn that an error or mismatch has occurred.

DISPLAY DIFFERENCE LIST – turns on the Differences List window and immediately continues to the next instruction. There are a total of six self-explained Display instructions, that you can insert in your script. The additional five are **DISPLAY MATCH NOTES, DISPLAY NETLIST, DISPLAY OPENS, DISPLAY SHORTS** and **DISPLAY WIRING.**

REPEAT – jumps execution to the specified line. In most cases, the specified line is higher in the list than the current line, causing the execution to go back to a previous instruction. However, you may jump forward if desired by inserting a line number that is greater than the current line. You would enter the value of the line at the time you create the Macro and this value would then be shown embedded in the instruction (for example, REPEAT 3). The REPEAT instruction automatically increments the Test counter, regardless of whether you jump backward or forward. Use **SKIP CABLE COUNT** prior to REPEAT whenever you don't want the Test Count to be increased.

Explanation of Example 1: (Line 1) Learn match data from the model cable and store it in the Match Data buffer. (2) Sound a tone to acknowledge completion. (3) While the system waits for the pushbutton to be pressed, remove the model cable and mount a test cable. Then, press the pushbutton to proceed. (4) Acquire Test Data from the cable under test. (5) Compare Test Data and Match Data. (6,7) If they match, sound a single tone. (8,9,10) If they do not match, sound a double tone and display the wiring differences. (11) Jump back to line 3 to repeat the process.

EXAMPLE 2: CONTINUOUS TEST - Before starting, manually save a cable to be loaded by the macro!

```
1   LOAD...MY CABLE
2   WAIT FOR PB
3   CONTINUOUS TEST
4   IF NO MATCH THEN...
5     BEEP BEEP
6     DISPLAY DIFFERENCE LIST
7   REPEAT 2
```

New instructions used in this example:
LOAD . . . – loads a specific cable file from the database into the Match Data buffer during Macro execution. When you add this instruction to the macro during editing, a selection list appears showing all available cable files in the database. From this list, you must choose one. The cable file name you chose is embedded in the instruction (LOAD... MY CABLE). When executed, this instruction always loads the specified file.

CONTINUOUS TEST – measures a cable continuously while looking for changes in continuity. When the instruction is inserted, choose to compare against Match, First Pass or Default. When this executes during a Macro, you should press the TEST pushbutton when ready to stop the measurement and advance to the next instruction. CONTINUOUS TEST sets the match condition like TEST CABLE does, so you should follow with IF MATCH THEN... or IF NOMATCH THEN... for conditional operations.

Explanation of Example 2: (1) After the Macro starts, mount the cable that you loaded in line 1. (2) Press TEST button to start the procedure. (3) the cable is tested continuously until you press the pushbutton again. While the test is running, flex the cable at various points to check for intermittent connections. When you flex the cable, be sure to hold the connector nearest the flex point firmly down so that it does not work loose from the tester. If changes in the continuity occur during flexing, short beeps will sound and the red "Fail" LED will turn on. (4) When you press the pushbutton, the macro checks if the cable doesn't match. (5,6) If the cables don't match, it sounds a double tone and displays the difference list. If no intermittent connections were found, the green "Pass" LED will turn on. (7) Jump back to line 2 to repeat the process. At this time, you may remove the cable and mount another.

EXAMPLE 3: SEARCH DATA BASE AND PRINT - Before starting set your printer accordingly!

```
1   WAIT FOR PB
2   TEST CABLE
3   SEARCH DISK FOR MATCH
4   IF MATCH THEN...
5     PRINT MATCH DATA
6   STOP
```

New instructions used in this example:
SEARCH DISK FOR MATCH – searches the database for an identical match based on the contents of the Test Data buffer. Connector types, genders, and wiring must all match. If the software finds an identical match, it loads that cable file into the Match Data buffer, sets the condition to match and turns on the PASS LED. Anything other than an identical match will leave the Match Data buffer unchanged. You may follow SEARCH DISK FOR MATCH with IF MATCH THEN... or IF NOMATCH THEN... for conditional operations.

Note 1: If multiple matches are found, a pop-up window will appear asking you to choose which one to load. Although the wiring will be the same for both, the Descriptive Notes and Label Text may be different, and this would affect printed output should the Macro later print reports or labels.

Note 2: Caution! If a defective cable, because of its defect, matches another cable in the database, SEARCH DISK FOR MATCH will cause that incorrect cable to load and no error will be asserted.

Note 3: If no Test Data is present when SEARCH DISK FOR MATCH executes, an error message will appear and the Macro will be stopped; this would represent a Macro design error.

PRINT MATCH DATA – prints the currently loaded Match Data Report, including the title block with the wiring schematic, wire list, descriptive notes, and label text. This functions exactly like clicking the print button. To learn more about printing preferences and customizing your reports, check section 5.

STOP – terminates a Macro and returns control to the Macro menu.

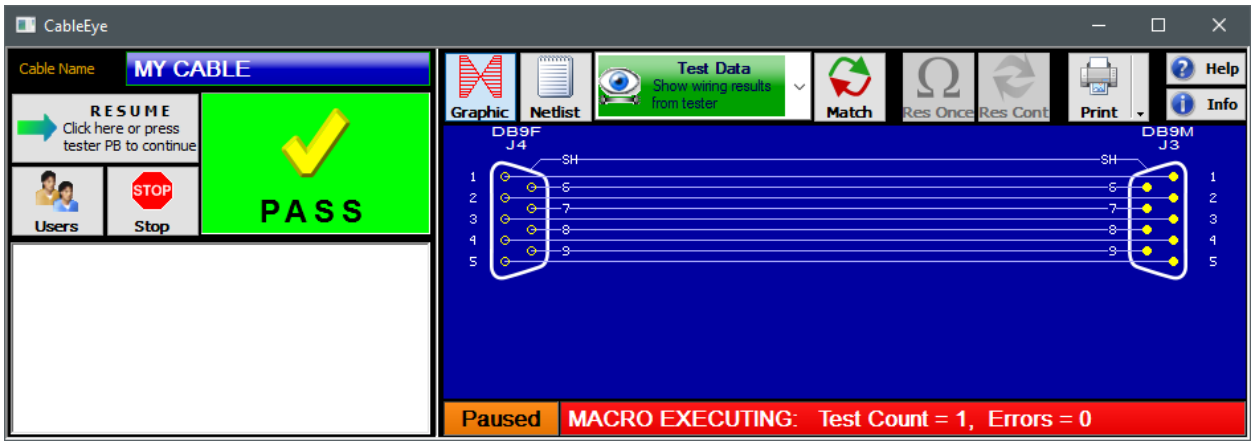
Explanation of Example 3: This macro is used to search for a cable in the database that matches the tested cable. (1) Waits for you to push the Test button. Before pressing the button, you should connect the cable to the tester. Press the button. (2) The cable will be tested. (3) SEARCH DISK FOR MATCH will grad the tested data and search for a cable in the database that matches it. (4) If an identical cable is found, the pass condition is set to match, (5) and the Match Data report is printed. (6) If no match for the cable is found, the program stops.

The are several reports and labels to print with macros, as shown with the following instructions:

PRINT DIFFERENCE LIST - Prints a report showing the differences between match and test data.
PRINT INTERMITTENT CX - Prints a report showing the intermittent connections found.
PRINT LOG FILE - Print a Log or Batch report, explained in section 7.
PRINT MATCH DATA - Used above, it prints the match data report.
PRINT MATCH DATA LABEL - Prints the contents of the match data label. Read section 5 for more info.
PRINT TEST DATA - Very similar to the match report, this prints the Test Data Report. This is the most common report print. It contains the PASS or FAIL result in the header, as well as the graphic wiring and netlist.
PRINT TEST DATA LABEL - This instruction prints the test data label.

Simplified Screen

CableEye is a powerful program with a main screen full of menus, fields and buttons. Sometimes you might want access to a simplified screen for production testing, to show only basic buttons and functions to the operator. This is possible using the SHOW PANEL macro instruction, which on execution, brings to the front the simplified screen shown below, and hides the main screen completely.



The simplified screen has basic buttons to Stop the macro, change User, change between Graphic and Netlist, choose between Test Data or Differences, measure Resistance or Print. Also, a big indicator will give you the Pass or Fail result of the test. The usual status indicator is shown at the bottom, displaying the Test Count and Errors.

The simplified screen is particularly useful when using a touch screen monitor, which allows the operator to easily touch on the big buttons that it provides. Below is an example showing the use of SHOW PANEL.

EXAMPLE 4: SIMPLIFIED SCREEN AND EXPORT TO PDF

```
1   SHOW PANEL 1
2   LOAD...MY CABLE
3   WAIT FOR PB
4   TEST CABLE
5   COMPARE TEST TO MATCH
6   EXPORT TEST TO PDF...<CABL
7   REPEAT 3
```

New instructions used in this example:

SHOW PANEL - Changes the macro execution to the simplified screen view. There are 2 types of simplified screens. Choose 1 or 2 depending on which want you want to display.

EXPORT TEST TO PDF - Very similar to the Print instructions, this one generates a PDF file for the report. The file is then saved in the default location, *Report Output* in the CableEye folder.

When you insert any "Export to PDF" you need to define the name of the file in advance. In this example, we use the variables <cablename> and <count> separated by "-". This will ensure that every report has a unique name, consisting of the name of the cable being tested and the sequence number. The file will be named "MY CABLE-120.pdf" for example. Learn more about variables in section 6.1.6.

As with printing, there are several types of reports that you can Export to PDF:

EXPORT DIFFERENCES TO PDF - Exports the differences between match and test data.
EXPORT INTERMITTENTS TO PDF - Exports a report showing the intermittent connections found.
EXPORT LOG TO PDF - Exports the log report, explained in section 7.
EXPORT MATCH TO PDF - Exports the match data report to PDF.
EXPORT TEST TO PDF - Exports the Test Data Report to PDF.

EXAMPLE 5: PRINT BATCH OF LABELS - Before starting set your label printer accordingly!

```
1   ENTER CABLE NAME
2   PRINT MATCH DATA LABEL
3   IF COUNT = 8
4      STOP
5   REPEAT 2
6   STOP
```

New instructions used in this example:
ENTER CABLE NAME – displays a window when executed asking the operator to enter a cable name. When the operator chooses a file, the selected cable file is loaded into the Match Data buffer. Unlike LOAD..., no specific file is loaded with this instruction. Rather, it is a "run-time" load in that the operator is asked to select the desired file at the time of execution, making this macro more universal to be used with as many different models if they require the same automatic test logic.

IF COUNT = – executes indented commands that follow when the Test Counter equals the number specified. You would enter the value at the time you create the Macro, and this value would then be shown embedded in the instruction (for example, IF COUNT = 8). The value may be any numeric value from 1 through 9999. Note that at the beginning of the macro, the count starts with 1 and not 0.

Explanation of Example 5: (1) The macro will ask you to choose the file to load. Once selected, (2) it will print the match data label saved with that cable. At this point the Test Count is equal to 1 (COUNT = 1). (3) Because COUNT is not equal to 8 yet, the STOP instruction is not executed, (5) and the macro is repeated back to line 2, where it prints another label and Test Count is increased to 2, and so on. Once the Test Count equals 8, the STOP instruction will be executed (4) and the macro will be terminated.

EXAMPLE 6: COMBINED SIMPLE AND CONTINUOUS TESTS

```
1   LEARN CABLE
2   WAIT FOR PB
3   TEST CABLE
4   COMPARE TEST TO MATCH
5   IF MATCH THEN...
6      BEEP
7   IF NOMATCH THEN...
8      BEEP BEEP
9      DISPLAY DIFFERENCE LIST
10  PAUSE 2
11  CONTINUOUS TEST
12  IF MATCH THEN...
13     BEEP
14  IF NOMATCH THEN...
15     BEEP BEEP
16     PRINT INTERMITTENT CX
17  REPEAT 2
```

New instructions used in this example:
PAUSE – inserts a pause of "n" seconds in the script without having any other effect. You may wish to use PAUSE in place of WAIT FOR PB to insert a fixed delay while a test cable is being changed. You would enter the number of seconds at the time you create the Macro, and this value would then be shown embedded in the instruction (for example, PAUSE 2). The value may be any number in seconds from 1 through 9999.

Explanation of Example 6: Like example 1, a master cable is learned (1), the script waits for the Test Button to be pressed (2) while you disconnect the master cable and connect a test cable. Press the button and (3,4) the cable is tested and compared. (5) If the cables match (6) a sound tone is emitted, but (7) if the cables don't match, (8,9) a double tone is emitted and the differences are displayed.

(10) A two second pause will be taken before (11) performing a Continuous test. (12,13) If the cables match another beep will sound, but (14,15,16) if the cables don't match after the continuous test, a double beep will sound and the intermittent report will be printed. Finally (17) the script will be repeated from line 2.

This macro showed you how to perform multiple tests to the same cable if needed. You can have as many different tests in the same macro before actually repeating the cycle.

EXAMPLE 7: SKIPPING CABLE COUNT

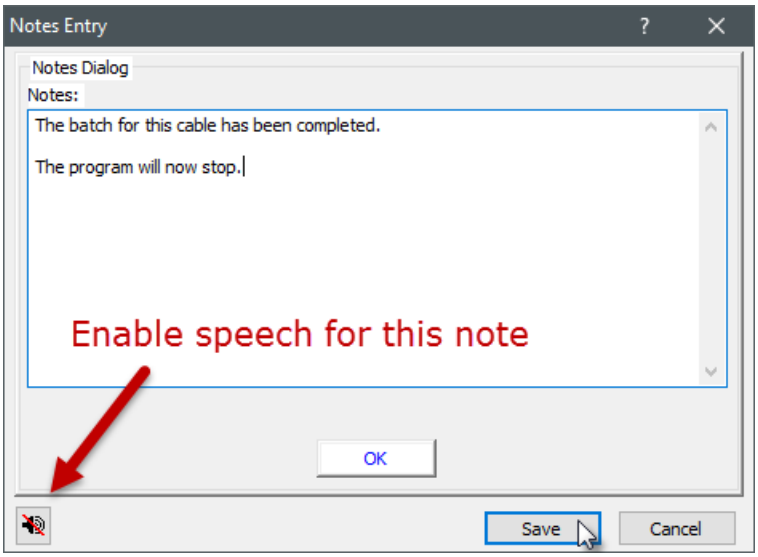
```
1   LOAD...MY CABLE
2   SET INITIAL COUNT = 242
3   WAIT FOR PB
4   TEST CABLE
5   COMPARE TEST TO MATCH
6   IF MATCH THEN...
7      BEEP
8      PRINT MATCH DATA LABEL
9   IF NOMATCH THEN...
10     BEEP BEEP
11     SKIP CABLE COUNT
12  IF COUNT = 250
13     NOTE FILE...Batch Complete.TPL
14     STOP
15  REPEAT 3
```

New instructions used in this example:

SET INITIAL COUNT = – fixes the starting value of the loop-counter at the value "n". You would enter the value "n" at the time you create the Macro and this value would then be shown embedded in the instruction (for example, SET INITIAL COUNT = 242). "n" may be any numeric value from 1 through 9999.

SKIP CABLE COUNT – prevents the loop-counter from advancing when REPEAT executes. Use this when creating serial numbers, or during data logging, to freeze the loop-counter if you encounter a defective cable. This may also be helpful when jumping forward using the REPEAT instruction.

NOTE FILE . . . – pauses execution and displays a box with a text message, containing instructions or comments you have written for the operator. When you choose this instruction during creation of your Macro, a selection list appears showing all available note file names (files ending in .TPL). You can choose one from the list or create a new one by typing the name and clicking open. A new window will open as shown in the right. Enter any desired instructions for the operator to see when the macro runs.



You can press the speech button in the lower left corner, to enable the text to be spoken trough the PC speakers when the Note File instruction executes. Click Save to insert the Note.

The name you chose is embedded in the instruction (like: NOTE FILE... Batch Complete.TPL). When executed, an "Operator Note" window appears with your text shown. The notes window remains visible until you press the TEST pushbutton, click OK in the pop-up window, or press ENTER on the keyboard. At that time the Macro resumes. *You may use NOTE FILE... in place of WAIT FOR PB if you wish to show a message at the beginning of each test loop.*

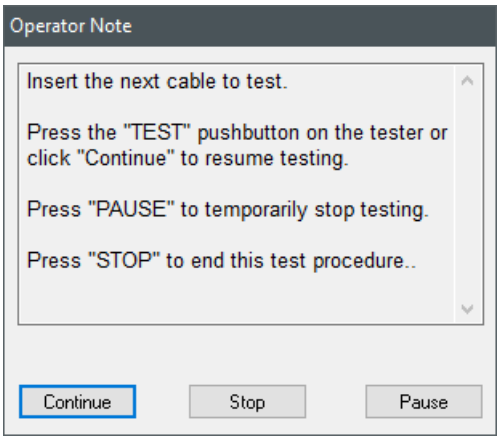
Explanation of Example 7: Let's assume that you have a cable named "MY CABLE" in the database. You need to complete a batch of 250 "good cables". The previous day you tested 242 good cables and you need to test the remaining 8. In this example, you will first need to edit the macro and set the initial count to 242, Save and then Execute. Then (1) you load your cable, (2) sets the initial count to 242. (3,4,5) wait for the button, connect the cable, test the cable and compare it. (6) If the cables match, (7,8) beep and print the match data label, but (9) if they don't match, (10) beep beep and (11) skip the cable count, (because we don't want to count bad cables, only good cables). (12) If the Test Count is equal to 250, (13) the script will open a note file, indicating that the batch has been completed and (14) the program will stop.

EXAMPLE 8: EXTERNAL CONTROL - This example requires a CB35 relay board.

```
1   LEARN CABLE
2   WAIT FOR PB...Next.TPL
3     STOP
4   TEST CABLE
5   COMPARE TEST TO MATCH
6   IF MATCH THEN...
7     SET RELAY 1
8     CLEAR RELAY 1
9   IF NOMATCH THEN...
10    SET RELAY 2
11    CLEAR RELAY 2
12  REPEAT 2
```

New instructions used in this example:

WAIT FOR PB . . . (conditional wait for pushbutton) – very similar to a NOTE FILE... instruction, however this one has three control buttons, as shown on the image below. To resume normal execution, click the button on the left ("Continue"), press the TEST pushbutton on the tester, or press the ENTER key on the keyboard. To execute the indented commands following this instruction, click the middle button ("STOP"). Clicking the right button always pauses the Macro.



Use this instruction instead of the standard WAIT FOR PB if you wish to alter the operation of the Macro based on the operator's decision.

You insert this instruction in the same way as the NOTE FILE..., selecting one note from the list or creating a new one by typing the name and clicking Open. A new window is displayed, containing the area to insert the operator notes, and three buttons. You can rename the left and middle buttons if you need to. You can also leave the middle button label blank, making the button hidden when the macro runs.

Clicking Save inserts the instruction in the macro script, with the note name you chose embedded in the instruction (for example, WAIT FOR PB . . . Next.TPL).

SET RELAY / CLEAR RELAY - These macro instructions allow you to activate or deactivate one of the ten relays available in the CB35 Relay Board, to send signals to control external instruments, counters, PLCs, etc. When you insert this instruction, the software will ask you to type in the number of the relay that you want to activate or clear. Type any number from 1 to 10 for the first relay board, 11 to 20 for the second board and so on. When this instruction is executed, you should hear the relay closing and see the corresponding LED "ON" in the relay board.

*Note: Before using a relay board, be sure to define it in the **Tester Control** preferences under the **Test Settings** Menu. Under Relay Board (CB35) select the number of CB35s that you are using.*

Explanation of Example 8: This case can be used for applications where an external signal is required to count good and bad cables for example. First, (1) the cable is learned and (2) a pop-up window appears with instructions to the Operator. If the operator presses the middle button, (3) the macro stops. If he presses the right button, the macro will Pause until the Test Button is pressed again. If the operator presses the left button, (4,5) the cable is tested and compared. (6) If the cables match, (7) relay #1 is activated and sends the signal to whatever device is wired to, and immediately after (8) the same relay is deactivated. This worked as a simple pulse, that many devices require to be activated. (9) If the cables don't match, then (10,11) relay #2 is the one activated and deactivated, sending the signal to a different device. (12) The process repeats, back to line 2, where the operator can choose to continue or not.

EXAMPLE 9: OPERATOR NAME AND SUPERVISOR APPROVAL - This example requires you to setup one administrator and one operator users. Check section 2, page 2-25 for more information.

```
1   LOAD...MY CABLE
2   ENTER OPERATOR NAME
3   WAIT FOR PB
4   TEST CABLE
5   COMPARE TEST TO MATCH
6   IF MATCH THEN...
7      COPY MATCH NOTES TO TEST
8      PRINT TEST DATA
9      REPEAT 3
10  ENTER VERIFICATION ID
11     REPEAT 10
12  REPEAT 3
```

New instructions used in this example:

ENTER OPERATOR NAME - This instruction pops-up the Operator Name selection window. You can choose a name from the drop-down list, type in the name of an existing user or input the name with a bar code scanner. If the operator has a password assigned to it, you will need to enter it before proceeding. The variable <OPERATOR> will change to reflect the new operator. More about variables in the following example.

COPY MATCH NOTES TO TEST - This instruction copies any notes saved from the match data cable, to the test data cable. You will normally use this instruction to copy the notes before printing the test data report or saving the test cable with the SAVE TEST DATA instruction.

ENTER VERIFICATION ID - This instruction requests the authorization of a supervisor or administrator to proceed with the macro execution. The administrator or supervisor should have a password assigned to them in the Operators preferences. When a valid name and password are provided, the macro resumes execution. If the operator clicks on the Cancel button, all the indented instructions after ENTER VERIFICATION ID are executed. You will normally indent a REPEAT instruction to go back and ask for the password again.

Explanation of Example 9 - (1) A cable with existing notes is loaded, (2) and a pop-up window asks for the name of the operator doing the test. The operator chooses his name from the list, and if he has a password, types the password as well. (3,4,5) The button is pressed, the cable is tested and compared. (6) If the cables match, (7) The notes stored in "MY CABLE" file are copied to the notes in the cable just tested, (8) the Test Data Report is printed, which will include the notes just copied in the previous step, and (9) the macro is repeated back to step 3. Note that we didn't use IF NOMATCH THEN this time. That's fine, because the only way that step 10 can be executed is if the cables don't match. So, If the cables don't match, (10) the program is paused, and a screen appears asking for the name and password of a supervisor or administrator. (11) If a wrong password is provided, the program will not resume. This is useful to keep control of bad cables that you don't want otherwise mixed by mistake with good ones. (12) Finally if the proper password is entered, the macro resumes and jumps back to step 3.

6.1.5 Variables and more examples

CableEye allows the use of variables as placeholders in the notes field, label field and reports. These variables store a value which will be replaced when the note, label or report is printed. For example, if you had entered the following text in the Match Data Notes: "Cable number <COUNT>, tested on <DATE> at <TIME>."

You would see the following text printed on the report when the instruction PRINT MATCH DATA NOTES executes: "Cable number 234, tested on 8-22-16 at 9:41 AM"

There are many system variables ready for you to use, like operator, testresult and count. However, you may also create your own variables to store data like serial number, batch number, work order, etc. Any information that you want to display in a report or label can be stored in a variable.

System Variables - The following is a list of the existing system variables and a detailed explanation:

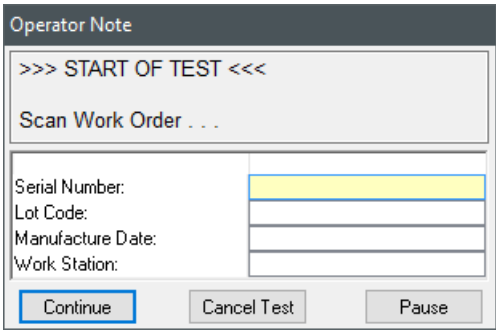
<CABLENAME> - It stores the name of the currently loaded cable in the match data window.
<COUNT> - It keeps record of the current macro loop. Its value increases with the REPEAT instruction.
<CONTCYCLES> - Stores the number of cycles that the continuous test ran before proceeding.
<CONTERRORS> - Stores the number of errors found during a continuous test before proceeding.
<DATE> - Stores today's date.
<FAULTLIST> - Stores a summary of faults with the following format: "J1:1-J2:1 OPEN".
<LOGFILE> - Stores the name of the currently open log file
<MACRONAME> - Stores the name of the currently executed macro
<MATCHNOTES> - Stores the existing match notes.
<OPERATOR> - Stores the currently active operator username.
<REPORTRESULT> - Stores the result of the test, but with a space between the letters. Ex. "P A S S".
<SOCKETLIST> - Stores a list of the sockets in use by the current loaded cable.
<TESTRESULT> - Stores the result of the test.
<TESTTIME> - Stores the time at which the test was performed
<TESTTIMELOCAL> - Stores the local time at which the test was performed
<THRESHHIGH> - Stores the cable's Resistance High Threshold.
<THRESHLOW> - Stores the cable's Resistance Low Threshold.
<TIME> - Stores the time at which the report was printed.
<VERIFYID> - Stores the username of the current verification ID user.

Custom Variables - You can define your own variables as mentioned before, that will allow you to store and display any pertinent information in your reports. These variables are defined using the WAIT FOR SCAN... instruction in a macro as shown in the following example:

EXAMPLE 10: SCAN CUSTOM VARIABLES

```
1   LOAD...MY CABLE
2   WAIT FOR SCAN...Enter_Batch_D
3     STOP
4   TEST CABLE
5   COMPARE TEST TO MATCH
6   IF MATCH THEN...
7     PRINT MATCH DATA LABEL
8   REPEAT 2
```

New instructions used in this example:
WAIT FOR SCAN... - This instruction operates very much like WAIT FOR PB... but in addition to offering branch options, it accepts either scanned or typed data input for the specified variables. When embedded in a cable test loop, it permits you to enter a unique serial for every cable tested, either by scanning a bar code or typing it. Other data may be scanned in also, such as lot number, workstation, etc. The screen that you see below provides an example.

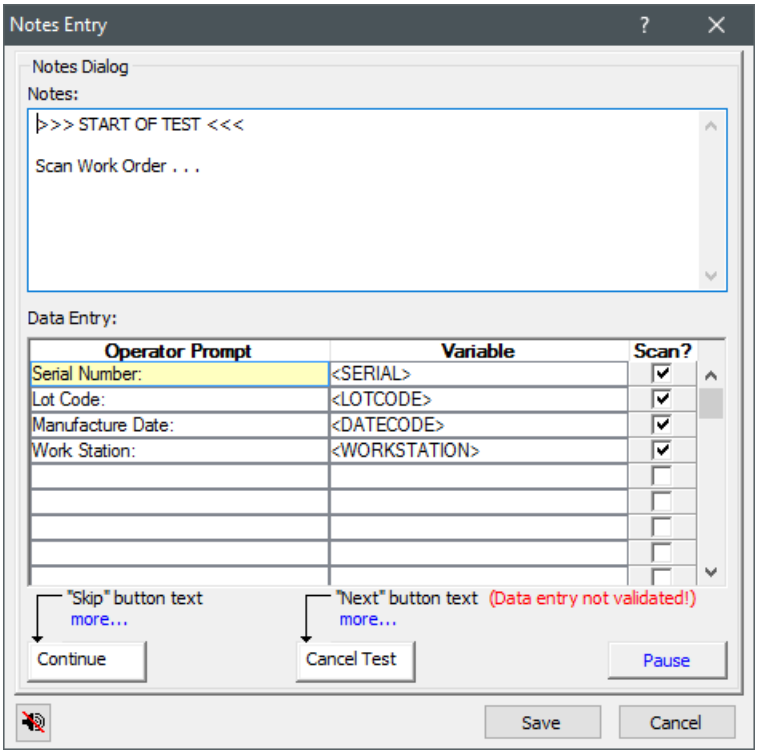


In this example, there are 4 data entries that the operator needs to scan in order to proceed with the macro execution.

If the operator presses the Continue button, it will skip any following indented instructions, as with WAIT FOR PB... If the middle button is pressed, then any following indented instruction will be executed, and for this macro example, the macro would stop.

You can scan as much data as you need with the WAIT FOR SCAN... instruction. Note that some data you may want to scan only once, at the beginning of the script, but others, like the serial number, you want to scan it in every loop in the program. You accomplish this by having two WAIT FOR SCAN instructions in different parts of the script.

When you insert the WAIT FOR SCAN... instruction in the macro, you will be asked to choose the file, as with the NOTE FILE and WAIT FOR PB instructions. You can create a new file by typing in the name and clicking open.



The window shown in the right will open. You may enter any instructions in the top field for the operator.

The table in the middle is where you define your custom variables and operator prompts that would be displayed when the instruction is executed. Your variable name should be enclosed in angle brackets like "<SERIAL >".

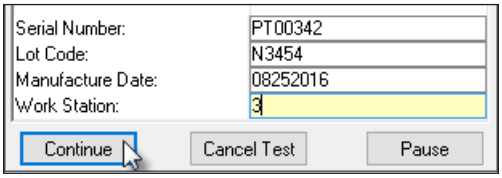
The "Scan" checkbox at the right side of the table, controls if the variable has to be scanned or not before proceeding. A checkmark on one of the variables, will force the operator to scan or type in a value for that variable before resuming the execution of the macro. This would normally be checked, unless you have some non-critical data that can be omitted.

Finally, the 3 buttons in the bottom work exactly as in WAIT FOR PB... You can change the labels of the left and middle buttons, and if you leave the label for the middle button blank, the button will not be displayed. It is critical to remember that pressing the left side button when the macro is running, will skip any indented instruction following WAIT FOR SCAN, where clicking the middle button will execute those instructions.

Explanation of Example 10 - Assume that we have a cable named "MY CABLE". We saved a label for this cable as shown on the right. When the macro is executed, (1) The cable is loaded and (2) the WAIT FOR SCAN window appears, as shown in the previous page.

Test Result: <TESTRESULT>
Serial Number: <SERIAL>
Lot Code: <LOTCODE>
Manufacture Date: <DATECODE>
Work Station: <WORKSTATION>

The operator scans or types in the data on the right and clicks the Continue button or presses the TEST button on the tester. Note that many bar code readers will send an Enter signal after scanning, which will automatically work as pressing the Continue button.



(4,5) The cable will be tested and compared, and (6) if the cables match, (7) the match label will be printed as shown on the right. Note that all the custom variables were replaced by their scanned value, and the system variable TESTRESULT was replaced with the actual result of the test, "PASS".

Test Result: PASS
Serial Number: PT00342
Lot Code: N3454
Manufacture Date: 08252016
Work Station: 3

Note: You can use your custom variables not only on the label field, but also in the notes field, test reports and log reports.

EXAMPLE 11: TEST MULTIPLE CONFIGURATIONS AND RICH TEXT NOTES

```
1   CLEAR TEST DATA
2   LOAD... MY CABLE POS 1
3   WAIT FOR PB
4   TEST CABLE
5   COMPARE TEST TO MATCH
6   IF NOMATCH THEN...
7      BEEP BEEP
8      STOP
9   CLEAR TEST DATA
10  LOAD... MY CABLE POS 2
11  RTF NOTE...Position2|Continue
12  TEST CABLE
13  COMPARE TEST TO MATCH
14  IF MATCH THEN...
15     BEEP
16     PRINT TEST DATA
17  REPEAT 1
```

New instructions used in this example:
CLEAR TEST DATA - This instruction will clear the test data buffer. This is useful when you need to load another cable in the middle of the script, and clear up the results of the previous test.

RTF NOTE... - Displays a popup window with an rtf (rich text file) that you specify, which may include pictures as well as formatted text. RTF NOTE... works exactly like WAIT FOR PB... but with a rich format message instead.

When you insert the RTF NOTE... instruction in edit mode, you will be asked to choose the rtf file from the list. You can choose an existing one or create a new one by typing the name and clicking Open. This will open your default rtf file editor program.

You can add any formatted text or pictures to your file as shown in the image below. At any point, you can save the rtf file, (but don't close it) and you will see the changes transferred to the rtf note window in CableEye. This can be used as a preview to check if everything fits correctly on the note screen. You can go back to the rtf editor and make more changes. Once you are happy with the results, you can save it and close it.

Now, back to the CableEye note screen, as with the WAIT FOR PB... instruction, choose the label for your buttons, which will be used for skipping (left) or executing (middle) indented instructions.

Explanation of Example 11 - In this example we have a cable that contains a switch, which if flipped, changes the configuration of the cable itself. To test a cable like this, you need to save 2 cables, one for each one of the switch positions. We call these cables POS1 and POS2.

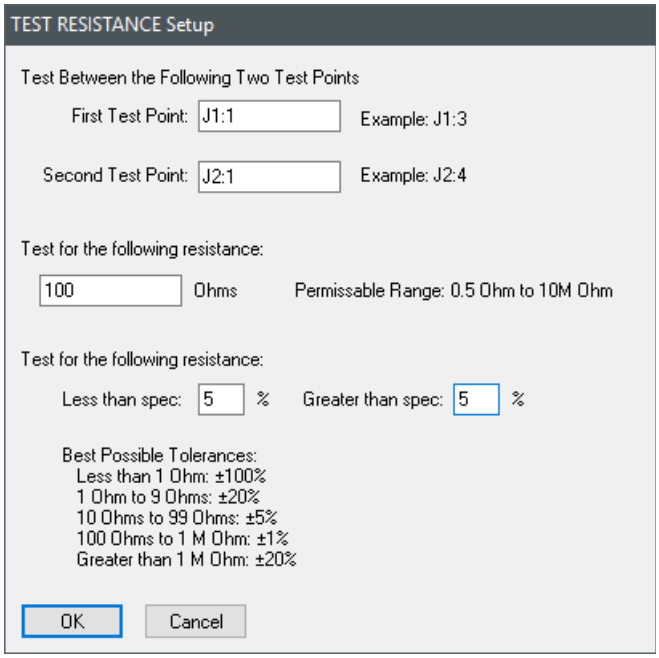
(1) Test data is cleared and (2) cable POS1 is loaded. (3,4,5) Cable is tested as usual and (6,7,8) if the cable doesn't match, it stops the test, otherwise, (9) it clears the test results and (10) loads the POS2 cable. (11) An RTF note is called to give instructions to the operator to flip the switch in the shown position, and once the operator clicks Continue or presses the TEST push button on the tester, (12) the new cable POS2 is tested and (13) compared. (14) If the cable matches, (15,16) it beeps and prints the test data report.

MORE INSTRUCTIONS NOT EXPLAINED IN EXAMPLES

CALL JAVASCRIPT - Allows you to call a JavaScript script from within macros. This is very useful when you need to perform a complex function not possible in macros, but you don't want to write your entire program in JavaScript. More about JavaScript Automatic testing on page 6-23.

CLEAR MATCH DATA - Clears the match data buffer, similarly to the CLEAR TEST DATA instruction. Note that this instruction will clear the match data buffer, even if changes have been made to it. Normally when you make changes to the match data, and you manually try to clear the screen by clicking the CLEAR button in the left side, the program asks you if you want to save the changes made to it. This will be omitted with this instruction.

CHECK RESISTANCE... - Use this instruction to check the resistance between two test points. As shown in the image in the right, the two test points should be defined as the Connector Name followed by a colon ":" and then the number of the pin in that connector. You need to specify the resistance, and you can use k, or M for kΩ and MΩ. Finally set the tolerance in the Less than spec and Greater than spec fields.



Use IF MATCH or IF NOMATCH instructions after this one, to control what happens if the resistance matches or not.

When the instruction is entered, it will be displayed in the script as follow:

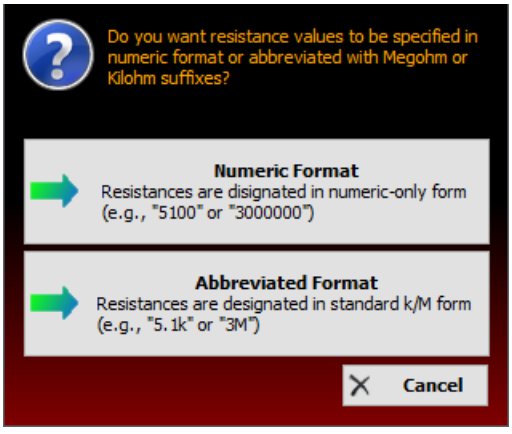
CHECK RESISTANCE... J1:1,J2:1,100,5,5

CONTINUOUS SCAN - This instruction is very similar to CONTINUOUS TEST. The only difference is that this instruction will stop immediately if an intermittence is found, and will proceed to the next instruction. If a difference is not found, the instruction will keep cycling until the resume button is clicked or the TEST button pressed.

COPY MATCH TO TEST - This instruction will copy the contents of the match data buffer to the test data buffer. Everything stored with the match data cable will be transferred, like notes, labels and variables.

COPY TO CLIPBOARD - This instruction will copy the test buffer connections and values to the clipboard. You can paste the clipboard into a program like excel to save the result in a different format if required. Once you copy the items to the clipboard, you can paste them (Ctrl + v) in any program you like.

When you insert this instruction, the software asks you how do you want the resistance values to be formatted. You can either choose Numeric Format or Abbreviated Format as shown on the right.



Numeric format will display the numbers as “5100” and “3000000”, where Abbreviated format will display them as “5.1k” and “3M” respectively.

DISABLE HIPOT UNIT - Used only with a High Voltage tester, it will disable HiPot usage, and you will need to press the black HV Enable button to enable it again.

DISABLE PAUSE - Disables the Pause buttons in the WAIT FOR PB and WAIT FOR SCAN instructions. Also, note that when you run a macro, if you press the TEST button on the tester at any point, the macro will pause, but if you use this instruction at the beginning of the macro, it will also disable this behavior and pressing the TEST button will no longer pause the macro.

ENTER INITIAL COUNT – Displays a window asking the operator to enter the starting value of the Test Count, (<COUNT>). When the operator enters a numeric value and clicks OK or presses ENTER, the entered value becomes the current test count. Valid entries range between 1 and 9999. Unlike SET INITIAL COUNT =, no specific count value is connected with this instruction. Rather, it is a “run-time” entry in that the operator is asked to select the desired count at the time of execution. Use this when creating serial numbers based on the test count and starting at a value other than “1”.

EXTERN HIPOT - Only used with a High Voltage tester External Terminals. This instruction will run one of your HiPot External Terminals presets. The instruction will ask you for the preset that you want to run and a delay in seconds before starting the test.

IF NO TEST DATA... - It is possible that when you are testing in a macro loop, the cable that you are going to test has not yet been connected to the tester. If this happens and the macro runs, you will get a FAIL result, even if a cable wasn’t really tested. To avoid this, you can use IF NO TEST DATA... after TEST CABLE, to verify that a cable has been tested, and nest any indented instruction to display a message for example, to let the operator know that a cable has not been connected yet. In this way, the errors counter is not increased and the operator gets a clear message of what the problem is.

ON ERROR GOTO... - Use this macro instruction to declare an error handler at a line number in your macro. This lets you keep the macro running if CableEye encounters an error (such as no cable connected) when executing a TEST CABLE or LEARN CABLE instruction. When you declare the error handler, execution resumes at the line number specified, when an error in one of the instructions would otherwise cause the macro to stop.

PROBED MATCH - This instruction will initiate the Probe Testing function to test single side cables. This is the same as manually enabling the Probe checkbox in the Test Data Window and clicking the Start button after. A cable should be loaded in the match data buffer for this instruction to work. For more information about Probe testing, check Section 3 - Basic Operations, on page 3-33.

RELAY ASSERT ON / OFF - This 2 instructions control the available output signal in the remote port in the rear side of the CableEye testers, (not M2U-Basic). You can use this signal when you only need one output to control an external device. If you need more outputs, you can use the CB35 Relay Board, along with the SET / CLEAR RELAY instructions already explained. Please refer to your tester’s Getting Started Booklet, which includes an schematic showing you the signal that is controlled by this instruction.

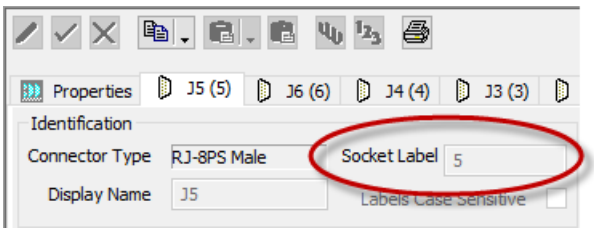
SAVE VARIABLES - Saves the current macro user variables (e.g., “<SERIAL>”) to disk so that they can be restored by the RESTORE VARIABLES instruction.

RESTORE VARIABLES - Restores the macro user variables that were previously saved for the user by the SAVE VARIABLES instruction.

SAVE TEST DATA - Use this instruction to save the current Test Data in the database. The contents of the Test Data Buffer along with any currently active variables save to the database using the name specified. The specified name should use a variable to create a unique filename, like <cablename>-<count> or <testtime>, very similar to how the EXPORT TO PDF files are named.

SET CABLE TYPE - Sets the acquired cable to the specified type. The type can be any word that you want to use to organize your cables. Used before SAVE TEST DATA to set the type for the test data.

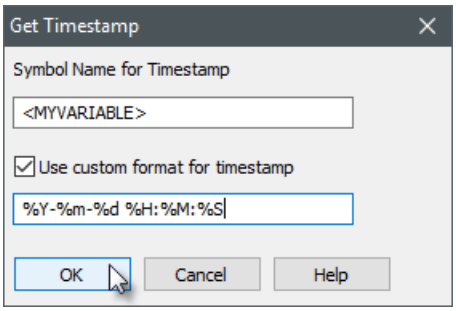
SET LED - Turns on the LED defined in the custom test fixture. The Socket Label for that connector is used with this instruction. The picture in the right shows where to set the Socket Label in the PinMap software. When you insert this instruction, it will ask you for the LED that you what to set, in this case 5. The instruction will look like this: SET LED 5.



CLEAR LED - Clear LED will power off a LED turned on by the SET LED instruction. In the example above we used SET LED 5 to turn on the LED of connector J5. Use CLEAR LED 5 to turn it off.

SET THRESHOLDS... - This instruction allows you to set resistance measurement thresholds that only are on effect while you are running the macro.

TIMESTAMP... - This instruction copies the current system date and time into the specified variable, optionally with a custom format. When you enter the TIMESTAMP instruction in the macro editor, CableEye displays a dialog in which you enter the name of your variable to copy the current time into when the instruction executes. After the instruction executes, the variable contains the time and date formatted as MM-DD-YYYY HH:MM:SS. To store the date in a different format, check the Use custom format for timestamp box and enter your custom formatting string in the field provided, as shown in the right.



You can use the following symbols for formatting the timestamp:

%a	Abbreviated weekday name	%A	Full weekday name
%b	Abbreviated month name	%B	Full month name
%d	Day of month as decimal number (01 – 31)	%H	Hour in 24-hour format (00 – 23)
%I	Hour in 12-hour format (01 – 12)	%m	Month as decimal number (01 – 12)
%M	Minute as decimal number (00 – 59)	%p	A.M./P.M. indicator for 12-hour clock
%S	Second as decimal number (00 – 59)		
%j	Day of year as decimal number (001 – 366)		
%U	Week of year as decimal number, with Sunday as first day of week (00 – 53)		
%w	Weekday as decimal number (0 – 6; Sunday is 0)		
%W	Week of year as decimal number, with Monday as first day of week (00 – 53)		
%y	Year without century, as decimal number (00 – 99)		
%Y	Year with century, as decimal number		

Thus %Y-%m-%d %H:%M:%S will appear as 2016-08-30 16:16:25, when CableEye executes the instruction.

USE CABLE... - Use this instruction to load an specific cable, based on the string that you define. Very similar to how SAVE TEST DATA is defined, you can use variables and text strings to call the file that you want to load. Note that if a cable file with the name of the resulting string doesn't exist, you will get an error message and the macro will stop, unless you handle it with the ON ERROR GOTO instruction.

USE LABEL... - Specifies the custom label definition file to use for the next PRINT MATCH DATA LABEL or PRINT TEST DATA LABEL instructions. Just enter the name of the label file that you want to use and then use any print label instruction after it.

6.1.6 Other Information about Macros

Macro Files: Macros are stored as individual files in the "Macros" folder located within your CableEye folder in the following directory: **Program Files (x86)\CableEyeV5\Software\Data\Macros**

All Macro files end in ".MAC" to identify them to the CableEye software, and must be located in the Macros folder to be recognized. To delete a Macro, you must drag the Macro file into the trash from the Windows desktop. To ensure that removal is a deliberate operation, we do not provide a "Delete" button in the Macro window.

Printing Macros: Click the print button visible in the Edit window. You must open the Edit window to access this button.

Run-Time Errors: Minor errors that occur during Macro execution sound an error tone and may generate a message, but do not halt execution. Errors like this usually result from missing data (such as trying to compare Test and Match data when Match data is missing), and may correct themselves on successive loops.

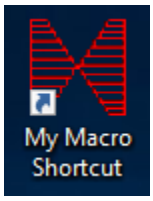
Serious errors sound an error tone and halt execution, usually with a message. Such errors result from a defect in logic (such as branching to a nonexistent line) or a missing or duplicate file (such as loading a cable file that does not exist). If you test newly created Macros thoroughly before employing them in production, this should never happen.

Don't forget about the STEP function (Macro menu) – it lets you check the result of each Macro instruction step-by-step to ensure that it operates correctly.

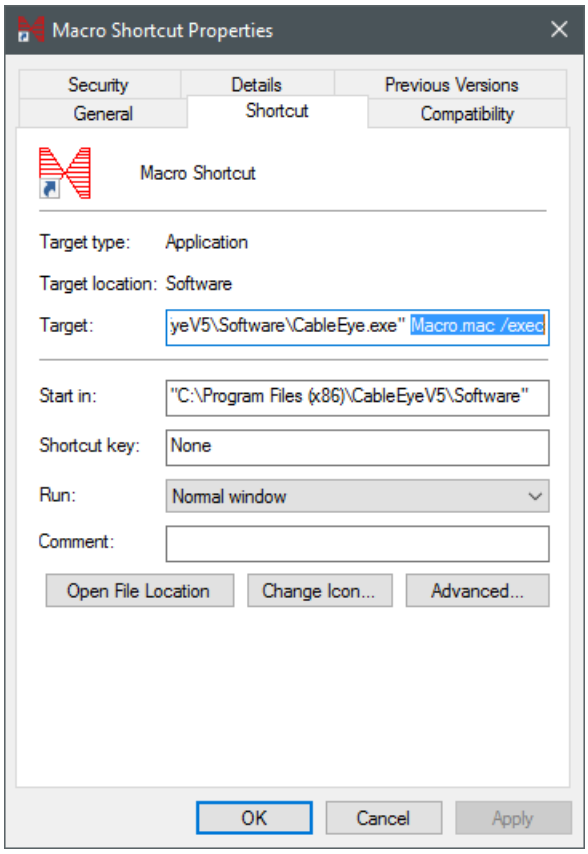
Infinite Loops: Many Macros are designed to operate indefinitely in an endless loop. In these cases, the operator must click "Stop" to halt execution. Macros that repeat indefinitely should include at least one WAIT FOR PB instruction so that you may cue the system when you are ready to start the next test cycle.

Loading a Macro at Startup: The window setup that exists when you close the CableEye application will be reinstated when you start again. So, if you close the application with the Automatic Test window visible, it will reappear in exactly the same configuration with the same Macro selected.

Using Shortcuts to Start Specific Macros: You may create individual shortcuts for CableEye, each with its own properties. One property you may customize defines which Macro loads when you start the software. You may thus create different shortcuts, each loading a different Macro at startup, to address a different job or test requirement. This eliminates the need for an unskilled operator to make a Macro choice when using CableEye and simplifies operation.



To set this up, create a shortcut for the CableEye executable (CableEye.exe) and right-click on the shortcut. Choose "Properties" and click the "Shortcut" tab to reveal the window seen at the right. In the "Target" text box, after the executable's name, enter a space and then type the full name of the Macro you wish to load.



Add a space and the text **/exec** to execute the macro immediately when the program opens. Don't include /exec if you which to press the TEST button to start the macro execution.

Note that adding /exec will force the program to close when the macro stops. This is very useful if you don't wish to give any access to the operator outside of macros.

If desired, assign a hotkey in "Shortcut Key". For example, type F5 and pressing F5 starts CableEye and loads "Macro.mac". Once the software opens, pressing the TEST button on the tester is all it takes to start execution.

6.2 JavaScript

Like macros, JavaScript allows you to automate the test procedure using CableEye functions. You can write JavaScript programs and run them directly from within CableEye. The programs you write can perform the same automated testing tasks traditionally accomplished with Macros, but the full-featured programming structure of JavaScript makes it vastly more suitable for implementing moderate to complex scripts.

6.2.1 About JavaScript

JavaScript is primarily known as a scripting tool for customizing and automating web pages. However it is a complete programming language in itself and can be used for purposes that have nothing to do with web programming. In CableEye, the web page functions (known as the Document Object Model or DOM) are replaced by a set of functions that let you control your testing process.

Note that knowledge of JavaScript programming is required, and the teaching of JavaScript goes beyond the scope of this manual. For online resources for learning JavaScript as well as language reference material, visit the following sites:

http://eloquentjavascript.net/
This is an excellent introduction to JavaScript programming by Marijn Haverbeke. There are numerous interactive examples to help guide you through learning the language.

Chapters 1-10 describe the language syntax and usage and provide you everything you need to know to use JavaScript in CableEye. The remaining chapters (11-14) cover web programming and don't directly pertain to using JavaScript in CableEye (though they make interesting reading anyway).

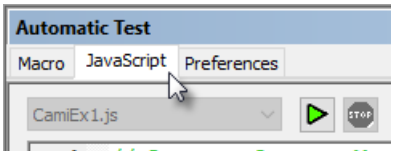
https://developer.mozilla.org/en/JavaScript/A_re-introduction_to_JavaScript
If you've used JavaScript before but would like a quick single-page refresher course in the language's syntax and capabilities, try this page. It links heavily to the following reference material for more detailed information.

https://developer.mozilla.org/en/JavaScript/Reference
This is a detailed and complete JavaScript language reference from the same organization that publishes the Firefox browser. If you're familiar or moderately familiar with JavaScript syntax, this is an excellent reference source.

6.2.2 JavaScript Panels

To create, edit, and run JavaScript programs in CableEye, click the AUTO button on the main CableEye toolbar.

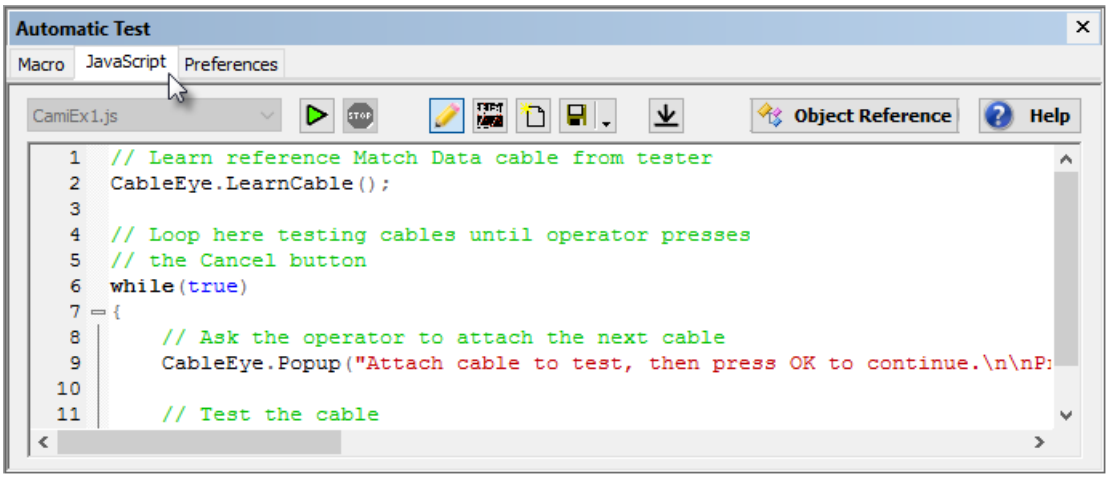
Then select JavaScript by clicking the tab at the top of the Automatic Test Window, as shown in the right.



When you are developing your JavaScript program, you will want to be able to see and edit the code you're working on. You can do this directly from within CableEye using the development panel, shown at the bottom. When you have finished developing the script, however, and are using it in production, your operators may not care to see your code which, elegant as it may be, takes up valuable screen real estate that can be used to show relevant test information. In this case, you run your script from the docked runtime panel:

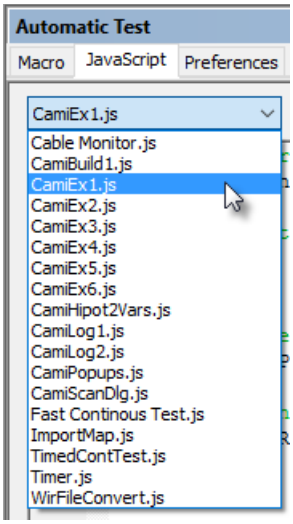
6.2.3 Developing Scripts

You can display, edit, and run your program using the JavaScript development panel, shown here:



The program itself is displayed in the editing window that occupies most of this panel. When your script is not running, you can use it to view and edit the script itself. The controls at the top of the panel allow you to load, run, save, and edit the script.

Loading and Running Scripts



Select the script that you want to work on using the file selector dropdown shown in the right. Click on the control to reveal the available script files (stored in the Macros subdirectory of the CableEye data directory), and select the file to load.

Click on the run script button to begin executing the script.

When the script is running, you can stop it by clicking the stop script button . Note that if there is a dialog box in the foreground, only buttons on that dialog box may be enabled and the stop button will not receive your mouse clicks.

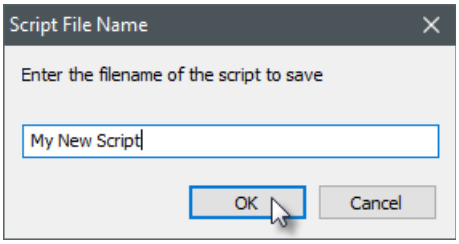
Editing and Saving the Script

To begin editing the script click the start edit button . This allows you to enter text into the script window below.

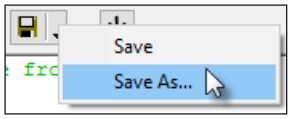
Click the cancel edit button if you want to discard your most recent edits and return the script window to read-only.

To clear the script window and begin editing, click the new script button .

To save your script, click the save button . Normally this will save your script back to the same file you read it from. However if this is a new script, CableEye will ask to specify a file name for the script, as shown on the right.



Likewise, you can also click on the drop down area of the save button and click the Save As item to save under a new file name



Switching Between Development and Run Panels

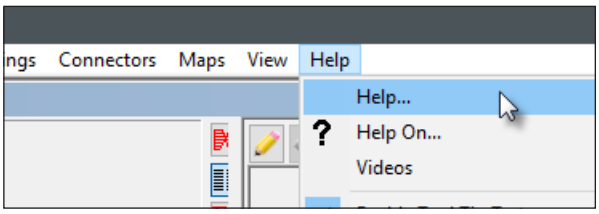
To switch to the small docked runtime panel, click the shrink button . This collapses the script window to a small panel docked to the bottom of the main window

To expand the window back to the script development window, click the expand button .

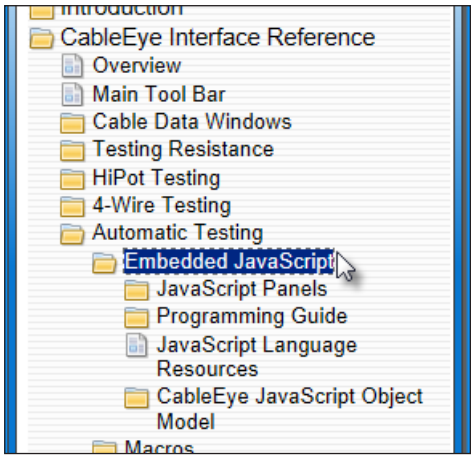
Note on the image above that when the script is executed, the rectangular indicator changes from **Stopped** to **Running**. If you are running a script, you can't expand the panel.

6.2.4 Programming Guide

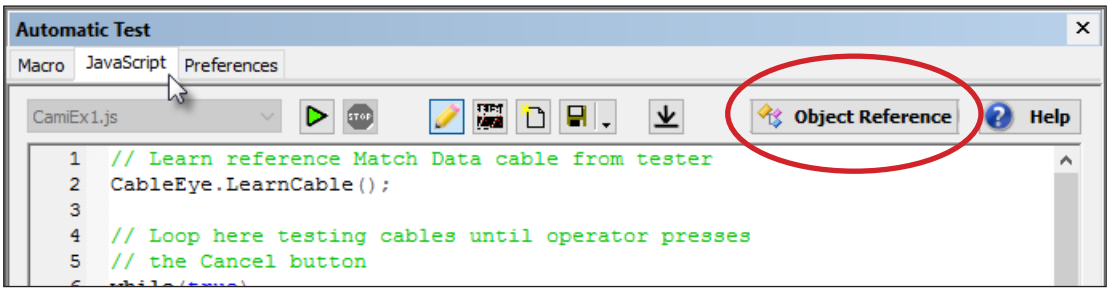
A JavaScript programming guide would be too extensive to be included in this manual. However, the CableEye Online Help System contains a detailed explanation, which includes General Hints, Examples, Result Logging and Operator Input. To access the Online Help System, just open the CableEye software and **select Help...** from the **Help Menu** as shown below.



Once the help system opens, use the left side panel to navigate to: ***CableEye Interface Reference | Automatic Testing | Embedded JavaScript,*** as shown in the right.



From the JavaScript panel you also have direct access to the Object Reference, which is a section that describes the object model (methods, properties, and constants) available to embedded JavaScript programs in CableEye.



The information included in the Help System is detailed and constantly updated. We encourage you to read and follow the examples explained there.

6.2.5 CALL JAVASCRIPT

CALL JAVASCRIPT is a macro instruction that calls a JavaScript script to perform more complex tasks, not possible with a macro.

The first step is to create your Javascript script as explained in page 6-24 and Save it, assigning a unique name to it.

Now you can call that Javascript from the Macro program using CALL JAVASCRIPT. A pop up appears, where you should enter the name of the script that you just saved. When this instruction runs, it will execute the script and resume the macro execution when done.

The macro instruction will look something like this: **CALL JAVASCRIPT... My Javascript**

6.3 CableEye's Application Programming Interface (API)

The CableEye V5 Application Programming Interface (API) allows you to perform many of the functions of the standard CableEye application from your own application. Using the API you can:

- Acquire Test Data from your CableEye tester
- Load Match Data from your cables database
- Determine the comparison status between the Test and Match data
- Get a list of Test or Match Data connections
- Get a list of differences (shorts / opens / tolerance violations) between Test and Match Data
- Control LED states on the tester to give visual PASS/FAIL indications to your operators
- Programmatically construct cable data records and save them to the cables database

> Note that the Application Programming Interface (API) is an optional add-on software site license not included with the main CableEye software. Contact CAMI Research to get a license.

CAMI offers two separate interfaces that you can use to control your CableEye tester from your own application program:

- An ActiveX control that can be embedded in a Visual Basic® form or used in a .NET program or any other programming environment that can use an ActiveX control.
- An interface for LabVIEW™ that provides a library of Virtual Instruments (VIs) that you can add to or reference from your LabVIEW program.

To learn more about the Application Programming Interface, consult the CableEye Online Help System in the software.