# Adaptrade Editor

Version 1

# User's Guide

# Disclaimer

HYPOTHETICAL OR SIMULATED PERFORMANCE RESULTS HAVE CERTAIN INHERENT LIMITATIONS. UNLIKE AN ACTUAL PERFORMANCE RECORD, SIMULATED RESULTS DO NOT REPRESENT ACTUAL TRADING. ALSO, SINCE THE TRADES HAVE NOT ACTUALLY BEEN EXECUTED, THE RESULTS MAY HAVE UNDER- OR OVER-COMPENSATED FOR THE IMPACT, IF ANY, OF CERTAIN MARKET FACTORS, SUCH AS LACK OF LIQUIDITY. SIMULATED TRADING PROGRAMS IN GENERAL ARE ALSO SUBJECT TO THE FACT THAT THEY ARE DESIGNED WITH THE BENEFIT OF HINDSIGHT. NO REPRESENTATION IS BEING MADE THAT ANY ACCOUNT WILL OR IS LIKELY TO ACHIEVE PROFITS OR LOSSES SIMILAR TO THOSE SHOWN.

Last Revision: December 2022 (version 1.3.0)

# Software License Agreement

These license terms are an agreement between Adaptrade Software and you. Please read them. They apply to the software named above, which includes the media on which you received it, if any. The terms also apply to any

- updates,
- supplements, including EasyLanguage code files for TradeStation, code for other trading platforms, and
- support services

for this software provided by Adaptrade Software, unless other terms accompany those items. If so, those terms apply.

BY CLICKING ON THE "I AGREE" BUTTON WHERE INDICATED, OR BY COPYING, INSTALLING OR OTHERWISE USING THE SOFTWARE, YOU ACCEPT THESE TERMS. IF YOU DO NOT AGREE TO THE TERMS OF THIS AGREEMENT, DO NOT INSTALL OR USE THE PROGRAM AND DESTROY ALL COPIES OF IT.

If you comply with these license terms, you have the rights below.

1. LICENSE MODEL. The software is licensed on a per user basis.
2. INSTALLATION AND USE RIGHTS. You may install any number of copies of the software on your devices, provided it is for your use only. A "single user" license permits the use of the software on no more than one device at a time. A "two-user" license permits the software to be run on two devices at the same time, and so on.
3. SCOPE OF LICENSE. The software is licensed, not sold. This agreement only gives you some rights to use the software. Adaptrade Software reserves all other rights. Unless applicable law gives you more rights despite this limitation, you may use the software only as expressly permitted in this agreement. You may not
   - reverse engineer, decompile or disassemble the software, except and only to the extent that applicable law expressly permits, despite this limitation;
   - make more copies of the software than specified in this agreement or allowed by applicable law, despite this limitation;
   - publish the software for others to copy;
   - rent, lease or lend the software;
4. BACKUP COPY. You may make backup copies of the software. You may use these copies only to reinstall the software.
5. EXPORT RESTRICTIONS. The software is subject to United States export laws and regulations. You must comply with all domestic and international export laws and regulations that apply to the software. These laws include restrictions on destinations, end users and end use.
6. SUPPORT SERVICES. Support services are as described on the Adaptrade Software web site, www.Adaptrade.com.
7. ENTIRE AGREEMENT. This agreement, and the terms for supplements, updates and support services that you use, are the entire agreement for the software and support services.
8. APPLICABLE LAW.
   a. United States. If you acquired the software in the United States, California state law governs the interpretation of this agreement and applies to claims for breach of it, regardless of conflict of laws principles. The laws of the state where you live govern all other claims, including claims under state consumer protection laws, unfair competition laws, and in tort.
   b. Outside the United States. If you acquired the software in any other country, the laws of that country apply.
9. LEGAL EFFECT. This agreement describes certain legal rights. You may have other rights under the laws of your country. You may also have rights with respect to the party from whom you acquired the software. This agreement does not change your rights under the laws of your country if the laws of your country do not permit it to do so.
10. DISCLAIMER OF WARRANTY. THE SOFTWARE IS LICENSED “AS-IS.” YOU BEAR THE RISK OF USING IT. ADAPTRADE SOFTWARE GIVES NO EXPRESS WARRANTIES, GUARANTEES OR CONDITIONS. YOU MAY HAVE ADDITIONAL CONSUMER RIGHTS UNDER YOUR LOCAL LAWS WHICH THIS AGREEMENT CANNOT CHANGE. TO THE EXTENT PERMITTED UNDER YOUR LOCAL LAWS, ADAPTRADE SOFTWARE EXCLUDES THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT.
11. LIMITATION ON AND EXCLUSION OF REMEDIES AND DAMAGES. YOU CAN RECOVER FROM ADAPTRADE SOFTWARE ONLY DIRECT DAMAGES UP TO THE AMOUNT PAID FOR THE SOFTWARE. YOU CANNOT RECOVER ANY OTHER DAMAGES, INCLUDING CONSEQUENTIAL, LOST PROFITS, SPECIAL, INDIRECT OR INCIDENTAL DAMAGES.

    This limitation applies to
    a. anything related to the software, services, content (including code) on third party internet sites, or third party programs; and
    b. claims for breach of contract, breach of warranty, guarantee or condition, strict liability, negligence, or other tort to the extent permitted by applicable law.

    It also applies even if Adaptrade Software knew or should have known about the possibility of the damages. The above limitation or exclusion may not apply to you because your country may not allow the exclusion or limitation of incidental, consequential or other damages.

# Table of Contents

# Chapter 1

## Introduction

### Overview

Adaptrade Editor ("Editor") is a software program for Windows that allows the user to write trading strategies without coding. All elements of the strategy can be specified using a combination of menus, check boxes, and other point-and-click user interface elements. The program writes the corresponding strategy code in the scripting language selected by the user for any of the following trading platforms: TradeStation, MultiCharts, NinjaTrader 7/8, MetaTrader 4, and AmiBroker.

The strategy logic and other settings are specified on a series of tabbed windows, as shown below in Fig. 1.1. The strategy logic is entered as one or more logical conditions, such as (expressed in EasyLanguage for TradeStation/MultiCharts):

Average(C, 14) >= Highest(H, 10)[4].

Conditions can be simple or complex, including nested conditions (subject to language restrictions), such as:

TriAverage(Lowest(Average(C, 10), 15), 25) < XAverage(AS_PivotS(1), 12).

The open document represents a single trading strategy. Only one strategy can be open in the program at a time. Saving the current strategy writes the trading strategy settings and other program options to a file with the extension .gpcode (e.g., MyNewStrategy.gpcode).

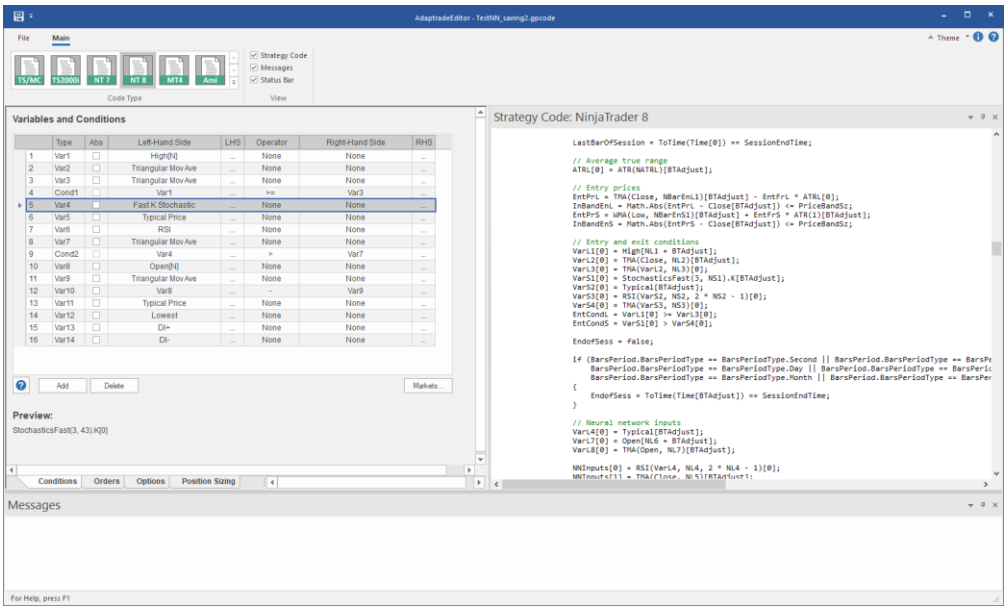



**Figure 1.1. Main window of Adaptrade Editor.**

After the logical conditions are defined on the Conditions tab, they can be selected on the Orders tab as conditions for entry and exit orders. For example, if the first condition shown above were selected for a long entry, a long entry order would only be placed if the average close over the last 14 bars was greater than the highest high over the last 10 bars four bars ago. Different conditions and other order details can be specified separately for entries and exits and for long and short trades.

Options on the Orders tab allow for market, stop, and limit orders. The prices for entering or exiting on a stop or limit (target) order can be specified in a variety of ways, including as prices calculated from indicators.

The Options tab includes a series of check-box options for limiting the number of entries per day, limiting the entry times, specifying the exit time, and selecting other options. In addition, you can add a neural network entry condition on this tab.

The neural network option allows you to select any variables entered on the Conditions tab as inputs to the neural network. You can specify the number of nodes in the hidden layer and initialize the weights of the neural network; the optimal weight values could then be determined in the trading platform (e.g., in TradeStation or in the Strategy Analyzer of NinjaTrader 8) using the platform's built-in strategy input optimizer.

Editor also includes six optional position sizing methods via the Position Sizing tab to add position sizing features to your strategy code, including fixed fractional, fixed ratio, fixed size, and constant value methods.

After making a change to your selections or at any other time, you can click one of the Code Type buttons in the ribbon menu to update the strategy code in the code pane, as shown on the right side in Fig. 1.1. The code in the code pane can be copied to the clipboard for pasting into the editor or compiler of your trading platform, such as into the NinjaScript Editor of NinjaTrader 8.

In addition to its purpose as a stand-alone trading strategy editor, Adaptrade Editor is also designed to be integrated with Adaptrade Builder, an auto-coding trading strategy generator. In Builder version 4.0 and later, Editor can be opened directly from within Builder so that strategies auto generated by Builder can be modified in Editor. Likewise, any strategy saved in Editor can be directly imported into Builder, and any strategy exported from Builder can be opened in Editor.

The complete contents of this user's guide are available in the program help files, which can be accessed at any time while using Editor by pressing the F1 key or by clicking any of the help buttons ("?") displayed throughout the program.

Please refer to the remaining chapters of this user's guide for additional details on each tab and feature of Editor.

## Strategy Features, Limitations and Program Scope

There are many possible ways to trade the markets and therefore many different types and styles of trading strategies. To understand the types of strategies that Adaptrade Editor was developed to help you create, please note the following:

1. Strategies can have one entry order (market, stop, or limit), multiple exit order types, and one exit (i.e., when the first exit order is hit, the other exit orders are cancelled). This means there is no pyramiding (scaling in) or scaling out.

2. Strategies trade one symbol at a time; i.e., no spread trading or inter-market strategies.
3. Strategies can be based on intraday or longer timeframes.
4. Multiple data series can be used. The primary data series is the one on which the trades are taken. Secondary series can be optionally used for logical conditions.
5. The indicators available in the program are described in the appendix. Not all indicators are available for all platforms; see the appendix for restrictions. If an indicator is used that is not available for the chosen scripting language, an error message will be displayed in the Messages window when the code type button is clicked. Custom indicators present in strategies exported from Adaptrade Builder will be processed correctly, but custom indicators cannot be added through Editor.
6. Indicator nesting is not available for MQL4 (MetaTrader 4) code. If a nested indicator is selected, an error message will be displayed in the Messages window when the code type button for MT4 is clicked.
7. For AFL (AmiBroker) code, strategies must be limited to either long-only or short-only trading logic. If both long and short entries are defined for a strategy, an error message will be displayed in the Messages window when the code type button for AmiBroker is clicked.
8. Because the program is based on point-and-click user interface elements, which necessarily constrain the number of choices, the options and features in the program do not include all possible features of the scripting language.
9. The program follows the common scripting language convention that all strategy logic is evaluated once per bar, on the close, at which time any necessary new orders are placed.
10. To implement proper order management and provide code for proprietary indicators, additional code needs to be installed into each platform; see Installation, below.
11. Adaptrade Editor is an editor only. It has no back-testing features (or access to price data). It is intended to help you create the strategy code with the assumption that the code is then copied to the trading platform for evaluation, parameter optimization, etc.

## Installation

**Recommended System Requirements:**

3 GHz or faster processor
8 GB or more RAM
64-bit Windows Vista or newer operating system
Monitor: 17 inch or larger

Adaptrade Editor can be downloaded by clicking the "Customer Login" link in your purchase receipt. Enter the login information provided in the receipt then click the Download button. The installation file for Editor will be named similarly to the following setup file for version 1.2.0: EditorLicensedSetup_v120.exe.

To install the program, browse to the location of the installation file via Windows Explorer and double-click the file to open and run it. Alternatively, select Run from the Accessories menu under the list of programs in the Start menu, browse to the location of the file, click Open, then click OK in the Run window. The installation should begin.

The installation program will prompt you for the location to install the program files. The default location is the folder Program Files\Adaptrade Software\Adaptrade Editor x.x.x\,

where x.x.x is the version number (e.g., 1.3.0). You can install the program in another location if you wish.

**Please be sure to save the installation file in a secure location, such as Dropbox or OneDrive, for future installations. If you choose not to renew your Editor license, you will be unable to download the installation file through your license portal after 12 months, but the saved file can be used indefinitely for future installations.**

**Note about 32 vs. 64-bit versions:** Adaptrade Editor is available only for 64-bit versions of Windows. If you're not sure which version of Windows you have, you can check the System settings under the Control Panel. **Editor should run on all recent releases of Windows from Windows 7 to the present.**

Once the installation is complete, you should find the Editor icon, as displayed on the title page of the user's guide, on your desktop and the Editor program in the Programs menu. You should also find a folder called **Examples** in the Adaptrade Editor folder. The Examples folder contains sample files that can be opened by Editor.

To activate Editor, enter the license ID provided in your purchase receipt and the activation password in the spaces provided on the activation screen, which will be displayed when the program is first run.

**Important:** While the licensed version of Editor and the trial version are functionally identical, the **trial version** only runs for the length of the trial period and **cannot be activated**. The licensed version of Editor, which is the version activated by your license ID and password, must be downloaded after purchase via the Download link in your online account, as described above.

**Platform Files:** Editor includes several functions and indicators that need to be installed into the trading platforms for Editor-written strategies to run properly. If these functions are not installed, the strategies generated by the program will not compile or run. These functions must be installed separately for each platform you intend to use for trading. **Please read the sections below for the platforms you intend to use.** To determine if the platform files have been updated for the current release and therefore need to be re-installed (if installed previously), please refer to the Release Notes page at Adaptrade.com or see the notes on the trial download page for the current release.

**Note: These files are the same as the ones required by Adaptrade Builder. If you've already installed these files as part of the installation process for Builder, it's not necessary to install them again.**

**TradeStation/MultiCharts Installation Notes.** Several files that work in conjunction with Editor need to be installed separately into TradeStation or MultiCharts. These files can be found in the EasyLanguage folder in the Adaptrade Editor folder under Program Files after Editor is installed. For current versions of TradeStation, the files end with the file extension .eld. For TS 2000i, the file extension is .els, and for MultiCharts, the extension is .pla. The files should be imported into TradeStation using the import command of the File menu. For MultiCharts, the files can be imported via the Import command of the File menu in the PowerLanguage Editor. Please refer to the platform documentation for instructions on importing .eld/.els/.pla files into TradeStation/MultiCharts.

The specific files that need to be imported into TradeStation/MultiCharts include NNCOMPUTE, which contains functions for use with the neural network option in Editor,

and BUILDER_INDICATORS_V3 (e.g., BUILDER_INDICATORS_V3-TS.ELD for current versions of TS), which contains the adaptive indicators (Adaptive VMA, Adaptive ZLT, etc.).

**NinjaTrader 7/8 Installation Notes.** Strategies written for either NinjaTrader 7 or NinjaTrader 8 require several indicator and strategy files to run. Without these files, the NinjaScript code will not compile or execute. These files are contained in a zip file named NinjaTrader.Adaptrade.zip or NinjaTrader8.Adaptrade.zip for NinjaTrader 7 or NinjaTrader 8, respectively, which can be found in the NinjaScript7 or NinjaScript8 folder in the Adaptrade Editor folder under Program Files after Editor is installed.

To install the files into NinjaTrader 7, open NinjaTrader 7, select Utilities from the File menu and select Import NinjaScript… Browse to the NinjaScript7 folder in the Adaptrade Editor installation folder and select the NinjaTrader.Adaptrade.zip file. Follow the prompts to install the files into NinjaTrader 7.

To install the files into NinjaTrader 8, open NinjaTrader 8, select Import from the Tools menu and select NinjaScript Add-On... Browse to the NinjaScript8 folder in the Adaptrade Editor installation folder and select the NinjaTrader8.Adaptrade.zip file. Follow the prompts to install the files into NinjaTrader 8.

**MetaTrader 4 Installation Notes.** MetaTrader 4 strategies created in Editor require files contained in three folders: Include, Indicators, and Libraries. These folders can be found in the MT4 folder in the Adaptrade Editor folder under Program Files after Editor is installed. The files in these folders provide basic functionality that any MetaTrader 4 strategy written in Editor may require. All the files contained in the three folders should be copied to the corresponding folders within your MetaTrader 4 installation so that MetaTrader 4 can find them when compiling strategies. For MT4 version 600 and newer, the correct folder locations can be found using the "Open Data Folder" command of the File menu in MT4. The general form of the folder path is C:\Users\User_account_name\AppData\Roaming\MetaQuotes\Terminal\Instance_id, in which User_account_name and Instance_id are specific to the user's computer. Within this folder, the Include, Indicators, and Libraries folders are in the MQL4 folder.

In versions of MT4 prior to version 600, the three folders are located in the installation folder in the experts folder (C:\Program Files (x86)\MetaTrader 4\experts\).

**AmiBroker Installation Notes.** AmiBroker strategies created in Editor require an associated "include" file: ASBuilderCommon.afl. This file can be found in the AFL folder in the Adaptrade Editor folder after Editor is installed. This file needs to be copied to the Include folder in the AmiBroker Formulas folder under Program Files. The full path under 64-bit Windows is C:\Program Files\AmiBroker\Formulas\Include. This file provides basic functionality that all AmiBroker strategies written in Editor require.

When installing a new version of Editor on a computer that currently has a prior version installed, please note the following:

- Since new versions of Editor install in folders that are specific to the version number, it should not be necessary to uninstall a prior version of Editor before installing a newer one. It's not necessary to uninstall the trial version prior to installing the licensed version of Editor.
- Provided the installation is on the same computer as the prior installation, no new activation code should be required. The new version should install already activated. If not, it may be necessary to enter your license ID and password, as provided on your purchase receipt. If you need an additional activation, please contact Adaptrade Software.

- Uninstalling an older version will not affect any strategy files (.gpcode files) you may have saved.
- New versions of Editor are designed to read files (.gpcode files) from prior versions. However, once a file is saved in the new version of Editor, you will not be able to open it in an older version. Use File -> Save As if you wish to retain both versions.

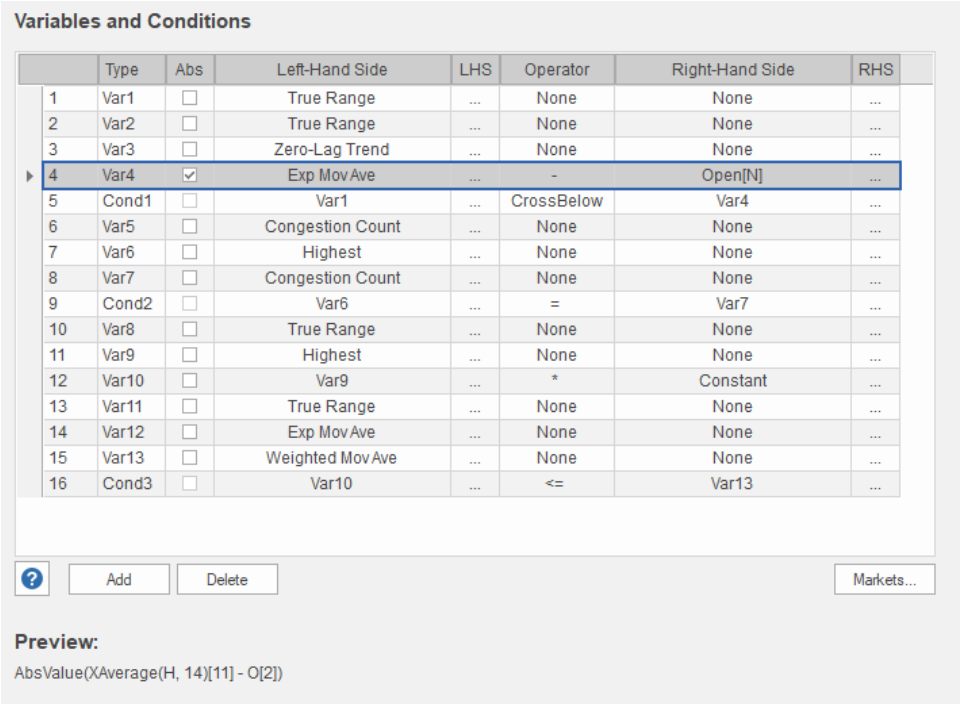
Editor can be uninstalled through the Windows Control Panel.

Please note that uninstalling and reinstalling Editor will not affect the pane layout and other user interface settings. These settings are stored in the Windows registry. Deleting the registry entry for Editor will reset the program to the same state as when it was initially installed. If this is necessary, it can be done using the Windows Regedit program. Type "regedit" into the search box in the Windows Start menu, browse to the location of the registry entry for Editor, and delete the folder SettingsXXX, where XXX is the version number (e.g., Settings100 for version 1.0.0). If necessary, the entire entry for Editor can be deleted. The registry entry for Editor is located at HKEY_CURRENT_USER\Software\Adaptrade Software\AdaptradeEditor\.

# Chapter 2
## Conditions Tab

### Variables and Conditions

The Conditions tab in Editor allows you to define the logical conditions for the strategy's entries and exits. This is a key element of the program and would typically be the starting point for writing a new strategy.




| | Type | Abs | Left-Hand Side | LHS | Operator | Right-Hand Side | RHS |
|---|---|---|---|---|---|---|---|
| 1 | Var1 | ☐ | True Range | ... | None | None | ... |
| 2 | Var2 | ☐ | True Range | ... | None | None | ... |
| 3 | Var3 | ☐ | Zero-Lag Trend | ... | None | None | ... |
| 4 | Var4 | ☑ | Exp Mov Ave | ... | - | Open[N] | ... |
| 5 | Cond1 | ☐ | Var1 | ... | CrossBelow | Var4 | ... |
| 6 | Var5 | ☐ | Congestion Count | ... | None | None | ... |
| 7 | Var6 | ☐ | Highest | ... | None | None | ... |
| 8 | Var7 | ☐ | Congestion Count | ... | None | None | ... |
| 9 | Cond2 | ☐ | Var6 | ... | = | Var7 | ... |
| 10 | Var8 | ☐ | True Range | ... | None | None | ... |
| 11 | Var9 | ☐ | Highest | ... | None | None | ... |
| 12 | Var10 | ☐ | Var9 | ... | * | Constant | ... |
| 13 | Var11 | ☐ | True Range | ... | None | None | ... |
| 14 | Var12 | ☐ | Exp Mov Ave | ... | None | None | ... |
| 15 | Var13 | ☐ | Weighted Mov Ave | ... | None | None | ... |
| 16 | Cond3 | ☐ | Var10 | ... | <= | Var13 | ... |

**Figure 2.1. Conditions tab in Adaptrade Editor. All variables and logical conditions are defined on this tab.**

Rows in the table are either variables (automatically labeled "Var1", "Var2", etc.) or conditions (automatically labeled "Cond1", "Cond2", etc.). You can select whether the item will be a variable or condition by clicking in the Type field and selecting from the pull-down menu (e.g., select either Var1 or Cond1).

Variables return a numeric value, such as a price or indicator value. Conditions return a logical value (i.e., either true or false). Variables can be selected as inputs to indicators in other variables or conditions. For example, if Var1 is defined as Average(C, 10), Var2 could be defined as XAverage(Var1, 15). Subsequently, a condition could be defined based on one or both of these variables, such as Var1 >= Var2 or High[2] < Var2.

After a new variable or condition is added to the table, Editor automatically adds a new, empty condition to the bottom of the table to facilitate the next entry. Initially, this entry will be colored yellow to indicate that it is invalid (since it's undefined). If you don't need the new entry, simply ignore it or select it and click the Delete button to remove it. In general, unused variables or conditions have no effect on the program or on the strategy. Only variables or

conditions used as entry or exit conditions or as inputs to the optional neural network affect the strategy.

## Specifying Variables and Conditions

As shown in Fig. 2.1, each variable or condition has the following general form:

LHS operator RHS

in which LHS is the indicator, variable, or condition selected for the left-hand side, RHS is the indicator, variable, or condition selected for the right-hand side, and operator is the logical or arithmetic operator relating the left and right-hand sides. The check box in the column labeled “Abs” takes the absolute value of the entire statement and only applies to variables.

The LHS and RHS items are selected by clicking the corresponding field, which brings up a pull-down menu listing all available indicators, variables, and conditions. For variables, the RHS and operator are optional; if not utilized, the corresponding fields can be left at the default selection of “None”. Because variables must return a numeric value, the operator for variables, if selected, is limited to -, +, or * (multiplication). Like the other items in the table, the operator is selected by clicking in the corresponding field and selecting from the drop-down menu.

Conditions relate the selected indicators, variables, or conditions on the left and right-hand sides using relational or logical operators, which consist of And, Or, =, != (not equal), <=, <, >=, >, crosses above, and crosses below. Examples of conditions include: Cond1 Or Cond2, Var1 != Var2, Highest(High, 10) <= Low[20], and RSI(Var1, 25) > 50.

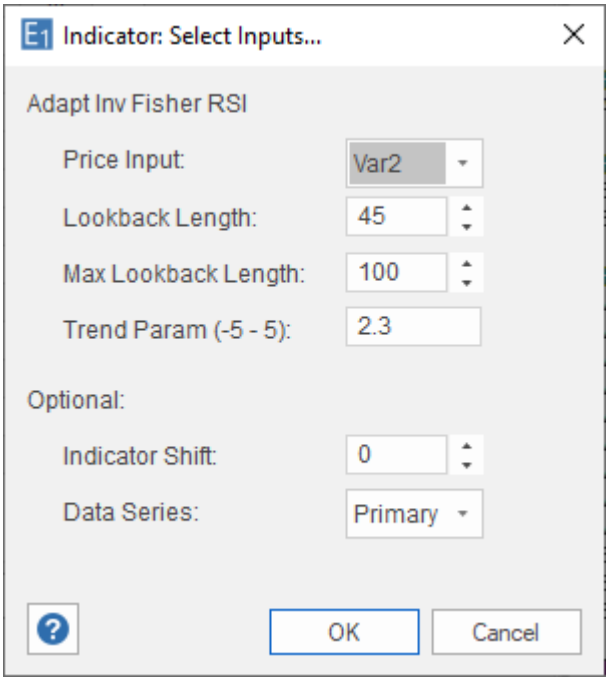



**Figure 2.2. Indicator input selection window, shown here for the Adaptive Inverse Fisher RSI indicator.**

If the selected indicator has inputs, an additional window allows you to select the indicator inputs. This window, shown in Fig. 2.2, is opened after the indicator is selected from the LHS or RHS field (Note: it's necessary to click elsewhere in the table to complete the selection of the LHS/RHS indicator. Doing so prompts the program to open the input selection window).

The available entries on the indicator input selection window depend on the indicator. Please refer to the appendix for a description of each indicator available in the program. Most indicators also include the optional inputs for the indicator shift and data series. The indicator shift is the number of bars back that the indicator references. For example, Average(C, 10)[2] is shifted 2 bars, which means it returns the value 2 bars back, rather than the average on the current bar. The data series can be either "Primary" or "Data2" through "Data20" and represents the symbol and timeframe on which the indicator is calculated. The primary series is the symbol on which trading takes place. Other data series can be selected as desired.

One of the possible indicators that can be selected for the LHS or RHS is "Constant", which represents a fixed numeric value. If this is selected, rather than opening the window shown in Fig. 2.2, the program opens a similar window prompting you to enter the value for the constant. For example, to enter the logical condition RSI(C, 15) < 50, the constant indicator would be selected for the RHS, and the value 50 would be entered via the "Constant Value" window.

## Defining the Data Series

Whenever a new data series is selected on the indicator input selection window, the program opens the Market Symbols window (Fig. 2.3), prompting you to enter the symbol details. This is necessary to provide the information required by the scripting languages that display the symbol name and/or bar size in the strategy code, such as NinjaScript. Other language options, such as EasyLanguage, don't need these details, so the default values on the Market Symbols window can be selected for these languages. Click the "Markets" button on the Conditions tab, as shown in Fig. 2.1, at any time to open the Market Symbols window and to edit the market symbols for the different data series.

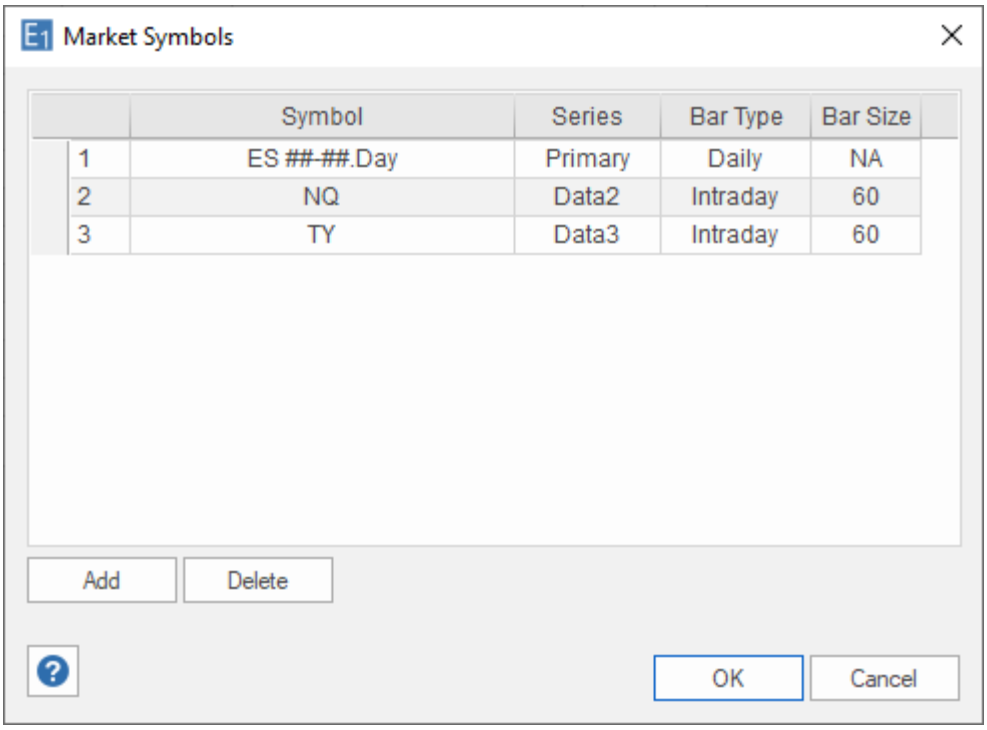



**Figure 2.3. Market Symbols window, showing three symbols selected as primary (ES ##-##.Day) and secondary (NQ for Data2; TY for Data3). The symbol name and bars size (if applicable) can be edited by typing directly into the corresponding fields. The Series and Bar Type fields can be changed by clicking and selecting from the drop-down menus.**

## Editing Variables and Conditions

The table columns labeled "LHS" and "RHS" contain fields consisting of ellipses ("…"). These buttons bring up the indicator input selection window, as shown in Fig. 2.2, for the indicator selected in the Left-Hand Side and Right-Hand Side columns, respectively. This enables you to edit the indicator without having to re-select it. For example, this is how you would change the value of an indicator input, such as changing the look-back length from 10 to 25. If the indicator is a constant, clicking the ellipsis button brings up the Constant Value window for editing the constant value.

## Adding and Deleting Variables and Conditions

To add a new variable or condition to the table, click the Add button below the table, or right-click on the table and select "Add Table Row". This adds an empty row to the table, as shown by the last row of the table in Fig. 2.4. Notice that the Type entry is colored yellow. This indicates that the condition is invalid, which, in this case, occurs because the condition has not been defined yet. A condition or variable can be invalid for several reasons, such as indicator types that are incompatible with the chosen operator. If you have an invalid condition or variable and don't understand what the problem is, right-click over the corresponding row in the table and select "Explain Errors", as shown in the figure. This will write out messages to the Messages window explaining the errors.

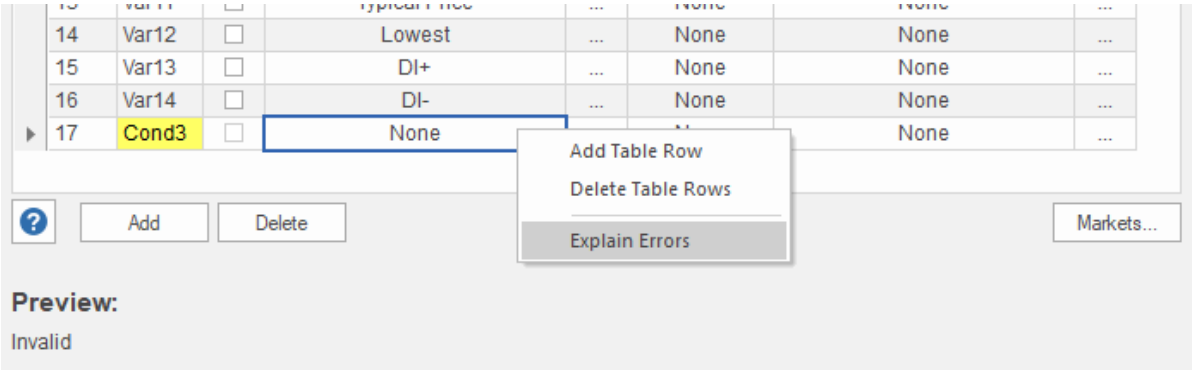



**Figure 2.4. An invalid row in the Conditions table is colored yellow. Selecting "Explain Errors" from the context menu displays an explanation in the Messages window.**

To delete one or more variables or conditions from the table, select them – drag the mouse to select multiple rows or use shift-click and/or Ctrl-click for multiple selections – and click the Delete button below the table, or right-click on the table and select "Delete Table Rows". The delete key can also be used. Note that if a variable or condition is used by another variable or condition in the table, a warning message will be displayed advising you that deleting the entry will invalidate the variable or condition that depends on it. Deleting a table entry cannot be undone.

The code for a variable or condition is previewed in the Preview area of the Conditions tab, as shown in Figs. 2.1 and 2.4. The preview code is written in the language currently selected in the Code Type menu of the Main menu. Select the row in the table to see the corresponding preview text. This can a convenient way to quickly see the indicator inputs for the variable or condition. As shown in Fig. 2.4, if a variable or condition is invalid, the word "Invalid" is displayed in the preview area.

# Chapter 3
## Orders Tab

### Trading Orders
The Orders tab in Adaptrade Editor is where entry and exit orders are defined. There are four tabs under Trading Orders, one for each type of order: Long Entry, Short Entry, Long Exit, and Short Exit.

### Entry Orders
The Long Entry order tab is shown in Fig. 3.1. To include long trades in your strategy, click the check box “Include long entry”. A long entry order will be placed when the condition selected in the drop-down menu for the “If” statement is true. The conditions available in the drop-down menu are the ones defined on the Conditions tab (see Chapter 2). If you want a long entry order to be placed regardless of any condition, select “None (“true”)” as the condition. This is the same as not having an entry condition; i.e., the entry condition is always true, so the entry order will always be placed on the next bar (subject to other restrictions).

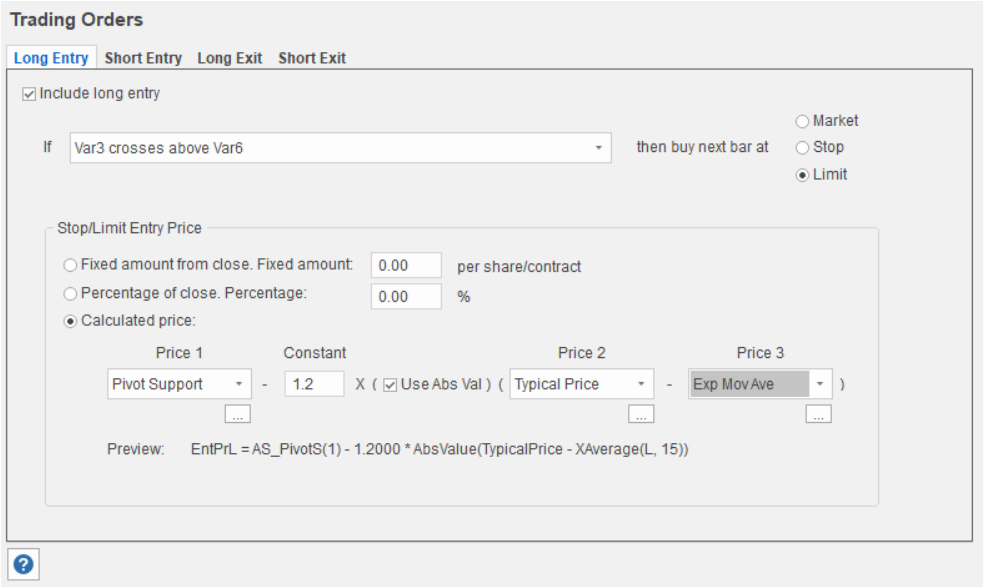



**Figure 3.1. Long entry orders are entered on the Long Entry tab of the Orders tab.**

The type of entry order is determined by the selection on the right side of the If statement: Market, Stop, or Limit order. Only one entry order per strategy can be defined for each side of the market (long and short). By definition, market orders are placed at the market price, which means the open of the next bar, so no further specification is necessary. If you select either Stop or Limit as the order type, the section “Stop/Limit Entry Price” allows you to specify the price at which the stop or limit order will be placed.

## Specifying the Entry Price

There are three options for specifying the entry price for stop and limit orders. The option to use a “Fixed amount from close” calculates the entry price as a fixed-size deviation from the current bar's close using the entered currency value (e.g., dollars for accounts denominated in dollars) per share or contract. The entered value is divided by the point value for the symbol to obtain the number of points. For example, a fixed value of $500 means the entry price is calculated to be $500 from the current bar's close per share/contract. For the E-mini S&P futures, for example, this means an entry stop would be 10 points above the close for a long trade because each point is worth $50 for the E-mini.

The “Percentage of close” option sets the entry price at a percentage of the current bar's close above (below) the current bar's close for a long (short) trade. For example, if the close is 25, a 5% limit entry would be placed 1.25 points below the close for a long trade.

The third option calculates the entry price from the entered prices, indicators, and other settings, as shown in Fig. 3.1. The basic form of the calculated price is a price (“Price1” in Fig. 3.1) plus or minus a multiple of a price difference, which itself can be either the average true range or the difference between two specified prices (“Price2”, “Price3”). There’s also a check box to apply an optional absolute value to the price difference.

Optionally, the price difference can be excluded by leaving Price2 and Price3 set to "None" and the constant set to zero. In this case, the entry price will be set to Price1. If both Price2 and Price3 are "None" and the constant is non-zero, the entry price will be set to Price1 +/- the constant.

The prices (Price1, Price2, Price3) are selected from drop-down menus. As in the Variables and Conditions table, selecting an indicator that has inputs prompts the program to open an indicator input selection window for entering the required indicator input values. The ellipsis buttons (“…”) below each of the three price drop-down menus bring up the indicator input selection window so that the input values can be changed if desired. If either true range or average true range is chosen as Price2, then Price3 should be set to “None”, in which case it will be ignored.

The following are examples of long stop entry prices:

```
EntryPrice = Average(C, 10) + 3.5 * AbsValue(C[5] – H[14])

EntryPrice = H + 2.1 * AbsValue(Average(C,20) – Lowest(H,15))

EntryPrice = Highest(C, 15) + 35

EntryPrice = BollingerBand(C, 25, 1.2)
```

These could also be short limit entries since short limit entries are also above the market and therefore use a “+” sign to add the price difference to the price value.

Examples of short stop entry prices are shown below:

```
EntryPrice = OpenD(0) – 1.7 * AvgTrueRange(11)

EntryPrice = C[3] – 4.3*AbsValue(XAverage(L,5)– Xaverage(C,2))

EntryPrice = Average(C, 5) – 74
```

```
EntryPrice = Lowest(L, 35)
```

These could also be long limit entries since long limit entries are also below the market and therefore use a "-" sign to subtract the price difference from the price value.

A preview area below where the calculated price is specified displays the calculated price in the scripting language currently selected in the main menu.

The Short Entry tab works the same way as described above.

## Exit Orders

The Long Exit order tab is shown in Fig. 3.2. Multiple exit orders can be selected using the check boxes. There are two types of stop orders: protective (money management) and trailing stops. If both are selected, the stop order will be placed at the price closest to the market. A price target ("Target exit") can be placed as a limit order. Other than the end-of-day exit, which is not available for all code types, the remaining order types are market orders, which means they'll be placed at the open of the next bar.

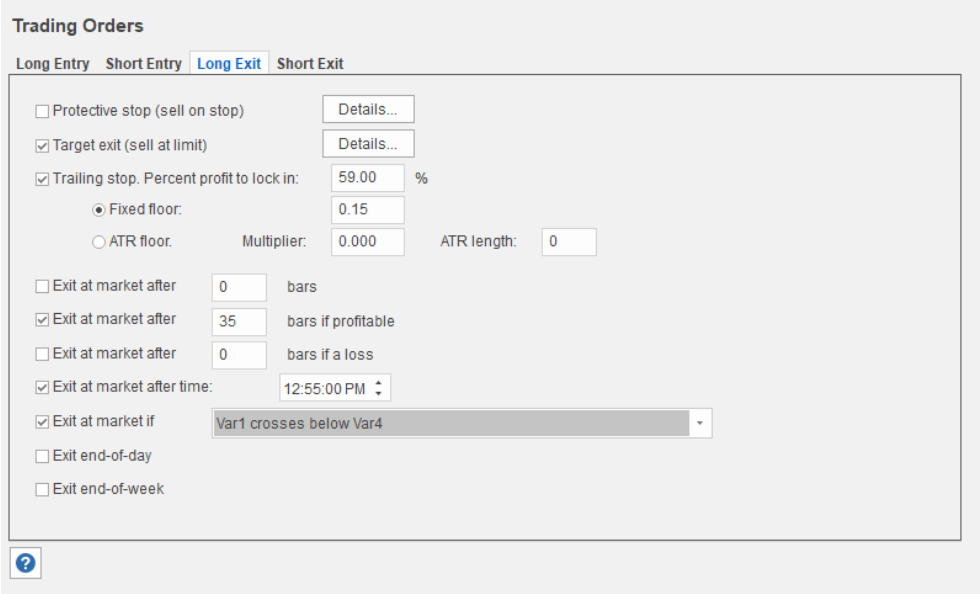



**Figure 3.2. Long exit orders are entered on the Long Exit tab of the Orders tab.**

If the protective stop or target exit order types are selected, the specifications are entered by clicking the "Details" buttons, which opens an additional window, as shown in Fig. 3.3 for protective stops.

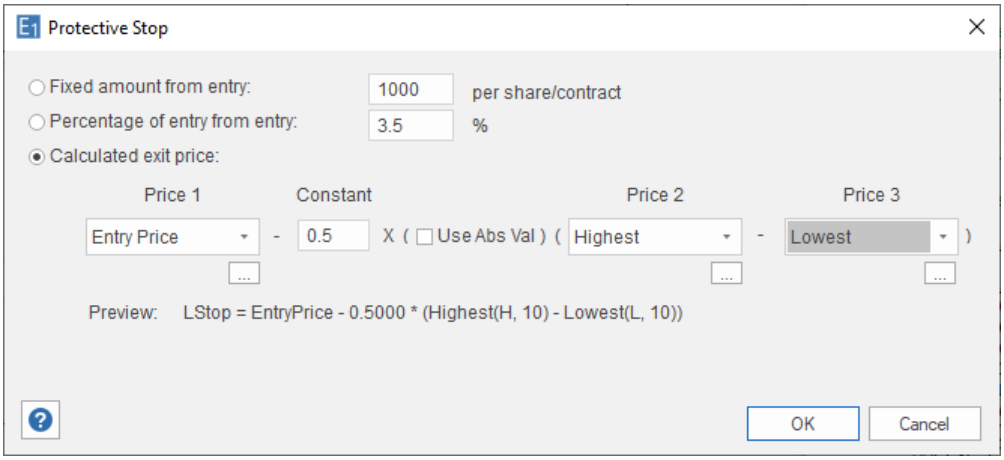



**Figure 3.3. Protective stop orders are specified by clicking the Details button on the exit order tab, which opens this window. A similar window applies to target exit orders.**

The entries on this window are very similar to those for the calculated price of entry orders. Similar to entry prices, protective stops and targets can be calculated based on a fixed amount per share or contract from the entry price, as a percentage of the entry price from the entry, or as a calculated price.

For a stock trade, for example, a fixed-size $2 stop would be placed two points below the entry price for a long trade. Percentage protective stops are often used for stock trading. For example, if the entry price is 25, a 5% protective stop would be placed 1.25 points below the entry for a long trade. The price difference option is calculated as described above for entry orders. The only difference is that Price1 can be optionally set to the “EntryPrice”, in which case the remaining terms (Constant, Price2, and Price3) define the price amount that subtracts from or adds to the entry price to determine the stop or target price.

**Trailing Stops**

Trailing stops are activated when the open profit on a closed-bar basis is above a threshold called the *floor*. There are two types of trailing stops in Editor: ones with fixed-size floors, and ones where the floor is calculated as a multiple of the average true range (ATR floor). For fixed-size floors, enter the size of the floor per contract or share. For ATR-based floors, enter the multiple of the ATR and the ATR averaging period. Once the threshold has been reached, the trailing stop is placed so that the entered percentage of the open profit is locked in. The stop remains active until the trade exits.

**Exiting After N Bars**

The “Exit at market after N bars” order type causes the trade to exit at the open of the next bar when the number of bars since entry is greater than or equal to the entered number of bars. The "Exit at market after N bars if profitable" and "Exit at market after N bars if a loss" order types work similarly. With the "Exit at market after N bars if profitable" order type, the trade is exited at the open of the next bar if the number of bars since entry is greater than or equal to the entered number of bars *and* the close of the bar is greater (less) than the entry price for a long (short) trade; i.e., if the trade is profitable before costs.

Likewise, with the "Exit at market after N bars if a loss" order type, the trade is exited at the open of the next bar if the number of bars since entry is greater than or equal to the entered number of bars *and* the close of the bar is less (greater) than the entry price for a long (short) trade; i.e., if the trade is unprofitable before costs.

**Time Exit**
The "Exit at market after time" order type causes the trade to exit at the open of the next bar when the time of the current bar is greater than or equal to the selected time. For example, the exit order might read "If time >= 1030 then sell next bar at market". Note that this implies the time stamp for the trade exit will be the time of the bar following the bar where the exit is triggered. In the preceding example, if the bars are 30-minute bars, the exit time would be shown as 11:00. *Note that this exit type may not work as expected if the trading session spans two days. For example, if the exit time is 1:00 pm and the session starts at 5:00 pm and ends as 4:00 pm the next day, the trade will exit as soon as the session starts because 5:00 pm is already later than the specified exit time of 1:00 pm.*

**Condition Exit**
The "Exit at market if" order type causes the trade to exit at the open of the next bar when the selected exit condition is true on the current bar. The condition pull-down menu displays the conditions defined on the Conditions tab (see Chapter 2).

**End-of-Day Exit**
The "Exit end-of-day" order type causes the trade to exit at the close of the last bar of the current day if intraday or daily bars are used. On weekly or monthly data, this exit causes the trade to exit at the close of the current bar. *For TradeStation and MultiCharts, this exit type is primarily for back-testing purposes. It can be problematic because the market will be closed by the time the order is sent, which means it may be placed in the after-hours market as a limit order and therefore may not be filled. To achieve end-of-day exits in real-time trading on intraday data in any of the supported languages, the "Exit at market after time" order type is recommended. The End-of-Day exit type is not available for MetaTrader 4 or AmiBroker. For NinjaTrader, this exit only applies to intraday data.*

**End-of-Week Exit**
The "Exit end-of-week" order type is essentially the end-of-day exit applied on a Friday. It causes the trade to exit at the close on Friday. This order type is only available for EasyLanguage code (TradeStation/MultiCharts), which is the only supported language that includes an end-of-session order type ("SetExitOnClose"), other than NinjaTrader, which does not allow the end-of-session exit to be restricted to a specific day. Please note that the exit occurs only on a Friday, so if there are no bars on Friday, there will be no exit that week via this order type. *As with the end-of-day exit, this exit type can be problematic because the market will be closed by the time the order is sent, which means it may be placed in the after-hours market as a limit order and therefore may not be filled. As an alternative to achieve end-of-week exits in real-time trading on intraday data in any of the supported languages, the optional time-based exit on the Options tab can be used with the optional day-of-week restriction (see Chapter 4).*

# Chapter 4
## Options Tab

### Strategy Logic Options

Several strategy logic options can be selected from the table on the left side of the Options tab, as shown in Fig. 4.1. After changing an option, click one of the code type buttons in the main menu to see the change reflected in the strategy code of the code window.

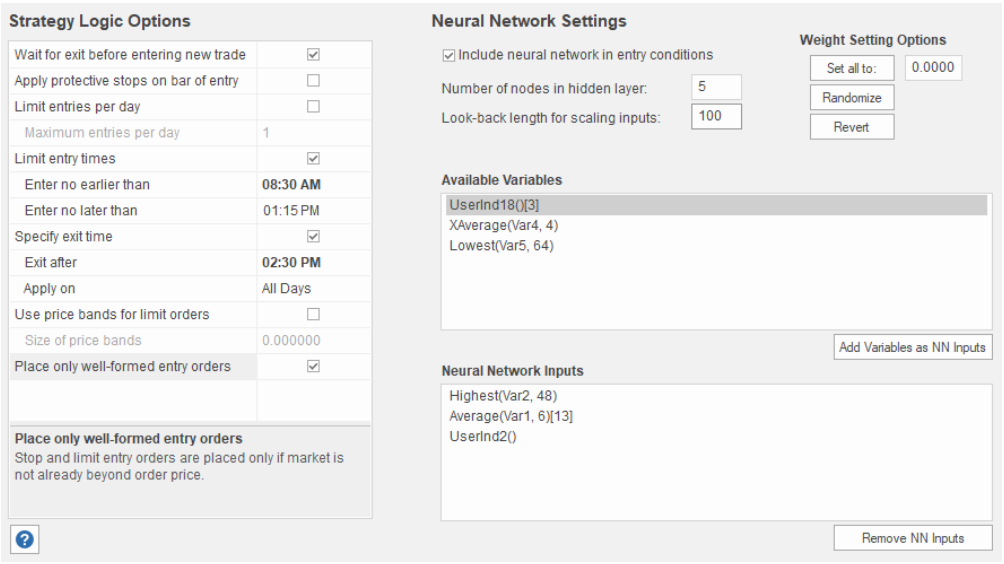



**Figure 4.1. The Options tab contains strategy logic options and settings for the optional neural network. After making selections on this window, click one of the code type buttons in the main menu to see the changes to the strategy code.**

**Wait for exit before entering new trade**

The option to wait for an exit before entering a new trade is checked by default. This adds an entry condition that only allows an entry if the current position is flat. If this option is unchecked, a long entry may reverse a short position and vice-versa. Uncheck this option to create "stop and reverse" strategies that are always in the market, either long or short.

**Apply protective stops on bar of entry**

The option to apply protective stops on the entry bar immediately applies the protective stop order upon trade entry. This means the protective stop will be applied on the bar of entry, rather than waiting until the close of the bar to submit the order. In TradeStation, selecting this option applies the protective stop order using the SetStopLoss EasyLanguage command. This option only applies to existing protective stop orders; it does not add a protective stop if one does not already exist. Also, because this option must accommodate the SetStopLoss command in EasyLanguage, it uses the same size stop for both long and short trades.

**Limit entries per day**

For intraday strategies, you can specify the maximum number of trade entries per day by checking the box **Limit entries per day** and entering the desired maximum number of daily

entries in the box. Limiting the number of entries per day can often improve results for intraday strategies because it reduces the number of different market conditions that must be accommodated.

**Note:** The strategy code checks to make sure the number of entries on the day is less than the specified limit before placing the orders. However, it can still place two orders – one long and one short – if both entry conditions apply, either of which or both may be filled, depending on the market. That means you could get up to two entries if both are filled, which could cause the actual number of entries to exceed the specified limit.

**Limit entry times**
Entry times for intraday trades can be restricted to a time range using the **Limit entry times** option. Enter the starting and ending times using the time selectors. This will restrict entries to the time range chosen. The exit time must be later than the entry time. If the trading session spans two days, this implies the time range must be restricted to the same day.

**Specify exit time**
To force trades to exit by a certain time, check the box and enter the desired exit time in the **Exit after** option. If an intraday trade is open at that time, it will be closed at market on the next bar. You can also specify the day of the week on which the exit-after option will apply using the drop-down menu next to “Apply on”. For example, select "Friday" to exit after the specified time only on Fridays. To apply the option on every day, select the option "All Days", which is the default.

**Note:** It's important to keep in mind that the time stamp on bars is typically the time the bar closes. For example, if you're using 30-minute bars, a bar time of 3:30 pm means the bar closes at 3:30. Since it's a 30-minute bar, that means it opens at 3:00 pm. The exit time option causes a trade to exit on the next bar's open if the time of the current bar is greater than or equal to the exit time. So, if you've specified an exit time of 3:00 pm, the trade will exit on the open of the 3:30 pm bar (assuming 30-minute bars). The exit time will be listed as 3:30 pm because that's the time stamp on the bar, but the actual time will be a moment after 3:00 pm, which corresponds to the open of the 3:30 pm bar.

Also keep in mind that a trading strategy can only trade the bars of price data it has. If you're using 60-minute bars that end on even hours (e.g., 2:00 pm, 3:00 pm, 4:00 pm, etc.), and you specify an exit time of 3:15 pm, your trades will not end at 3:15 pm because there's no bar with that time. In this case, the trade would exit on the open of the 5:00 pm bar, which would be a moment after 4:00 pm.

**Please note:** The **Specify exit time** option uses the "Exit at market after time" order type, as shown on the Long Exit and Short Exit tabs of the Orders tab. Consequently, selecting this option also selects the "Exit at market after time" order type for both long and short trades. See the previous chapter for additional information on this order type.

**Use price bands for limit orders**
When trading futures contracts through the Chicago Mercantile Exchange (CME), limit orders may be subject to price banding. This means that the limit order will only be accepted by the exchange if the order price is within a band size of where the market is currently trading. The option to use price bands for limit orders means that the limit order will only be placed if the order price is within a band size of the current bar’s close. To use this option, check the box and enter a value for the size of the price band in points (e.g., 6 points for the E-mini S&P 500 futures).

**Place only well-formed entry orders**
This option only places a stop or limit entry order if the market has not moved beyond the order price. For long stop and short limit orders, this means the market must be below the order price at the time the order is placed (i.e., on the close of the prior bar). For short stop and long limit orders, the market must be above the order price on the close of the prior bar. If this option is not checked, the stop or limit entry order will be placed regardless of where the market is trading relative to the order price. In this case, if the market is beyond the order price, the order will be automatically converted into a market order.

## Neural Network Settings

Neural networks are nonlinear models loosely based on the way the brain is structured as a network of interconnected neurons. A neural network is often depicted graphically as a set of nodes connecting an input layer through one or more "hidden" layers of nodes to one or more outputs. The inputs are summed together using a set of weights that connect the nodes to produce the output. The weights are determined by the fitting or *training* process. A neural network with a single output and one hidden layer is shown below in Fig. 4.2.

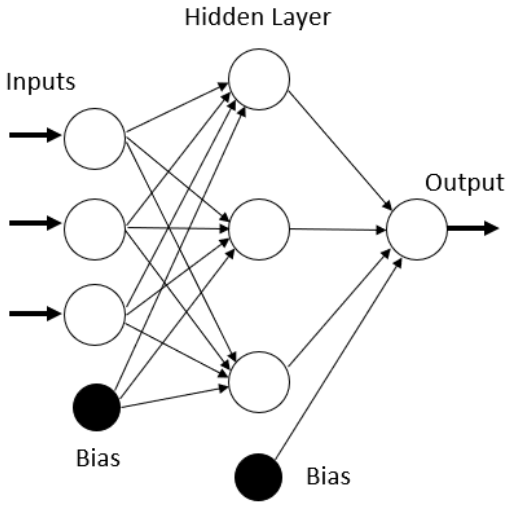



**Figure 4.2. A neural network with a single output and one hidden layer. It includes three input nodes, three nodes in the hidden layer, and two bias nodes.**

The output from the network is defined by the following equations:

$$H_1 = \text{ReLU}(w_{11} * I_1 + w_{21} * I_2 + \ldots + w_{n1} * I_n + B_1)$$

$$H_2 = \text{ReLU}\ (w_{12} * I_1 + w_{22} * I_2 + \ldots + w_{n2} * I_n + B_2)$$

$$\ldots$$

$$H_m = \text{ReLU}\ (w_{1m} * I_1 + w_{2m} * I_2 + \ldots + w_{nm} * I_n + B_m)$$

$$\text{Output} = \tanh(w_{nm+1} * H_1 + w_{nm+2} * H_2 + \ldots + w_{nm+m} * H_m + B_o)$$

The $I_i$ represent the inputs. In the context of trading, these could be anything that might have some predictive value for trading, such as momentum, stochastics, ADX, moving averages, etc. The $H_j$ represent the hidden nodes, the weights are given by the $w_{kl}$, $B_j$ are the weights of the bias nodes for the hidden layer, $B_o$ is the weight of the bias node for the output layer, and the output value is given by Output. Bias nodes always have a value of 1 and so are

represented by their respective weight values only. ReLU is the so-called rectified linear unit, which is defined by the function f(x) = max(0, x). The hyperbolic tangent function, tanh, returns a value in the range -1 to +1, so the output will lie in this range. In Editor, the inputs are scaled so that they also lie between -1 and +1.

To add a neural network to the trading strategy, check the box on the Options tab “Include neural network in entry conditions”. Then enter the number of nodes for the hidden layer and the look-back length for scaling the inputs (e.g., 100 bars).

The inputs to the network are chosen from the variables defined on the Conditions tab and displayed in the “Available Variables” list of the Options tab, as shown in Fig. 4.1. To select inputs for the neural network, select them from the list and click the button “Add Variables as NN Inputs” or right-click on the list and select “Add Variables”. This will add the selected variables to the “Neural Network Inputs” list (see Fig. 4.1). Similarly, variables can be removed from the “Neural Network Inputs” list using the “Remove NN Inputs” button, by right-clicking and selecting “Remove Variables” or using the Delete key.

Because training the neural network to determine the weight values would involve backtesting the strategy on market data, the training process must be conducted outside of Editor, such as using the built-in input optimization feature of your trading platform. As a placeholder, the weight values can be set in Editor using the buttons shown in Fig. 4.1. Use the “Set all to” button to set all weight values to the entered value or use the “Randomize” button to set the weight values randomly. To revert to the original weight values, click the “Revert” button.

The total number of weights in the network will be given by (n + 2) * m + 1, where n is the number of inputs and m is the number of nodes in the hidden layer, provided m is at least one. If there is no hidden layer (i.e., m is zero), the number of weights is the same as the number of inputs plus 1 (for the bias). In the strategy code, each weight is listed as a strategy input. Other strategy inputs may result from the network inputs, such as the look-back length of a moving average.

In the strategy code, the neural network is used to filter trade entries. The output of the network allows for a long entry if it's greater than or equal to 0.5 and for a short entry if it's less than or equal to -0.5. This condition is in addition to any existing entry conditions selected on the Orders tab. For example, if there is a long entry condition, it must be true and the neural network output must be at least 0.5 for a long entry.

# Chapter 5

## Position Sizing Tab

Position sizing determines how many shares or contracts are traded on each trade. The position sizing settings are made on the Position Sizing tab, as shown in Fig. 5.1. After making a change to the settings, click one of the Code Type buttons in the main menu to update the strategy code displayed in the code window.

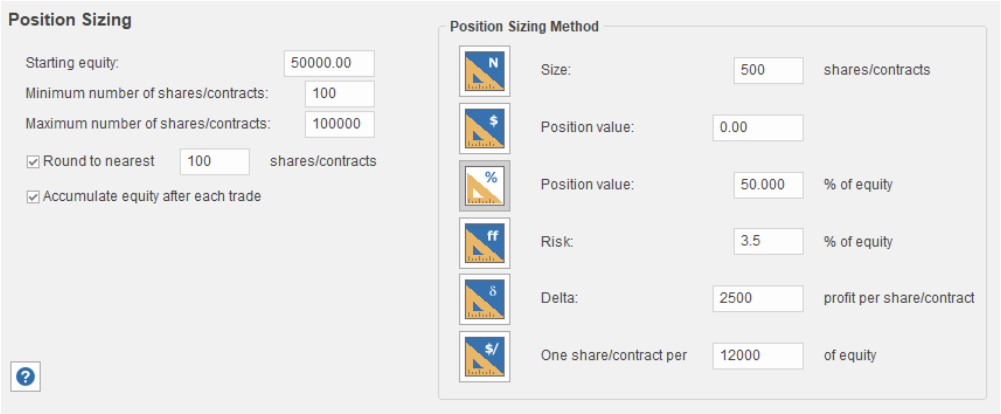



**Figure 5.1. The Position Sizing tab allows you make settings and select options that determine how many shares or contracts are traded.**

### Basic Settings

Most of the position sizing methods depend on the value of account equity being traded. The "Starting equity" setting represents the value of the account equity at the start of a strategy back-test. Depending on whether the option to "Accumulate equity after each trade" is checked or not (see below), the simulated account equity is either accrued during the back-test or set equal to the starting equity for each trade.

As shown in Fig. 5.1, you can enter the minimum and maximum number of shares or contracts that the strategy will trade by entering values for the "Minimum number of shares/contracts" and "Maximum number of shares/contracts". The minimum value, which defaults to 1, is applied to all trades. After the position size is calculated, the position size is increased to the minimum if the position size would otherwise fall below the minimum. Likewise, the maximum, which defaults to 100, is applied to all trades. After the position size is calculated, the position size is reduced to the maximum if the position size would otherwise exceed the maximum.

The "Round to nearest" option rounds the position size down to the nearest number of shares or contracts. For example, if you enter a value of 100, the number of shares will be rounded down to the nearest 100 shares. The default value is 1. For forex trading, this can be used to trade in even lot sizes of 10,000 or 100,000 by setting the rounding value to 10,000 or 100,000 for mini and full-size lots, respectively. In this case, the minimum number of shares should be set to the same value and the maximum value to some multiple of the minimum.

There are two options for how to calculate the account equity value used for position sizing and back-testing. If the option "Accumulate equity after each trade" is checked, the strategy

code will track the account equity during the back-test from trade to trade, starting with the first date in the trading period, where the account equity is equal to the starting equity value, and accumulate the profits and losses as trades close. As the equity changes, the position size will generally change as well. In all position sizing methods except for fixed size and constant value, the position size increases as the account equity rises and decreases when the account equity drops.

If the option is unchecked, the account equity for every trade will simply be the starting equity value. In this case, the profits and losses will not be accumulated from trade to trade. This option is useful when you want to see what the position size would be for a specific value of account equity, such as when executing the trading strategy in real time. In this case, you could enter the current value of your account equity as the starting equity value, and the strategy would generate the correct position size for the entered account value.

## Position Sizing Methods

Editor includes six different position sizing methods. Each method has an associated parameter value, which is entered in the edit box to right of the button. Hovering the mouse cursor over the button displays a “tooltip” with information about the associated method.

**Fixed Size.** This method simply uses the value you enter as the number of shares or contracts for each trade. A fixed size of 1, for example, will trade one contract or share for each trade. For forex trading, a typical fixed size would be either 10000 or 100000. It's important to set the upper size limit to a value at least as big as the fixed size entered here.

**Constant Value.** With constant value position sizing, each position is sized so that it has a specified value, such as $1000 per trade. This method can be used when you want to allocate a specified amount of equity to each trade. The constant value method will determine the number of shares or contracts corresponding to your specified amount. For example, if you plan to purchase a stock at a price of $25 and you want to spend $35,000, you will trade 35000/25 or 1400 shares. In this case, the position value is 35000; each trade will be sized so that the position value is $35,000.

**Percent of Equity.** In this method, the number of shares or contracts is chosen so that the value of the position is equal to the selected percent of account equity. For example, if the percent of equity is 40%, the position size will have a value equal to 40% of the account equity. If the account equity were $30,000, the position would have a value of 0.4 x 30000 or $12,000. If the share price were $25, the position size would be 12000/25 or 480 shares. In other words, a position size of 480 shares at $25 per share has a value of $12,000, which is 40% of the equity value of $30,000.

**Fixed Fractional.** In fixed fractional position sizing, the number of shares or contracts is based on the risk of the trade. For example, you might risk 2% of account equity on each trade. Fixed fractional position sizing has been written about extensively by Ralph Vince. See, for example, his book "Portfolio Management Formulas," John Wiley & Sons, New York, 1990.

The risk of a trade is defined as the dollar amount that the trade would lose per contract or share if it were a loss. In strategies created in Editor, the trade risk is taken as the size of the money management stop applied, if any, to each trade. If the strategy doesn’t use protective (money management) stops, the trade risk will be taken as the largest historical loss per contract or share. The largest loss is the largest loss up until the point at which the trade is taken. If subsequent losses are larger, the trade risk will be larger for subsequent trades.

As an example, consider a stock trading system that uses a 2-point stop. It might enter long at a price of 48 with a sell stop at 46. The risk per share would be $2. With a fixed fraction of 5% and account equity of $20,000, fixed fractional position sizing would risk 0.05 * 20000 or $1,000 on the next trade. Since the risk per share is $2, this means 1000/2 or 500 shares would be traded.

**Fixed Ratio.** In fixed ratio position sizing the key parameter is the delta. This is the amount of profit per share or contract to increase the number of shares or contracts by one. A delta of $3,000, for example, means that if you're currently trading one contract, you would need to increase your account equity by $3,000 to start trading two contracts. Once you get to two contracts, you would need an additional profit of $6,000 to start trading three contracts. At three contracts, you would need an additional profit of $9,000 to start trading four contracts, and so on. Fixed ratio position sizing was developed by Ryan Jones in his book "The Trading Game," John Wiley & Sons, New York, 1999.

Normally, the delta is the amount per share or contract, as in the example above. For some instruments, such as forex, this would be inconvenient because forex uses a point value of 1 and typically trades in increments of 10,000 or 100,000 shares. To accommodate this, the delta is specified per multiples of shares when the option to round the share size to the nearest number of shares (see above) is selected. For example, if the "round to nearest" option is selected with an increment of 10,000 shares (i.e., "Round to nearest 10,000 shares"), then the delta would be specified as the amount per 10,000 shares.

The fixed ratio method will start trading with a position size equal to the minimum number of shares/contracts specified (see above). For example, if the minimum number of shares for a forex strategy is set to 10,000, this will be the position size calculated by the fixed ratio method for a profit of zero. As profits are accrued, the position size will increase, based on the delta, in increments specified by the "round to nearest" option.

**Fixed Amount of Equity.** In this position sizing method, you choose the amount of account equity required to trade each share or contract. For example, if the dollar amount is $5,000, you would trade one contract for every $5,000 in the account. If the account equity is currently $50,000, the position size would be 10 contracts for the next trade.

# Chapter 6

## Output Windows

### Strategy Code Window

The code for the strategy is displayed in the Strategy Code window, as shown on the right side of Fig. 1.1. If desired, the location of the window can be moved by clicking the title bar and dragging the mouse to one of the other anchor locations, such as to the left of the tab windows.

The first part of the code contains a comment block, colored green, that lists key descriptors of the code, including the max bars back, creation date, scripting language, symbols, and strategy file name. In general, code comments are colored green, statements beginning with a hashtag character (#, so-called compiler directives) are colored blue, and other code elements are colored black.

The code in the Strategy Code window can be copied to the clipboard by right-clicking and selecting “Copy Strategy”. To copy only part of the strategy code, click and drag the mouse over the part you want to copy, then right-click and select “Copy” or use the Ctrl-C keyboard combination to copy the selected text to the clipboard.

If the code is in MetaTrader 4 code format, it can be saved directly to a .mq4 file by right-clicking in the code window and selecting "Save MT4 Strategy to File". Similarly, select "Save NinjaScript Strategy to File" to save the NinjaTrader 7/8 strategy code directly to a file in the required folder for NinjaTrader 7 or NinjaTrader 8.

Once selected, the folder location for NinjaTrader strategies is stored in the registry, so it only needs to be selected once. Editor initially defaults to the folders Documents\NinjaTrader 7\bin\Custom\Strategy\ and Documents\NinjaTrader 8\bin\Custom\Strategies\, which are the customary locations for NinjaTrader 7 and NinjaTrader 8, respectively. Editor creates a default strategy name for each NinjaTrader strategy of the form strategyname_member#_#inputs_maxbarsback. For example, for a strategy with the file name "MyStrategy.gpcode", a typical strategy name might be MyStrategy_3_7_72, which indicates the strategy is from *population member** 3 with 7 inputs and a MaxBarsBack requirement of 72. When saving the file, a new name can be entered, which will become the name for both the file and the strategy.

* The “population member” number refers to Adaptrade Builder, which identifies strategies by their number in a population of strategies. Strategies exported from Builder will appear in Editor with the population number generated by Builder. New strategies created in Editor will have a population number of zero (0). The default name containing the population number can be overwritten as desired when saving the strategy.

**Note:** The options to save the strategy code directly to a file for either MetaTrader 4 or NinjaTrader are only available if the code is the respective format. Otherwise, the option is grayed out. To save it in either format, change the code to the desired format by clicking the Code Type button in the main menu.

The strategy code cannot be edited directly in the strategy code window. All code changes must be made in the tabbed windows, as explained in previous chapters.

To transfer the code to your trading platform (if not saving the code directly to a MT4 or NinjaTrader file), copy the code from the Strategy Code window and paste it into the strategy editor of the trading platform. In TradeStation or MultiCharts, open a new strategy window in the EasyLanguage editor, choose a name, then paste the code into the empty strategy window. Finally, compile the EasyLanguage code by pressing F3.

In MetaTrader 4, open a new window in the MetaEditor, paste in the code from Editor, and click the compile button. In AmiBroker, in the Charts pane, right-click on the Systems folder and select "New" and then "Formula". Enter a name then right-click on the name and select "Edit". Select "Paste" from the Edit menu to paste the code into the edit window, then click the save icon and, finally, click the AFL icon to verify the code.

If you're pasting the strategy code over the code for an existing strategy, it will be necessary to reset the input values before running the back-test. In MetaTrader 4, this can be done by clicking the Expert properties button on the Tester window and clicking the Reset button on the Inputs tab. In TradeStation, delete the strategy from the chart and re-insert it to reset the input values. In AmiBroker, click the Parameters icon on the Analysis window and click "Reset all".

To test the strategy in TradeStation/MultiCharts, insert it into a chart window and set the ”Maximum number of bars study will reference” (in TradeStation, Format Strategies, Properties for All; in MultiCharts, Format Signal, Properties) to the “Max bars back” value listed in the comment block at the top of the code.

To test the strategy in NinjaTrader 7, first open the strategy in the NinjaTrader editor by selecting Edit NinjaScript from the Tools menu and click the Compile button. The strategy should then be ready to back-test via the Strategy Analyzer window. Select the strategy on the Backtest tab of the Strategy Analyzer window and click the button "Run Backtest".

In NinjaTrader 8, provided the strategy has already been saved to the NinjTrader 8 Strategies folder, the strategy should already be compiled. Open a Strategy Analyzer window and select the strategy from the Settings window. Verify the settings and click the Run button.

To test the strategy in MetaTrader 4, select the strategy (Expert Advisor) on the Tester window (Settings tab), and select the symbol and date range. For best results in MetaTrader, select "Every tick" as the Model, then click the Start button to run the back-test.

To test the strategy in AmiBroker, right-click on the strategy in the Systems folder of the Charts pane and select "Analysis". On the Analysis window, reset the parameter values by clicking on the Parameters icon, and check the settings by clicking on the Settings icon. Finally, click the Backtest icon to run the test.

## Messages Window

The Messages window displays messages generated by the program. The messages are of three basic types: error messages related to the definition of variables and conditions, code warning and error messages, and general informational messages. All messages displayed in the Messages window are preceded by a date and time stamp, followed by a label, such as “Code Error” or “Error in Condition1”. For example,

[8/5/2021 10:38:59 AM] -- Code Error -- End-of-week exit not allowed in NinjaTrader 8 code.

and

[8/5/2021 11:08:26 AM] -- Error in Condition3 -- Left-hand side undefined. Left-hand side must not be "None".

To clear (erase) the Messages window, right-click and select “Clear”. You can also select any line in the window (click and drag the mouse cursor to select) and copy it to the clipboard by right-clicking and selecting “Copy” or by using the Ctrl-C keyboard combination.

#### Variable and Condition Error Messages

As explained in Chapter 2, if a variable or condition has a recognized error, the corresponding line in the Variables and Conditions table will be colored yellow to indicate a potential problem. If you right-click the entry and select “Explain Errors”, one or more messages will be displayed in the Messages window. Examples of these types of error messages include:

- Left-hand side undefined. Left-hand side must not be "None".
- Left-hand side based on invalid condition.
- Input to indicator of left-hand side undefined. Must not be "None".
- Variable chosen as input to indicator of left-hand side is invalid.
- Indicator chosen as input to indicator of left-hand side is undefined. Must not be "None".
- Return type of right-hand side (Day of Week) doesn't match operator input type (Undefined).
- Conditions must return boolean (true/false).
- Left-hand side must return boolean (true/false) for a condition when no operator is defined.
- Right-hand side undefined. Right-hand side must not be "None" when operator is present.

If you see one of these messages, you will need to correct the variable or condition in order to use it elsewhere in the program.

#### Code Warning and Error Messages

When you click any one of the Code Type buttons in the main menu, the program checks the strategy code for problems before updating the code in the code window. If any problems are found, one or more warning or error messages are displayed in the Messages window. Examples include the following:

- Code Error -- Indicator Triangular Mov Ave is not supported for code type MetaTrader 4 (MQL4).
- Code Error -- End-of-week exit not allowed in NinjaTrader 8 code.
- Code Error -- For AmiBroker code, a strategy can only trade one side of the market, either long-only or short-only.
- Code Error -- Nested indicator found in short entry rule. Nested indicators not allowed in MT4 code.
- Code Error -- End-of-day exit not allowed in MetaTrader 4 (MQL4) code.
- Code Warning -- End-of-day exit only valid for intraday data in NinjaTrader 8 code.

- Code Error -- Strategy references data series 2 but no symbol for this data series has been defined.

If any code error messages are shown, they should be corrected to avoid problem when compiling the strategy in the trading platform.

#### General Informational Messages

This is a catch-all category for messages that can be displayed at any time when working with the program. Examples include:

- Absolute value can only be applied to a price difference.*
- Must have an entry for at least one side of market.
- Data copied to clipboard.
- Saving failed. Possible solutions: (1) change file permissions, (2) close file if currently open.
- Saving failed. Please try saving to another file.

* The absolute value operator is restricted to price differences in Editor to maintain compatibility with Adaptrade Builder, which restricts indicator and operator usage as part of its semantic rule system.

# Chapter 7
## Ribbon Menu

### File Menu

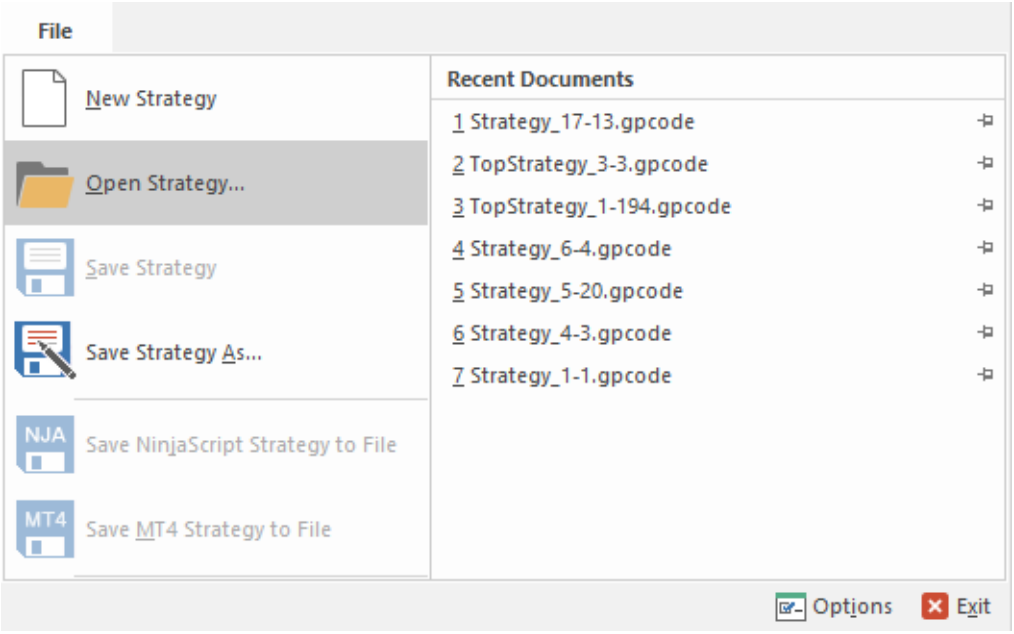



**Figure 7.1. File menu commands.**

The File menu is shown in Fig. 7.1. The file menu commands are listed below.

| | |
|---|---|
| **New Strategy** | Creates a new strategy file. |
| **Open Strategy** | Opens an existing strategy (.gpcode) file. |
| **Save Strategy** | Saves an opened strategy file using the same file name. |
| **Save Strategy As** | Saves an opened strategy file to a specified file name. |
| **Save NinjaScript Strategy to File** | Saves currently selected NinjaTrader strategy code to a file. |
| **Save MT4 Strategy to File** | Saves currently selected MetaTrader 4 strategy code to a file. |
| **Options** | Use this command to customize the ribbon menu and the Quick Access Toolbar. |
| **Exit** | Exits Editor. |

The Recent Documents list shows the 15 most recently opened strategy files. These strategy files can be opened by selecting them from the list.

If any command in the File menu is not applicable, the icon and associated text will be dimmed. For example, if there are no changes to the strategy file, the Save command will be dimmed. In

Fig. 7.1, the commands for saving NinjaScript and MT4 strategies are dimmed because the selected strategy in the open file is in EasyLanguage code.

#### New Strategy Command

Use this command to create a new strategy file in Editor. The new strategy will have mostly empty input fields with the code representing the default input values. If a strategy file is currently open, you will be prompted to save it if necessary before the new strategy file is initialized.

You can open an existing strategy file with the **Open Strategy** command.

#### Open Strategy Command

Use this command to open an existing strategy file. These documents have the file extension .gpcode; e.g., MyNewStrategy.gpcode.

You can create new strategy files with the **New Strategy** command.

**File Open Window**

The following options allow you to specify which file to open:

**File Name**

Specifies the file you want to open. This field lists files with the extension you select in the pull-down menu to the right of this field.

**Files of Type**

Specifies the type of file you want to open. Editor creates and opens files of type .gpcode. You can also select "All Files" in this box, but only files of type .gpcode can be opened.

#### Save Strategy Command

Use this command to save the active strategy to its current file name and directory. When you save a document for the first time, Editor displays the **Save As window** so you can name your strategy file. If you want to change the name and directory of an existing document before you save it, choose the **Save Strategy As** command.

#### Save Strategy As Command

Use this command to save and name the active strategy. Editor displays the **Save As window** so you can name your document.

To save a strategy with its existing name and directory, use the **Save Strategy** command.

**File Save As Window**

The following options allow you to specify the name and location of the file you are about to save:

**File Name**

Specifies a file name to save a document with a different name. Editor adds the extension you specify under **Save As Type**.

#### Save NinjaScript Strategy to File Command

Use this command to save the currently selected NinjaTrader strategy code to a .cs file. The selected strategy must be in NinjaScript code format.

#### Save MT4 Strategy to File Command

Use this command to save the currently selected MetaTrader 4 strategy code to a .mq4 file. The selected strategy must be in MetaTrader 4 (MQL4) code format.

#### Options Command

Use this command to add, remove, or rearrange items on the ribbon menu and to add or remove item from the Quick Access Toolbar (QAT). The QAT is the row of small icons in the upper left-hand corner of the title bar of the main frame window, above the ribbon. It is typically used to display icons for frequently used commands.

#### Exit Command

Use this command to end your Editor session. The program prompts you to save the open document if it has unsaved changes.

## Main Menu

The Main menu, shown in Fig. 7.2, contains the buttons for changing the code type and a set of check boxes for displaying or hiding the program windows.

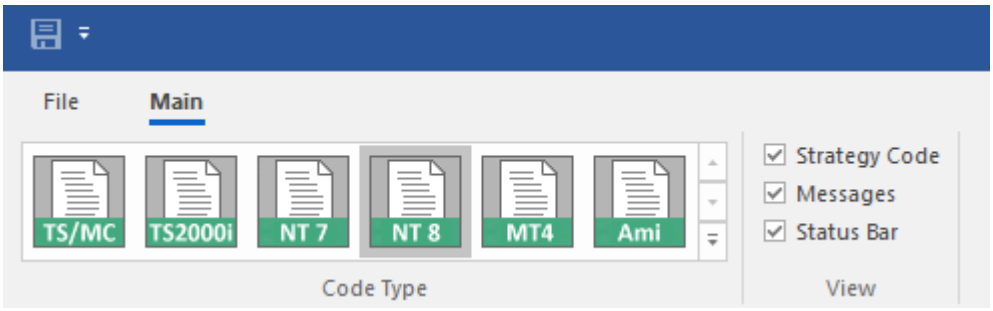



**Figure 7.2. Main menu commands.**

The type of scripting language displayed in the code window is selected via the Code Type panel of the main menu. The choices are as follows:

| | |
|---|---|
| TS/MC | EasyLanguage code for current version of TradeStation and MultiCharts. |
| TS2000i | EasyLanguage code for TradeStation version 2000i. |
| NT 7 | NinjaScript code for NinjaTrader 7. |
| NT 8 | NinjaScript code for NinjaTrader 8. |
| MT4 | MQL4 code for MetaTrader 4. |
| Ami | AFL code for AmiBroker. |

Most changes made in the program will not be reflected in the code window immediately. Click one of the code buttons at any time to update the code display. If there are any detected errors in the code, they will be displayed in the Messages window.

The View panel of the main menu allows you to hide or show the Strategy Code window, the Messages window, and the Status Bar. Toggling the status of a window has no effect on the content of the window.

## Ribbon Tab Buttons

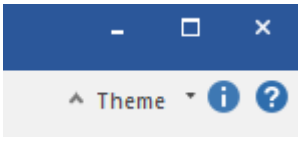



**Figure 7.3. Ribbon tab buttons.**

The ribbon tab buttons, shown in Fig. 7.3, are small buttons located in the upper right-hand corner of the ribbon. There are four ribbon tab buttons in Editor.

| | |
|---|---|
| **^ button** | Minimizes or maximizes the ribbon menu to hide or show the contents of the menus. |
| **Theme** | Allows you to choose from among different user interface themes. |
| **About Editor** | Displays information about this application. |
| **Help Topics** | Offers you an index to topics on which you can get help. |

**Theme**
Use this command to change the style of the menus, toolbars, and other visual elements of the user interface of the Editor program. To change the program style, select a theme from the popup menu. Once the theme has been changed, it will be stored locally, and the program will continue to use that theme until a new theme is selected.

**About Editor**
Use this command to display information about Adaptrade Editor, including the copyright notice and version number.

**Help Topics**
Use this command to display the table of contents for Help. From the table of contents, you can open various topics for using Adaptrade Editor, look up topics in the index, and perform word searches using Find. Press F1 at any time to open a help topic related to the selected command or active window. The help system contains the complete contents of this user's guide.

# Appendix: Technical Indicators

Adaptrade Editor currently includes the indicators described below. Most indicators include one or more inputs, such as the averaging length for a moving average. The input values are entered when the indicator is used in a variable or condition on the Conditions tab (see Chapter 2).

**Simple prices and volume (Open, High, Low, Close, Volume)**
Prices (open, high, low, and close) and volume for the current bar. The volume is the number of shares or contracts traded on the current bar.

**Look-back prices and volume (Open[N], High[N], Low[N], Close[N], Volume[N])**
Versions of simple prices (open, high, low, and close) and volume for the bar N bars ago. For example, Close[2] is the closing price of the bar two bars prior to the current bar.

**Day prices**
A day price is one of the following prices on intraday data: OpenD(0), HighD(0), LowD(0), or CloseD(1). OpenD is the opening price of the session. HighD is the highest price through the current bar of the session. LowD is the lowest price through the current bar of the session, and CloseD is the close of the prior session.

**Simple moving average**
Simple moving average of price over past the most recent N bars, where N is an input. The simple moving average adds up the price over the past N bars and divides by N.

**Exponential moving average**
Exponential moving average (XMA) of price. The XMA is an exponentially weighted average of price. As additional prices are added, their contribution decreases exponentially as (1 - alpha)^N, where alpha = 2/(1 + N). Because more recent prices are weighted more heavily, the XMA is considered to be more responsive to price changes than the simple moving average.

**Weighted moving average**
Weighted moving average of price. The weighted average assigns more weight to recent prices. The weight value decreases by 1 starting with the current bar. Because more recent prices are weighted more heavily, the weighted average is considered to be more responsive to price changes than the simple moving average.

**Triangular moving average[1,2]**
Triangular moving average of price. The triangular moving average is the simple moving average of the simple moving average of price, where the length of the simple moving averages is one-half the specified length of the triangular moving average. Because the triangular moving average is a double-smoothed simple moving average, it weights the middle portion of the look-back period most heavily.

**Adaptive variable moving average (Adapt Variable MA)**
This is an extension of Tushar Chande VIDYA indicator, which is essentially an exponential moving average in which the smoothing constant (alpha) adapts to market volatility or trend strength. In Chande's version, the effective period of the moving average decreases with the trend strength so that highly trending markets have a short moving average period and sideways or choppy markets have a long period moving average.

This version adds an extra input, TrendParam, which can be used to change the relationship between trend and moving average period. Positive values of TrendParam give the same relationship as in Chande's version, where a TrendParam of 1 is similar to VIDYA. Negative values of TrendParam reverse the relationship, so that the period increases with the trend strength and decreases with flat or

trendless markets. A TrendParam of zero is a regular exponential moving average with no dependence on trend. Reasonable values of TrendParam are between roughly -5 and +5.

**Zero-lag trend**
Moving average of price based on John Ehlers' ITrend or InstTrend indicator in Cybernetic Analysis for Stocks and Futures, John Wiley & Sons, Inc, New Jersey, 2004, pp. 16, 24. According to Dr. Ehlers, this indicator has essentially zero lag for price frequencies within the range of interest to most traders.

**Adaptive zero-lag trend (Adapt Zero-Lag Trend)**
This is an extension of the zero-lag trend indicator, which is based on John Ehlers' ITrend or InstTrend indicator in Cybernetic Analysis for Stocks and Futures, John Wiley & Sons, Inc, New Jersey, 2004, pp. 16, 24.

In this version, the smoothing constant (alpha) adapts to market volatility or trend strength, depending on the input TrendParam, which defines the relationship between trend and moving average period. Positive values of TrendParam give a short moving average period during highly trending markets and a long period moving average during sideways or choppy markets. Negative values of TrendParam reverse the relationship, so that the period increases with the trend strength and decreases with flat or trendless markets. A TrendParam of zero gives a period equal to Length (i.e., independent of trend strength). The efficiency ratio is used as a measure of trend strength. Reasonable values of TrendParam are between roughly -5 and +5.

**Moving average convergence divergence (MACD)**
Moving average convergence divergence indicator. The MACD is calculated as the difference between two exponential moving averages of price.

**Triple exponential moving average (TRIX)[1]**
The triple exponential moving average (TRIX) indicator starts by taking the natural logarithm of price, which is then smoothed by applying the exponential moving average three times. The TRIX is calculated as the difference between successive values of the triply smoothed result multiplied by a normalizing factor of 10000. The TRIX acts as an oscillator with values typically ranging from -100 to +100.

**Momentum**
Momentum is an oscillator calculated as the difference between the current price and the price N bars ago, where N is an input. Positive values indicate that prices are rising, whereas negative values indicator that prices are falling.

**Rate of change (ROC)[1]**
ROC is an oscillator calculated as the ratio of the current price to the price N bars ago, subtracted from 1 and multiplied by 100. Positive values indicate that prices are rising, whereas negative values indicator that prices are falling.

**Fast K stochastic[1,2]**
The Fast K stochastic is a price oscillator scaled to the range 0 to 100. It's calculated as the difference between the bar's close and the lowest low, all divided by the difference between the highest high and the lowest low. High values are generally considered to represent so-called over-bought conditions, whereas low values are generally considered to represent so-called over-sold conditions.

**Fast D stochastic**
The Fast D stochastic is a price oscillator scaled to the range 0 to 100. It's calculated as a 3-period exponential moving average of the Fast K stochastic. High values are generally considered to represent so-called over-bought conditions, whereas low values are generally considered to represent so-called over-sold conditions.

**Slow D stochastic**
The Slow D stochastic is a price oscillator scaled to the range 0 to 100. It's calculated as a 3-period exponential moving average of the Fast D stochastic, in which the exponential moving average uses a smoothing factor of 1/N, rather than the value 2/(1 + N) of the normal XMA. High values are generally considered to represent so-called over-bought conditions, whereas low values are generally considered to represent so-called over-sold conditions.

**Relative strength indicator (RSI)**
The RSI is a momentum oscillator that indicates the strength of the market within a range of 0 to 100. The calculation is based on the ratio of up price changes to down price changes. High values are generally considered to represent so-called over-bought conditions, whereas low values are generally considered to represent so-called over-sold conditions.

**RocketRSI**
John Ehlers' RocketRSI indicator calculates a smoothed version of an RSI scaled to [-1, +1] then applies the Fisher transform to convert the values to standard deviations. As with the standard RSI, high values are generally considered to represent so-called over-bought conditions, whereas low values are generally considered to represent so-called over-sold conditions. Reference: John Ehlers, "RocketRSI - A Solid Propellant For Your Rocket Science Trading", Technical Analysis of Stocks & Commodities, May 2018, V36:6 (8 - 12).

**Inverse Fisher RSI**
Inverse Fisher transform of the RSI indicator. The inverse Fisher transform provides sharper turning points for the RSI. The price values are smoothed with a 4-bar weighted moving average before applying the RSI. Indicator values lie between -1 and +1.

**Inverse Fisher cycle**
Inverse Fisher transform of John Ehlers' Stochastic Cyber Cycle indicator from Cybernetic Analysis for Stocks and Futures, John Wiley & Sons, Inc, New Jersey, 2004, p. 75. The Cyber Cycle is an oscillator derived from the cyclic component of price. After computing the stochastic cyber cycle and scaling it to -5 to +5, the inverse Fisher transform is taken to provide sharper turning points. Indicator values lie between -1 and +1.

**Adaptive inverse Fisher RSI (Adapt Inv Fisher RSI)**
This is an extension of the inverse Fisher RSI indicator, which applies the inverse Fisher transform to the RSI function to provide sharper turning points in the RSI. In this version, the smoothing constant (alpha) adapts to market volatility or trend strength, depending on the input TrendParam, which defines the relationship between trend and moving average period. Positive values of TrendParam give a short moving average period during highly trending markets and a long period moving average during sideways or choppy markets. Negative values of TrendParam reverse the relationship, so that the period increases with the trend strength and decreases with flat or trendless markets. A TrendParam of zero gives a period equal to Length (i.e., independent of trend strength). The efficiency ratio is used as a measure of trend strength. Reasonable values of TrendParam are between roughly -5 and +5. Indicator values lie between -1 and +1.

**Adaptive inverse Fisher cycle (Adapt Inv Fisher Cycle)**
This is an extension of the inverse Fisher cycle indicator, which is based on John Ehlers' Stochastic Cyber Cycle indicator from Cybernetic Analysis for Stocks and Futures, John Wiley & Sons, Inc, New Jersey, 2004, p. 75. The Cyber Cycle is an oscillator derived from the cyclic component of price. In this version, the smoothing constant (alpha) adapts to market volatility or trend strength, depending on the input TrendParam, which defines the relationship between trend and moving average period. Positive values of TrendParam give a short moving average period during highly trending markets and a long period moving average during sideways or choppy markets. Negative values of TrendParam reverse the relationship, so that the period increases with the trend strength and decreases with flat or trendless markets. A TrendParam of zero gives a period equal to Length (i.e., independent

of trend strength). The efficiency ratio is used as a measure of trend strength. Reasonable values of TrendParam are between roughly -5 and +5. Indicator values lie between -1 and +1.

**Commodity channel index (CCI)**
The commodity channel index (CCI) is an oscillator calculated as the deviation of the average price, (H+L+C)/3, divided by the average deviation of the average price. The ratio is adjusted by a factor that helps keep most values within the range -200 to +200.

**Directional indicator (DI+/DI-)**
DI+ is the positive directional indicator, whereas DI- is the negative directional indicator. DI+ indicates the presence of an up trend, whereas DI- indicates the presence of a down trend.

**Directional movement index (DMI)[1,2]**
The DMI indicates the strength of the price trend. The DMI is calculated as the absolute value of the difference between DI+ and DI- divided by the sum of DI+ and DI-, multiplied by 100.

**Average directional index(ADX)**
The ADX indicates the strength of the price trend. The ADX is calculated as an exponential moving average of the DMI.

**True range (TR)**
The TR is the difference between the true high (highest of the current bar's high and the prior close) and the true low (lowest of the current bar's low and the prior close). The TR is intended to more accurately represent the bar's range than the high minus the low when the prior close is outside of the current bar's range.

**Average true range (ATR)**
The ATR is the simple moving average of the true range over the past N bars, where N is an input.

**Standard deviation**
The standard deviation is the population (as opposed to sample) standard deviation of price over the past N bars, where N is an input. The standard deviation is a type of price difference and can be compared to other price differences.

**Ulcer index**
The Ulcer index calculates the average percentage drawdown in price, providing a general measure of downside risk in the market. See https://www.investopedia.com/terms/u/ulcerindex.asp.

**Bollinger band**
The Bollinger band is the average price plus a multiple of the standard deviation of price. If the number of standard deviations is positive, the result is an upper band. If the number of standard deviations is negative, the result is a lower band. The upper and lower bands are traditionally used to indicate a range of "normal" price movement.

**Keltner channel[1]**
The Keltner channel is the average price plus a multiple of the average true range. As with Bollinger bands, positive and negative multiples are used to produce upper and lower bands of the Keltner channel.

**Lowest**
The lowest price over the past N bars, where N is an input.

**Highest**
The highest price over the past N bars, where N is an input.

**Accumulation/distribution**
The Accumulation/distribution line is a volume oscillator, calculated by accumulating a portion of the volume of each bar. The amount of volume added at each bar is equal to the difference between the close and the open divided by the range.

**Chaiken oscillator[1]**
The Chaiken oscillator is the difference between two exponential moving averages of the accumulation distribution line.

**Crosses Above/Below[1]**
The crosses above indicator returns “true” if the first input is above the second input on the current bar and below it on the prior bar. If the two inputs are equal on the prior bar, the indicator looks back up to the maximum look-back length to determine if the first input was below the second. If not, the indicator returns “false”. The crosses below indicator works analogously to “crosses above” to return “true” if the first input crosses below the second input on the current bar and returns “false” otherwise.

**Consecutive bars up (AS_ConsecBarsUp)**
This indicator calculates the number of consecutive up prices, in which the "price" input can be any valid price, including any function that returns a price. If the price is lower, the count is reset. If the price is the same as the prior bar, the count is unchanged.

**Consecutive bars down (AS_ConsecBarsDn)**
This indicator calculates the number of consecutive down prices, in which the "price" input can be any valid price, including any function that returns a price. If the price is higher, the count is reset. If the price is the same as the prior bar, the count is unchanged.

**Congestion count (AS_CongestCount)**
The congestion count returns the number of consecutive bars where there is at least one price in common among all bars. High numbers indicate a "congested" or range-bound market.

**Typical price**
Average of high, low, and close; i.e., Typical price = (H + L + C)/3.

**Floor trader pivots (AS_PivotS, AS_PivotR)**
Floor trader pivots consist of three levels of support and three levels of resistance. With CP (central pivot) defined as the Typical Price of the prior bar, the floor trader pivots are defined as follows:

Support, level 1: 2 * CP - High[1]
Support, level 2: CP - (High[1] - Low[1])
Support, level 3: CP - 2 * (High[1] - Low[1])

Resistance, level 1: 2 * CP - Low[1]
Resistance, level 2: CP + High[1] - Low[1]
Resistance, level 3: CP + 2 * (High[1] - Low[1])

in which High[1] and Low[1] are the high and low price, respectively, on the prior bar.

**Day of week**
The day-of-week indicator returns an integer value from 0 to 6 to represent Sunday through Saturday, respectively.

**Time of day**
The time-of-day indicator returns the time of the price bar N bars ago, where N is an input. Typically, the time of a price bar is the time at which the bar closes.

**Absolute value**

The absolute value function can be used when computing the difference between prices to ensure that the price difference is a non-negative number.

[1]Indicator not available for MetaTrader 4 code. [2]Indicator not available for AmiBroker code.

# Appendix: Code Conventions

The strategy code created in Editor uses various naming conventions for inputs, variables, and order type labels. Unless otherwise noted, the same naming conventions apply to all code types available in Editor.

The strategy code naming conventions for input variables are shown below.

**N** : integer or number, typically an indicator look-back length (e.g., N1, NATR).
**X** : floating point value (e.g., X1, X2, etc.).
**En or Ent** : entry (e.g., EntFr, ATRFrEn).
**Ex** : exit (e.g., TimeEx, NBarEx1).
**Fr** : fraction or multiple (e.g., EntFr).
**Targ** : target exit (e.g., TargFr).
**MM** : money management (protective) stop (e.g., MMFr).
**StartEquity**: starting equity value for position sizing calculations.
**PSParam** : position sizing parameter value, such a fixed fraction, delta (fixed ratio), percent of equity, etc.
**RoundPS** : true/false variable. True means round position size to nearest RoundTo increment.
**RoundTo** : value to round position size to if RoundPS is true; ignored if RoundPS is false.
**SizeLimit** : maximum allowable number of shares or contracts per trade.

As an example, consider the following code from the input section of an EasyLanguage strategy, with comments added to explain each input:

```
NBarEn1 (60),          // Number of bars, entry
NBarEn2 (14),          // Number of bars, entry
NBarEn3 (5),           // Number of bars, entry
EntFr (2.6909),        // Entry fraction of price difference or average true range (ATR)
NATR (30),             // Look-back length for ATR
TargFr (1.7681),       // Target fraction; fraction/multiple of ATR
MMFr  (1.51),          // Money management stop fraction; fraction/multiple of ATR
N1 (27),               // Indicator look-back length
N2 (59),               // Indicator look-back length
N3 (27),               // Indicator look-back length
N4 (59),               // Indicator look-back length
N5 (61),               // Indicator look-back length
X1 (10.0000),          // Indicator parameter (floating point) value
NATR (79),             // Look-back length of ATR
ATRFrEn (2.8858),      // Fraction/multiple of ATR for entry
TimeEx  (1015),        // Exit time
StartEquity (100000.00),   // Starting account equity; for position sizing calculations
PSParam (28.39),       // Position sizing parameter value; e.g., percent of equity
RoundPS (true),        // Round position size to nearest RoundTo
RoundTo (1),           // Round position sizing to nearest 1
SizeLimit (100);       // Limit position size to 100 shares or contracts
```

Strategy code created in Editor also includes labels for the entry and exit orders. In TradeStation, these labels are listed in the Trade List and appear on the chart next to each entry and exit. The order labels added by Editor are shown below.

EnStop-L : Entry stop order
EnLimit-L : Limit entry order
EnMark-L : Market entry order

ExMark-L : Market exit order
ExTime-L : Exit-at-time order
ExNBars-L : Exit at N bars from entry order
ExStop-L : Money management (protective) stop exit order
ExTrail-L : Trailing stop exit order
ExTarg-L : Target exit order

The "L" at the end of each label indicates that the order is for a long trade. Order labels for short trades end in "S".

**Variables for Entry and Exit Conditions**
The entry and exit conditions in Editor can be quite complex and may consist of multiple variables and multiple logical condition statements. Variables for numeric quantities, such as indicators, start with "Var" and end with "L" for long conditions or "S" for short-side conditions, followed by a number. Examples include VarL1, VarL2, VarS1, VarS2, etc.

Variables for logical conditions, which involve logical or inequality operators, follow the same convention as numeric variables except that they begin with "Cond". Examples include CondL1, CondL2, CondS1, CondS2, etc.

**Note:** Because the strategy code adds “L” and “S” where necessary, whereas no such labels are necessary in the Variables and Conditions table, the numbering of variables and conditions in the code may not match the numbering shown in the Variables and Conditions table.

The entry and exit conditions themselves are assigned to variables EntCondL, EntCondS, ExCondL, and ExCondS for, respectively, long and short entry conditions and long and short exit conditions.

# Index