ProRealTime

PROGRAMMING GUIDE

Market Scans (ProScreener)

V 6.1.0 – 20251201

Table of contents

Introduction to ProScreener 1

Chapter I: Fundamentals 2

- Accessing ProScreener 2
- Using Top Movers (pre-defined scans) 6

Chapter II: Programming ProScreener 7

- Searching and filtering results 7
  - Use multiple filter criteria 9
  - Formatting the results data 9
- Volume Estimation 10
- Multi-period scanning 10
- Multi-instrument scan 11

Chapter III: Language specificity in ProScreener 13

- Print 13

Chapter IV: Practical Applications 15

- Simple Examples 15
  - RSI 1 hour: Oversold 15
  - RSI 1 hour: Overbought 15
  - Bullish moving average crossover 16
  - Bearish moving average crossover 16
- More elaborate examples 17
  - RSI and bullish reversal 17
  - RSI and bearish reversal 18
  - Bullish Engulfing with trend verification 18
  - Bearish Engulfing with trend verification 19
  - Triple bullish screen 20
  - Triple bearish screen 21

Glossary 22

Warning: ProRealTime does not provide investment advisory services. This document is not in any case personal or financial advice nor a solicitation to buy or sell any financial instrument. The example codes shown in this manual are for learning purposes only. You are free to determine all criteria for your own trading. Past performance is not indicative of future results. Any trading system may expose you to a risk of loss greater than your initial investment.

Introduction to ProScreener

ProScreener is a powerful scanning tool that will allow you to scan entire financial markets:

- to find securities meeting one or more customizable conditions defined by you
- in one timeframe view or in multiple different timeframe views (e.g. 1 minute and 1 hour)
- with results that update in real-time or on the close of each bar with tick-by-tick precision

ProScreener uses the ProBuilder language (you are advised to also consult the **ProBuilder manual**) with extensions that apply exclusively to market scanning conditions.
The scans can be done in real-time or on the close of the current bar via the following timeframes:

- 1 minute
- 2 minutes
- 3 minutes
- 5 minutes
- 10 minutes
- 15 minutes
- 30 minutes
- 1 hour
- 2 hours
- 3 hours
- 4 hours
- Daily
- Weekly
- Monthly
- Quarterly
- Annual

ProScreener performs all calculations using the last 256 candlesticks (1,024 candlesticks on Premium platforms) of the selected timeframes.

ProScreener results are displayed in a list that automatically updates to show the top 50 instruments (or 100 on Premium platforms) that meet the screener conditions, ranked according to the chosen sorting criteria.

This guide is written as a continuation of the **ProBuilder Manual** but may also be consulted independently. The goal is to clearly explain how to best create ProScreeners, with all information related to ProScreener commands and concrete examples. At the end, you will find a glossary with all the commands usable within ProScreener.

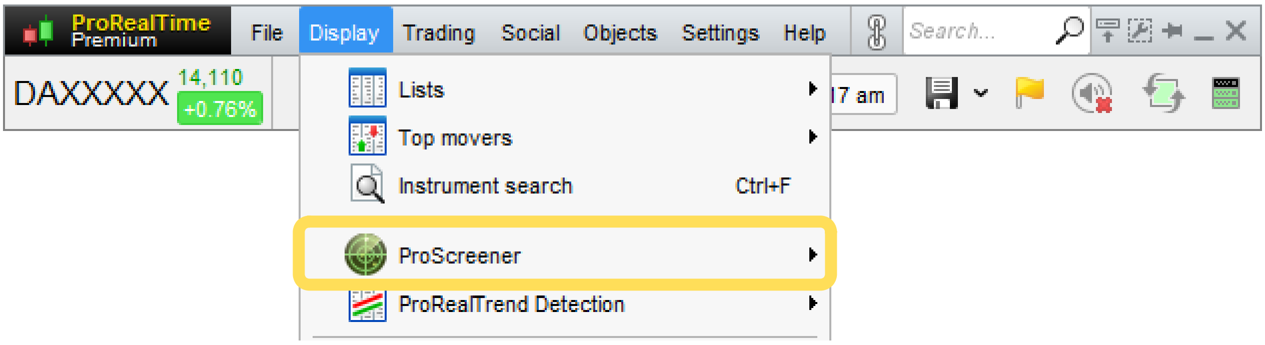
If you have any questions about using ProScreener, you can ask our ProRealTime community on the **ProRealCode forum**, where you will also find **online documentation** with many examples.

We hope you will enjoy the manual!

Chapter I: Fundamentals

Accessing ProScreener

You can access ProScreener by clicking on “Display” and then “ProScreener” as shown below:



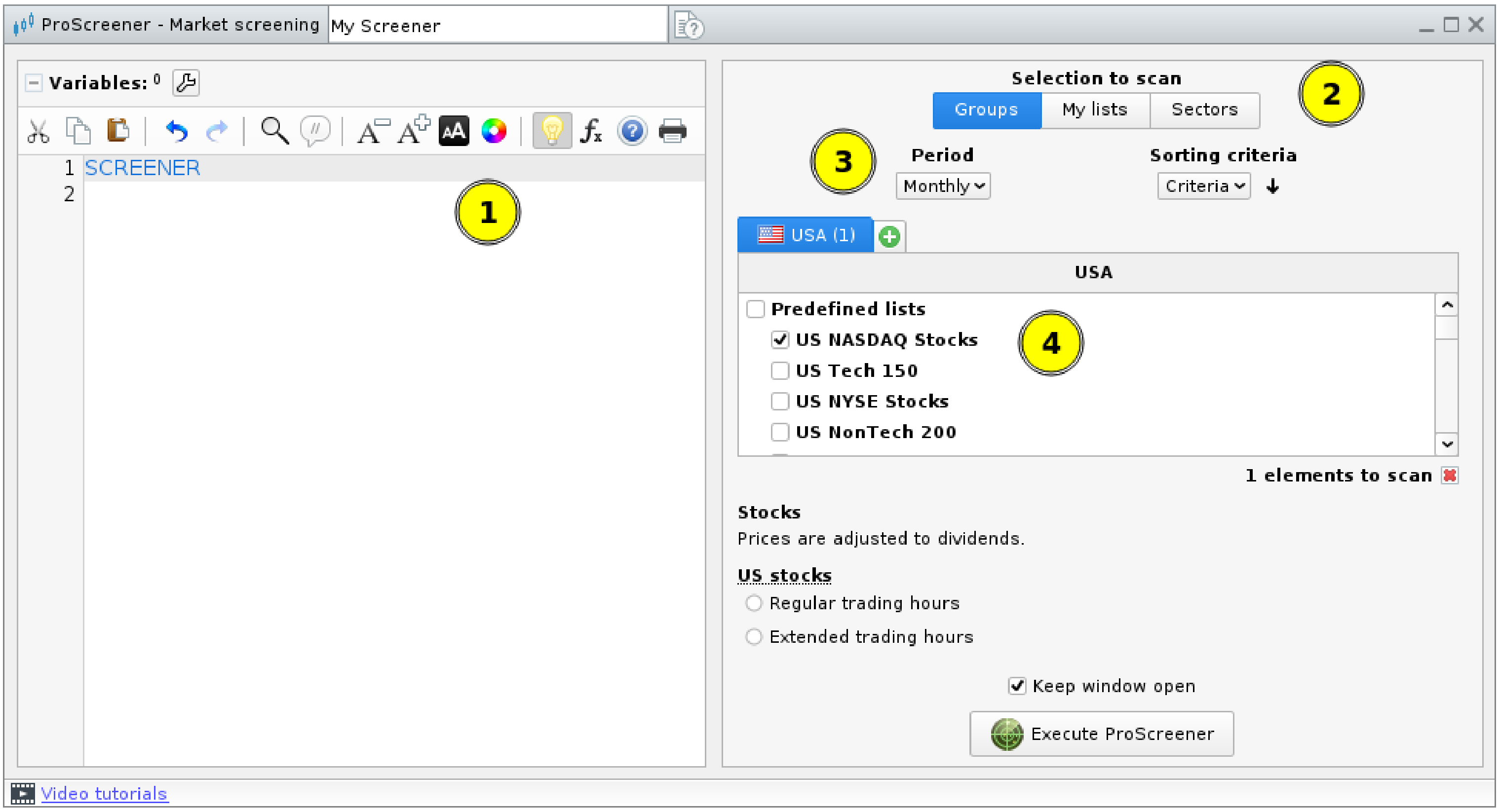
The ProScreener window will appear, which allows you to select an existing ProScreener or to create a new ProScreener.

To do this, click the wrench button at the top of the window to open the “Manage ProScreeners window” and then "New". From this window, you can choose 2 ways to create a new ProScreener:

- "Assisted Creation" mode which allows you to define the conditions of the ProScreener by simply pointing and clicking on one or more charts. To learn how to use this simple assisted-creation process, please watch the video tutorial: "**ProScreener: Real-time scan with multiple conditions without writing a single line of code**".
- "Creation by Programming" that allows you to create more complex codes and define the parameters of your ProScreener.

The Creation by Programming window is made of 4 sections:

1. Programming Zone
2. Selection to scan (financial instrument groups, personal lists, or sectors)
3. Time period and sorting direction
4. Choice of lists/groups to scan



ProScreener - Market screening
My Screener

Variables: 0

```
SCREENER
```

Selection to scan

Groups | My lists | Sectors

Period: Monthly

Sorting criteria: Criteria ↓

USA (1)

USA

□ Predefined lists
☑ US NASDAQ Stocks
□ US Tech 150
□ US NYSE Stocks
□ US NonTech 200

1 elements to scan

Stocks
Prices are adjusted to dividends.

US stocks
○ Regular trading hours
○ Extended trading hours

☑ Keep window open

Execute ProScreener

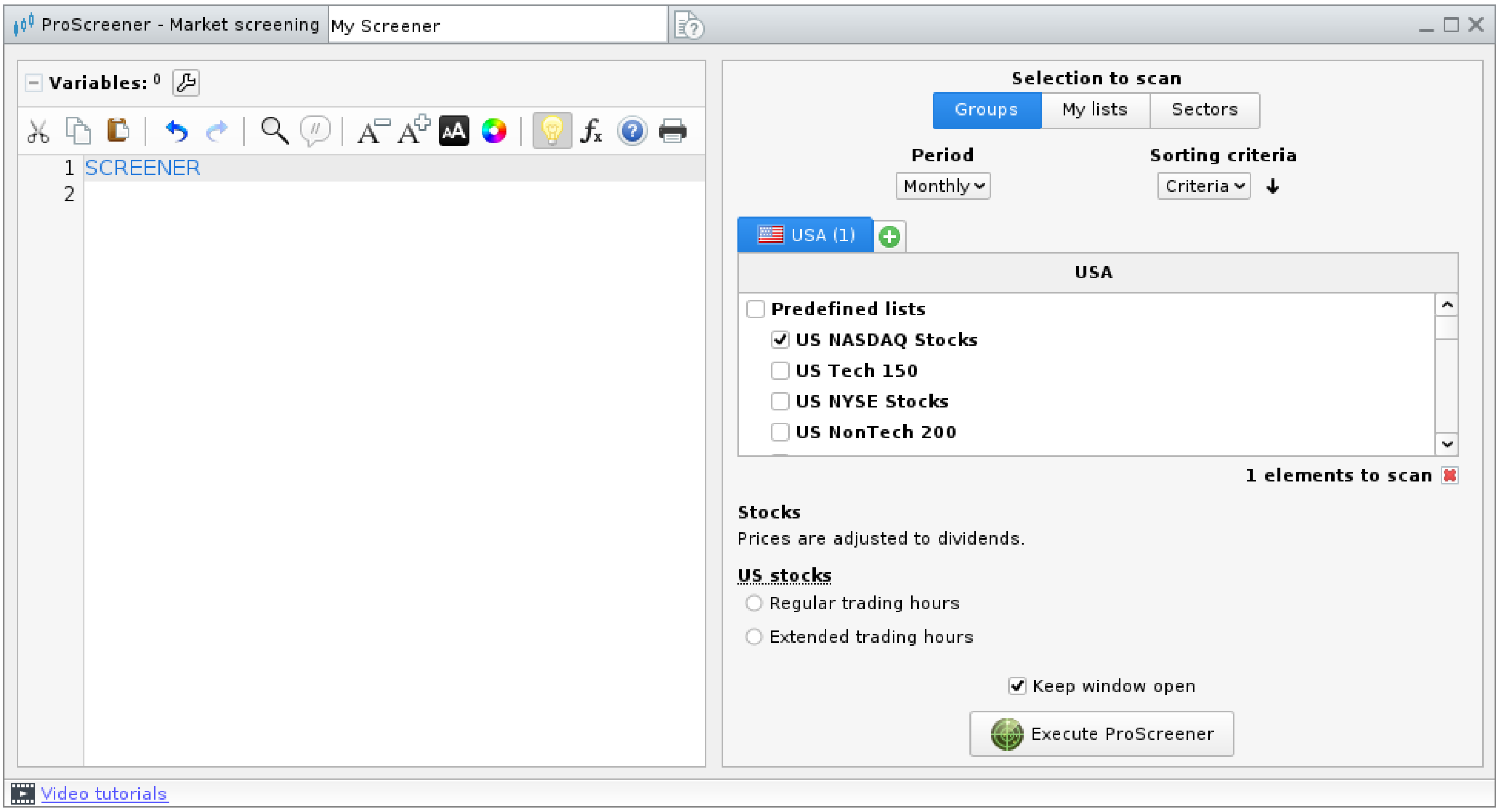
Video tutorials

The **first section (Programming of ProScreener)** allows you to:

- Program a ProScreener directly in the text zone or
- Use "Insert function" to open the function library and insert a function from the library. The function library is divided into categories to help you quickly find the function you need and insert it with correct syntax. It also contains help text related to each function.

If you click "Insert Function" you will notice a special category of "ProScreener commands", which are for use only within ProScreener.

Choose the command "**SCREENER**" whose purpose is to execute the screener and click OK. This will insert the screener command in your program.



ProScreener - Market screening My Screener

Variables: 0

```
SCREENER
```

Selection to scan

Period: Monthly

Sorting criteria: Criteria

USA (1)

USA

□ Predefined lists

☑ US NASDAQ Stocks

□ US Tech 150

□ US NYSE Stocks

□ US NonTech 200

1 elements to scan ☒

Stocks

Prices are adjusted to dividends.

US stocks

○ Regular trading hours

○ Extended trading hours

☑ Keep window open

The "**SCREENER**" command defines what conditions to scan for.

Suppose we want to look for all of the NASDAQ stocks for which the open of the current bar is greater than the close of the previous bar. In our program, we will write the following in the programming zone:

```
c1 = (Open > Close[1])
SCREENER[c1]
```

Once the code is defined, we will choose in the **second** section the type of selection on which the screener will be applied: a group, a customized list or a sector.

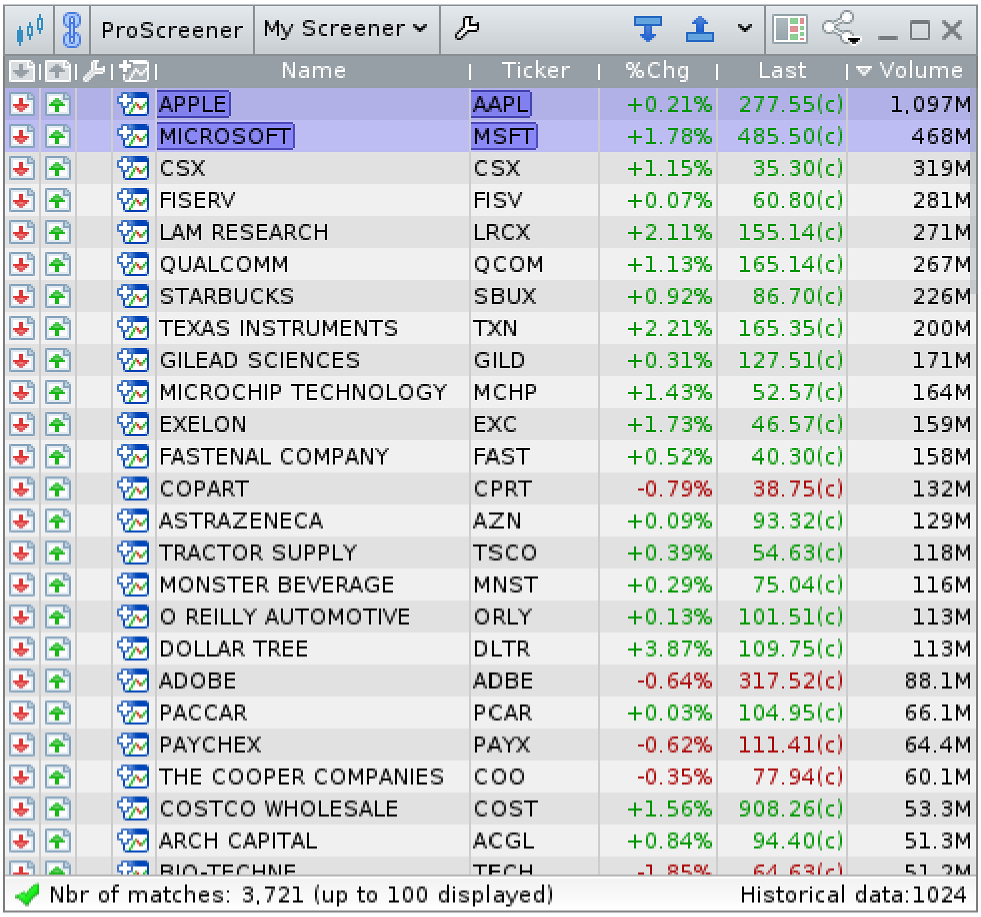
The **third section (Selection of Period)** allows you to define the time period used in the ProScreener. The period used for the detection is important because the conditions you are looking for in one chart timeframe may be different in a different timeframe. It is also possible to define the sorting criteria for the results. When the screening criteria are not very restrictive, a ProScreener can produce thousands of results (on the NASDAQ for example). In this case, you need to filter the results with one or multiple chosen criteria and choose to display:

- the results with the 50 (or 100 with the Premium version of ProRealTime) highest values of the sorting criteria
- the results with the 50 (or 100 with the Premium version of ProRealTime) lowest values of the sorting criteria

The **fourth section** concerns the choice of the group(s) of financial instruments, list(s) or activity sector(s) used for the search.

Example: Imagine your ProScreener searches for the NASDAQ stocks with volume higher than 20,000 on the daily chart. The number of results will logically exceed 50. In your program, you can define volume as the sorting criteria and choose highest 50 values of sorting criteria (which will return the securities with TOP volume first) or lowest values of sorting criteria (which will return the securities with LOWEST volume still meeting the conditions of the screener first, those closest to 20,000).

When you have set up the 4 sections as you want, click "Execute ProScreener" to validate the creation or modification of your ProScreener. If there are results, they will be displayed in the table as shown below.



Name	Ticker	%Chg	Last	▾ Volume
APPLE	AAPL	+0.21%	277.55(c)	1,097M
MICROSOFT	MSFT	+1.78%	485.50(c)	468M
CSX	CSX	+1.15%	35.30(c)	319M
FISERV	FISV	+0.07%	60.80(c)	281M
LAM RESEARCH	LRCX	+2.11%	155.14(c)	271M
QUALCOMM	QCOM	+1.13%	165.14(c)	267M
STARBUCKS	SBUX	+0.92%	86.70(c)	226M
TEXAS INSTRUMENTS	TXN	+2.21%	165.35(c)	200M
GILEAD SCIENCES	GILD	+0.31%	127.51(c)	171M
MICROCHIP TECHNOLOGY	MCHP	+1.43%	52.57(c)	164M
EXELON	EXC	+1.73%	46.57(c)	159M
FASTENAL COMPANY	FAST	+0.52%	40.30(c)	158M
COPART	CPRT	-0.79%	38.75(c)	132M
ASTRAZENECA	AZN	+0.09%	93.32(c)	129M
TRACTOR SUPPLY	TSCO	+0.39%	54.63(c)	118M
MONSTER BEVERAGE	MNST	+0.29%	75.04(c)	116M
O REILLY AUTOMOTIVE	ORLY	+0.13%	101.51(c)	113M
DOLLAR TREE	DLTR	+3.87%	109.75(c)	113M
ADOBE	ADBE	-0.64%	317.52(c)	88.1M
PACCAR	PCAR	+0.03%	104.95(c)	66.1M
PAYCHEX	PAYX	-0.62%	111.41(c)	64.4M
THE COOPER COMPANIES	COO	-0.35%	77.94(c)	60.1M
COSTCO WHOLESALE	COST	+1.56%	908.26(c)	53.3M
ARCH CAPITAL	ACGL	+0.84%	94.40(c)	51.3M
BIO-TECHNE	TECH	-1.85%	64.63(c)	51.2M

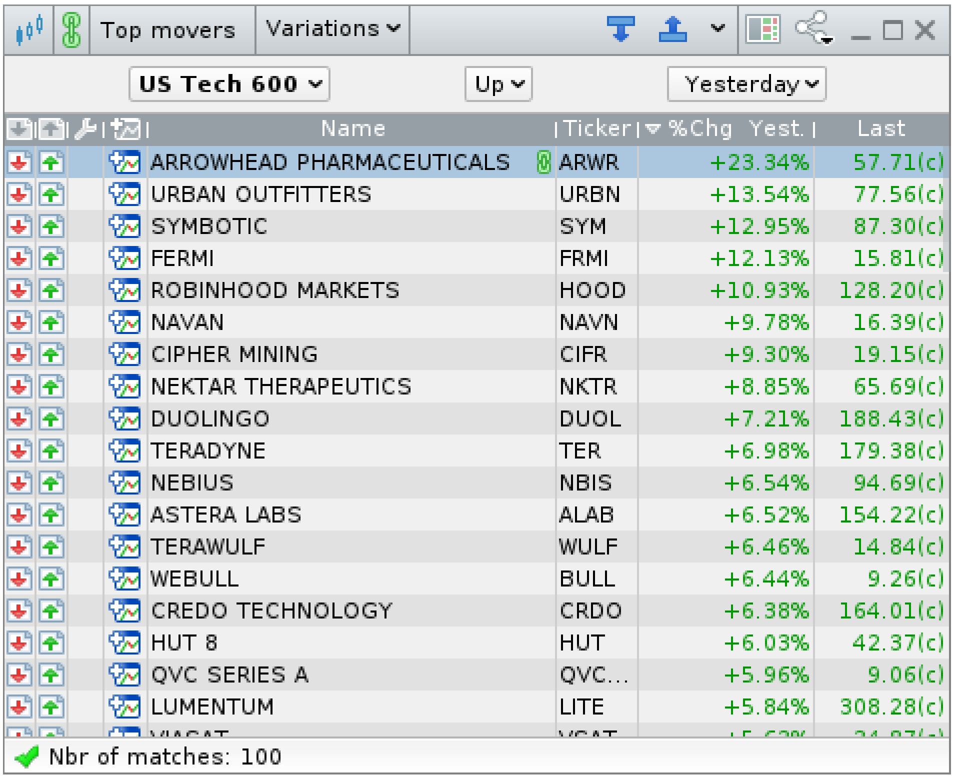
Nbr of matches: 3,721 (up to 100 displayed)   Historical data:1024

Using Top Movers (pre-defined scans)

In addition to ProScreener, ProRealTime provides the Top Movers scan tool. Top Movers scans the market with pre-defined criteria (unlike ProScreener, which is completely customizable). Top Movers lets you do simultaneous scans (example: stocks whose price has increased the most today in one Top Movers window and stocks whose price has decreased the most today in another Top Movers window).

The search criteria available in Top Movers include 4 categories and let you detect:

- Price variations, price gaps and abnormal trade volume
- Pre-opening variations
- The main candlestick formations
- Order book spread and order book volume



US Tech 600 · Up · Yesterday

Name	Ticker	%Chg Yest.	Last
ARROWHEAD PHARMACEUTICALS	ARWR	+23.34%	57.71(c)
URBAN OUTFITTERS	URBN	+13.54%	77.56(c)
SYMBOTIC	SYM	+12.95%	87.30(c)
FERMI	FRMI	+12.13%	15.81(c)
ROBINHOOD MARKETS	HOOD	+10.93%	128.20(c)
NAVAN	NAVN	+9.78%	16.39(c)
CIPHER MINING	CIFR	+9.30%	19.15(c)
NEKTAR THERAPEUTICS	NKTR	+8.85%	65.69(c)
DUOLINGO	DUOL	+7.21%	188.43(c)
TERADYNE	TER	+6.98%	179.38(c)
NEBIUS	NBIS	+6.54%	94.69(c)
ASTERA LABS	ALAB	+6.52%	154.22(c)
TERAWULF	WULF	+6.46%	14.84(c)
WEBULL	BULL	+6.44%	9.26(c)
CREDO TECHNOLOGY	CRDO	+6.38%	164.01(c)
HUT 8	HUT	+6.03%	42.37(c)
QVC SERIES A	QVC...	+5.96%	9.06(c)
LUMENTUM	LITE	+5.84%	308.28(c)

✓ Nbr of matches: 100

To learn more about the Top Movers pre-defined scanning tools, please watch the related videos on our **video tutorials page** including "**Top Movers: market scans with pre-defined criteria**"

Chapter II: Programming ProScreener

In this chapter we will illustrate the 4 commands in the ProBuilder language that are only for use in the ProScreener module and allow you to do scans. These commands are displayed in the "ProScreener commands" section when you press the "Insert Function" button in the programming zone. We will look at:

- Searching and filtering results
- Volume estimation
- Multi-period scanning
- Multi-instrument scanning

Searching and filtering results

The "**SCREENER**" command defines the condition to display an instrument in the results windows.

The syntax for "**SCREENER**" is as follows:

```
SCREENER[Condition]
```

Example:

```
c1 = (Close < BollingerDown[10](Close))
SCREENER[c1]
```

Let's find all of the instruments in a market for which the closing price is strictly less than the lower Bollinger Band. Bollinger bands are calculated using 10 periods and applied to the closing price.

It is possible to scan for more than one condition with **AND** (both conditions must be met) or **OR** (at least one condition must be met). To do this in ProScreener, the syntax is:

```
SCREENER[Condition1 AND Condition2]
```

OR

```
SCREENER[Condition1 OR Condition2]
```

Example:

Let's search for securities for which the price is greater than the upper Bollinger band and have an increasing trend shown by the fact that the close is greater than the open and the 7-period moving average is above the 23-period moving average.

```
// Close is higher than the Upper Bollinger band of 20 periods calculated on the close
Condition1 = Close > BollingerUp[20](Close)
// Close > Open
Condition2 = Close > Open
// MA7 calculated on close > MA23 calculated on close
Condition3 = (Average[7](Close) > Average[23](Close))
SCREENER[Condition1 AND Condition2 AND Condition3]
```

It is also possible to define a constant or an indicator that will also work as filter and sorting criteria for the results:

- If the scan returns more than 50 or 100 results, the indicator filters the securities to display. In this case, the third section of the programming window allows you to define whether you want to display the results with the highest or lowest values of the sorting criteria (see **description on page 5**).
- If the scan returns less than 50 or 100 results, the scanner will sort the results based on the sorting criteria that you have defined.

The syntax to use a constant as sorting criteria is:

```
SCREENER[c1](Criteria)
```

To use a pre-defined indicator as a sorting criteria, it is preferable to first store it in a variable (here we will call it "Criteria"):

```
Criteria = RSI[14](Close)
SCREENER[c1](Criteria)
```

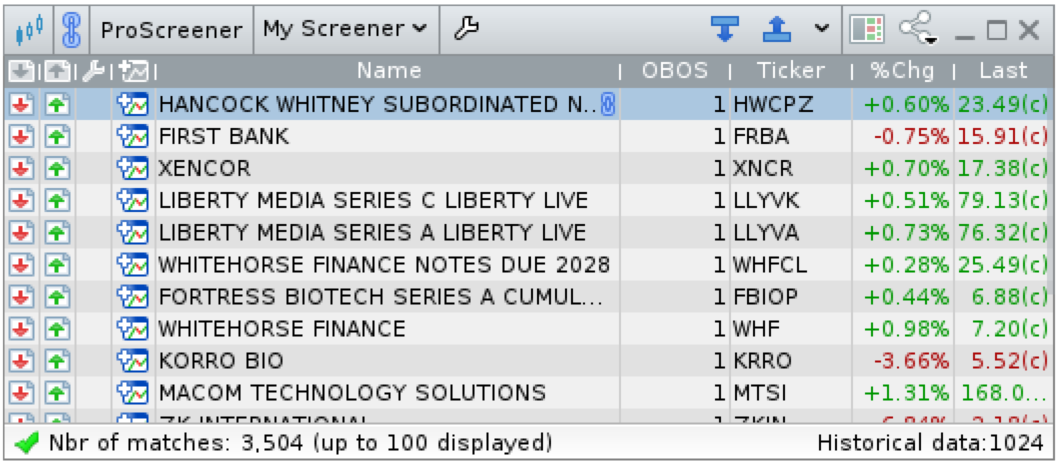
If we call a personalized indicator (indicator written in ProBuilder language), we need to use the "`CALL`", command as defined in the **ProBuilder Manual**.

```
MyRSI = CALL "PersonalRSI" [14]
SCREENER[c1](MyRSI)
```

Example:

Let's find securities whose volume is greater than 50,000. Then let's sort the results by whether the RSI is overbought or oversold. To measure whether the RSI is overbought or oversold, we create a binary indicator which is equal to 1 if it's greater than 70 and -1 if it's less than 30. We will display the result of the binary indicator in the sorting criteria column.

```
c1 = Volume > 50000
MyRSI = RSI[14](Close)
IF MyRSI > 70 THEN
   Criteria = 1
ELSIF MyRSI < 30 THEN
   Criteria = -1
ENDIF
SCREENER[c1](Criteria AS "OBOS")
```



Name	OBOS	Ticker	%Chg	Last
HANCOCK WHITNEY SUBORDINATED N...	1	HWCPZ	+0.60%	23.49(c)
FIRST BANK	1	FRBA	-0.75%	15.91(c)
XENCOR	1	XNCR	+0.70%	17.38(c)
LIBERTY MEDIA SERIES C LIBERTY LIVE	1	LLYVK	+0.51%	79.13(c)
LIBERTY MEDIA SERIES A LIBERTY LIVE	1	LLYVA	+0.73%	76.32(c)
WHITEHORSE FINANCE NOTES DUE 2028	1	WHFCL	+0.28%	25.49(c)
FORTRESS BIOTECH SERIES A CUMUL...	1	FBIOP	+0.44%	6.88(c)
WHITEHORSE FINANCE	1	WHF	+0.98%	7.20(c)
KORRO BIO	1	KRRO	-3.66%	5.52(c)
MACOM TECHNOLOGY SOLUTIONS	1	MTSI	+1.31%	168.0...

✓ Nbr of matches: 3,504 (up to 100 displayed) Historical data:1024

It is possible to not use the parentheses if we use the "SORT BY" command instead:

```
SCREENER[c1] SORT BY TypicalPrice
```

If we want to give a personalized name to this column, we should use the "AS" function as illustrated here:

```
SCREENER[c1](TypicalPrice AS "typical price")
```

or

```
SCREENER[c1] SORT BY TypicalPrice AS “typical price”
```

Use multiple filter criteria

The screener allows you to define multiple criteria (3 or 6 depending on your ProRealTime license) as filters in your code to display additional values in the results window. This also allows you to change the filter criteria without modifying your code.

Filter criteria within the "SCREENER" instructions will have to be separated by commas.

Example:

```
SCREENER[c1](Criteria AS "OBOS", Criteria2 AS "buy/sell",
number AS "Format Number" DATEFORMAT)
```

This functionality is only available if your code contains no more than one "SCREENER" instruction.

Formatting the results data

The screener also provides the ability to specify the type of data to be displayed thanks to the format instructions:

- **PERCENTFORMAT**: Displays the value as a percentage
- **STRINGFORMAT**: Displays the value without formatting
- **NUMBERFORMAT**: Displays the value in numerical format
- **DATEFORMAT**: Displays the value in the form of a date

Let's take the following number:

```
number = 20221202
```

Here are the results displayed for the different types of formats proposed:

Format	Result
`SCREENER(number AS "Format Number" PERCENTFORMAT)`	2,022,120,200.00%
`SCREENER(number AS "Format Number" STRINGFORMAT)`	20221202
`SCREENER(number AS "Format Number" NUMBERFORMAT)`	20.2M
`SCREENER(number AS "Format Number" DATEFORMAT)`	2 Dec 2022

Volume Estimation

The "**EstimatedVolume**" command allows us to do a linear estimation of what the ending volume of the current bar will be.

Estimated Volume is calculated in the following way:

EstimatedVolume = Volume * Multiplier

where

Multiplier = (unit of time) / (time elapsed since the current candlestick started)

This command is particularly interesting to compare estimated volume to actual volume.

For example:

In 10-minute view, if 1 minute has passed, we can estimate that the volume for the candle at the end will be 10 times the current volume for the current candle.

For example, let's calculate the ratio of today's estimated volume to the previous day's volume:

```
// Vol0 estimates the volume for the current bar
Vol0 = EstimatedVolume
// Vol1 is the volume of the previous bar
Vol1 = Volume[1]
// The screener will organize the results by the ratio of estimated volume of the current bar to actual volume of the previous bar.
SCREENER (Vol0 / Vol1 AS "Vratio")
```

Multi-period scanning

It’s possible to run a ProScreener with multiple conditions and on multiple time periods. This enables you to check for example if your condition is true on several different charting timeframes which you may want to look at (e.g. short and long-term chart). The command to use is "**TIMEFRAME**" and its syntax is as follows:

```
TIMEFRAME(code of the timeframe)
```

The timeframes available in ProScreener are listed below:

TIMEFRAME	CODE OF THE TIMEFRAME
1 minute	`1 minute`
2 minutes	`2 minutes`
3 minutes	`3 minutes`
5 minutes	`5 minutes`
10 minutes	`10 minutes`
15 minutes	`15 minutes`
30 minutes	`30 minutes`
1 hour	`1 hour`
2 hours	`2 hours`
3 hours	`3 hours`
4 hours	`4 hours`
Daily	`Daily`
Weekly	`Weekly`
Quarterly	`Quarterly`
Yearly	`Yearly`

Instructions following the "**TIMEFRAME**" command will be calculated only in the period indicated. It is possible to use multiple "**TIMEFRAME**" lines in the same program to do a multi-period scan.

Example:
We want to find all the NASDAQ stocks corresponding to the criteria below:

- In the weekly timeframe, the Williams %R applied to close of 14-periods is between 0 and -20
- In the 30-minute timeframe, the EMA of 20 periods applied to close recently crossed over the EMA of 12 periods.

Here is the code for this example ProScreener:

```
TIMEFRAME(weekly)
Condition1 = Williams[14](Close) < 0 AND Williams[14](Close) > -20
TIMEFRAME(30 minutes)
Condition2 = ExponentialAverage[20](Close) CROSSES OVER ExponentialAverage[12](Close)
SCREENER[Condition1 AND Condition2]
```

To return to the data of the selected period in the ProScreener interface, we can write:

```
TIMEFRAME(default)
```

Multi-instrument scan

With ProScreener, we have seen that we need to specify a market or list on which to do our scan.

"**EQUITYFRAME**" allows us to scan for criteria related to a specific financial instrument.

The syntax for "**EQUITYFRAME**" is:

```
EQUITYFRAME("market name","ticker")
```

The command allows us to compare the results of the scan to a particular security, or construct a new variable for use later in the code, or also to construct a new indicator as a sorting criteria.

Example:
To display 50 securities in the NASDAQ market, for which the closing price of the current bar is higher than the close of the stock Microsoft (Ticker "MSFT"), you could use the following code:

```
MyClose = Close
EQUITYFRAME("NASDAQ", "MSFT")
MSFTClose = Close
SCREENER MyClose > MSFTClose SORT BY MyClose AS "Close"
```

You can find the market names and tickers in the "Instrument Search" window. When running your screener, if there is a name error in your **EQUITYFRAME** instruction, the platform will automatically open the instrument search window to allow you to select the instrument.

As with the **TIMEFRAME** command, any code that follows **EQUITYFRAME** will be applied to the financial instrument specified by it. To return to the market data selected in the ProScreener interface, we can write:

```
EQUITYFRAME(default)
```

The 2 uses displayed above complement each other as the example below shows:

If you want to display in the ProScreener results table, the indicator that shows the difference between the close of the current bar for the currency pair EUR/USD (ticker "EURUSD") and the closing price of the results of the screener, you can do so in this way:

```
// We begin by choosing the EQUITYFRAME EURUSD and creating a variable to store its closing price
EQUITYFRAME("Forex 47 Major Pairs","EURUSD")
MyClose = Close
// We come back to the default equityframe for the market selected in the section "Selection of List" (Forex in this case)
EQUITYFRAME(default)
// We do the difference between the close of EURUSD and the close of the scan result
CloseVal = MyClose - Close
// We filter the results by the difference between EURUSD and the price of the scan result
SCREENER SORT BY CloseVal AS "MyIndicator"
```

Example: This screener allows us to visually display the “relative strength” between a security and other securities of the screener list used. We calculate the ratio of the closing prices of 2 securities selected using "**EQUITYFRAME**". We then calculate the difference between the current level of the ratio and the ratio for the previous day.

```
TIMEFRAME(daily)
CloseVal = Close
EQUITYFRAME("NASDAQ Shares","AMZN")
CloseInd = Close
EQUITYFRAME(default)
Ratio = CloseVal / CloseInd * 100
RelativeStrength = (Ratio - Ratio[1]) * 100
SCREENER SORT BY RelativeStrength AS "Relative Strength"
```

Chapter III: Language specificity in ProScreener

This section describes the specific behavior of certain ProBuilder instructions in the ProScreener module. Each indication below should be considered in addition to the information available in the general ProBuilder documentation.

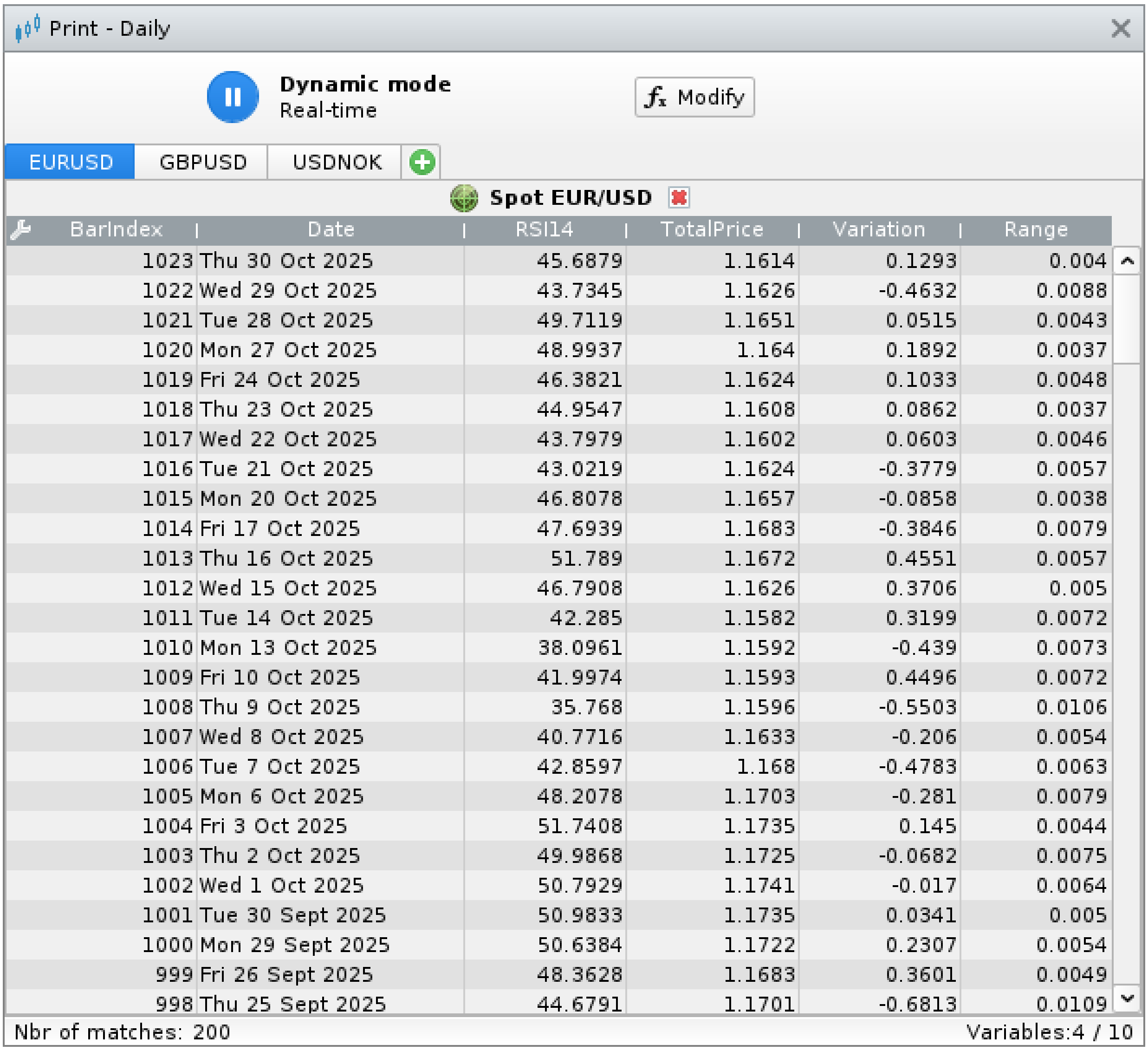
Print

In ProScreener, when you use the PRINT instruction, you must define the scanned instruments on which you want to display data (for debugging purposes, for example).

These instruments must be included in the list of scanned values.

Example: With a screener launched on the Forex 47 list, it is not possible to display calculations on the APPLE stock (AAPL code).

Tickers to print are selected after the screener has been launched by clicking the “+” button in the PRINT frame:



Spot EUR/USD

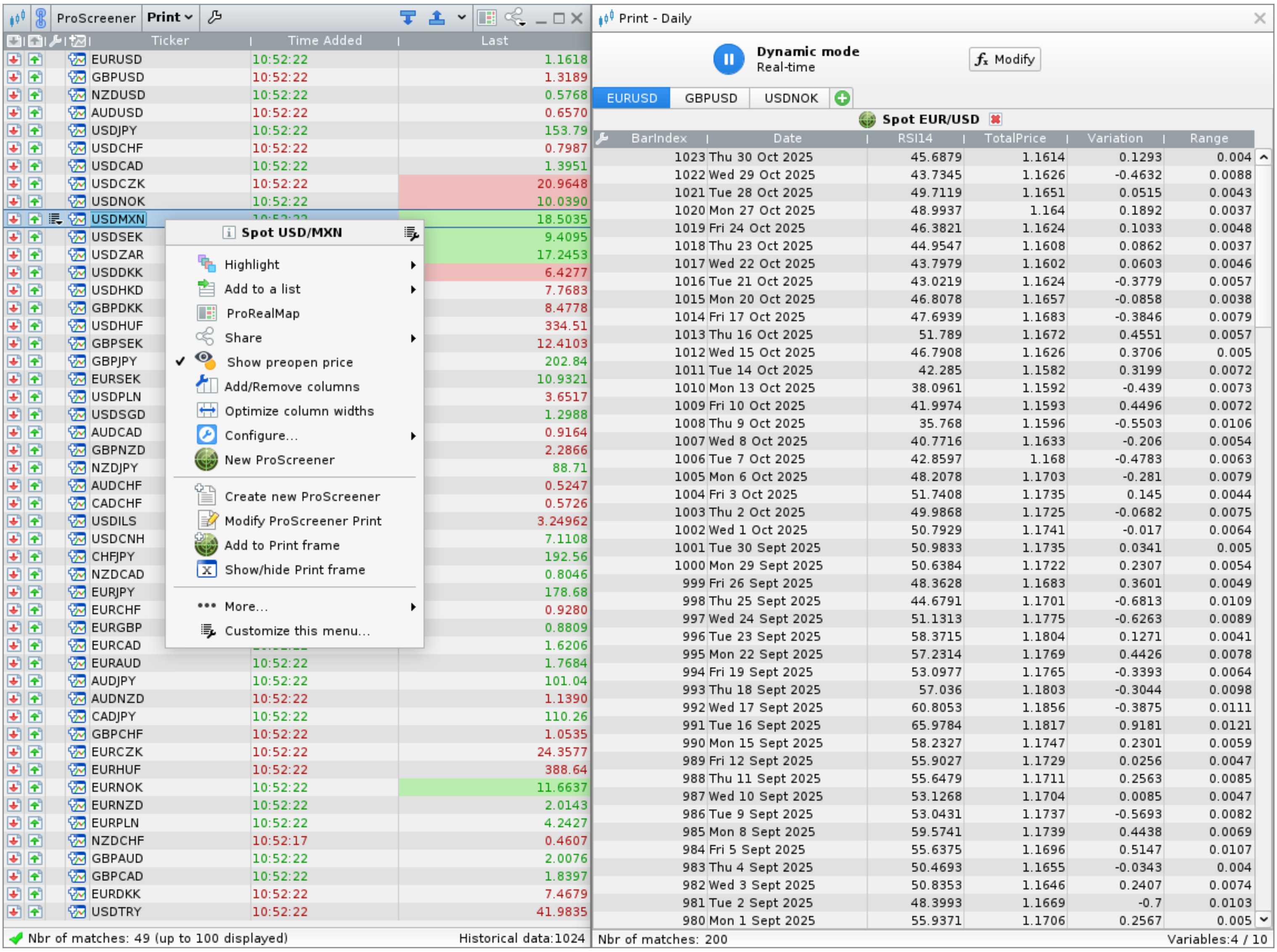
BarIndex	Date	RSI14	TotalPrice	Variation	Range
1023	Thu 30 Oct 2025	45.6879	1.1614	0.1293	0.004
1022	Wed 29 Oct 2025	43.7345	1.1626	-0.4632	0.0088
1021	Tue 28 Oct 2025	49.7119	1.1651	0.0515	0.0043
1020	Mon 27 Oct 2025	48.9937	1.164	0.1892	0.0037
1019	Fri 24 Oct 2025	46.3821	1.1624	0.1033	0.0048
1018	Thu 23 Oct 2025	44.9547	1.1608	0.0862	0.0037
1017	Wed 22 Oct 2025	43.7979	1.1602	0.0603	0.0046
1016	Tue 21 Oct 2025	43.0219	1.1624	-0.3779	0.0057
1015	Mon 20 Oct 2025	46.8078	1.1657	-0.0858	0.0038
1014	Fri 17 Oct 2025	47.6939	1.1683	-0.3846	0.0079
1013	Thu 16 Oct 2025	51.789	1.1672	0.4551	0.0057
1012	Wed 15 Oct 2025	46.7908	1.1626	0.3706	0.005
1011	Tue 14 Oct 2025	42.285	1.1582	0.3199	0.0072
1010	Mon 13 Oct 2025	38.0961	1.1592	-0.439	0.0073
1009	Fri 10 Oct 2025	41.9974	1.1593	0.4496	0.0072
1008	Thu 9 Oct 2025	35.768	1.1596	-0.5503	0.0106
1007	Wed 8 Oct 2025	40.7716	1.1633	-0.206	0.0054
1006	Tue 7 Oct 2025	42.8597	1.168	-0.4783	0.0063
1005	Mon 6 Oct 2025	48.2078	1.1703	-0.281	0.0079
1004	Fri 3 Oct 2025	51.7408	1.1735	0.145	0.0044
1003	Thu 2 Oct 2025	49.9868	1.1725	-0.0682	0.0075
1002	Wed 1 Oct 2025	50.7929	1.1741	-0.017	0.0064
1001	Tue 30 Sept 2025	50.9833	1.1735	0.0341	0.005
1000	Mon 29 Sept 2025	50.6384	1.1722	0.2307	0.0054
999	Fri 26 Sept 2025	48.3628	1.1683	0.3601	0.0049
998	Thu 25 Sept 2025	44.6791	1.1701	-0.6813	0.0109

Nbr of matches: 200 Variables:4 / 10

Up to 10 different instruments can be displayed in the PRINT frame.

Right-clicking on an instrument in the screener frame using a **PRINT** instruction allows the following actions:

- Adding the selected instrument to the **PRINT** frame
- Showing and hiding the **PRINT** frame



Ticker	Time Added	Last
EURUSD	10:52:22	1.1618
GBPUSD	10:52:22	1.3189
NZDUSD	10:52:22	0.5768
AUDUSD	10:52:22	0.6570
USDJPY	10:52:22	153.79
USDCHF	10:52:22	0.7987
USDCAD	10:52:22	1.3951
USDCZK	10:52:22	20.9648
USDNOK	10:52:22	10.0390
USDMXN	10:52:22	18.5035
USDSEK		9.4095
USDZAR		17.2453
USDDKK		6.4277
USDHKD		7.7683
GBPDKK		8.4778
USDHUF		334.51
GBPSEK		12.4103
GBPJPY		202.84
EURSEK		10.9321
USDPLN		3.6517
USDSGD		1.2988
AUDCAD		0.9164
GBPNZD		2.2866
NZDJPY		88.71
AUDCHF		0.5247
CADCHF		0.5726
USDILS		3.24962
USDCNH		7.1108
CHFJPY		192.56
NZDCAD		0.8046
EURJPY		178.68
EURCHF		0.9280
EURGBP		0.8809
EURCAD		1.6206
EURAUD	10:52:22	1.7684
AUDJPY	10:52:22	101.04
AUDNZD	10:52:22	1.1390
CADJPY	10:52:22	110.26
GBPCHF	10:52:22	1.0535
EURCZK	10:52:22	24.3577
EURHUF	10:52:22	388.64
EURNOK	10:52:22	11.6637
EURNZD	10:52:22	2.0143
EURPLN	10:52:22	4.2427
NZDCHF	10:52:17	0.4607
GBPAUD	10:52:22	2.0076
GBPCAD	10:52:22	1.8397
EURDKK	10:52:22	7.4679
USDTRY	10:52:22	41.9835

Nbr of matches: 49 (up to 100 displayed) Historical data:1024

Spot EUR/USD

BarIndex	Date	RSI14	TotalPrice	Variation	Range
1023	Thu 30 Oct 2025	45.6879	1.1614	0.1293	0.004
1022	Wed 29 Oct 2025	43.7345	1.1626	-0.4632	0.0088
1021	Tue 28 Oct 2025	49.7119	1.1651	0.0515	0.0043
1020	Mon 27 Oct 2025	48.9937	1.164	0.1892	0.0037
1019	Fri 24 Oct 2025	46.3821	1.1624	0.1033	0.0048
1018	Thu 23 Oct 2025	44.9547	1.1608	0.0862	0.0037
1017	Wed 22 Oct 2025	43.7979	1.1602	0.0603	0.0046
1016	Tue 21 Oct 2025	43.0219	1.1624	-0.3779	0.0057
1015	Mon 20 Oct 2025	46.8078	1.1657	-0.0858	0.0038
1014	Fri 17 Oct 2025	47.6939	1.1683	-0.3846	0.0079
1013	Thu 16 Oct 2025	51.789	1.1672	0.4551	0.0057
1012	Wed 15 Oct 2025	46.7908	1.1626	0.3706	0.005
1011	Tue 14 Oct 2025	42.285	1.1582	0.3199	0.0072
1010	Mon 13 Oct 2025	38.0961	1.1592	-0.439	0.0073
1009	Fri 10 Oct 2025	41.9974	1.1593	0.4496	0.0072
1008	Thu 9 Oct 2025	35.768	1.1596	-0.5503	0.0106
1007	Wed 8 Oct 2025	40.7716	1.1633	-0.206	0.0054
1006	Tue 7 Oct 2025	42.8597	1.168	-0.4783	0.0063
1005	Mon 6 Oct 2025	48.2078	1.1703	-0.281	0.0079
1004	Fri 3 Oct 2025	51.7408	1.1735	0.145	0.0044
1003	Thu 2 Oct 2025	49.9868	1.1725	-0.0682	0.0075
1002	Wed 1 Oct 2025	50.7929	1.1741	-0.017	0.0064
1001	Tue 30 Sept 2025	50.9833	1.1735	0.0341	0.005
1000	Mon 29 Sept 2025	50.6384	1.1722	0.2307	0.0054
999	Fri 26 Sept 2025	48.3628	1.1683	0.3601	0.0049
998	Thu 25 Sept 2025	44.6791	1.1701	-0.6813	0.0109
997	Wed 24 Sept 2025	51.1313	1.1775	-0.6263	0.0089
996	Tue 23 Sept 2025	58.3715	1.1804	0.1271	0.0041
995	Mon 22 Sept 2025	57.2314	1.1769	0.4426	0.0078
994	Fri 19 Sept 2025	53.0977	1.1765	-0.3393	0.0064
993	Thu 18 Sept 2025	57.036	1.1803	-0.3044	0.0098
992	Wed 17 Sept 2025	60.8053	1.1856	-0.3875	0.0111
991	Tue 16 Sept 2025	65.9784	1.1817	0.9181	0.0121
990	Mon 15 Sept 2025	58.2327	1.1747	0.2301	0.0059
989	Fri 12 Sept 2025	55.9027	1.1729	0.0256	0.0047
988	Thu 11 Sept 2025	55.6479	1.1711	0.2563	0.0085
987	Wed 10 Sept 2025	53.1268	1.1704	0.0085	0.0047
986	Tue 9 Sept 2025	53.0431	1.1737	-0.5693	0.0082
985	Mon 8 Sept 2025	59.5741	1.1739	0.4438	0.0069
984	Fri 5 Sept 2025	55.6375	1.1696	0.5147	0.0107
983	Thu 4 Sept 2025	50.4693	1.1655	-0.0343	0.004
982	Wed 3 Sept 2025	50.8353	1.1646	0.2407	0.0074
981	Tue 2 Sept 2025	48.3993	1.1669	-0.7	0.0103
980	Mon 1 Sept 2025	55.9371	1.1706	0.2567	0.005

Nbr of matches: 200 Variables:4 / 10

Chapter IV: Practical Applications

> *Warning: ProRealTime does not provide investment advisory services. This document is not in any case personal or financial advice nor a solicitation to buy or sell any financial instrument. The example codes shown in this manual are for learning purposes only. You are free to determine all criteria for your own trading. Past performance is not indicative of future results. Any trading system may expose you to a risk of loss greater than your initial investment.*

Simple Examples

RSI 1 hour: Oversold

RSI is an overbought/oversold indicator that can predict trend reversals. In this example, we will scan for oversold securities. "Oversold" is defined by **RSI** < 30. This becomes more significant as the **RSI** approaches the 0 line.

We will construct a ProScreener that shows the securities where **RSI** < 30. This could be applied to an hourly timeframe.

Example: Oversold RSI

```
// Calculate the 14-period RSI
MyRSI = RSI[14]
// Filter: RSI < 30
Filter = MyRSI < 30
SCREENER[Filter] SORT BY MyRSI AS "RSI"
```

RSI 1 hour: Overbought

Let’s look for securities where **RSI** > 70 (overbought). The overbought signal becomes stronger as the **RSI** approaches 100. This screener searches for securities with **RSI** > 70. It could be applied to an hourly timeframe.

Example: Overbought RSI

```
// Calculate the 14-period RSI
MyRSI = RSI[14]
// Filter: RSI > 70
Filter = MyRSI > 70
SCREENER[Filter] SORT BY MyRSI AS "RSI"
```

Bullish moving average crossover

The bullish moving average crossover system is one of the most famous. It's based on the observation that a short-term moving average crosses over a long-term moving average when a bullish trend appears.

We are going to build a ProScreener that detects securities with a 20-period MA crossing over a 50-period MA.

Furthermore, we calculate the "momentum" between both MAs. If the number is close to 0, the cross is slower and less significant. The higher this number is, the stronger the crossing. A typical sign of a strong crossing is a relatively flat long-term MA and a very upward sloping short-term MA.

We will use this momentum as the sorting criteria of the screener, showing the highest values of `Speed` first.

Example: Detection of bullish crossover with the 20 and 50-period simple moving average

```
MaShort = Average[20]
MaLong = Average[50]
// Determine the relative speed of the short MA to the long MA, used for sorting later.
Speed = Momentum(MaLong - MaShort) * 100 / Close
// Detect the securities on which the crossover has just occurred.
Filter = MaShort CROSSES OVER MaLong
SCREENER[Filter] SORT BY Speed AS "Speed"
```

Bearish moving average crossover

This ProScreener shows us the securities for which the 20-period moving average crosses under the 50-period moving average.

Example: Detection of bearish crossover with the 20 and 50-period simple moving average

```
MaShort = Average[20]
MaLong = Average[50]
// Determine the relative speed of the short MA to the long MA
Speed = Momentum(MaShort - MaLong) * 100 / Close
// Detect the securities on which the crossover has just occurred, sort the results by Speed
Filter = MaShort CROSSES UNDER MaLong
SCREENER[Filter] SORT BY Speed AS "Speed"
```

More elaborate examples

RSI and bullish reversal

This system allows you to detect in real-time securities that are likely to make a bullish reversal (go from downtrend to uptrend).

Traditionally analysts looked at simple indicators without considering all of the characteristics of the situation (because they did not have all the tools enabling them to do better).

With ProScreener, there are no more technical limitations. We can create a real system. To begin, we look for an oversold `RSI` that is increasing.

We create a filter on these 2 conditions which is written with one line of ProBuilder code:

```
Filter = RSI < 30 AND Momentum[1](RSI) > 0
```

Now, we can look for the most interesting securities. These are the ones whose fall was most severe compared to their normal volatility.

Example: RSI and bullish reversal

```
// Filter the securities whose RSI is oversold and in a reversal measured by the momentum of the RSI.
// Filter: RSI < 30 and increasing
Filter = RSI < 30 AND Momentum[1](RSI) > 0
// Determine the force of the bearish trend
// Find the highest high of the last 20 bars.
Highest20 = highest[20](High)
// Determine the decline since this period
Decline = Highest20 - Close
// Determine the normal volatility of the security (median of true range over the last 3 bars)
NormalV = summation[3](TR) - highest[3](TR) - lowest[3](TR)
// Display results. Sorting Criteria: Decline/NormalV (preceding down trend force)
SCREENER[Filter] SORT BY Decline / NormalV AS "Down Trend Force"
```

RSI and bearish reversal

This screener searches for an overbought **RSI** that is making a bearish reversal. As before, we create a filter:

```
Filter = RSI > 70 AND Momentum[1](RSI) < 0
```

In the same way as the previous ProScreener, we search for the securities for which the increase in price was the most marked compared to the normal volatility of the security.

Example: RSI and bearish reversal

```
// Filter the securities whose RSI is overbought and in a reversal, measured by the momentum of the RSI.
// Filter: RSI > 70 and decreasing
Filter = RSI > 70 AND Momentum[1](RSI) < 0
// Determine the "force" of the bullish trend
// Find the lowest low of the last 20 bars
Lowest20 = lowest[20](Low)
// Determine the variation of price between the current price and the lowest low
Increase = Close - Lowest20
// Determine the normal volatility of the security (median of true range over the last 3 bars)
NormalV = summation[3](TR) - highest[3](TR) - lowest[3](TR)
// Display results. Sorting Criteria: Increase/NormalV (preceding up trend force)
SCREENER[Filter] SORT BY Increase / NormalV AS "Up Trend Force"
```

Bullish Engulfing with trend verification

It’s easy to use ProScreener to detect many candlestick patterns. The basic Top Movers tool also lets you detect candlestick patterns, but if you look for them with ProScreener, you can have more control over the definition of the candlestick form and also add additional conditions as we will see in this example.

Now let’s look at a ProScreener for the candlestick form "Bullish Engulfing". The bullish engulfing is one of the most powerful candlestick trend reversal patterns, but of course you must also look at the overall context (e.g. existence of a prior down trend or not).

We can define a Bullish Engulfing in this way:

- Previous candlestick where Close < Open
- Open of current candle < Close of previous candle
- Current candle Close > Open of previous candle

These 3 conditions can be expressed in one line of code:

```
Filter = Close[1] < Open[1] AND Open < Close[1] AND Close > Open[1]
```

For a valid bullish engulfing to be detected, we want to also verify that a bearish trend existed prior to the appearance of the bullish engulfing. One way of doing this would be to use part of the previous code and tighten the time horizon. With the RSI 14 as examined in the previous bar, it was ok to look at the fall over the last 20 bars.

When looking for a reversal in the form of Japanese candlesticks, experience has shown that 8 bars are sufficient (these structures are more responsive and theoretically limited to 7 candles).

Example: Bullish Engulfing with trend verification

```
// Determine the "force" of the preceding down trend.
// Find the highest high over the last 8 bars
High8 = highest[8](High)
// Determine the decline since this point
Decline = High8 - Close
// Determine the normal volatility of the security (median of true range over the last 3 bars)
NormalV = summation[3](TR) - highest[3](TR) - lowest[3](TR)
// Condition: Bullish Engulfing
Filter = Close[1] < Open[1] AND Open < Close[1] AND Close > Open[1]
// Sorting criteria: Decline/NormalV (preceding down trend force)
SCREENER[Filter] SORT BY Decline / NormalV AS "Down Trend Force"
```

Bearish Engulfing with trend verification

Let's look at another example with a Bearish Engulfing with trend verification.

A bearish engulfing can be defined as follows:

- Previous candle where Close > Open
- Open of current candle > Close of previous candle
- Close of current candle < Open of previous candle

These 3 conditions can be expressed in one line of code:

```
Filter = Close[1] > Open[1] AND Open > Close[1] AND Close < Open[1]
```

To detect the existence of a previous increasing trend, we use a code similar to the one above.

Example: Bearish engulfing with trend verification

```
// Determine the "force" of the preceding up trend.
// Find the lowest point over the last 8 bars
Low8 = lowest[8](Low)
// Determine the increase since this point
Increase = Close - Low8
// Determine the normal volatility of the security (median of true range over the last 3 bars)
NormalV = summation[3](TR) - highest[3](TR) - lowest[3](TR)
// Condition: Bearish Engulfing
Filter = Close[1] > Open[1] AND Open > Close[1] AND Close < Open[1]
// Display results. Sorting Criteria: Increase/NormalV (preceding up trend force)
SCREENER[Filter] SORT BY Increase / NormalV AS "Up Trend Force"
```

Triple bullish screen

This example ProScreener is composed of 3 conditions in several units of time:

- **Condition 1:** MACD weekly < 0 and increasing.
- **Condition 2:** Stochastic Daily < 30.
- **Condition 3:** Price is less than previous day's high or no more than 5% above it.

The results will respect the conditions in all of the units of time specified (weekly and daily).

Example: Triple bullish screen

```
// Condition 1 and 2: MACD weekly < 0 and increasing
TIMEFRAME(weekly)
MyMACD = MACD[12,26,9](Close)
c1 = MyMACD < 0 AND MyMACD > MyMACD[1]
// Condition 2: Daily Stochastic < 30
TIMEFRAME(daily)
MySTO = Stochastic[14,3](Close)
c2 = MySTO < 30
// Set Stop Level
MyStop = High[1]
// Criteria: Proximity to the high of the previous day
Criteria = (Close / MyStop - 1) * 100
// Condition 3: Price is less than previous day's high or no more than 5% above it.
c3 = Criteria < 5
SCREENER[c1 AND c2 AND c3] SORT BY Criteria
```

Triple bearish screen

This ProScreener is made of 3 conditions using several units of time:

- **Condition 1:** MACD Weekly > 0 and decreasing
- **Condition 2:** Stochastic Daily > 70
- **Condition 3:** Price is greater than previous day's low or no more than 5% below it

We look for securities whose prices are the closest to the sell stop level recommended by the system. This level is the lowest point of the previous day.

The ProScreener displays the securities which are below this level and are still within -5% of it.
The results displayed respect the specified conditions in weekly and daily views.

Example: Triple bearish screen

```
// Condition 1: MACD weekly > 0 and decreasing
TIMEFRAME(weekly)
MyMACD = MACD[12,26,9](Close)
c1 = MyMACD > 0 AND MyMACD < MyMACD[1]
// Condition 2: Stochastic daily > 70
TIMEFRAME(daily)
MySTO = Stochastic[14,3](Close)
c2 = MySTO > 70
// Set Stop Level
MyStop = Low[1]
// Sorting Criteria: Position of price with regard to stop level
Criteria = (Close / MyStop - 1) * 100
// Condition 3: Price is greater than the stop level or less than 5% below it
c3 = Criteria > -5
SCREENER[c1 AND c2 AND c3] SORT BY Criteria
```

You can visit our ProRealTime community on the **ProRealCode forum** to find **online documentation** and many examples.

Warning: ProRealTime does not provide investment advisory services. This document is not in any case personal or financial advice nor a solicitation to buy or sell any financial instrument. The example codes shown in this manual are for learning purposes only. You are free to determine all criteria for your own trading. Past performance is not indicative of future results. Any trading system may expose you to a risk of loss greater than your initial investment.

Glossary

A – B

CODE	SYNTAX	FUNCTION
ABS	ABS(a)	Mathematical function "Absolute Value" of a.
AccumDistr	AccumDistr(price)	Classical Accumulation/Distribution indicator.
ACOS	ACOS(a)	Mathematical function "Arc cosine" (returns an angle in degrees).
ADX	ADX[N]	Indicator Average Directional Index or "ADX" of n periods.
ADXR	ADXR[N]	Indicator Average Directional Index Rate or "ADXR" of n periods.
AND	a AND b	Logical AND Operator.
ArrayMax	ArrayMax($MyArray)	Returns the highest value of the array.
ArrayMin	ArrayMin($MyArray)	Returns the lowest value of the array.
ArraySort	ArraySort($MyArray, ASCEND)	Sort the table in ascending (ASCEND) or descending (DESCEND) order.
AroonDown	AroonDown[P]	Aroon Down indicator of n periods.
AroonUp	AroonUp[P]	Aroon Up indicator of n periods.
ATAN	ATAN(a)	Mathematical function "Arc tangent" (returns an angle in degrees).
AS	PRINT x AS "ResultName"	Instruction used to name a column displayed with the PRINT instruction.
ASIN	ASIN(a)	Mathematical function "Arc sine" (returns an angle in degrees).
Average	Average[N](price)	Simple Moving Average of n periods.
AverageTrueRange	AverageTrueRange[N](price)	"Average True Range" - True Range smoothed with the Wilder method.
BarIndex	BarIndex	Number of bars since the beginning of data loaded (in a chart in the case of a ProBuilder indicator or for a trading system in the case of ProBacktest or ProOrder).
BarsSince	BarsSince(condition, occurrence)	Returns the number of candles since the nth occurrence of the specified condition (n=0 means last occurrence and is the default, n=1 means second last occurrence)
BollingerBandWidth	BollingerBandWidth[N](price)	Bollinger Bandwidth indicator.
BollingerDown	BollingerDown[N](price)	Lower Bollinger band.
BollingerUp	BollingerUp[N](price)	Upper Bollinger band.
BREAK	(FOR/DO/BREAK/NEXT) or (WHILE/DO/BREAK/WEND)	Instruction forcing the exit of a FOR or WHILE loop.

C

CODE	SYNTAX	FUNCTION
CALL	myResult = CALL myFunction	Calls a user indicator to be used in the program you are coding.
CCI	CCI[N](price)	Commodity Channel Index indicator.
ChaikinOsc	ChaikinOsc[Ch1, Ch2](price)	Chaikin oscillator.
Chandle	Chandle[N](price)	Chande Momentum Oscillator.
ChandeKrollStopUp	ChandeKrollStopUp[Pp, Qq, X]	Chande and Kroll Protection Stop on long positions.
ChandeKrollStopDown	ChandeKrollStopDown[Pp, Qq, X]	Chande and Kroll Protection Stop on short positions.
Close	Close[N]	Closing price of the current bar or of the n-th last bar.
COLOURED	PRINT x COLOURED(R,G,B)	Set the font color of the corresponding cell in the PRINT instruction.
COS	COS(a)	Cosine function (‘a’ argument in degrees).
CROSSES OVER	a CROSSES OVER b	Boolean Operator checking whether a curve has crossed over another one.
CROSSES UNDER	a CROSSES UNDER b	Boolean Operator checking whether a curve has crossed under another one.
cumsum	cumsum(price)	Sums a certain price on the whole data loaded.
CurrentDayOfWeek	CurrentDayOfWeek	Represents the current day of the week.

CODE	SYNTAX	FUNCTION
CurrentHour	CurrentHour	Represents the current hour.
CurrentMinute	CurrentMinute	Represents the current minute.
CurrentMonth	CurrentMonth	Represents the current month.
CurrentSecond	CurrentSecond	Represents the current second.
CurrentTime	CurrentTime	Represents the current time (HHMMSS).
CurrentYear	CurrentYear	Represents the current year.
CustomClose	CustomClose[N]	Constant which is customizable in the settings window of the chart (default: Close).
Cycle	Cycle(price)	Cycle Indicator.

D

CODE	SYNTAX	FUNCTION
Date	Date[N]	Reports the date of each bar loaded on the chart.
DATEFORMAT	SCREENER(date DATEFORMAT)	Displays the values of the column as a date.
Daily	TIMEFRAME(Daily)	Defines the "daily" period for further calculations in the screener code.
Day	Day[N]	Day number at the end of the current candle.
Days	Days[N]	Counter of days since 1900.
DayOfWeek	DayOfWeek[N]	Day of the week of each bar.
DClose	DClose(N)	Close of the n-th day before the current one.
Decimals	Decimals	Returns the number of decimals of the instrument.
DEMA	DEMA[N](price)	Double Exponential Moving Average.
DHigh	DHigh(N)	High of the n-th day before the current bar.
DI	DI[N](price)	Refers to the Demand Index.
DIminus	DIminus[N](price)	Represents the DI- indicator.
DIplus	DIplus[N](price)	Represents the DI+ indicator.
DLow	DLow(N)	Low of the n-th day before the current one.
DO	See FOR and WHILE	Optional instruction in FOR loop and WHILE loop to define the loop action.
DOpen	DOpen(N)	Open of the n-th day before the current one.
DOWNTO	See FOR	Instruction used in FOR loop to process the loop with a descending order.
DPO	DPO[N](price)	Detrended Price Oscillator.

E – F – G – H

CODE	SYNTAX	FUNCTION
EaseOfMovement	EaseOfMovement[I]	Ease of Movement indicator.
ELSE	See IF/THEN/ELSE/ENDIF	Instruction used to call the second condition of If-conditional statements.
ELSEIF	See IF/THEN/ELSE/ENDIF	Stands for Else If (to be used inside of conditional loop).
EMV	EMV[N]	Ease of Movement Value indicator.
EQUITYFRAME	EQUITYFRAME("market", "ticker")	Condition related to a specific security in a specific market (ProScreener command only).
ENDIF	See IF/THEN/ELSE/ENDIF	Ending Instruction of IF-conditional statement.
EndPointAverage	EndPointAverage[N](price)	End Point Moving Average.
EstimatedVolume	EstimatedVolume	Estimated volume of the current bar (ProScreener command only).
EXP	EXP(a)	Mathematical Function "Exponential".
ExponentialAverage	ExponentialAverage[N](price)	Exponential Moving Average.
FILLCOLOR	PRINT x FILLCOLOR (r,g,b)	Used to set the background color of the corresponding cell in the PRINT table.

CODE	SYNTAX	FUNCTION
FOR/TO/NEXT	FOR i =a TO b DO a NEXT	FOR loop (processes all the values with an ascending (TO) or a descending order (DOWNTO)).
ForceIndex	ForceIndex(price)	Force Index indicator determines who controls the market (buyer or seller).
GetTimeFrame	GetTimeFrame	Returns the current period in seconds.
High	High[N]	High of the current bar or of the n-th last bar.
Highest	Highest[N](price)	Highest price over a number of bars to be defined.
HighestBars	HighestBars[N]	Returns the barindex delta between the current candle and the candle with the highest value.
HistoricVolatility	HistoricVolatility[N](price)	Historic Volatility (or statistic volatility).
Hour	Hour[N]	Represents the hour of each bar loaded in the chart.
Hours	TIMEFRAME(X Hours)	Defines the "X hour" period for further calculations in the screener code. (between 1 and 4, see Multi-period search).

I – J – K – L

CODE	SYNTAX	FUNCTION
IF/THEN/ENDIF	IF a THEN b ENDIF	Group of conditional instructions without second instruction.
IF/THEN/ELSE/ENDIF	IF a THEN b ELSE c ENDIF	Group of conditional instructions.
IntradayBarIndex	IntradayBarIndex[N]	Counts how many bars are displayed in one day on the whole data loaded.
IsSet	IsSet($MyArray[index])	Returns 1 or 0 if the value at the index of the array is defined or not.
KeltnerBandCenter	KeltnerBandCenter[N]	Central band of the Keltner indicator of N periods.
KeltnerBandDown	KeltnerBandDown[N]	Lower band of the Keltner indicator of N periods.
KeltnerBandUp	KeltnerBandUp[N]	Upper band of the Keltner indicator of N periods.
KijunSen	KijunSen[T,K,S]	Returns the KijunSen value of the Ichimoku indicator.
LastSet	LastSet($MyArray)	Returns the highest defined index of the array.
LinearRegression	LinearRegression[N](price)	Linear Regression indicator.
LinearRegressionSlope	LinearRegressionSlope[N](price)	Slope of the Linear Regression indicator.
LOG	LOG(a)	Mathematical Function "Neperian logarithm" of a.
Low	Low[N]	Low of the current bar or of the n-th last bar.
Lowest	Lowest[N](price)	Lowest price over a number of bars to be defined.
LowestBars	LowestBars[N]	Returns the barindex delta between the current candle and the candle with the lowest value.

M – N

CODE	SYNTAX	FUNCTION
MACD	MACD[S,L,Si](price)	Moving Average Convergence Divergence (MACD) in histogram.
MACDline	MACDLine[S,L](price)	MACD line indicator.
MassIndex	MassIndex[N]	Mass Index Indicator applied over N bars.
MAX	MAX(a,b)	Mathematical Function "Maximum".
MedianPrice	MedianPrice	Average of the high and the low.
MIN	MIN(a,b)	Mathematical Function "Minimum".
Minute	Minute	Designates the minute of the closing time of each bar in the historical data.
MOD	a MOD b	Mathematical Function "remainder of the division".
Momentum	Momentum[I]	Momentum indicator (close – close of the n-th last bar).
MoneyFlow	MoneyFlow[N](price)	MoneyFlow indicator (result between -1 and 1).
MoneyFlowIndex	MoneyFlowIndex[N]	MoneyFlow Index indicator.
Month	Month[N]	Represents the month of each bar loaded in the chart.

CODE	SYNTAX	FUNCTION
Monthly	TIMEFRAME(Monthly)	Defines the "monthly" period for further calculations in the screener code.
NEXT	See FOR/TO/NEXT	Ending Instruction of FOR loop.
NOT	NOT a	Logical Operator NOT.
NUMBERFORMAT	SCREENER(var NUMBERFORMAT)	Displays the values of the column as a number.

O

CODE	SYNTAX	FUNCTION
OBV	OBV(price)	On-Balance-Volume indicator.
ONCE	ONCE Variable = Value	Introduces a definition statement which will be processed only once.
Open	Open[N]	Open of the current candle or of the n-th previous candle.
OpenDay	OpenDay[N]	Opening day of the current candle or the nth previous candle.
OpenDayOfWeek	OpenDay[N]	Day of the week of the opening of the current candle or the nth previous candle.
OpenHour	OpenHour[N]	Opening time of the current candle or the nth previous candle.
OpenMinute	OpenMinute[N]	Opening minute of the current candle or the nth previous candle.
OpenMonth	OpenMonth[N]	Opening month of the current candle or the nth previous candle.
OpenSecond	OpenSecond[N]	Opening second of the current candle or the nth previous candle.
OpenTime	OpenTime[N]	Time (HHMMSS) of the opening of the current candle or the nth previous candle.
OpenTimestamp	OpenTimestamp[N]	UNIX opening timestamp of the current candle or the nth previous candle.
OpenWeek	OpenWeek[N]	Opening week of the current candle or the nth previous candle.
OpenYear	OpenYear[N]	Opening year of the current candle or the nth previous candle.
OR	a OR b	Logical Operator OR.

P – Q – R

CODE	SYNTAX	FUNCTION
PERCENTFORMAT	SCREENER(var PERCENTFORMAT)	Displays the values of the column as a percentage.
PriceOscillator	PriceOscillator[S,L](price)	Percentage Price oscillator.
PRINT	PRINT x	Displays the variable in its own window, useful for debugging.
PositiveVolumeIndex	PositiveVolumeIndex(price)	Positive Volume Index indicator.
POW	POW(N,P)	Returns the value of N at power P.
PVT	PVT(price)	Price Volume Trend indicator.
Quarterly	TIMEFRAME(Quarterly)	Defines the period "quarterly" for further calculations in the screener code.
R2	R2[N](price)	R-Squared indicator (error rate of the linear regression on price).
RANDOM	RANDOM(Min, Max)	Generates a random integer between Min and Max bounds (included).
Range	Range[N]	Returns the range (High – Low) of the current candle.
Repulse	Repulse[N](price)	Repulse indicator (measure the buyers and sellers force for each candle).
ROC	ROC[N](price)	Price Rate of Change indicator.
RSI	RSI[N](price)	Relative Strength Index indicator.
ROUND	ROUND(a)	Mathematical Function "Round a to the nearest whole number".

S

CODE	SYNTAX	FUNCTION
SAR	SAR[At,St,Lim]	Parabolic SAR indicator.
SARatdmf	SARatdmf[At,St,Lim](price)	Refers to the ATDMF Parabolic SAR indicator.
SCREENER	SCREENER[c](price)	Displays results of the ProScreener (ProScreener command only).
Second	Second[n]	Returns the second of the bar n periods before the current bar.

CODE	SYNTAX	FUNCTION
SIN	SIN(a)	Mathematical Function "Sine" (‘a’ argument in degrees).
SGN	SGN(a)	Mathematical Function "Sign of" a (it is positive or negative).
SMI	SMI[N,SS,DS](price)	Stochastic Momentum Index indicator.
SmoothedStochastic	SmoothedStochastic[N,K](price)	Smoothed Stochastic.
SORT BY	Screener(c1) SORT BY price	Filters or sorts the results of the screener (ProScreener command only).
SQUARE	SQUARE(a)	Mathematical Function "a Squared".
SQRT	SQRT(a)	Mathematical Function "Squared Root" of a.
STD	STD[N](price)	Statistical Function "Standard Deviation".
STE	STE[N](price)	Statistical Function "Standard Error".
STRINGFORMAT	SCREENER(var STRINGFORMAT)	Displays the values of the column without formatting.
Stochastic	Stochastic[N,K](price)	%K Line of the Stochastic indicator.
Summation	Summation[N](price)	Sums a certain price over the N last candles.
Supertrend	Supertrend[STF,N]	Super Trend indicator.

T

CODE	SYNTAX	FUNCTION
TAN	TAN(a)	Mathematical Function "Tangent" of a (‘a’ argument in degrees).
TEMA	TEMA[N](price)	Triple Exponential Moving Average.
TenkanSen	TenkanSen[T,K,S]	Returns the TenkanSen value of the Ichimoku indicator.
THEN	See IF/THEN/ELSE/ENDIF	Instruction following the first condition of "IF".
Ticksize	Ticksize	Minimum price variation of the current instrument.
Time	Time[N]	Represents the time (HHMMSS) of each bar loaded in the chart.
TimeSeriesAverage	TimeSeriesAverage[N](price)	Temporal series moving average.
Timestamp	Timestamp[N]	UNIX date of the close of the Nth previous candle.
TO	See FOR/TO/NEXT	Directional Instruction in the "FOR" loop.
Today	Today	Today's date (YYYYMMDD format).
TotalPrice	TotalPrice[N]	(Close + Open + High + Low) / 4.
TR	TR(price)	True Range indicator.
TriangularAverage	TriangularAverage[N](price)	Triangular Moving Average.
TRIX	TRIX[N](price)	Triple Smoothed Exponential Moving Average.
TypicalPrice	TypicalPrice[N]	Represents the Typical Price (Average of the High, Low and Close).

U – V – W

CODE	SYNTAX	FUNCTION
Undefined	a = Undefined	Sets the value of a variable to undefined.
UnSet	UnSet($MyArray)	Resets the data in the table.
Variation	Variation(price)	Difference between the close of the last bar and the close of the current bar in %.
Volatility	Volatility[S, L]	Chaikin volatility.
Volume	Volume[N]	Volume indicator.
VolumeAdjustedAverage	VolumeAdjustedAverage[N](Price)	Volume Adjusted Moving Average.
VolumeOscillator	VolumeOscillator[S,L]	Volume Oscillator.
VolumeROC	VolumeROC[N]	Volume of the Price Rate Of Change.
Weekly	TIMEFRAME(Weekly)	Defines the "weekly" timeframe for further calculations in the screener code.

CODE	SYNTAX	FUNCTION
WeightedAverage	WeightedAverage[N](price)	Represents the Weighted Moving Average.
WeightedClose	WeightedClose[N]	Average of (2 * Close), (1 * High) and (1 * Low).
WEND	See WHILE/DO/WEND	Ending Instruction of WHILE loop.
WHILE/DO/WEND	WHILE condition DO action WEND	WHILE loop.
WilderAverage	WilderAverage[N](close)	Represents Wilder Moving Average.
Williams	Williams[N](close)	Returns the %R of Williams indicator.
WilliamsAccumDistr	WilliamsAccumDistr(close)	Accumulation/Distribution of Williams Indicator.

X – Y – Z

CODE	SYNTAX	FUNCTION
XOR	a XOR b	Logical Operator exclusive OR.
Year	Year[N]	Returns the year of the bar n periods before the current bar.
Yearly	TIMEFRAME(Yearly)	Sets the Yearly timeframe for following instructions.
Yesterday	Yesterday[N]	Date of the day preceding the bar n periods before the current bar.
ZigZag	ZigZag[Zr](price)	Zig-Zag based on Elliott wave theory.
ZigZagPoint	ZigZagPoint[Zp](price)	Zig Zag calculated at Zp points.

Other

CODE	FUNCTION	CODE	FUNCTION
+	Addition Operator.	<>	Difference Operator.
-	Subtraction Operator.	<	Strict Inferiority Operator.
*	Multiplication Operator.	>	Strict Superiority Operator.
/	Division Operator.	<=	Inferiority Operator.
=	Equality Operator.	>=	Superiority Operator.

ProRealTime

www.prorealtime.com