ProRealTime

PROGRAMMING GUIDE

Market Scans (ProScreener)

TABLE OF CONTENTS

Introduction to ProScreener

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

- meeting one or multiple customizable conditions defined by you
- in one timescale view or in multiple different timescale views (ex: 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 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 with 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 uses the last 256 candlesticks (1024 for premium platforms) of the chosen timeframe to perform the calculations of your code.

The results of a ProScreener scan are displayed in a list whose results update automatically of the top 50 or 100 top financial instruments meeting the market scan conditions based on your selected sorting criteria.[1]

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 commands with ProScreener 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 them to our ProRealTime community on the **ProRealCode forum**, where you will also find **online documentation** with many examples.

We hope you will enjoy the manual!

---

[1] *Depending on the ProRealTime version used*

Chapter I: Fundamentals

Accessing ProScreener

You can access ProScreener by clicking on "Display" and then "ProScreener as shown below:
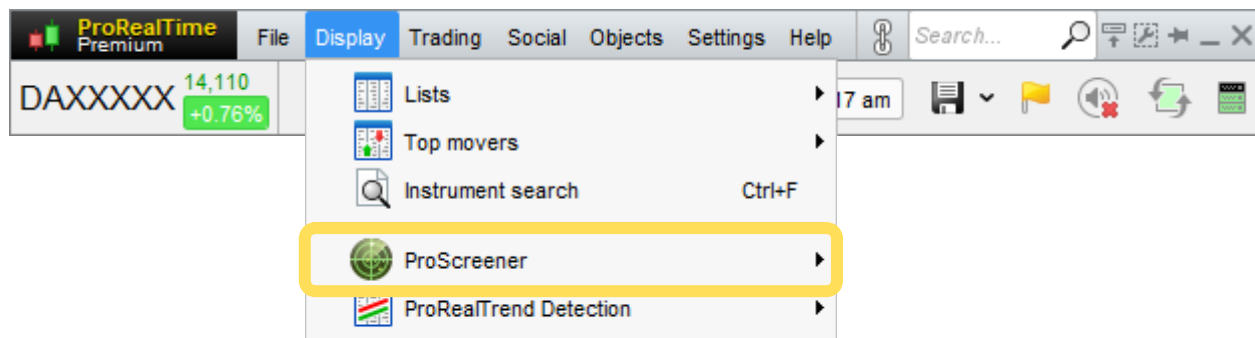
The ProScreener window will then appear, which allows you to select a ProScreener that is already created or to create a new ProScreener. To do this, click the wrench key at the top of the window to open the "Manage ProScreeners window" and then "New". From this window, you can choose 2 ways to create a new ProScreener:

- "Assisted Creation" mode which allows you to define the conditions of the ProScreener by simply point-and-clicking on one or more charts. To learn how to do this very easy assisted creation, 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
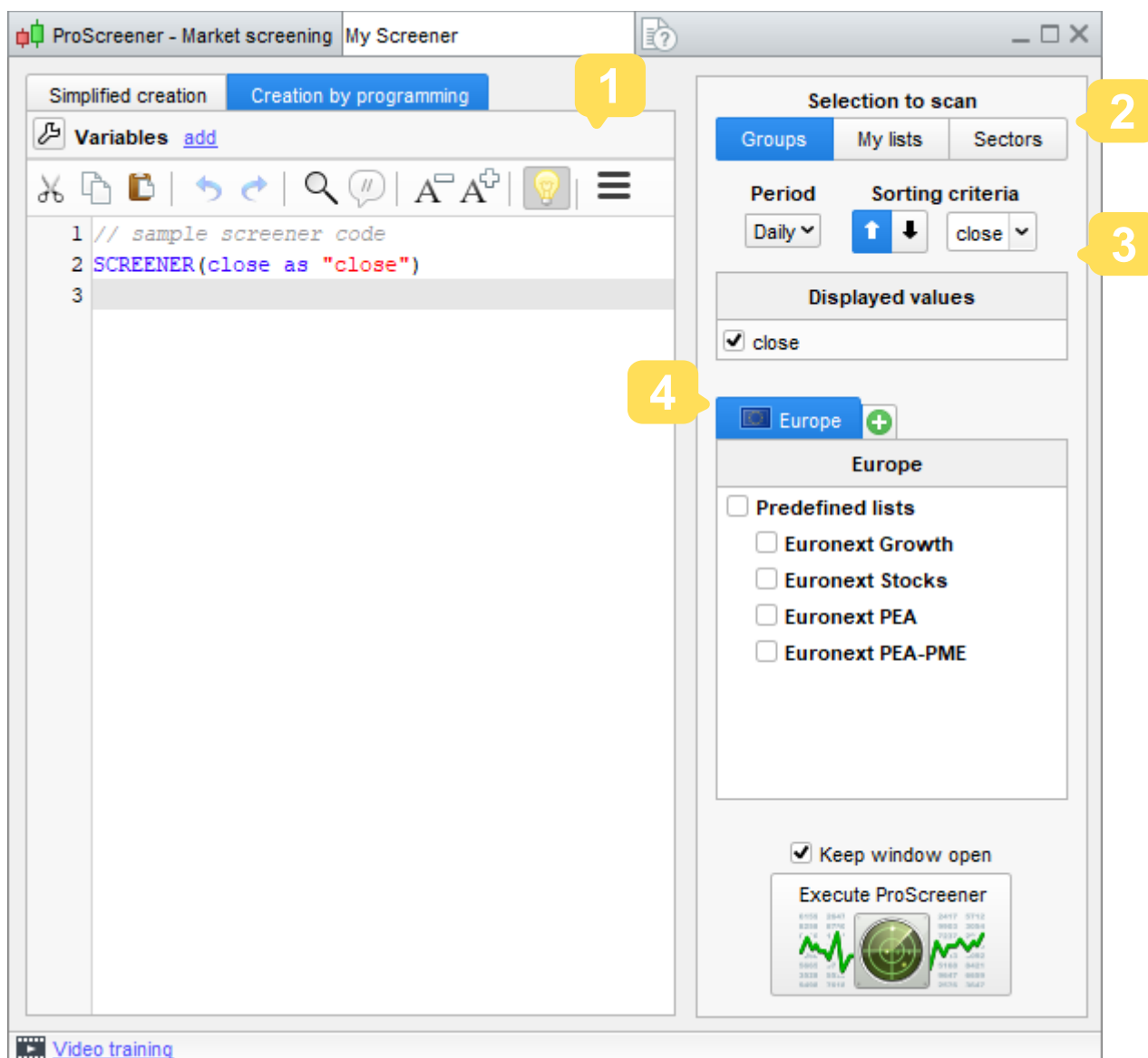
The **first section (Programming of ProScreener)**, allows you to:

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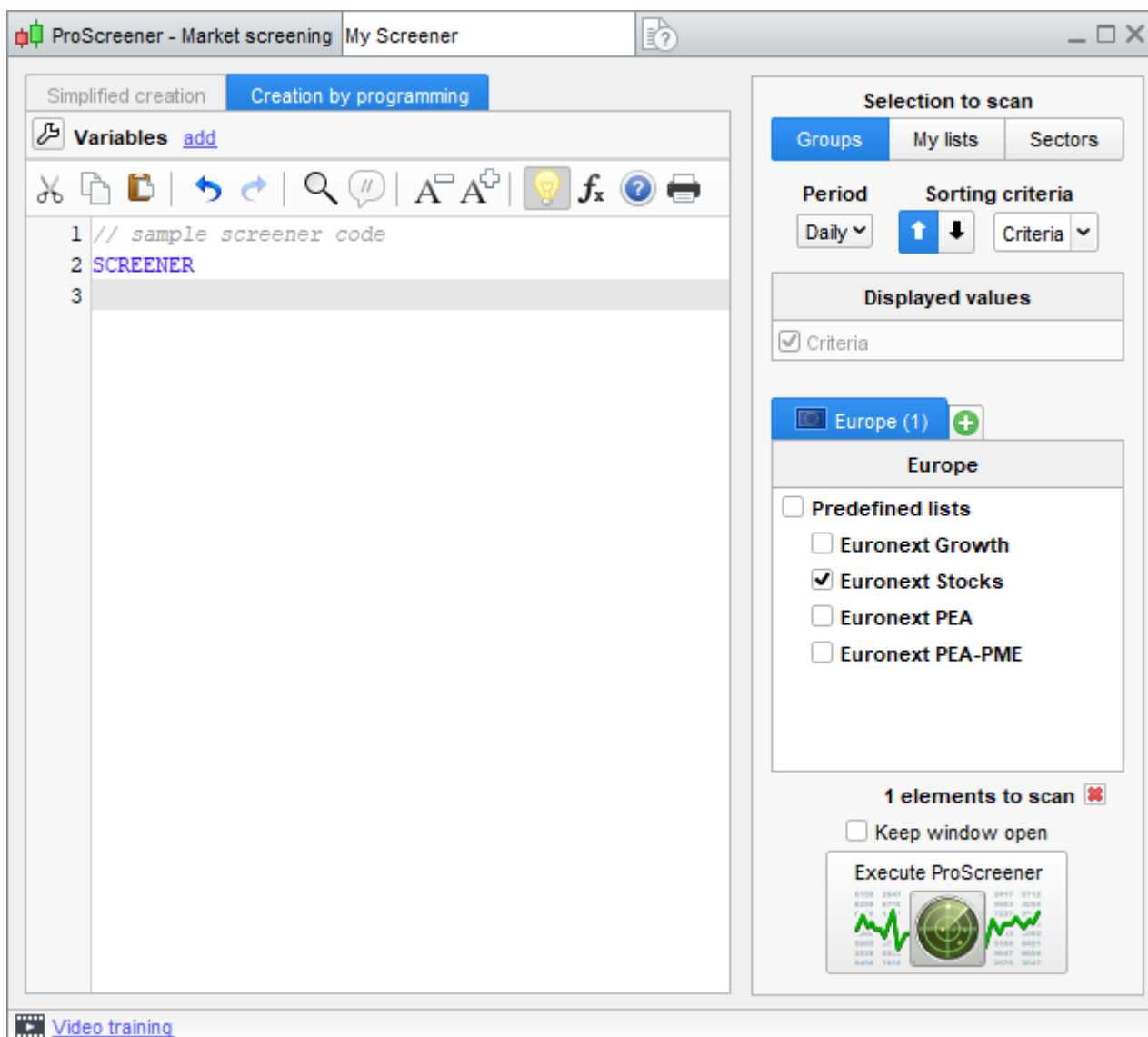
> ⭐ The "**SCREENER**" command defines what conditions to scan for.

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 criterias for the results. With conditions that have very little restrictions, the results of a ProScreener may be very numerous (several thousand on the NASDAQ for example). In this case, you need to filter the results with one or multiple chosen criterias and choose to display:

- the results with the 50 highest values (of the sorting criteria(s))
- the results with the 50 lowest values (of the sorting criteria(s))

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	%Chg	Last	Volume
PROSUS	+2.12%	64.91	309,339
AIRBUS	+1.36%	113.26	168,731
BNP PARIBAS ACT.A	+0.78%	54.22	212,609
ING GROEP N.V.	+0.75%	11.614	2,537k
SCHNEIDER ELECTRIC	+1.33%	132.92	101,345
KERING	+1.98%	481.30	31,311
ADYEN	+1.17%	1,315.2	12,060
ARCELORMITTAL SA	+0.84%	25.285	499,819
LVMH	+2.34%	689.6	70,078
AXA	+0.38%	26.530	429,335
SANOFI	+0.10%	91.07	205,963
LOREAL	+1.13%	337.00	40,538
TOTALENERGIES	+0.90%	59.74	1,096k
AIR LIQUIDE	+0.73%	134.68	97,372
ASML HOLDING	+0.35%	518.7	104,078
SHELL PLC	+0.67%	27.235	2,354k
AIR FRANCE -KLM	+2.68%	1.3025	6,978k
SAFRAN	+1.28%	119.06	69,495
ASR NEDERLAND	-0.57%	45.30	59,205
AKZO NOBEL	+1.01%	64.04	62,984
EUROFINS SCIENT.	+0.88%	66.76	93,887
WORLDLINE	+0.57%	37.15	62,391
JUST EAT TAKEAWAY	+1.18%	19.694	278,341
RELX	+0.31%	26.23	42,993
AHOLD DEL	+0.23%	27.850	236,191

Nbr of matches: 1,009 (up to 100 displayed) Historical data:1024

Using Top Movers (predefined scans)

In addition to ProScreener, ProRealTime provides the Top Movers scan tool. Top Movers scans the market with predefined 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

Name	%Chg Yest.	Last
GRAB HOLDINGS LTD.	+6.86%	3.27(c)
VIPER ENERGY PARTNERS LP	+5.95%	32.22(c)
APA CORP.	+5.73%	47.25(c)
PATTERSON-UTI ENERGY INC.	+5.25%	16.85(c)
CREDIT ACCEPTANCE	+5.14%	465.23(c)
CHORD ENERGY CORP.	+5.02%	139.83(c)
HIGHPEAK ENERGY INC.	+4.64%	22.11(c)
GRUPO AEROPORTUARIO DEL CENTRO NORTE...	+4.51%	65.62(c)
PDC ENERGY INC.	+4.42%	65.37(c)
GOLAR LNG LTD.	+4.01%	23.33(c)
QURATE RETAIL INC. SERIES B	+4.00%	6.24(c)
STONECO LTD.	+3.90%	9.60(c)
LIBERTY MEDIA CORP.	+3.78%	41.17(c)
PLAINS GP HLD.	+3.74%	12.47(c)
CHESAPEAKE ENERGY	+3.72%	97.37(c)
XP INC.	+3.67%	15.52(c)
AVIS BUDGET GROUP INC.	+3.67%	161.41(c)
HERTZ GLOBAL HOLDINGS INC	+3.54%	15.21(c)
PETCO HEALTH AND WELLNESS CO.	+3.44%	9.31(c)
PAYCHEX INC.	+3.41%	116.01(c)
WEATHERFORD INTERNATIONAL PLC	+3.36%	50.15(c)
VIASAT INC.	+3.29%	31.74(c)
ACADEMY SPORTS AND OUTDOORS	+3.28%	51.40(c)
PLAINS ALL AMERICAN PIPELINE L.P.	+3.17%	11.71(c)
DIAMONDBACK ENERGY INC.	+3.12%	137.03(c)
CHARTER COMMUNICATIONS INC.	+3.10%	333.92(c)

Nbr of matches: 100

To learn more about the Top Movers predefined scanning tools, please watch the related videos on our video tutorials page including "**Top movers scanner: detect price and volume top movers or opening gaps**"

Chapter II: Programming ProScreener

In this chapter we will illustrate the 5 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 what conditions to scan for. The last line of every ProScreener must begin with "**SCREENER**". Like the return function in ProBuilder, it can be followed by other commands.

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

$$\text{SCREENER[Condition]}$$

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:

$$\text{SCREENER[Condition1 AND Condition2]}$$
OR
$$\text{SCREENER[Condition1 OR Condition2]}$$

Example:

Lets search for securities for which 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 a 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 a sorting criteria is:

$$SCREENER[c1](TypicalPrice)$$

To use a predefined 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**.

$$MyRSI = CALL \; "RSI" \; [14]$$
$$SCREENER[c1](MyRSI)$$

Example:

Let's find securities whose volume is more than 50000. Then lets 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 its greater than 70 and -1 if its less than 30. We will display the result of the binary indicator in the sorting criteria column.

```
c1 = Volume > 50000
IF RSI[20](Close) > 70 THEN
   Criteria = 1
ELSIF RSI[20](Close) < 30 THEN
   Criteria = -1
ENDIF
SCREENER[c1](Criteria AS "OBOS")
```

Name	OBOS	%Chg	Last	Volume
TRACTOR SUPPLY CO.	1	+1.88%	217.06(c)	728,575
TEXAS INSTRUMENTS INC.	1	-0.19%	164.38(c)	2,597k
ANALOG DEVICES INC.	1	+0.05%	163.84(c)	1,498k
BROADCOM INC.	1	-0.12%	552.43(c)	1,376k
LULULEMON ATHLETICA INC.	1	-0.44%	311.54(c)	933,601
MARRIOTT INTERNATIONAL	1	+0.92%	148.58(c)	1,068k
ON SEMICONDUCTOR	1	+0.43%	62.39(c)	3,351k
FIRST SOLAR INC.	1	-0.09%	156.75(c)	1,028k
CADENCE DESIGN SYSTEM...	1	-0.19%	161.45(c)	709,726
CHECK POINT SOFTWARE ...	1	+0.40%	126.97(c)	328,934
AUTOMATIC DATA PROCE...	1	+0.74%	240.94(c)	1,050k
APPLIED MATERIALS INC.	1	-0.39%	97.22(c)	4,754k
ULTA BEAUTY INC.	1	-0.14%	456.40(c)	361,205
STEEL DYNAMICS INC	1	-0.25%	102.59(c)	1,424k

Nbr of matches: 146 (up to 100 displayed)   Historical data:1024

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

$$\texttt{SCREENER[c1] SORT BY TypicalPrice}$$

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

$$\texttt{SCREENER[c1](TypicalPrice AS "typical price")}$$

Use multiple filter criterias

The screener allows you to define multiple criteria 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 criterias within the SCREENER instructions will have to be separated by commas.
Exemple:

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

This functionality can only be available if your code contains not more than one SCREENER instruction.

Format results data

The screener also gives the possibility 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 that the current candlestick started

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

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 = 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

Its possible to do a ProScreener with multiple conditions and on multiple time periods. This enables you to check for example if your a condition is true on several different charting timeframes which you may want to look at (ex: 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

The conditions 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.

This example has the goal of detecting overbought values on the weekly timeframe with a bearish trend on the 30-minute timeframe. 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-security 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 a 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 correlation between a security and other securities in the same market. 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 = SORT BY CloseVal / CloseInd * 100
RelativeStrength = (Ratio - Ratio[1]) * 100
SCREENER SORT BY RelativeStrength AS "RelativeStrength"
```

Chapter III: Practical Applications

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 is 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. Its 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 "momentum" which we will define as the difference between the 2 Mas in order to learn the power of the cross. 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 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 TrendForce first.

Example: Detection of bullish crossover with the 20 and 50-period simple moving average
```
// Calculate the short-term ma
MaShort = Average[20]
// Calculate the long-term ma
MaLong = Average[50]
// Determine the relative speed of the short MA to the long MA
Speed = Momentum(MaLong - MaShort) * 100 / Close
// Detect the securities on which the crossover has just occurred, sort the results by Speed
Filter = MaShort CROSSES OVER MaLong
SCREENER[Filter] SORT BY Speed AS "Dynamic"
```

Bearish moving average crossing

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

We also use "momentum" in this example in the same way as in the previous example.

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

```
// Calculate the short-term ma
MaShort = Average[20]
// Calculate the long-term ma
MaLong = Average[50]
// Determine the relative speed of the short MA to the long MA
Speed = Momentum(MaShort - MaLong) * 100 / Close
// Select securities when the cross occurs
Filter = MaShort CROSSES UNDER MaLong
SCREENER[Filter] SORT BY Speed AS "Dynamic"
```

More elaborate examples

RSI and bullish reversal

This system allows you to detect in real-time securities susceptible to make a bullish reversal (go from down trend to up-trend).

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 oversold 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) - lest[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 lets look at a ProScreener for the candlestick form "Bullish Engulfing". The bullish engulfing is one of the more powerful candlestick trend reversal patterns, but of course you must also look at the overall context (ex: 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 is 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 8bars
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 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 BYCriteria
```

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

Glossary

A

CODE	SYNTAX	FUNCTION
ABS	ABS(a)	Mathematical function "Absolute Value" of a
AccumDistr	AccumDistr(close)	Classical Accumulation/Distribution indicator
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[N]	Aroon Down indicator of n periods
AroonUp	AroonUp[N]	Aroon Up indicator of n periods
ATAN	ATAN(a)	Mathematical function "Arctangent" of a
AS	RETURN result AS "ResultName"	Instruction used to name a line or indicator displayed on chart. Used with "RETURN"
Average	Average[N](price)	Simple Moving Average of n periods
AverageTrueRange	AverageTrueRange[N](price)	"Average True Range" - True Range smoothed with the Wilder method

B

CODE	SYNTAX	FUNCTION
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)
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 FOR loop 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) or CCI[N]	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	RETURN Result COLOURED(R,G,B)	Colors a curve with the color you defined using the RGB convention
COS	COS(a)	Cosine Function
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
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
Day	Day[N]	Reports the day of each bar loaded in the chart
Daily	TIMEFRAME(Daily)	Defines the "daily" period for further calculations in the screener code.
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
DEMA	DEMA[N](price)	Double Exponential Moving Average
DHigh	DHigh(N)	High of the n-th bar before the current bar
DI	DI[N](price)	Represents DI+ minus DI-
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

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/ELSIF/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 of a

CODE	SYNTAX	FUNCTION
EstimatedVolume	EstimatedVolume	Estimated volume of the current bar (ProScreener command only)
EXP	EXP(a)	Mathematical Function "Exponential"
ExponentialAverage	ExponentialAverage[N](price)	Exponential Moving Average

F - G

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)

H

CODE	SYNTAX	FUNCTION
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
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

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.

L

CODE	SYNTAX	FUNCTION
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

M

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
Monthly	TIMEFRAME(Monthly)	Defines the "monthly" period for further calculations in the screener code.

N

CODE	SYNTAX	FUNCTION
NEXT	See FOR/TO/NEXT	Ending Instruction of FOR loop
NOT	Not A	Logical Operator NOT
NUMBERFORMAT	SCREENER(number NUMBERFORMAT)	Displays the values of the column as a number

O

CODE	SYNTAX	FUNCTION
OBV	OBV(price)	On-Balance-Volume indicator
ONCE	ONCE VariableName = VariableValue	Introduces a definition statement which will be processed only once
Open	Open[N]	Open of the current bar or of the n-th last bar
OR	a OR b	Logical Operator OR

P - Q

CODE	SYNTAX	FUNCTION
PERCENTFORMAT	SCREENER(var PERCENTFORMAT)	Displays the values of the column as a percentage
PriceOscillator	PriceOscillator[S,L](price)	Percentage Price oscillator
PositiveVolumeIndex	PriceVolumeIndex(price)	Positive Volume Index indicator
PVT	PVT(price)	Price Volume Trend indicator
Quarterly	TIMEFRAME(Quarterly)	Defines the period "quarterly" for further calculations in the screener code.

R

CODE	SYNTAX	FUNCTION
R2	R2[N](price)	R-Squared indicator (error rate of the linear regression on price)
Range	Range[N]	calculates the Range (High minus Low)
//	comment	Introduces a remark (not taken into account by the code)
Repulse	Repulse[N](price)	Repulse indicator (measure the buyers and sellers force for each candlestick)

CODE	SYNTAX	FUNCTION
RETURN	RETURN Result	Instruction returning the result
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)	Display results of the ProScreener (ProScreener command only)
SIN	SIN(a)	Mathematical Function "Sine"
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	SORT BY	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(number 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 candlesticks
Supertrend	Supertrend[STF,N]	Super Trend indicator

T

CODE	SYNTAX	FUNCTION
TAN	TAN(a)	Mathematical Function "Tangent" of a
TEMA	TEMA[N](price)	Triple Exponential Moving Average

CODE	SYNTAX	FUNCTION
THEN	See IF/THEN/ELSE/ENDIF	Instruction following the first condition of "IF"
Time	Time[N]	Represents the time of each bar loaded in the chart
TimeSeriesAverage	TimeSeriesAverage[N](price)	Temporal series moving average
TIMEFRAME	TIMEFRAME(default)	Allows you to write conditions for your ProScreener in different timeframes (ProScreener command only)
TO	See FOR/TO/NEXT	Directional Instruction in the "FOR" loop
Today	Today[N]	Date of the bar n-periods before the current bar
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

CODE	SYNTAX	FUNCTION
Undefined	a = Undefined	Sets a the value of a variable to undefined
UnSet	UnSet($MyArray)	Resets the data in the table

V

CODE	SYNTAX	FUNCTION
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
VolumeOscillator	VolumeOscillator[S,L]	Volume Oscillator
VolumeROC	VolumeROC[N]	Volume of the Price Rate Of Change

W

CODE	SYNTAX	FUNCTION
Weekly	TIMEFRAME(Weekly)	Defines the "weekly" timeframe for further calculations in the screener code.
WeightedAverage	WeightedAverage[N](price)	Represents the Weighted Moving Average
WeightedClose	WeightedClose[N]	Average of (2 * Close), (1 * High) and (1 * Low)

CODE	SYNTAX	FUNCTION
WEND	See WHILE/DO/WEND	Ending Instruction of WHILE loop
WHILE/DO/WEND	WHILE (condition) DO (action) WEND	WHILE loop
WilderAverage	WilderAverage[N](price)	Represents Wilder Moving Average
Williams	Williams[N](close)	%R Williams indicator
WilliamsAccumDistr	WilliamsAccumDistr(price)	Accumulation/Distribution of Williams Indicator

X

CODE	SYNTAX	FUNCTION
XOR	a XOR b	Logical Operator eXclusive OR

Y

CODE	SYNTAX	FUNCTION
Year	Year[N]	Year of the bar n periods before the current bar
Yearly	TIMEFRAME(Yearly)	Defines the "yearly" timeframe for further calculations in the screener code.
Yesterday	Yesterday[N]	Date of the day preceding the bar n periods before the current bar

Z

CODE	SYNTAX	FUNCTION
ZigZag	ZigZag[Zr](price)	Represents the Zig-Zag indicator introduced in the Elliott waves theory
ZigZagPoint	ZigZagPoint[Zp](price)	Represents the Zig-Zag indicator in the Elliott waves theory calculated on 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 SOFTWARE