

Developer KIT

**ChartNet**.nl

**Indicator  
Builder**

2004 editie

1.03



<a href="#">ChartNet Indicator Builder.....</a>	<a href="#">2</a>
<a href="#">Termen in de taal.....</a>	<a href="#">3</a>
<a href="#">Koersen termen.....</a>	<a href="#">3</a>
<a href="#">Afgeleide termen van de koersen.....</a>	<a href="#">4</a>
<a href="#">Tijd termen.....</a>	<a href="#">5</a>
<a href="#">Minute, Hour, DayOfWeek, Month, BarIndex.....</a>	<a href="#">6</a>
<a href="#">Ongedefinieerde termen.....</a>	<a href="#">8</a>
<a href="#">Expressies en operatoren.....</a>	<a href="#">9</a>
<a href="#">Toegang tot voorgaande bars.....</a>	<a href="#">9</a>
<a href="#">Tweevoudige wiskundige operatoren.....</a>	<a href="#">12</a>
<a href="#">Enkelvoudige wiskundige operatoren.....</a>	<a href="#">14</a>
<a href="#">Vergelijkings operatoren.....</a>	<a href="#">20</a>
<a href="#">Logical operatoren.....</a>	<a href="#">23</a>
<a href="#">Taal structuur, instructies.....</a>	<a href="#">25</a>
<a href="#">Korte omschrijving.....</a>	<a href="#">25</a>
<a href="#">Invoegen van commentaar.....</a>	<a href="#">26</a>
<a href="#">Variabelen, parameters en definities.....</a>	<a href="#">27</a>
<a href="#">Controle structuren.....</a>	<a href="#">28</a>
<a href="#">Gebruik van een gebruikers functie.....</a>	<a href="#">34</a>
<a href="#">Return van een gebruikers functie.....</a>	<a href="#">35</a>
<a href="#">Functie overzicht.....</a>	<a href="#">36</a>
<a href="#">Moving averages.....</a>	<a href="#">36</a>
<a href="#">IBuilder functies.....</a>	<a href="#">45</a>
<a href="#">ChartNet technische indicatoren lijst.....</a>	<a href="#">49</a>

## ChartNet Indicator Builder

IBuilder is de programmeertaal van ChartNet. Het stelt u in staat om custom indicatoren te bouwen en helpt u binnenkort bij het maken van trading strategien en custom alerts.

IBuilder lijkt veel op de programmeertaal BASIC. Het is erg eenvoudig te gebruiken en is zeer compleet voor wat betreft de mogelijkheden. U kunt dus eigen indicatoren maken gebruik makend van de koersen die door ChartNet worden verstrekt:

- Openenings koers van elke bar/candle
- Slot koers van elke bar/candle
- Hoogste koers van elke bar/candle
- Laagste koers van elke bar/candle
- Volume van elke bar/candle

Bars/candles zoals te zien in de ChartNet Software.

Custom IBuilder indicators kunnen zichtbaar worden gemaakt in de ChartNet software.

U kunt er zoveel maken als u wil.

De IBuilder parser houdt rekening met de waarden van de eerste tot de laatste bar/candle en met uw eigen formules worden uw eigen indicatoren in real time getoond.

**De basis taal van IBuilder is Engels. Alle voorbeelden zijn dus zo natuurgetrouw mogelijk in het Engels.**

## Termen in de taal

### *Koersen termen*

#### **Open, High, Low, Close, Volume**

Zijn respectievelijk:

- Openings koers van de huidige bar
- Hoogste koers van de huidige bar
- Laagste koers van de huidige bar
- Slot koers van de huidige bar
- Volume van de huidige bar

### **Omschrijving**

Dit zijn de basis termen van de technische analyse. U kunt deze combineren om een bepaalde aspect van de informatie van de financiële markten te benadrukken.

#### Voorbeeld open, high, low, close, volume

```
REM Easy arithmetic calculation
```

```
myIndicator = open + high + low
```

```
myIndicator = volume * (myIndicator - close)
```

```
REM returns 'myIndicator' as output of Met de functie
```

```
RETURN myIndicator
```

## Afgeleide termen van de koersen

### Range, TypicalPrice, WeightedClose, MedianPrice, TotalPrice

Zijn respectievelijk:

- *Range* Het verschil tussen **high** and **low**
- *TypicalPrice* Gemiddelde tussen **high**, **low** en **close**
- *WeightedClose* Gewogen gemiddelde tussen **high** (weight 1) **low**(weight 1) en **close**(weight 2)
- *MedianPrice* Gemiddelde tussen **high** and **low**
- *TotalPrice* Gemiddelde tussen **open**, **high**, **low** and **close**

### Omschrijving

De termen benadrukken enkele aspecten van de psychologie van de financiële markten in de huidige bar:

De *Range* laat de volatiliteit van de huidige bar zien. Dit geeft aan hoe nerveus de beleggers zijn

De *WeightedClose* richt zich op de het belang van de laatste bar (is zelfs zeer belangrijk bij dag- of weekbars)

De termen *TypicalPrice* en *TotalPrice* benadrukken de intraday psychologie van de financiële markten, omdat ze van de drie of vier belangrijkste prijzen van de huidige bar uit gaan.

*MedianPrice* richt zich op het gebruik van het middelingsconcept in plaats van gemiddeldenconcept. Dit is bruikbaar bij het maken van theoretische modellen die geen rekening houden met de psychologie van de belegger.

#### Voorbeeld Range, TypicalPrice, WeightedPrice, MedianPrice, TotalPrice

REM Indicator that emphasizes on investors emotivity

InvestorsEmotivity = **Range** \* Abs(**WeightedClose** – **MedianPrice**)

REM returns 'InvestorsEmotivity' as output of Met de functie

RETURN InvestorsEmotivity

## *Tijd termen*

### **Time, Date**

Uurf en datum van het slot van de huidige bar.

### **Omschrijving**

- **Time**           Uur HHMMSS
- **Date**           Datum YYYYMMDD

Deze functies stellen u in staat om de veranderingen van de dag tussen twee bars of een specifieke datum te detecteren. In de volgende pagina's, leggen we u uit hoe u intraday delen en andere tijdseenheden kunt gebruiken.

#### **Voorbeeld Time**

REM Detecting 14 o'clock (local hour)

RETURN **Time** = 140000

#### **Voorbeeld 1 Date**

REM Décting the 11.09.2001

RETURN **Date** = 20010911

#### **Voorbeeld 2 Date**

REM Detecting the changes of day

RETURN **Date** > **Date**[1]

## ***Minute, Hour, DayOfWeek, Month, BarIndex***

Deze functies helpen u om intraday dalen en andere tijdseenheden te gebruiken.

### **Omschrijving**

- Minute de minuut van het slot van de huidige bar
- Hour het uur van het slot van de huidige bar
- Day de dag van het slot van de huidige bar
- Month de maand van het slot van de huidige bar
- Year het jaar van de huidige bar
- DayOfWeek de dag van de huidige bar (0=zondag, 1=maandag ... 6 = zaterdag)

Tijd termen zijn worden niet direct gebruikt voor technische analyse, hoewel beleggers weten dat sommige tijden op een dag of sommige dagen van van een jaar belangrijker zijn dan andere.

### **Voorbeeld Minute, Hour, DayOfWeek, Month, BarIndex**

```
REM A measurement of the professionals power ( on daily bars)
```

```
REM 1st supposition: There are twice more professionals on Wednesday, Thursday and Friday
```

```
REM 2nd supposition: They control closing price
```

```
IF DayOfWeek < 3 THEN
```

```
  Weight = 1
```

```
ELSE
```

```
  Weight = 2
```

```
ENDIF
```

```
REM Builds a cumulative indicator based on « professionals power »
```

```
ProPower = ProPower + Weight * (Close - MedianPrice)
```

```
REM returns 'ProPower' as output of Met de functie
```

```
RETURN ProPower
```

## BarIndex, IntradayBarIndex, Days

Tellers van bars en dagn.

### Omschrijving

- **BarIndex** de index van de huidige bar tussen de beschikbare bars
- **IntradayBarIndex** de index van de huidige bar tussen de bars van de dag
- **Days** de index van de dag vanaf 1900

Deze variabelen stellen u in staat om systemen te schrijven die onafhankelijk zijn van intervallen tussen 2 bars.

### Voorbeeld IntradayBarIndex

```
REM This indicator is not null when we change of day (en intraday)
```

```
RETURN IntradayBarIndex = 0
```

## Ongedefinieerde termen

### Undefined

Stelt een niet gedefinieerde waarde voor

### Omschrijving

Wanneer we een technische indicator bouwen, gebruiken we vaak bars uit het verleden. In zulke gevallen is het onmogelijk om de eerste warden van een dergelijke calculator te berekenen

De term *undefined* staat ChartNet niet toe om waarden weer te geven in plaats van het getal 0 dat standaard wordt weergegeven.

### Voorbeeld UNDEFINED

```
REM A non optimized calculation of a 20 bars moving average

IF BarIndex < 19 THEN

    myMA = undefined

ELSE

    myMA = 0

    FOR i = 0 TO 19 DO
        myMA = myMA + close[i]    // lecture of closing prices of previous bars
    NEXT

    myMA = myMA / 20

ENDIF

REM Returns 'myMA' as output of Met de functie

RETURN myMA
```

## Expressies en operatoren

### *Toegang tot voorgaande bars*

[.]

Toegang tot de termen of variabelen van voorafgaande bars

### Omschrijving

expressie[*count*]

De waarden berekend door IBuilder worden voor elke bar opgeslagen. U kunt dus toegang hebben tot deze waarden vanaf latere bars. Dit is zeer belangrijk omdat technische analyse gebaseerd is op het feit dat financiële markten een geheugen hebben.

#### Voorbeeld1 [.]

REM Calculation of the moving average value of the current bar, using closing prices:

```
myMA = average[30](close)
```

REM Difference between myMA current value and myMA previous value:

```
myGrowth = myMA - myMA[1]
```

REM Returns 'myGrowth' as output of Met de functie. You may display it with a histogram view

```
RETURN myGrowth
```

#### Voorbeeld2 [.]

REM Voorbeeld of multiple calculation

```
RETURN (open+close[1])[3]
```

REM Such function returns for each bar, the calculated value of the third previous bar when  
REM adding its opening price and its previous closing price

REM We thus get the addition of the opening price of the third previous bar  
REM and the closing price of the fourth previous bar

#### Waarschuwing

De berekening van een variabele past uitsluitend de huidige waarde aan zonder de voorgaande warden te veranderen. In het bovenstaande voorbeeld geldt: de voorafgaande waarden of de varabele  $x$  zijn gelijk aan 0.

## Voorbeeld3 [.]

```
REM Voorbeeld of a bad use of the [.]
```

```
x = open
```

```
REM A common mistake is to think that we get the opening price value of previous bars
```

```
myIndicator = x[3]
```

```
REM Actually, we get the x value calculated in the past
```

```
REM and such value is equal to 0 since of the below instruction :
```

```
x = 0
```

```
REM Returns 'myIndicator' as output of Met de functie
```

```
RETURN myIndicator
```

### **Opmerking**

De IBuilder interpreter gebruikt de waarden van alle bars, van de eerste tot de laatste en voert de formule uit om de waarden van de huidige bar van indicatoren te berekenen

## *Twvoudige wiskundige operatoren*

### **+ - \* / MOD**

Basis wiskundige berekeningen.

### **Omschrijving**

$a + b$

$a - b$

$a * b$

$a / b$

$a \text{ MOD } b$

Deze operatoren stellen u in staat alle basis wiskundige berekeningen te maken:

- optellen (+)
- aftrekken (-)
- vermenigvuldigen (\*)
- delen (/)
- rest van een deling (**MOD**)

#### **Voorbeeld1 + - \* /**

```
REM Such program returns an arithmetic expression using price terms
```

```
myIndicator = open * volume + (close - high) / low
```

```
REM Returns 'myIndicator' as output of Met de functie
```

```
RETURN myIndicator
```

#### **Voorbeeld2 MOD**

```
REM Creation of a binary oscillator
```

```
myOscillator = BarIndex MOD 2
```

```
REM Returns 'myOscillator' as output of Met de functie
```

```
RETURN myOscillator
```

## MIN MAX

Traditionele wiskundige berekeningen.

### Omschrijving

**MIN**(a, b)

**MAX**(a, b)

Deze operatoren produceren respectievelijk de minimum en de maximum waarde tussen beide haakjes.

#### Voorbeeld1 MIN

```
REM Such program returns the lowest price between the opening and closing price
```

```
myLowest = MIN(open, close)
```

```
REM Returns 'myLowest' as output of Met de functie
```

```
RETURN myLowest
```

#### Voorbeeld2 MAX

```
REM Such program returns the highest price between the opening and closing price
```

```
myHighest = MAX(open, close)
```

```
REM Returns 'myHighest' as output of Met de functie
```

```
RETURN myHighest
```

## *Enkelvoudige wiskundige operatoren*

### **ROUND ABS SGN (-)**

Traditionele wiskundige berekeningen.

### **Omschrijving**

**ROUND(a)**

**ABS(a)**

**SGN(a)**

-a

Deze operatoren stellen u in staat traditionele wiskundige berekeningen te maken:

- afgerond (**ROUND**)
- absolute waarde (**ABS**)
- 1 als  $a > 0$ , -1 als  $a < 0$ , 0 als  $a = 0$  (**SGN**)
- $-ABS(a)$  als  $a > 0$ ,  $ABS(a)$  als  $a < 0$ , 0 als  $a = 0$  (-)

#### **Voorbeeld1 ROUND**

```
REM Such program indicates the psychological crossings each 10 euros
```

```
REM 1: up crossing
```

```
REM -1: low crossing
```

```
my10 = ROUND(close / 10)
```

```
myCross = SGN(my10 - my10[1])
```

```
REM Returns 'myCross' as output of Met de functie
```

```
RETURN myCross
```

#### **Voorbeeld2 ABS**

```
REM A volatility indicator
```

```
myVolatility = ABS(close - close[1])
```

```
REM Returns 'myVolatility' as output of Met de functie
```



## Voorbeeld3 SGN

REM Such indicator emphasizes on the trend of prices and not on its power  
REM Divergences with prices may be of an interest for technical analysis...

REM Indicates the variation trend (+1 = moves up, -1 = moves down)

```
myVariation = SGN(close – close[1])
```

REM Returns the cumulative sum of the variation trend

```
RETURN CUMSUM(myVariation)
```

## SQUARE SQRT

Kwadraat en vierkantswortel

### Omschrijving

**SQUARE(a)**

**SQRT(a)**

Deze operatoren berekenen het kwadraat en de vierkantswortel van een waarde.

#### Voorbeeld1 SQUARE, SQRT

```
REM Such program calculates the standard deviation between latest 20 closing prices
```

```
sumY = 0
```

```
sumY2 = 0
```

```
FOR i = 0 TO 19
```

```
    sumY = sumY + close[i]
```

```
    sumY2 = sumY2 + SQUARE(close[i])
```

```
NEXT
```

```
myVolatility = sumY2/20 - SQUARE(sumY/ 20)
```

```
myStandardDeviation = SQRT(myVolatility)
```

```
REM Returns 'myStandardDeviation' as output of Met de functie
```

```
RETURN myStandardDeviation
```

## LOG EXP

Logaritmische en exponentiele functies.

### Omschrijving

**LOG(a)**

**EXP(a)**

Deze operatoren berekenen het logaritme en het exponent van een getal

#### Voorbeeld LOG, EXP

```
REM Such program calculates the geometric moving average between latest 20 closing prices
```

```
sum = 0
```

```
FOR i = 0 TO 19 // Calculate the average between latest 20 logarithm of closing prices
```

```
    sum = sum + LOG(close[i])
```

```
NEXT
```

```
sum = sum / 20
```

```
REM Returns the geometric moving average
```

```
RETURN EXP(sum)
```

## COS SIN TAN ATAN

Goniometrische functies.

### Omschrijving

**COS(a)**

**SIN(a)**

**TAN(a)**

**ATAN(a)**

Deze operatoren berekenen de sinus, cosinus, tangens en de cotanges van een getal

#### Voorbeeld ATAN

```
REM Such program calculates the angle of the talus of the geometric moving average  
REM between latest 30 closing prices
```

```
// Build the moving average  
mm = AVERAGE[30](close)
```

```
// Calculate the talus  
dy = (mm / mm[1] - 1)*100 // 1 unit = 1% variation of prices  
dx = 1 // 1 unit = duration of one bar
```

```
// Convert the talus to an angle  
myAngle = ATAN(dy/dx)
```

```
REM Returns 'myAngle' as output of the function
```

```
RETURN myAngle
```

## Vergelijkings operatoren

< <= =< > >= => <> =

Voor het vergelijken van elementen:

### Omschrijving

$a < b$

$a \leq b$  of  $a \leq b$

$a > b$

$a \geq b$  of  $a \geq b$

$a = b$

$a \neq b$

Met deze operatoren kunt u elementen (hoofdzakelijk getallen) vergelijken. U kunt ze echter ook gebruiken als wiskundige operatoren voor de volgende vergelijking:

- Als het resultaat TRUE is, dan produceert de vergelijking +1
- Als het resultaat FALSE is, dan produceert de vergelijking 0

#### Voorbeeld1 < >

```
REM Such indicator emphasizes on the prices trend and not on its strength
```

```
myPositiveVariation = close > close[1] // +1 when prices go up
```

```
myNegativeVariation = close < close[1] // 1 when prices go down
```

```
myVariation = myPositiveVariation - myNegativeVariation // +1=bull, -1=bear
```

```
REM Returns cumulative values of 'myVariation'
```

```
RETURN CUMSUM(myVariation)
```

#### Voorbeeld2 >=

```
REM Such indicator emphasizes on the prices trend when volume increase
```

```
IF Volume >= Volume[1] THEN
```

```
myIndicator = myIndicator + (close - close[1])
```

```
ENDIF
```

```
REM Returns 'myIndicator' as output of Met de functie
```

RETURN myIndicator

## **Waarschuwing**

In het bovenstaande voorbeeld, geldt uiteraard dat voor de expressie « **close** - **close**[1] » niet is gedefinieerd voor de eerste bar. Er is dus een risico dat « *myIndicator* » begint met een niet gedefinieerdewaarde , met als consequentie dat alle waarden van « *myIndicator* » fout zijn. **MAAR**, een dergelijk risico doet zich niet voor omdat « **volume** >= **volume**[1]» is ook niet gedefinieerd, daarom wordt het als fout beoordeeld.

## CROSSES OVER - CROSSES UNDER

Test het kruisen van twee items (bijv. lijnen)

### Omschrijving

a **Crosses Over** b

a **Crosses Under** b

Deze operatoren testen respectievelijk de opwaartse kruising en de neerwaartse kruising van beide items.

#### Voorbeeld Crosses over, Crosses under

REM Creating an indicator that counts the support and resistance breaks.

REM We first have to define what a support or resistance break is.

REM In our Voorbeeld, we use the opening price and closing price of the day before

REM to define such support or resistance levels.

Resistance = MAX(open[1], close[1])

Support = MIN(open[1], close[1])

REM We now count the number of resistance breaks.

countResistanceBreak = CUMSUM(close CROSSES OVER Resistance)

REM We then count the number of support breaks.

countSupportBreak = CUMSUM (close CROSSES UNDER Support)

REM Returns the difference between both numbers

RETURN countResistanceBreak - countSupportBreak

## *Logical operators*

### **NOT OR AND XOR**

Traditionele Boolean logic

### **Omschrijving**

**NOT**(a)

a **OR** b

a **AND** b

a **XOR** b

Deze operatoren voeren respectievelijk de volgende combinaties van vergelijkingen uit:

- Logical NOT (**NOT**)
- Logical OR (**OR**)
- Logical AND (**AND**)
- Logical Exclusive OR (**XOR**)

De operatoren kunnen worden gebruikt als instructie om de datastream te bewerken of als wiskundige operatoren.

#### **Voorbeeld1 XOR**

REM Indicator emphasizing the appearance of bearish strength

REM Assumption 1: Decrease on volume may induce a bearish trend on prices

REM Assumption 2: Bearish trend on prices creates new bearish trend

REM Assumption 3: If the above assumptions are TRUE at the same time, a bullish trend

REM can be expected

```
myVolumeCriteria = volume < volume[1]
```

```
myTrendCriteria = close < close[1]
```

```
myCriteria = myVolumeCriteria XOR myTrendCriteria
```

REM Returns the sum of the bearish strength of the latest 10 bars

```
RETURN SUMMATION[10](myCriteria)
```

#### **Voorbeeld2 NOT**

REM This indicator emphasizes the trend of the prices when volume don't decrease

```
IF NOT (Volume < Volume[1]) THEN
  myIndicator = myIndicator + (close - close[1])
ENDIF

REM Returns 'myIndicator' as output of Met de functie
RETURN myIndicator
```

## Taal structuur, instructies

### *Korte omschrijving*

De code van elke functie in IBuilder is opgebouwd uit meerdere instructies gevolgd door RETURN dat de te plotten data definieert.

De syntax is hetzelfde als in *BASIC* : Het is niet nodig om variabelen te definiëren. De gebruikelijke structuren (*IF THEN ELSE ENDIF, FOR TO NEXT, WHILE WEND*) zijn beschikbaar. Instructies zijn gescheiden door een Return naar een andere regel.

Met IBuilder, hoeft u zich geen zorgen te maken over het definiëren van de variabelen. IBuilder doet dat voor u! Een variabele mag een getal, boolean of een vector structuur zijn. Het hangt alleen af van de door zo'n variabele gebruikte instructies. Bijvoorbeeld, wanneer u voorafgaande gebruikt, begrijpt u dat een dergelijke bar een vector structuur is.

### Voorbeeld Variabelen hoeven niet gedefinieerd te worden

```
REM This program doesn't make anything at all.

Variable = 10+8*7           // Variable is an integer
Variable = 3.14159         // Variable is a decimal
Variable = 5 > 2           // Variable is a boolean
Variable = open            // Variable is a vectorial structure

REM Returns 'variable' as output of Met de functie
RETURN variable
```

Het is echter mogelijk (en meestal aan te raden) om uw variabelen te initialiseren met het keyword ONCE. De uitvoering van een dergelijke instructie geldt alleen voor de eerste bar.

### Voorbeeld ONCE

```
REM Calculation of the On Balance Volume

ONCE myOBV = 10000         // Same instruction as IF BarIndex = 0 THEN myOBV = 10000

IF BarIndex > 0 THEN
  myOBV = myOBV + SGN(close - close[1]) * volume
ENDIF

REM Returns 'myOBV' as output of Met de functie
RETURN myOBV
```

## Invoegen van commentaar

### REM

Invoegen van commentaar

### Omschrijving

**REM** commentaar

Met dit keyword kunt u commentaar toevoegen. U dient het te plaatsen aan het begin van elke commentaarregel. IBuilder kijkt niet verder naar een regel die met *REM* begint

#### Voorbeeld1 REM

```
REM Such program returns the closing price
```

```
RETURN close
```

Het is ook mogelijk om commentaar in het midden van een regel te starten met *//*.

#### Voorbeeld2 //

```
RETURN close // Such program returns the closing price
```

## **Variabelen, parameters en definities**

Parameters worden gedefinieerd in de ChartNet interface, aan de rechterkant van het venster dat u kunt gebruiken om uw programma te schrijven/aanpassen. Het is mogelijk om een typering aan elk van uw parameters te geven om het berekenen van uw indicator te optimaliseren vanuit het menu: "Eigenschappen"

- integer
- decimal
- boolean
- type moving average

Locale variabelen hoeven niet gedefinieerd te worden. Hun default waarde is 0. Het is mogelijk om ze initialiseren met de *ONCE* instructie.

### **[ONCE]**

Initialisatie van locale variabelen.

### **Omschrijving**

**[ONCE]** *variabele* = *expressie*

Met het keyword *ONCE* kunt u een variabele initialiseren als een reeds gegev en expressie (waarde van de eerste bar).

#### **Voorbeeld ONCE**

```
REM Calculation of the On Balance Volume

ONCE myOBV = 10000           // Same instruction as IF BarIndex = 0 THEN myOBV = 10000

IF BarIndex > 0 THEN
  myOBV = myOBV + SGN(close - close[1]) * volume
ENDIF

REM Returns 'myOBV' as output of Met de functie
RETURN myOBV
```

## Controle structuren

### IF THEN ELSE ENDIF

Conditionele instructies.

### Omschrijving

```
IF test THEN  
  Instructie1  
ELSE  
  Instructies2  
ENDIF
```

De uitvoering van de instructies is afhankelijk van de *test*. Als *test* is TRUE, wordt *Instructie1* uitgevoerd. Als *test* is FALSE, wordt *Instructie2* uitgevoerd.

#### Voorbeeld 1 IF THEN ELSE ENDIF

```
REM Calculation of the On Balance Volume  
  
IF BarIndex > 0 THEN  
  
  myOBV = myOBV + SGN(close – close[1]) * volume  
  
ELSE  
  
  myOBV = 10000  
  
ENDIF  
  
REM Returns 'myOBV' as output of Met de functie  
  
RETURN myOBV
```

## Voorbeeld 2 IF THEN ELSIF ELSE ENDIF

```
REM "States" indicator

IF close > average[20](close) AND volume > volume[1] THEN

    myState = 1

ELSIF close > average[20](close) AND volume < volume[1] THEN

    myState = 2

ELSIF close < average[20](close) AND volume > volume[1] THEN

    myState = 3

ELSE

    myState = 4

ENDIF

REM Returns 'myState' as output of the function

RETURN myState
```

## WHILE DO WEND

Conditionele instructie.

### Omschrijving

```
WHILE test [DO]  
  instructie  
WEND
```

Zolang de *test* waar is wordt de instructie uitgevoerd. Het keyword **DO** is optioneel.

#### Voorbeeld WHILE DO WEND

```
REM Calculation of the number of consecutive bullish days
```

```
bullish = close > close[1]
```

```
count = 0
```

```
WHILE bullish[count] AND count < BarIndex DO
```

```
  count = count + 1
```

```
WEND
```

```
REM Returns 'count' as output of Met de functie
```

```
RETURN count
```

## FOR TO|DOWNTO DO NEXT

Iterative instruction.

### Omschrijving

**FOR** *variabele* = *expressie1* **TO|DOWNTO** *expressie* [**DO**]  
*instructie*  
**NEXT**

Instructies worden een x aantal malen uitgevoerd. *variabele* wordt geïnitieerd met de *expressie1* en vervolgens vermeerderd (door het gebruik van **TO**) of verminderd (door het gebruik van **DOWNTO**) na elke herhaling. Het eindigt als *variabele* kruist met *expressie2*.

### Voorbeeld FOR TO|DOWNTO DO NEXT

```
REM A non optimized calculation of a 20 bars moving average

IF BarIndex < 19 THEN

    myAverage = undefined

ELSE

    myAverage = 0

    FOR i = 0 TO 19 DO                // calculation of SUMMATION[20](close)
        myAverage = myAverage + close[i]
    NEXT

    myAverage = myAverage / 20

ENDIF                                // equal to AVERAGE[20](close)

REM Returns 'myAverage' as output of Met de functie

RETURN myAverage
```

### Opmerking

Waar mogelijk wordt het aanbevolen om de functie AVERAGE te gebruiken.

## BREAK

Einde van een voorwaardelijke of herhaalde instructie

## Omschrijving

### BREAK

De instructie **BREAK** geeft de mogelijkheid om een **WHILE...WEND** of een **FOR...NEXT** instructie te beëindigen.

### Voorbeeld BREAK

```
REM A calculation of the consecutive bullish days
```

```
bullish = close > close[1]
```

```
count = 0
```

```
WHILE count < BarIndex DO
```

```
  IF NOT bullish[count] THEN
```

```
    BREAK
```

```
  ENDIF
```

```
  count = count + 1
```

```
WEND
```

```
REM Returns 'count' as output of Met de functie
```

```
RETURN count
```

## CONTINUE

Continuering van een voorwaardelijke of herhaalde instructie

### Omschrijving

## CONTINUE

De instructie **CONTINUE** geeft u de mogelijkheid om terug te gaan naar de initiële test van een **WHILE...WEND** of **FOR...NEXT** instructie.

### Voorbeeld CONTINUE

```
REM A calculation of the consecutive bullish days
```

```
bullish = close > close[1]
```

```
count = 0
```

```
WHILE count < BarIndex DO
```

```
  IF bullish[count] THEN
```

```
    count = count + 1
```

```
    CONTINUE
```

```
  ENDIF
```

```
  BREAK
```

```
WEND
```

```
REM Returns 'count' as output of Met de functie
```

```
RETURN count
```

## Gebruik van een gebruikers functie

### CALL | GOSUB

Roep een gebruikers functie aan.

### Omschrijving

*var1, var2, ...* = **CALL|GOSUB** *functienaam*[*p1,p2,...*]( *price* )

Parameters worden tussen hoekhaakjes genoteerd, de price tussen haakjes.

De notatie van de parameters dient in de zelfde volgorde te staan als in de aangeroepen functie.

Nieuwe prices kunnen alleen uit de functie volgen wanneer in de opgeroepen functie *customClose* wordt toegepast.

De uitvoer die wordt berekend door functie krijgt haar variabelen in dezelfde volgorde als in de instructie *RETURN* van de opgeroepen functie

### Voorbeeld CALL|GOSUB

#### FUNCTION « myFunction » - Parameter « p » integer

REM Calculation of the value of the variable price on the previous bar

RETURN **customClose**[*p*]

#### FUNCTION « myPrincipalFunction »

REM Calculation of myFunction (parameter p = 10) of the curve « open+close »

**myIndicator** = CALL **myFunction**[**10**](**open + close**)

REM Calculation of my smoothed function with a moving average

**mySmoothedIndicator** = **average**[**10**](**myIndicator**)

RETURN **mySmoothedIndicator**

## *Return van een gebruikers functie*

### **RETURN**

Publiceren van de resultaten van een gebruikers functie.

### **Omschrijving**

**RETURN** expressie [**AS** « naam »], ...

De uitvoer kan een naam krijgen. Deze namen verschijnen in het indicator venster.

#### Voorbeeld RETURN, AS

REM Calculation of Bollinger Bands

```
mm = AVERAGE[20](close)
```

```
StandardDeviation = STD[20](close)
```

```
bsup = mm + 2 * StandardDeviation
```

```
binf = mm - 2 * StandardDeviation
```

REM Returns 2 curves with customized names

```
RETURN bsup AS « Bollinger Sup », binf AS « Bollinger Inf »
```

#### Voorbeeld RETURN

REM Calculation of the moving average of Bollinger

```
mm = AVERAGE[20](close)
```

REM Returns a curve without customized name

```
RETURN mm
```

## Functie overzicht

### *Moving averages*

#### **Average**

Berekening van moving averages.

#### **Omschrijving**

**Average**[count]( price)

**Average**[count, type]( price)

Met de functie *Average* kunt u een moving average van de price van *count* bars berekenen.

Standaard wordt een wiskundige moving average gebruikt. Het is echter mogelijk om een soort moving average te definiëren van het indicator venster door het selecteren van een « MATYPE » parameter en deze parameter in de functie te verwerken zoals hieronder staat aangegeven.

#### **Voorbeeld1 Average**

REM Calculation of a simple moving average on 20 bars

```
mm = Average[20](close)
```

```
RETURN mm
```

#### **Voorbeeld2 Average**

REM Calculation of a customizable moving average on 20 bars

REM A parameter named « choice » of type « MATYPE » has to be created

```
mm = Average[20, choice](close)
```

```
RETURN mm
```

## ExponentialAverage

Berekening van moving averages.

### Omschrijving

**ExponentialAverage**[count]( price)

Met de functie *ExponentialAverage* kunt u het exponentieel moving average van de price van *count* bars berekenen.

#### Voorbeeld ExponentialAverage

REM Calculation of an exponential moving average on 20 bars

```
mm = ExponentialAverage[20](close)
```

```
RETURN mm
```

## WeightedAverage

Berekening van moving averages.

### Omschrijving

**WeightedAverage**[count](price)

Met de functie *WeightedAverage* kunt u de weighted moving average van de *price* van *count* bars berekenen

### Voorbeeld WeightedAverage

```
REM Calculation of a weighted moving average on 20 bars
```

```
mm = WeightedAverage[20](close)
```

```
RETURN mm
```

## WilderAverage

Berekening van moving averages.

## Omschrijving

**WilderAverage**[count](price)

Met de functie *WilderAverage* kunt u de Wilder moving average van de *price* van *count* bars berekenen.

## Voorbeeld WilderAverage

REM Calculation of a Wilder moving average on 20 bars

```
mm = WilderAverage[20](close)
```

```
RETURN mm
```

## TriangularAverage

Berekening van moving averages.

### Omschrijving

**TriangularAverage**[count](price)

Met de functie *TriangularAverage* kunt u een Triangular moving average van de *price* van *count* bars berekenen.

### Voorbeeld TriangularAverage

REM Calculation of a Triangular moving average on 20 bars

```
mm = TriangularAverage[20](close)
```

```
RETURN mm
```

## EndPointAverage

Berekening van moving averages.

### Omschrijving

**EndPointAverage**[count](price)

Met de functie *EndPointAverage* kunt u de End Point moving average van de *price* van *count* bars berekenen.

### Voorbeeld EndPointAverage

```
REM Calculation of an End point moving average on 20 bars
```

```
mm = EndPointAverage[20](close)
```

```
RETURN mm
```

## TimeSeriesAverage

Berekening van moving averages.

### Omschrijving

**TimeSeriesAverage**[count](price)

Met de functie *TimeSeriesAverage* kunt u de Time Series moving average van de *price* van *count* bars berekenen.

### Voorbeeld TimeSeriesAverage

REM Calculation of a Time Series moving average on 20 bars

```
mm = TimeSeriesAverage[20](close)
```

```
RETURN mm
```

## DEMA

Berekening van moving averages.

### Omschrijving

**DEMA**[count](price)

Met de functie *DEMA* kunt u de double exponential moving average van de *price* van *count* bars berekenen.

### Voorbeeld DEMA

REM Calculation of the double exponential moving average on 21 bars

```
mm = DEMA[21](close)
```

```
RETURN mm
```

## TEMA

Berekening van moving averages.

### Omschrijving

**TEMA**[count](price)

Met de functie *TEMA* kunt u de triple exponential moving average van de *price* van *count* bars berekenen

### Voorbeeld TEMA

REM Calculation of the triple exponential moving average on 21 bars

```
mm = TEMA[21](close)
```

```
RETURN mm
```

## *IBuilder functies*

### **CUMSUM**

Berekening van de cumulatieve som.

### **Omschrijving**

#### **CUMSUM(price)**

Met de functie *CUMSUM* kunt u de cumulatieve som van een indicator berekenen.

#### Voorbeeld CUMSUM

REM Calculation of the On Balance Volume

```
deltaPrice = SGN(close - close[1])
```

```
deltaVolume = deltaPrice * Volume
```

```
myOBV = CUMSUM(deltaVolume)
```

```
RETURN myOBV
```

## SUMMATION

Berekening van een specifieke som

### Omschrijving

**SUMMATION**[count](price)

Met de functie *SUMMATION* kunt u de som van de *count* voorgaande waarden van een indicator berekenen.

### VoorbeeldSUMMATION

REM Calculation of a simple moving average

```
SpecificSum10 = SUMMATION[10](close)
```

```
myMM10 = SpecificSum10 / 10
```

```
RETURN myMM10
```

## LOWEST

Berekening van the laagste waarde gedurende een bepaalde periode

### Omschrijving

**LOWEST**[count](price)

Met de functie *LOWEST* kunt u de laagste waarde van de *count* voorgaande waarden van een indicator berekenen.

### Voorbeeld LOWEST

```
REM Calcul of the Williams %R
```

```
CustomHigh = HIGHEST[14](high)
```

```
CustomLow = LOWEST[14](low)
```

```
myWilliams = (close – CustomLow) / (CustomHigh – CustomLow) * 100
```

```
RETURN myWilliams
```

## HIGHEST

Berekening van the hoogste waarde gedurende een bepaalde periode

### Omschrijving

**HIGHEST**[count](price)

Met de functie *HIGHEST* kunt u de hoogste waarde van de *count* voorgaande waarden van een indicator berekenen.

### Voorbeeld Highest

```
REM Calcul of the Williams %R
```

```
CustomHigh = HIGHEST[14](high)
```

```
CustomLow = LOWEST[14](low)
```

```
myWilliams = (close - CustomLow) / (CustomHigh - CustomLow) * 100
```

```
RETURN myWilliams
```

## ***ChartNet technische indicatoren lijst***

### **AccumDistr**

Berekening van de « Accumulation Distribution » indicator

### **Omschrijving**

**AccumDistr**(price)

Met de functie *AccumDistr* kunt u de « Accumulation Distribution » indicator berekenen.

### **Voorbeeld AccumDistr**

```
REM Calculation of Accumulation Distribution indicator using opening prices
```

```
accumOuv = AccumDistr(open)
```

```
RETURN accumOuv
```

## ADX

Berekening van the « ADX » indicator

### Omschrijving

**ADX**[count]

Met de functie *ADX* kunt u de « ADX » indicator van de *count* voorafgaande bars berekenen.

### Voorbeeld ADX

```
REM Calculation of the ADX14
```

```
myADX = ADX[14]
```

```
RETURN myADX
```

## ADXR

Berekening van de ADXR indicator

### Omschrijving

**ADXR**[count]

De functie *ADXR* berekent de ADXR line van de “ADX” indicator op de *count* voorafgaande bars.

### Voorbeeld ADXR

```
REM Calculation of the ADXR14
```

```
myADXR = ADXR[14]
```

```
RETURN myADXR
```

## AroonUp, AroonDown

Berekening van the « Aroon » indicator

### Omschrijving

**AroonUp**[count]

**AroonDown**[count]

Met de functies *AroonUp* and *AroonDown* kunt u de « Aroon » indicator van de *count* voorafgaande bars berekenen.

### Voorbeeld AroonUp, AroonDown

```
REM Calculation of the difference between both lines of Aroon14
```

```
HighLine = AroonUp[14]
```

```
LowLine = AroonDown[14]
```

```
myIndicator = HighLine - LowLine
```

```
RETURN myIndicator
```

## AverageTrueRange

Berekening van the True Range moving average

### Omschrijving

**AverageTrueRange**[count](price)

Met de functie *AverageTrueRange* kunt u de True Range moving average berekenen gebruikmakend van price van de *count* voorafgaande bars.

#### Voorbeeld AverageTrueRange

REM Calculation of the AVT14 using closing prices

```
monAveragetrueRange = AveragetrueRange[14](close)
```

```
RETURN monAveragetrueRange
```

## BollingerUp, BollingerDown

Berekening van the Bollinger Bands

### Omschrijving

**BollingerUp**[count](price)

**BollingerDown**[count](price)

Met de functies *BollingerUp* en *BollingerDown* kunt u de Bollinger Bands berekenen gebruikmakend van de *price* van de *count* voorafgaande bars.

#### Voorbeeld1 BollingerUp, BollingerDown

```
REM Calculation of the Bollinger Bands
```

```
HighLine = BollingerUp[20](close)
```

```
LowLine = BollingerDown[20](close)
```

```
RETURN HighLine, LowLine
```

#### Voorbeeld2 BollingerUp, BollingerDown

```
REM Calculation of the average standard deviation of the 20 latest opening prices
```

```
HighLine = BollingerUp[20](open)           // 2 Standard Deviation above the average
```

```
LowLine = BollingerDown[20](open)         // 2 Standard Deviation below the average
```

```
myStandardDeviation = (HighLine - LowLine)/4
```

```
RETURN myStandardDeviation
```

#### Opmerking

Het is eenvoudiger om met de functie *STD* de standard deviatie te berekenen.

## BollingerBandWidth

Berekening van the BollingerBandWidth indicator.

### Omschrijving

**BollingerBandWidth**[count](price)

Met de functie *BollingerBandWidth* kunt u de ratio tussen de afstand tussen de BollingerBands en hun gemiddelde berekenen. Deze functie gebruikt de price van de *count* voorafgaande bars.

### Voorbeeld BollingerBandWidth

```
REM Calculation of the ratio between the gap of the BollingerBands and their average
```

```
myGap = BollingerBandWidth[20](close)
```

```
RETURN myGap
```

## CCI

Berekening van the “Commodity Channel Index” indicator

### Omschrijving

**CCI**[count]

**CCI**[count](price)

Met de functie *CCI* kunt u de « Commodity Channel Index » indicator op de voorafgaande *count* bars berekenen. Standaard gebruikt de indicator ***typicalPrice***. Het is echter mogelijk om andere price te kiezen.

### Voorbeeld CCI

```
REM Calculation of the Commodity Channel Index indicator
```

```
myCCI = CCI[20](typicalPrice)
```

```
RETURN myCCI
```

## ChaikinOsc

Berekening van the “Chaikin Oscillator” indicator

### Omschrijving

**ChaikinOsc**[p,q](price)

Met de functie *ChaikinOsc* kunt u de «Chaikin Oscillator» berekenen gebruikmakend van *price*.

#### Voorbeeld ChaikinOsc

```
REM Calculation of the Chaikin Oscillator (MA3 short term, MA10 long term)
```

```
myChaikinOsc = ChaikinOsc[3,10](close)
```

```
RETURN myChaikinOsc
```

## ChandeKrollStopUp, ChandeKrollStopDown

Berekening van de “Chande & Kroll’s Volatility Stop” indicator

### Omschrijving

**ChandeKrollStopUp**[p, q, x]

**ChandeKrollStopDown**[p, q, x]

Deze functies claculeren de stop niveaus van de «Chande & Kroll’s Volatility Stop» indicator.

*P : Average True Range parameter*

*Q : « Hoogste » functie parameter*

*X : True Range coefficient*

### Voorbeeld ChandeKrollStopUp, ChandeKrollStopDown

REM Calculation of the Chande & Kroll’s Volatility Stop indicator

```
myUpStop = ChandeKrollStopUp[10,20,30](close)
```

```
myDownStop = ChandeKrollStopDown[10,20,30](close)
```

```
RETURN myUpStop AS « Stop + », myDownStop AS “Stop -“
```

## Chandle

Berekening van the “Chandle Momentum Oscillator” indicator

### Omschrijving

**Chandle**[count](price)

Met de functie *Chandle* kunt u de « Chandle Momentum Oscillator » berekenen gebruikmakend van de *prices* van de *count* previous voorafgaande bars.

#### Voorbeeld Chandle

```
REM Calculation of the Chandle Momentum Oscillator indicator
```

```
myChandle = Chandle[20](close)
```

```
RETURN myChandle
```

## Cycle

Berekening van the “Cycle” indicator

## Omschrijving

**Cycle**(price)

Met de functie *Cycle* kunt u de « Cycle » on the *price*.

### VoorbeeldCycle

```
REM Calculation of the «Cycle»
```

```
myCycle = Cycle(close)
```

```
RETURN myCycle
```

## DI

Berekening van the DI indicator (directional system)

### Omschrijving

**DI**[count](price)

Met de functie *DI* kunt u de DI on the count previous bars of the *price*.

### VoorbeeldDI

REM Calculation of the DI14 (using closing prices)

```
myDI = DI[14](close)
```

```
RETURN myDI
```

## Diplus, DIminus

Berekening van de lijnen van de DI indicator (directioneel systeem)

### Omschrijving

**Diplus**[count](price)

**DIminus**[count](price)

De functie *Diplus* and *DIminus* berekenen de DI+ and DI- lijnen op de *count* voorafgaande bars van de *price*.

### Voorbeeld Diplus, DImoins

REM Calculation of the DI14 (using closing prices)

```
myDIplus = Diplus[14](close)
```

```
myDIminus = DIminus[14](close)
```

```
RETURN myDIplus AS "DI+", myDIminus AS "DI-"
```

## DPO

Berekening van the « Detrended Price Oscillator » indicator

### Omschrijving

**DPO**[count](price)

Met de functie *DPO* kunt u de «Detrended Price Oscillator» op de voorgaande bars of *price* berekenen.

### Voorbeeld DPO

```
REM Calculation of the Detrended Price Oscillator
```

```
myDPO = DPO[21](close)
```

```
RETURN myDPO
```

## EaseOfMovement

Berekening van the « Ease Of Movement » indicator

### Omschrijving

#### EaseOfMovement[count]

Met de functie *EaseOfMovement* kunt u de «Ease Of Movement» van de count voorafgaande bars berekenen.

#### Voorbeeld EaseOfMovement

```
REM Calculation of the Ease Of Movement 14
```

```
myEase = EaseOfMovement[14]
```

```
RETURN myEase
```

## ForceIndex

Berekening van the « Force Index » indicator

## Omschrijving

**ForceIndex**(price)

Met de functie *ForceIndex* kunt u de « Force Index » van de *price* berekenen.

### Voorbeeld Force Index

```
REM Calculation of the Force Index (using closing prices)
```

```
myFI = ForceIndex(close)
```

```
RETURN myFI
```

## HistoricVolatility

Berekening van de historische volatiliteit.

## Omschrijving

**HistoricVolatility**[count](price)

De functie *HistoricVolatility* berekent de historische volatiliteit op de *count* voorafgaande bars van de *price*.

### Voorbeeld HistoricVolatility

```
REM Calculation of the historic volatility of the 20 previous bars
```

```
myVolatility = HistoricVolatility[20] (close)
```

```
RETURN myVolatility
```

## LinearRegression

Berekening van the linear regression

### Omschrijving

**LinearRegression**[count](price)

Met de functie *LinearRegression* kunt u de linear regression berekenen gebruikmakend van de *price* van de *count* voorafgaande bars.

### Voorbeeld LinearRegression

REM Calculation of the linear regression using closing prices of the 10 latest bars

```
myRegression = LinearRegression[10](close)
```

```
RETURN myRegression
```

## LinearRegressionSlope

Berekening van the linear regression slope

### Omschrijving

**LinearRegressionSlope**[count](price)

Met de functie *LinearRegressionSlope* kunt u de linear regression slope berekenen gebruikmakend van de *price* van de *count* voorafgaande bars.

### Voorbeeld LinearRegressionSlope

```
REM Calculation of the linear regression slope of the 10 latest bars
```

```
mySlope = LinearRegressionSlope[10](close)
```

```
RETURN mySlope
```

## MACD

Berekening van the « Moving Average Convergence Divergence » indicator

### Omschrijving

**MACD**[p,q,r](price)

Met de functie *MACD* kunt u de « Moving Average Convergence Divergence » indicator van de *price* berekenen.

### Voorbeeld MACD

```
REM Calculation of the Moving Average Convergence Divergence
```

```
myMACD = MACD[12,26,9](close)
```

```
RETURN myMACD
```

## MACDline

Berekening van de MACD lijn van de « Moving Average Convergence Divergence » indicator

### Omschrijving

**MACDline**[p,q,r](price)

De functie *MACDline* berekent de MACD lijn van de « Moving Average Convergence Divergence » indicator op *price*.

#### Voorbeeld MACDline

```
REM Calculation of the Moving Average Convergence Divergence
```

```
myMACD = MACDline[12,26,9](close)
```

```
mySignal = ExponentialAverage[9](myMACD)
```

```
RETURN myMACD AS "MACD", mySignal AS "Signal"
```

## MassIndex

Berekening van the « Mass Index » indicator

## Omschrijving

### MassIndex[count]

Met de functie *MassIndex* kunt u de « Mass Index » indicator van *count* bars berekenen.

### Voorbeeld MassIndex

```
REM Calculation of the Mass Index
```

```
myMass = MassIndex[25]
```

```
RETURN myMass
```

## Momentum

Berekening van the momentum

### Omschrijving

**Momentum**[count](price)

Met de functie *Momentum* kunt u de momentum indicator van de *count* voorafgaande bars van de *price* berekenen.

### Voorbeeld Momentum

```
REM Calculation of the Momentum
```

```
myMomentum = Momentum[12](close)
```

```
RETURN myMomentum
```

## MoneyFlow

Berekening van the « Money Flow » indicator

### Omschrijving

**MoneyFlow**[count](price)

Met de functie *MoneyFlow* kunt u de « Money Flow » indicator van de *count* voorafgaande bars van de *price* berekenen.

### Voorbeeld MoneyFlow

```
REM Calculation of the Money Flow
```

```
myMoneyFlow = MoneyFlow[14](close)
```

```
RETURN myMoneyFlow
```

## MoneyFlowIndex

Berekening van the «Money Flow Index» indicator

### Omschrijving

#### MoneyFlowIndex[count]

Met de functie *MoneyFlowIndex* kunt u de «Money Flow Index» indicator van *count* bars berekenen.

#### Voorbeeld MoneyFlowIndex

```
REM Calculation of the Money Flow Index
```

```
myMoneyFlow = MoneyFlowIndex[14]
```

```
RETURN myMoneyFlow
```

## NegativeVolumeIndex

Berekening van the «Negative Volume Index» indicator

### Omschrijving

#### NegativeVolumeIndex(price)

Met de functie *NegativeVolumeIndex* kunt u de «Negative Volume Index» van de *price* berekenen.

#### Voorbeeld NegativeVolumeIndex

```
REM Calculation of the Negative Volume Index
```

```
myNegativeVolumeIndex = NegativeVolumeIndex(close)
```

```
RETURN myNegativeVolumeIndex
```

## OBV

Berekening van de «On Balance Volume» indicator

### Omschrijving

**OBV**(price)

Met de functie *OBV* kunt u het «On Balance Volume» van de *price* berekenen.

#### Voorbeeld OBV

```
REM Calculation of the On Balance Volume
```

```
myOBV = OBV(close)
```

```
RETURN myOBV
```

## PositiveVolumeIndex

Berekening van de «Positive Volume Index » indicator

### Omschrijving

#### PositiveVolumeIndex(price)

Met de functie *PositiveVolumeIndex* kunt u de «Positive Volume Index» van de *price* berekenen.

#### Voorbeeld PositiveVolumeIndex

```
REM Calculation of the Positive Volume Index
```

```
myPositiveVolumeIndex = PositiveVolumeIndex(close)
```

```
RETURN myPositiveVolumeIndex
```

## PriceOscillator

Berekening van de Price Oscillator

### Omschrijving

**PriceOscillator** [p,q](price)

Met de functie *PriceOscillator* kunt u de Price Oscillator van de *price* berekenen.

#### Voorbeeld PriceOscillaor

```
REM Calculation of the Price Oscillator (MA5 short term, MA25 long term)
```

```
myOscillator = PriceOscillator[5,25](close)
```

```
RETURN myOscillator
```

## PVT

Berekening van the « Price Volume Trend » indicator

### Omschrijving

**PVT(price)**

Met de functie *PVT* kunt u de «Price Volume Trend» indicator van de *price* berekenen.

### Voorbeeld PVT

```
REM Calculation of the Price Volume Trend
```

```
myPriceVolumandrend = PVT(close)
```

```
RETURN myPriceVolumandrend
```

## R2

Berekening van the R2

### Omschrijving

**R2**[count](price)

Met de functie *R2* kunt u de R2 berekenen gebruikmakend van de *price* van de *count* voorafgaande bars.

### Voorbeeld R2

```
REM Calculation of the R2
```

```
myR2 = R2[10](close)
```

```
RETURN myR2
```

## Repulse

Berekening van the Repulse

## Omschrijving

**Repulse**[count](price)

Met de functie *Repulse* kunt u de Repulse van de *price* van de *count* voorafgaande bars berekenen.

### Voorbeeld Repulse

```
REM Calculation of the Repulse
```

```
myRepulse = Repulse[5](close)
```

```
RETURN myRepulse
```

## ROC

Berekening van the Rate Of Change

### Omschrijving

**ROC**[count](price)

Met de functie *ROC* kunt u de Rate Of Change van de *price* van de *count* voorafgaande bars berekenen.

### Voorbeeld ROC

```
REM Calculation of the Rate Of Change
```

```
myROC = ROC[12](close)
```

```
RETURN myROC
```

## RSI

Berekening van the Relative Strength Index

### Omschrijving

**RSI**[count](price)

Met de functie *RSI* kunt u de Relative Strength Index van de *price* van de *count* voorafgaande bars berekenen.

### Voorbeeld RSI

```
REM Calculation of the Relative Strength Index
```

```
myRSI = RSI[14](close)
```

```
RETURN myRSI
```

## Parabolic SAR

Berekening van the Parabolic SAR

### Omschrijving

#### SAR

Met de functie Parabolic *SAR* kunt u de Parabolic SAR berekenen.

#### Voorbeeld SAR

```
REM Calculation of the Parabolic SAR
```

```
mySAR = SAR
```

```
RETURN mySAR
```

## SMI

Berekening van the Stochastic Momentum Index indicator

### Omschrijving

**SMI**[p,q,r](price)

Met de functie *SMI* kunt u de Stochastic Momentum Index van de *price* berekenen.

### Voorbeeld SMI

```
REM Calculation of the Stochastic Momentum Index[14,3,5]
```

```
mySMI = SMI[14,3,5](close)
```

```
RETURN mySMI
```

## SmoothedStochastic

Berekening van the smoothed Stochastic

### Omschrijving

**SmoothedStochastic**[p,q](price)

Met de functie *SmoothedStochastic* kunt u de smoothed Stochastic van de *price* berekenen.

### Voorbeeld SmoothedStochastic

```
REM Calculation of the smoothed Stochastic 14, 3
```

```
mySmoothedStochastic = SmoothedStochastic[14,3](close)
```

```
RETURN mySmoothedStochastic
```

## STD

Berekening van the standard deviation

### Omschrijving

**STD**[count](price)

Met de functie *STD* kunt u de standard deviation van de *price* van de *count* voorafgaande bars berekenen.

### Voorbeeld STD

```
REM Calculation of the standard deviation
```

```
mySTD = STD[20](close)
```

```
RETURN mySTD
```

## STE

Berekening van the error deviation

### Omschrijving

**STE**[count](price)

Met de functie *STE* kunt u de error deviation berekenen gebruikmakend van de *price* van de *count* voorafgaande bars.

### Voorbeeld STE

REM Calculation of the error deviation indicator

```
mySTE = STE[10](close)
```

```
RETURN mySTE
```

## Stochastic

Berekening van the Stochastic %K indicator

## Omschrijving

**Stochastic**[p,q](price)

Met de functie *Stochastic* kunt u de %K van de *price* berekenen.

### Voorbeeld Stochastic

```
REM Calculation of the Stochastic 14,3
```

```
myStochastic = Stochastic[14,3](close)
```

```
RETURN myStochastic
```

## SuperTrend

Berekening van de O.Seban's SuperTrend indicator

## Omschrijving

**SuperTrend**[x: coef, y: count of bars](price)

De functie *SuperTrend* berekent de O.Seban's trend volgende indicator.

### Voorbeeld SuperTrend

```
REM Calculation of the O.Seban's Super Trend indicator
```

```
myTrend = SuperTrend[2, 10](close)
```

```
RETURN myTrend
```

## TR

Berekening van the True Range indicator

### Omschrijving

**TR**(price)

Met de functie *TR* kunt u de «True Range» indicator van de *price* berekenen.

### Voorbeeld TR

```
REM Calculation of the True Range
```

```
myTrueRange = TR(close)
```

```
RETURN myTrueRange
```

## TRIX

Berekening van the TRader IndeX indicator

### Omschrijving

**TRIX**[count](price)

Met de functie *TRIX* kunt u de TRader IndeX indicator van de *price* van de *count* voorafgaande bars berekenen.

### Voorbeeld TRIX

```
REM Calculation of the TRader IndeX
```

```
myTRIX = TRIX[15](close)
```

```
RETURN myTRIX
```

## Variation

Berekening van the price variation (in %)

## Omschrijving

**Variation**(price)

Met de functie *Variation* kunt u de *price variation* berekenen.

## Voorbeeld Variation

```
REM Calculation of the price variation in %
```

```
myVariation = Variation(close)
```

```
RETURN myVariation
```

## Volatility

Berekening van the Chaikin Volatility indicator

## Omschrijving

**Volatility**[p,q]

Met de functie *Volatility* kunt u de Chaikin Volatility indicator berekenen waarbij als parameters de lengte van de moving average en het aantal bars wordt gebruikt

### Voorbeeld Volatility

```
REM Calculation of the Chaikin Volatility (MA10, and on 10 bars)
```

```
myVolatility = Volatility[10,10]
```

```
RETURN myVolatility
```

## VolumeOscillator

Berekening van the Volume Oscillator indicator

### Omschrijving

**VolumeOscillator**[p,q]

Met de functie *VolumeOscillator* kunt u de Volume Oscillator indicator berekenen door de parameters van de lengte van de short en van de long moving average te gebruiken

#### Voorbeeld VolumeOscillator

```
REM Calculation of the Volume Oscillator (MA5 short term, MA25 long term)
```

```
myOscillator = VolumeOscillator[5,25]
```

```
RETURN myOscillator
```

## VolumeROC

Berekening van the Volume Rate Of Change indicator

### Omschrijving

**VolumeROC**[count]

Met de functie *VolumeROC* kunt u de Volume Rate Of Change indicator van de *count* voorafgaande bars berekenen.

### Voorbeeld VolumeROC

```
REM Calculation of the Volume Rate Of Change
```

```
myRate = VolumeROC[12]
```

```
RETURN myRate
```

## Williams

Berekening van the Williams %R indicator

### Omschrijving

**Williams**[count](price)

Met de functie *Williams* kunt u de Williams %R indicator berekenen over de *count* voorafgaande bars van de *price*.

### Voorbeeld Williams

```
REM Calculation of the Williams%R
```

```
myWilliams = Williams[14](close)
```

```
RETURN myWilliams
```

## WilliamsAccumDistr

Berkening van de Williams Accumulation Distribution indicator

### Omschrijving

**WilliamsAccumDistr**(price)

Met de functie *WilliamsAccumDistr* kunt u de Williams Accumulation Distribution indicator van *price* berekenen.

### Voorbeeld WilliamsAccumDistr

```
REM Calculation of the Williams Accumulation Distribution
```

```
myWilliams = WilliamsAccumDistr(close)
```

```
RETURN myWilliams
```