NanoTrader

Fonctions Express

www.whselfinvest.fr

FIPERTEC
Financial Performance Technology

Contents

Ce qui est nouveau

Version 3.0

Nouvelles fonctions:

- GetCurrency()
- GetExpiration()
- GetMasterChartAggregation()
- GetMasterChartAggregationValue()

Nouvelles fonctionnalités:

- Importation de valeurs de tableau à partir d'un symbole.

Introduction

Ce manuel propose un aperçu de toutes les fonctions disponibles dans le module express. C'est un outil éducatif permettant un processus d'apprentissage par étapes. Les fonctions sont logiquement classées par catégorie et regroupées pour mettre en lumière leurs similarités.

Pour chaque fonction décrite dans ce manuel, un exemple est présenté pour illustrer son utilité et son exécution. Les exemples peuvent être copiés-collés de ce manuel dans l'éditeur Express (pour plus de détails, voir l'annexe à l'introduction à la p. 7). Merci de noter que les exemples sont uniquement proposés à des fins d'évaluation et ne doivent pas être considérés comme une stratégie pour le trading.

Avant de commencer la programmation sur WHS NanoTrader en utilisant le module Express, il est important de se familiariser avec les principes les plus élémentaires de la plateforme. Ces principes sont documentés et expliqués dans les manuels de la plateforme, les films et les nombreux forums, qui peuvent tous être accessibles à partir de notre site Web.

Nos films, en particulier, sont d'une grande utilité. Il y a 80 films illustrant les capacités générales de la plateforme et 42 films décrivant les procédures de programmation. Les films sur la programmation sont clairement séparés en une partie théorique et un exemple pratique.

Ci-dessous un bref résumé des catégories décrites dans ce manuel:

Symbole de l'information / Information relative au temps / Alerter et Informer

Ces trois groupes de fonctions sont étroitement liés étant donné que le Symbole de l'information et les Informations relatives au temps contiennent des informations pouvant être présentées visuellement ou auditivement à l'aide d'Alertes ; le but étant d'Informer.

Non seulement l'utilisateur pourra constater la liberté qu'il aura grâce à sa plateforme, étant donné qu'il sera informé des modifications par un indicateur ou un signal, mais il aura également la possibilité d'expérimenter sa créativité, grâce à un nombre presque infini d'applications et d'outils de personnalisation (couleurs, messages, sons, etc).

Boucles et tableaux

Cette section traite les principes les plus élémentaires et fondamentaux de la programmation express. Elle est à lire absolument et ne doit donc pas être omise.

D'autre part, les fonctions Tableaux sont destinées à des utilisateurs plus expérimentés ayant des besoins très spécifiques.

Indexation et programmation efficace

Les fonctions d'indexation permettent à l'utilisateur de choisir une barre afin d'exécuter une série d'instructions sur cette barre en particulier.

Une fonctionnalité très pratique qui relève de cette catégorie est CalculateAtEveryTick (), qui permet à l'utilisateur d'économiser une énorme capacité informatique en sauvegardant les calculs jusqu'à la fin d'une période, plutôt que de recalculer l'ensemble de la série à chaque tick.

IsFirstBar / IsFinalBar et Fonctions Associées

Ces fonctions devraient être utilisées pour regrouper toutes les instructions n'ayant pas besoin d'être répétées à chaque barre. C'est par exemple la raison pour laquelle IsFirstBar/IsFinalBar est utilisé en combinaison avec MovingAverage et ExpMovingAverage.

Fonctions Mathématiques

Cette section est un récapitulatif simple des outils mathématiques intégrés dans Express, ce qui rappellera probablement de bons souvenirs du lycée et des enseignants.

Fonctions de Traçage

Il s'agit d'un domaine très intéressant avec de nombreuses fonctions permettant à l'utilisateur d'exprimer sa créativité à travers la personnalisation du graphique (fond, couleur, type de graphique).

Importation de Séries

Cela répond à la demande d'appliquer une série à une autre, par exemple appliquer un RSI à une moyenne mobile.

Création de Stops et Targets personnalisés

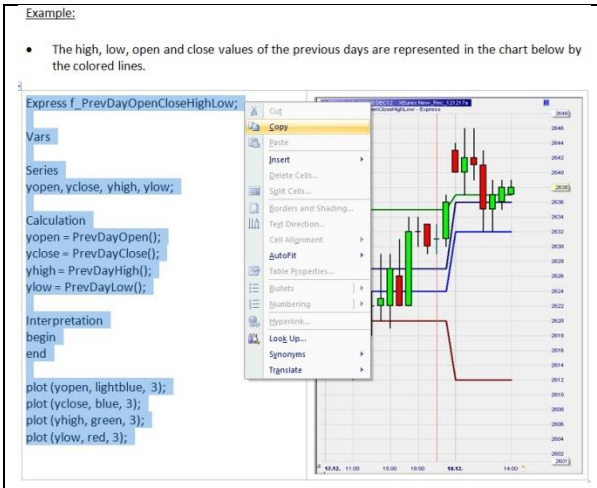
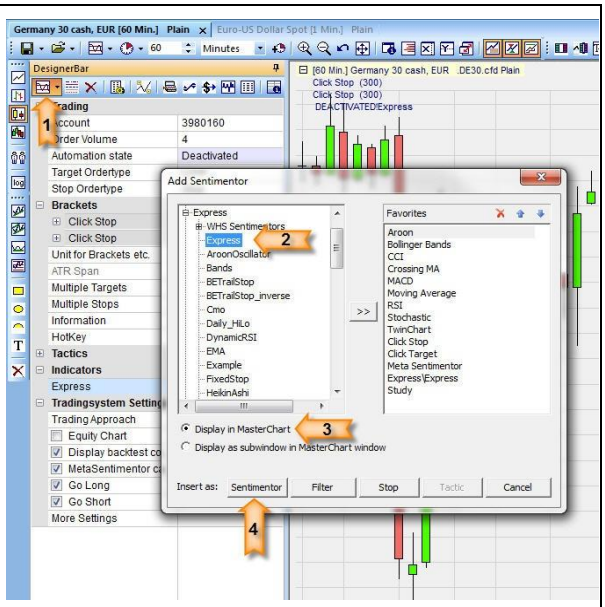
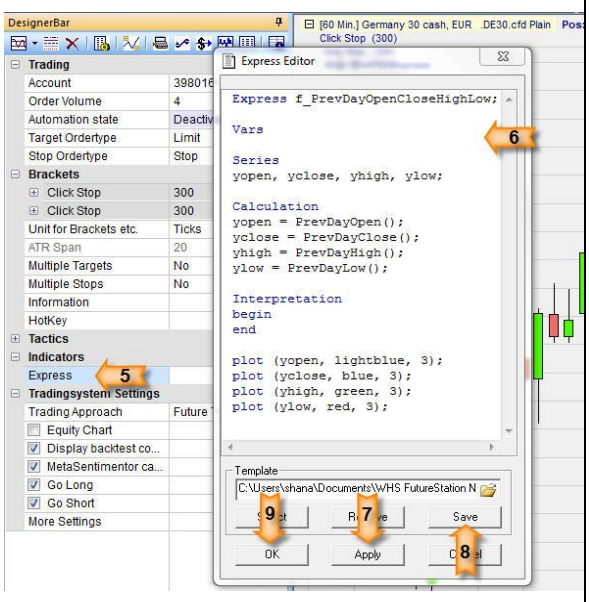
C'est un autre domaine dans lequel le trading et la créativité se rejoignent. Cette fonction permet à l'utilisateur d'affiner les stops et les targets afin de parfaire une stratégie personnalisée.

Outils d'interprétation prédéfinis

Cette section propose une explication des outils prédéfinis de swing trading et conçus pour que l'utilisateur évite les longs programmes.

Annexe à l'introduction

Comment copier un code Express du manuel et l'ajouter à l'éditeur Express pour affichage.

<table>
<tr>
<td> 
</td>
<td> 
</td>
</tr>
<tr>
<td>Sélectionnez le texte de l'exemple du manuel (Ctrl + A), faites un clic-droit puis « copier » (ou Ctrl + C)</td>
<td>1. Cliquez sur Ajout d'un indicateur
2. Sélectionnez Express puis Express à nouveau
3. Cliquez sur Afficher dans le graphique principal ou dans une sous-fenêtre
4. Insérer un Sentimentor</td>
</tr>
<tr>
<td> 
</td>
<td>  </td>
</tr>
<tr>
<td>5. Double-cliquez sur Express
6. Ctrl + V pour coller le texte sélectionné
7. Cliquez sur Appliquer
8. Cliquez sur Enregistrer
9. Cliquer sur Ok</td>
<td>Mission accomplie!!</td>
</tr>
</table>

Symbole de l'information

SymbolName()

Définition:

- Indique le nom du symbole utilisé dans une étude.

Format:

- string SymbolName()

Exemple:

- Ci-dessous, nous générons un message à chaque fois que la condition (close>close [1]) et (close> close [2]) est vérifiée.

```
Express f_SymbolName

Calculation

if (close > close[1]) and
   (close > close[2]) then MessageBox(
  "Possible trend detected"
  +
  SymbolName());

Interpretation
begin
end
```

PointValue()

Définition:

- Donne la valeur monétaire du point d'un instrument en particulier.

Format:

- Float PointValue()

Exemple:

- Affiche les valeurs monétaires d'un point du DAX et du MINI S&P futures: respectivement 25 EUR et 50 USD.

```
Express f_PointValue

vars

Series a;

calculation

a = PointValue();

interpretation
begin
end

plot(a, blue, 2);
```

TickSize()

Définition:

- Change la taille du tick d'un instrument.

Format:

- Float TickSize()

Exemple:

- L'indicateur affiche le nombre de ticks par barre pour un instrument donné.

```
Express f_TickSize

Vars

Series
a;

Calculation
a = (h-l)/TickSize();

Interpretation
begin
end

plot (a, blue, 2);
```

TickValue()

Définition:

- Donne la valeur monétaire d'un tick pour un instrument donné.

Format:

- Float TickValue ()

Exemple:

- L'indicateur affiche les valeurs monétaires des variations quotidiennes d'un instrument.

```
Express f_TickValue;

Vars

Series
a;

Calculation

a = (h - l)/TickSize()*TickValue();

Interpretation
begin
end

plot(a, blue, 2);
```

PrevDayHigh() / Low() / Open() / Close()

Définition:

- Les fonctions ci-dessous
  - PrevDayHigh()
  - PrevDayLow()
  - PrevDayOpen()
  - PrevDayClose()

  Affichent le high, le low, le cours d'ouverture et de clôture de la veille. Dans le cas où la donnée n'est pas disponible, la valeur affichée sera nulle.

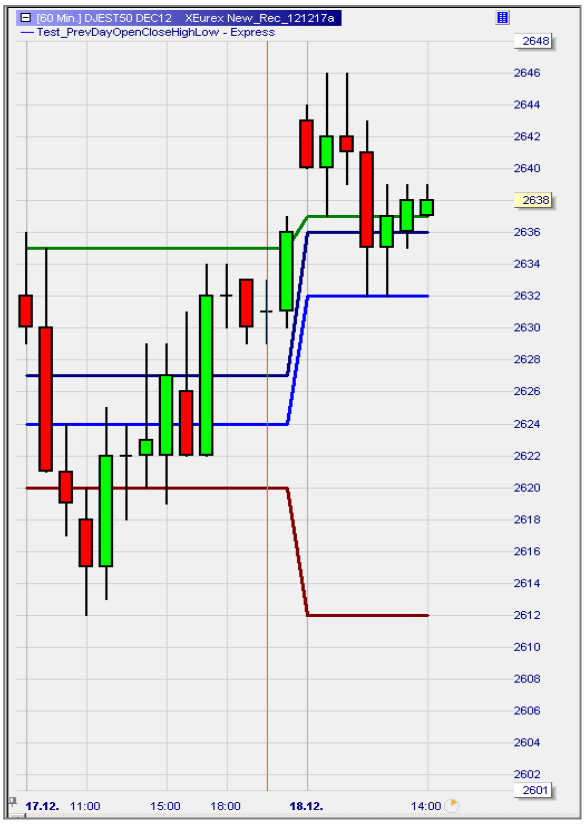
Format:

- Float PrevDayHigh() …

Exemple:

- Le high, low, le cours d'ouverture et de clôture des jours précédents sont représentés dans le graphique ci-dessous par les lignes colorées.

```
Express f_PrevDayOpenCloseHighLow;

Vars

Series
yopen, yclose, yhigh, ylow;

Calculation
yopen = PrevDayOpen();
yclose = PrevDayClose();
yhigh = PrevDayHigh();
ylow = PrevDayLow();

Interpretation
begin
end

plot (yopen, lightblue, 3);
plot (yclose, blue, 3);
plot (yhigh, green, 3);
plot (ylow, red, 3);
```



PrevDayVol()

Définition:

- Affiche le volume de la veille ou une valeur nulle dans le cas où les données ne sont pas disponibles.

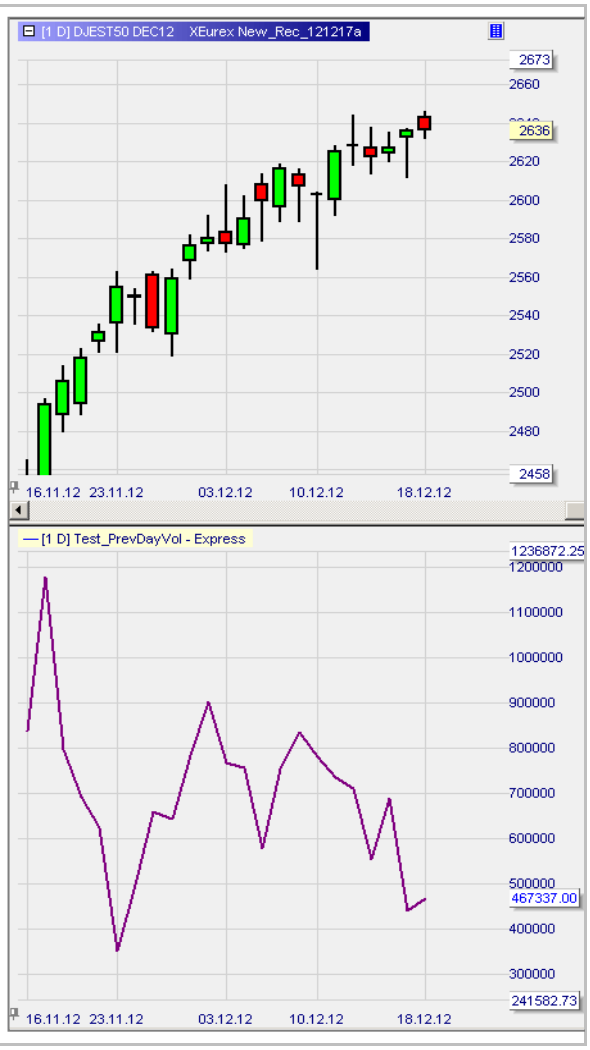
Format:

- Float PrevDayVol ()

Exemple:

- La valeur du volume des jours précédents est affichée dans la fenêtre ci-dessous:

```
Express f_PrevDayVol

Vars

Series
yvolume;

Calculation
yvolume = PrevDayVol();

Interpretation
begin
end

plot (yvolume, lightmagenta, 2);
```



GetCurrency()

Définition:

- Renvoie la devise de base du symbole utilisé dans une étude.

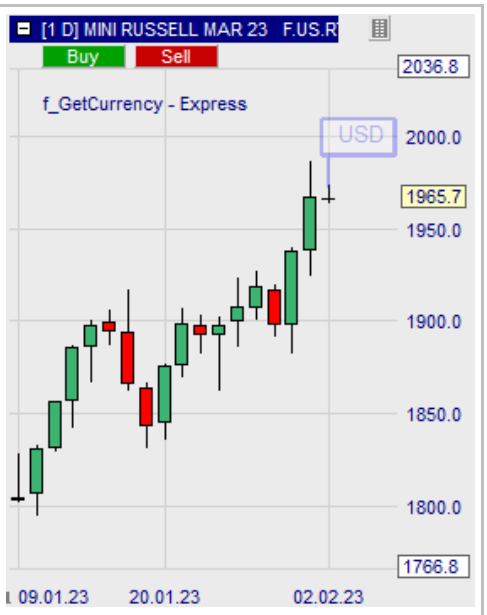
Format:

- string GetCurrency()

Exemple:

- La devise de base du symbole est affichée dans une boîte de message dans la barre finale.

```
Express f_GetCurrency


vars


calculation

if IsFinalBar() then Highlight("textabove:
"+GetCurrency(),"blue");


interpretation
begin
end
```



GetMasterChartAggregation()

Définition:

- Renvoie la devise de base du symbole utilisé dans une étude. Renvoie l'unité d'agrégation du graphique principal exprimée sous la forme d'un nombre entier compris entre 0 et 10.
    - 0 - journalier, 1 - hebdomadaire, 2 - mensuel, 3 - minutes, 4 - secondes, 5 - ticks, 6 - volume, 7 – enverg. abs, 8 – enverg. %, 9 - Renko abs, 10 - WL Bars

Format:

- int GetMasterChartAggregation()

Exemple:

- Dans le graphique en 10 minutes, l'unité d'agrégation du graphique principal est affichée sous la forme d'une valeur entière dans une boîte de message dans la dernière barre.

```
express f_GetMasterChartAggregation

calculation

if IsFinalBar() then Highlight("textabove:"

+NumericToString(GetMasterChartAggregation(),
"%6.0f"),"blue");

interpretation
begin
end
```

GetMasterChartAggregationValue()

Définition:

- Renvoie la valeur d'agrégation du graphique principal exprimée sous forme de valeur flottante.

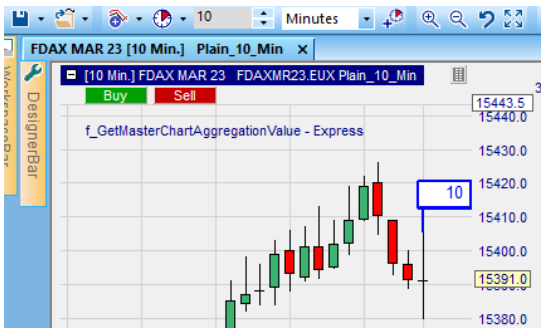
Format:

- float GetMasterChartAggregationValue()

Exemple:

- La valeur d'agrégation du graphique principal, pour le graphique en 10 minutes, est affichée dans une boîte de message dans la dernière barre.

```
express f_GetMasterChartAggregationValue

calculation

if IsFinalBar() then Highlight("textabove:"
+NumericToString(GetMasterChartAggregationValue(),
"%6.0f"),"blue,100");


interpretation
begin
end
```



Information relative au temps

DayOfWeek()

Définition:

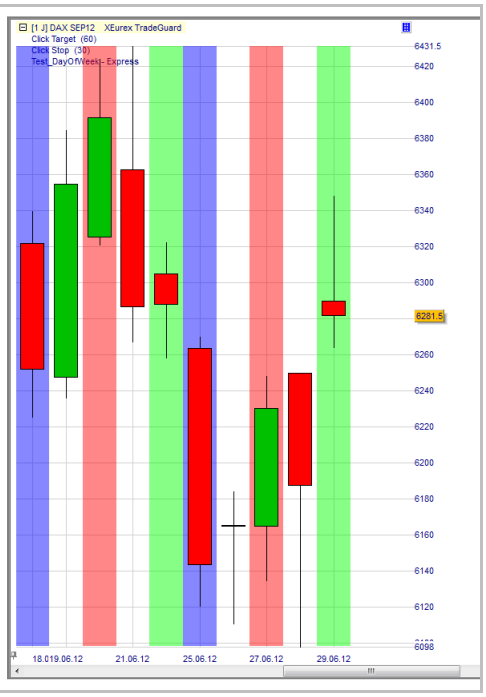
- Exprime le jour de la semaine en un nombre entier compris entre 1 et 5:
  - 0 = Dimanche, 1 = Lundi, 2 = Mardi, 3 = Mercredi, 4 = Jeudi, 5 = Vendredi, 6 = Samedi.

Format:

- int DayOfWeek (time time)

Exemple:

```
Express f_DayOfWeek

vars

calculation

if DayOfWeek(date) = 1 then Highlight("slot","lightblue");
if DayOfWeek(date) = 3 then Highlight("slot","lightred");
if DayOfWeek(date) = 5 then Highlight("slot","lightgreen");

interpretation
begin
end
```



Duration()

Définition:

- Montre le nombre de secondes entre l'instant t1 et t2[1].

Format:

- int Duration (time start, time end)

Exemple:

- Ci-dessous, nous considérons la dernière bougie d'un graphique en 1min sur le Dax future:
  - L'heure d'ouverture est 12:59:00.
  - L'heure actuelle est 12:59:51.
- Notre indicateur affiche trois périodes(Durations) qui sont exprimées en secondes:
  - Rouge: Période entre l'heure d'ouverture et l'heure actuelle = 51.
  - Bleu: Période entre l'heure actuelle et le $StartTime (= 12:52) = 471[2].
  - Vert: Période entre le "date time" actuelle et celle d'il y a 3 bougies: 173[3].

```
Express f_Duration

vars
series
a, b, d;

input
$StartTime(0800, 2359, 1252);

calculation
if IsFinalBar() then
begin
  a = Duration(timeOpen, time);
  b = Duration(NumericToTime($StartTime), time);
  d = Duration(dateTime[3], dateTime);
end

interpretation
begin
end

plot(a, red, 2);
plot(b, blue, 2);
plot(d, green, 2);
```

[1] Les séries de temps Express prédéfinies sont: date, dateOpen, time, timeOpen, dateTime, dateTimeOpen.
[2] 471 secondes = 7 x 60 + 0.85 x 60 secondes = 7 minutes + 51 seconds. Ainsi l'heure actuelle est: 12:52:00 + 00:07:51 = 12:59:51.
[3] 173 secondes = 2 x 60 + 51 secondes.

IsNewDay()

Définition:

- Fait état de la situation dès la première barre de la journée.

Format:

- bool IsNewDay()

Exemple:

- Ci-dessous on voit que l'arrière-plan de la première bougie de la journée est coloré en magenta:

```
Express f_IsNewDay

calculation

if IsNewDay() then Highlight("slot", "magenta");

interpretation
begin
end
```

NumericToTime()

Définition:

- Convertit un nombre à 4 chiffres en une valeur de temps HHMM (par exemple 1545 devient 15:45)
  - Si les deux premiers chiffres sont supérieurs à 23, HH sera réglé sur 23.
  - Si les 2 chiffres qui suivent sont supérieurs à 59, HH sera réglé sur 59.

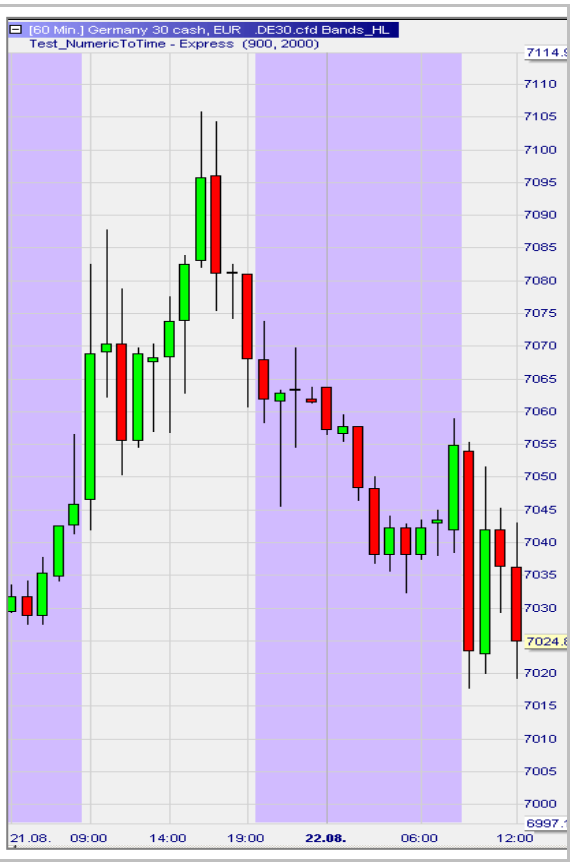
Format:

- time NumericToTime (float value)

Exemple:

- Ci-dessous, nous avons créé un filtre qui bloque les trades en dehors de l'intervalle l'09:00 - 20:00 comme indiqué par la couleur de l'arrière-plan:

```
Express Blocker f_NumericToTime

Vars

Input
$t1(0,2400,900), $t2(0,2400,2000);

Calculation

Interpretation
begin
  if (time >= NumericToTime($t1)) and
     (time <= NumericToTime($t2)) then
  sentiment = senti_pass;
  else
  sentiment = senti_block;
end
```



TimeToNumeric()

Définition:

- Convertit une valeur de temps HHMM en un nombre à 4 chiffres (par exemple 15h45 devient 1545).

Format:

- time TimeToNumeric (valeur flottante)

Exemple:

- Ci-dessous nous affichons dans une sous-fenêtre le nombre de minutes de temps d'ouverture de chaque barre:

```
express f_TimeToNumeric

vars

series oo,cc,hh,ll,tt, hhmn, mn;
input $a(1,20,1);

calculation

hhmn = TimeToNumeric(timeopen);
mn = round(100*((hhmn/100) - floor(hhmn/100)),2);

interpretation
begin
end

plot(mn,blue,2);
```

NumericToDate()

Définition:

- Convertit un nombre de 6 chiffres en une valeur de temps AAAAMMJJ (par exemple 130423 devient 23/04/2013).
    - Le premier groupe de 2 chiffres ne peut pas être supérieur à 99.
    - Si le deuxième groupe de deux chiffres est supérieur à 12, MM sera fixé à 12.
    - Si le troisième groupe de 2 chiffres est plus que le nombre de jours dans un mois donné, DD sera fixé à ce nombre de jours.

Format:

- time NumericToDate (float value)

Exemple:

- Ci-dessous, nous avons créé un filtre qui bloque les trades en dehors de l'intervalle [10/04/2013; 10/05/2013] comme indiqué par la couleur de l'arrière plan :

```
Express Blocker f_NumericToDate

Vars

Input
$day1(130401,131231,130410),
$day2(130401,131231,130510);

Calculation

Interpretation
begin
  if (date >= NumericToDate($day1)) and
     (date <= NumericToDate($day2)) then
  sentiment = senti_pass;
  else
  sentiment = senti_block;
end
```

DateToNumeric()

Définition:

- Convertit une valeur de temps AAAAMMJJ en un certain nombre constitué de 6 chiffres (par exemple 23/04/2013 devient 130423).
  - Le premier groupe de 2 chiffres ne peut pas être supérieur à 99.
  - Si le deuxième groupe de deux chiffres est supérieur à 12, MM sera fixé à 12.
  - Si le troisième groupe de 2 chiffres est plus grand que le nombre de jours dans un mois donné, DD est fixé à ce nombre de jours.

Format:

- time DateToNumeric (float value)

Exemple:

- Ci-dessous nous affichons dans la sous-fenêtre du graphique journalier le jour du mois:

```
express f_DateToNumeric

vars

series yymmdd, dd;

calculation

yymmdd = DateToNumeric(dateopen);
dd = 100*round(((yymmdd/100) -
floor(yymmdd/100)),2);

interpretation
begin
end

plot(dd,blue,2);
```

GetExpiration()

Définition:

- Renvoie la date et l'heure d'expiration du symbole auquel le sentimentor Express est rattaché. Si le symbole n'a pas de date d'échéance, la fonction indiquera le 1er janvier 1970 à 0:00.

Format:

- time GetExpiration()

Exemple:

- Ci-dessous, nous utilisons cette fonction comme un bloqueur afin d'empêcher l'exécution des signaux de trading le jour d'expiration.

Express f_GetExpiration

Calculation

Interpretation

```
begin
  if (date >= GetExpiration()) then
  begin
    sentiment = senti_block;
  end
end
```

Alerter et Informer

MessageBox()

Définition:

- Ouvre une fenêtre pop-up avec un message prédéfini quand une condition particulière rencontre un flux de données. Sans les données en direct, il ne pourra pas y avoir de message.
    - Seule une fenêtre de message peut être affichée sur la durée d'une barre. Une fenêtre de message est affichée qu'une seule fois et à la première occurrence.
    - Pour envoyer un message uniquement à la fin d'une période, utilisez l'instruction suivante : si IsBarCompleted() alors MessageBox("Price > Moving Average");
    - Pour écrire 2 lignes dans un message, utilisez cette instruction:
        - MessageBox("Line 1 \n Line 2");

Format:

- void MessageBox(string message)

Exemple:

- Ci-dessous nous générons des messages à chaque fois que le cours croise la moyenne mobile bleue. Les messages peuvent apparaître plusieurs fois dans la même barre.

```
Express f_MessageBox

Vars

Series
myMA;

Input
$span(1, 200, 30);

Calculation
If IsFirstBar() then MovingAverage(close,
myMA, $span);

if CrossesAbove(close, myMA) then
MessageBox("Price > Moving Average");
if CrossesBelow(close, myMA) then
MessageBox("Price < Moving Average");

Interpretation
begin
end

plot (myMA, blue, 3);
```

PlaySound()

Définition:

- Joue un son prédéfini quand une condition particulière rencontre un flux de données.
  Sans les données en direct il ne pourra pas y avoir de notification sonore.
  - Un son n'est déclenché qu'une seule fois par évènement et par barre.
  - Pour jouer un son à la fin d'une période, utilisez l'instruction suivante:
    si IsBarCompleted() et …… puis PlaySound("gong");
- Les fichiers sonores sont disponibles dans le dossier suivant: NanoTrader\Wav
- Si plusieurs fonctions PlaySound () sont déclenchées simultanément, seule la première fonction PlaySound () sera exécutée.

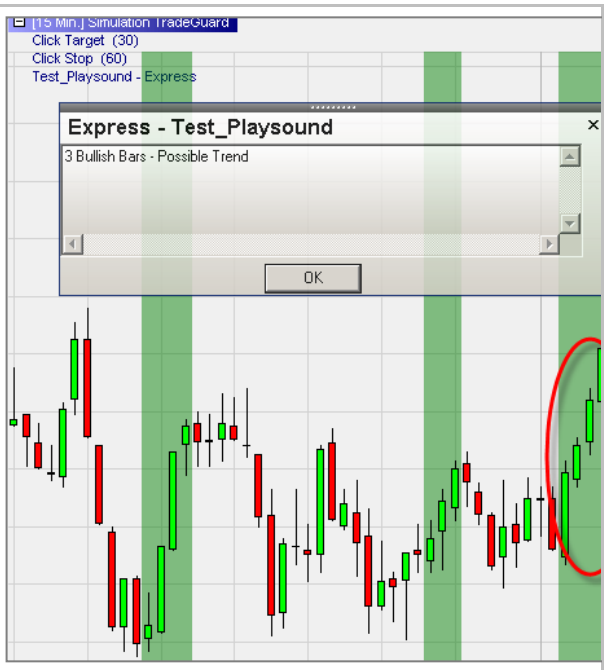
Format:

- void PlaySound(string sound)

Exemple:

- Ci-dessous un son qui se déclenche à chaque fois que trois barres haussières se succèdent.

```
Express f_PlaySound

vars

calculation

if (c > c[1]) and (c[1] > c[2]) and (c[2] > c[3])
then begin
  Highlightat(0, "slot","green");
  Highlightat(1, "slot","green");
  Highlightat(2, "slot","green");
  MessageBox("3 Bullish Bars - Possible
Trend");
  Playsound("corkpop");
 end

Interpretation
begin
end
```



SendEmail()

Définition:

- Envoie un e-mail selon certaines conditions.
  - Un e-mail ne peut être déclenché qu'une seule fois par événement et par barre.
  - L'adresse e-mail doit être enregistrée dans la plateforme dans le champ « adresse de messagerie pour les notifications » sous Extras / Options.
  - Le texte doit être inscrit entre “ “.
  - Pour envoyer un message à la fin d'une période, utilisez l'instruction suivante: Si IsBarCompleted() et …… alors SendEmail();

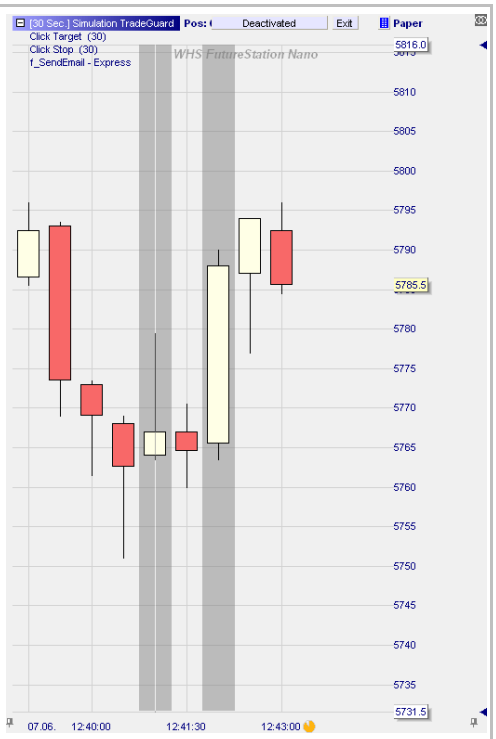
Format:

- void SendEmail(string subject, string message)

Exemple:

- Ci-dessous, nous générons un e-mail de notification après la reconnaissance d’un modèle graphique préenregistré, c'est à dire Market Structure low.

```
express f_SendEmail

vars

series signal(50);

input $EmailActivate(0,1,1);

calculation

if (l > l[1]) and (l[1] < l[2]) and IsBarCompleted() then
begin
  signal = 100;
  if ($EmailActivate = 1) then
  SendEmail("Signal!","close = " + PriceToString(close));
  Highlight("slot","grey");
end

interpretation
begin
  sentiment = signal;
end
```



Screenshot()

Définition:

- Crée une capture d'écran du graphique principal lorsqu'une condition particulière est remplie en utilisant un flux de données en temps réel. Sans données en temps réel, il ne peut pas y avoir de dossiers de captures d'écran.
- Les fichiers peuvent être sauvegardés dans un endroit particulier du PC
  - Le chemin et le nom du fichier sont définis en utilisant une syntaxe similaire à: "C:\NanoFiles\screenshot1.png".
- Une capture d'écran peut être crée une fois par occurrence et par barre. Pour créer une capture d'écran uniquement à la fin de la période, utilisez l'instruction suivante : if IsBarCompleted() and …… then Screenshot("My path\filename.png").
- Si plusieurs fonctions Screenshot() sont déclenchées simultanément, seule la première fonction Screenshot() du code sera exécutée.

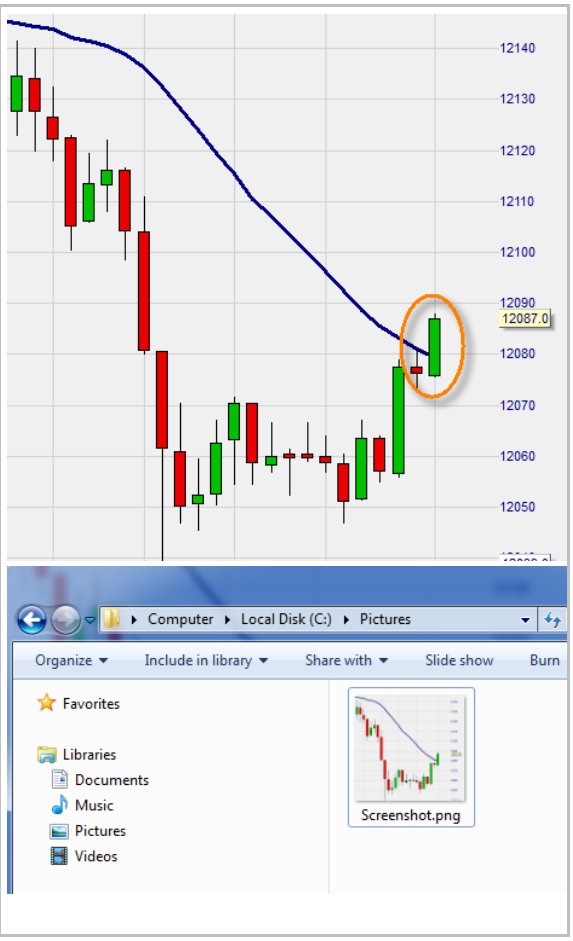
Format:

- void Screenshot(string file)

Exemple:

- Ci-dessous nous générons une capture d'écran à chaque foix que le prix croise la moyenne mobile bleue. Les captures d'écran sont sauvegardées dans le fichier "C:\Pictures\Screenshot.png").

```
Express f_Screenshot

Vars
Series

myMA;
Input
$span(1, 200, 30);

Calculation
If IsFirstBar() then MovingAverage(close, myMA,
$span);

If IsBarCompleted() then
begin
if CrossesAbove(close, myMA) then
Screenshot("C:\Pictures\Screenshot.png");
if CrossesBelow(close, myMA) then
Screenshot("C:\Pictures\Screenshot.png");
end

Interpretation Begin end
plot (myMA, blue, 3);
```



ScreenshotEx()

Définition:

- Crée une capture d'écran soit du graphique principal ou de la fenêtre du capital lorsqu'une condition particulière est remplie.
- Les fichiers sont sauvegardés dans un dossier .png dans un endroit particulier du PC.
  - Le chemin et le nom du fichier sont définis en utilisant une syntaxe similaire à: "C:\NanoFiles\screenshot1.png".
  - Si le nom du fichier ne contient pas un chemin complet alors il est sauvegardé dans répertoire général des captures d'écran de NanoTrader.
- Le `style' détermine les éléments du graphique principal qui seront affichés et les dimensions par défaut. Les réglages sont les suivants:
  - 0 = exactement comme montré sur l'écran
  - 1 = moins de garniture, taille moyenne
  - 2 = encore moins de garniture, petite taille
  - 3 = montre la fenêtre de capital, taille moyenne
  - 4 = montre la fenêtre de capital, petite taille
- Le `width' et `height' sont utilisés pour définir les dimensions de l'image crée. Mettre à 0 pour utiliser le paramétrage par défaut comme définit par le style.
- `periodsOrDays' définit le nombre de périodes à afficher, à partir de la dernière période disponible.
  - Si la valeur est plus grande que 0, alors il définit le nombre total de périodes à afficher.
  - Si la valeur est de 0 alors il montre le point de départ du zoom actuel dans le graphique principal.
  - Si la valeur est négative alors c'est interprété comme des jours complets, par exemple -2 signifie "montrer aujourd'hui et hier".
- Une capture d'écran peut être crée une fois par occurrence et par barre. Pour créer une capture d'écran uniquement à la fin de la période, utilisez l'instruction suivante : if IsBarCompleted() and …… then Screenshot("My path\filename.png").
- Si plusieurs fonctions Screenshot() sont déclenchées simultanément, seule la première fonction Screenshot() du code sera exécutée.

Format:

- void ScreenshotEx(string filename, int style, int width, int height, int periodsOrDays)

Exemple:

- Ci-dessous nous générons une capture d'écran chaque jour à la même heure à 3:30 PM // 15:30. Les captures d'écran sont sauvegardées dans le fichier "C:\ NanoPictures\Chart.png") avec les mêmes éléments que ceux montrés sur l'écran (=0). Les dimensions sont 600x300 pixels et le graphique ne montre que les bougies de la journée actuelle (=-1).

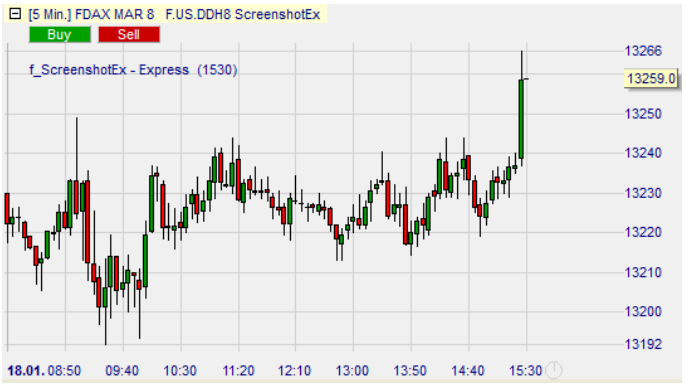
Express f_ScreenshotEx

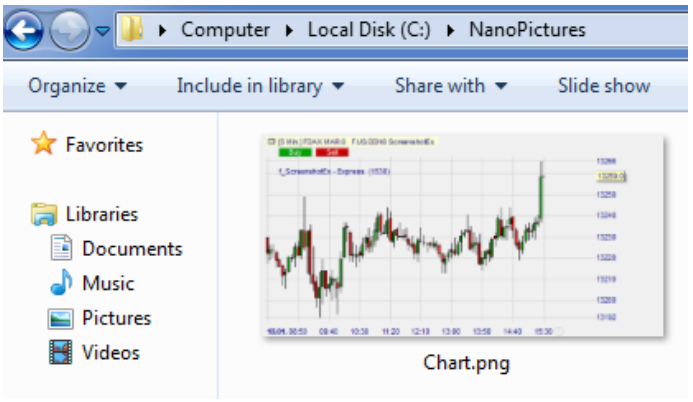
Vars

Input
$EnterAT(000,2359,1530);

Calculation
If (time >= NumericToTime($EnterAT)) AND ((time[1] < NumericToTime($EnterAT)) OR IsNewDay()) then ScreenshotEx("C:\NanoPictures\Chart.png",0,600,300,-1);

Interpretation
begin
end





ShowTip()

Définition:

- Affiche un message lorsque la souris est placée sur une bougie.

Format:

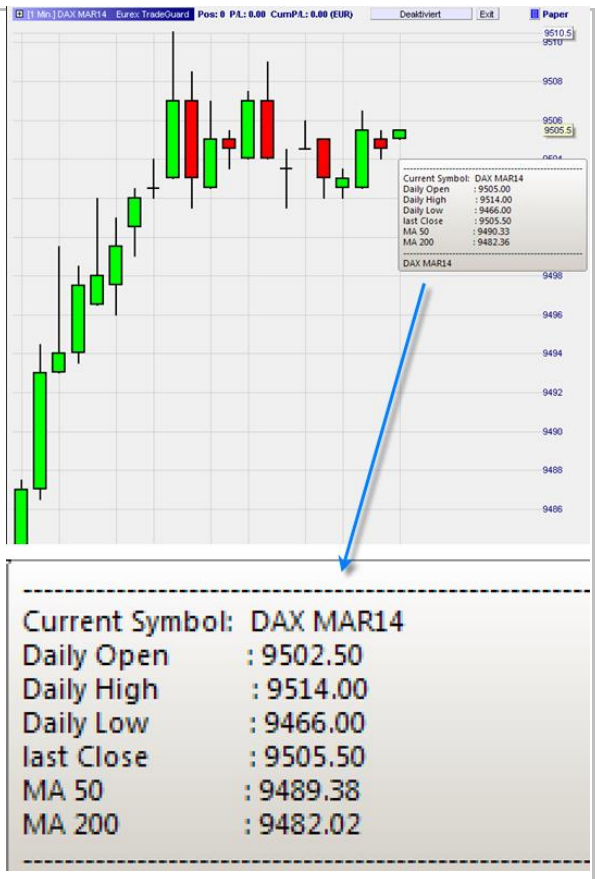
- void ShowTip (string message)

Exemple:

- Exemple 1: L'indicateur des bandes de Bollinger est calculé et les valeurs des bandes supérieure et inférieure sont affichées dans la pointe.
- Exemple 2: Diverses informations sur l'instrument ainsi que les valeurs des indicateurs actifs sont affichées dans la pointe de la dernière bougie.

```
Express f_ShowTip_Example1

Vars

Series
TL, BL, ML, delta;

Input
$span(1, 200, 20),
$StD_factor(1, 200, 20);

Calculation
If IsFirstBar() then
begin
  MovingAverage(close, ML, $span);
  StdDev(close, delta, $span);
end
TL = ML + (($StD_factor/10) *delta);
BL = ML - (($StD_factor/10) *delta);

If IsFinalBar() then ShowTip(" UpperBand:
"+NumericToString(TL, "%6.2f"+"\n LowerBand:
"+NumericToString(BL, "%6.2f") ));

Interpretation
begin
end

plot(ML, blue, 1);
plotband(TL, "red", 1, BL, "red", 1, "lightyellow");
```

```
Express f_ShowTip_Example2

Vars

Input
$MA_50(1, 500, 50),
$MA_200(1, 500, 200);

Series
MA50,
MA200;

Numeric
MinLo,
MaxHi;

Calculation
If IsFirstBar() then
begin
MovingAverage(c, MA50, $MA_50);
MovingAverage(c, MA200, $MA_200);
end
if h > MaxHi then MaxHi = h; else MaxHi = MaxHi;
if l < MinLo then MinLo = l; else MinLo = MinLo;

If IsNewDay() then begin
MaxHi = h;
MinLo = l;
end

If IsFinalBar() then
ShowTip(
"-----------------------------------------------------"
+ "\n" + "Current Symbol:  " + Symbolname()
+ "\n" + "Daily Open        : " + NumericToString(o,"%6.2f")
+ "\n" + "Daily High        : " + NumericToString(Maxhi,"%6.2f")
+ "\n" + "Daily Low         : " + NumericToString(MinLo,"%6.2f")
+ "\n" + "last Close        : " + NumericToString(c,"%6.2f")
+ "\n" + "MA 50             : " + NumericToString(MA50,"%6.2f")
+ "\n" + "MA 200            : " + NumericToString(MA200,"%6.2f")
+ "\n" + "-----------------------------------------------------");

Interpretation
begin
end
```



Highlight() / HighlightRGB()

définition:

- Les deux fonctions permettent à l'utilisateur de mettre en évidence la barre actuelle en utilisant un marquage et une couleur spécifiques.
- Les marquages disponibles sont: "ellipse", "upTriangle", "downTriangle", "slot", "bottomLine", "topLine", "textAbove" and "textBelow".
- Les couleurs suivantes sont disponibles pour la fonction Highlight(): "red", "lightRed", "green", "lightGreen", "blue", "lightBlue", "magenta", "lightMagenta", "yellow", "lightYellow", "cyan", "lightCyan", "grey", "black", "white".
- Pour la fonction HighlightRGB(), un nombre quasi infini de couleurs peut être choisi en utilisant la palette de couleur RGB (RGB = Red-Green-Blue).
- Se référer à la section relative aux fonctions de traçage pour apprendre à créer un code couleur RGB.

Format:

- Void Highlight (string type, string color)
- Void HighlightRGB (string type, int red, int green, int blue)

Exemple 1:

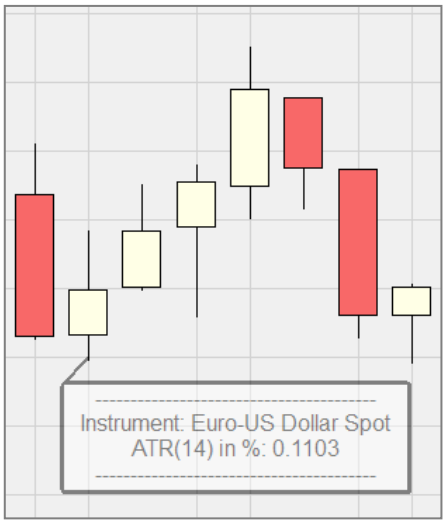
- Mise en évidence de la 50ème bougie du graphique en colorant l'arrière-plan de la bougie en vert clair.

```
Express f_Highlight

vars

calculation

if CurrentBarIndex() = 50 then
Highlight("slot", "lightgreen");

interpretation
begin
end
```

Exemple 2:

- Ici nous créons un cadre à l'aide de la fonction highlight(textbelow,"black") avec un message dont l'information est mise à jour tick par tick. Voici quelques règles très utiles pour la création de contenus:
  - Les zones de textes doivent être encadrées par des guillemets: "Instrument: "
  - Les nombres doivent être convertis au format texte en utilisant NumericToString().
  - Les variables « string » peuvent être encodées sans guillemet: contract
  - Pour passer à la ligne suivante utilisez l'expression: "\n"
  - Toutes les parties doivent être assemblées avec le signe +.

```
express f_Highlight2

vars

input
$period(1,100,14);

string
contract;

calculation

if (CurrentBarIndex()= (FinalBarIndex() - 20)) then
begin
  contract = SymbolName();
  HighLight("textbelow:"
  + "----------------------------------------"
  + "\n" + "Instrument: " + contract
  + "\n" + "ATR" + "(" + NumericToString($period,"") + ") in %: " +
NumericToString(atr($period),"%2.2f")
  + "\n" + "----------------------------------------"
, "black");
end
interpretation
begin
end
```



HighlightAT() / HighlightRGBat()

Définition:

- Les deux fonctions permettent à l'utilisateur de mettre en évidence une barre donnée en utilisant un marqueur et une couleur spécifiques. La barre donnée fait référence à une barre qui se situe à X barres de la barre actuelle. Par exemple:
  - HighlightAT(0,"slot","blue"): la barre donnée est la barre actuelle.
  - HighlightAT(10,"slot","blue"): La barre donnée est la 10ème barre à gauche de la barre actuelle.
  - HighlightAT(-5,"slot","blue"): la barre donnée est la 5ème barre à droite de la barre actuelle.
- Les marqueurs disponibles sont: "ellipse", "upTriangle", "downTriangle", "slot", "bottomLine", "topLine", "textAbove" and "textBelow".
- Les couleurs suivantes sont disponibles pour la function HighlightAT(): "red", "blue", "green", "yellow", "black", "grey", "white", "magenta", "lightred", lightblue", "lightgreen", "lightyellow", "black", "lightgrey", "white" and "lightmagenta".
- Pour la fonction HighlightRGBat(), un nombre quasi infini de couleurs peut être sélectionné en utilisant la palette de couleur RGB (RGB = Red-Green-Blue).
- Se référer à la section relative aux fonctions de traçage pour apprendre à créer un code couleur RGB.

Format:

- Void HighlightAt (string type, string color)
- Void HighlightRGBAt (int offset, string type, int red, int green, int blue)

Exemple:

- Quatre applications de la fonction HighlightAT sont affichées:

Express f_HighlightAT

Calculation

```
If IsFirstBar() then
  Highlightat(-3,"slot","magenta");

If IsFinalBar() then
begin
  Highlightat(0,"slot","lightred");
  Highlightat(1,"slot","blue");
  Highlightat(5,"ellipse","lightgreen");
end
```

Interpretation

```
begin
end
```

Annotate() / AnnotateRGB()

Définition:

- Les deux fonctions permettent à l'utilisateur d'enrichir le graphique avec des annotations personnelles qui sont ajoutées à la barre actuelle en fonction du 'type' choisi et dans la couleur ('color') spécifiée.
- Les types disponibles sont : "ellipse", "upTriangle", "downTriangle", "labelLeft", "labelCenter" et "labelRight".
- La position verticale et la taille de l'annotation pour les types "ellipse", "upTriangle" et "downTriangle" est définie par les paramètres « high » et « low ».
- Le texte à afficher avec "labelLeft", "labelCenter" et "labelRight" est ajouté au type en le séparant par deux points.
- Un saut de ligne peut être inséré en utilisant la séquence \n. Les lignes ne sont pas enveloppées automatiquement.
- La position verticale du texte est définie par le paramètre `high'.
- Les couleurs suivantes sont disponibles pour la fonction Highlight() : "red", "lightRed", "green", "lightGreen", "blue", "lightBlue", "magenta", "lightMagenta", "yellow", "lightYellow", "cyan", "lightCyan", "grey", "black", "white".
- Pour la fonction HighlightRGB(), un nombre presque infini de couleurs peut être sélectionné en utilisant le schéma de couleur RGB (RGB = Red-Green-Blue).
- Reportez-vous à la section sur les fonctions de traçage pour savoir comment créer un code couleur RGB.

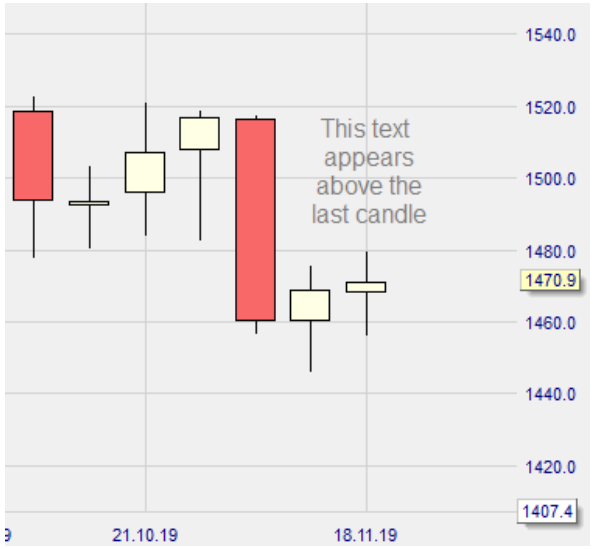
Format:

- void Annotate(string type, string color, float high, float low)
- void AnnotateRGB(string type, int red, int green, in blue, float high, float low)

Exemple:

- Affiche un texte arbitraire au-dessus de la bougie en cours.

```
Express f_Annotate

Calculation

If IsFinalBar() then
Annotate("labelCenter:This text\n
appears\n above the\n last candle",
"black", (h+(h-l)), 0);

Interpretation
begin
end
```



AnnotateAT() / AnnotateRGBAT()

Définition:

- Les deux fonctions permettent à l'utilisateur d'enrichir le graphique avec des annotations personnelles qui sont ajoutées à une barre donnée en fonction du type choisi et dans la couleur spécifiée. La barre donnée fait référence à une barre qui se trouve à x barres de la barre actuelle. Par exemple :
  - HighlightAT(0,"slot","blue"): la barre donnée est la barre actuelle.
  - HighlightAT(10,"slot","blue"): la barre donnée est la 10ème barre à gauche de la barre actuelle.
  - HighlightAT(-5,"slot","blue"): la barre donnée est la 5ème barre à droite de la barre actuelle.
- Les types disponibles sont: "ellipse", "upTriangle", "downTriangle", "labelLeft", "labelCenter" et "labelRight".
- La position verticale et la taille de l'annotation pour les types "ellipse", "upTriangle" et "downTriangle" est définie par les paramètres « high » et « low ».
- Le texte à afficher avec "labelLeft", "labelCenter" et "labelRight" est ajouté au type en le séparant par deux points.
- Un saut de ligne peut être inséré en utilisant la séquence \n. Les lignes ne sont pas enveloppées automatiquement.
- La position verticale du texte est définie par le paramètre `high'.
- Les couleurs suivantes sont disponibles pour la fonction Highlight() : "red", "lightRed", "green", "lightGreen", "blue", "lightBlue", "magenta", "lightMagenta", "yellow", "lightYellow", "cyan", "lightCyan", "grey", "black", "white".
- Pour la fonction HighlightRGB(), un nombre presque infini de couleurs peut être sélectionné en utilisant le schéma de couleur RGB (RGB = Red-Green-Blue).
- Reportez-vous à la section sur les fonctions de traçage pour savoir comment créer un code couleur RGB.

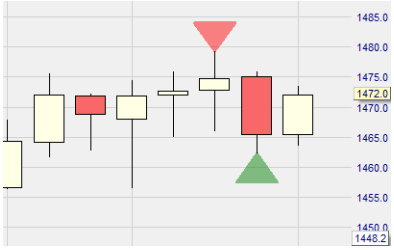
Format:

- void AnnotateAT(int offset, string type, string color, float high, float low)
- void AnnotateRGBAT(int offset, string type, int red, int green, in blue, float high, float low)

Exemple:

- Les deux avant-dernières bougies sont mises en évidence par un 'upTriangle".

```
Express f_AnnotateAT

Calculation
If IsFinalBar() then
begin
  AnnotateAT(1,"upTriangle","green",l[1],(l[1]-(h-l)/2));
 AnnotateAT(2,"upTriangle","lightred",h[2],(h[2]+(hl)/2));
End
Interpretation
begin
end
```



CreateFile()

Définition:

- Cette fonction permet de créer un fichier texte qui sera sauvegardé dans un endroit donné sur un PC, et d'y ajouter n'importe quel texte donné.
- Le chemin et le nom du fichier sont définis à l'aide d'une syntaxe telle que: C:\NanoLogs\nano-actions.txt". Le texte doit être mis entre " ".
    - Si le fichier existe déjà, le texte donné sera ajouté au texte existant.
- Un fichier peut être crée une fois par événement et par barre. Pour créer un fichier uniquement à la fin de la période il faut utiliser l'instruction: if IsBarCompleted() and …… then CreateFile ("My Text");
- Si plusieurs fonctions CreateFile() sont déclenchées au même moment seule la première fonction CreateFile() dans le code sera exécutée.

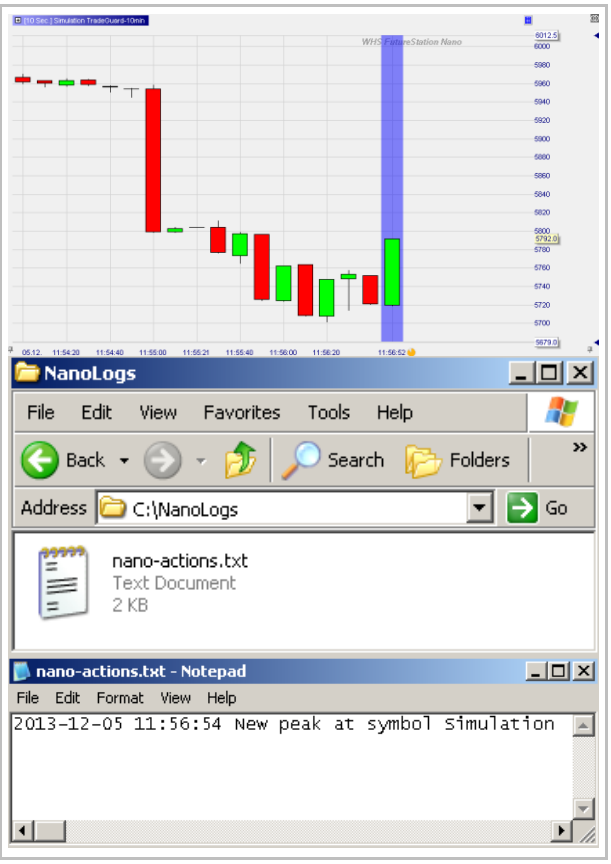
Format:

- void CreateFile(string file, string text)

Exemple:

- Dans l'exemple ci-dessous un fichier texte contenant le message: "New peak at symbol " + SymbolName()" est créé et sauvegardé dans le dossier C:\NanoLogs sur le PC si la condition suivante est remplie: (close > high[1]) and (close > high[2])
- Pour des raisons visuelles la bougie est aussi mise en évidence en bleu.

```
Express f_CreateFile

Calculation

if (close > high[1]) and (close > high[2]) then
begin
  Highlight("slot","lightblue");
  CreateFile("C:\NanoLogs\nano-actions.txt",
TimeToString(datetime, "%Y-%m-%d
%H:%M:%S")
  + " New peak at symbol " + SymbolName());
end

Interpretation
begin
end
```



NumericToString()

Définition:

- Applique un format aux éléments d'une série donnée et les convertis en string.
  - NumericToString applique le format par défaut"%g".
  - Supporte tous les formats utilisés pour C-function "printf()". Voici les plus importants:
    "%f" virgule flottante décimale
    "%6.2f" arrondis à deux décimales
    "%g" ignore les zéros de fin
    "%e" notation scientifique

Format:

- string NumericToString (float value, string format)

Exemple:

- Ci-dessous est affiché le format du prix de clôture devant être indiqué à l'aide de la fonction ShowTip function.
  - En sélectionnant, par exemple le type 4 le prix est indiqué en notation scientifique :

```
Express f_NumericToString

vars

input
$type(0,4,0);

calculation

// Rounds to three decimals
if $type = 1 then
ShowTip(NumericToString(c, "%6.3f"));
else
// Decimal floating point
if $type = 2 then
ShowTip(NumericToString(c, "%f"));
else
// Discards trailing zeroes
if $type = 3 then
ShowTip(NumericToString(c, "%g"));
else
// Scientific notation
if $type = 4 then
ShowTip(NumericToString(c, "%e"));

interpretation
begin
end
```

PriceToString()

Définition:

- Arrondit le cours auquel il est appliqué au cours le plus proche en respectant la taille définie du tick et la précision du symbole analysé et le convertit en string. Prend en compte les notations fractionnaires.

Format:

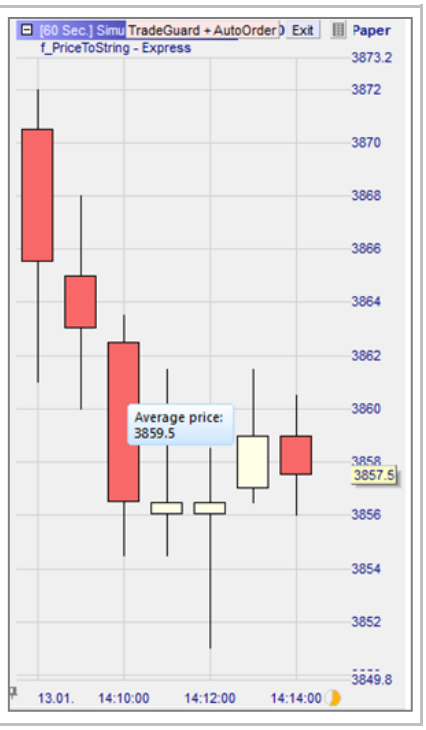
- string PriceToString (float value)

Exemple:

- Ci-dessous est affichée une information au travers de la fonction ShowTip.
  - Cela inclut un titre “Average price” suivi à la ligne suivante du cours moyen.
  - Le format est un nombre rationnel arrondi au tick le plus proche et affiché avec une décimale.

Example:

```
Express f_PriceToString

calculation

ShowTip("Average price: \n" + PriceToString((o + h + l +
2*c)/5));

interpretation
begin
end
```



TimeToString()

Définition:

- Donne une variable de temps en utilisant un des formats ci-dessous:

%a	Nom de jour de la semaine abrégé
%A	Nom de jour de la semaine complet
%b	Nom de mois abrégé
%B	Nom de mois complet
%c	Représentation de la date et de l'heure adaptée au lieu
%d	Jour du mois en nombre décimal (01 – 31)
%H	Heure au format 24-heures (00 – 23)
%I	Heure au format 12-heures (01 – 12)
%j	Jour de l'année en nombre décimal (001 – 366)
%m	Mois en nombre décimal (01 – 12)
%M	Minute en nombre décimal (00 – 59)
%p	Indicateur de Matin/après-midi pour horloge 12 heures, selon la localisation
%S	Seconde en nombre décimal (00 – 59)
%U	Semaine de l'année en nombre décimal, avec dimanche comme premier jour de la semaine (00 – 53)
%w	Jour de la semaine en nombre décimal (0 – 6; Dimanche est 0)
%W	Semaine de l'année en nombre décimal, avec lundi comme premier jour de la semaine (00 – 53)
%x	Représentation de la date pour le lieu actuel
%X	Représentation de l'heure pour le lieu actuel
%y	Année sans siècle, comme nombre décimal (00 – 99)
%Y	Année sans siècle, comme nombre décimal
%z, %Z	Nom du fuseau horaire ou abréviation; Aucun caractère si le fuseau horaire est inconnu
%%	Signe pourcent

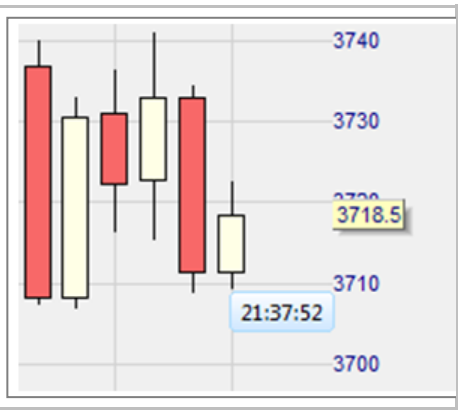
Format:

- string TimeToString (time timeVal, string format)

Exemple:

- Voici une astuce qui permet d'afficher l'heure lorsqu'on passe la souris sur une bougie:

```
Express f_TimeToString

Calculation

showtip(TimeToString(time, "%H:%M:%S"));

interpretation
begin
end
```



IsBarCompleted()

Définition:

- Est avéré une fois que la période actuelle est terminée.

Format:

- bool IsBarCompleted()

Exemple 1:

- If IsBarCompleted() and (volume > 1000) then PlaySound(“gong”);

Exemple 2:

- Ci-dessous, un message est généré chaque fois qu’une barre est terminée:
  - Les fenêtres contenant les messages vont s’accumuler dans le graphique.

Express f_IsBarCompleted

Calculation
If IsBarCompleted() then
MessageBox("A new period has begun");

Interpretation
begin
end

Loops & Arrays

For … to / For … downto

Définition:

- Permet l'exécution d'une boucle dans un programme (see Syntax below).
  - Une boucle exécute un bloc donné de codes pour un nombre donné de fois.

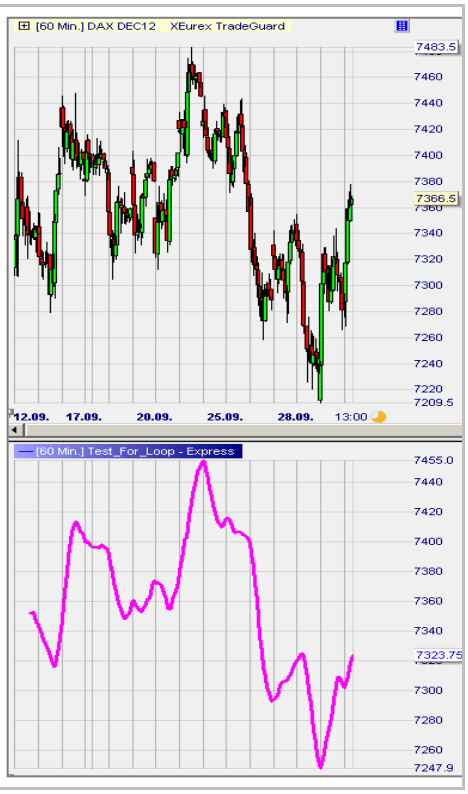
Syntax:

- for <variable> = <start value> to <numerical expression>
  begin
  <statement>;
  <statement>;
  ...
  <statement>;
  end

Exemple 1:

- Ci-dessous on utilise For … loop afin de calculer la moyenne des 10 dernières barres, barre par barre. Veuillez noter que:
  - La boucle est exécutée à chaque barre pour ajouter les dix derniers cours de clôture. On fait varier i entre 0 and 9 puis on exécute = sum + close[i].
    - i = 0 est la barre actuelle, i = 1 est la première barre sur la gauche, i = 2 est la seconde barre sur la gauche, etc.

```
Express f_ForLoop_1

Vars

Numeric i;
Series avgsum, sum;

Calculation
Calculateateverytick(false);
for i = 0 to 9
begin
  sum = sum + close[i];
end
avgsum = (sum/10);

Interpretation
begin
end

plot(avgsum, magenta, 3);
```



Exemple 2:

- Ci-dessous un indicateur est créé en utilisant une boucle pour calculer la différence entre deux moyennes mobiles exponentielles. Veuillez noter que:
  - Ici tous les calculs sont réalisés sur la première barre. Les moyennes mobiles exponentielles sont facilement calculées à l'aide d'une fonction express interne.
  - La MACD est calculée à l'aide d'une boucle. Nous faisons varier i entre 0 et FinalBarIndex.
    - Nous n'utilisons pas 'begin … end' parce qu'il n'y a qu'une seule expression.
    - Etant à la première barre: i = 0 est la première barre, -1 est la deuxième barre à droite, -2 est la troisième à droite, etc.
    - Note: Nous aurions aussi pu utiliser l'instruction suivante:
      For i = 0 downto (- FinalBarIndex())

```
Express f_ForLoop_2

Vars

Series
ema1, ema2, macd, emacd, FBI;

numeric
i;

Calculation

CalculateAtEveryTick(false);

if IsFirstBar() then
begin
  ExpMovingAverage(close, ema1, 12);
  ExpMovingAverage(close, ema2, 26);
  for i = 0 to FinalBarIndex() macd[-i] = ema1[-i] - ema2[-i];
  ExpMovingAverage(MACD, eMACD, 9);
end

Interpretation
begin
end

plot(eMACD, blue, 3);
plot(MACD, red, 3);
```

While … do

Définition:

- Permet qu'une partie d'un programme soit exécuté plusieurs fois de suite aussi longtemps qu'une condition donnée reste vraie (voir la syntaxe ci-dessous).

Syntax:

- while <boolean expression>
  begin
  <statement>;
  <statement>;
  ...
  <statement>;
  end

Exemple:

- Ci-dessous nous comptabilisons le nombre de fois de suite où le cours clôture en hausse(ligne verte) et en baisse(ligne rouge) et enregistrons les résultats barre par barre.

```
express f_WhileDo;

vars
series longPeriods, shortPeriods;
numeric counter;

calculation
counter = 1;
longPeriods = 0;
while (c[counter-1] >= c[counter]) and (c[counter] <> void)
begin
  longPeriods = longPeriods + 1;
  counter = counter +1;
end
counter = 1;
shortPeriods = 0;
while (c[counter-1] <= c[counter]) and (c[counter] <> void)
begin
  shortPeriods = shortPeriods + 1;
  counter = counter +1;
end

interpretation
begin
end

plot(longPeriods, green, 2);
plot(shortPeriods, red, 2);
```

SetArrayTo() / SetArraySize() / GetArraySize()

Définitions:

- Un **array (tableau)** est un ensemble d'un nombre donné d'éléments servant à stocker des numéros.
  - a[0] = 12, a[1] = 45, a[2] = -1, a[3] = -4/5 sont les 4 éléments de l'ensemble appelé a.
  - Pour définir le tableau abc contenant 4 éléments on écrit Vars:
    Array abc[4];

- Les éléments d'un tableau sont paramétrés par défaut à zéro. Pour affecter 500 à tous les éléments on écrit:
  - SetArrayTo(abc, 500);

- Le tableau abc peut être redimensionné à 250 éléments en écrivant:
  - SetArraySize(abc, 250);

- Pour connaître la dimension du tableau abc on écrit:
  - GetArraySize(abc)

- Pour attribuer un nombre à un élément donné du tableau abc on écrit:
  - First element: abc[0] = 10;
  - Second element: abc[1] = -7;
  - Element i: abc[i-1] = -5;

Format:

- void SetArrayTo (array arr, float value)
- void SetArraySize (array arr, int size)
- int GetArraySize (array arr)

Exemple:

- L'indicateur ci-dessous affiche les éléments d'un tableau.
  - A la dernière barre nous affichons les éléments du tableau dans une série bleue ShowResults.
  - Nous modifions et affichons les nouveaux éléments dans une série rouge ShowResults2.

```
Express f_Array;

vars

input
$AbcSize(1, 100, 10), $AbcValue(1, 100, 15);

array
abc[0];

series
ShowResults, ShowResults2;

numeric
i;

calculation

if IsFinalBar() then
begin
  SetArraySize(abc, $AbcSize);
  SetArrayTo(abc, $AbcValue);
  for i=0 to GetArraySize(abc)-1
  begin
    ShowResults[i] = abc[i];
    abc[i] = i * power(-1,i);
    ShowResults2[i] = abc[i];
  end
end

interpretation
begin
end

plot(ShowResults, blue, 2);
plot(ShowResults2, red, 2);
```

Indexing & Efficient programming

CurrentBarIndex() / FinalBarIndex()

Définition:

- La fonction FinalBarIndex() indique, pour un graphique donné, le nombre de barres montrées moins une, dans un graphique,
  Par exemple:
  - L'index de la première barre à gauche du graphique est 0
  - L'index de la dernière ou barre finale sur la droite du graphique est 99
  - Si le graphique affiche 1000 barres, FinalBarIndex() = 999
- CurrentBarIndex() indique le numéro d'index d'une barre.
  Par exemple:
  - Lorsque l'on considère la première barre CurrentBarIndex() = 0
  - Lorsque l'on considère la deuxième barre CurrentBarIndex() = 1
  - Lorsque l'on considère la dixième barre CurrentBarIndex() = 9
  - Lorsque l'on considère la énième barre CurrentBarIndex() = n - 1

Format:

- int CurrentBarIndex()

Exemple:

- CurrentBarIndex() est assigné aux séries x and FinalBarIndex() est assigné aux séries y. Ces deux séries sont tracées sous le graphique principal:
  - CurrentBarIndex() est la ligne droite verte qui va de 0 à 148
  - FinalBarIndex() est la ligne rouge horizontale à 148
  - A l'heure actuelle il y a 149 (= 148 + 1) barres dans le graphique

```
Express f_CurrentBarIndex

vars

series x,y;

calculation

x = CurrentBarIndex();
y = FinalBarIndex();

interpretation
begin
end

plot (x, green, 2);
plot (y, lightred, 2);
```

IndexOfHighest() / IndexOfLowest()

Définition:

- Donne l'index de la valeur la plus haute ou la plus basse pour les éléments series[0], ... series[span – 1].

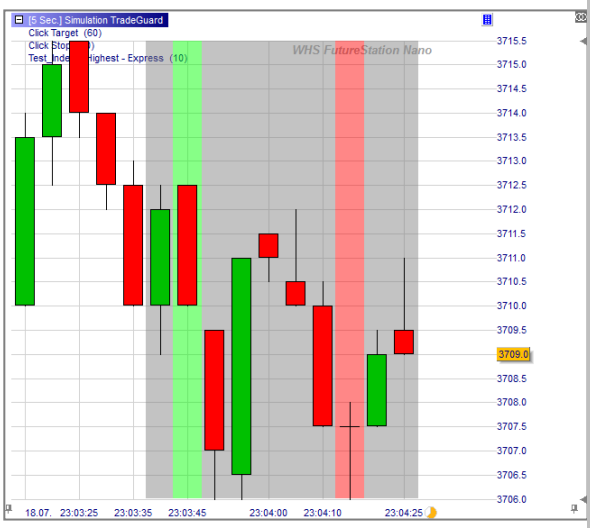
Format:

- int IndexOfHighest (series series, int span)

Exemple:

- Ci-dessous nous colorons en gris d'abord l'arrière-plan des dix dernières bougies.
- Ensuite nous identifions la barre la plus haute et la barre la moins haute des 10 dernières bougies en colorant l'arrière-plan du graphique en vert et rouge. Pour cette raison il est nécessaire d'indiquer l'index des barres à la fonction HighlightAT:

```
Express f_IndexOfHighest

vars

input $period(1,50,10);

calculation

if (CurrentBarIndex() > (FinalBarIndex() - $period))
then Highlight("slot","grey");

if IsFinalBar() then
begin
  for i = 0 to ($period - 1)
  begin
    if ((FinalBarIndex() - i) =
IndexOfHighest(h,$period)) then
HighlightAT(i,"slot","lightgreen");
    if ((FinalBarIndex() - i) =
IndexOfLowest(l,$period)) then
HighlightAT(i,"slot","lightred");
  end
end

interpretation
begin
end
```



CalculateAtEveryTick()

Définition:

- Cette fonction peut être utilisée de deux façons :
  - Avec **CalculateAtEveryTick(false)** un code express est calculé une fois à la clôture de la barre.
    - On utilise cette instruction lorsque le résultat du programme est nécessaire à la clôture de la bougie ou lorsque l'utilisateur souhaite limiter le nombre d'itérations et l'utilisation du CPU.
  - Avec **CalculateAtEveryTick(true)** un code express est calculé à chaque tick. Lorsque cette instruction n'est pas présente dans un code, le programme est exécuté par défaut à tous les ticks.
    - Les utilisateurs doivent tenir compte que les codes express sont exécutés chaque fois qu'un tick est reçu par la plateforme. Aussi longtemps que le nombre de barres et le nombre d'itérations sont limités cette instruction peut être ignorée.

Exemple 1:

- Ici nous calculons à chaque tick.
- La ligne bleue tracée ci-dessous passe par les cours de clôture. Comme le cours de clôture de la dernière bougie est mis à jour à chaque tick entrant, la ligne bleue en fait de même : la ligne bleue se déplace en corrélation avec le prix de clôture.

```
Express f_CalculateAtEveryTick

vars
series x;

calculation
x = c;

interpretation
begin
end

plot(x,blue,2);
```

Exemple 2:

- Ici nous calculons uniquement à la clôture de la barre.
- Contrairement à l'exemple précédent ici la ligne bleue est mise à jour seulement une fois à la clôture de chaque barre: la ligne bleue est fixe, positionnée sur la clôture précédente, alors que le cours continue à baisser.

Express f_CalculateAtEveryTick

vars
series x;

calculation

CalculateAtEveryTick(false);
x = c;

interpretation
begin
end

plot(x,blue,2);

IsFirstBar / IsFinalBar and associated functions

IsFinalBar() / IsFirstBar()

Définition:

- S'avère vrai si à la première barre.
  - IsFirstBar() est utilisé avec l'expression if … then pour identifier la première barre à la gauche du graphique.
- S'avère vrai si à la dernière barre.
  - IsFinalBar() est utilize avec l'expression If … then pour identifier la dernière barre à la droite du graphique.
  - Il s'agit de fonctions très utiles qui peuvent être utilisées pour réduire le nombre d'itérations dans les codes express. Ces fonctions devraient être systématiquement utilisées en conjonction avec les fonctions suivantes: MovingAverage, ExpMovingAverage, StdDev and RSI.

Format:

- bool IsFinalBar()
- bool IsFirstBar()

Exemple 1:

- ici nous nous avons besoin de calculer une variable numérique constante qui sera utilisée à chaque barre successive. La constante K est égale au prix d'ouverture de la première barre. Le résultat, series x, est la différence entre le prix de clôture et le prix d'ouverture de la première barre.

```
Express f_IsFirstBar_example1

vars
series x, zero;
numeric k;

calculation

if IsFirstBar() then
begin
  k = o;
end

x = c – k;

interpretation
begin
end

plot(x, blue, 2);
plot(zero, red, 2);
```

Exemple 2:

- ici nous aimerions calculer et tracer la ligne MACD.
- Nous devons d'abord calculer une moyenne mobile exponentielle (9) et une moyenne mobile exponentielle (26). Leur différence resultant en la ligne MACD.
- La moyenne mobile exponentielle (9) et la moyenne mobile exponentielle (26) sont calculées à la première barre à l'aide de la fonction ExpMovingAverage.
    - Pourquoi ne sont-elles pas calculées à chaque barre? Car ainsi on élimine tout calcul superflu:
        - La function ExpMovingAverage produit la série ema basée sur la série close. Imaginons que notre graphique se compose de 1000 barres. La série ema est obtenue la première après la première exécution de la fonction ExpMovingAverage à la première barre. Si nous avions permis l'exécution de la fonction ExpMovingAverage à chaque barre, nous aurions juste produit 999 fois le même résultat que nous avons obtenu à la première barre. C'est pourquoi nous imposons une seule fois l'exécution de la fonction ExpMovingAverage. Nous sommes effectivement plus efficaces en utilisant moins de ressources.

```
Express f_IsFirstBar_example2

vars

series ema1, ema2, macd, zero;
input $period1(1, 50, 9);
input $period2(1, 100, 26);

calculation

if IsFirstBar() then
begin
  ExpMovingAverage(c, ema1, $period1);
  ExpMovingAverage(c, ema2, $period2);
end

macd = ema1 - ema2;

interpretation
begin
end

plot(macd, blue, 2);
plot(zero, black, 1);
```

MovingAverage()

Définition:

- Donne la moyenne mobile d'une série sur une période donnée.
    - Cette fonction ne doit être utilisée qu'en combinaison avec IsFirstBar() ou IsFinalBar() afin d'économiser une quantité énorme de calcul (voir IsFirstBar ou IsFinalBar).

Format:

- MovingAverage(series source, series target, int span)

Exemple:

- Ci-dessous nous traçons MA1 et MA2.
- Note: Comme les séries MA1 et MA2 sont calculées à la première barre nous n'exécutons pas la fonction MovingAverage aux autres barres (voir IsFirstBart et IsFinalBar pour plus de détails).

```
Express f_MovingAverage

Vars

Series
MA1, MA2;

Input
$spanMA1(1, 200, 8), $spanMA2(1, 200, 30);

Calculation

If IsFirstBar() then begin
  MovingAverage(close, MA1, $spanMA1);
  MovingAverage(close, MA2, $spanMA2);
end

Interpretation
begin
end

plot (MA1, red, 3);
plot (MA2, green, 3);
```

ExpMovingAverage()

Définition:

- Donne la moyenne mobile exponentielle d'une série sur une période donnée.
  - Cette fonction ne doit être utilisée qu'en combinaison avec IsFirstBar() ou IsFinalBar()afin d'économiser une quantité énorme de calcul (voir IsFirstBar ou IsFinalBar).

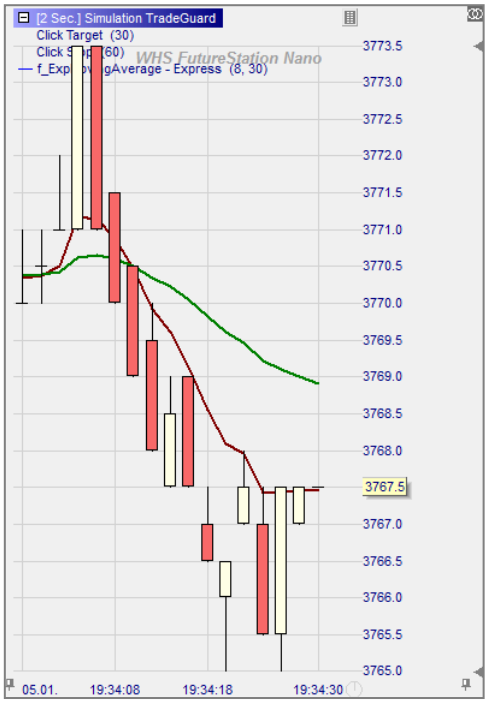
Format:

- ExpMovingAverage (series source, series target, int span)

Exemple:

- Ci-dessous nous traçons deux moyennes mobiles exponentielles MA1 and MA2.
- Note: Comme les séries MA1 and MA2 sont calculées à la première barre nous n'exécutons pas la fonction ExpMovingAverage aux autres barres (voir IsFirstBart et IsFinalBar pour plus de détails).

```
Express f_ExpMovingAverage

Vars

Series
MA1, MA2;

Input
$spanMA1(1, 200, 8), $spanMA2(1, 200, 30);

Calculation

If IsFirstBar() then
begin
  ExpMovingAverage(close, MA1, $spanMA1);
  ExpMovingAverage(close, MA2, $spanMA2);
end

Interpretation
begin
end

plot (MA1, red, 2);
plot (MA2, green, 2);
```



RSI()

Définition:

- Donne l'indice de force relative d'une série(RSI)
  - Cette fonction ne doit être utilisée qu'en combinaison avec IsFirstBar() ou IsFinalBar() afin d'économiser une quantité énorme de calcul(see IsFirstBar or IsFinalBar).

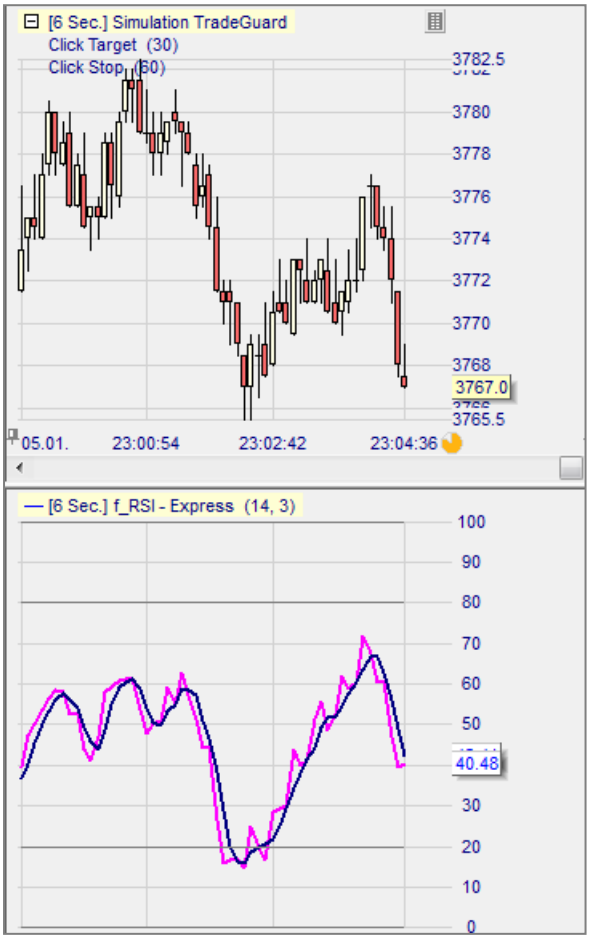
Format:

- RSI(series source, series target, int span)

Exemple:

- Ci-dessous nous traçons un RSI lissé.
- Note: Comme les séries RSI et smoothedRSI sont calculées à la première barre, nous n'exécutons pas les fonctions RSI et MovingAverage aux autres barres (voir IsFirstBar et IsFinalBar pour plus de détails).

```
Express f_RSI

Vars

Series
myRSI, smoothedRSI;

Input
$a(1,100,14), $b(1,100,3);


Calculation
If IsFirstBar() then
begin
  RSI(close, myRSI, $a);
  MovingAverage(myRSI, smoothedRSI, $b);
end


Interpretation
begin
end

plot(myRSI, magenta, 2);
plot(smoothedRSI, blue, 2);
plotline(0, grey, 1);
plotline(20, grey, 1);
plotline(80, grey, 1);
plotline(100, grey, 1);
```



StdDev()

Définition:

- Donne la déviation standard d'une série pour une période donnée.
    - Cette fonction ne doit être utilisée qu'en combinaison avec IsFirstBar() ou IsFinalBar() afin d'économiser une quantité énorme de calcul (voir IsFirstBar ou IsFinalBar).

Format:

- void StdDev (series source, series target, int span)

Exemple:

- Ci-dessous nous traçons les bandes de Bollinger qui sont basées sur une moyenne mobile de 20 périodes avec plus et moins deux fois la déviation standard.
- Note: Comme les séries RSI et smoothedRSI sont calculées à la première barre nous n'exécutons pas les séries RSI et MovingAverage aux autres barres (voir IsFirstBar et IsFinalBar pour plus de détails).

```
Express f_StdDev

Vars

Series
MA, SD, BBH, BBL;

Calculation

If IsFirstBar() then
begin
  MovingAverage(close, MA, 20);
  StdDev(close, SD, 20);
end

BBH = MA + 2*SD;
BBL = MA - 2*SD;

Interpretation
begin
end

plot(MA, green, 2);
plot(BBL, lightblue, 2);
plot(BBH, lightblue, 2);
```

Unaggregate()

Définition:

- Prend un indicateur dans une unité de temps plus importante, par exemple, un MACD 5 minutes, et projette sa valeur dans un autre indicateur qui peut être affiché dans le graphique principal dans une unité de temps inférieure, par exemple, 1 minute.
- Cette fonction ne doit être utilisée qu'en combinaison avec with IsFirstBar() ou IsFinalBar() afin d'économiser une quantité énorme de calcul (see IsFirstBar or IsFinalBar).

Format:

- Void Unaggregate(series source, series target)

Exemple:

- On commence par afficher un graphique en bougies de 5 secondes dans le graphique principal.
- On charge ensuite deux indicateurs:
  - D'abord, AggregatedSentimentor est chargé dans la sous-fenêtre. Il affiche un graphique en bougies avec une moyenne mobile de 10 périodes appelée MAbig. Ce graphique en bougies est paramétré avec une agrégation de 20x[4].
  - Ensuite, UnaggregatedSignal est chargé dans le graphique principal. Cet indicateur importe MAbig et utilise la fonction Unaggregate pour transformer les données de la Moyenne mobile agrégée, par exemple MAbig, en données, unaggregated, par exemple MA[5].
- L'interprétation est donnée par le croisement de 2 courbes: clôture et MA. Cela permet d'entrer en position plus tôt que si nous devions attendre la clôture de bougies de 100 secondes.
- Une autre application peut être d'importer les niveaux de prix d'une unité de temps plus grande pour définir les stops, objectifs, indicateurs...

```
express AggregatedSentimentor

vars
series MAbig, colorbig ;
input $Span(1,200,10);

calculation
CalculateAtEveryTick(false);
if IsFirstBar() then MovingAverage(c,MAbig,$Span);
if (colorbig[1] = 0) then colorbig = 1; else colorbig = 0;

interpretation
begin
end

plot(MAbig,blue,2);
plotcandles(o,c,h,l);
```

[4] Clic droit sur la barre bleue en haut à gauche, choisir l'agrégation, et taper 20.

[5] Pour que ce soit plus clair la couleur de l'arrière-plan du graphique change à chaque nouvelle barre agrégée.

```
express UnaggregatedSignal

vars
series
colorbig(AggregatedSentimentorExpress.colorbig),color,
MAbig(AggregatedSentimentorExpress.MAbig), MA;

calculation
CalculateAtEveryTick(false);
if IsFirstBar() then
begin
  Unaggregate(colorbig,color);
  Unaggregate(MAbig,MA);
end
if (color = 0) then HighlightRGB("slot",192,192,192);

interpretation triggerline(c,MA);

plot(MA,blue,2);
```

Mathematical functions

AbsValue()

Définition:

- Donne la valeur absolue d'un nombre.

Format:

- float AbsValue (float value)

Exemple 1:

- AbsValue(5) = AbsValue(-5) = 5

Exemple 2:

- Ci-dessous nous affichons la valeur absolue de la différence en points entre le cours d'ouverture et le cours de clôture de chaque bougie.

```
Express f_AbsValue

Vars

Series
span;

Calculation
span = AbsValue(open-close);

Interpretation
begin
end

plot (span, green, 3);
```

Sign()

Définition:

- Retourne le signe d'un chiffre donné.
  - Le signe d'un chiffre négatif est -1.
  - Le signe d'un chiffre positif est 1.
  - Le signe de zéro est 0.

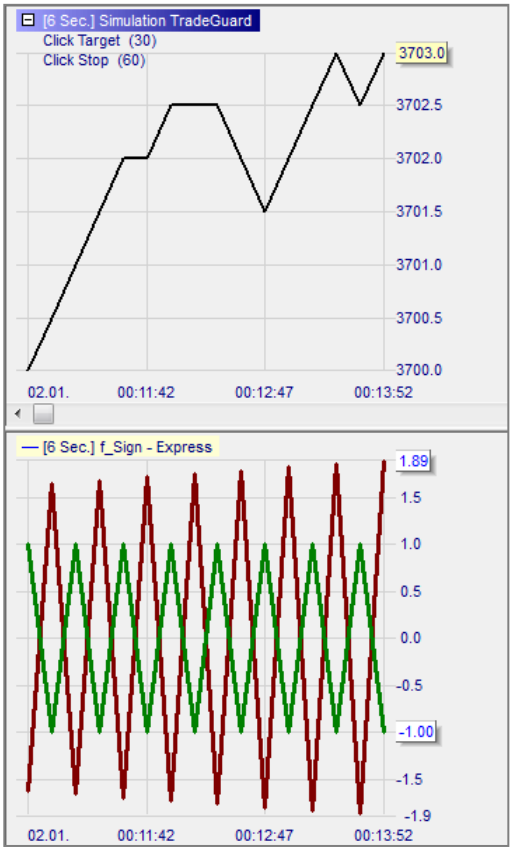
Format:

- Sign (float value).

Exemple:

- Ci-dessous nous traçons une courbe Y en rouge et une autre en vert qui affiche le signe opposé de Y :

```
Express f_Sign

Vars

Series
Y, OppositeSignY;

Calculation

Y = Power(-1.01, CurrentBarIndex());
OppositeSignY = -Sign(Y);

Interpretation
begin
end

plot(Y, red, 3);
plot(OppositeSignY, green, 3);
```



Floor() / Ceiling()

Définition:

- floor() donne le plus grand nombre entier qui est inférieur ou égal à un chiffre donné.
- ceiling() donne le plus petit nombre entier qui est supérieur ou égal à un chiffre donné.

Format:

- float Floor (float value)

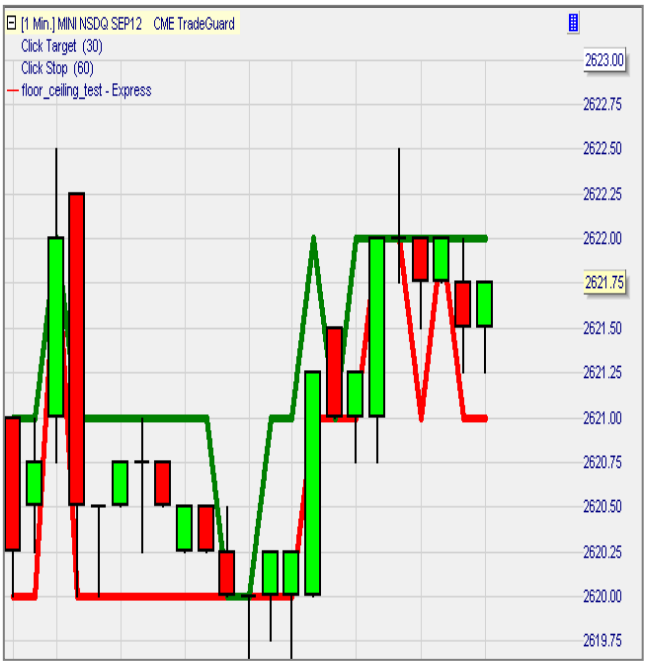
Exemple 1:

x	floor(x)	ceiling(x)
2.3	2.0	3.0
5.7	5.0	6.0
0.0	0.0	0.0
-0.5	-1.0	0.0
-1.5	-2.0	-1.0

Exemple 2:

- Ci-dessous nous dessinons une ligne verte de valeur 'ceiling' (plafond) et une ligne rouge de valeurs 'floor' (plancher).

```
Express f_Floor_Ceiling

Vars

Series
line_floor, line_ceiling;

Calculation

line_floor = floor(c);
line_ceiling = ceiling(c);

Interpretation
begin
end

plot (line_floor, lightred, 3);
plot (line_ceiling, green, 3);
```



Sum()

Définition:

- Donne la somme des éléments de serie[0], ... series[span – 1].
  - Le résultat sera nul si au moins un élément est nul.

Format:

- float Sum(series series, int span)

Exemple:

- Ci-dessous nous traçons la somme des cinq derniers prix de clôture:

```
Express f_Sum

Vars

Series
amount;

Input
$span(1, 100, 5);

Calculation

amount = sum(close, $span);

Interpretation
begin
end

plot (amount, green, 3);
```

Atr()

Définition:

- Mesure la volatilité des prix sur une période donnée sous la forme d'un pourcentage. Il s'agit de la moyenne du True Range (amplitude réelle) sur une période donnée. Le True Range est défini comme ceci:

  TrueRange = Plus haut de la période – plus bas de la période;
  Si (clôture de la période précédente <= plus haut de la période) alors TrueRange = max(TrueRange, plus haut de la période – clôture de la période précédente);
  Si (clôture de la période précédente >= plus bas de la période) alors TrueRange = max(TrueRange, clôture de la période précédente - plus bas de la période);
  Atr(14) est la moyenne mobile de TrueRange sur 14 périodes divisée par le prix de clôture.

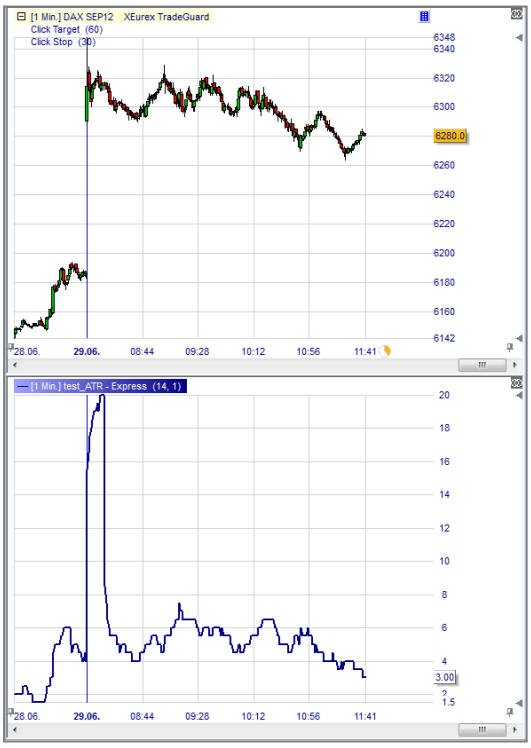
Format:

- float Atr (int span)

Exemple:

- l'indicateur ci-dessous trace trois versions différentes de l'ATR: 1/ ATR en points, 2/ ATR en pourcentage et 3/ ATR en ticks

```
Express f_Atr
Vars

input
$Period(1, 50, 14), $Type("points;percent;ticks",1);

series
ATRpoints, ATRpercent, ATRticks, ATRline;

calculation

// ATR in points
if $Type = 0 then ATRline =
Round(Atr($Period)/100*close/TickSize(),0)*TickSize()
;
// ATR in percent
if $Type = 1 then ATRline = Atr($Period);

//ATR in ticks
if $Type = 2 then ATRline =
Round(Atr($Period)/100*close/TickSize(),0);

interpretation
begin
end

plot (ATRline,blue,2);
```



AtrAbs()

Définition:

- Mesure la volatilité des prix sur une période donnée en points[6]. Il s'agit d'une moyenne du True Range (Amplitude réelle) sur une période donnée. Le True Range est défini comme ceci:
  TrueRange = Plus haut de la période – plus bas de la période;
  Si (clôture de la période précédente <= plus haut de la période) alors TrueRange = max(TrueRange, plus haut de la période – clôture de la période précédente);
  Si (clôture de la période précédente >= plus bas de la période) alors TrueRange = max(TrueRange, clôture de la période précédente - plus bas de la période);
  Atr(14) est la moyenne mobile de TrueRange sur 14 périodes divisée par le prix de clôture.

Format:

- float AtrAbs (int span)

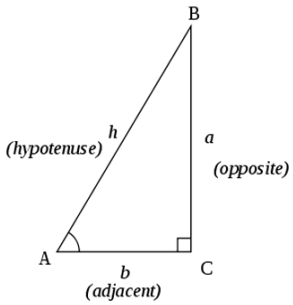
Exemple:

```
Express f_AtrAbs

vars

input $Period(1, 50, 14);
series ATRline;

calculation

ATRline = AtrAbs($Period);

interpretation
begin
end

plot (ATRline,red,2);
```

[6] AtrAbs() diffère d'Atr() car i est exprimé en valeur absolue (= points) et non en pourcentage.

Sine()

Définition:



- Donne la valeur du sinus d'un angle donné.
  - Considérant que le triangle à côté du sinus de l'angle AB-AC est égal au ratio (a/h):

Format:

- float Sine(float value)

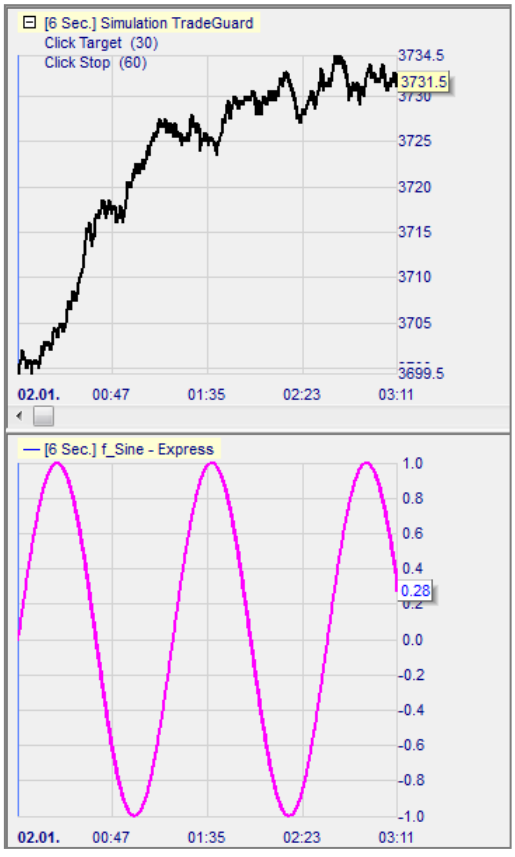
Exemple 1:

- Sine(0) = 0
- Sine(45) = 0.7071…
- Sine(90) = 1

Exemple 2:

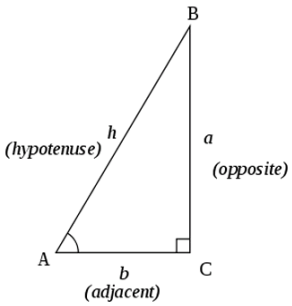
- Ci-dessous nous traçons le sinus de l'indice des barres:

```
Express f_Sine

Vars

Series
val;

Calculation
val = sine(CurrentBarIndex());

Interpretation
begin
end

plot (val, magenta, 2);
```



Cosine()

Définition:



- Donne le cosinus d'un angle donné.
    - Considérant que le triangle à côté du cosinus de l'angle AB-AC est égal au ratio (b/h):

Format:

- float Cosine (float value)

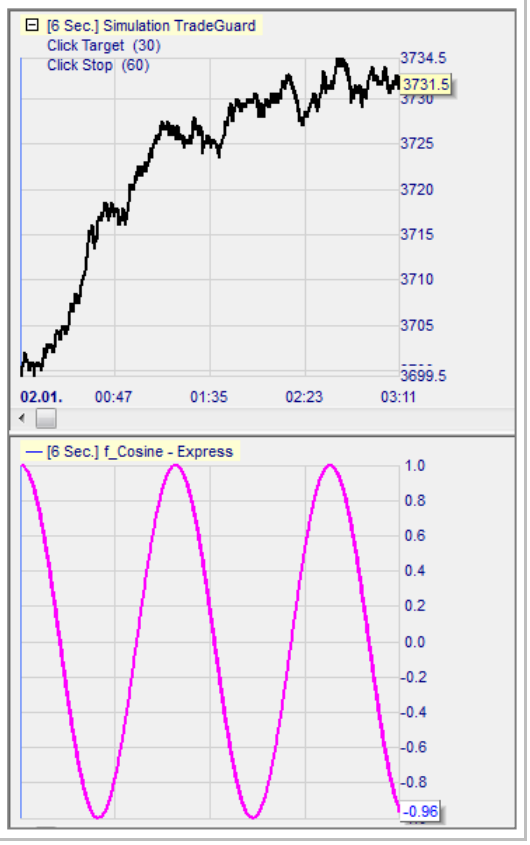
Exemple 1:

- Cosine(0) = 1
- Cosine(45) = 0.7071…
- Cosine(90) = 0

Exemple 2:

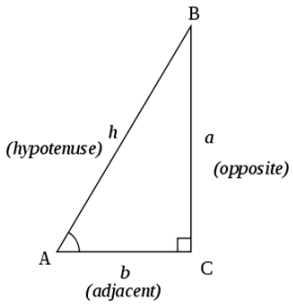
- Ci-dessous nous traçons le cosinus de l'indice des barres:

```
Express f_Cosine

Vars

Series
val;

Calculation

val = cosine(CurrentBarIndex());

Interpretation
begin
end

plot (val, magenta, 2);
```



Tangent()

Définition:



- Donne la tangente d'un angle donné.
    - Considérant que le triangle à côté de la tangente de l'angle AB-AC est égale au ratio (a/b):

Format:

- float Tangent (float value)

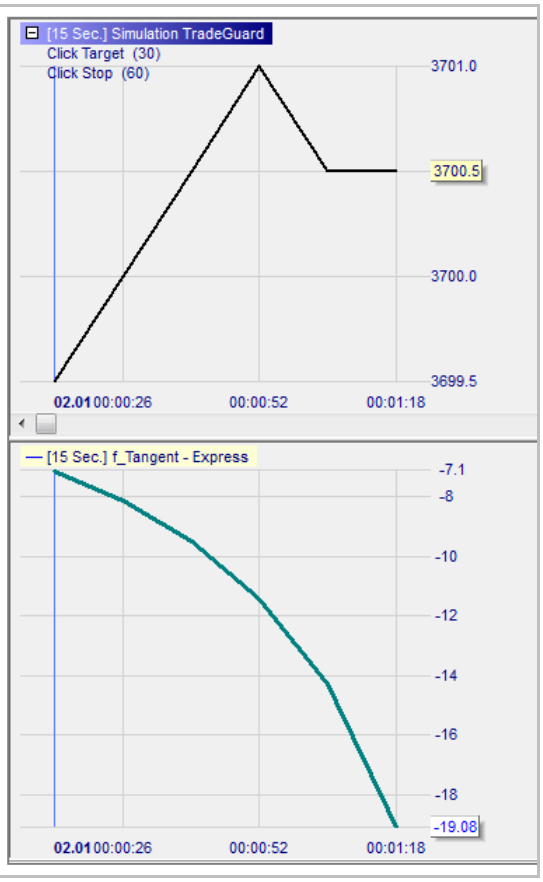
Exemple 1:

- Tangent(0) = 0
- Tangent(45) = 1

Exemple 2:

- Ci-dessous nous dessinons la tangente de l'indice des barres:

```
Express f_Tangent

Vars

Series
value;

Calculation

value = tangent(FinalBarIndex() - CurrentBarIndex());

Interpretation
begin
end

plot (value, cyan, 3);
```



ArcTangent()

Définition:

- Il s'agit de la fonction inverse de la fonction Tangent():
  - ArcTangent(Tangent(angle)) = angle
  - L'angle calculé est exprimé en un nombre entre – 90° and +90°
    - ArcTangent(0) = 0
    - ArcTangent(1) = 45

Format:

- float ArcTangent (float value)

Exemple:

- Ci-dessous nous affichons l'angle définissant la pente de la ligne verte de la moyenne mobile.
  - Deux paramètres sont entrés, afin d'ajuster le calcul de l'angle à ce que nous voyons:
    - Le nombre de bougies par centimètre
    - Le nombre de points par centimètre
  - Dans l'exemple ci-dessous l'angle va de 0° (ligne verte plancher) à environ 50° (ligne verte fortement ascensionnelle) avant de retourner à 0° (ligne verte qui s'aplatit).

```
Express f_ArcTangent

vars

series
ma, angle;

input
$NumberOfpointsPerCM(0.1, 10.0, 2.9, 0.1, 1),
$NumberOfcandlesPerCM(0.1, 10.0, 4.0, 0.1, 1), $period(1, 50, 10);

calculation

if IsFirstBar() then
begin
  CalculateAtEveryTick(false);
  MovingAverage(c, ma, $period);
end
angle = ArcTangent((ma -
ma[1])/$NumberOfpointsPerCM*$NumberOfcandlesPerCM);

interpretation
begin
end

plot(angle,blue,2);
```

Exp()

Définition:

- Donne la valeur exponentielle d'un nombre donné.
  - Il s'agit de la valeur de la constante *e* à la puissance d'un nombre donné.
  - *e* est un nombre a transcendental qui est la base de logarithmes naturels et sa valeur approximative est de 2.718281828….

Format:

- float Exp(float value)

Exemple 1:

- Exp(0) = 1
- Exp(1) = 2.718281828….

Exemple 2:

- Ci-dessous nous traçons la valeur exponentielle du prix de clôture:

```
Express f_Exp

Vars


Series
val;

Calculation

val = exp(close);

Interpretation
begin
end

plot (val, red, 2);
```

Log()

Définition:

- Donne la valeur de logarithme dans la base *e* d'un nombre donné.
  - *e* est un nombre a transcendental qui est la base de logarithmes naturels et sa valeur approximative est de 2.718281828….
  - Log(Exp(number) = nombre.

Format:

- float Log (float value)

Exemple 1:

- Log(0.1) = -2.30…
- Log(1) = 0
- Log(1000) = 6.91…

Exemple 2:

- Ci-dessous nous traçons le logarithme dans la base *e* du prix de clôture:

```
Express f_Log

Vars

Series
val;

Calculation

val = log(close);

Interpretation
begin
end

plot (val, cyan, 3);
```

Power()

Définition:

- Donne le résultat d'un nombre élevé à une puissance donnée.

Format:

- float Power (float value, float exponent)

Exemple 1:

- Power(3, 0) = 1
- Power(3, 1) = 3
- Power(3, 2) = 9

Example 2:

- Ci-dessous nous traçons une courbe exponentielle égale aux indices de barres élevés à la puissance 2:

```
Express f_Power

Vars

series
Result;

input
$Exponent(1, 50, 2);

Calculation

Result = Power(CurrentBarIndex(), $Exponent);

Interpretation
begin
end

plot(Result, green, 2);
```

SquareRoot()

Définition:

- Donne la racine carrée d'un nombre donné.
    - Note: le nombre doit être positif ou nul.

Format:

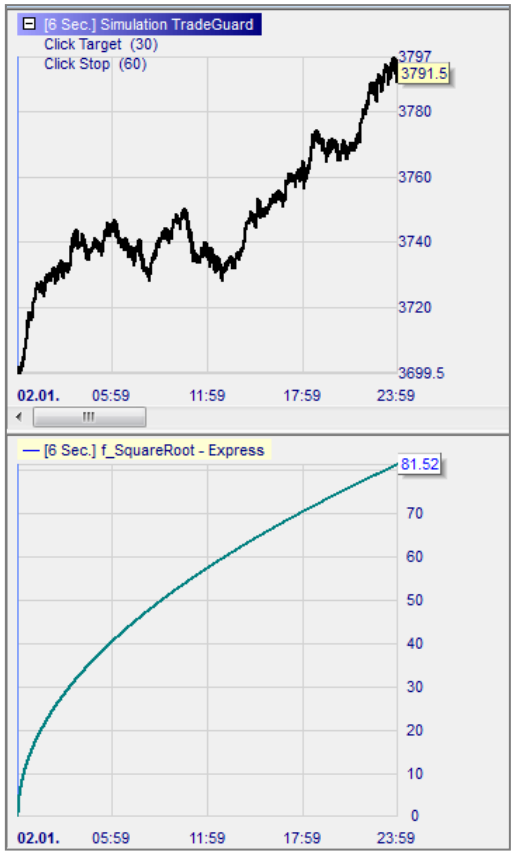
- float SquareRoot (float value)

Exemple 1:

Number	SquareRoot(Number)
1	1
4	2
9	3
16	4

Exemple 2:

```
Express f_SquareRoot

Vars

Series
val;

Calculation

val = SquareRoot(CurrentBarIndex());

Interpretation
begin
end

plot (val, cyan, 2);
```



Highest()

Définition:

- Donne la valeur la plus élevée pour les séries d'éléments [0], ... , series[span – 1].

Format:

- Float Highest (series series, int span)

Exemple:

- Ci-dessous la ligne verte TenBarHigh calcule la valeur la plus élevée des prix les plus élevés des dix dernières périodes:

```
Express f_Highest

Vars

Series TenBarHigh;

Calculation

TenBarHigh = Highest(High, 10);

interpretation
begin
end

plot (TenBarHigh, green, 4);
```

Lowest()

Définition:

- Donne la valeur la plus basse pour les séries d'éléments [0], ... , series[span – 1].

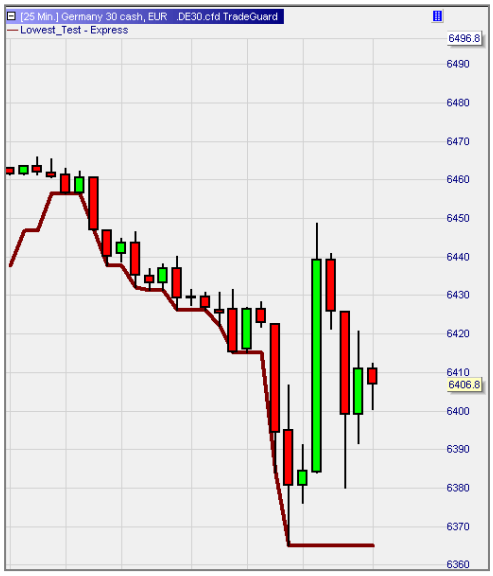
Format:

- float Lowest (series series, int span)

Example:

- Ci-dessous la ligne verte TenBarLow calcule la valeur la plus basse des prix les plus bas des dix dernières périodes:

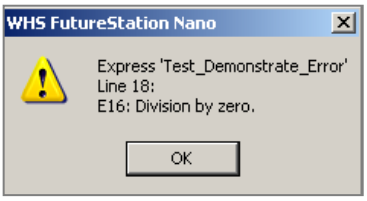
```
Express f_Lowest

Vars

Series TenBarLow;

Calculation

TenBarLow = Lowest(Low, 10);

interpretation
begin
end

plot (TenBarLow, red, 4);
```



IsZero()

Définition:

- IsZero(nombre) est vérifié si AbsValue(nombre) <= 0.000 001
  - Conséquence 1: cette fonction traite comme zéro tout chiffre qui n'est pas nul s'il est inférieur à 0.000 001.
  - Conséquence 2: si un nombre est divisé par un autre nombre non nul qui est inférieur à 0.000 001 le message suivant apparaitra:



Format:

- bool IsZero(float value)

Exemple:

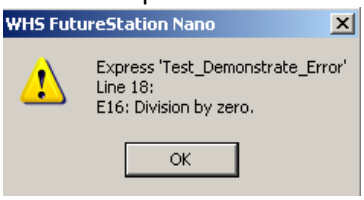
- Ci-dessous nous traçons la ligne CurrentBarIndex et nous colorons en rouge clair l'arrière-plan si IsZero(CurrentBarIndex()/1 000 000 000) est vrai:
  - Pour les premières 1001 barres la condition est vraie et l'arrière-plan est rouge clair.
  - A partir de la barre 1002 bar la condition est fausse.

```
Express f_IsZero

Vars

series
x, y;

numeric
a, b;

Calculation

if IsFirstBar() then a = 1/1000000000;

x = a*CurrentBarIndex();
y = CurrentBarIndex();

if IsZero(x) then Highlight("slot", "lightred");

Interpretation
begin
end

plot(y, blue, 2);
```

IsNonZero()

Définition:

- IsNonZero(number) est vérifiée si AbsValue(number) > 0.000 001
  - Conséquence 1: cette fonction ne capte pas un nombre non nul s'il est inférieur à 0.000 001.
  - Conséquence 2: si un nombre est divisé par un autre nombre non nul qui est inférieur à 0.000 001 le message suivant apparaitra:



Format:

- bool IsNonZero(float value)

Exemple:

- Ci-dessous nous traçons la ligne CurrentBarIndex et nous colorons en vert l'arrière-plan si IsNonZero(CurrentBarIndex()/1 000 000 000) est vrai:
  - Pour les premières 1001 barres la condition est fausse.
  - A partir de la barre 1002 la condition est vraie et l'arrière-plan est vert.

```
Express f_IsNonZero

Vars

series
x, y;

numeric
a, b;

Calculation

if IsFirstBar() then a = 1/1000000000;

x = a*CurrentBarIndex();
y = CurrentBarIndex();

if IsNonZero(x) then Highlight("slot", "green");

Interpretation
begin
end

plot(y, blue, 2);
```

Max()

Définition:

- Donne le plus grand de deux nombres.

Format:

- float Max(float value1, float value2)

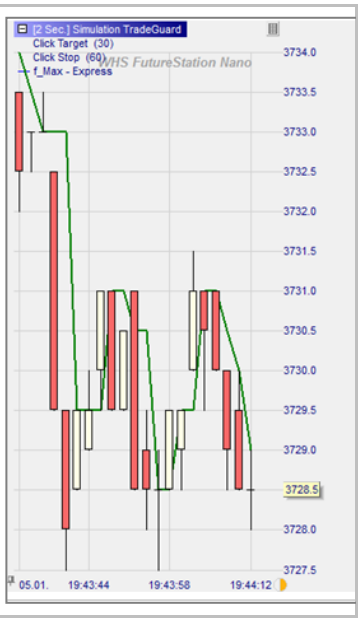
Exemple 1:

- Max(3, 8) donne 8.

Exemple 2:

- Ci-dessous nous traçons une ligne constituée du plus élevé des deux derniers prix de clôture:

```
Express f_Max

Vars

series
MaxClose;

Calculation

MaxClose = Max(close[2], close[1]) ;

interpretation
begin
end

plot(MaxClose, green, 2);
```



Min()

Définition:

- Donne le plus petit de deux nombres.

Format:

- float Min(float value1, float value2)

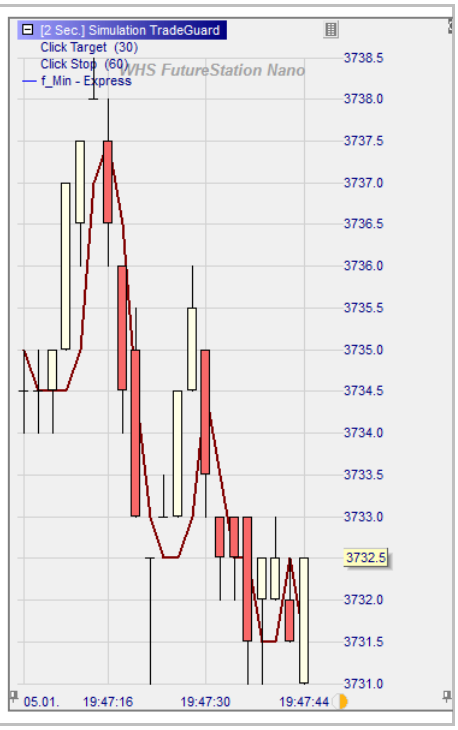
Example 1:

- Min(3, 8) donne 3.

Exemple 2:

- Ci-dessous nous traçons une ligne constituée du plus bas des deux derniers prix de clôture:

```
Express f_Min

Vars

series
MinClose;

Calculation

MinClose = Min(close[2], close[1]) ;

interpretation
begin
end

plot(MinClose, red, 2);
```



Round()

Définition:

- Donne le nombre rationnel le plus proche d'un nombre donné pour un nombre donné de décimales.
    - Le milieu (0.5) est arrondi au nombre rationnel le plus proche.

Format:

- float Round (float value, int precision)

Exemple 1:

- Round(1.1550, 0) = 1
- Round(1.1550, 1) = 1.2
- Round(1.1550, 2) = 1.16
- Round(1.1550, 3) = 1.155

Exemple 2:

- Ci-dessous nous traçons une courbe constituée des valeurs des cours EUR/USD arrondis à trois décimales:

```
Express f_Round

Vars

series
myClose;

Calculation

myClose = Round(close, 3);

Interpretation
begin
end

plot (myClose, green, 3);
```

RoundMultiple()

Définition:

- Donne un nombre qui est le plus proche multiple d'une base donnée à un nombre donné. Par exemple:
  - Si la base est 2.5 et le nombre est 3743.5, le résultat sera 3742.5:
    - 3743.5 / 2.5 = 1497.4 qui est arrondi à 1497.0
    - 1497 x 2.5 = 3742.5
  - Si la base est 2.5 et le nombre est 3739.5 le résultat sera 3740.0:
    - 3739.5 / 2.5 = 1495.8 qui est arrondi à 1496.0
    - 1496 x 2.5 = 3740.0

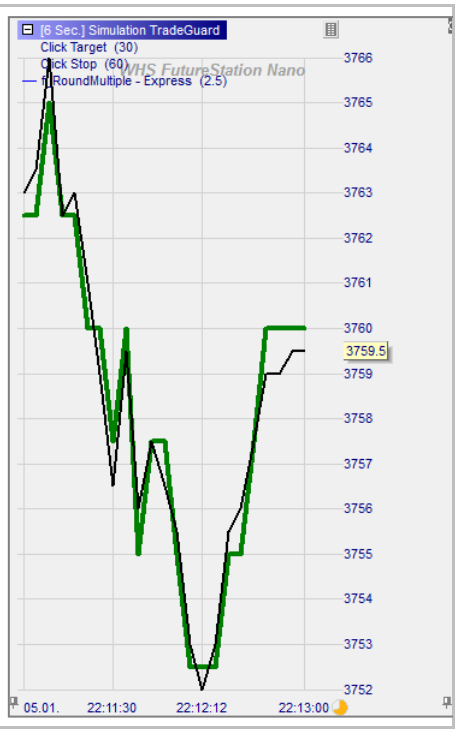
Format:

- float RoundMultiple (float value, float multiple)

Exemple:

- Ci-dessous nous traçons une ligne en vert du RoundMultiple appliqué au prix de clôture et basé sur un multiple de 2.5:

```
Express f_RoundMultiple

Vars

input
$Base(0.1, 5.0, 2.5, 0.1, 1);

Series
val;

Calculation
val = RoundMultiple (close, $Base);

Interpretation
begin
end

plot (val, green, 4);
```



NormalCDF()

Définition:

- Donne la valeur de la fonction distribution cumulative normale (CDF).

  Il s'agit de la fonction de distribution cumulative avec un moyen de 0 et un écart-type de 1. Cela peut être utile pour calculer des fonctions de cotation à base de probabilités comme Black-Scholes.

Format:

- NormalCDF (float value)

Exemple:

- Ci-dessous nous traçons une courbe représentant la fonction CDF normale (utilisez les paramètres a et b pour l'ajuster):

```
Express f_NormalCDF

vars

series
CBI, x;

input
$a(1, 500, 100), $b(1, 500, 10);

calculation

CBI = CurrentBarIndex();
x = NormalCDF((CBI - $b) / $a);

interpretation
begin
end

plot(x, red, 2);
```

NormalPDF()

Définition:

- Donne la valeur de la fonction de densité de probabilité normale (PDF).

  Il s'agit de la fonction de densité de probabilité avec un moyen de 0 et un écart-type de 1. Cela peut être utile pour calculer des fonctions de cotation à base de probabilités comme black-Scholes.

Format:

- NormalPDF (float value)

Exemple:

- Ci-dessous nous traçons une courbe représentant la fonction PDF normale (utilisez les paramètres a et b pour l'ajuster):

```
Express f_NormalPDF

vars

series
CBI, x;

input
$a(1,500,10), $b(1,500,150);

calculation

CBI = CurrentBarIndex();
x = NormalPDF((CBI - $b)/$a);

interpretation
begin
end

plot(x, blue, 2);
```

Les fonctions de charting

Plotline()

Définition:

- Trace une ligne à un niveau donné.
  - Tant les couleurs prédéfinies que les couleurs RGB peuvent être utilisées.

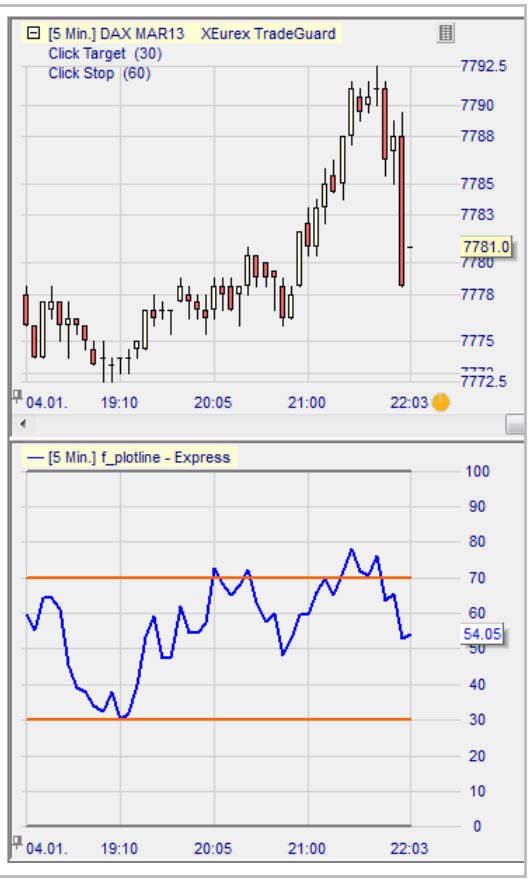
Format:

- plotline (<constant ou variable>, <colorname>, <pen width>);

Exemple:

- Ci-dessous nous traçons une RSI:
  - Les lignes horizontales 100 et 0 sont colorées en utilisant une couleur prédéfinie, gris.
  - Les lignes horizontales 70 et 30 sont colorées en utilisant RGB et sont définies en utilisant des variables numériques.

```
Express f_plotline

Vars

Series
myRSI;

Numeric
upper(70), lower(30);

Calculation
If IsFirstBar() then RSI(close, myRSI, 14);

interpretation
begin
end

plot(myRSI, “lightblue”, 2);
plotline(upper, 250, 100, 0, 2);
plotline(lower, 250, 100, 0, 2);
plotline(0, “grey”, 2);
plotline(100, “grey”, 2);
```

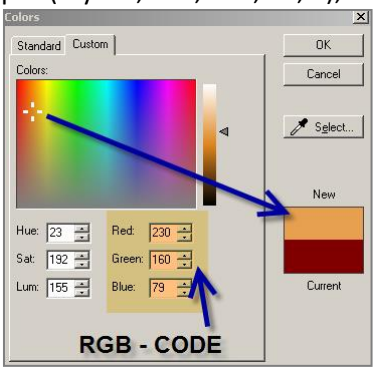


Plot()

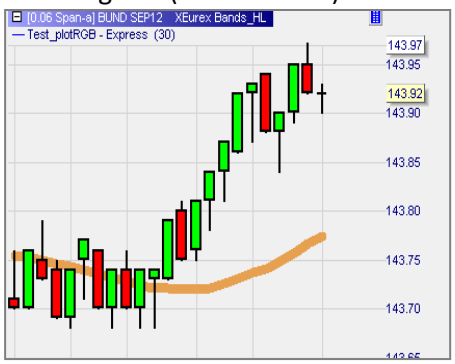
Définition:

- Trace une courbe simple.
  - Tant les couleurs prédéfinies[7] que les couleurs RGB peuvent être utilisées (RGB= Red-Green-Blue).
  - En cas d'erreur de frappe lors de la définition de la couleur, celle-ci sera bleue.
  - Les couleurs RGB sont définies par trois chiffres. Par exemple:

plot(myMA, 230, 160, 79, 8);



Resulting line (thickness = 8):



Format:

- plot(<series name>, "<color name>", <pen width>);

Exemple:

- Ci-dessous nous traçons deux moyennes mobiles simples:
  - La première est basée sur le code RGB.
  - La deuxième sur la couleur "bleu" prédéfinie.

```
Express f_plotRGB
Vars
Series
myMA, myMA2;

Calculation
If IsFirstBar() then
begin
  MovingAverage(close, myMA, 30);
  MovingAverage(close, myMA2, 10);
end

Interpretation
begin
end
plot(myMA, 230, 160, 79, 5);
plot(myMA2, "blue", 5);
```

[7] Les couleurs prédéfinies sont: black, white, red, green, blue, magenta, yellow, grey, lightred, lightgreen, lightblue.

Plotband()

Définition:

- Trace une bande faite de deux courbes et remplit l'espace entre les deux en couleur.
- Pour Plotband() un nombre quasi infini de couleurs peut être choisi en utilisant la palette de couleur RGB (RGB = Red-Green-Blue).
- Se référer à la section relative aux fonctions de traçage pour apprendre à créer un code couleur RGB.

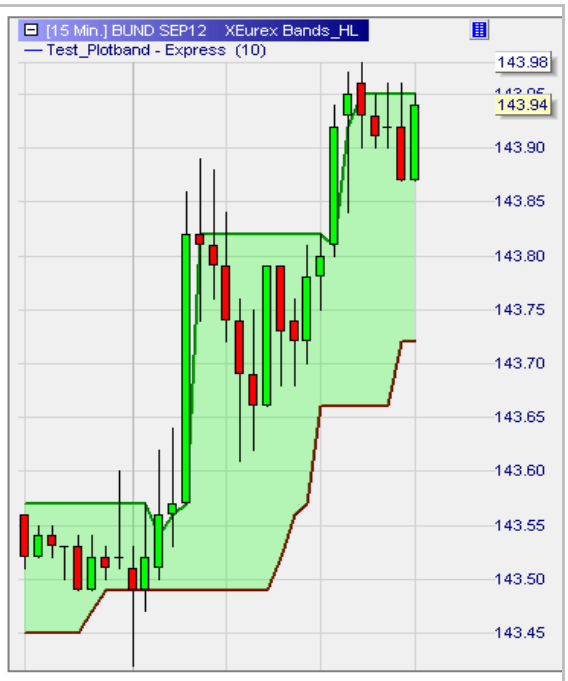
Format:

- plotband(<upper series name>, <color name>, <pen width>, <lower series name>, <color name>, <pen width>,<fill color>);
- plotband(<upper series name>, int red, int green, int blue, <pen width>, <lower series name>, int red2, int green2, int blue2, <pen width>, int red3, int green3, int blue3);

Exemple:

- Ci-dessous nous traçons une bande basée sur les hauts les plus hauts et les bas les plus bas des 10 dernières barres:
  - La courbe des hauts les plus hauts est verte et la courbe des bas les plus bas est rouge.
  - La zone entre les deux courbes est en vert clair.
  - Les couleurs doivent être entrées entre guillemets: "blue".

```
Express f_plotband

Vars

series
upper, lower, upper1, lower1;

input
$span (0, 200, 10);

Calculation
upper1 = Highest(close, $span);
lower1 = Lowest(close, $span);

upper = upper1[1];
lower = lower1[1];

interpretation Bands (close, lower, upper);

PlotBand(upper, "green", 2, lower, "red", 2,
"lightgreen");
```



Plotcrossinglines()

Définition:

- Dessine une bande composée de deux courbes et peint en deux couleurs l'espace compris entre celles-ci.
- Pour Plotcrossinglines() un nombre quasi infini de couleurs peut être choisi en utilisant la palette de couleur RGB (RGB = Red-Green-Blue).
- Se référer à la section relative aux fonctions de traçage pour apprendre à créer un code couleur RGB.

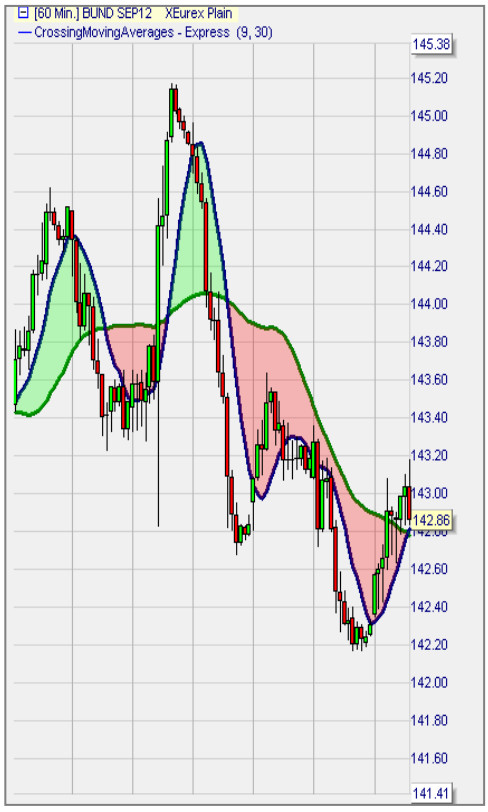
Format:

- plotcrossinglines (<series1 name>, <color name>, <pen width>, <series2 name>, <color name>, <pen width>, <fill color series1 above series2>, <fill color series1 below series2>);
- plotcrossinglines (<series1 name>, int red, int green, int blue, <pen width>, <series2 name>, int red2, int green2, int blue2, <pen width>, int red3, int green3, int blue3, int red4, int green4, int blue4);

Exemple:

- Ci-dessous, nous dessinons une bande sur base des plus hauts hauts et des plus bas bas des 10 dernières chandelles:
  - La courbe des plus hauts hauts est en vert, et la courbe des plus bas bas en rouge.
  - L'espace entre les deux courbes est en vert clair lorsque la MM rapide est au-dessus de la MM lente et en rouge clair le cas contraire.
  - Les couleurs sont indiquées avec des guillemets: "blue".

```
Express f_CrossingMovingAverages
Vars

Series
FastMA, SlowMA;

Input
$spanFMA(1, 200, 9), $spanSMA(1, 200, 30);

Calculation
If IsFirstBar() then
begin
  MovingAverage(close, SlowMA, $spanSMA);
  MovingAverage(close, FastMA, $spanFMA);
end

Interpretation
begin
end

plotcrossinglines(SlowMA, "green", 3, FastMA, "blue",
3, "lightred", "lightgreen");
```



Plotbars()

Définition:

- Dessine un graphique en barres.
  - Par défaut, les couleurs du graphique en barres sont les mêmes que celles du graphique principal. Les couleurs par défaut du graphique en barres peuvent être modifiées dans Extras / Couleurs (Barre Haussière Bull, Barre Baissière).
  - Des couleurs personnalisées peuvent également être utilisées (les couleurs prédéfinies et les couleurs RGB sont toutes deux possibles).
  - Reportez-vous à la section sur les fonctions de traçage pour savoir comment créer un code couleur RGB.

Format:

- plotbars(<open series>, <close series>, <high series>, <low series>);
- plotbars(<open series>, <close series>, <high series>, <low series>, <color name bull bar>,<color name bear bar>);
- plotbars (<open series>, <close series>, <high series>, <low series>, int red1, int green1, int blue1, int red2, int green2, int blue2);

Exemple:

- Ci-dessous nous traçons un graphique en barres dans des couleurs personnalisées :

```
Express f_Plotbars

Calculation

Interpretation
begin
end

plotbars(o ,c, h, l, "cyan","lightyellow");
```

Plotcandles()

Définition:

- Dessinez un graphique en bougies.
  - Par défaut, les couleurs du graphique en bougies sont les mêmes que dans le graphique principal. Les couleurs par défaut du graphique en bougies peuvent être modifiées dans Extras / Couleurs (Bougie haussière, Bougie baissière)
  - Toutefois, les couleurs personnalisées peuvent être utilisées (aussi bien les couleurs RGB que prédéfinies sont possibles).
  - Consultez la section sur les fonctions de « Plotting » (traçage) pour apprendre à créer un code couleur RGB.

Format:

- plotcandles(<open series>, <close series>, <high series>, <low series>);
- plotcandles(<open series>, <close series>, <high series>, <low series>, <color name bull candle>,<color name bear candle>);
- plotcandles(<open series>, <close series>, <high series>, <low series>, int red1, int green1, int blue1, int red2, int green2, int blue2);

Exemple:

- Nous avons dessiné deux graphiques en bougies, avec les couleurs prédéfinies et RGB, qui se basent sur le graphique en barre du dessus.

Express f_PlotcandleChart vars calculation interpretation begin end plotcandles (open, close, high, low);	
Express f_PlotcandleChart vars calculation interpretation begin end plotcandles (open, close, high, low, 124, 123, 532, 124, 534, 123);	

GetPriceFormat()

Définition:

- Applique le format de l'axe y du graphique principal à la sous-fenêtre.

Format:

- string GetPriceFormat()

Exemple:

- Ci-dessous nous appliquons à l'axe y de notre indicateur le même format que celui du graphique principal.

```
Express f_GetPriceFormat_EFX

Vars

series
daxopen(studyDaxsep.open),
daxclose(studyDaxsep.close),
daxhigh(studyDaxsep.high),
daxlow(studyDaxsep.low);

Calculation
SetYscaleFormat (GetPriceFormat());

Interpretation
begin
end

plotcandles(daxopen, daxclose, daxhigh, daxlow);
```

SetYscaleFormat()

Définition:

- Définit le format de l'axe y de la sous-fenêtre où les indicateurs sont montrés lorsqu'ils ne sont pas dans le graphique principal.
    - Supporte tous les formats utilisés pour la fonction C "printf()". Les plus importants sont:
      "%f" point des décimales flottant
      "%6.2f" arrondi à deux décimales
      "%g" enlève les zéros superflus
      "%e" notation scientifique

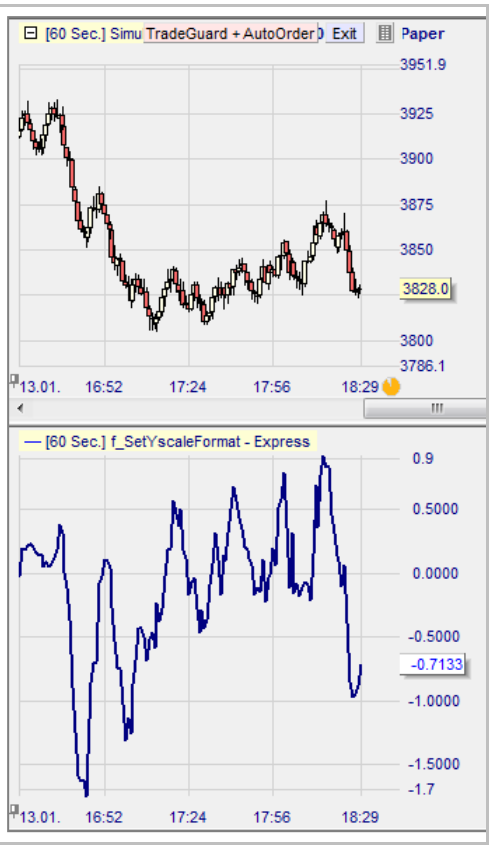
Format:

- Void SetYscaleFormat (string format)

Exemple:

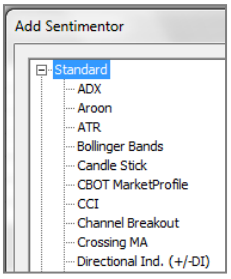
- Ci-dessous nous avons donné à l'axe y de la sous-fenêtre le format d'un nombre à 4 décimales.

```
express f_SetYscaleFormat

vars

series
a;

numeric ;
input ;

calculation

a = (c - c[10])/c[10]*100;
SetYscaleFormat("%9.4f");

interpretation
begin
end

plot(a,blue,2);
```



Importer des séries

Importer une série d'un indicateur programmé dans la plateforme



Définition:

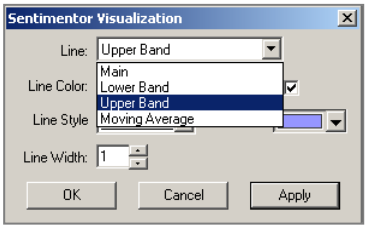
- Importer une série d'un des indicateurs standards de la plateforme (ci-contre un extrait de la liste des indicateurs standards).

Syntaxe:

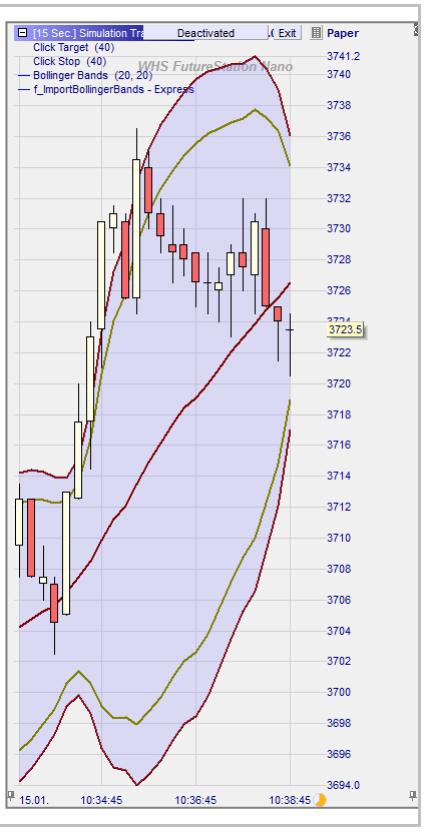
- series myseries (indicatorName.seriesName);
  - Important: Au moment d'écrire indicatorName enlever tous les espaces et les caractères non alphabétiques, comme les nombres, _, -, +, /, …, etc.
  - S'il y a deux séries du même indicateur, il faut indiquer lequel il faut importer, par exemple series BBUB(BollingerBands.UpperBand2), le nombre 2 indique qu'il s'agit de la série en rapport avec le second indicateur.

Exemple:

- Ci-dessous, nous créons deux bandes similaires aux Bollinger Bands: La même moyenne mobile mais avec des bandes 20% plus proches.
  - Nous avons le nom des séries à partir de la fenêtre de visualisation (ci-contre).




```
Express f_ImportBollingerBands

Vars

series

BBMA(BollingerBands.MovingAverage), NBUB, NBLB,
BBUB(BollingerBands.UpperBand),
BBLB(BollingerBands.LowerBand);

Calculation

NBUB = BBMA + (BBUB - BBLB)*0.8*0.5;
NBLB = BBMA - (BBUB - BBLB)*0.8*0.5;

interpretation
begin
end

plot(NBUB, yellow, 2);
plot(NBLB, yellow, 2);
```



Importer une série issue d'un autre indicateur express

Définition:

- Importer une série d'un indicateur encodé en express.

Syntaxe:

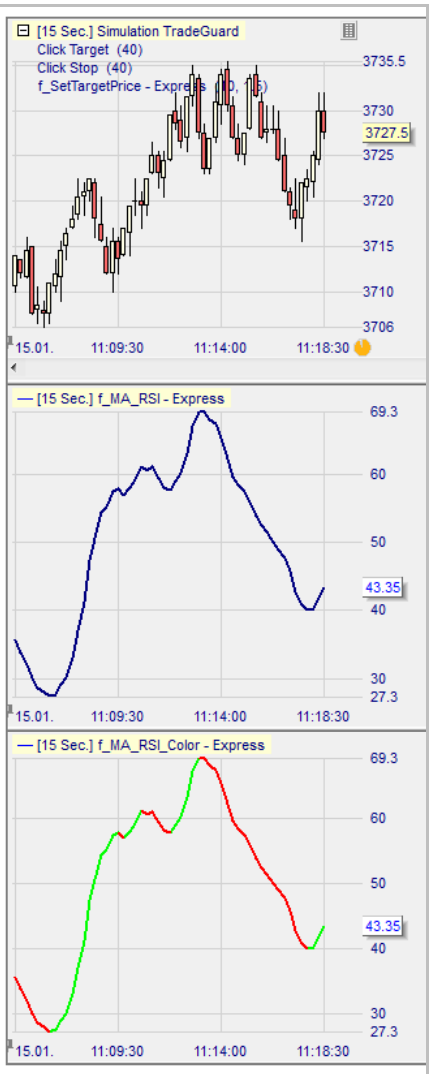
- series myseries (indicatorName**Express**.seriesName);
  - Notez l'ajout du terme **Express** ci-dessus.
  - Important: Au moment d'écrire indicatorName enlever tous les espaces et les caractères non alphabétiques, comme les nombres, _, -, +, /, …, etc. Cette restriction ne s'applique pas au nom de la série même.

Exemple:

- Le premier indicateur ci-dessous crée une moyenne mobile 6 d'un RSI(14) appelé MAR (2eme graphique). Le second indicateur importe MAR et le remontre en couleur:

```
Express f_MA_RSI

Vars

Series
R, MAR;

Calculation

If IsFirstBar() then
begin
  RSI(c, R, 14);
  MovingAverage(R, MAR, 6);
end

Interpretation
begin
end

plot(MAR, blue, 2);
```

```
express f_MA_RSI_Color

vars

series
MAR(fMARSIExpress.MAR),
MARup, MARdw;

calculation

if MAR >= MAR[1] then
begin
  MARup = MAR;
  MARup[1] = MAR[1];
end
else MARup = void;

if MAR < MAR[1] then
begin
  MARdw = MAR;
  MARdw[1] = MAR[1];
end
else MARdw = void;

interpretation
begin
end

plot(MARup, lightgreen, 2);
plot(MARdw, lightred, 2);
```



Importer une série de prix issue d'un symbole

Définition:

- Importe toutes les séries open, close, high, low et volume du symbole 2 dans une étude du symbole 1.

Syntaxe:

- series myseries (**study**SymbolName.seriesName);
  - Nous utilisons l'indicateur Study pour amener le symbole 2 dans notre étude.
    - Le graphique de symbole 2 peut être montré dans une sous-fenêtre ou dans la fenêtre principale.
  - Notez le terme Study ci-dessus suivi du nom du symbole. SeriesName peut être: open, close, high, low ou volume.
  - Important: Au moment d'écrire indicatorName enlever tous les espaces et les caractères non alphabétiques, comme les nombres, _, -, +, /, …, etc.
  - Si nous importons seulement un symbole nous pouvons laisser tomber quelques informations contenue dans l'expression "studySymbolName" (voir l'exemple ci-dessous).

Exemple:

- Ci-dessous le symbole 1 est le Dax future. Le symbole 2 importé est le Mini Nasdaq future. Le graphique au-dessus montre le symbole 1. Le graphique du milieu reflète le symbole 2. Le troisième est un graphique en barre, construit avec la série de prix importée du symbole 2.
  - C'est parce que nous n'importons qu'un seul symbole, que nous pouvons utiliser une expression réduite studyMININSDQ.close au lieu de studyMININSDQMAR.close.

```
express f_ImportAnotherSymbol

vars

series
oo(studyMININSDQMAR.open),
cc(studyMININSDQ.close),
ll(studyMINI.low),
hh(study.high);

numeric ;
input ;

calculation

interpretation
begin
end

plotbars(oo, cc, hh, ll);
```

Importation de valeurs de tableau à partir d'un symbole.

Définition:

- Importation de valeurs de tableau à partir d'un symbole.
- Exemple d'utilisation:
  - Transfert de valeurs d'indicateurs d'une unité de temps supérieure vers une unité de temps inférieure.
  - Importation de valeurs d'indicateurs à partir d'un autre symbole.

Syntaxe:

- Nous avons besoin d'un indicateur d'exportation dans le symbole 1 et d'un indicateur d'importation dans le symbole 2.
- Les deux indicateurs ont besoin d'une variable de tableau (variable de tableau d'exportation → variable de tableau d'importation).
- Valeur d'exportation (symbole 1) :
  - array arrayName[0];
- Valeur d'importation (symbole 2) :
  - array arrayName[study.indicatorName.arrayName];
- Important : Lorsque vous écrivez le nom de l'indicateur, supprimez tous les espaces et les caractères non alphabétiques, c'est-à-dire les chiffres, _, -, +, /, ..., etc.

Exemple:

- Dans cet exemple, nous calculons une moyenne mobile simple dans l'agrégation 60 minutes (symbole 1) et importons la valeur actuelle de la moyenne mobile dans l'agrégation 10 minutes (symbole 2) en utilisant les variables du tableau. Les deux symboles affichent le FDAX Future dans les deux cadres temporels différents.
  - Exportation de la syntaxe + exportation du graphique :

```
express ArrayToExport
vars

array exportdata[0];

series ma;

calculation
If IsFinalBar() then
begin
  MovingAverage(c,ma,10);
  SetArraySize(exportdata, 1);
  Exportdata[0] = ma;
End

Interpretation
begin end
Plot (ma, green, 2);
```

- Importing syntax + importing chart:

```
express ArrayToImport

vars

array
importdata[study.ArrayToExportExpress.exportdata];



calculation
if IsFinalBar() then SetArraySize(importdata,1);



interpretation

begin
end



plotline (importdata[0], blue, 2);
```

Créer des stops et targets personnalisés

SetIntraPeriodUpdate()

Cette fonction ne peut être utilisée que pour les stops.

Définition:

- Autorise le stop à être mis à jour tick par tick à chaque barre.
  - Si la position est ouverte sans SetIntraPeriodUpdate, le stop est mis à jour tick par tick uniquement à la barre d'entrée (entry bar). Pour les autres barres le stop est mis à jour une fois à la clôture de chaque barre.
  - Les stops avec SetIntraPeriodUpdate sont ignorés en backtesting.

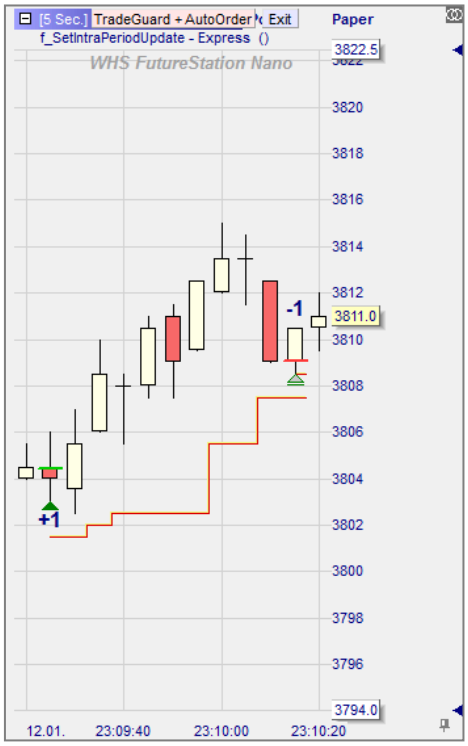
Format:

- void SetIntraPeriodUpdate()

Exemple:

- Ci-dessous nous avons créé un High/Low stop qui est mis à jour tick par tick, et qui donc est positionné au plus bas 'bas' des 5 dernières barres ou au plus haut 'haut' des 5 dernières barres.

```
express Stop f_SetIntraPeriodUpdate

Calculation

SetIntraPeriodUpdate();

If Marketposition() = 1 then SetStopPrice(lowest(l, 5));
If Marketposition() = -1 then SetStopPrice(highest(h,
5));
```



EntryPrice()

Cette fonction ne peut être utilisée que pour les stops.

Définition:

- Donne deux valeurs différentes en fonction du fait que TradeGuard[8] soit activé ou pas et du moment de son activation:
  - TradeGuard déactivé: EntryPrice = Executed price[9].
  - TradeGuard activé avant ouverture d'une position: EntryPrice = Executed price[9].
  - TradeGuard activé après ouverture d'une position: EntryPrice = TradeGuard activation price[3].

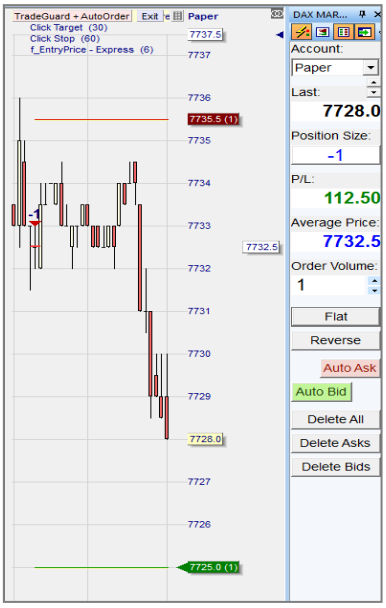
Format:

- float EntryPrice()

Exemple:

- Ci-dessous nous créons un stop à 6 ticks (3 points) du entry price. Vu que le TradeGuard a été activé quand la position a été ouverte entry price = executed price:
  - Executed price de 7732.5 comme on peut le visualiser sous Average Price
  - Sur le graphique, le stop est 3 points (6 ticks) au-dessus de 7732.5 comme souhaité.

```
Express Stop f_EntryPrice

vars

numeric
StopLong, StopShort;

calculation

if BarsSinceEntry() = 0 then
begin
  StopLong = EntryPrice() - 6*TickSize();
  StopShort = EntryPrice() + 6*TickSize();
end
If MarketPosition() = 1 then SetStopPrice(StopLong);
else
if MarketPosition() = -1 then SetStopPrice(StopShort);
```



[8] Avec l'activation du Tradeguard, les stops sont placés automatiquement par la plateforme.
[9] Nous nous référons à deux prix. Le premier est le prix actuel auquel la position a été ouverte. Il est appelé **Executed price**. Le second est le prix au moment de l'activation du TradeGuard. Il est appelé **TradeGuard activation price**. Ces prix sont différents.

EntryPriceOriginal()

Cette fonction ne peut être utilisée que pour les stops.

Définition:

- Donne le executed price[10].

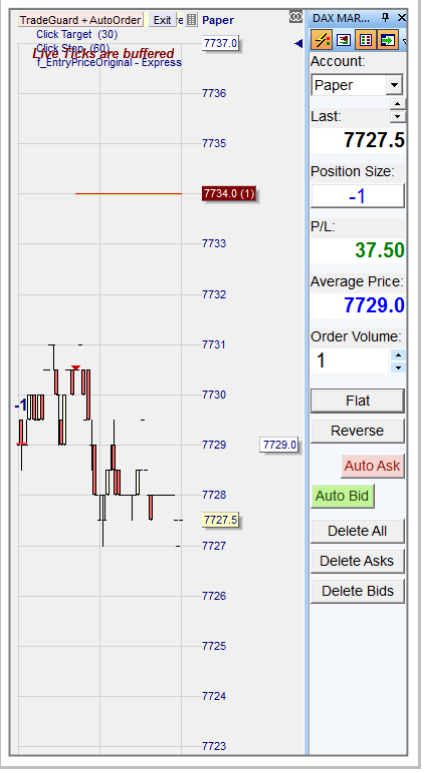
Format:

- float EntryPriceOriginal()

Exemple:

- Ci-dessous nous créons un fixed stop à 10 ticks (5 points) du prix d'entrée original:
  - Nous ouvrons la position au executed price de 7729.0 comme on peut le visualiser sous « Average Price » et sur le graphe avec le signe '-1'.
  - Nous activons TradeGuard par la suite et la plateforme positionne notre stop 10 ticks (5 points) au-dessus du niveau d'entrée, ce que nous souhaitions.

```
express Stop f_EntryPriceOriginal

vars

numeric
StopLong, StopShort;

calculation

if BarsSinceEntry() = 0 then
begin
  StopLong = EntryPriceOriginal() - 10*TickSize();
  StopShort = EntryPriceOriginal() + 10*TickSize();
end

If MarketPosition() = 1 then SetStopPrice(StopLong);
else
if MarketPosition() = -1 then SetStopPrice(StopShort);
```



[10] Nous nous référons à deux prix. Le premier est le prix actuel auquel la position a été ouverte. Il est appelé **Executed price**. Le second est le prix au moment de l'activation du TradeGuard. Il est appelé **TradeGuard activation price**. Ces prix sont différents.

BarsSinceEntry()

Cette fonction ne peut être utilisée que pour les stops.

Définition:

- donne le nombre de barres depuis l'ouverture de la position.
  - Donne 0 à la première barre, 1 à la seconde, 2 à la troisième, …

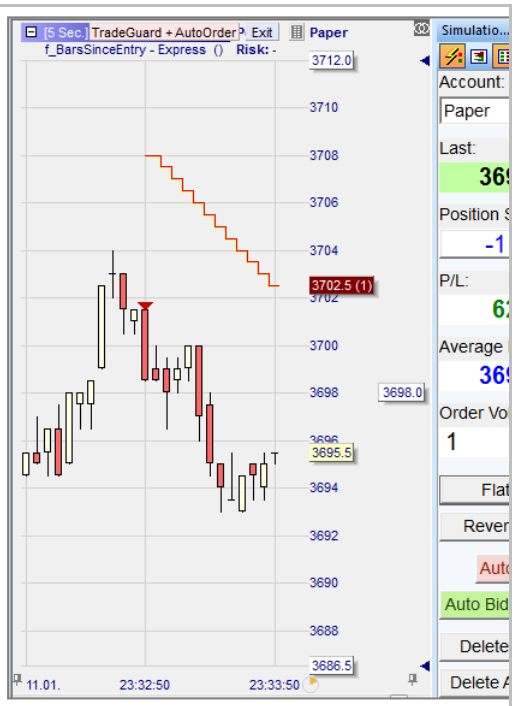
Format:

- int BarsSinceEntry()

Exemple:

- Ci-dessous nous créons un linear stop qui démarre à 20 ticks de notre entry price avant de se rapprocher d'un tick à la clôture de chaque barre. BarsSinceEntry() est utilisé pour définir la barre d'entrée et pour adapter ensuite le stop barre après barre.

```
express Stop f_BarsSinceEntry

vars

numeric
StpLong, StpShort;

calculation

if BarsSinceEntry() = 0 then
begin
  StpLong = EntryPriceOriginal() - 20*TickSize();
  StpShort = EntryPriceOriginal() + 20*TickSize();
end
else
begin
  StpLong = max(StpLong, c - (20 +
BarsSinceEntry())*TickSize());
  StpShort = min(StpShort, c + (20 -
BarsSinceEntry())*TickSize());
end

if MarketPosition() = 1 then SetStopPrice(StpLong);
if MarketPosition() = -1 then SetStopPrice(StpShort);
```



IsIntradayEntry()

Cette fonction ne peut être utilisée que pour les stops.

Définition:

- S'avère vrai si la position a été ouverte et pas encore fermée dans la bougie en cours.

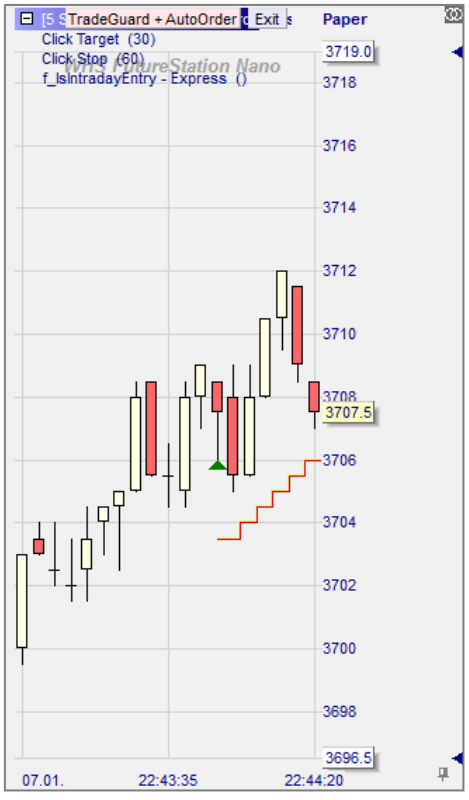
Format:

- bool IsIntradayEntry()

Exemple:

- Ci-dessous, nous créons un linear stop qui démarre à 10 ticks du prix d'entrée d'origine avant de se rapprocher d'un tick à la clôture de chaque barre. IsIntradayEntry est utilisé pour définir la barre d'entrée (comme le fait BarsSinceEntry() = 0).

```
Express Stop f_IsIntradayEntry

vars

numeric
last, StopL, StopS;

calculation

if MarketPosition() = 1 then
begin
  if IsIntradayEntry() then
  last = EntryPriceOriginal() - 10*TickSize();
  else last = max(last, last + TickSize());
end
else if MarketPosition() = -1 then
begin
  if IsIntradayEntry() then
  last = EntryPriceOriginal() + 10*TickSize();
  else last = min(last, last - TickSize());
end
SetStopPrice(last);
```



MarketPosition()

Cette fonction ne peut être utilisée que pour les stops.

Définition:

- donne la position actuelle: 1 = long, -1 = short.
    - Note: MarketPosition() = 0 ou un autre nombre ne fonctionne pas.

Format:

- int MarketPosition()

Exemple:

- Ci-dessous nous créons un fixed stop à 10 ticks (5 points) du prix d'entrée original. MarketPosition nous autorise à placer un stop pour chaque position longue ou short:

```
express Stop f_MarketPosition

vars

numeric
StopLong, StopShort;

calculation

if BarsSinceEntry() = 0 then
begin
  StopLong = c - 10*TickSize();
  StopShort = c + 10*TickSize();
end

if MarketPosition() = 1 then SetStopPrice(StopLong);
else
if MarketPosition() = -1 then SetStopPrice(StopShort);
```

MinPriceEntryBar() / MaxPriceEntryBar()

Cette fonction ne peut être utilisée que pour les stops.

Définition:

- MinPriceEntryBar()
  Si une position est ouverte quand TradeGuard est activé, MinPriceEntryBar donne le plus bas prix tradé dans la première barre (la barre d'entrée) après que la position soit ouverte.
  Si TradeGuard est activé quand une position est déjà ouverte, MinPriceEntryBar donne le plus bas prix tradé dans la première barre (la barre d'entrée) après l'activation de TradeGuard.

- MaxPriceEntryBar()
  Si une position est ouverte quand TradeGuard est activé, MaxPriceEntryBar donne le plus haut prix tradé dans la première barre (la barre d'entrée) après que la position soit ouverte.
  Si TradeGuard est activé quand une position est déjà ouverte, donne leplus haut prix tradé dans la première barre (la barre d'entrée) après l'activation de TradeGuard.

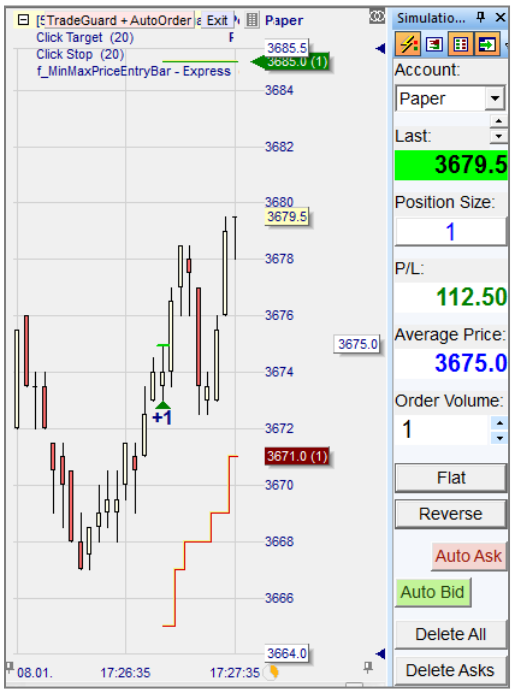
Format:

- float MaxPriceEntryBar(), float MinPriceEntryBar()

Exemple:

- Ci-dessous nous créons un stop qui est à 20 ticks du prix d'entrée

```
Express Stop f_MinMaxPriceEntryBar

Vars

numeric
StopH, StopL;

Calculation

if BarsSinceEntry() = 0 then
begin
  StopH = MaxPriceEntryBar() + 20*TickSize();
  StopL = MinPriceEntryBar() - 20*TickSize();
end
else
begin
  StopH = min(StopH, Highest(h, 10));
  StopL = max(StopL, Lowest(l, 10));
end

If (MarketPosition() = -1) then SetStopPrice(StopH);
Else SetStopPrice(StopL);
```



SetLongTrigger() / SetShortTrigger()

Définition:

- Crée des niveaux d'entrée conditionnels[11].
  - Suivant un certain signal, nous définissons un niveau de prix à partir duquel un ordre marché, stop ou limite sera déclenché pour ouvrir une position. Si l'ordre n'est pas exécuté pendant la première période qui suit le signal, il sera automatiquement annulé, et il faudra attendre un nouveau signal.

Format:

- void SetLongTrigger (float value)

Exemple 1:

- Ci-dessous nous générons des Signaux bases sur le croisement de deux moyennes mobiles. Nous essayons d'entrer à des prix plus favorables en utilisant un ordre **conditional limit**[12] placé à mi-hauteur de la barre du signal. Notez ci-dessous les conditions paramétrées ainsi que la description de ce qui arrive en bleu:

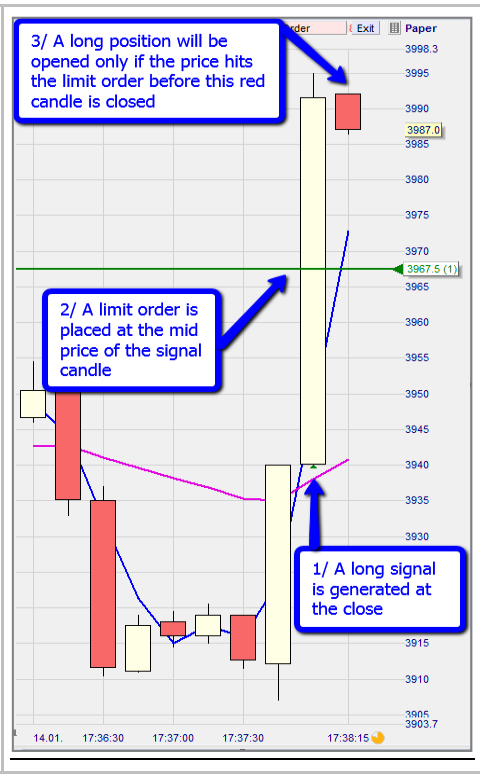
Express Stop f_SetTrigger

Vars

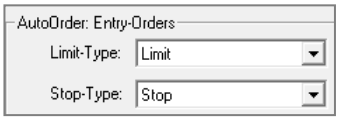
numeric
bl, bs;

Calculation

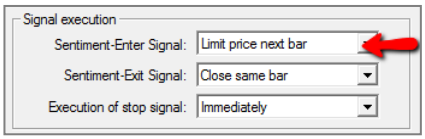
bl = (h + l)/2;
bs = (h + l)/2;
SetLongTrigger(bl);
SetShortTrigger(bs);



Set-up of Order Default:



Set-up of Evaluator Settings:



[11] Ne s'applique qu'à des trades automatiques (AutoOrder doit être activé) avec des ordres d'entrée AutoOrder paramétrés sur "Limit = Limit" et "Stop = Stop" (Dans Ordre par défaut) et Signal Entrée mis sur "confirmation prix barre suivante" ou "prix limite barre suivante" (Dans Paramètres Evaluator).

[12] Si Paramètres Evaluator est mis sur "prix limite barre suivante" et qu'il n'y a pas de fonction comme like SetLongTrigger ou SetShortTrigger dans un programme, le prix limite sera mis à la clôture de la période qui génère le signal. Si plusieurs indicateurs font appel à cette routine, le prix le plus restrictif est considéré.

Exemple 2:

- Ci-dessous, nous générons des Signaux basés sur le croisement de deux moyennes mobiles. Nous essayons d'entrer à des prix plus favorables en utilisant un ordre **conditional stop**[13] placé à 5 ticks du prix de clôture de la barre en cours. Notez ci-dessous les conditions de paramétrage et la description de ce qui se passe en bleu:
  - Quand un signal est créé la plateforme place un ordre stop au début de la prochaine barre au niveau de prix donné par SetLongTrigger ou SetShortTrigger.
  - Si le stop est touché avant la clôture de la prochaine barre, une position est ouverte, sinon, le stop est annulé.

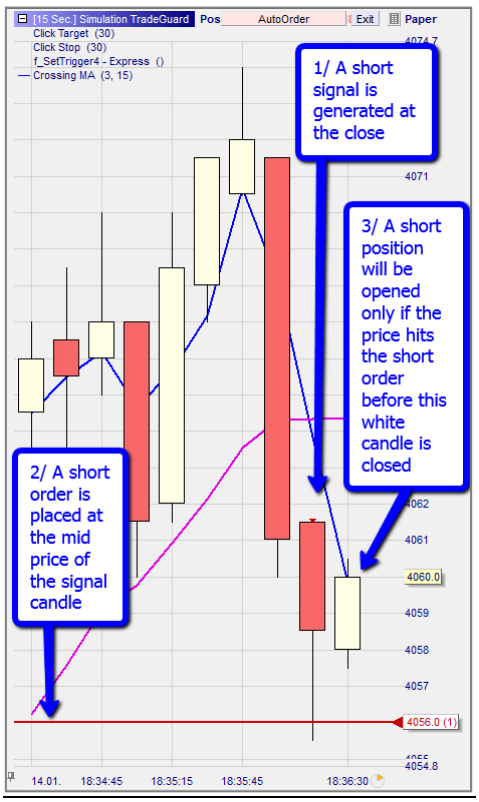
Express Stop f_SetTrigger

Vars

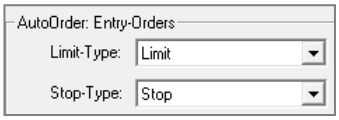
numeric
bl, bs;

Calculation

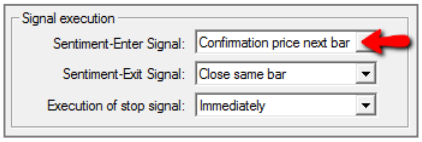
bl = c + 5* TickSize();
bs = c - 5*TickSize();
SetLongTrigger(bl);
SetShortTrigger(bs);



Set-up of Order Default:



Set-up of Evaluator Settings:



[13] Si Paramètres Evaluator est mis sur "confirmation prix barre suivante" et qu'il n'y a pas de fonctions comme SetLongTrigger ou SetShortTrigger dans un programme, le prix stop sera placé au haut/bas de la période qui génère le signal. Si plusieurs indicateurs font appel à cette routine, le prix le plus restrictif est considéré.

SetStopPrice()

Cette fonction ne peut être utilisée que pour les stops.

Définition:

- Place un ordre stop à un niveau de prix donné.

Format:

- void SetStopPrice (float value)

Exemple:

- Ci-dessous nous plaçons un stop à 2 ticks du plus bas 'bas' ou du plus haut 'haut' sur les 10 dernières bougies:
    - Comme nous sommes entrés short deux stops sont positionnés. Le click stop à 4047 et le stop programmé ci-dessous qui est 2 ticks au-dessus de la ligne des plus hauts 'hauts'.

```
Express Stop f_SetStopPrice

vars

series
ll, hh;

Calculation
ll = Lowest(l, 10);
hh = Highest(h, 10);

If MarketPosition() = 1 then
SetStopPrice (ll - 2*TickSize());
else
if MarketPosition() = -1 then
SetStopPrice (hh + 2*TickSize());
```

SetTargetPrice()

Cette fonction ne peut être utilisée que pour les stops.

Définition:

- Place un ordre target à un niveau de prix donné.

Format:

- void SetTargetPrice (float value)

Exemple:

- Ci-dessous, nous créons un target dynamique dépendant du range entre les plus hauts 'hauts' et les plus bas 'bas' des 5 dernières barres.
  - Juste après l'entrée le range a augmenté, donc le prix target est descendu à 3720.
  - Plus tard le range s'est contracté donc notre prix target reste au même niveau.
  - Ensuite, vu que le range a recommencé à augmenter pendant que le marché descendait notre prix target a baissé également.

```
express Stop f_SetTargetPrice

vars

numeric
TargetL, TargetS, Range;

input
$n(1, 50, 5), $k(0.1, 2.0, 1.0, 0.1, 1);

calculation

Range = Highest(h, $n) - Lowest(l, $n);
if BarsSinceEntry() = 0 then
begin
  TargetL = c + $k*Range;
  TargetS = c - $k*Range;
end
else
begin
  TargetL = max(TargetL, c + $k*Range);
  TargetS = min(TargetS, c - $k*Range);
end

If MarketPosition() = 1 then SetTargetPrice(TargetL);
else
if MarketPosition() = -1 then SetTargetPrice(TargetS);
```

Outils prédéfinis d'interprétation

CrossesAbove() / CrossesBelow()

Définition:

- Fonctions utilisées pour détecter le croisement de deux courbes.
  - CrossesAbove(curve, trigger) = true if (curve[1] <= trigger[1]) and (curve > trigger)
  - CrossesBelow(curve, trigger) = true if (curve[1] >= trigger[1]) and (curve < trigger)

Format:

- bool CrossesAbove (series curve, series trigger)

Exemple:

- Ci-dessous, l'indicateur est une Bollinger Bands (qui a été importée).
- Les fonctions CrossesAbove et CrossesBelow aident à définir l'interprétation:
  - Un signal long est atteint quand la clôture survient au-dessus de la bande supérieure
  - Un signal short est atteint quand la clôture survient en-dessous de la bande inférieure

```
Express f_CrossesAboveThreshold

Vars

Series
upperband(BollingerBands.upperband),
lowerband(BollingerBands.lowerband);

Calculation

Interpretation
begin
  if CrossesAbove(close, upperband) then sentiment
= 100;
  if CrossesBelow(close, lowerband) then sentiment
= 0;
end

plot (upperband, blue, 2);
plot (lowerband, blue, 2);
```

CrossesAboveThreshold() / CrossesBelowThreshold()

Définition:

- Fonctions utilisées pour détecter le franchissement d'un seuil par une courbe.
  - CrossesAboveThreshold(curve, threshold) = true if (curve[1] <= threshold) and (curve > threshold)
  - CrosseBelowThreshold(curve, threshold) = true if (curve[1] >= threshold) and (curve < threshold)

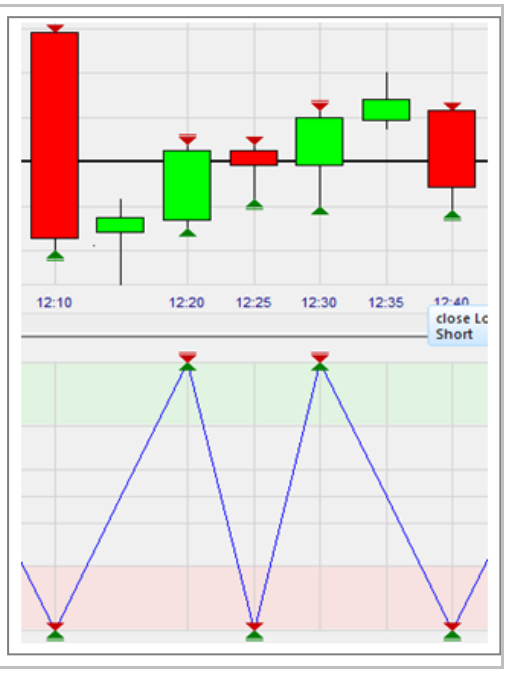
Format:

- bool CrossesAboveThreshold (series curve, float threshold)

Exemple:

- Ci-dessous, l'indicateur est une ligne horizontale (le seuil est à 6930 points).
- Les fonctions CrossesAboveThreshold et CrosseBelow aident à définir l'interprétation:
  - Un signal long est atteint quand la clôture survient au-dessus 6930
  - Un signal short est atteint quand la clôture survient en-dessous 6930

```
Express f_CrossesAboveBelowThreshold

Vars
series;
input $span (0, 200, 10);

Calculation

interpretation
begin
if CrossesAboveThreshold(close, 6930) then sentiment = 100;
if CrossesBelowThreshold(close, 6930) then sentiment = 0;
end

plotline(6930, black, 2);
```



Triggerline()

Définition:

- Schéma d'interprétation pour des indicateurs basés sur le croisement d'une courbe par une autre courbe.
    - Par exemple le prix de clôture croisant une moyenne mobile.

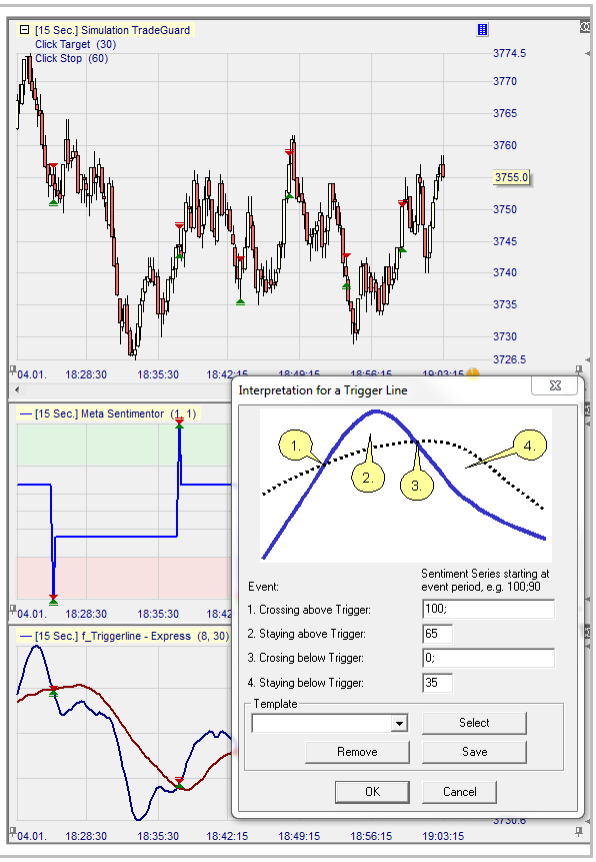
Format:

- TriggerLine (series curve, series trigger)

Exemple:

- Ci-dessous, les indicateurs sont deux moyennes mobiles.
- L'interprétation est définie en utilisant l'interprétation de TriggerLine.
- En cliquant droit sur l'indicateur, et en sélectionnant Editer Interprétation on aperçoit une fenêtre dans laquelle l'interprétation peut être modifiée.
    - Ici nous achetons lorsque la MM rapide croise au-dessus de la MM lente (= 100) et nous vendons lorsque la MM rapide croise en-dessous de la MM lente (= 0).

```
Express f_Triggerline

Vars

Series
FastMA, SlowMA;

Input
$spanFMA(1, 200, 8), $spanSMA(1, 200, 30);

Calculation

If IsFirstBar() then
begin
  MovingAverage(close, SlowMA, $spanSMA);
  MovingAverage(close, FastMA, $spanFMA);
end

Interpretation triggerline(FastMA, SlowMA);

plot (FastMA, blue, 2);
plot (SlowMA, red, 2);
```



Swing()

Définition:

- Schéma d'interprétation basé sur les changements de croissance/décroissance d'une courbe.
  - Par exemple les changements de croissance/décroissance d'indicateurs momentum ou d'autres oscillateurs.

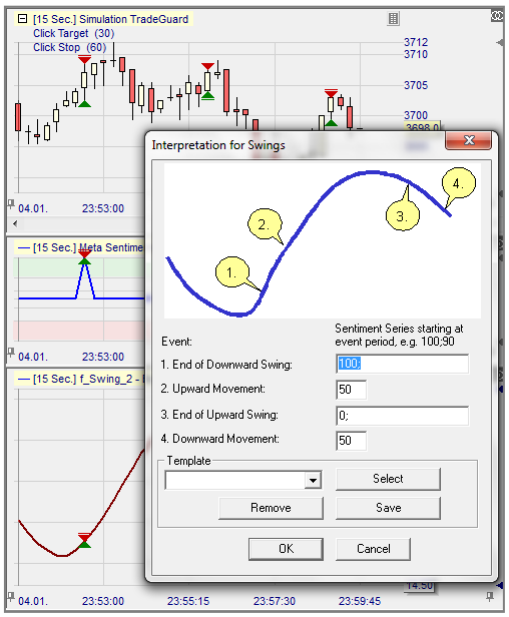
Format:

- void Swing (series series, input spanLeft, input spanRight)

Exemple:

- L'indicateur ci-dessous est une MM d'un RSI.
- L'interprétation est définie en utilisant l'interprétation des swings.
- En cliquant droit sur l'indicateur, et en sélectionnant Editer Interprétation on aperçoit une fenêtre dans laquelle l'interprétation peut être modifiée.
  - Ici nous achetons au début d'une nouvelle croissance (= 100).
    - Une nouvelle croissance est constatée après une décroissance dès que deux nouveaux hauts consécutifs sont constatés (ce paramètre peut être ajusté).
  - Ici nous vendons au début d'une nouvelle décroissance (= 0).
    - Une nouvelle décroissance est constatée après une croissance dès que deux nouveaux bas consécutifs sont constatés (ce paramètre peut être ajusté).

```
Express f_Swing

Vars

Series
myRSI, smoothedRSI;

Input
$spanLEFT(1,100,2), $spanRIGHT(1,100,2),
$RSIspan(1, 100, 14), $MAspan(1, 200, 10);

Calculation

If IsFirstBar() then
begin
  RSI(close, myRSI, $RSIspan);
  MovingAverage(myRSI, smoothedRSI, $MAspan);
end

Interpretation Swing(smoothedRSI, $spanLEFT,
$spanRIGHT);

plot(smoothedRSI, red, 2);
```



Bands()

Définition:

- Schéma d'interprétation pour des indicateurs basés sur le croisement d'une courbe au travers deux autres courbes formant une bande.
    - Par exemple le prix de clôture croisant ce type de bandes: Bollinger Bands, Donchian-Channel, Price-Channel, Commodity-Channel, ….

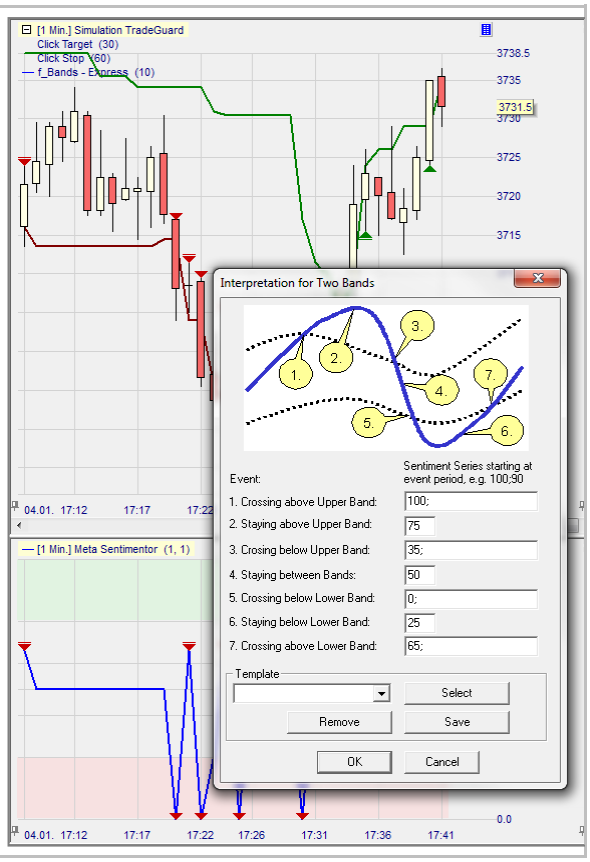
Format:

- void Bands (series series, series lower, series upper)

Exemple:

- Ci-dessous, l'indicateur est un canal basé sur les plus hauts 'hauts' et les plus bas 'bas' des 10 dernières barres.
- L'interprétation est définie en utilisant l'interprétation de bandes.
- En cliquant droit sur l'indicateur, et en sélectionnant Editer Interprétation on aperçoit une fenêtre dans laquelle l'interprétation peut être modifiée.
    - Ici, nous achetons lorsque la clôture passe au-dessus de la courbe supérieure (= 100) et nous vendons lorsque la clôture passe en-dessous de la courbe inférieure (= 0).

```
Express f_Bands

Vars

series
upper, lower, upper1, lower1;

input
$span (0, 200, 10);

Calculation

upper1 = Highest(h, $span);
lower1 = Lowest (l, $span);

upper = upper1[1];
lower = lower1[1];

interpretation Bands (close, lower, upper);

plot (upper, green, 2);
plot (lower, red, 2);
```



TwoThresholds()

Définition:

- Schéma d'interprétation pour des indicateurs basés sur le croisement d'une courbe au travers deux lignes horizontales.
    - Par exemple un RSI, une stochastique ou d'autres oscillateurs croisant deux lignes supérieures et inférieures.

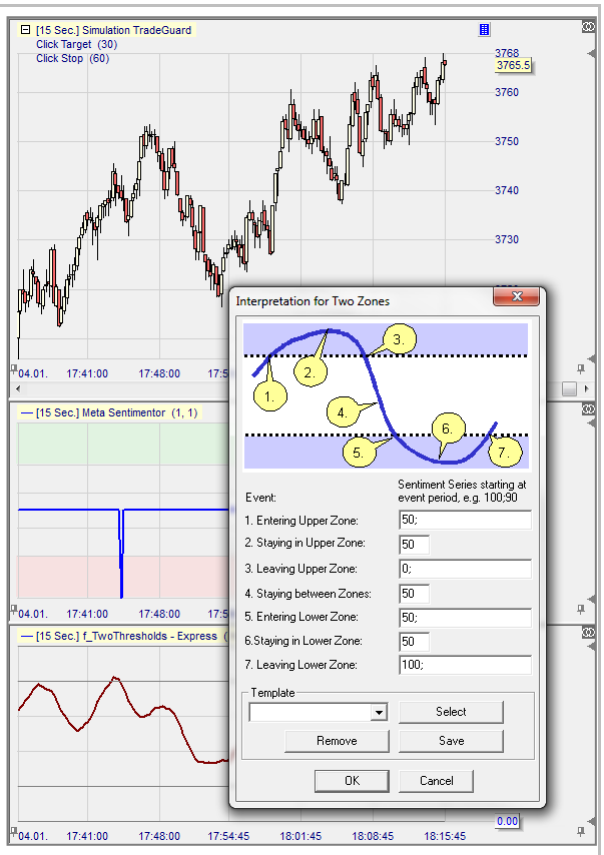
Format:

- void TwoThresholds (series series, input upThreshold, input downThreshold)

Exemple:

- Ci-dessous, l'indicateur est une MM sur 10 barres d'un RSI(14).
- L'interprétation est définie en utilisant l'interprétation de deux seuils.
- En cliquant droit sur l'indicateur, et en sélectionnant Editer Interprétation on aperçoit une fenêtre dans laquelle l'interprétation peut être modifiée.
    - Un signal long est donné dans le cas #7
    - Un signal short est donné dans le cas #3

```
Express f_TwoThresholds

Vars

Series
myRSI, smoothedRSI;

Input
$RSIspan(1, 100, 14), $MAspan(1, 200, 10),
$upperzone(51, 99, 80), $lowerzone(1, 49, 20);

Calculation
If IsFirstBar() then
begin
  RSI(close, myRSI, $RSIspan);
  MovingAverage(myRSI, smoothedRSI, $MAspan);
end

Interpretation TwoThresholds(smoothedRSI,
$upperzone, $lowerzone);

plot(smoothedRSI, red, 2);
plotline(100, grey, 1);
plotline(80, grey, 1);
plotline(20, grey, 1);
plotline(0, grey, 1);
```



Autres conseils

Utilisation d'éditeurs de texte externes

Un moyen performant, rapide et sûr d'améliorer l'expérience de programmation dans NanoTrader est de recourir à des éditeurs de texte externes comme Notepad+++, UltraEdit etc. Certains d'entre eux sont entièrement gratuits et offrent des fonctionnalités très utiles telles que :

- La sauvegarde automatique
- La recherche et le remplacement de chaînes de code
- Déplacer les blocs de code d'une tabulation vers la droite
- Les alignements verticaux
- Travailler sur plusieurs scripts simultanément

Pour utiliser un éditeur de texte externe, il suffit de copier/coller le code de l'éditeur de texte externe dans l'éditeur Express de NanoTrader et vice versa.

Exemple:

Ci-dessous un morceau de code NanoTrader Express qui a été copié dans l'éditeur de texte gratuit Notepad++.