Sistema di scripting di FiutoPRO per le strategie in opzioni

1. Introduzione

Il sistema di scripting di FiutoPRO (FPSS) è stato progettato con l'intento di dare all'utente di FiutoPRO la maggiore libertà di personalizzazione possibile mantenendo al tempo stesso il più basso livello di difficoltà possibile. FPSS è un interprete, cioè un software in grado di ricevere un testo o una serie di testi in ingresso, e di convertirli in azioni su FiutoPRO. Praticamente qualsiasi aspetto di una strategia in opzioni di FiutoPRO può essere manipolato, modificato ed esteso tramite FPSS. I testi che FPSS è in grado di comprendere devono essere scritti tramite un linguaggio formale, cioè un linguaggio di programmazione. FPSS è in grado di interpretare due diversi linguaggi di programmazione, Basic e Pascal. Il linguaggio Basic interpretato da FPSS è molto simile a Visual Basic ® di Microsoft ®, mentre il Pascal è simile a Delphi ® di Embarcadero ®.
Entrambi i linguaggi utilizzabili in FPSS sono ampiamente documentati sulla rete, con innumerevoli siti dedicati e tutorial guidati per l'apprendimento dei concetti fondamentali e della struttura complessiva del linguaggio, pertanto in questo manuale questi aspetti verranno illustrati in modo breve e conciso, quanto più chiaramente possibile, rimandando alla consultazione sulla rete per la ricerca di informazioni più approfondite.
In FiutoPRO, ogni strategia ha il suo interprete FPSS, perciò ogni strategia può essere programmata in modo indipendente dalle altre.

1. Progetti e script

Il sistema di scripting di FiutoPRO è in grado di interpretare il codice di un progetto. Un progetto di FPSS è un insieme di uno o più files sorgenti (scritti indifferentemente in linguaggio Basic e/o Pascal). Tra i file che compongono un progetto di FPSS, uno deve essere designato come file sorgente principale, cioè come file dal quale inizierà l'interpretazione del codice. Inizialmente FPSS chiama questo file "MainUnit". I files sorgenti che compongono un progetto possono esportare classi, funzioni, procedure e variabili in modo che altri files del progetto vi possano accedere.

2. Sintassi e struttura del linguaggio Basic

3.1. Introduzione

FPSS è in grado di eseguire script scritti in linguaggio Basic. La sintassi del linguaggio Basic supporta:

- dichiarazioni `sub .. end` e `function .. end`
- direttive `byref` e `dim`
- istruzioni `if .. then .. else .. end`
- istruzioni `for .. to .. step .. next`
- istruzioni `do .. while .. loop` e `do .. loop .. while`
- istruzioni `do .. until .. loop` e `do .. loop .. until`
- operatori `^, *, /, and, +, -, or, <>, >=, <=, =, >, <, div, mod, xor, shl, shr`
- blocchi `try .. except` e `try .. finally`
- blocchi `try .. catch .. end try` e `try .. finally .. end try`
- istruzioni `select case .. end select`
- array ed array inizializzati
- istruzione `exit`
- accesso alle proprietà ed ai metodi di tutti gli oggetti che compongono la strategia di FiutoPRO.

3.2. Struttura degli script

La struttura degli script è costituita da due blocchi principali:
a) dichiarazione di funzioni e procedure
b) blocco principale
Entrambi questi blocchi sono opzionali, ma in uno script almeno uno dei due dovrebbe sempre essere presente. Ecco alcuni esempi.

SCRIPT 1:

```
SUB DoSomething
     CallSomething
END SUB
CallSomethingElse
```

SCRIPT 2:

```
CallSomethingElse
```

SCRIPT 3:

```
FUNCTION MyFunction
     MyFunction = "Ok!"
END FUNCTION
```

E' anche possibile scrivere più istruzioni su una singola riga, separandole tra loro usando il carattere ":".

3.3. Identificatori

I nomi degli identificatori negli script (nomi delle variabili, delle funzioni, delle procedure, ecc.) devono seguire le più comuni regole: devono iniziare con un carattere alfabetico (a..z o A..Z), oppure '_', seguito da un qualsiasi numero di caratteri alfanumerici o '_'. I nomi

degli identificatori non possono contenere altri caratteri o spazi. Ecco alcuni esempi di nomi di identificatori validi:

```
VarName
_Some
V1A2
_____Some____
```

Ecco invece dei nomi di identificatori non validi:

```
2Var
My Name
Some-more
This,is,not,valid
```

3.4. L'istruzione di assegnamento

Le istruzioni di assegnamento (assegnare un valore o il risultato di una espressione ad una variabile o una proprietà di un oggetto) sono realizzate usando il simbolo "=". Alcuni esempi:

```
MyVar = 2
Button.Caption = "This " + "is ok."
```

3.5. L'istruzione new

FPSS supporta l'istruzione "new" per creare nuovi oggetti. L'istruzione new specifica la classe di appartenenza del nuovo oggetto che si sta creando. Ogni oggetto creato all'interno di un progetto di FPSS deve essere anche "distrutto" prima della conclusione dell'esecuzione dello script stesso, usando la funzione `Free`. Ecco alcuni esempi:

```
MyLabel = new TLabel(Form1)
MyFont = new TFont
MyLabel.Free
MyFont.Free
```

Se previsto, è possibile anche fornire dei parametri durante la costruzione di nuovi oggetti tramite l'istruzione new, come nell'esempio precedente dove il nuovo oggetto TLabel viene creato usando come parametro l'oggetto Form1.

3.6. Stringhe di testo

Le stringhe (sequenze di caratteri) sono dichiarate in basic usando il carettere ("). Alcuni esempi:

```
A = "This is a text"
Str = "Text "+"concat"
```

3.7. Commenti

FPSS supporta l'inserimento di commenti all'interno degli script. E' possibile utilizzare il carattere ' oppure la direttiva `REM`. I commenti terminano sempre alla fine della riga. Esempi:

```
' This is a comment before ShowMessage
ShowMessage("Ok")
REM This is another comment
ShowMessage("More ok!")
' And this is a comment
' with two lines
ShowMessage("End of okays")
```

3.8. Variabili

Negli script non è necessario dichiarare il tipo delle variabili, che pertanto si possono dichiarare usando solo la direttiva `DIM` ed il nome. Ecco alcuni esempi:

SCRIPT 1:

```
SUB Msg
     DIM S
     S = "Hello world!"
     ShowMessage(S)
END SUB
```

SCRIPT 2:

```
DIM A
A = 0
A = A+1
ShowMessage(A)
```

Le variabili globali possono essere dichiarate anche come private o pubbliche. Una variabile globale è una variabile dichiarata nello script non all'interno di funzioni o procedure. Una variabile globale dichiarata come privata è visibile soltanto all'interno dello script dove è dichiarata. Una variabile globale dichiarata come pubblica è visibile all'interno di tutti gli script che compongono un progetto. Esempi:

SCRIPT 2:

```
PRIVATE A
PUBLIC B
B = 0
A = B + 1
ShowMessage(A)
```

Le variabili dichiarate tramite la direttiva `DIM` sono considerate come pubbliche. Le variabili possono anche essere inizializzate in fase di dichiarazione, usando la sintassi degli esempi che seguono:

```
DIM A = "Hello world"
DIM B As Integer = 5
```

3.9. Indici

Stringhe ed array possono essere indicizzati usando i caratteri "[" e "]". Per esempio, se Str è una variabile che contiene una stringa di testo, l'espressione Str[3] restituisce il terzo carattere della stringa Str, mentre Str[I + 1] restituisce il carattere immediatamente successivo a quello indicizzato da I. Altri esempi:

```
MyChar = MyStr[2]
MyStr[1] = "A"
MyArray[1,2] = 1530
Lines.Strings[2] = "Some text"
```

3.10. Arrays

FPSS supporta la costruzione di array sia ad una singola dimensione che multi-dimensionali. Per costruire un array è necessario utilizzare i caratteri "[" e "]". La costruzione di array multi-dimensionali richiede di utilizzare la ricorsione. Gli array possono essere utilizzati tramite indici. Se l'array è multi-dimensionale, i diversi indici (uno per ogni dimensione) devono essere separati tramite il carattere ",". Gli array sono sempre indicizzati con base 0, cioè l'indice 0 permette di accedere al primo elemento dell'array. Alcuni esempi:

```
NewArray = [ 2,4,6,8 ]
Num = NewArray[1] //Num receives "4"
MultiArray = [ ["green","red","blue"] , ["apple","orange","lemon"] ]
Str = MultiArray[0,2] //Str receives 'blue'
MultiArray[1,1] = "new orange"
```

3.11. L'istruzione If

Esistono due sintassi per l'istruzione if: `if...then..end` e `if...then...else..end if`. Se l'espressione compresa tra `if` e `then` è vera, allora vengono eseguite le istruzioni comprese tra `then` ed `end if` o tra `then` ed `else`. Se l'espressione compresa tra `if` e `then` è falsa ed è stata usata la sintassi con else, allora le istruzioni comprese tra `else` ed `end if` vengono eseguite. Ecco alcuni esempi:

```
IF J <> 0 THEN Result = I/J END IF
IF J = 0 THEN Exit ELSE Result = I/J END IF
IF J <> 0 THEN
     Result = I/J
     Count = Count + 1
ELSE
     Done = True
END IF
```

Se l'istruzione `if` termina su una stessa riga, allora non è necessario chiudere l'istruzione con `end if`:

```
IF J <> 0 THEN Result = I/J
IF J = 0 THEN Exit ELSE Result = I/J
```

3.12. L'istruzione While

L'istruzione while è usata per ripetere un blocco di istruzioni, finché una condizione di controllo (espressione) viene valutata come vera. La condizione di controllo viene valuta prima di iniziare ad eseguire il blocco di istruzioni. Di conseguenza, se la condizione di controllo viene valutata come falsa alla prima iterazione, il blocco di istruzioni non viene mai eseguito. L'istruzione while esegue continuamente il blocco di istruzioni, verificando l'espressione della condizione di controllo prima di ogni iterazione. Finchè l'espressione

della condizione di controllo viene valutata come vera, l'esecuzione continua. Alcuni esempi:

```
WHILE (Data[I] <> X) I = I + 1 END WHILE
WHILE (I > 0)
      IF Odd(I) THEN Z = Z * X END IF
      X = Sqr(X)
END WHILE
WHILE (not Eof(InputFile))
      Readln(InputFile, Line)
      Process(Line)
END WHILE
```

3.13. Le istruzioni di ciclo

FPSS supporta le istruzioni di ciclo. Le sintassi riconosciute sono le seguenti:

```
DO WHILE expr statements LOOP
DO UNTIL expr statements LOOP
DO statements LOOP WHILE expr
DO statement LOOP UNTIL expr
```

Le istruzioni (statements) saranno eseguite fintanto che (`WHILE`) l'espressione (expr) è vera, oppure fino a quando (`UNTIL`) l'espressione (expr) non diventa vera. Se l'espressione è scritta prima delle istruzioni, la condizione di controllo viene valutata prima dell'esecuzione delle istruzioni, altrimenti la condizione di controllo viene valutata dopo l'iterazione del ciclo. Esempi:

```
DO
      K = I mod J
      I = J
      J = K
LOOP UNTIL J = 0
DO UNTIL I >= 0
      Write("Enter a value (0..9): ")
      Readln(I)
LOOP
DO
      K = I mod J
      I = J
      J = K
LOOP WHILE J <> 0
DO WHILE I < 0
      Write("Enter a value (0..9): ")
      Readln(I)
LOOP
```

3.14. L'istruzione For

FPSS supporta l'istruzione for con la seguente sintassi:

```
FOR counter = initialValue TO finalValue STEP stepValue
      statements
NEXT
```

L'istruzione for imposta il contatore (counter) al valore iniziale (initialValue), ripete l'esecuzione delle istruzioni (statements) fino all'istruzione `NEXT`, ed incrementa il valore del contatore (counter) di stepValue, finché il contatore non raggiunge il valore finale (finalValue). La sezione Step è opzionale: se omessa stepValue è paria 1. Esempi:

SCRIPT 1:
```
FOR c = 1 TO 10 STEP 2
      a = a + c
NEXT
```

SCRIPT 2:
```
FOR I = a TO b
      j = i ^ 2
      sum = sum + j
NEXT
```

3.15. L'istruzione Select Case
FPSS supporta le istruzioni support select con la seguente sintassi:

```
SELECT CASE selectorExpression
      CASE caseexpr1
            statement1
      …
      CASE caseexprn
            statementn
      CASE ELSE
            elsestatement
END SELECT
```

Se l'espressione di selezione (selectorExpression) è uguale ad una delle espressioni di controllo (caseexprn), il rispettivo blocco di istruzioni (statements) viene eseguito, altrimenti, il blocco di istruzioni elsestatement viene eseguito. La parte else dell'istruzione può essere omessa. Esempi:

```
SELECT CASE uppercase(Fruit)
      CASE "lime" ShowMessage("green")
      CASE "orange"
            ShowMessage("orange")
      CASE "apple" ShowMessage("red")
      CASE ELSE
            ShowMessage("black")
END SELECT
```

3.16. Dichiarazione delle funzioni e delle procedure
Una funzione (`function`) è un blocco di istruzioni che restituisce un risultato. Una procedura (`sub`) è un blocco di istruzioni che non restituisce alcun risultato. Per restituire il risultato in una funziona, è necessario assegnare il valore alla variabile implicita individuata dal nome stesso della funzione, oppure usare l'istruzione `Return`. Sia le funzioni che le procedure possono accettare dei valori in ingresso, chiamati parametri. I parametri vengono normalmente inviati alle funzioni ed alle procedure come valori, mentre per poterli inviare per riferimento, è necessario usare la direttiva `ByRef` nella dichiarazione del parametro della funzione o procedura. Alcuni esempi:

```
SUB HelloWord
     ShowMessage("Hello world!")
END SUB
SUB UpcaseMessage(Msg)
     ShowMessage(Uppercase(Msg))
END SUB
FUNCTION TodayAsString
     TodayAsString = DateToStr(Date)
END FUNCTION
FUNCTION Max(A,B)
     IF A>B THEN
     MAX = A
          ELSE
          MAX = B
     END IF
END FUNCTION
SUB SwapValues(BYREF A, B)
     DIM TEMP
     TEMP = A
     A = B
     B = TEMP
END SUB
```

Funzioni e procedure possono essere dichiarate come private o pubbliche, esattamente come avviene per le variabili, con lo stesso significato.

```
PRIVATE SUB Hello
END SUB
PUBLIC FUNCTION Hello
END FUNCTION
```

Funzioni e procedure, se non specificato, sono considerate come pubbliche. Funzioni e procedure private non sono accessibili da altri script. E' possibile utilizzare l'istruzione `Return` per uscire dalle procedure e dalle funzioni. Esempi:

```
SUB UpcaseMessage(Msg)
     ShowMessage(Uppercase(Msg))
     Return
     'This line will be never reached
     ShowMessage("never displayed")
END SUB
FUNCTION TodayAsString
     Return DateToStr(Date)
END FUNCTION
```

3.17. Inclusione di altri file sorgenti

Nei progetti di FPSS è possibile fare riferimento agli oggetti, funzioni e classi con visibilità pubblica definiti in altri file di script. Il meccanismo di inclusione di altri file sorgenti all'interno prevede l'utilizzo della clausola `USES`. Ecco un esempio:

SCRIPT 1 - Unit1:

```
Public Function Test()
     Test = 1
End Function
```

SCRIPT 2 - Unit2:

```
USES Unit1

DIM t
t = Test()
```

E’ possibile includere più files contemporaneamente, separando i nomi con il carattere virgola (,).

3. Sintassi e struttura del linguaggio Pascal

4.1. Introduzione

FPSS è in grado di eseguire script scritti in linguaggio Pascal. La sintassi del linguaggio Pascal supporta:

- dichiarazioni di `procedure` e `function`
- blocchi di istruzioni delimitati da `begin` e `end`
- istruzioni `if .. then .. else`
- istruzioni `for .. to .. do .. step`
- istruzioni `while .. do`
- istruzioni `repeat .. until`
- operatori `^, *, /, and, +, -, or, <>, >=, <=, =, >, <, div, mod, xor, shl, shr`
- blocchi `try .. except` e `try .. finally`
- istruzioni `case`
- array ed array inizializzati
- istruzione `exit`
- accesso alle proprietà ed ai metodi di tutti gli oggetti che compongono la strategia di FiutoPRO.

4.2. La struttura dello script

La struttura degli script è costituita da due blocchi principali:
a) dichiarazione di funzioni e procedure
b) blocco principale
Entrambi questi blocchi sono opzionali, ma in uno script almeno uno dei due dovrebbe sempre essere presente. Il blocco principale non deve essere necessariamente definito tramite le istruzioni begin ed end, e può essere costituito anche da una sola riga. Ecco alcuni esempi.

SCRIPT 1:

```
procedure DoSomething;
begin
     CallSomething;
end;
begin
     CallSomethingElse;
end;
```

SCRIPT 2:

```
begin
     CallSomethingElse;
end;
```

SCRIPT 3:

```
function MyFunction;
begin
     result := 'Ok!';
end;
```

SCRIPT 4:

```
CallSomethingElse;
```

Le istruzioni devono essere terminate dal carattere ";". I blocchi Begin..end sono permessi per poter raggruppare più istruzioni.

4.3. Identificatori

I nomi degli identificatori negli script (nomi delle variabili, delle funzioni, delle procedure, ecc.) devono seguire le più comuni regole: devono iniziare con un carattere alfabetico (a..z o A..Z), oppure '_', seguito da un qualsiasi numero di caratteri alfanumerici o '_'. I nomi degli identificatori non possono contenere altri caratteri o spazi. Ecco alcuni esempi di nomi di identificatori validi:

```
VarName
_Some
V1A2
_____Some____
```

Ecco invece dei nomi di identificatori non validi:

```
2Var
My Name
Some-more
This,is,not,valid
```

4.4. L'istruzione di assegnamento

Le istruzioni di assegnamento (assegnare un valore o il risultato di una espressione ad una variabile o una proprietà di un oggetto) sono realizzate usando ":=". Alcuni esempi:

```
MyVar := 2;
Button.Caption := 'This ' + 'is ok.';
```

4.5. Le stringhe di caratteri

Le stringhe (sequenze di caratteri) sono dichiarate in pascal usando il carettere ('). Il carattere (") non è usato. E' inoltre possibile utilizzare la notazione #nn per definire un carattere all'interno di una stringa. Non è necessario usare l'operatore '+' per aggiungere caratteri ad una stringa. Alcuni esempi:

```
A := 'This is a text';
Str := 'Text '+'concat';
B := 'String with CR and LF char at the end'#13#10;
C := 'String with '#33#34' characters in the middle';
```

4.6. Commenti

FPSS supporta l'inserimento di commenti all'interno degli script. I commenti vengono individuati dai caratteri //, per un commento che termina alla fine della riga, oppure con le coppie (* e *) o { e } per dei commenti multi-linea. Alcuni esempi:

```
//This is a comment before ShowMessage
ShowMessage('Ok');
(* This is another comment *)
ShowMessage('More ok!');
{ And this is a comment
with two lines }
ShowMessage('End of okays');
```

4.7. Variabili

Negli script non è necessario dichiarare il tipo delle variabili, che pertanto si possono dichiarare usando solo la direttiva `var` ed il nome. Ecco alcuni esempi:

SCRIPT 1:

```
procedure Msg;
var S;
begin
     S := 'Hello world!';
     ShowMessage(S);
end;
```

SCRIPT 2:

```
var A;
begin
     A := 0;
     A := A + 1;
end;
```

SCRIPT 3:

```
var S: string;
begin
     S := 'Hello World!';
     ShowMessage(S);
end;
```

4.8. Indici

Stringhe ed array possono essere indicizzati usando i caratteri "[" e "]". Per esempio, se Str è una variabile che contiene una stringa di testo, l'espressione Str[3] restituisce il terzo carattere della stringa Str, mentre Str[I + 1] restituisce il carattere immediatamente successivo a quello indicizzato da I. Altri esempi:

```
MyChar := MyStr[2];
MyStr[1] := 'A';
MyArray[1, 2] := 1530;
Lines.Strings[2] := 'Some text';
```

4.9. Arrays

FPSS supporta la costruzione di array sia ad una singola dimensione che multi-dimensionali. Per costruire un array è necessario utilizzare i caratteri "[" e "]". La costruzione di array multi-dimensionali richiede di utilizzare la ricorsione. Gli array possono essere utilizzati tramite indici. Se l'array è multi-dimensionale, i diversi indici (uno per ogni dimensione) devono essere separati tramite il carattere ",". Gli array sono sempre

indicizzati con base 0, cioè l'indice 0 permette di accedere al primo elemento dell'array. Alcuni esempi:

```
NewArray := [ 2,4,6,8 ];
Num := NewArray[1]; //Num receives "4"
MultiArray := [['green', 'red', 'blue'], ['apple', 'orange', 'lemon']];
Str := MultiArray[0, 2]; //Str receives 'blue'
MultiArray[1, 1] := 'new orange';
```

4.10. L'istruzione If

Esistono due sintassi per l'istruzione if: `if...then` e `if...then...else`. Se l'espressione compresa tra `if` e `then` è valutata come vera, allora l'istruzione o il blocco di istruzioni che seguono `then` viene eseguita. Se esiste la parte `else` e l'espressione compresa tra `if` e `then` è valutata come falsa, allora l'istruzione o blocco di istruzioni che segue `else` viene eseguita. Esempi:

```
if J <> 0 then Result := I/J;
if J = 0 then Exit else Result := I/J;
if J <> 0 then
begin
     Result := I/J;
     Count := Count + 1;
end
else
     Done := True;
```

ATTENZIONE: l'istruzione immediatamente precedente else non deve mai essere terminata dal carattere ";".

4.11. L'istruzione While

L'istruzione while è usata per ripetere un'istruzione o un blocco di istruzioni fintanto che la condizione di controllo è vera. La condizione di controllo viene valutata prima di eseguire le istruzioni, quindi, se è falsa alla prima iterazione, le istruzioni non vengono mai eseguite. L'istruzione while esegue le istruzioni ripetutamente, verificando la condizione di controllo prima di ogni iterazione. Finché la condizione di controllo è vera l'esecuzione continua. Esempi:

```
while Data[I] <> X do I := I + 1;
while I > 0 do
begin
     if Odd(I) then Z := Z * X;
     I := I div 2;
     X := Sqr(X);
end;
while not Eof(InputFile) do
begin
     Readln(InputFile, Line);
     Process(Line);
end;
```

4.12. L'istruzione Repeat

La sintassi dell'istruzione repeat è:

```
repeat
     statement1;
     ...;
     statementn;
until expression
```

dove `expression` ritorna un valore boolean vero/falso. L'istruzione repeat esegue le istruzioni continuamente, verificando la condizione di controllo dopo ogni iterazione. Quando la condizione di controllo è vera, l'istruzione repeat termina. La sequenza di istruzioni è sempre eseguita almeno una volta perchè la condizione di controllo viene valutata per la prima volta soltanto al termine della prima iterazione. Esempi:

```
repeat
     K := I mod J;
     I := J;
     J := K;
until J = 0;
repeat
     Write('Enter a value (0..9): ');
     Readln(I);
until (I >= 0) and (I <= 9);
```

4.13. L'istruzione For

FPSS supporta l'istruzione for con la seguente sintassi:

```
for counter := initialValue to finalValue do
     statement
```

L'istruzione for imposta la variabile di controllo (counter) al valore iniziale (initialValue), ripete l'esecuzione del blocco di istruzioni (statement) ed incrementa il valore della variabile di controllo fino a quando non raggiunge il valore finale (finalValue). Esempi:

SCRIPT 1:

```
for c := 1 to 10 do
     a := a + c;
```

SCRIPT 2:

```
for i := a to b do
begin
     j := i^2;
     sum := sum + j;
end;
```

4.14. L'istruzione Case

FPSS supporta l'istruzione case con la seguente sintassi:

```
case selectorExpression of
     caseexpr1: statement1;
     ...
     caseexprn: statementn;
     else
          elsestatement;
end
```

Se l'espressione di selezione (selectorExpression) è uguale al risultato di una delle espressioni caseexprn, il rispettivo blocco di istruzioni viene eseguito, altrimenti, viene eseguito il blocco di istruzioni elsestatement. La parte else dell'istruzione case è opzionale. E' possibile utilizzare espressioni che ritornano un qualsiasi tipo di dato come espressione di selezione. Esempio:

```
case uppercase(Fruit) of
     'lime': ShowMessage('green');
     'orange': ShowMessage('orange');
     'apple': ShowMessage('red');
     else
          ShowMessage('black');
end;
```

4.15. Dichiarazione di funzioni e procedure

La dichiarazione delle funzioni e delle procedure segue le regole di base del Pascal/Object Pascal/Delphi, con la differenza che non è necessario dichiarare il tipo di dato utilizzato dai parametri. Nelle funzioni, per restituire il risultato è necessario utilizzare la variabile implicita result. E' inoltre possibile utilizzare parametri per riferimento. Alcuni esempi:

```
procedure HelloWord;
begin
     ShowMessage('Hello world!');
end;
procedure UpcaseMessage(Msg);
begin
     ShowMessage(Uppercase(Msg));
end;
function TodayAsString;
begin
     result := DateToStr(Date);
end;
function Max(A,B);
begin
     if A > B then
          result := A
     else
          result := B;
end;
procedure SwapValues(var A, B);
Var Temp;
begin
     Temp := A;
     A := B;
     B := Temp;
end;
```

4.16. Inclusione di altri file sorgenti

Nei progetti di FPSS è possibile fare riferimento agli oggetti, funzioni e classi con visibilità pubblica definiti in altri file di script. Il meccanismo di inclusione di altri file sorgenti all'interno prevede l'utilizzo della clausola `USES`. Ecco un esempio:

SCRIPT 1 - Unit1:

```
function Test;
begin
     Result := 1;
end;
```

SCRIPT 2 - Unit2:

```
USES Unit1

Var t;
t := Test();
```

E' possibile includere più files contemporaneamente, separando i nomi con il carattere virgola (,).

4. Funzioni e costanti predefinite

5.1. Le funzioni derivate dal Basic

In FPSS sono disponibili alcune funzioni base derivate dai linguaggi Basic più comuni. Queste funzioni sono disponibili sia negli script Basic che negli script Pascal all'interno di FPSS, non solamente negli script Basic. Essendo tutte funzioni standard del linguaggi, sulla rete si trova ampia documentazione con relativi esempi di utilizzo su ognuna di esse.

```
Function Asc(ByVal String As Char) As Integer
```
Restituisce un valore intero che rappresenta il codice ASCII del carattere passato come parametro alla funzione

```
Function Atn(ByVal v As Double) As Double
```
Restituisce, espresso in radianti, l'arcotangente del valore v

```
Function CBool(ByVal v As Variant) As Bool
```
Converte il valore v in un valore boolean vero/falso

```
Function CByte(ByVal v As Variant) As Byte
```
Converte il valore v in BYTE

```
Function CCur(ByVal v As Variant) As Currency
```
Converte il valore v nel tipo monetario Currency

```
Function CDate(ByVal v As Variant) As Date
```
Converte il valore v in una data e ora

```
Function CDbl(ByVal v As Variant) As Double
```
Converte il valore v in un numero decimale a virgola mobile a doppia precisione

```
Function Cint(ByVal v As Variant) As Integer
```
Converte il valore v in un numero intero

```
Function CLng(ByVal v As Variant) As Long
```
Converte il valore v in un numero intero di tipo Long

```
Function CSng(ByVal v As Variant) As Single
```
Converte il valore v in un numero decimale a virgola mobile a singola precisione

```
Function CStr(ByVal v As Variant) As String
```
Converte il valore v in una stringa

```
Function DatePart(ByVal Interval As String,
ByVal DateValue As DateTime) As Integer
```
Restituisce un valore intero contenente la parte specificata di un valore DateTime.

```
Function DateSerial(ByVal Year As Integer, ByVal Month As Integer, ByVal
Day As Integer) As DateTime
```
Restituisce un valore DateTime che rappresenta l'anno, il mese e il giorno specificati, con l'ora impostata su mezzanotte (00.00.00).

```
Function DateValue(ByVal StringDate As String) As DateTime
```
Converte il testo in ingresso in un valore DateTime, con l'ora impostata su mezzanotte (00.00.00).

```
Function Day(ByVal DateValue As DateTime) As Integer
```
Restituisce un valore Integer compreso tra 1 e 31 che rappresenta il giorno del mese

```
Function Fix(ByVal Number As Variant) As Variant
```
Restituisce la parte intera di un numero

```
Function FormatCurrency(ByVal Expression As Object) As String
```
Ritorna una stringa formattata come un valore monetario usando le impostazioni internazionali del pannello di controllo

```
Function FormatDateTime(ByVal Expression As DateTime) As String
```
Ritorna una stringa che rappresenta il valore di data e ora

```
Function FormatNumber(ByVal Expression As Object) As String
```
Converte l'oggetto in una stringa che ne rappresenta il valore numerico

```
Function Hex(ByVal Number As Variant) As String
```
Restituisce una stringa che rappresenta il valore esadecimale di un numero

```
Function Hour(ByVal TimeValue As DateTime) As Integer
```
Ritorna un valore intero compreso tra 0 e 23 che rappresenta l'ora del giorno

```
Function InputBox(ByVal Prompt As String, ByVal Title As String, ByVal
DefaultResponse As String) As String
```
Mostra una finestra di dialogo, attende che l'utente inserisca il testo o clicchi su un pulsante, e ritorna una stringa contenente il testo inserito dall'utente. Il parametro prompt rappresenta il testo che verrà visualizzato all'interno della finestra di dialogo. Il parametro title rappresenta il titolo della finestra di dialogo. Il parametro DefaultResponse rappresenta il valore che verrà restituito se l'utente annullerà la finestra di dialogo.

```
Function InStr(ByVal String1 As String, ByVal String2 As String) As
Integer
```
Restituisce un valore integer che specifica la posizione di inizio della prima occorrenza di una stringa in un'altra

```
Function Int(ByVal Number As Variant) As Variant
```
Restituisce la parte intera di un numero

```
Function IsArray(ByVal VarName As Object) As Boolean
```
Restituisce un valore Boolean vero/falso che indica se la variabile fa riferimento a una matrice

```
Function IsDate(ByVal Expression As Object) As Boolean
```
Restituisce un valore Boolean vero/falso che indica se l'espressione rappresenta un valore di data valido

```
Function IsEmpty(ByVal Expression As Object) As Boolean
```
Restituisce un valore Boolean vero/falso che indica se l'espressione rappresenta una variabile non inizializzata

```
Function IsNull(ByVal Expression As Object) As Boolean
```
Restituisce un valore Boolean vero/falso che indica se l'espressione rappresenta una variabile che non contiene dati validi

```
Function IsNumeric(ByVal Expression As Object) As Boolean
```
Restituisce un valore Boolean vero/falso in cui è specificato se il valore di un'espressione può corrispondere a un numero

```
Function LBound(ByVal Array As System.Array,
Optional ByVal Rank As Integer = 1) As Integer
```
Restituisce l'indice minimo disponibile per la dimensione specificata di una matrice

```
Function LCase(ByVal Value As String) As String
```
Restituisce una stringa convertita in minuscolo

```
Function Left(ByVal str As String, ByVal Length As Integer) As String
```
Restituisce una stringa contenente un numero di caratteri specificato a partire dalla sinistra della stringa

```
Function Len(ByVal Expression As Object) As Integer
```
Restituisce un valore integer corrispondente al numero di caratteri di una stringa oppure al numero nominale di byte necessari per memorizzare una variabile

```
Function Log(ByVal v As Double) As Double
```
Restituisce un valore Double che contiene il logaritmo naturale di un numero specificato

```
Function LTrim(ByVal str As String) As String
```
Restituisce una stringa contenente una copia di una determinata stringa senza spazi iniziali

```
Function Mid(ByVal str As String, ByVal Start As Integer, Optional ByVal
Length As Integer) As String
```
Restituisce una stringa contenente il numero di caratteri specificato da una stringa

```
Function Minute(ByVal TimeValue As DateTime) As Integer
```
Restituisce un valore Integer compreso tra 0 e 59 che rappresenta il minuto dell'ora

```
Function Month(ByVal DateValue As DateTime) As Integer
```
Restituisce un valore Integer compreso tra 1 e 12 che rappresenta il mese dell'anno

```
Function MonthName(ByVal Month As Integer, Optional ByVal Abbreviate As
Boolean = False) As String
```
Restituisce un valore String contenente il nome del mese specificato

```
Function MsgBox(ByVal Prompt As Object, Optional ByVal Buttons As
MsgBoxStyle = MsgBoxStyle.OKOnly, Optional ByVal Title As Object =
Nothing) As MsgBoxResult
```
Visualizza un messaggio in una finestra di dialogo e attende che l'utente scelga un pulsante, quindi restituisce un valore integer che indica il pulsante scelto dall'utente

```
Function Replace(ByVal Expression As String, ByVal Find As String, ByVal
Replacement As String, Optional ByVal Start As Integer = 1, Optional
ByVal Count As Integer = -1, Optional ByVal Compare As CompareMethod =
CompareMethod.Binary) As String
```

Restituisce una stringa nella quale la sottostringa specificata è stata sostituita con un'altra sottostringa per il numero di volte indicato

```
Function Right(ByVal str As String, ByVal Length As Integer) As String
```

Restituisce una stringa contenente un numero di caratteri specificato a partire dalla destra della stringa

```
Function Rnd[(Number)] As Single
```

Restituisce un numero casuale di tipo Single

```
Function RTrim(ByVal str As String) As String
```

Restituisce una stringa contenente una copia di una determinata stringa senza spazi finali

```
Function Second(ByVal TimeValue As DateTime) As Integer
```

Restituisce un valore Integer compreso tra 0 e 59 che rappresenta il secondo del minuto

```
Function Sgn(ByVal Number As Variant) As Integer
```

Restituisce un valore Integer che indica il segno di un numero

```
Function Space(ByVal Number As Integer) As String
```

Restituisce una stringa composta dal numero di spazi specificato

```
Function StrComp(ByVal String1 As String, ByVal String2 As String,
Optional ByVal Compare As CompareMethod) As Integer
```

Restituisce -1, 0 o 1 in base al risultato di un confronto tra stringhe

```
Function String(ByVal Number As Integer, ByVal Character As Char) As
String
```

Ritorna una stringa costituita da Number caratteri Character

```
Function TimeSerial(ByVal Hour As Integer, ByVal Minute As Integer,
ByVal Second As Integer) As DateTime
```

Restituisce un valore Date che rappresenta l'ora, i minuti e i secondi specificati, con le informazioni relative alla data impostate sull'1 gennaio dell'anno 1

```
Function TimeValue(ByVal StringTime As String) As DateTime
```

Restituisce un valore Date contenente le informazioni relative all'ora rappresentate da una stringa, con la data impostata sull'1 gennaio dell'anno 1

```
Function UBound(ByVal Array As System.Array, Optional ByVal Rank As
Integer = 1) As Integer
```

Restituisce l'indice massimo disponibile per la dimensione specificata di una matrice

```
Function UCase(ByVal Value As String) As String
```

Restituisce una stringa contenente la stringa specificata convertita in lettere maiuscole

```
Function Weekday(ByVal DateValue As DateTime) As Integer
```

Restituisce un valore Integer contenente un numero che rappresenta il giorno della settimana

```
Function WeekdayName(ByVal Weekday As Integer) As String
```
Restituisce un valore String contenente il nome del giorno specificato

```
Function Year(ByVal DateValue As DateTime) As Integer
```
Restituisce un valore Integer compreso tra 1 e 9999 che rappresenta l'anno

5.2. Le funzioni derivate dal Pascal
In FPSS è stato integrato un sotto-insieme delle librerie VCL di Delphi (versione XE2). In particolare, sono state integrate le classi, le funzioni e procedure, le costanti, i tipi, ed infine le variabili definite nei seguenti file di Delphi XE2:
SysUtils.pas
Classes.pas
Graphics.pas
Controls.pas
Forms.pas
Dialogs.pas
StdCtrls.pas
Windows.pas
System.pas
Menus.pas
Buttons.pas

Una documentazione esaustiva delle librerie VCL di Delphi XE2 si può trovare a partire dal seguente indirizzo:
http://docwiki.embarcadero.com/RADStudio/XE2/en/Main_Page

5.3. Altre costanti e funzioni
Oltre alle funzioni fornite con i linguaggi Basic e Pascal, in FPSS sono state integrate altre funzioni e costanti specifiche della piattaforma FiutoPRO.
5.3.1. L'enumerazione THighlightStatus
Questa enumerazione definisce gli stati di evidenziazione relative al ricevimento di prezzi aggiornati in tempo reale. Gli stati definiti sono:

```
HIGHLIGHT_NONE
```
Nessuna evidenziazione

```
HIGHLIGHT_UP
```
Evidenziazione per valore in aumento

```
HIGHLIGHT_NEUTRAL
```
Evidenziazione per valore costante

```
HIGHLIGHT_DOWN
```
Evidenziazione per valore in discesa

5.3.2. Funzioni globali

```
Function CalculateFutureConstant(ByVal reference As Double, ByVal last
As Double, ByVal expiryDate As TDateTime, ByVal analysisDate As
TDateTime) As Double
```

Calcola la costante esponenziale di un future

reference	Prezzo di riferimento del sottostante (indice o azione)
last	Prezzo Last del future
expiryDate	Data di scadenza del future
analysisDate	Data di analisi

```
Function CalculateTheoreticalDividends(ByVal reference As Double, ByVal
expiryDate As TDateTime, ByVal analysisDate As TDateTime, ByVal futureK
As Double, ByVal  riskFreeRate As Double) As Double
```

Calcola il valore teorico dei dividendi associati alla coppia sottostante (indice o azione) e future

reference	Prezzo di riferimento del sottostante (indice o azione)
expiryDate	Data di scadenza del future
analysisDate	Data di analisi
futureK	Costante esponenziale del future
riskFreeRate	Tasso di interesse privo di rischio (per 1% usare 0.01)

```
Function CalculateTheoreticalFuture(ByVal reference As Double, ByVal
expiryDate As TDateTime, ByVal analysisDate As TDateTime, ByVal futureK
As Double, ByVal tick As Double, Optional ByVal defValue As Double = 0)
As Double
```

Calcola il valore teorico del future

reference	Prezzo di riferimento del sottostante (indice o azione)
expiryDate	Data di scadenza del future
analysisDate	Data di analisi
futureK	Costante esponenziale del future
tick	Tick minimo di scostamento del prezzo del future
defValue	Valore di default in caso di errore (parametro opzionale)

```
Function CalculateTheoreticalIndex(ByVal last As Double, ByVal
expiryDate As TDateTime, ByVal analysisDate As TDateTime, ByVal futureK
As Double, ByVal tick As Double, Optional ByVal defValue As Double = 0)
As Double
```

Calcola il valore teorico del sottostante (indice o azione)

last	Prezzo Last del future
expiryDate	Data di scadenza del future
analysisDate	Data di analisi
futureK	Costante esponenziale del future
tick	Tick minimo scostamento del prezzo del sottostante (indice o azione)
defValue	Valore di default in caso di errore (parametro opzionale)

```
Function CommonDataPath() As String
```

Restituisce il percorso dove si trovano i dati delle applicazioni dell'utente di Windows

```
Function CommonReadOnlyDataPath() As String
```

Restituisce il percorso dove si trovano i dati non modificabili delle applicazioni di Windows

```
Function CorrectIsin(ByVal inputIsin As String) As String
```

Restituisce il codice isin modificato in modo da poterlo utilizzare come parte del nome di una tabella di database

inputIsin Codice isin da modificare

```
Function DateTimeToString(ByVal value As TDateTime, Optional ByVal
format As String = 'yyyy-mm-dd hh.nn.ss') As String
```

Converte una variabile di tipo data/ora in una stringa di testo

value Valore data/ora da convertire

format Formato della stringa di output (parametro opzionale)

Maggiori informazioni sul formato della stringa di output possono essere reperite a questo indirizzo:

http://docwiki.embarcadero.com/Libraries/XE2/en/System.SysUtils.FormatDateTime

```
Function EscapeHTTPRequest(ByVal text As String) As String
```

Sostituisce nel testo in ingresso tutti i caratteri speciali con la relativa codifica HTTP

text Testo in ingresso

```
Function EscapeSQL(ByVal text As String) As String
```

Sostituisce nel testo in ingresso tutti i caratteri speciali con la relativa codifica SQL

text Testo in ingresso

```
Function Exec(ByVal command As String, Optional ByVal waitCompletion As
Boolean = false) As Boolean
```

Esegue un'applicazione con relativi parametri

command Applicazione (full path) e parametri

waitCompletion Indica se attendere il completamento dell'esecuzione

```
Function ExecHidden(ByVal command As String, Optional ByVal
waitCompletion As Boolean = false) As Boolean
```

Esegue un'applicazione con relativi parametri senza mostrare alcuna finestra sul video

command Applicazione (full path) e parametri

waitCompletion Indica se attendere il completamento dell'esecuzione

```
Function FastRoundTo(ByVal value As Double, ByVal decimals As Integer)
As Double
```

Arrotonda al numero di cifre decimali specificato

value Valore da arrotondare

decimals Numero di cifre decimali

Il parametro decimals può contenere anche un valore negativo, nel qual caso l'arrotondamento verrà considerato sul numero di cifre intere

```
Function FloatStrEx(ByVal value As Double, ByVal format As TFloatFormat,
ByVal  precision As Integer, ByVal digits As Integer) As String
```

Converte un valore numerico in testo

value Valore da convertire

format Formato di conversione

precision Precisione della conversione

digits Numero di cifre della conversione

Maggiori informazioni relative ai parametri format, precision e digits possono essere ottenute a questo indirizzo:

http://docwiki.embarcadero.com/Libraries/XE2/en/System.SysUtils.FloatToStrF

```
Function FloatStrP(ByVal value As Double, ByVal decimals As Integer) As
String
```
Converte un valore numerico in testo

value	Valore da convertire
decimals	Numero di cifre decimali

```
Function FloatStrS(ByVal value As Double, Optional ByVal showCurrSymbol
As Boolean = false, ByVal decimals As Integer = 1) As String
```
Converte un valore numerico in testo

value	Valore da convertire
showCurrSymbol	Indica se il testo deve contenere anche il simbolo monetario
decimals	Numero di cifre decimali

```
Function FromInt64(ByVal value As OBJECT_ID) As String
```
Converte un valore intero a 64 bit in testo

value	Valore da convertire

```
Function GenerateID() As OBJECT_ID
```
Genera un valore intero a 64 bit univoco. Ogni chiamata alla funzione GenerateID garantisce la generazione di un valore diverso dai precedenti.

```
Function GetDocPath() As String
```
Restituisce il percorso della cartella Documenti dell'utente di Windows

```
Function GetLocalePath() As String
```
Restituisce il percorso della cartella dove sono memorizzati i file di traduzione del software

```
Function GetTempFile(Optional ByVal extension As String = '.tmp') As
String
```
Genera un nome di file temporaneo univoco.

extension	Estensione da utilizzare (parametro opzionale)

```
Function GetToolsPath() As String
```
Restituisce il percorso della cartella dove sono installati i Fiuto Tools

```
Function LinearRegValue(ByVal priceData As Pointer, ByVal numElements As
Integer, ByVal length As Integer, ByVal targetBar As Integer) As Double
```
Calcola il valore della regressione lineare

priceData	Vettore dei dati su cui calcolare la regressione lineare
numElements	Numero di elementi del vettore priceData
length	Lunghezza sulla quale calcolare la regressione lineare
targetBar	Offset di calcolo della regressione lineare (consigliato 0)

```
Function NormalizePath(ByVal inputPath As String) As String
```
Normalizza il percorso in ingresso rimuovendo separatori doppi

inputPath	Percorso in ingresso

```
Function ParseDateTime(ByVal value As String) As TDateTime
```
Converte il testo in ingresso in un valore di tipo data/ora

value	Valore da convertire

```
Function RoundTick(ByVal price As Double, ByVal tick As Double) As
Double
```
Arrotonda un prezzo al tick più vicino

price	Prezzo da arrotondare
tick	Scostamento minimo del prezzo

```
Function StatisticalVolatility(ByVal priceData As Pointer, ByVal
numElements As Integer, ByVal length As Integer) As Double
```
Calcola la volatilità statistica

priceData	Vettore dei dati su cui calcolare la volatilità statistica
numElements	Numero di elementi del vettore priceData
length	Lunghezza sulla quale calcolare la volatilità statistica

```
Function ToCurrency(ByVal value As Double, Optional ByVal showCurrSymbol
As Boolean = true, Optional ByVal decimals As Integer = 2) As String
```
Converte il valore numerico in ingresso in un testo monetario

value	Valore da convertire
showCurrSymbol	Indica se il testo deve contenere anche il simbolo monetario
decimals	Numero di cifre decimali

```
Function ToDouble(ByVal text As String, Optional ByVal default_value As
Double = 0) As Double
```
Converte un testo in un valore numerico

text	Testo da convertire
default_value	Valore di default in caso di errore di conversione

```
Function ToInt64(ByVal value As String) As OBJECT_ID
```
Converte un testo in un valore intero a 64 bit

value	Testo da convertire

```
Function XAverage(ByVal currentValue As Double, ByVal previousXAvg As
Double, ByVal length As Integer) As Double
```
Calcola la media mobile esponenziale

currentValue	Valore attuale dei dati di ingresso
previousXAvg	Valore precedente della media mobile esponenziale
length	Lunghezza su cui calcolare la media mobile esponenziale

```
Sub Debug(ByVal text As String)
```
Scrive una riga di testo nel file di log di FiutoPRO

text	Testo da scrivere nel file di log di FiutoPRO

```
Sub MostraMessaggio(ByVal text As String)
```
Visualizza un messaggio di testo a video

text	Messaggio da mostrare a video

```
Sub POPlaySound(ByVal fileName As String)
```
Avvia la riproduzione di un file sonoro .wav

fileName	Nome del file da riprodurre

6. Gli oggetti di FiutoPRO

6.1. La struttura della strategia di FiutoPRO

In FPSS sono disponibili tutte le costanti e le classi utilizzate dall'applicazione stessa. Di seguito vengono illustrate queste caratteristiche specifiche di FPSS, fornendone una rapida descrizione.

6.2. Enumerazioni specifiche di FiutoPRO

Le enumerazioni rappresentano un tipo di dato che può assumere solo un insieme predefinito di valori. Un tipo enumerazione è assimilabile ad un valore di tipo intero.

6.2.1. Enumerazione TOptionClass

Identifica lo stile di una opzione

`AMERICAN`

Identifica un'opzione di stile Americano. E' equivalente al carattere 'A'.

`EUROPEAN`

Identifica un'opzione di stile Europeo. E' equivalente al carattere 'E'.

6.2.2. Enumerazione TOptionType

Identifica il tipo di una opzione

`CALL`

Identifica un'opzione Call. E' equivalente al carattere 'C'.

`PUT`

Identifica un'opzione Put. E' equivalente al carattere 'P'.

6.2.3. Enumerazione TOptionCalcMethod

Identifica il metodo di calcolo per una opzione

`BAW`

Metodo di calcolo Barone-Adesi-Whaley

`BINOMIAL`

Metodo di calcolo con albero Binomiale

`BLACKSCHOLES`

Metodo di calcolo Black & Scholes (default)

`FINITEDIFF`

Metodo di calcolo ad elementi finiti

`FUTUREBINOMIAL`

Metodo di calcolo Binomiale per opzioni su futures

`FUTUREBLACKSCHOLES`

Metodo di calcolo Black & Scholes per opzioni su futures

6.2.4. Enumerazione TLimitPrice

Identifica il tipo di prezzo utilizzato per la creazione di un ordine, sia in Paper Trading che a mercato reale

`MARKET`

Prezzo di mercato, ossia “Al Meglio”.

`JOIN`

In caso di acquisto è pari al prezzo Bid. In caso di vendita è pari al prezzo Ask.

`IMPROVE`

In caso di acquisto è pari al prezzo Bid più un tick. In caso di vendita è pari al prezzo Ask meno un tick.

`SPLIT`

E’ pari al prezzo medio Bid-Ask.

`SHAVE`

In caso di acquisto è pari al prezzo Ask meno un tick. In caso di vendita è pari al prezzo Bid più un tick.

`HIT_TAKE`

Equivalente a MARKET.

`HIT_TAKE_PLUS`

In caso di acquisto è pari al prezzo Ask più un tick. In caso di vendita è pari al prezzo Bid meno un tick.

`USER_DEFINED_PRICE`

Prezzo definito dall’utente.

`MEAN`

Equivalente a SPLIT.

`AtMarket`

Equivalente a MARKET.

6.2.5. Enumerazione TRTOrderStatus

Identifica lo stato di ordine, sia in Paper Trading che a mercato reale

`RTOS_INBASKET`

Indica che l’ordine è “parcheggiato” nel Basket di FiutoPRO, pronto per essere eseguito in Paper Trading o a mercato reale.

`RTOS_INBOOK`

Indica che l’ordine è stato inviato al mercato reale ed è esposto sul book.

`RTOS_PARTIAL`

Indica che l’ordine è stato parzialmente eseguito.

`RTOS_COMPLETED`

Indica che l’ordine è stato completato. Questo stato non indica che l’ordine è stato completamente eseguito, ma soltanto che è stato completato.

`RTOS_CANCELLED`
Indica che l'ordine è stato cancellato.

`RTOS_NOT_EXECUTED`
Indica che l'ordine non è stato eseguito, ad esempio perchè rifiutato dal broker o dal mercato.

`RTOS_HIDDEN`
Indica che l'ordine è nascosto, cioè è utilizzato internamente da FiutoPRO per le valutazioni teoriche.

`ORDER_STATUS_IN_BASKET`
Equivalente ad RTOS_INBASKET.

`ORDER_STATUS_IN_BOOK`
Equivalente ad RTOS_INBOOK.

`ORDER_STATUS_COMPLETED`
Equivalente ad RTOS_COMPLETED.

`ORDER_STATUS_CANCELLED`
Equivalente ad RTOS_CANCELLED.

`ORDER_STATUS_ERROR`
Equivalente ad RTOS_NOT_EXECUTED.

6.2.6. Enumerazione TRTHistoryTimeFrame

Identifica il TimeFrame di un insieme di dati storici e relativi indicatori

`RTH_1MIN`
Identifica il TimeFrame ad 1 minuto.

`RTH_5MIN`
Identifica il TimeFrame a 5 minuti.

`RTH_15MIN`
Identifica il TimeFrame a 15 minuti.

`RTH_30MIN`
Identifica il TimeFrame a 30 minuti.

`RTH_1H`
Identifica il TimeFrame ad 1 ora.

`RTH_2H`
Identifica il TimeFrame a 2 ore.

`RTH_4H`
Identifica il TimeFrame a 4 ore.

`RTH_1D`
Identifica il TimeFrame Daily.

`RTH_1W`
Identifica il TimeFrame Weekly.

`RTH_1M`
Identifica il TimeFrame Monthly.

6.2.7. Enumerazione THistoryDataState
Identifica lo stato del contesto di gestione dei dati storici. Ogni singolo contesto ha il suo stato, ed in FiutoPRO un contesto di gestione dei dati storici è individuato dall'insieme di codice isin, timeframe, tipo di grafico, numero massimo di barre.
`HIST_DATA_STATE_INVALID`
Identifica uno stato non valido o non definito.

`HIST_DATA_STATE_NONE`
Identifica lo stato in cui il contesto di gestione dei dati storici non ha ulteriori operazioni da compiere perchè completamente inizializzato o perchè nessun'altra operazione può essere eseguita.

`HIST_DATA_STATE_STOP_RT`
Identifica lo stato di ricezione dei dati in tempo reale è spenta.

`HIST_DATA_STATE_READ_CACHE`
Identifica lo stato di lettura dei dati storici dalla cache di FiutoPRO.

`HIST_DATA_STATE_REFRESH`
Identifica lo stato di attesa dei dati storici aggiornati dalla connessione con il broker.

`HIST_DATA_STATE_WAIT_REFRESHED`
Identifica lo stato di dati storici aggiornati ricevuti dalla connessione con il broker.

`HIST_DATA_STATE_SAVE`
Identifica lo stato di salvataggio dei dati storici aggiornati sulla cache di FiutoPRO.

`HIST_DATA_STATE_READ_FINAL`
Identifica lo stato di rilettura finale dei dati storici aggiornati dalla cache di FiutoPRO.

`HIST_DATA_STATE_START_RT`
Identifica lo stato di avvio della connessione real-time per ricevere i prezzi aggiornati dalla connessione con il broker.

6.2.8. Enumerazione THistoryChartMode
Questa enumerazione definisce i modi di funzionamento dei contesti di gestione dei dati storici.
`HCM_NORMAL`
Identifica il modo normale di utilizzo dei dati storici. Per sottostanti di tipo indice con relativo futures, i dati storici ed i prezzi in tempo reale sono relativi all'indice. In tutti gli altri casi è l'unico modo di funzionamento disponibile.

`HCM_CALC_INDEX`
Identifica il modo di calcolo teorico dell'indice a partire dal valore del future. Per sottostanti di tipo indice con relativo futures, i dati storici sono relativi all'indice, mentre i dati in tempo reale sono calcolati a partire dal valore del future.

`HCM_FUTURE`
Identifica il modo Future. Per sottostanti di tipo indice con relativo future, i dati storici ed i prezzi in tempo reale sono relativi al future. Viene utilizzato il future di tipo Continuos se disponibile, altrimenti il future con scadenza più breve possibile.

6.2.9. Enumerazione THedgeMode
Questa enumerazione definisce in quali modi il sistema di hedging automatico può essere eseguito.
`HEDGE_INSTITUTIONAL`
Hedging istituzionale. Il valore delle greche da proteggere è calcolato come somma algebrica dei valori di tutte le opzioni che compongono la strategia che risultano abilitate al calcolo.

`HEDGE_THRESHOLD`
Hedging a Soglie. Il valore delle greche da proteggere è calcolato come somma algebrica dei valori delle sole opzioni che compongono la strategia che risultano abilitate al calcolo e la cui soglia di intervento risulti attiva.

6.2.10. Enumerazione THedgeDeltaMode
Definisce in quali modi il sistema di hedging automatico può calcolare il valore del Delta di portafoglio delle opzioni che costituiscono la strategia.
`HEDGE_DELTA_CLASSIC`
Calcolo del Delta di portafoglio classico

`HEDGE_DELTA_SMART`
Calcolo del Delta di portafoglio Smart. L'algoritmo di calcolo Smart è una esclusiva di FiutoPRO. L'obietto di questo algoritmo è quello di, in caso di opzione ITM venduta, cercare di avere in portafoglio la quantità necessaria di sottostante al prezzo di carico migliore possibile.

6.2.11. Enumerazione THedgeFrequency
Definisce i modi disponibili per selezionare la frequenza di hedging automatico.
`HEDGE_FREQ_CONTRACTS`
Hedging automatico avviato quando la quantià di contratti da eseguire supera una soglia minima impostata dall'utente. E' l'impostazione di default.

`HEDGE_FREQ_TIME`
Hedging automatico avviato ad intervalli regolari di tempo definiti dall'utente.

`HEDGE_FREQ_FIXED`
Hedging automatico avviato ad un orario prefissato definito dall'utente. Esegue l'hedging automatico una sola volta al giorno.

`HEDGE_FREQ_PERCENT`
Hedging automatico avviato solo quando la variazione di prezzo percentuale del sottostante principale della strategia supera una soglia minima impostata dall'utente.

6.2.12. Enumerazione THedgePrice
Questa enumerazione definisce quali prezzi devono essere usati per le opzioni dal sistema di calcolo dell'hedging automatico.
`HEDGE_PRICE_MARKET`
Utilizza sempre e solamente i prezzi di mercato. E' l'impostazione di default.

`HEDGE_PRICE_THEORICAL`
Utilizza sempre e solamente i prezzi teorici calcolati dal sistema eMaker di FiutoPRO.

`HEDGE_PRICE_MIXED`
Utilizza i prezzi di mercato finchè sono identificati come validi da FiutoPRO, poi passa automaticamente ai prezzi teorici calcolati dal sistema eMaker.

6.2.13. Enumerazione TOrderSource
Identifica il modo in cui un ordine è stato creato.
`ORDER_SOURCE_MANUAL`
Ordine creato manualmente dall'utente.

`ORDER_SOURCE_HEDGING`
Ordine creato automaticamente dal sistema di hedging.

`ORDER_SOURCE_WORKFLOW`
Ordine creato automaticamente dai Workflows.

`ORDER_SOURCE_SCRIPT`
Ordine creato automaticamente durante l'esecuzione di uno script.

6.2.14. Enumerazione TOrderCompletedReason
Specifica per quale ragione un ordine ha raggiunto lo stato di completamento (RTOS_COMPLETED).
`ORDER_COMPLETED_NONE`
Ragione di completamento non conosciuta.

`ORDER_COMPLETED_REGISTERED`
Ordine registrato manualmente.

`ORDER_COMPLETED_PAPERTRADING`
Ordine completato perchè inviato in Paper Trading.

`ORDER_COMPLETED_MARKET`
Ordine completato perchè è stato ricevuto lo stato di completamento dal mercato.

`ORDER_COMPLETED_AUTOHEDGE`
Ordine completato dal sistema di hedging automatico

6.2.15. Enumerazione TCorrectionType

Specifica quali tipi di correzione di greche sono disponibili.

`CT_NONE`

Nessun tipo di correzione

`CT_RHO`

Correzione di Rho

`CT_VEGA`

Correzione di Vega

`CT_THETA`

Correzione di Theta

`CT_GAMMA`

Correzione di Gamma

`CT_DELTA`

Correzione di Delta

6.2.16. Enumerazione TUsedPrice

Specifica quale prezzo viene usato per calcolare il Profit/Loss attuale dello strumento. FiutoPRO sceglie automaticamente quale prezzo utilizzare, calcolando la condizione peggiore possibile tra le scelte disponibili.

`PRICE_USE_LAST`

Utilizza il prezzo Last.

`PRICE_USE_BID`

Utilizza il prezzo Bid.

`PRICE_USE_ASK`

Utilizza il prezzo Ask.

6.2.17. Enumerazione TSymbolType

Specifica il tipo dello strumento.

`STOCK`

Azione o Indice

`STOCK_OPTION`

Opzione su azione o indice

`FUTURE`

Future

`FUTURE_OPTION`

Opzione su future

`FUTURE_CONTINUOS`

Future Continuos

6.2.18. Enumerazione TMoneyness

Specifica la moneyness dell'opzione

`OTM`

Opzione Out-The-Money

`ATM`

Opzione At-The-Money

`ITM`

Opzione In-The-Money

6.2.19. Enumerazione TStrategyRecalcMode

Identifica i modi di ricalcolo dei valori delle strategie

`CALC_MODE_NORMAL`

Metodo di calcolo completo

`CALC_MODE_FAST`

Metodo di calcolo ridotto, non tutte le proprietà vengono ricalcolate

`CALC_MODE_PAYOFF_PREVIEW`

Metodo di calcolo per visualizzare l'anteprima del grafico del payoff

`CALC_MODE_ATNOW`

Calcola esclusivamente le proprietà della strategia dipendenti dai prezzi di mercato

6.2.20. Enumerazione TConfidenceLevel

Specifica i valori disponibili per il livello di confidenza utilizzato nel calcolo del Value At Risk.

`CONFIDENCE_90`

Livello di confidenza del 90%

`CONFIDENCE_95`

Livello di confidenza del 95%

`CONFIDENCE_99`

Livello di confidenza del 99%

`CONFIDENCE_9999`

Livello di confidenza del 99,99%

6.3. Classi specifiche di FiutoPRO

6.3.1. Classe TDividends

La classe TDividends è utilizzata per gestire l'elenco dei dividendi associati ad un sottostante. Ogni singolo dividendo è composto da una coppia di valori: la data di stacco e l'ammontare dei dividendi. I singoli dividendi vengono automaticamente ordinati in base alla data di stacco.

6.3.1.1. Proprietà

```
OBJECT_ID ObjectID
```
Codice identificativo dell'oggetto

```
Integer Size
```
[Read-Only] Numero di dividendi presenti nell'oggetto

6.3.1.2. Metodi

```
Sub Add(ByVal date As TDateTime, ByVal value As Double)
```
Aggiunge un dividendo all'elenco. Se un dividendo con la stessa data esiste già nell'elenco dei dividendi, questo viene aggiornato con il nuovo valore.

date	Data di stacco del dividendo
value	Ammontare del dividendo

```
Sub Clear
```
Azzera l'elenco dei dividendi

```
Sub CopyFrom(ByVal source As TDividends)
```
Copia l'elenco dei dividendi da un altro oggetto di tipo TDividends

source	Oggetto di tipo TDividends da cui copiare i valori

```
Sub Delete(ByVal index As Integer)
```
Elimina un dividendo in base al suo indice nell'elenco dei dividendi

index	Indice del dividendo da eliminare

```
Sub DeleteByDate(ByVal date As TDateTime)
```
Elimina un dividendo in base alla data di stacco

date	Data di stacco del dividendo da eliminare dall'elenco

```
Sub Edit(ByVal index As Integer, ByVal date As TDateTime, ByVal value As
Double)
```
Modifica un dividendo in base al suo indice nell'elenco dei dividendi. Attenzione: se si modifica la data di stacco del dividendo, il suo indice nell'elenco dei dividendi potrebbe modificarsi.

index	Indice del dividendo da modificare
date	Nuova data di stacco del dividendo
value	Valore del dividendo

```
Sub EditByDate(ByVal oldDate As TDateTime, ByVal newDate As TDateTime,
ByVal value As Double)
```
Modifica un dividendo in base alla sua data di stacco

oldDate	Data di stacco del dividendo da modificare
newDate	Nuova data di stacco del dividendo
value	Valore del dividendo

```
Function GetDate(ByVal index As Integer) As TDateTime
```
Ottiene la data di stacco di un dividendo in base al suo indice nell'elenco dei dividendi

index	Indice del dividendo di cui ottenere la data di stacco

```
Function GetValue(ByVal index As Integer) As Double
```
Ottiene l'ammontare del dividendo in base al suo indice nell'elenco dei dividendi

index	Indice del dividendo di cui ottenere l'ammontare

```
Function GetValueByDate(ByVal date As TDateTime) As Double
```
Ottiene l'ammontare del dividendo in base alla sua data di stacco
date Data di stacco del dividendo di cui ottenere l'ammontare

```
Function Sum(ByVal startDate As TDateTime, ByVal endDate As TDateTime)
As Double
```
Somma l'ammontare di tutti i dividendi la cui data di stacco è compresa nell'intervallo specificato
startDate Data minima di stacco dei dividendi da sommare
endDate Data massima di stacco dei dividendi da sommare

6.3.2. Classe TOptionCalc

Questa classe viene utilizzata per calcolare i valori teorici delle opzioni in modo quanto più possibile automatizzato. L'utilizzo di oggetti di questa classe è generalmente del tipo "set and forget", cioè "imposta e dimentica". Di fatto, dopo la prima fase di impostazione, è necessario cambiare un singolo parametro di calcolo per ottenere automaticamente i valori teorici corrispondenti alla nuova combinazione di parametri di calcolo.

6.3.2.1. Proprietà

```
TDateTime AnalisysDate
```
Data di analisi

```
Boolean CalcEnabled
```
Ricalcolo automatico abilitato/disabilitato

```
Double Delta
```
[Read-Only] Delta dell'opzione

```
TDividends Dividends
```
Oggetto che gestisce i dividendi associati al sottostante dell'opzione

```
TDateTime ExpiryDate
```
Data di scadenza dell'opzione

```
Double Gamma
```
[Read-Only] Gamma dell'opzione

```
TOptionCalcMethod Method
```
Metodo di calcolo utilizzato

```
Integer NumSteps
```
Numero di "passaggi" di calcolo massimi utilizzati per i calcoli teorici con i metodi ricorsivi oppure nel calcolo inverso della volatilità implicita

```
OBJECT_ID ObjectID
```
Numero identificativo dell'oggetto

```
TOptionClass OptionClass
```
Stile dell'opzione

```
TOptionType OptionType
```
Tipo dell'opzione

```
Double Price
```
[Read-Only] Premio dell'opzione

```
Double Rho
```
[Read-Only] Rho dell'opzione

```
Double RiskFreeRate
```
Tasso di interesse privo di rischio, espresso in percentuale (ad esempio un valore di 1 è pari ad un tasso di interesse privo di rischio di 1%)

```
Double Strike
```
Strike dell'opzione

```
Double StrikeStock
```
[Read-Only] Rapporto Strike su prezzo del sottostante

```
Double TheoreticalDividends
```
Dividendi teorici associati al sottostante calcolati fino alla data di scadenza dell'opzione

```
Double Theta
```
[Read-Only] Theta dell'opzione

```
Double TimeValue
```
[Read-Only] Valore temporale intrinseco dell'opzione

```
Double Underlying
```
Prezzo del sottostante dell'opzione

```
Double Vega
```
[Read-Only] Vega dell'opzione

```
Double Volatility
```
Volatilità implicita dell'opzione, espressa in percentuale (ad esempio un valore di 20 è pari ad una volatilità implicita di 20%)

```
Double YearPrice
```
[Read-Only] Premio annualizzato dell'opzione

6.3.2.2. Metodi

```
Sub AutoSetByClass(ByVal optionClass As TOptionClass, ByVal futureOption
As Boolean)
```
Imposta automaticamente il metodo di calcolo dell'opzione

optionClass	Stile dell'opzione
futureOption	Indica se l'opzione ha come sottostante un future

```
Function CalculateVolatility(ByVal targetPrice As Double, Optional ByVal
autoset As Boolean = false, Optional ByVal precision As Integer = -1) As
Double
```
Calcola la volatilità implicita dell'opzione in base al premio

targetPrice	Premio per il quale calcolare la volatilità implicita
autoset	Indica se deve essere automaticamente impostata la proprietà Volatility

precision Indica quante cifre decimali di precisione si desidera nel calcolo del premio teorico dell'opzione. Un valore negativo indica che il software sceglie autonomamente il numero di cifre decimali da utilizzare

```
Sub CopyFrom(ByVal source As TOptionCalc)
```
Copia tutte le proprietà da un altro oggetto di tipo TOptionCalc
source Oggetto da cui copiare tutte le proprietà

```
Function GetCanHighlight(ByVal property As String) As Boolean
```
Verifica se la proprietà specifica supporta l'highlight per evidenziare le variazioni del valore nel tempo
property Proprietà richiesta

```
Function GetHighlightStatus(ByVal property As String) As
THighlightStatus
```
Ottiene lo stato di evidenziazione della proprietà specificata
property Proprietà richiesta

```
Function GetProperty(ByVal property As String) As String
```
Ottiene il valore istantaneo della proprietà specificata
property Proprietà richiesta

```
Sub GetPropertyList(ByVal list As TStringList, Optional ByVal clearList
As Boolean = false, Optional ByVal Prefix As String = "", Optional ByVal
Suffix As String = "")
```
[Class Method] Ottiene l'elenco dell proprietà disponibili per gli oggetti della classe TOptionCalc
list Lista di stringhe dove scrivere l'elenco delle proprietà disponibili
clearList Indica se azzerare la lista prima di scrivere l'elenco delle proprietà
Prefix Prefisso da utilizzare prima del nome di ogni proprietà
Suffix Suffisso da utilizzare dopo il nome di ogni proprietà

```
Sub SetVolatility(ByVal value As Double, Optional ByVal forceRecalc As
Boolean = false)
```
Imposta la volatilità implicita dell'opzione e ricalcola i valori teorici se richiesto
value Valore della volatilità implicita
forceRecalc Indica se il ricalcolo dei valori teorici è comunque necessario

6.3.3. Classe TDataVector

Questa classe viene utilizzata per memorizzare una serie di dati storici. Un oggetto di tipo TDataVector rappresenta un singolo insieme di dati, ad esempio i soli prezzi di chiusura delle quotazioni storiche. I dati memorizzati sono ordinati ed indicizzati in modalità zero-based.

6.3.3.1. Proprietà

```
Integer Capacity
```
[Read-Only] Capacità attuale del vettore di dati, non indica il numero di dati presenti nel vettore

```
Double Last
```
[Read-Only] Ultimo valore in ordine temporale

```
Double LastClosed
```
[Read-Only] Penultimo valore in ordine temporale

```
Double Max
```
[Read-Only] Valore massimo nel vettore di dati

```
Integer MaxIndex
```
[Read-Only] Indice nel vettore di dati dove si trova il valore massimo

```
Double Min
```
[Read-Only] Valore minimo nel vettore di dati

```
Integer MinIndex
```
[Read-Only] Indice nel vettore di dati dove si trova il valore minimo

```
Integer Size
```
[Read-Only] Numero di elementi presenti nel vettore di dati

```
Double Value [index As Integer]
```
[Read-Only] Valore dell'elemento specificato del vettore di dati

6.3.3.2. Metodi

```
Sub Append(ByVal value As Double)
```
Aggiunge un nuovo valore al vettore dei dati, in coda agli altri come ultimo valore più recente

value Valore da accodare

```
Sub Clear
```
Azzera il vettore dei dati

```
Sub ClearAndResize(ByVal size As Integer)
```
Azzera il vettore dei dati e ne prepara uno nuovo della dimensione specificata, in cui tutti i valori saranno inizialmente impostati a zero

size Dimensione del nuovo vettore di dati

```
Sub CopyFrom(ByVal source As Pointer, ByVal size As Integer)
```
Copia i dati dai parametri in ingresso al vettore dei dati interno, ridimensionandolo se necessario

source Indirizzo in memoria dove si trovano i dati da copiare

size Numero di elementi da copiare

```
Function CrossesAbove(ByVal compareTo As TDataVector) As Boolean
```
Verifica se i valori più recenti attraversano verso l'alto i valori più recenti del vettore specificato

compareTo Vettore di dati di comparazione per l'attraversamento

```
Function CrossesAboveValue(ByVal value As Double) As Boolean
```
Verifica se i valori più recenti attraversano verso l'alto il valore specificato

value Valore di comparazione per l'attraversamento

```
Function CrossesBelow(ByVal compareTo As TDataVector) As Boolean
```
Verifica se i valori più recenti attraversano verso il basso i valori più recenti del vettore specificato

compareTo Vettore di dati di comparazione per l'attraversamento

```
Function CrossesBelowValue(ByVal value As Double) As Boolean
```
Verifica se i valori più recenti attraversano verso il basso il valore specificato
value Valore di comparazione per l'attraversamento

```
Function GetVector() As Pointer
```
Restituisce l'indirizzo di memoria del vettore dei dati interno

```
Function Highest(ByVal lenght As Integer) As Double
```
Restituisce il valore più elevato tra i più recenti
lenght Numero di elementi tra cui ricercare il valore più elevato

```
Function Lowest(ByVal lenght As Integer) As Double
```
Restituisce il valore minore tra i più recenti
lenght Numero di elementi tra cui ricercare il valore minore

```
Sub Resize(ByVal size As Integer)
```
Ridimensiona il vettore dei dati
size Nuova dimensione del vettore

```
Sub Set(ByVal index As Integer, ByVal value As Double)
```
Imposta un valore ad una posizione specifica nel vettore dei dati
index Indice nel vettore dei dati da impostare
value Valore da impostare

```
Sub ShiftRight(ByVal numElems As Integer, ByVal beginIndex As Integer)
```
Inserisci dei valori nulli prima della posizione specificata del vettore dei dati ridimensionandolo se necessario
numElems Numero di elementi nulli da inserire
beginIndex Posizione da cui iniziare ad inserire i valori nulli

```
Sub Zero()
```
Azzera tutti i valori del vettore dei dati senza ridimensionarlo

6.3.4. Classe THistoricalData

Questa classe gestisce un insieme di oggetti TDataVector che costituiscono i dati storici per un sottostante. Permette inoltre di attivare la connessione real-time per aggiornare i prezzi, e supporta le diverse modalità di calcolo e visualizzazione del grafico storico.

6.3.4.1. Proprietà

```
Double Ask
```
[Read-Only] Prezzo Ask

```
Integer AskSize
```
[Read-Only] Quantità presente nel primo livello del book per il prezzo Ask

```
Double Bid
```
[Read-Only] Prezzo Bid

```
Integer BidSize
```
[Read-Only] Quantità presente nel primo livello del book per il prezzo Bid

```
THistoryChartMode ChartMode
```
Modalità di calcolo e visualizzazione del grafico storico associato a questo oggetto

```
Double Close [Optional Index As Integer]
```
[Read-Only] Prezzo Close. E' possibile specificare di quale barra si vuole ottenere il prezzo Close, usando l'indicizzazione. L'indicizzazione è zero-based a partire dalla barra più recente.

```
Boolean Connected
```
[Read-Only] Indica se è attiva una connessione per ricevere i prezzi aggiornati in real-time

```
TDateTime Date [Optional Index As Integer]
```
[Read-Only] Data e Ora. E' possibile specificare di quale barra si vuole ottenere la data, usando l'indicizzazione. L'indicizzazione è zero-based a partire dalla barra più recente.

```
String DateTimeFormat
```
[Read-Only] Stringa di formattazione consigliata per il campo data e ora in base al time frame attivo

```
Double DisplayLast
```
[Read-Only] Valore da visualizzare come prezzo Last sul grafico storico

```
Double High [Optional Index as Integer]
```
[Read-Only] Prezzo High. E' possibile specificare di quale barra si vuole ottenere il prezzo High, usando l'indicizzazione. L'indicizzazione è zero-based a partire dalla barra più recente.

```
Boolean IsCached
```
[Read-Only] Indica se il sistema sta utilizzando dati prelevati dalla cache di FiutoPRO oppure aggiornati dalla connessione con il broker

```
String Isin
```
[Read-Only] Isin del sottostante a cui i dati si riferiscono

```
Boolean IsIndex
```
[Read-Only] Indica se il sottostante a cui i dati si riferiscono sono relativi ad un indice

```
Double Last
```
[Read-Only] Prezzo Last

```
Double LastClosed
```
[Read-Only] Prezzo Last dell'ultima barra completata

```
Integer LastSize
```
[Read-Only] Quantità dell'ultimo scambio avvenuto

```
Double Low [Optional Index As Integer]
```
[Read-Only] Prezzo Low. E' possibile specificare di quale barra si vuole ottenere il prezzo Low, usando l'indicizzazione. L'indicizzazione è zero-based a partire dalla barra più recente.

```
String Name
```
[Read-Only] Nome del sottostante a cui si riferiscono i dati

```
Double Open [Optional Index As Integer]
```
[Read-Only] Prezzo Open. E' possibile specificare di quale barra si vuole ottenere il prezzo Open, usando l'indicizzazione. L'indicizzazione è zero-based a partire dalla barra più recente.

```
Integer RecordCount
```
[Read-Only] Indica quante barre sono presenti nei dati storici

```
Double Reference
```
[Read-Only] Prezzo di riferimento.

```
Integer ReserveSize
```
[Read-Only] Dimensione di allocazione consigliata per i vettori dei dati interni

```
Integer SizeLimit
```
Specifica il numero massimo iniziale di barre da utilizzare. Con il trascorrere del tempo, il numero di barre presenti potrebbe superare questo valore.

```
THistoryDataState Status
```
Stato dell'oggetto

```
TRTHistoryTimeFrame TimeFrame
```
TimeFrame dei dati storici

```
TDataVector VectorClose
```
[Read-Only] Ottiene il vettore di dati usato per i prezzi Close

```
TDataVector VectorDate
```
[Read-Only] Ottiene il vettore di dati usato per la data e l'ora

```
TDataVector VectorHigh
```
[Read-Only] Ottiene il vettore di dati usato per i prezzi High

```
TDataVector VectorLow
```
[Read-Only] Ottiene il vettore di dati usato per i prezzi Low

```
TDataVector VectorOpen
```
[Read-Only] Ottiene il vettore di dati usato per i prezzi Open

```
TDataVector VectorVolume
```
[Read-Only] Ottiene il vettore di dati usato per i Volumi

```
Double Volume [Optional Index As Integer]
```
[Read-Only] Volume. E' possibile specificare di quale barra si vuole ottenere il Volume, usando l'indicizzazione. L'indicizzazione è zero-based a partire dalla barra più recente.

6.3.4.2. Metodi

```
Function CanHighlight(ByVal property As String) As Boolean
```
Verifica se una proprietà supporta l'evidenziazione della variazione di valore
property Proprietà

```
Sub Export(ByVal fileName As String)
```
Esporta i dati storici in un file
fileName Nome del file di destinazione

```
Function GetHiglightStatus(ByVal property As String) As THighlightStatus
```
Ottiene lo stato di evidenziazione della proprietà specificata
property Proprietà

```
Function GetHistoricalVolatility(ByVal length As Integer) As Double
```
Calcola la volatilità storica alla lunghezza specificata
length Lunghezza alla quale calcolare la volatilità storica

```
Function GetProperty(ByVal property As String) As String
```
Ottiene il valore della proprietà specificata
property Proprietà

```
Sub GetPropertyList(ByVal list As TStringList, Optional ByVal clearList
As Boolean = false, Optional ByVal prefix As String = '', Optional
suffix As String = '')
```
[Class Method] Ottiene l'elenco dell proprietà disponibili per gli oggetti della classe THistoricalData
list Lista di stringhe dove scrivere l'elenco delle proprietà disponibili
clearList Indica se azzerare la lista prima di scrivere l'elenco delle proprietà
Prefix Prefisso da utilizzare prima del nome di ogni proprietà
Suffix Suffisso da utilizzare dopo il nome di ogni proprietà

```
Sub Import(ByVal fileName As String)
```
Importa i dati storici da un file
fileName Nome del file da utilizzare come sorgente dei dati storici

```
Function IsNewBarTime(ByVal currentDateTime As TDateTime, ByVal
previousDateTime As TDateTime, ByVal timeFrame As TRTHistoryTimeFrame)
As Boolean
```
[Class Method] Verifica se l'orario specificato appartiene ad una nuova barra di dati sotirici oppure appartiene ancora alla precedente
currentDateTime Data e ora da verificare
previousDateTime Data e ora della barra precedente
timeFrame Time Frame dei dati storici

```
Sub ReadData()
```
Carica i dati storici dalla cache di FiutoPRO

```
Sub SetLastBar(ByVal dateTime As TDateTime, ByVal open As Double, ByVal
high As Double, ByVal low As Double, ByVal close As Double, ByVal volume
As Double)
```
Imposta i valori della barra più recente
dateTime Data e ora da associare alla barra
open Prezzo Open
high Prezzo High
low Prezzo Low
close Prezzo Close
volume Volume

```
Sub StartRT()
```
Avvia la connessione real-time per ricevere i prezzi aggiornati dal broker

```
Sub StopRT()
```
Spegne la connessione real-time che riceve i prezzi aggiornati dal broker

6.3.5. Classe TIndicators
Rappresenta una collezione di indicatori

6.3.5.1. Metodi
```
Sub DeleteAll()
```
Elimina tutti gli indicatori dalla collezione

```
Sub Delete(ByVal serieName As String)
```
Elimina un indicatore dalla collezione in base al nome della serie di dati associata
serieName Nome della serie di dati associata all'indicatore da eliminare

```
Sub Edit(ByVal serieName As String)
```
Mostra la finestra di impostazione dell'indicatore
serieName Nome della serie di dati associata all'indicatore da modificare

```
Sub Recalc(ByVal recordCount As Integer)
```
Ricalcola tutti gli indicatori della collezione
recordCount Numero di barre di dati storici da utilizzare per il ricalcolo

```
Sub Redraw(ByVal recordCount As Integer)
```
Ridisegna tutti gli indicatori della collezione
recordCount Numero di barre di dati storici da utilizzare per il ridisegno

```
Sub RecalcAndRedraw(ByVal recordCount As Integer)
```
Ricalcola e ridisegna tutti gli indicatori della collezione
recordCount Numero di barre di dati storici da utilizzare per il ricalcolo e ridisegno

6.3.6. Classe THistoryDataContext
Raggruppa un oggetto di tipo THistoricalData ed uno di tipo TIndicators per creare un contesto di dati storici. Ogni serie di dati è associata ad un nome univoco per identificarla.

6.3.6.1. Proprietà
```
THistoryChartMode ChartMode
```
Modo di calcolo e visualizzazione del grafico storico

```
THistoricalData HistoricalData
```
[Read-Only] Oggetto di tipo THistoricalData contenuto all'interno del contesto di dati storici

```
TIndicators Indicators
```
[Read-Only] Oggetto di tipo TIndicators contenuto all'interno del contesto di dati storici

```
String Isin
```
Isin dello strumento a cui sono riferiti i dati storici

```
Integer SizeLimit
```
Numero di barre iniziali dei dati storici

```
TRTHistoryTimeFrame TimeFrame
```
Time Frame dei dati storici

6.3.6.2. Metodi

```
Function Create(ByVal isin As String, ByVal timeFrame As
TRTHistoryTimeFrame, ByVal  chartMode As THistoryChartMode, ByVal
sizeLimit As Integer) As THistoryDataContext
```
[Class Method] Crea un nuovo contesto di dati storici

isin	Isin dello strumento di cui creare il contesto di dati storici
timeFrame	Time Frame dei dati storici
chartMode	Modo di calcolo e visualizzazione del grafico storico
sizeLimit	Numero di barre massimo iniziali

```
Function GetSerieByName(ByVal serieName As String) As TDataVector
```
Restituisce l'oggetto di tipo TDataVector che contiene i dati associati alla serie con il nome specificato

serieName	Nome della serie di cui ottenere i dati

```
Sub GetSeriesNames(ByVal list As TStringList)
```
Ottiene l'elenco dei nomi di tutte le serie di dati del contesto

list	Lista su cui verranno scritti i nomi delle serie

6.3.7. Classe TPayoffLineProperties

Definisce le proprietà di una singola linea disegnata sul grafico di payoff della strategia

6.3.7.1. Proprietà

```
TColor Color
```
Colore della linea

```
TPenStyle Style
```
Stile di disegno della linea, tra psSolid, psDash e psDot

```
Boolean Visible
```
Specifica se la linea deve essere visibile o meno sul grafico di payoff

```
Integer Width
```
Spessore della linea in pixels

6.3.7.2. Metodi

```
Sub CopyFrom(ByVal source As TPayoffLineProperties)
```
Copia le proprietà da un altro oggetto di tipo TPayoffLineProperties

source	Oggetto da cui copiare le proprietà

6.3.8. Classe TStrategySettings

Costituisce l'insieme delle impostazioni di una strategia

6.3.8.1. Proprietà

`Boolean AutoExecAutoOrders`

Indica se gli ordini generati dai sistemi automatici (hedging, workflow e script) devono essere automaticamente eseguiti

`Boolean AutoSave`

Indica se la strategia deve essere automaticamente salvata ad intervalli regolari di tempo e dopo l'esecuzione di un ordine

`Boolean ContinueOrdersAfterError`

Specifica se l'esecuzione degli ordini presenti nel basket deve continuare anche dopo che almeno uno di quelli già eseguiti ha generato un errore

`Boolean DeltaHedge`

Sistema di hedging del Delta attivo per la strategia

`THedgeDeltaMode DeltaMode`

Modo in cui calcolare il Delta da proteggere della strategia

`Boolean DontShowOrderConfirm`

Indica se il software deve evitare di mostrare la finestra di conferma di ordine eseguito

`TDateTime EndSession1`

Orario di chiusura della prima sessione di borsa della strategia

`TDateTime EndSession2`

Orario di chiusura della seconda sessione di borsa della strategia

`Boolean ForceStockDrawing`

Indica se per la strategia tutti i sottostanti devono essere calcolati e disegnati nel grafico di payoff come fossero delle azioni

`Boolean GammaHedge`

Sistema di hedging del Gamma attivo o meno per la strategia

`Boolean GlobalWorkflowsEnable`

Workflows della strategia abilitati alla verifica ed esecuzione

`Boolean GlobalEnableRealOrders`

Ordini a mercato reale per la strategia abilitati

`Boolean GlobalHedgeEnable`

Sistema di hedging per la strategia abilitato

`THedgeFrequency HedgeFrequencyMode`

Modo in cui viene calcolata la frequenza di esecuzione dell'hedging

`Double HedgeFrequencyValue`

Valore definito dall'utente utilizzato per calcolare la frequenza di esecuzione dell'hedging. Per i modi HEDGE_FREQ_TIME e HEDGE_FREQ_FIXED, il valore deve essere specificato come frazione di un giorno, cioè 24h corrispondono al valore 1.

```
THedgeMode HedgeMode
```
Modo in cui il sistema di hedging calcola le quantità per eseguire le coperture

```
Integer MaxHedgingQuantity
```
Quantità massima che il sistema di hedging può proporre per eseguire le coperture

```
Double MaxTickDifferencePercent
```
Differenza massima in percentuale tra il valore di un tick ed il precedente ricevuti dal mercato per considerare il prezzo ricevuto come valido. Il valore è espresso in percentuale, cioè un valore di 1 significa 1%.

```
Double MaxXAverageDifferencePercent
```
Differenza massima in percentuale tra il valore di un tick e la media mobile esponenziale calcolata sugli stessi ricevuti dal mercato per considerare il prezzo ricevuto come valido. Il valore è espresso in percentuale, cioè un valore di 1 significa 1%.

```
Integer MinTickCount
```
Numero di tick minimi da ricevere dal mercato prima di considerare valido il prezzo ricevuto

```
Boolean NotifyRealOrders
```
Notifica via email dell'esecuzione degli ordini inviati a mercato reale

```
String PriceFilterError
```
Messaggio di errore relativo alla verifica dei prezzi ricevuti dal mercato

```
Integer PriceXAverageLen
```
Lunghezza della media mobile esponenziale calcolata sui prezzi ricevuti dal mercato

```
Boolean RhoHedge
```
Sistema di hedging del Rho attivo o meno per la strategia

```
Boolean SendOrdersToBasket
```
Specifica se gli ordini generati manualmente dall'utente devono andare direttamente nel basket senza essere prima confermati

```
Boolean ShowNotExecOrders
```
Specifica se gli ordini non eseguiti devono visualizzare un messaggio di errore

```
TDateTime StartSession1
```
Orario di apertura della prima sessione di borsa della strategia

```
TDateTime StartSession2
```
Orario di apertura della seconda sessione di borsa della strategia

```
Boolean ThetaHedge
```
Sistema di hedging del Theta attivo o meno per la strategia

```
Integer TickTimeout
```
Durata di validità (in millesimi di secondo) di un tick ricevuto dal mercato

```
Boolean VegaHedge
```
Sistema di hedging del Vega attivo o meno per la strategia

```
Boolean VerifyMinTickCount
```
Indica se il sistema di verifica dei prezzi ricevuti dal mercato deve controllare il numero di tick minimi ricevuti

```
Boolean VerifyOptionsLastPrice
```
Indica se il sistema di verifica dei prezzi ricevuti dal mercato deve controllare anche il prezzo Last delle opzioni che costituiscono la strategia

```
Boolean VerifyTickDifference
```
Indica se il sistema di verifica dei prezzi ricevuti dal mercato deve controllare la differenza percentuale tra un tick ed il precedente

```
Boolean VerifyTickTimeout
```
Indica se il sistema di verifica dei prezzi ricevuti dal mercato deve controllare la durata temporale dei tick

```
Boolean VerifyTimeSession
```
Indica se il sistema di verifica dei prezzi ricevuti dal mercato deve controllare gli orari delle sessioni di borsa della strategia

```
Boolean VerifyXAverageDifference
```
Indica se il sistema di verifica dei prezzi ricevuti dal mercato deve controllare la differenza percentuale tra un tick e la media mobile esponenziale calcolata sugli stessi

```
Boolean VerifyZeroTick
```
Indica se il sistema di verifica dei prezzi ricevuti dal mercato deve controllare che il tick ricevuto non sia pari a zero

6.3.8.2. Metodi
```
Sub Clear()
```
Azzera tutte le impostazioni

6.3.9. Classe TStrategyChartInfo
Questa classe contiene tutte le impostazioni necessarie al disegno del grafico di payoff della strategia.

6.3.9.1. Proprietà
```
TPayoffLineProperties AtExpiryLine
```
Proprietà della linea di payoff a scadenza

```
TPayoffLineProperties AtNowLine
```
Proprietà della linea di payoff At Now

```
TColor BackgroundColor
```
Colore di sfondo del grafico di payoff

```
TPayoffLineProperties Compare1Line
```
Proprietà della linea di payoff della strategia 1 nei grafici di comparazione

```
TPayoffLineProperties Compare2Line
```
Proprietà della linea di payoff della strategia 2 nei grafici di comparazione

```
TColor CrosshairColor
```
Colore dell'indicatore CrossHair del grafico di payoff

```
TColor DividendColor
```
Colore del simbolo del prezzo comprensivo di dividendi del grafico di payoff

```
TColor ForegroundColor
```
Colore del testo del grafico di payoff

```
TColor GridColor
```
Colore della griglia del grafico di payoff

6.3.9.2. Metodi

```
Sub SetDefault()
```
Ripristina i valori di default di tutte le proprietà

6.3.10. Classe TBrokerCosts

Questa classe è utilizzata per calcolare i costi commissionali associati ad un singolo ordine

6.3.10.1. Proprietà

```
Double LotSize
```
Dimensione del lotto

```
Double Maximum
```
Valore massimo dei costi commissionali se applicabile. Utilizzare il valore 0 quando non è previsto un valore massimo

```
Double Minimum
```
Valore minimo dei costi commissionali se applicabile. Utilizzare il valore 0 quando non è previsto un valore minimo

```
OBJECT_ID ObjectID
```
Codice identificativo dell'oggetto

```
Boolean PerContract
```
Specifica se i costi devono essere calcolati per singolo contratto o per intera transazione

```
Double PointValue
```
Point Value del contratto di cui calcolare i costi commissionali

```
Double Price
```
Prezzo di eseguito dell'ordine di cui calcolare i costi commissionali

```
Integer Quantity
```
Quantità di contratti eseguiti dell'ordine di cui calcolare i costi commissionali

```
Double Total
```
[Read-Only] Valore calcolato totale dei costi commissionali dell'ordine

```
Double Value
```
Costo base o valore percentuale base

```
Boolean ValueBased
```
Specifica se i costi commissionali devono essere calcolati a partire dal costo base oppure dal valore percentuale base

6.3.10.2. Metodi

```
Function Calculate() As Double
```
Esegue il calcolo dei costi commissionali

```
Sub Clear()
```
Azzera tutte le proprietà

```
Sub CopyFrom(ByVal source As TBrokerCosts)
```
Copia tutte le proprietà da un altro oggetto di tipo TBrokerCosts

source	Oggetto da cui copiare le proprietà

```
Function GetProperty(ByVal property As String) As String
```
Ottiene il valore di una proprietà dell'oggetto

property	Proprietà di cui ottenere il valore

```
Sub GetPropertyList(ByVal list As TStringList, Optional ByVal clearList
As Boolean = false, Optional ByVal prefix As String = "", Optional ByVal
suffix As String = "")
```
[Class Method] Ottiene l'elenco dell proprietà disponibili per gli oggetti della classe TBrokerCosts

list	Lista di stringhe dove scrivere l'elenco delle proprietà disponibili
clearList	Indica se azzerare la lista prima di scrivere l'elenco delle proprietà
Prefix	Prefisso da utilizzare prima del nome di ogni proprietà
Suffix	Suffisso da utilizzare dopo il nome di ogni proprietà

6.3.11. Classe TOrderEditData

La classe TOrderEditData è utilizzata per modificare i parametri di un ordine già inviato a mercato reale ed in attesa di esecuzione.

6.3.11.1. Proprietà

```
Integer Quantity
```
Quantità di contratti dell'ordine

```
Double Price
```
Prezzo dell'ordine se il tipo di prezzo utilizzato è USER_DEFINED_PRICE

```
TLimitPrice PriceType
```
Tipo di prezzo da utilizzare per l'ordine

6.3.12. Classe TOrderData

Questa classe rappresenta un ordine

6.3.12.1. Proprietà

`Integer BasketSequence`
Valore di sequenza con cui inviare l'ordine in Paper Trading o a mercato reale

`Double BrokerCosts`
Costi commissionali associati all'ordine

`TOrderCompletedReason CompletedReason`
Ragione per la quale l'ordine è segnalato come completato

`TCorrectionType CorrectionType`
Tipo di correzione di hedging che ha generato l'ordine

`TDateTime CreationDateTime`
Data e ora di creazione dell'ordine

`TDateTime ExecDateTime`
Data e ora di esecuzione dell'ordine

`Double ExecPrice`
Prezzo di eseguito dell'ordine

`String Notes`
Note associate all'ordine

`OBJECT_ID ObjectID`
Codice identificativo dell'oggetto

`TOrderSource OrderSource`
Specifica quale sistema ha generato l'ordine

`TRTOrderStatus OrderStatus`
Stato dell'ordine

`Double PreviousMeanPrice`
Prezzo medio di carico dello strumento prima della generazione dell'ordine

`Integer PreviousQuantity`
Quantità in carico dello strumento prima della generazione dell'ordine

`TLimitPrice PriceType`
Tipo di prezzo di invio dell'ordine

`Integer Quantity`
Quantità dell'ordine

`TSecurity Security`
Strumento a cui l'ordine è associato

`Double UnderlyingPrice`
Prezzo del sottostante dello strumento associato all'ordine al momento dell'esecuzione

```
Boolean WhatIf
```
Specifica se l'ordine è un ordine generato su una strategia di valutazione What-If

6.3.12.2. Metodi

```
Sub CalcBorkerCosts()
```
Calcola i costi commissionali dell'ordine

```
Function Cancel() As Boolean
```
Annulla l'ordine se inviato a mercato reale e non ancora eseguito

```
Sub CopyFrom(ByVal source As TOrderData)
```
Copia tutte le proprietà da un altro oggetto di tipo TOrderData

source	Oggetto da cui copiare le proprietà

```
Function Create(ByVal security As TSecurity, ByVal quantity As Integer,
Optional ByVal  price As Double = 0, Optional ByVal  orderSource As
TOrderSource = ORDER_SOURCE_MANUAL) As TOrderData
```
Crea un nuovo oggetto di tipo TOrderData

security	Strumento a cui il nuovo ordine sarà associato
quantity	Quantità dell'ordine
price	Prezzo dell'ordine
orderSource	Identifica quale sistema ha creato il nuovo ordine

```
Function Edit(ByVal quantity As Integer, ByVal type As TLimitPrice,
ByVal price As Double) As Boolean
```
Modifica l'ordine inviato a mercato reale ma non ancora eseguito

quantity	Nuova quantità dell'ordine
type	Nuovo tipo di prezzo dell'ordine
price	Nuovo prezzo dell'ordine

```
Function GetByID(ByVal id As OBJECT_ID) As TOrderData
```
[Class Method] Recupera un oggetto di tipo TOrderData in base al suo codice identificativo

id	Codice identificativo dell'oggetto da recuperare

```
Function GetRealized() As Double
```
Calcola il Profit/Loss consolidato associato all'ordine

```
Function GetRequestQuantity() As Integer
```
Ottiene la quantità iniziale dell'ordine

```
Function GetRequestUserPrice() As Double
```
Ottiene il prezzo di invio iniziale dell'ordine

```
Function IsComplete() As Boolean
```
Calcola se l'ordine è completato oppure no

```
Function Place() As Boolean
```
Invia l'ordine a mercato reale

```
Function PlacePaperTrading() As Boolean
```
Invia l'ordine in Paper Trading

```
Function ToString() As String
```
Ottiene una stringa descrittiva dell'ordine

6.3.13. Classe TPrice
Questa classe rappresenta un prezzo con possibilità di aggiornamento in tempo reale, quale Bid, Ask, Last o Riferimento.

6.3.13.1. Proprietà
```
String DDEItem
```
Item DDE per ricevere prezzi aggiornati in tempo reale tramite DDE

```
Integer DDELinkID
```
Codice identificativo univoco della connessione DDE

```
String DDEService
```
Service DDE per ricevere prezzi aggiornati in tempo reale tramite DDE

```
String DDETopic
```
Topic DDE per ricevere prezzi aggiornati in tempo reale tramite DDE

```
TDDELinkType LinkType
```
Tipo di link DDE associato a questo oggetto

```
Double MinimumTick
```
Valore minimo del tick

```
OBJECT_ID ObjectID
```
Codice identificativo dell'oggetto

```
TSecurity Owner
```
Strumento al quale questo oggetto appartiene

```
Integer TickCounter
```
Numero di tick ricevuti dal mercato

6.3.13.2. Metodi
```
Sub CopyFrom(ByVal source As TPrice)
```
Copia tutte le proprietà da un altro oggetto di tipo TPrice
source Oggetto da cui copiare le proprietà

```
Function Create(ByVal owner As TSecurity, ByVal type As TDDELinkType) As
TPrice
```
[Class Method] Crea un nuovo oggetto di tipo TPrice
owner Strumento a cui il prezzo appartiene
type Tipo di link DDE associato al prezzo

```
Function GetCanHighlight(ByVal property As String) As Boolean
```
Indica se la proprietà specificata supporta l'evidenziazione della variazione di valore
property Proprietà

```
Function GetLast() As Double
```
Ottiene il valore di prezzo più aggiornato

```
Function GetLastUpdateTime() As Integer
```
Ottiene il timestamp dell'ultimo aggiornamento di prezzo ricevuto. Questo valore è comparabile con il valore ritornato dalla API GetTickCount() di Windows, ed è espresso in millesimi di secondo.

```
Function GetPreviousValue() As Double
```
Ottiene il valore di prezzo immediatamente precedente a quello attuale

```
Function GetPriceHighlightStatus() As THighlightStatus
```
Ottienen lo stato di evidenziazione del prezzo

```
Function GetProperty(ByVal property As String) As String
```
Ottiene il valore della proprietà specificata

property	Proprietà

```
Function GetPropertyHighlightStatus(ByVal property As String) As
THighlightStatus
```
Ottiene lo stato di evidenziazione della proprietà specificata

property	Proprietà

```
Sub GetPropertyList(ByVal list As TStringList, Optional ByVal clearList
As Boolean = false, Optional ByVal prefix As String = "", Optional ByVal
suffix As String = "")
```
[Class Method] Ottiene l'elenco dell proprietà disponibili per gli oggetti della classe TPrice

list	Lista di stringhe dove scrivere l'elenco delle proprietà disponibili
clearList	Indica se azzerare la lista prima di scrivere l'elenco delle proprietà
Prefix	Prefisso da utilizzare prima del nome di ogni proprietà
Suffix	Suffisso da utilizzare dopo il nome di ogni proprietà

```
Function GetStatusColor() As TColor
```
Ottiene il colore relativo allo stato di aggiornamento del prezzo, con una sfumatura che parte dal verde chiaro, per un prezzo appena aggiornato, al rosso chiaro, per un prezzo non più aggiornato da diverso tempo

```
Function GetXAverage() As Double
```
Ottiene il valore della media mobile esponenziale calcolata sui tick ricevuti dal mercato

```
Function IsValid(ByRef reason As String) As Boolean
```
Verifica se il prezzo è da considerarsi come un prezzo valido, in base alle impostazioni della strategia

reason	[Output] Contiene il testo che specifica per quale motivo il prezzo non è da considerarsi valido

```
Sub SetLast(ByVal price As Double)
```
Imposta un nuovo valore per il prezzo

```
Sub SetStrategySettings(ByVal settings As TStrategySettings)
```
Imposta l'oggetto TStrategySettings da utilizzare come impostazioni della strategia

```
Sub StartDDE()
```
Avvia la connessione DDE per ricevere i prezzi aggiornati dal mercato

```
Sub StopDDE()
```
Spegne la connessione DDE

6.3.14. Classe TSecurity
La classe TSecurity rappresenta la classe base dalla quale poi derivano le classi TUnderlying e TOption. Un oggetto TSecurity consiste in uno qualsiasi degli strumenti che compongono la strategia tra sottostanti, future ed opzioni.

6.3.14.1. Proprietà
```
TDateTime AnalisysDate
```
Data e ora di analisi per tutti i calcoli che necessitano di un valore temporale accoppiato alla eventuale data di scadenza dello strumento. Normalmente a questa proprietà va ssegnato il valore restituito dalla funzione Now().

```
Double Ask
```
Prezzo Ask

```
TPrice AskObject
```
[Read-Only] Oggetto di tipo TPrice utilizzato per il prezzo Ask

```
Double AtNow
```
[Read-Only] Totale AtNow dello strumento, assimilabile al profit/loss della open position

```
Double Bid
```
Prezzo Bid

```
TPrice BidObject
```
[Read-Only] Oggetto di tipo TPrice utilizzato per il prezzo Bid

```
Double BrokerCosts
```
[Read-Only] Totale delle spese commissionali sostenuto sullo strumento

```
TBrokerCosts BrokerCostsObject
```
[Read-Only] Oggetto di tipo TBrokerCosts utilizzato per il calcolo delle spese commissionali per lo strumento

```
Double Commitment
```
[Read-Only] Commitment della open position dello strumento

```
Double CurrentPrice
```
[Read-Only] Prezzo utilizzato per i calcoli della open position dello strumento. FiutoPRO utilizza sempre il prezzo più svantaggioso possibile per i calcoli relativi alla open position.

```
TPrice CurrentPriceObject
```
[Read-Only] Oggetto di tipo TPrice corrispondente al prezzo utilizzato per i calcoli della open position dello strumento

```
Double DaysToExpiry
```
[Read-Only] Giorni a scadenza dello strumento. Il valore può anche essere frazionario.

```
Integer DecimalDigits
```
Numero di cifre decimali utilizzate per visualizzare i prezzi dello strumento

```
Double Delta
```
[Read-Only] Delta dello strumento

```
Double DeltaPercent
```
[Read-Only] Delta per 1% dello strumento

```
TDateTime ExpiryDate
```
Data e ora di scadenza dello strumento

```
Double Gamma
```
[Read-Only] Gamma dello strumento

```
Double GammaPercent
```
[Read-Only] Gamma per 1% dello strumento

```
Boolean HasOrdersInBasket
```
[Read-Only] Indica se lo strumento ha ordini pendenti nel basket della strategia

```
Boolean HasRT
```
[Read-Only] Indica se lo strumento ha almeno una connessione attiva per ricevere aggiornamenti di prezzo in tempo reale

```
Integer HedgeQuantity
```
Quantità da acquistare o vendere per lo strumento suggerita dal sistema automatico di hedging per la copertura

```
Boolean IsExpired
```
[Read-Only] Indica se lo strumento è scaduto

```
String Isin
```
Codice isin dello strumento

```
Double Last
```
Prezzo Last dello strumento

```
TPrice LastObject
```
[Read-Only] Oggetto di tipo TPrice utilizzato per il prezzo Last

```
Double LotSize
```
Dimensione del lotto del contratto descritto dallo strumento

```
String MarketSymbol
```
[Read-Only] Stringa di testo che rappresenta il simbolo utilizzato nella connessione real-time per i prezzi Ask, Bid e Last. Questo valore cambia da strumento a strumento ed anche in base al broker utilizzato. Inoltre, nell'invio di ordini a mercato, essi verranno inviati al broker usando questo valore per identificare lo strumento.

```
Double MaximumGain
```
[Read-Only] Valore di massimo guadagno raggiunto dalla open position attuale

```
String Name
```
Nome dello strumento

```
Integer NumTrades
```
[Read-Only] Numero di trades eseguiti sullo strumento

```
OBJECT_ID ObjectID
```
Codice identificativo dello strumento

```
Integer OrdersRTLink
```
[Read-Only] Codice identificativo della connessione real-time tra FiutoPRO ed il broker che sarà utilizzato per l'invio degli ordini a mercato

```
TPayoffLineProperties PayoffLineProperties
```
[Read-Only] Insieme delle proprietà che descrivono come disegnare la linea relativa allo strumento nel grafico di payoff della strategia

```
Double PointValue
```
Point Value dello strumento

```
Double PortfolioDelta
```
[Read-Only] Delta di portafoglio dello strumento

```
Double PortfolioGamma
```
[Read-Only] Gamma di portafoglio dello strumento

```
Double PortfolioPrice
```
[Read-Only] Prezzo medio di carico della open position dello strumento

```
Double PortfolioRho
```
[Read-Only] Rho di portafoglio dello strumento

```
Double PortfolioTheta
```
[Read-Only] Theta di portafoglio dello strumento

```
Double PortfolioVega
```
[Read-Only] Vega di portafoglio dello strumento

```
Integer Quantity
```
[Read-Only] Quantità di contratti aperti nella open position attuale dello strumento

```
String RealIsin
```
[Read-Only] Codice isin dello strumento sottostante in caso di future

```
Double RealizedPL
```
[Read-Only] Profit/Loss consolidato dello strumento

```
String ReferenceIsin
```
Codice isin relativo allo strumento di riferimento

```
Double Rho
```
[Read-Only] Rho dello strumento

```
Double RhoPercent
```
[Read-Only] Rho per 1% dello strumento

```
Integer RTLink
```
[Read-Only] Codice identificativo della connessione real-time utilizzata per ricevere gli aggiornamenti di prezzo su Ask, Bid e Last

```
Double Settlement
```
Prezzo di Settlement per lo strumento

```
Double Spread
```
[Read-Only] Spread Ask-Bid dello strumento

```
TStrategySettings StrategySettings
```
Impostazioni della strategia a cui lo strumento appartiene

```
TSymbolType SymbolType
```
Tipo di simbolo associato allo strumento

```
Double Theta
```
[Read-Only] Theta dello strumento

```
Double ThetaPercent
```
[Read-Only] Theta per 1% dello strumento

```
Double Tick
```
Scostamento di prezzo minimo per lo strumento. Assegnare il valore corretto prima di inviare ordini al mercato.

```
Double TotalCredit
```
[Read-Only] Totale incassato nei vari trades eseguiti sullo strumento. Può assumere un valore negativo.

```
Double TotalQuantity
```
[Read-Only] Quantità totale della open position dello strumento

```
String UnderlyingIsin
```
Codice isin dello strumento sottostante

```
TUsedPrice UsedPrice
```
Identifica quale tipo di prezzo è utilizzato per tutti i calcoli relativi alla open position dello strumento. FiutoPRO utilizza sempre il prezzo più svantaggioso possibile nei calcoli relativi alla open position.

```
Double Vega
```
[Read-Only] Vega dello strumento

```
Double VegaPercent
```
[Read-Only] Vega per 1% dello strumento

6.3.14.2. Metodi

```
Function AddOrder(ByVal quantity As Integer, Optional ByVal price As
Double = MARKET, Optional ByVal orderSource As TOrderSource =
ORDER_SOURCE_SCRIPT) As TOrderData
```
Genera un nuovo ordine per lo strumento

quantity Numero di contratti da acquistare (positivo) o vendere (negativo)
price Prezzo dell'ordine o tipo di prezzo dell'ordine
orderSource Identifica il sistema che ha generato l'ordine

```
Function Buy(ByVal quantity As Integer, Optional ByVal price As Double =
MARKET) As TOrderData
```

Genera un nuovo ordine di acquisto per lo strumento
quantity Numero di contratti da acquistare
price Prezzo dell'ordine o tipo di prezzo dell'ordine

```
Sub ClearDDE()
```

Azzera tutte le informazioni relative alle connessioni DDE

```
Sub ClearOrders()
```

Elimina tutti gli ordini eseguiti e pendenti sullo strumento

```
Sub CopyFrom(ByVal source As TSecurity)
```

Copia tutte le proprietà da un altro oggetto di tipo TSecurity
source Oggetto di cui copiare le proprietà

```
Sub Flat()
```

Genera un nuovo ordine di acquisto o vendita per chiudere la open position attuale dello strumento a prezzo MARKET

```
Function GetByID(ByVal id As OBJECT_ID) As TSecurity
```

[Class Method] Restituisce l'oggetto TSecurity il cui codice identificativo è pari a quello specificato
id Codice identificato dell'oggetto da restituire

```
Function GetLimitPrice(ByVal price As Double, ByVal quantity As Integer)
As Double
```

Restituisce il prezzo calcolato in base ai parametri specificati
price Tipo di prezzo (TLimitPrice) oppure prezzo specificato dall'utente
quantity Quantità da utilizzare per calcolare il prezzo

```
Function GetNowPayoffAt(ByVal price As Double, Optional ByVal
showDividends As Boolean = true) As Double
```

Calcola il valore di payoff AtNow dello strumento al prezzo specificato
price Prezzo del sottostante a cui calcolare il payoff dello strumento
showDividends Indica se devono essere utilizzati gli eventuali dividendi del sottostante nel calcolo del valore di payoff

```
Function GetOrderByIndex(ByVal orderID As OBJECT_ID) As TOrderData
```

Restituisce l'ordine sullo strumento corrispondente al codice identificato specificato
orderID Codice identificativo dell'ordine da restituire

```
Function GetPayoffAt(ByVal price As Double, Optional ByVal expiryDate As
TDateTime = -1, Optional ByVal showDividends As Boolean = true) As
Double
```

Calcola il payoff al prezzo ed alla data di scadenza specificata.
price Prezzo del sottostante a cui calcolare il payoff dello strumento
expiryDate Data di scadenza a cui calcolare il payoff dello strumento

showDividends Indica se devono essere utilizzati gli eventuali dividendi del sottostante nel calcolo del valore di payoff

```
Function GetProperty(ByVal property As String) As String
```

Ottiene il valore della proprietà specificata
property Proprietà

```
Sub GetPropertyList(ByVal list As TStringList, Optional ByVal clearList
As Boolean = false, Optional ByVal prefix As String = "", Optional ByVal
suffix As String = "")
```

[Class Method] Ottiene l'elenco dell proprietà disponibili per gli oggetti della classe TSecurity
list Lista di stringhe dove scrivere l'elenco delle proprietà disponibili
clearList Indica se azzerare la lista prima di scrivere l'elenco delle proprietà
Prefix Prefisso da utilizzare prima del nome di ogni proprietà
Suffix Suffisso da utilizzare dopo il nome di ogni proprietà

```
Function IsValid(ByRef reason As String) As Boolean
```

Verifica se i prezzi dello strumento sono da considerare come prezzi validi, in base alle impostazioni della strategia
reason [Output] Contiene il testo che specifica quale prezzo e per quale motivo non è da considerarsi valido, o una stringa vuota se tutti i prezzi sono validi

```
Sub RecalcPortfolioPrice()
```

Ricalcola il prezzo medio di carico della open position attuale dello strumento

```
Sub RemoveOrder(ByVal order As TOrderData)
```

Elimina un ordine dai trades dello strumento
order Oggetto di tipo TOrderData da eliminare

```
Sub RemoveOrderByID(ByVal orderID As OBJECT_ID)
```

Elimina un ordine dai trades dello strumento
orderID Codice identificativo dell'ordine da rimuovere

```
Function Sell(ByVal quantity As Integer, Optional ByVal price As Double
= MARKET) As TOrderData
```

Genera un nuovo ordine di vendita per lo strumento
quantity Numero di contratti da vendere
price Prezzo dell'ordine o tipo di prezzo dell'ordine

```
Sub StartDDE()
```

Avvia le connessioni DDE per ricevere gli aggiornamenti di prezzo real-time

```
Sub StartRT()
```

Avvia la connessione real-time per ricevere gli aggiornamenti di prezzo dal broker

```
Sub StartRTCustom()
```

Avvia la connessione real-time per ricevere gli aggiornamenti di prezzo dal broker usando impostazioni personalizzate dall'utente

```
Sub StopDDE()
```

Spegne le connessioni DDE

```
Sub StopRT()
```
Spegne la connessione real-time

6.3.15. Classe TUnderlying
Questa classe deriva dalla classe TSecurity e rappresenta i sottostanti (azioni, futures o indici) che fanno parte della strategia di FiutoPRO.

6.3.15.1. Proprietà
```
TDateTime AnalisysDate
```
Data e ora di analisi per tutti i calcoli che necessitano di un valore temporale accoppiato alla eventuale data di scadenza dello strumento. Normalmente a questa proprietà va ssegnato il valore restituito dalla funzione Now().

```
Double Ask
```
Prezzo Ask

```
TPrice AskObject
```
[Read-Only] Oggetto di tipo TPrice utilizzato per il prezzo Ask

```
Double AtNow
```
[Read-Only] Totale AtNow dello strumento, assimilabile al profit/loss della open position

```
Double Bid
```
Prezzo Bid

```
TPrice BidObject
```
[Read-Only] Oggetto di tipo TPrice utilizzato per il prezzo Bid

```
Double BrokerCosts
```
[Read-Only] Totale delle spese commissionali sostenuto sullo strumento

```
TBrokerCosts BrokerCostsObject
```
[Read-Only] Oggetto di tipo TBrokerCosts utilizzato per il calcolo delle spese commissionali per lo strumento

```
Double Commitment
```
[Read-Only] Commitment della open position dello strumento

```
Double CurrentPrice
```
[Read-Only] Prezzo utilizzato per i calcoli della open position dello strumento. FiutoPRO utilizza sempre il prezzo più svantaggioso possibile per i calcoli relativi alla open position.

```
TPrice CurrentPriceObject
```
[Read-Only] Oggetto di tipo TPrice corrispondente al prezzo utilizzato per i calcoli della open position dello strumento

```
Double DaysToExpiry
```
[Read-Only] Giorni a scadenza dello strumento. Il valore può anche essere frazionario.

```
Integer DecimalDigits
```
Numero di cifre decimali utilizzate per visualizzare i prezzi dello strumento

```
Double Delta
```
[Read-Only] Delta dello strumento

```
Double DeltaPercent
```
[Read-Only] Delta per 1% dello strumento

```
Double ETFRatio
```
Rapporto di moltiplicazione tra lo strumento attuale ed il sottostante corrispondente. Questa proprietà è usata per gli strumenti ETF usati nell'hedging automatico. Per tutti gli altri casi è consigliabile impostare questa proprietà ad 1.

```
TDateTime ExpiryDate
```
Data e ora di scadenza dello strumento

```
Double FutureK
```
Valore utilizzato nel calcolo del valore del future a partire dal valore dell'indice/azione sottostante e nel calcolo dei dividendi teorici.

```
Double Gamma
```
[Read-Only] Gamma dello strumento

```
Double GammaPercent
```
[Read-Only] Gamma per 1% dello strumento

```
Boolean HasOrdersInBasket
```
[Read-Only] Indica se lo strumento ha ordini pendenti nel basket della strategia

```
Boolean HasRT
```
[Read-Only] Indica se lo strumento ha almeno una connessione attiva per ricevere aggiornamenti di prezzo in tempo reale

```
Integer HedgeQuantity
```
Quantità da acquistare o vendere per lo strumento suggerita dal sistema automatico di hedging per la copertura

```
Double HedgeWeightPercent
```
Percentuale di pesatura dello strumento nel calcolo del sistema automatico di hedging

```
Double HistoricalVolatility
```
Volatilià storica dello strumento espressa in percentuale

```
Boolean IsExpired
```
[Read-Only] Indica se lo strumento è scaduto

```
String Isin
```
Codice isin dello strumento

```
Double Last
```
Prezzo Last dello strumento

```
TPrice LastObject
```
[Read-Only] Oggetto di tipo TPrice utilizzato per il prezzo Last

```
Integer LinkedOptions
```
Numero totale di opzioni della strategia il cui sottostante è questo strumento

```
Double LotSize
```
Dimensione del lotto del contratto descritto dallo strumento

```
String MarketSymbol
```
[Read-Only] Stringa di testo che rappresenta il simbolo utilizzato nella connessione real-time per i prezzi Ask, Bid e Last. Questo valore cambia da strumento a strumento ed anche in base al broker utilizzato. Inoltre, nell'invio di ordini a mercato, essi verranno inviati al broker usando questo valore per identificare lo strumento.

```
Double MaximumGain
```
[Read-Only] Valore di massimo guadagno raggiunto dalla open position attuale

```
String Name
```
Nome dello strumento

```
Integer NumTrades
```
[Read-Only] Numero di trades eseguiti sullo strumento

```
OBJECT_ID ObjectID
```
Codice identificativo dello strumento

```
Integer OrdersRTLink
```
[Read-Only] Codice identificativo della connessione real-time tra FiutoPRO ed il broker che sarà utilizzato per l'invio degli ordini a mercato

```
TPayoffLineProperties PayoffLineProperties
```
[Read-Only] Insieme delle proprietà che descrivono come disegnare la linea relativa allo strumento nel grafico di payoff della strategia

```
Boolean PayoffModeFuture
```
Indica se il grafico di payoff dello strumento deve considerare il tipo di simbolo se questo è un future. Il grafico di payoff di un future non è una linea retta, ma una curva logaritmica/esponenziale.

```
Double PointValue
```
Point Value dello strumento

```
Double PortfolioDelta
```
[Read-Only] Delta di portafoglio dello strumento

```
Double PortfolioGamma
```
[Read-Only] Gamma di portafoglio dello strumento

```
Double PortfolioPrice
```
[Read-Only] Prezzo medio di carico della open position dello strumento

```
Double PortfolioRho
```
[Read-Only] Rho di portafoglio dello strumento

`Double PortfolioTheta`
[Read-Only] Theta di portafoglio dello strumento

`Double PortfolioVega`
[Read-Only] Vega di portafoglio dello strumento

`Integer Quantity`
[Read-Only] Quantità di contratti aperti nella open position attuale dello strumento

`String RealIsin`
[Read-Only] Codice isin dello strumento sottostante in caso di future

`Double RealizedPL`
[Read-Only] Profit/Loss consolidato dello strumento

`Double Reference`
Prezzo di riferimento dello strumento

`String ReferenceIsin`
Codice isin relativo allo strumento di riferimento

`String ReferenceMarketSymbol`
[Read-Only] Stringa di testo che rappresenta il simbolo utilizzato nella connessione real-time per il prezzo di riferimento. Questo valore cambia da strumento a strumento ed anche in base al broker utilizzato.

`TPrice ReferenceObject`
[Read-Only] Oggetto di tipo TPrice utilizzato per il prezzo di riferimento

`Double Rho`
[Read-Only] Rho dello strumento

`Double RhoPercent`
[Read-Only] Rho per 1% dello strumento

`Integer RTLink`
[Read-Only] Codice identificativo della connessione real-time utilizzata per ricevere gli aggiornamenti di prezzo su Ask, Bid e Last

`Integer RTLinkReference`
[Read-Only] Codice identificativo della connessione real-time utilizzata per ricevere gli aggiornamenti del prezzo di riferimento

`Double Settlement`
Prezzo di Settlement per lo strumento

`Double Spread`
[Read-Only] Spread Ask-Bid dello strumento

`TStrategySettings StrategySettings`
Impostazioni della strategia a cui lo strumento appartiene

`TSymbolType SymbolType`
Tipo di simbolo associato allo strumento

`Double TheoreticalDividends`
Dividendi teorici calcolati sullo strumento fino alla sua scadenza

`Double TheoreticalIndex`
[Read-Only] Valore teorico dell'indice/azione sottostante calcolato in base al valore del future

`Double TheoreticalIndexAtExpiry`
[Read-Only] Valore teorico dell'indice/azione sottostante calcolato in base al valore del future alla data di scadenza del future stesso

`Double Theta`
[Read-Only] Theta dello strumento

`Double ThetaPercent`
[Read-Only] Theta per 1% dello strumento

`Double Tick`
Scostamento di prezzo minimo per lo strumento. Assegnare il valore corretto prima di inviare ordini al mercato.

`Double TotalCredit`
[Read-Only] Totale incassato nei vari trades eseguiti sullo strumento. Può assumere un valore negativo.

`Double TotalQuantity`
[Read-Only] Quantità totale della open position dello strumento

`String UnderlyingIsin`
Codice isin dello strumento sottostante

`TUsedPrice UsedPrice`
Identifica quale tipo di prezzo è utilizzato per tutti i calcoli relativi alla open position dello strumento. FiutoPRO utilizza sempre il prezzo più svantaggioso possibile nei calcoli relativi alla open position.

`Boolean UseReferencePrice`
Indica se per lo strumento deve essere utilizzato un prezzo di riferimento diverso dal prezzo Last

`Double Vega`
[Read-Only] Vega dello strumento

`Double VegaPercent`
[Read-Only] Vega per 1% dello strumento

6.3.15.2. Metodi

```
Function AddOrder(ByVal quantity As Integer, Optional ByVal price As
Double = MARKET, Optional ByVal orderSource As TOrderSource =
ORDER_SOURCE_SCRIPT) As TOrderData
```

Genera un nuovo ordine per lo strumento

quantity	Numero di contratti da acquistare (positivo) o vendere (negativo)
price	Prezzo dell'ordine o tipo di prezzo dell'ordine
orderSource	Identifica il sistema che ha generato l'ordine

```
Function AssertHistory(ByVal timeFrame As TRTHistoryTimeFrame) As
THistoryDataContext
```

Aggiunge se non è già presente un contesto di dati storici allo strumento con il Time Frame specificato

timeFrame	Time Frame del contesto di dati storici da aggiungere

```
Function Buy(ByVal quantity As Integer, Optional ByVal price As Double =
MARKET) As TOrderData
```

Genera un nuovo ordine di acquisto per lo strumento

quantity	Numero di contratti da acquistare
price	Prezzo dell'ordine o tipo di prezzo dell'ordine

```
Sub CalcProperties()
```

Esegue un ricalcolo completo di tutte le proprietà dello strumento

```
Sub ClearDDE()
```

Azzera tutte le informazioni relative alle connessioni DDE

```
Sub ClearOrders()
```

Elimina tutti gli ordini eseguiti e pendenti sullo strumento

```
Sub CopyFrom(ByVal source As TUnderlying, Optional ByVal withHistory As
Boolean = true)
```

Copia tutte le proprietà da un altro oggetto di tipo TUnderlying

source	Oggetto di cui copiare le proprietà
withHistory	Specifica se deve copiare anche tutti i contesti di dati storici e relativi indicatori

```
Sub Flat()
```

Genera un nuovo ordine di acquisto o vendita per chiudere la open position attuale dello strumento a prezzo MARKET

```
Function GetLimitPrice(ByVal price As Double, ByVal quantity As Integer)
As Double
```

Restituisce il prezzo calcolato in base ai parametri specificati

price	Tipo di prezzo (TLimitPrice) oppure prezzo specificato dall'utente
quantity	Quantità da utilizzare per calcolare il prezzo

```
Function GetNetPrice(ByVal date As TDateTime) As Double
```

Restituisce il prezzo del sottostante al netto dei dividendi staccati tra la data di analisi attuale e la data specificata

date	Data finale di calcolo dei dividendi

```
Function GetNowPayoffAt(ByVal price As Double, Optional ByVal
showDividends As Boolean = true) As Double
```

Calcola il valore di payoff AtNow dello strumento al prezzo specificato

price Prezzo del sottostante a cui calcolare il payoff dello strumento

showDividends Indica se devono essere utilizzati gli eventuali dividendi del sottostante nel calcolo del valore di payoff

```
Sub GetObjectPropertyList(ByVal list As TStringList, Optional ByVal
clearList As Boolean = false, Optional ByVal prefix As String = "",
Optional ByVal suffix As String = "")
```

Ottiene l'elenco delle proprietà disponibili per il sottostante, compresi eventuali contesti di dati storici

list Lista di stringhe dove scrivere l'elenco delle proprietà disponibili

clearList Indica se azzerare la lista prima di scrivere l'elenco delle proprietà

Prefix Prefisso da utilizzare prima del nome di ogni proprietà

Suffix Suffisso da utilizzare dopo il nome di ogni proprietà

```
Function GetOrderByIndex(ByVal orderID As OBJECT_ID) As TOrderData
```

Restituisce l'ordine sullo strumento corrispondente al codice identificato specificato

orderID Codice identificativo dell'ordine da restituire

```
Function GetPayoffAt(ByVal price As Double, Optional ByVal expiryDate As
TDateTime = -1, Optional ByVal showDividends As Boolean = true) As
Double
```

Calcola il payoff al prezzo ed alla data di scadenza specificata.

price Prezzo del sottostante a cui calcolare il payoff dello strumento

expiryDate Data di scadenza a cui calcolare il payoff dello strumento

showDividends Indica se devono essere utilizzati gli eventuali dividendi del sottostante nel calcolo del valore di payoff

```
Function GetProperty(ByVal property As String) As String
```

Ottiene il valore della proprietà specificata

property Proprietà

```
Sub GetPropertyList(ByVal list As TStringList, Optional ByVal clearList
As Boolean = false, Optional ByVal prefix As String = "", Optional ByVal
suffix As String = "")
```

[Class Method] Ottiene l'elenco delle proprietà disponibili per gli oggetti della classe TUnderlying

list Lista di stringhe dove scrivere l'elenco delle proprietà disponibili

clearList Indica se azzerare la lista prima di scrivere l'elenco delle proprietà

Prefix Prefisso da utilizzare prima del nome di ogni proprietà

Suffix Suffisso da utilizzare dopo il nome di ogni proprietà

```
Function IsValid(ByRef reason As String) As Boolean
```

Verifica se i prezzi dello strumento sono da considerare come prezzi validi, in base alle impostazioni della strategia

reason [Output] Contiene il testo che specifica quale prezzo e per quale motivo non è da considerarsi valido, o una stringa vuota se tutti i prezzi sono validi

```
Sub RecalcPortfolioPrice()
```

Ricalcola il prezzo medio di carico della open position attuale dello strumento

```
Sub RemoveHistory(ByVal timeFrame As TRTHistoryTimeFrame)
```
Elimina il contesto di dati storici associato allo strumento con il time frame specificato
timeFrame Time Frame del contesto di dati storici da eliminare

```
Sub RemoveOrder(ByVal order As TOrderData)
```
Elimina un ordine dai trades dello strumento
order Oggetto di tipo TOrderData da eliminare

```
Sub RemoveOrderByID(ByVal orderID As OBJECT_ID)
```
Elimina un ordine dai trades dello strumento
orderID Codice identificativo dell'ordine da rimuovere

```
Function Sell(ByVal quantity As Integer, Optional ByVal price As Double
= MARKET) As TOrderData
```
Genera un nuovo ordine di vendita per lo strumento
quantity Numero di contratti da vendere
price Prezzo dell'ordine o tipo di prezzo dell'ordine

```
Sub StartDDE()
```
Avvia le connessioni DDE per ricevere gli aggiornamenti di prezzo real-time

```
Sub StartRT()
```
Avvia la connessione real-time per ricevere gli aggiornamenti di prezzo dal broker

```
Sub StartRTCustom()
```
Avvia la connessione real-time per ricevere gli aggiornamenti di prezzo dal broker usando impostazioni personalizzate dall'utente

```
Sub StartRTCustomReference()
```
Avvia la connessione real-time per ricevere gli aggiornamenti del solo prezzo di riferimento dal broker usando impostazioni personalizzate dall'utente

```
Sub StartRTFuture()
```
Avvia la connessione real-time per ricevere gli aggiornamenti di prezzo Ask, Bid e Last del future associato al sottostante dal broker usando impostazioni personalizzate dall'utente

```
Sub StopDDE()
```
Spegne le connessioni DDE

```
Sub StopRT()
```
Spegne la connessione real-time

```
Sub StopRTFuture()
```
Spegne la connessione real-time utilizzata per ricevere i prezzi Ask, Bid e Last del future associato al sottostante

```
Sub StopRTReference()
```
Spegne la connessione real-time utilizzata per ricevere il prezzo di riferimento

```
Function SumDividends(ByVal startDate As TDateTime, ByVal endDate As
TDateTime) As Double
```
Calcola l'ammontare totale dei dividendi staccati sul sottostante tra le date specificate
startDate Data iniziale del calcolo

endDate Data finale del calcolo

6.3.16. Classe TOption

Questa classe è utilizzata per rappresentare le opzioni che compongono la strategia di FiutoPRO. Deriva dalla classe TSecurity.

6.3.16.1. Proprietà

`TDateTime AnalisysDate`

Data e ora di analisi per tutti i calcoli che necessitano di un valore temporale accoppiato alla eventuale data di scadenza dello strumento. Normalmente a questa proprietà va ssegnato il valore restituito dalla funzione Now().

`Double Ask`

Prezzo Ask

`Double AskEMaker`

[Read-Only] Prezzo Ask eMaker

`Double AskMarket`

[Read-Only] Prezzo Ask ricevuto dal broker

`TPrice AskObject`

[Read-Only] Oggetto di tipo TPrice utilizzato per il prezzo Ask

`Double AskVolatility`

[Read-Only] Volatilità implicita calcolata sul prezzo Ask, espressa in percentuale

`Double AtNow`

[Read-Only] Totale AtNow dello strumento, assimilabile al profit/loss della open position

`Double Bid`

Prezzo Bid

`Double BidAskAverage`

[Read-Only] Media dei prezzi Ask e Bid

`Double BidEMaker`

[Read-Only] Prezzo Bid eMaker

`Double BidMarket`

[Read-Only] Prezzo Bid ricevuto dal broker

`TPrice BidObject`

[Read-Only] Oggetto di tipo TPrice utilizzato per il prezzo Bid

`Double BidVolatility`

[Read-Only] Volatilità implicita calcolata sul prezzo Bid, espressa in percentuale

`Double BreakEven`

[Read-Only] Prezzo del sottostante al quale il payoff dell'opzione raggiunge il valore 0

```
Double BreakEvenPercent
```
[Read-Only] Distanza in percentuale tra il prezzo attuale del sottostante dell'opzione ed il punto di Break-Even

```
Double BrokerCosts
```
[Read-Only] Totale delle spese commissionali sostenuto sullo strumento

```
TBrokerCosts BrokerCostsObject
```
[Read-Only] Oggetto di tipo TBrokerCosts utilizzato per il calcolo delle spese commissionali per lo strumento

```
TOptionCalc CalcObject
```
[Read-Only] Oggetto di tipo TOptionCalc utilizzato internamente per i calcoli teorici dell'opzione

```
Double Commitment
```
[Read-Only] Commitment della open position dello strumento

```
Double CurrentPrice
```
[Read-Only] Prezzo utilizzato per i calcoli della open position dello strumento. FiutoPRO utilizza sempre il prezzo più svantaggioso possibile per i calcoli relativi alla open position.

```
TPrice CurrentPriceObject
```
[Read-Only] Oggetto di tipo TPrice corrispondente al prezzo utilizzato per i calcoli della open position dello strumento

```
Double DaysToExpiry
```
[Read-Only] Giorni a scadenza dello strumento. Il valore può anche essere frazionario.

```
Integer DecimalDigits
```
Numero di cifre decimali utilizzate per visualizzare i prezzi dello strumento

```
Double Delta
```
[Read-Only] Delta dello strumento

```
Double DeltaExp2
```
[Read-Only] Delta esponenziale quadro

```
Double DeltaExp3
```
[Read-Only] Delta esponenziale cubico

```
Double DeltaPercent
```
[Read-Only] Delta per 1% dello strumento

```
Double DeltaSmartPro
```
[Read-Only] Delta calcolato con l'algoritmo SmartPRO

```
Double Elasticity
```
[Read-Only] Elasticità dell'opzione, a volte chiamato Lambda

```
Double EMaker
```
[Read-Only] Prezzo eMaker dell'opzione

```
Double EMakerVolatility
```
[Read-Only] Volatilità implicita utilizzata per calcolare il prezzo eMaker dell'opzione

```
TDateTime ExpiryDate
```
Data e ora di scadenza dello strumento

```
Double Gamma
```
[Read-Only] Gamma dello strumento

```
Double GammaPercent
```
[Read-Only] Gamma per 1% dello strumento

```
Boolean HasOrdersInBasket
```
[Read-Only] Indica se lo strumento ha ordini pendenti nel basket della strategia

```
Boolean HasRT
```
[Read-Only] Indica se lo strumento ha almeno una connessione attiva per ricevere aggiornamenti di prezzo in tempo reale

```
Boolean HedgeEnable
```
Indica se l'opzione deve essere presa in considerazione dal sistema automatico di hedging

```
THedgePrice HedgePriceType
```
Tipo di prezzo da utilizzare durante il calcolo dal sistema di hedging automatico

```
Integer HedgeQuantity
```
Quantità da acquistare o vendere per lo strumento suggerita dal sistema automatico di hedging per la copertura

```
Boolean IsExpired
```
[Read-Only] Indica se lo strumento è scaduto

```
String Isin
```
Codice isin dello strumento

```
Boolean IsTheoretical
```
[Read-Only] Indica se l'opzione è calcolata e visualizzata usando i prezzi teorici eMaker

```
Double Last
```
Prezzo Last dello strumento

```
TPrice LastObject
```
[Read-Only] Oggetto di tipo TPrice utilizzato per il prezzo Last

```
Double LastVolatility
```
[Read-Only] Volatilità implicita calcolata sul prezzo Last

```
Double LotSize
```
Dimensione del lotto del contratto descritto dallo strumento

```
Double MarketPressure
```
[Read-Only] Market Pressure dell'opzione calcolata sui tick ricevuti per i prezzi Bid ed Ask

```
String MarketSymbol
```
[Read-Only] Stringa di testo che rappresenta il simbolo utilizzato nella connessione real-time per i prezzi Ask, Bid e Last. Questo valore cambia da strumento a strumento ed anche in base al broker utilizzato. Inoltre, nell'invio di ordini a mercato, essi verranno inviati al broker usando questo valore per identificare lo strumento.

```
Double MaximumGain
```
[Read-Only] Valore di massimo guadagno raggiunto dalla open position attuale

```
TMoneyness Moneyness
```
[Read-Only] Moneyness dell'opzione

```
String Name
```
Nome dello strumento

```
Integer NumSteps
```
Numero di passi da utilizzare nel calcolo inverso delle volatilità implicite e nel calcolo diretto dei prezzi teorici eMaker nel caso di utilizzo di un metodo di calcolo binomiale

```
Integer NumTrades
```
[Read-Only] Numero di trades eseguiti sullo strumento

```
OBJECT_ID ObjectID
```
Codice identificativo dello strumento

```
Double OFFThreshold
```
Prezzo del sottostante di spegnimento dell'opzione utilizzata nel calcolo del sistema di hedging automatico quando il modo scelto è "A Soglie"

```
Double ONThreshold
```
Prezzo del sottostante di attivazione dell'opzione utilizzata nel calcolo del sistema di hedging automatico quando il modo scelto è "A Soglie"

```
TOptionCalcMethod OptionCalcMethod
```
Metodo di calcolo dei valori teorici eMaker

```
TOptionClass OptionClass
```
Stile dell'opzione

```
TOptionType OptionType
```
Tipo dell'opzione

```
Integer OrdersRTLink
```
[Read-Only] Codice identificativo della connessione real-time tra FiutoPRO ed il broker che sarà utilizzato per l'invio degli ordini a mercato

```
TPayoffLineProperties PayoffLineProperties
```
[Read-Only] Insieme delle proprietà che descrivono come disegnare la linea relativa allo strumento nel grafico di payoff della strategia

```
Double PointValue
```
Point Value dello strumento

```
Double PortfolioDelta
```
[Read-Only] Delta di portafoglio dello strumento

```
Double PortfolioGamma
```
[Read-Only] Gamma di portafoglio dello strumento

```
Double PortfolioPrice
```
[Read-Only] Prezzo medio di carico della open position dello strumento

```
Double PortfolioRho
```
[Read-Only] Rho di portafoglio dello strumento

```
Double PortfolioTheta
```
[Read-Only] Theta di portafoglio dello strumento

```
Double PortfolioVega
```
[Read-Only] Vega di portafoglio dello strumento

```
TOptionClass PredefinedOptionClass
```
Stile delle opzioni per il sottostante. FiutoPRO utilizza per tutti i calcoli di default lo stile Europeo. Questa proprietà rappresenta lo stile effettivo che l'opzione dovrebbe avere per ottenere una accuratezza del 100% nei calcoli teorici. La differenza ottenuta nei calcoli teorici usando i due differenti stili è in genere inferiore all'1%, a fronte di una complessità di calcolo molto maggiore con lo stile Americano, di circa 9 ordini di grandezza.

```
Integer Quantity
```
[Read-Only] Quantità di contratti aperti nella open position attuale dello strumento

```
String RealIsin
```
[Read-Only] Codice isin dello strumento sottostante in caso di future

```
Double RealizedPL
```
[Read-Only] Profit/Loss consolidato dello strumento

```
String ReferenceIsin
```
Codice isin relativo allo strumento di riferimento

```
Double Rho
```
[Read-Only] Rho dello strumento

```
Double RhoPercent
```
[Read-Only] Rho per 1% dello strumento

```
Double RiskFreeRate
```
Tasso di interesse privo di rischio

```
Integer RTLink
```
[Read-Only] Codice identificativo della connessione real-time utilizzata per ricevere gli aggiornamenti di prezzo su Ask, Bid e Last

```
Double Settlement
```
Prezzo di Settlement per lo strumento

```
Double Spread
```
[Read-Only] Spread Ask-Bid dello strumento

```
TStrategySettings StrategySettings
```
Impostazioni della strategia a cui lo strumento appartiene

```
Double Strike
```
Strike dell'opzione

```
Double StrikeStock
```
[Read-Only] Rapporto tra lo strike dell'opzione ed il prezzo del sottostante

```
TSymbolType SymbolType
```
Tipo di simbolo associato allo strumento

```
Double Theta
```
[Read-Only] Theta dello strumento

```
Double ThetaPercent
```
[Read-Only] Theta per 1% dello strumento

```
Boolean ThresholdActive
```
[Read-Only] Indica se l'opzione è attualmente attiva nel calcolo del sistema di hedging automatico quando il modo scelto è "A Soglie"

```
Double Tick
```
Scostamento di prezzo minimo per lo strumento. Assegnare il valore corretto prima di inviare ordini al mercato.

```
Double TimeValue
```
[Read-Only] Time Value dell'opzione

```
Double TotalCredit
```
[Read-Only] Totale incassato nei vari trades eseguiti sullo strumento. Può assumere un valore negativo.

```
Double TotalQuantity
```
[Read-Only] Quantità totale della open position dello strumento

```
TUnderlying Underlying
```
Oggetto che rappresenta il sottostante dell'opzione

```
String UnderlyingIsin
```
Codice isin dello strumento sottostante

```
Double UnderlyingPrice
```
Prezzo di riferimento del sottostante dell'opzione

```
TUsedPrice UsedPrice
```
Identifica quale tipo di prezzo è utilizzato per tutti i calcoli relativi alla open position dello strumento. FiutoPRO utilizza sempre il prezzo più svantaggioso possibile nei calcoli relativi alla open position.

```
Double Vega
```
[Read-Only] Vega dello strumento

```
Double VegaPercent
```
[Read-Only] Vega per 1% dello strumento

```
Double Volatility
```
Volatilità implicita dell'opzione

```
Double YearPrice
```
[Read-Only] Premio annualizzato dell'opzione

6.3.16.2. Metodi

```
Function AddOrder(ByVal quantity As Integer, Optional ByVal price As
Double = MARKET, Optional ByVal orderSource As TOrderSource =
ORDER_SOURCE_SCRIPT) As TOrderData
```
Genera un nuovo ordine per lo strumento

quantity	Numero di contratti da acquistare (positivo) o vendere (negativo)
price	Prezzo dell'ordine o tipo di prezzo dell'ordine
orderSource	Identifica il sistema che ha generato l'ordine

```
Function Buy(ByVal quantity As Integer, Optional ByVal price As Double =
MARKET) As TOrderData
```
Genera un nuovo ordine di acquisto per lo strumento

quantity	Numero di contratti da acquistare
price	Prezzo dell'ordine o tipo di prezzo dell'ordine

```
Sub CalcMoneyness()
```
Esegue un ricalcolo della moneyness dell'opzione

```
Sub CalcProperties()
```
Esegue un ricalcolo di tutte le proprietà dell'opzione che dipendono dai prezzi e dal tempo

```
Sub CalcVolatility()
```
Esegue un ricalcolo di tutte le volatilità implicite a partire dai prezzi

```
Sub ClearDDE()
```
Azzera tutte le informazioni relative alle connessioni DDE

```
Sub ClearOrders()
```
Elimina tutti gli ordini eseguiti e pendenti sullo strumento

```
Sub CopyFrom(ByVal source As TOption)
```
Copia tutte le proprietà da un altro oggetto di tipo TOption

source	Oggetto di cui copiare le proprietà

```
Sub Flat()
```
Genera un nuovo ordine di acquisto o vendita per chiudere la open position attuale dello strumento a prezzo MARKET

```
Function GetDeltaExp(ByVal exponent As Integer) As Double
```
Calcola il Delta esponenziale usando l'esponente specificato

exponent Esponente da usare nel calcolo del Delta esponenziale

```
Function GetLimitPrice(ByVal price As Double, ByVal quantity As Integer)
As Double
```

Restituisce il prezzo calcolato in base ai parametri specificati
price Tipo di prezzo (TLimitPrice) oppure prezzo specificato dall'utente
quantity Quantità da utilizzare per calcolare il prezzo

```
Function GetNowPayoffAt(ByVal price As Double, Optional ByVal
showDividends As Boolean = true) As Double
```

Calcola il valore di payoff AtNow dello strumento al prezzo specificato
price Prezzo del sottostante a cui calcolare il payoff dello strumento
showDividends Indica se devono essere utilizzati gli eventuali dividendi del sottostante nel calcolo del valore di payoff

```
Function GetOrderByIndex(ByVal orderID As OBJECT_ID) As TOrderData
```

Restituisce l'ordine sullo strumento corrispondente al codice identificato specificato
orderID Codice identificativo dell'ordine da restituire

```
Function GetPayoffAt(ByVal price As Double, Optional ByVal expiryDate As
TDateTime = -1, Optional ByVal showDividends As Boolean = true) As
Double
```

Calcola il payoff al prezzo ed alla data di scadenza specificata.
price Prezzo del sottostante a cui calcolare il payoff dello strumento
expiryDate Data di scadenza a cui calcolare il payoff dello strumento
showDividends Indica se devono essere utilizzati gli eventuali dividendi del sottostante nel calcolo del valore di payoff

```
Function GetProperty(ByVal property As String) As String
```

Ottiene il valore della proprietà specificata
property Proprietà

```
Sub GetPropertyList(ByVal list As TStringList, Optional ByVal clearList
As Boolean = false, Optional ByVal prefix As String = "", Optional ByVal
suffix As String = "")
```

[Class Method] Ottiene l'elenco dell proprietà disponibili per gli oggetti della classe TOption
list Lista di stringhe dove scrivere l'elenco delle proprietà disponibili
clearList Indica se azzerare la lista prima di scrivere l'elenco delle proprietà
Prefix Prefisso da utilizzare prima del nome di ogni proprietà
Suffix Suffisso da utilizzare dopo il nome di ogni proprietà

```
Function IsValid(ByRef reason As String) As Boolean
```

Verifica se i prezzi dello strumento sono da considerare come prezzi validi, in base alle impostazioni della strategia
reason [Output] Contiene il testo che specifica quale prezzo e per quale motivo non è da considerarsi valido, o una stringa vuota se tutti i prezzi sono validi

```
Sub RecalcPortfolioPrice()
```

Ricalcola il prezzo medio di carico della open position attuale dello strumento

```
Sub RemoveOrder(ByVal order As TOrderData)
```

Elimina un ordine dai trades dello strumento

order Oggetto di tipo TOrderData da eliminare

```
Sub RemoveOrderByID(ByVal orderID As OBJECT_ID)
```
Elimina un ordine dai trades dello strumento
orderID Codice identificativo dell'ordine da rimuovere

```
Function Sell(ByVal quantity As Integer, Optional ByVal price As Double
= MARKET) As TOrderData
```
Genera un nuovo ordine di vendita per lo strumento
quantity Numero di contratti da vendere
price Prezzo dell'ordine o tipo di prezzo dell'ordine

```
Sub SetDividends(ByVal dividends As TDividends)
```
Imposta l'oggetto di tipo TDividends da utilizzare per i calcoli relativi ai dividendi
dividends Oggetto di tipo TDividends da utilizza per i calcoli sui dividendi

```
Sub SetTheoreticalDividends(ByVal dividends As Double)
```
Imposta il valore dei dividendi teorici per l'opzione
dividends Valore dei dividendi teorici

```
Sub StartDDE()
```
Avvia le connessioni DDE per ricevere gli aggiornamenti di prezzo real-time

```
Sub StartRT()
```
Avvia la connessione real-time per ricevere gli aggiornamenti di prezzo dal broker

```
Sub StartRTCustom()
```
Avvia la connessione real-time per ricevere gli aggiornamenti di prezzo dal broker usando impostazioni personalizzate dall'utente

```
Sub StopDDE()
```
Spegne le connessioni DDE

```
Sub StopRT()
```
Spegne la connessione real-time

6.3.17. Classe TStrategyContext

La classe TStrategyContext rappresenta una strategia di FiutoPRO. In FPSS esiste sempre la variabile CurrentStrategy che rappresenta la strategia corrente di FiutoPRO. Una strategia in FiutoPRO è sempre composta da un insieme di sottostanti ed un insieme di opzioni sui sottostanti.

6.3.17.1. Proprietà

```
Double AtNow
```
[Read-Only] Totale AtNow della strategia

```
Double BrokerCosts
```
[Read-Only] Totale dei costi commissionali sostenuti sulla strategia

```
TStrategyChartInfo ChartSettings
```
[Read-Only] Insieme delle proprietà relative al grafico di payoff della strategia

```
TColor Color
```
Colore identificativo della strategia

```
Double Commitment
```
[Read-Only] Commitment totale della strategia

```
TConfidenceLevel ConfidenceLevel
```
Intervallo di confidenza della strategia da utilizzare nel calcolo del Value At Risk

```
Double Cost
```
[Read-Only] Costo complessivo sostenuto per la strategia. Se la strategia è a credito questa proprietà è negativa

```
Double CostToFlatAll
```
[Read-Only] Costo totale da sostenere per chiudere tutte le posizioni aperte della strategia

```
Double CostToHedge
```
[Read-Only] Costo totale da sostenere per eseguire le coperture calcolate dal sistema di hedging automatico

```
Double CurrentPortfolioValue
```
[Read-Only] Valore totale di portafoglio di tutte le posizioni aperte della strategia

```
Integer DaysToExpiry
```
[Read-Only] Giorni a scadenza

```
Double Delta
```
[Read-Only] Delta di portafoglio totale della strategia

```
Double DeltaCost
```
[Read-Only] Rapporto tra Delta e costo della strategia

```
Double DeltaPercent
```
[Read-Only] Delta di portafoglio per 1% della strategia

```
Double DeltaProtect
```
[Read-Only] Delta di portafoglio da coprire calcolato dal sistema di hedging automatico

```
Double DownsideBreakEven
```
[Read-Only] Downside Break-Even sul prezzo del sottostante della strategia

```
Double ExpectedProfit
```
[Read-Only] Profitto atteso della strategia

```
Double ExpectedProfitRisk
```
[Read-Only] Rapporto tra profitto atteso e massimo rischio della strategia

```
Double Gamma
```
[Read-Only] Gamma di portafoglio della strategia

```
Double GammaPercent
```
[Read-Only] Gamma di portafoglio per 1% della strategia

```
Double GammaProtect
```
[Read-Only] Gamma di portafoglio da coprire calcolato dal sistema di hedging automatico

```
Integer HoldingPeriod
```
Holding period della strategia da utilizzare nel calcolo del Value At Risk

```
Double KellyBetFraction
```
[Read-Only] Kelly Bet Fraction della strategia

```
TDateTime MaxExpiryDate
```
[Read-Only] Data di scadenza più lunga della strategia

```
Double MaxProfit
```
[Read-Only] Massimo profitto della strategia

```
Double MaxProfitCost
```
[Read-Only] Rapporto tra massimo profitto e costo della strategia

```
Double MaxProfitRisk
```
[Read-Only] Rapporto tra massimo profitto e massimo rischio della strategia

```
Double MaxRisk
```
[Read-Only] Massimo rischio della strategia

```
TDateTime MinExpiryDate
```
[Read-Only] Data di scadenza più corta della strategia

```
Double MontecarloMaxProfit
```
[Read-Only] Massimo profitto della strategia calcolato con il metodo MonteCarlo

```
Double MontecarloMaxRisk
```
[Read-Only] Massimo rischio della strategia calcolato con il metodo MonteCarlo

```
String Name
```
Nome della strategia

```
Double NetProfit
```
[Read-Only] Profit/Loss netto totale della strategia

```
TStringList Notes
```
[Read-Only] Note della strategia

```
Integer NumTrades
```
[Read-Only] Numero totale di trades eseguiti sulla strategia

```
OBJECT_ID ObjectID
```
Codice identificativo della strategia

```
Double Odds
```
[Read-Only] Odds della strategia

`TStringList OptionViewDetails`
[Read-Only] Elenco delle proprietà aggiuntive da visualizzare nella griglia delle opzioni della strategia

`Double PercentDownsideBreakEven`
[Read-Only] Distanza del downside Break-Even rispetto al prezzo attuale del sottostante, espressa in percentuale

`Double PercentUpsideBreakEven`
[Read-Only] Distanza dell'upside Break-Even rispetto al prezzo attuale del sottostante, espressa in percentuale

`Double ProfitProbability`
[Read-Only] Probabilità di profitto della strategia

`Double RealizedPL`
[Read-Only] Totale consolidato della strategia

`Double ReturnRateToExpiry`
[Read-Only] Return rate calcolato fino alla scadenza della strategia, espresso in percentuale

`Double Rho`
[Double] Rho totale di portafoglio della strategia

`Double RhoPercent`
[Read-Only] Rho totale di portafoglio per 1% della strategia

`Double RhoProtect`
[Read-Only] Rho di portafoglio da coprire calcolato dal sistema di hedging automatico

`TStrategySettings StrategySettings`
[Read-Only] Impostazioni della strategia

`TStringList StrategyViewDetails`
[Read-Only] Elenco delle proprietà da visualizzare nella griglia dei dettagli della strategia

`Double SyntheticProfit`
[Read-Only] Profitto sintetico della strategia

`Double Theta`
[Read-Only] Theta totale di portafoglio della strategia

`Double ThetaPercent`
[Read-Only] Theta totale di portafoglio per 1% della strategia

`Double ThetaProtect`
[Read-Only] Theta di portafoglio da coprire calcolato dal sistema di hedging automatico

`TStringList UnderlyingViewDetails`
[Read-Only] Elenco delle proprietà aggiuntive dei sottostanti da mostrare nella griglia dei sottostanti della strategia

```
Double UpsideBreakEven
```
[Read-Only] Upside Break-Even sul prezzo del sottostante della strategia

```
Double ValueAtRisk
```
[Read-Only] Value At Risk della strategia

```
Double Vega
```
[Read-Only] Vega totale di portafoglio della strategia

```
Double VegaPercent
```
[Read-Only] Vega totale di portafoglio per 1% della strategia

```
Double VegaProtect
```
[Read-Only] Vega di portafoglio da coprire calcolato dal sistema di hedging automatico

6.3.17.2. Metodi

```
Function AddFuture(ByVal underlyingIsin As String, ByVal expiry As
TDateTime) As TUnderlying
```
Aggiunge un future, se non è già presente, alla strategia

underlyingIsin	Codice isin del sottostante del future da aggiungere
expiry	Data di scadenza del future da aggiungere

```
Function AddParametricOption(ByVal underlyingIsin As String, ByVal type
As TOptionType, ByVal atmDistance As Integer, ByVal minDays As Integer)
As TOption
```
Aggiunge un'opzione, se non è già presente, alla strategia in base alle sue caratteristiche

underlyingIsin	Codice isin del sottostante dell'opzione da aggiungere
type	Tipo dell'opzione da aggiungere
atmDistance	Distanza, in strikes, dallo strike ATM. Valori positivi indicano strike superiori all'ATM, valori negativi indicano strike inferiori all'ATM, il valore zero indica lo strike ATM
minDays	Numero di giorni minimo prima della scadenza dell'opzione da aggiungere

```
Function AddOption(ByVal underlyingIsin As String, ByVal expiry As
TDateTime, ByVal  strike As Double, ByVal type As TOptionType) As
TOption
```
Aggiunge un'opzione, se non è già presente, alla strategia

underlyingIsin	Codice isin del sottostante dell'opzione da aggiungere
expiry	Data di scadenza dell'opzione da aggiungere
strike	Strike dell'opzione da aggiungere
type	Tipo dell'opzione da aggiungere

```
Sub CalcMaxGainLoss(ByRef maxGain As Double, ByRef maxLoss As Double)
```
Ricalcola i valori di massimo profitto e massimo rischio della strategia e li restituisce nei due parametri

maxGain	[Output] Massimo profitto della strategia
maxLoss	[Output] Massimo rischio della strategia

```
Function CheckAutoOrdersStatus() As Boolean
```
Verifica se i prezzi degli strumenti che compongono la strategia sono tutti validi in modo che i sistemi automatici possano generare ordini

```
Sub Clear()
```
Elimina tutti gli strumenti dalla strategia

```
Sub ClearTrades()
```
Elimina tutti i trades eseguiti sulla strategia

```
Sub CopyFrom(ByVal source As TStrategyContext)
```
Copia tutte le proprietà da un altro oggetto di tipo TStrategyContext
source Oggetto di cui copiare le proprietà

```
Function CreateSecurityFromIsin(ByVal isin As String, Optional ByVal
deleteAllCurrents As Boolean = false, Optional ByVal skipRTConnection As
Boolean = false, Optional ByVal  forceEuropean As Boolean = false,
Optional ByVal forceUnderlyingIsin As String = "") As TSecurity
```
Aggiunge uno strumento alla strategia in base al suo codice isin
isin Codice isin dello strumento da aggiungere alla strategia
deleteAllCurrents Specifica se prima di aggiungere lo strumento deve rimuovere tutti quelli già presenti nella strategia
skipRTConnection Specifica se la connessione real-time non deve essere eseguita sullo strumento aggiunto
forceEuropean Specifica, in caso di opzione, se lo stile deve essere forzato ad europeo anche se per il sottostante le opzioni sono di stile americano
forceUnderlyingIsin Specifica, in caso di opzione, il codice isin del sottostante

```
Sub DeleteSecurity(ByVal security As TSecurity)
```
Elimina uno strumento dalla strategia
security Strumento da eliminare dalla strategia

```
Sub DisableOptionExpiry(ByVal expiryDate As TDateTime)
```
Disabilita tutte le opzioni della strategia con la data di scadenza specificata
expiryDate Data di scadenza delle opzioni da disabilitare

```
Sub DisableSecurity(ByVal security As TSecurity)
```
Disabilita lo strumento specificato
security Strumento da disabilitare

```
Sub EnableOptionExpiry(ByVal expiryDate As TDateTime)
```
Abilita tutte le opzioni della strategia con la data di scadenza specificata
expiryDate Data di scadenza delle opzioni da abilitare

```
Sub EnableSecurity(ByVal security As TSecurity)
```
Abilita lo strumento specificato
security Strumento da abilitare

```
Sub ExecHedging(Optional ByVal skipGlobalFlagCheck As Boolean = false)
```
Esegue il sistema di hedging automatico
skipGlobalFlagCheck Specifica se si deve ignorare la condizione di attivazione generale del sistema di hedging automatico

```
Function FindOption(ByVal underlyingIsin As String, ByVal optionType As
TOptionType, ByVal strike As Double, ByVal expiry As TDateTime) As
TOption
```
Restituisce l'opzione della strategia specificata

underlyingIsin	Codice isin del sottostante dell'opzione
optionType	Tipo dell'opzione
strike	Strike dell'opzione
expiry	Data di scadenza dell'opzione

```
Sub FlatAll()
```
Chiude tutte le posizioni aperte della strategia

```
Function GetByIsin(ByVal isin As String) As TSecurity
```
Restituisce lo strumento della strategia con il codice isin specificato

isin	Codice isin dello strumento da restituire

```
Function GetByName(ByVal name As String) As TSecurity
```
Restituisce lo strumento della strategia con il nome specificato

name	Nome dello strumento da restituire

```
Function GetByObjectID(ByVal id As OBJECT_ID) As TSecurity
```
Restituisce lo strumento della strategia con il codice identificativo specificato

id	Codice identificato dello strumento da restituire

```
Function GetMainUnderlying() As TUnderlying
```
Restituisce il sottostante principale della strategia

```
Function GetMainUnderlyingPrice() As Double
```
Restituisce il prezzo del sottostante principale della strategia

```
Function GetOptionByIndex(ByVal index As Integer) As TOption
```
Restituisce l'opzione della strategia in base al suo indice nell'elenco delle opzioni

index	Indice dell'opzione da restituire

```
Function GetOptionCount() As Integer
```
Restituisce il numero di opzioni presenti nella strategia

```
Function GetPayoffAt(ByVal price As Double, ByVal expiryDate As
TDateTime) As Double
```
Calcola il valore di payoff della strategia al prezzo ed alla data specificati

price	Prezzo del sottostante a cui calcolare il valore di payoff
expiryDate	Data a cui calcolare il valore di payoff

```
Function GetProperty(ByVal property As String) As String
```
Restituisce il valore della proprietà specificata

property	Proprietà

```
Sub GetPropertyList(ByVal list As TStringList, Optional ByVal clearList
As Boolean = false, Optional ByVal prefix As String = "", Optional ByVal
suffix As String = "")
```
[Class Method] Ottiene l'elenco dell proprietà disponibili per gli oggetti della classe TStrategyContext

list	Lista di stringhe dove scrivere l'elenco delle proprietà disponibili
clearList	Indica se azzerare la lista prima di scrivere l'elenco delle proprietà
Prefix	Prefisso da utilizzare prima del nome di ogni proprietà
Suffix	Suffisso da utilizzare dopo il nome di ogni proprietà

```
Function GetUnderlyingByIndex(ByVal index As Integer) As TUnderlying
```
Restituisce il sottostante della strategia in base al suo indice nell'elenco dei sottostanti
index Indice del sottostante da restituire

```
Function GetUnderlyingCount() As Integer
```
Restituisce il numero di sottostanti presenti nella strategia

```
Function IsOptionExpiryDisable(ByVal expiryDate As TDateTime) As Boolean
```
Verifica se la scadenza delle opzioni specificata è disabilitata
expiryDate Data di scadenza delle opzioni da verificare

```
Function LoadFromFile(ByRef fileName As String, Optional ByVal flags As
Integer = LOADFLAG_FULL) As String
```
Carica la strategia da un file su disco e restituisce il nome del file caricato
fileName [Input - Output] Nome del file da caricare/caricato
flags Opzioni di caricamento del file

```
Sub Recalc(Optional ByVal mode As Integer = CALC_MODE_NORMAL)
```
Ricalcola tutti i valori della strategia
mode Metodo di ricalcolo dei valori della strategia

```
Sub RecalcAtNowPayoff()
```
Ricalcola il payoff AtNow della strategia

```
Sub RecalcNow()
```
Ricalcola tutti i valori della strategia dipendenti dal prezzo e dal tempo

```
Sub SaveToFile(ByVal fileName As String, Optional ByVal flags As Integer
= LOADFLAG_FULL)
```
Salva la strategia su di un file su disco
fileName Nome del file da salvare
flags Opzioni di salvataggio del file

```
Sub StartDDE()
```
Avvia tutte le connessioni DDE della strategia

```
Sub StartRT()
```
Avvia tutte le connessioni real-time della strategia

```
Sub StopDDE()
```
Spegne tutte le connessioni DDE della strategia

```
Sub StopRT()
```
Spegne tutte le connessioni real-time della strategia

6.4. Costanti, proprietà e funzioni globali

6.4.1. Costanti

```
DEFAULT_RISK_FREE_RATE
```
Valore di default per il parametro Risk Free Rate (Tasso di interesse privo di rischio)

`PROVIDER_ID_QUICKTRADE`
Identifica la connessione real-time con IWBank QuickTrade Personal External Interface

`PROVIDER_ID_TWS`
Identifica la connessione real-time con InteractiveBroker TraderWorkStation

`PROVIDER_ID_T3`
Identifica la connessione real-time con WeBank T3Open

`PROVIDER_ID_SELLA`
Identifica la connessione real-time con Banca Sella Trading Bridge

`IDX_LAST_BAR`
Rappresenta sempre l'indice dell'ultimo valore disponibile in una delle serie di dati che compongono un contesto di dati storici ed indicatori. Usando questa costante si accede al dato più recente possibile.

`PANEL_PRICE`
Identifica il pannello del grafico storico dove è disegnato il prezzo dello strumento.

`PANEL_VOLUME`
Identifica il pannello del grafico storico dove è disegnato il volume degli scambi dello strumento.

`PANEL_AUTO`
Indica al software di scegliere autonomamente su quale pannello del grafico storico una specifica serie di dati deve essere disegnata.

`DEFAULT_TICK_TIMEOUT`
Identifica il valore di default della durata di validità dei prezzi real-time ricevuti dal mercato. E' espresso in millesimi di secondo.

`UNLIMITED_HEDGE_QUANTITY`
Indica che non viene applicato un quantitativo massimo durante le operazioni di hedging automatico.

`DEFAULT_TICK_DIFF_PERCENT`
Rappresenta il valore di default dello scostamento massimo percentuale che due tick real-time consecutivi ricevuti dal mercato debbono avere per considerare l'ultimo come un tick valido.

`DEFAULT_XAVERAGE_DIFF_PERCENT`
Rappresenta il valore di default dello scostamento massimo percentuale tra l'ultimo tick real-time ricevuto dal mercato e la media mobile esponenziale calcolata sui tick stessi per considerare l'ultimo come un tick valido.

`DEFAULT_PRICE_XAVERAGE_LEN`
Rappresenta il valore di default della lunghezza della media mobile esponenziale calcolata sui tick real-time ricevuti dal mercato.

`MAX_FILTER_MOTIVATION_LEN`
Rappresenta la lunghezza massima, in caratteri, del messaggio di errore relativo alla verifica dei prezzi real-time ricevuti dal mercato.

6.4.2. Proprietà

```
TStrategyContext CurrentStrategy
```

Rappresenta la strategia corrente di FiutoPRO. E' l'oggetto principale all'interno di FPSS.

6.4.3. Funzioni

```
Sub AutoGenerateChain(ByVal underlying as TUnderlying, Optional ByVal
numStrikes as Integer = 16, Optional ByVal numExpiries As Integer = 4)
```

Genera automaticamente la chain delle opzioni per il sottostante specificato

underlying	Sottostante di cui generare la chain delle opzioni
numStrikes	Numero di strikes da creare
numExpiries	Numero di scadenze da creare

```
Sub CalibrateEMaker(ByVal underlyingIsin As String)
```

Calibra il sistema eMaker per il sottostante con il codice isin specificato

underlyingIsin	Codice isin del sottostante di cui eseguire la calibrazione del sistema eMaker

```
Function GetExpiryByIndex(ByVal underlyingIsin As String, ByVal index As
Integer) As TDateTime
```

Restituisce la data di scadenza delle opzioni del sottostante specificato in base all'indice nell'elenco delle scadenze

underlyingIsin	Codice isin del sottostante
index	Indice della scadenza nell'elenco delle scadenze

```
Function GetFutureExpiryByIndex(ByVal underlyingIsin As String, ByVal
index As Integer) As TDateTime
```

Restituisce la data di scadenza del future del sottostante specificato in base all'indice nell'elenco delle scadenze

underlyingIsin	Codice isin del sottostante
index	Indice della scadenza nell'elenco delle scadenze

```
Function GetNearestExpiry(ByVal underlyingIsin As String, ByVal dateTime
As TDateTime) As TDateTime
```

Trova la data di scadenza delle opzioni più vicina possibile alla data specificata

underlyingIsin	Codice isin del sottostante
dateTime	Data di riferimento

```
Function GetNearestStrike(ByVal underlyingIsin As String, ByVal price As
Double) As Double
```

Trova lo strike delle opzioni più vicino possibile al valore specificato

underlyingIsin	Codice isin del sottostante
strike	Valore di riferimento

```
Function GetNextExpiry(ByVal underlyingIsin As String, ByVal dateTime As
TDateTime) As TDateTime
```

Restituisce la data di scadenza delle opzioni immediatamente successiva alla data specificata

underlyingIsin	Codice isin del sottostante
dateTime	Data di riferimento

```
Function GetNextStrike(ByVal underlyingIsin As String, ByVal price As
Double) As Double
```

Restituisce lo strike delle opzioni immediatamente superiore al valore specificato

underlyingIsin	Codice isin del sottostante
price	Valore di riferimento

```
Function GetNumExpiries(ByVal underlyingIsin As String) As Integer
```

Restituisce il numero di scadenza delle opzioni per il sottostante specificato

underlyingIsin	Codice isin del sottostante

```
Function GetNumFutureExpiries(ByVal underlyingIsin As String) As Integer
```

Restituisce il numero di scadenze dei future per il sottostante specificato

underlyingIsin	Codice isin del sottostante

```
Function GetNumStrikes(ByVal underlyingIsin As String) As Integer
```

Restituisce il numero di strike delle opzioni per il sottostante specificato

underlyingIsin	Codice isin del sottostante

```
Function GetPersistentVar(ByVal name As String) As String
```

Recupera il valore di una variabile persistente. Le variabili persistenti mantengono il valore impostato anche per esecuzioni successive degli script e fanno parte della strategia di FiutoPRO

name	Nome della variabile persistente da leggere

```
Function GetPreviousExpiry(ByVal underlyingIsin As String, ByVal
dateTime As TDateTime) As TDateTime
```

Restituisce la data di scadenza delle opzioni immediatamente precedente alla data specificata

underlyingIsin	Codice isin del sottostante
dateTime	Valore di riferimento

```
Function GetPreviousStrike(ByVal underlyingIsin As String, ByVal price
As Double) As Double
```

Restituisce lo strike delle opzioni immediatamente precedente al valore specificato

underlyingIsin	Codice isin del sottostante
price	Valore di riferimento

```
Function GetStrikeByIndex(ByVal underlyingIsin As String, ByVal index As
Integer) As Double
```

Restituisce lo strike delle opzioni in base al suo indice nell'elenco degli strike

underlyingIsin	Codice isin del sottostante
index	Indice dello strike

```
Sub GetTheoreticalOption(ByVal underlyingIsin:String, ByVal expiry As
TDateTime, ByVal strike As Double, ByVal type As TOptionType, ByRef
price As Double, ByRef delta As Double, ByRef gamma As Double, ByRef
theta As Double, ByRef vega As Double, ByRef rho As Double, ByRef
timeValue As Double, ByRef implVolatility As Double)
```

Calcola i valori teorici di una opzione

underlyingIsin	Codice isin del sottostante
expiry	Data di scadenza dell'opzione
strike	Strike dell'opzione
type	Tipo dell'opzione

price	[Output] Premio dell'opzione
delta	[Output] Delta dell'opzione
gamma	[Output] Gamma dell'opzione
theta	[Output] Theta dell'opzione
vega	[Output] Vega dell'opzione
rho	[Output] Rho dell'opzione
timeValue	[Output] Time Value dell'opzione
implVolatility	[Output] Volatilità implicita dell'opzione, espressa in percentuale

```
Sub SetPersistentVar(ByVal name As String, ByVal value As String)
```

Imposta il valore di una variabile persistente. Le variabili persistenti mantengono il valore impostato anche per esecuzioni successive degli script e fanno parte della strategia di FiutoPRO

name	Nome da assegnare alla variabile persistente
value	Valore da assegnare alla variabile persistente