

18 HkScript

18.1 Introduktion

I Hitta kursvinnare är det möjligt att skapa egna tekniska indikatorer. Det gör du genom att använda ett enkelt skriptspråk som heter HkScript-language. Detta fungerar som en mycket förenklad form av programmering som låter dig använda en mängd olika funktioner. Med hjälp av skriptspråket kan du i princip bearbeta kursdatan hur som helst. Du kan också utgå från programmets redan förprogrammerade indikatorer, något som kan vara smidigt ifall du bara vill göra en variant av en vanlig indikator.

För att komma igång och sedan behärska HkScript-language till fullo så krävs det en del övning. Vi rekommenderar att du går igenom introduktionsavsnitten nedan. Använd sedan Exempelbiblioteket och Referensbiblioteket för att lära dig skapa mer avancerade skript.

Kom igång

1. Introduktionen (här är du nu)
2. Översikt av HkScript
3. Kodning med HkScript
4. Parametrar, Plot
5. Variabler
6. Se enkla exempel

Mängder med exempel

7. Kodexempel: RSI m fl med hela skript. Vi går från det mest grundläggande till avancerade skript.
8. Referensbibliotek. Här beskriver vi alla de olika funktioner som finns att använda i skriptspråket. Naturligtvis finns exempel till samtliga funktioner.
9. Indikatorbibliotek. Indikatorerna är ofta viktiga i egna skript. I vårt indikatorbibliotek kan du läsa om de olika indikatorerna och se exempel på hur de kan användas.

Klicka på [nästa](#) för att få en översikt av användargränssnittet, dvs. miljön där du skapar skripten.

18.2 Översikt av HkScript

När du ska arbeta med HkScript går du in på menyn **Verktyg -> Inställningar -> Indikatorer & Skript**. Här får du en överblick över alla dina skript, funktioner för att importera och exportera skript, kopiera osv. Klicka på knappen **Skapa ny** för att öppna arbetsytan där du skapar dina skript.

I denna ruta finns följande rutor och knappar:

Namn och kortnamn: Här anger du namn på indikatorn, i den mindre rutan till höger anger du förkortning. Det är denna förkortning som visas i Indikatorknapppraden till höger i TA-modulen.

Beskrivning: Här anger du en beskrivning av din indikator.

Kod: I denna ruta skriver du din HkScript-kod.

Verifiering: I den här rutan verifieras att din kod fungerar. Om den gör det så kommer det stå "Verifiering OK" i grön text. Om något är fel så kommer du få ett felmeddelande som visar vad som är fel.

Under fliken **inställningar** hittar du ett antal viktiga alternativ.

Skapa/ändra egen indikator

M2(p1) : plot1

KursMA M2

Beskrivning:
Visar medelvärdet av stängningskurs det två senaste perioderna.

Kod Inställningar

Typ:

Indikatortyp: Teknisk indikator

Visa indikatorn i: ny panel.

Inparametrar: [Lägg till inparameter](#)

p1	heltal	10	beskr.
----	--------	----	--------

Utdata/diagramlinjer: [Lägg till utdata](#)

plot1			1pt	pl1
-------	--	--	-----	-----

Globala variabler: [Lägg till global variabel](#)

Kodexempel... Hjälp Verifiera Avbryt OK

Indikatortyp: Välj typ av indikator.

Visa indikatorn i: Välj ifall du vill att indikatorn ska visas i huvudfönstret eller i en egen panel.

Inparametrar: För att optimera dina indikatorer kan det vara användbart att använda sig av parametrar. Genom att definiera dina parametrar kan du sedan direkt i diagrammet smidigt göra inställningar för exempelvis perioder, procentnivåer, avstånd till övre/undre linje som du använder i dina indikatorer.

Utdata/diagramlinjer: Inställningar för de olika plottarna (dvs. dataserierna/linjerna). Använd knappen **Lägg till utdata** om du använder flera linjer.

Globala variabler: Du kan använda egna variabler för att använda i skriptet. Genom att använda dig av variabler kan du exempelvis definiera startpunkter.

Obs! *Ichu_ChikouSpan (Lagging Span) har aktiekurs flyttad bakåt t ex 26 dagar (default). Om ChikouSpan ingår i en Strategi / backtest så blir resultat därför missvisande.*

När du är klar med din indikator, eller vill testa den så klickar du på **OK**. Lägg till indikatorn i indikatorknappraden genom att kryssa i rutan i den högra kolumnen under **Verktyg -> Inställningar -> Indikatorer & script**, den kan även läggas till som vanligt under **Verktyg -> Inställningar -> TA -> Indikatorer**).

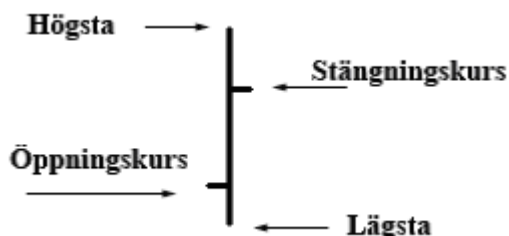
Klicka på [nästa](#) för läsa om kodning och skriptspråkets uppbyggnad.

18.3 Byggstenar HkScript

I det här avsnittet beskriver vi skriptspråkets uppbyggnad. Genom att förstå logiken bakom blir det enklare för dig att lära dig skapa dina egna indikatorer.

Dataserier (kursdata)

Grunden i teknisk analys är historiska dataserier, som innehåller information om prisnivåer och handelsvolym. En vanlig handelsdag delas upp i ett antal tidsenheter (även kallat perioder), exempelvis i 5- eller 10-minutersintervall. I diagrammen representeras varje tidsenhet av en stapel. Stapeln är ett sätt att visuellt visa upp ett antal dataserier; öppningskurs, högsta kurs, lägsta kurs och stängningskurs.



När du ska skapa egna indikatorer med hjälp av skriptspråk behöver du veta hur du plockar ut just den datan som du är intresserad av. Ifall du exempelvis ska skapa en indikator som är ett kort glidande medelvärde, som indikatorn MA1, använder du senaste 20 värdena i dataserien Stängningskurs. De 20

värdena plockas ut ur dataserien, och bearbetas sedan för att bearbetas matematiskt för att få fram ett nytt värde.

Tidsenhet	15:30-15:35	15:35-15:40	15:40-15:45
Öppningskurs (Open)	10	11	10
Högsta kurs (High)	11	13	12
Lägsta kurs (Low)	9	10	10
Stängningskurs (Close)	11	10	12
Volym (Volume)	10 000	15 000	12 500

För att plocka ut dessa värden måste du veta hur du anger vilken dataserie du ska hämta värden ifrån, samt hur du anger att det är just de 20 senaste värdena som ska hämtas. I HkScript anges detta genom att skriva `Close[-20]` där `Close` anger att det är stängningskursen, och `[-20]` anger att det är de senaste 20 värdena som ska hämtas. Man anger alltid hur många tidsenheter *bakåt* som ska hämtas, eftersom indikatorer bygger på historisk data.

För att visa exempelvis stängningskursen skriver du: `plot1[0] = Close[0];`

Notera att *diagrammets dataserier (kursdata)* är serier med enbart data, till skillnad från *Indikatorer* som även möjliggör för inparametrar att sättas inom parentes efter namnet "()". Diagrammets dataserier är följande fem: Open, High, Low, Close, Volume.

Dataserier (indikatorer)

Indikatorer är precis som exempelvis stängningskurser värden som ligger i en dataserie. Även om de bygger på mycket information så är resultatet bara en dataserie.

Tidsenhet	15:30-15:35	15:35-15:40	15:40-15:45
MA1	200	205	210

För att visa MA1 skriver du: `plot1[0] = MA1()[0];`

Hitta kursvinnare känner igen när du anropar en indikator och färgar texten till *rosa*.

Indikatorer kan ha olika parameterinställningar. MA1 tar exempelvis fram medelvärdet av de senaste 20 tidsenheterna, men detta kan ändras till exempelvis 25 ifall du så önskar. Ifall du vill använda standardinställningarna räcker det med att skriva två parenteser mot varandra såhär: "()". Ifall du vill använda andra parametrar, som 25 istället för 20, så skriver du det inom parentes: "(25)". Vill du visa MA1 med parameter 25 skriver du alltså: `plot1[0] = MA1(25)[0];`

Matematiska funktioner

Du kan naturligtvis bearbeta dina dataserier enkelt med addition, division etc. genom att använda plustecken (+), divisionstecken (-) m.m. Exempelvis såhär: `plot1[0] = Close[0] + 10;` för att visa slutkursen plus tio.

Programmet innehåller även mer avancerade matematiska funktioner, som ger möjlighet att exempelvis summera eller räkna fram standardavvikelsen. Du kan hitta alla dessa funktioner i Referensbiblioteket. Vill du exempelvis använda en funktion för att summera värdena i en dataserie (i detta fall Close) skriver du: `plot1[0] = Close.sum(10);`

Plot (*rita ut*)

Plot är engelska och betyder ungefär *rita ut*. Som tidigare nämnts är de nya indikatorer du skapar även de dataserier. För att indikatorn ska visas i ett diagram måste du ange detta. Exempelvis genom att skriva: `plot1[0] = Close[0];` för att visa slutkursen.

Indikatorer består ofta av flera dataserier som du vill rita ut. Detta är exempelvis vanligt när du vill visa ett band eller en kanal. Ifall du vill visa linjer behöver du lägga till dessa under **inställningar -> lägg till utdata**. Plot och parametrar kan även läggas till direkt i koden, så som beskrivs här.

Exempel med tre utritade dataserier:

```
plot1[0] = Close[0];
plot2[0] = Close[0]*1.1;
plot3[0] = Close[0]*0.9;
```

`plot1[0]` visar i detta fall stängningskursen, medan `plot2[0]` visar stängningskurs plus tio procent och `plot3[0]` stängningskurs minus tio procent. Du kan välja färg, linjebredd m.m. för varje "plot" som visas. Du bör inte ändra värdet inom hakparentes [0] annat än vid specialfall, utan låt det alltid vara noll.

Men ovanstående översikt så bör du kunna göra enklare skript. För att göra mer avancerade skript kan det vara användbart med variabler.

Klicka på [nästa](#) för att läsa mer om parametrar och plot.

18.4 Parametrar, Plot

I det här avsnittet beskriver vi hur du kan använda dig av parametrar i skriptspråket. De flesta indikatorer innehåller parametrar, som anger exempelvis hur många tidsenheter/perioder indikatorn beräknas på. Kort period, MA1, är exempelvis ett glidande medelvärde av de senaste 20 tidsenheterna/perioderna. För indikatorn MA1 är parametern alltså förinställd till 20.

Om man i skriptspråket inte definierar en egen parameter så kommer de parametrar som är satta i inställningarna för TA-indikatorerna att gälla. Titta i Indikatorbiblioteket för att se vilka parametrar som finns, och vilka standardinställningar som gäller, för den indikator du är intresserad av.

Fördelen men att sätta parametrarna specifikt i skriptet är att de då blir oberoende av andra ändringar av parameterinställningarna i Hitta kursvinnare. Du kan exempelvis ange att i ditt egna skript ska MA1 beräknas på 25 tidsenheter, även om MA1 i resten av programmet använder standardinställningen 20 tidsenheter.

Lägg till parameter

Notera: Max 6 parametrar kan användas

I de flesta indikatorer används flera inparametrar. Ifall du vill använda flera inparametrar så under fliken Inställningar klicka på **Lägg till inparameter** ovanför rutan Inparametrar. I rutan anger du typ (heltal eller decimaltal), ett värde, samt en beskrivning. Det är viktigt att vara uppmärksam på att ifall du inte anger att det är ett decimaltal kommer värdet att avrundas. Parametrars namn i koden är alltid "p1" till "p6". Exempelen nedan gäller indikatorer, men även **strategiskript** kan använda inparametrar, se avsnitt Strategier i HkScript.

I koden skriver du sedan exempelvis: `plot1[0] = MA1(p1)[0];`

Du kan nu ändra inställningar för parameter i skriptet utan att det påverkar MA1-inställningarna i övrigt.

Lägg till parameter via script

Istället för att använda fliken Inställningar kan parametrar också läggas till direkt i koden, så som beskrivs här.

T ex "p1" ovan läggs till som:

```
#global param("p1", 20); // parameter1, default = 20
```

Plot (rita ut)

Notera: Max 9 plot kan användas

Plot är engelska och betyder ungefär *rita ut*. Som tidigare nämnts är de nya script-indikatorer du skapar även de dataserier. För att en ny script-indikator ska visas i ett diagram måste du ange hur. Exempelvis genom att skriva: `plot1[0] = Close[0];` för att visa slutkursen.

Namn för plot i koden är alltid "plot1" till "plot9", men ett presentationsnamn kan också sättas.

Indikatorer består ofta av flera dataserier som du vill rita ut. Detta är exempelvis vanligt när du vill visa ett band eller en kanal. Ifall du vill visa linjer (via plot)

behöver du lägga till dessa under **inställningar** -> **lägg till utdata**. (eller deklarerar plot i koden)

Exempel med tre utritade dataserier:

```
plot1[0] = Close[0];  
plot2[0] = Close[0]*1.1;  
plot3[0] = Close[0]*0.9;
```

`plot1[0]` visar i detta fall stängningskursen, medan `plot2[0]` visar stängningskurs plus tio procent och `plot3[0]` stängningskurs minus tio procent. Du kan välja färg, linjebredd m.m. för varje "plot" som visas. Du bör inte ändra värdet inom hakparentes [0] annat än vid specialfall, utan låt det alltid vara noll.

Lägg till plot via script

Istället för att använda fliken Inställningar kan plot också läggas till direkt i koden, så som beskrivs här.

T ex "plot1" ovan läggs till som:

```
#global plot("plot1", RGB(0,200,0), 2); //grön plot1,  
tjocklek 2
```

eller enklare, så kan bara antalet läggas till:

```
#global nbrplots(3) //använd 3 st plot
```

Men ovanstående översikt så bör du kunna göra enklare skript. För att göra mer avancerade skript kan det vara användbart med variabler.

Klicka på [nästa](#) för att läsa mer om variabler.

18.5 Variabler

I det här avsnittet beskriver vi hur du kan använda dig av variabler i skriptspråket. Variabler är ett bra sätt att hålla skript korta och enkla att arbeta med. T ex för scriptkod som återkommer på flera ställen som du enkelt vill kunna uppdatera, eller beräkna någonting. Särskilt för globala variabler ska du gå till fliken **inställningar** -> **Lägg till global variabel**.

Alternativt lägg till global variabel **direkt i koden**, så som beskrivs här.

Lokal variabel

Variabler skiljer på heltal och decimaltal. Observera att första gången variabeln tilldelas ett värde, så bestäms om typen är heltal eller decimaltal.

*Lokala variabler behåller endast sitt värde **för en scriptkörning** (för aktuell bar).*

Heltal

```
local heltalsvarde = 100;  
plot1[0] = heltalsvarde - 50;
```

Decimaltal

```
local lagsta_pris = 99.25;  
plot1[0] = Close[0] - lagsta_pris;
```

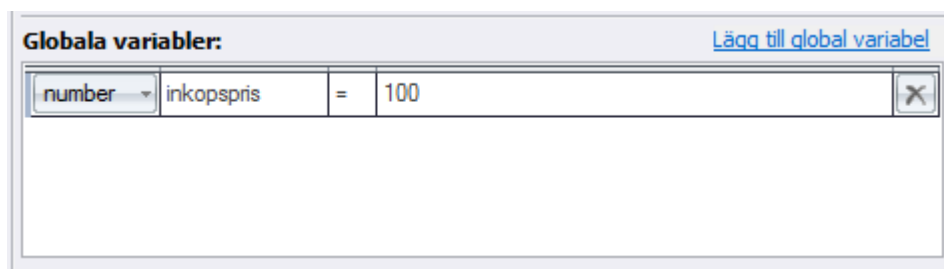
Boolean (sant/falskt)

```
local villkor1 = true;
if (villkor1)
    plot1[0] = 10;
```

Global variabel

Global variabel läggs till via fliken Inställningar->Lägg till global variabel , eller direkt i koden med deklarationen **#global**.

Globala variabler behåller sitt värde **för alla scriptkörningar** (för alla bar i diagrammet).



Ett exempel:

```
plot1[0] = inkopspris;
plot2[0] = inkopspris*1.1;
plot3[0] = inkopspris*0.9;
```

I detta fall är "inkopspris" en variabel där värdet är satt till inköpspriset på vår aktie. Det räcker alltså med att byta värdet på variabeln på ett ställe, du slipper ändra det på varenda ställe i skriptkoden.

Alternativ: Direkt i koden kan ovan istället deklarerars **#global**
`number("inkopspris", 100);`

Text som variabel

Inget användningsområde i dagsläget. Kan komma att implementeras i senare versioner av Hitta kursvinnare.

Dataserie som variabel

Du kan spara en hel dataserie som scriptvariabel. Lägg exempelvis till en scriptvariabel och döp den till `rsi_diff`.

Du definierar sedan `rsi_diff` i koden. Exempelvis så här: `rsi_diff[0] = RSI()[0] - RSI()[-1];`

Du kan sedan bearbeta denna serie precis som alla andra serier, exempelvis genom att skriva:

```
plot1[0] = rsi_diff.sma(20);
```

Alternativ: Direkt i koden kan ovan istället deklarerars **#global**
`dataserie("rsi_diff", 100);`

Instrument som variabel

Instrument är normalt ett index eller en akties all data. Anger du ex Eric = Ericsson B så når du all data som gäller denna aktie.

Du kan sedan bearbeta aktiedatan. Ett mycket enkelt exempel:

```
plot1[0] = Eric.Close[0];
```

Även tekniska indikatorer kan användas, exempel:

```
plot1[0] = Eric.RSI()[0];
```

Alternativ: Direkt i koden kan ovan istället deklarerars `#global`
`instrument("Eric", "Ericsson B");`

Se även avsnittet Periodlås hur man kan låsa Instrument till en viss beräknings-period (Fixed timeframe).

Se alla exempel i Exempelbiblioteket

18.6 FA i HkScript

Beskrivning av HkScript kan använda fundamental analys och bolagens rapportinformation. Att kombinera fundamental och teknisk analys kan vara ett värdefullt sätt att hitta långsiktiga kursvinnare. Se komplett lista med funktionerna här: FA-Nyckeltal.

Bakgrund: Dataserier för vanlig kurs- och indikatordata

I HkScript körs scriptet en gång för varje bar. För ett diagram på 1 år så blir det ca 250 värden (bars) i ett vanligt dagschart.

Dataserier för `Close` eller t ex `RSI` och all annan indikatordata har samma längd 250.

* Vid varje tillfälle kommer `Close[0]` visa aktuell slutkurs för den dagen.

* Medan t ex föregående dag skrivs `Close[-1]`.

För att visa en graf med stängningskursen skriver du: `plot1[0] = Close[0];`

FA-dataserier: Särskilda dataserier för Rapportdata

Dataserien för Rapportdata (FA-dataserie) har till skillnad från kurs- och indikatordata istället **samma längd som antalet rapporter tillgängliga**.

För ett diagram på 1 år är det normalt endast 4 st värden, dvs 4 rapporter. Obs ett rapportvärde visar alltid 12 månader med rullande kvartal.

Så när det finns 16 kvartalsrapporter är längden 16, och funktionen "get(0)" ger alltid det senaste rapportvärdet (t ex "`Fa.Sales().get(0)`").

* Vid varje tillfälle kommer `Fa.Sales().get(0)` visa senaste rapports Omsättning per aktie. (försäljning)

* För korrekt historisk backtesting och i *enlighet med verkligheten* finns historiska rapportvärden först samma dag som bolaget presenterat sin rapport - dvs inte sista dagen i respektive kvartal.

* All rapportdata anges som 12 månader rullande kvartal. Dvs 1 helt år, med ny info varje kvartal.

* Föregående rapports värde skrivs `Fa.Sales().get(-1)`. (dvs den föregående rapportens 12 månader rullande kvartal)

För att visa en graf med aktuell Försäljning per aktie skriver du:

```
plot1[0] = Fa.Sales().get(0);
```

Innehåll i FA-dataserier

Varje värde i en FA-dataserie innehåller ett helt rapportår, baserat på 4 kvartals sammanslagna data. Efter ett bokslut (Q4) så innehåller

`Fa.Sales().get(0)` hela fjolårets omsättning per aktie. Dvs

Q4+Q3+Q2+Q1.

Men ifall tidpunkten istället är efter Q1 (se tabell nedan) så innehåller

`Fa.Sales().get(0)` i det fallet Q1 innevarande år plus Q4+Q3+Q2

föregående år.

Det här gör att alla värden blir jämförbara och att eventuella säsongsvariationer inte påverkar resultaten.

Se exempel nedan.

Exempel: Dagens datum är 30 april 2025 och bolagets Q1-rapport släpptes nyligen, den 20 april:

Funktion:	Sales().get(0)	Sales.get(-1)	Sales.get(-4)
Slutkvartal (rullande 12m)	Q1-2025	Q4-2024	Q1-2024
Tillgänglig från (exempel)	2025-04-10	2025-02-10	2024-04-10
Startkvartal (rullande 12m)	Q2-2024	Q1-2024	Q2-2023

Matematiska funktioner

Du kan bearbeta dina dataserier enkelt med addition, division etc. genom att använda plustecken (+), divisionstecken (-) osv. Exempelvis så här: "

`plot1[0] = Fa.Sales().get(0) * 2.5`" för att visa rapportvärde gånger två och en halv. (observera att decimaltal skrivs med punkt: '.')

Programmet innehåller även mer avancerade matematiska funktioner. Vill du exempelvis använda en funktion för att se årsvis snitt-tillväxt (%) under 5 år (för exemplet Sales), så anges: `Fa.Sales().yearGrowthAvg(5)`

Du finner hela listan i avsnittet Fa-dataserier-funktioner.

Exempel funktioner

Några ytterligare exempel:

```
if (Fa.Sales().growth(1, "Y") > 25)
```

```
    plot1[0] = 1; // Köp om Försäljningstillväxt
    senaste året > 25% ("1 Year")
```

```
if (Fa.Sales().growth(1, "Q") > 20)
```

```
    plot1[0] = 1; // Köp om Försäljningstillväxt
    senaste kvartalet > 20% ("1 Quarter") - baserat på
    rullande 4 kvartal.
```

```
if ((Fa.PE().get(0) < 10) && (Close[0] > MA3(200)
[0]))
```

```
    plot1[0] = 1; // Köp om PE är under 10 OCH Close
    är över MA200
```

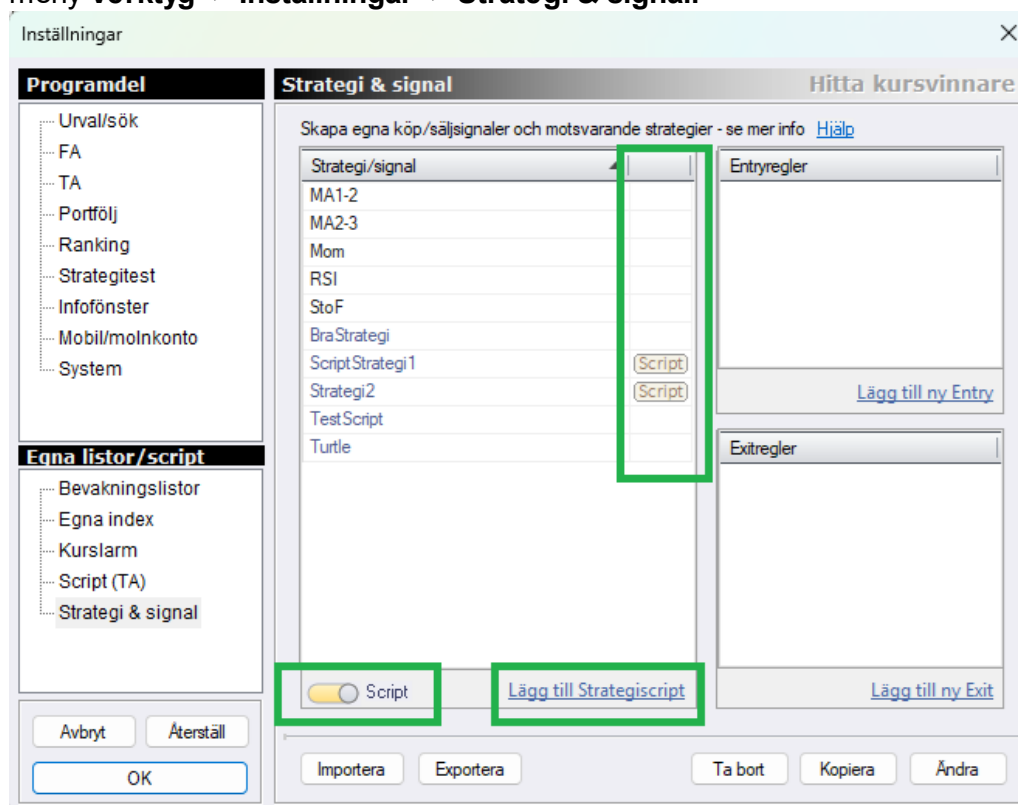
```
if (Fa.PE().get(0) < Fa.PE().yearAvg(5))
```

```
plot1[0] = 1; // Köp om PE är under sitt 5-års-  
snitt
```

18.7 Strategiscript

Strategi-skript är ett flexibelt sätt att **skapa, testa och utveckla handelsstrategier direkt med kod**.

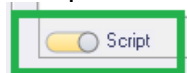
För att skapa strategiskript (eller övriga egna signaler & strategier), öppna meny **Verktøy -> Inställningar -> Strategi & signal**.



* Välj här att ändra till alternativet "Script" enligt bilden ovan.

* Strategiscripten markeras i listan med symbolen "Script" i högerkolumnen.

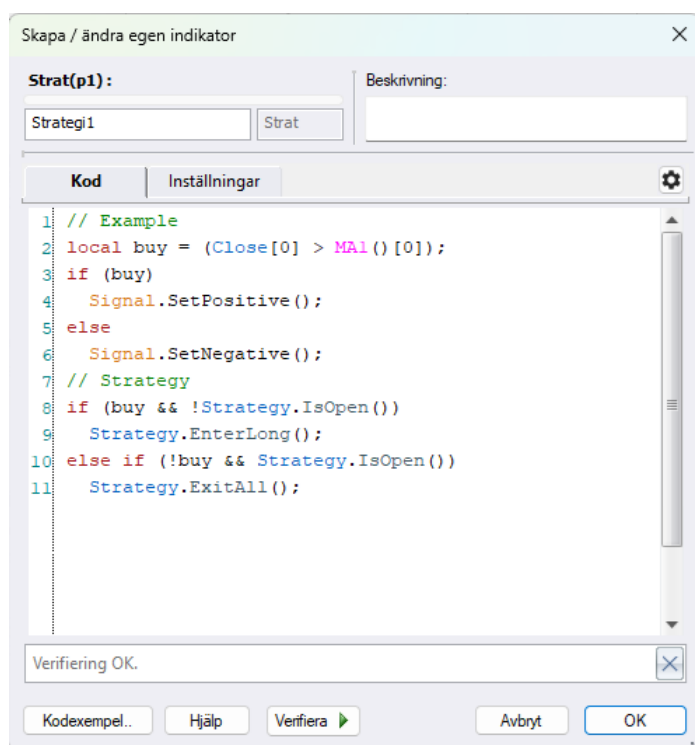
Hitta kursvinnarens grundinställning är att strategier skapas via ett enklare användargränssnitt, se avsnittet Skapa strategi & signal, och **inte med script**. Dvs det finns två lägen, som ändras via alternativet "Script".



1. "Lägg till enkel Strategi". (Script **Av**)
2. "Lägg till Strategiscript". (Script **På**)

När "Script" är aktiverat:

Klicka "**Lägg till Strategiscript**" för att skapa nytt strategiscript.



* Ett kodfönster kommer nu skapas med det nya strategiskriptet.

1. Villkor för Signalstyrka

Nästa steg är att ange de köpvillkor och eventuella säljvillkor som ska gälla. Villkoren kan baseras på kurser, tekniska indikatorer och tekniska formationer. Ett enkelt exempel med glidande medelvärden kan se ut så här, i fallet där villkoret först sparas i Signal-objektet:

```
if (MA1(20)[0] > MA2(50)[0])
    Signal.SetPositive();
else if (MA1(20)[0] < MA2(50)[0])
    Signal.SetNegative();
```

I exemplet ovan använde vi "större än" som villkor när vi jämförde MA1 mot MA2. (Signal sattes till Positiv).

Förutom "större än" så finns såklart många andra sätt att skapa köpvillkor, och det går även direkt göra Köp/sälj via "Enter", till exempel:

Indikator korsar upp över en nivå:

```
if (StoS(14).crossAbove(20))
    Strategy.EnterLong();
```

Indikator korsar upp över annan indikator:

```
if (MA1(5).crossAbove(MA2(10)))
    Strategy.EnterLong();
```

Stigande indikator:

```
if (RSI(14)[0] > 70 && RSI(14).isUp())
    Strategy.EnterLong();
```

Händelse har inträffat (här Rapport och Key reversal):

```
if (Fa._isReportDay() && Pattern.Rev_pos())
    Strategy.EnterLong();
```

Jämförelsevillkor och logiska operatörer

Vid skapande av mer avancerade villkor används logiska operatörer, såsom **och**, **eller** och **inte**, för att kombinera flera uttryck.

De skrivs i HkScript på samma sätt som i många andra programspråk:

Operator	Förklaring	Exempel
&&	Och – båda villkoren måste vara sanna	<villkor1> && <villkor2>
	Eller – något av villkoren måste vara sanna	<villkor1> <villkor2>
!	Inte – inverterar ett villkor	!<villkor1>
&& !	Och inte – ett sant villkor och ett inte sant	<villkor1> && !<villkor2>

Du kan alltså bygga upp komplexa logiska uttryck genom att kombinera flera villkor.

För att hålla koden tydlig rekommenderas att först lagra varje delvillkor i en **lokal variabel**, och sedan kombinerar dessa i en if-sats.

Det gör koden mer läsbar och enklare att felsöka.

Exempel 1

```
local villkor1 = RSI(14)[0] < 30;
local villkor2 = MA1(20)[0] > MA2(50)[0];
local villkor3 = VOL()[0] > VOL_MA(20)[0];
if (villkor1 && (villkor2 || villkor3) &&
    Strategy.IsZeroPos())
    Strategy.EnterLong();
```

Exempel 2

```
local oversalt = RSI(14)[0] < 30;
local stigande = Close[0] > Close[-1];
if (oversalt && stigande && Strategy.IsZeroPos())
    Strategy.EnterLong();
```

2. Entry för Köp (eller kort position)

Nästa steg är att ange hur strategin faktiskt ska genomföra köp- eller säljaffärer.

Detta görs med funktionerna i objektet `Strategy`, som används för att öppna och stänga positioner.

De mest grundläggande funktionerna är:

- `Strategy.EnterLong()` – öppnar en lång position (köp)
- `Strategy.EnterShort()` – öppnar en kort position (sälj/kort)
- `Strategy.ExitAll()` – stänger alla öppna positioner

Dessa funktioner anropas vanligtvis när dina köp- och säljvillkor uppfylls.

Du kan välja att anropa dem direkt, eller låta signalerna (`Signal.SetPositive()` / `Signal.SetNegative()`) styra köpen.

Ett enkelt exempel:

```
// --- Entry ---  
if (Strategy.IsZeroPos() && Signal.ChangedToPositive())  
    Strategy.EnterLong();
```

Här utförs köpet först när signalen ändrats till "+", dvs `ChangedToPositive()`, och positionen stängs när negativ signal uppstår. Denna struktur är mycket tydlig och används ofta i strategier som testas över tid.

Du kan även använda hjälpfunktioner för att kontrollera aktuell position:

- `Strategy.IsOpen()` – har öppen position
- `Strategy.IsZeroPos()` – ingen öppen position
- `Strategy.IsLong()` – lång position är öppen
- `Strategy.IsShort()` – kort position är öppen

Dessa gör det enkelt att exempelvis förhindra dubbla köp, eller att skapa särskilda exit-regler för korta positioner.

3. Exit- och stoppvillkor

Nästa steg är att ange hur strategin ska stänga positioner – antingen via säljsignaler eller automatiska stoppar.

Detta kan göras med både manuella villkor och funktioner i `Strategy`-objektet.

Vanliga sätt att stänga en position:

- Villkorsbaserad exit (t.ex. när en indikator passerar ett värde)
- Stop-loss – stänger när priset faller under viss nivå
- Profit target – stänger vid uppnådd vinstnivå
- Tidsbaserad exit – stänger efter ett visst antal staplar

De viktigaste funktionerna är:

- `Strategy.SetPriceStop(price)` – anger en prisnivå där positionen ska stängas om kursen går emot dig.
- `Strategy.SetPriceTarget(price)` – anger en prisnivå för vinsttagning.
- `Strategy.ExitAll()` – stänger manuellt alla positioner.

Exempel på villkorsbaserad exit:

```
if (RSI(14)[0] > 70)  
    Strategy.ExitAll();
```

Exempel på prisnivå för vinsttagning:

```
if (Strategy.IsZeroPos() && Signal.ChangedToPositive())  
{ // --- Entry ---  
    Strategy.EnterLong();  
    Strategy.SetPriceTarget(Close[0] * 1.05);  
}
```

Exempel på tidsbaserad exit:

```
if (Strategy.BarsSinceEntry() >= 5)
    Strategy.ExitAll();
```

Genom att kombinera olika exits (prisstop, vinstmål, villkorsstyrda exits) kan du bygga flexibla och realistiska handelsmodeller.
Se alla verktyg i Referenslistan för Strategiscript.

18.8 Enkla exempel

Här ett antal korta exempel på hur indikatorer byggs upp. För fler exempel titta i referensbiblioteket och kodexempel RSI m fl.

Indikator	Beskrivning	Skript
Slut, högsta och lägsta		
Slutkurs	Visa kurva med slutkursen	<code>plot1[0] = Close[0];</code>
Lägsta, Högsta kurs	Visa kurva med snittet av lägsta- och högstakursen	<code>plot1[0] = avg (Low[0], High[0]);</code>
Öppningskurs	Visa kurva med öppningskursen	<code>plot1[0] = Open[0];</code>
Tidsenheter bakåt		
Slutkurs en tidsenhet bakåt		<code>plot1[0] = Close[-1];</code>
Högsta kurs en tidsenhet bakåt		<code>plot1[0] = High[-1];</code>
Kurs +/- * / ÷		
Slutkurs + 10 (+10kr)		<code>plot1[0] = Close[0]+10;</code>
Slutkurs * 1.05 (+5%)	Notera 1,05 skrivs med punkt (ej kommatecken) . Skrivs alltså 1.05	<code>plot1[0] = Close[0]*1.05;</code>

Indikator	Beskrivning	Skript
Slutkurs ÷ 2		<code>plot1[0] = Close[0]/2;</code>
Gårdagens slutkurs - 5%	Visa gårdagens stängningskurs minus 5%	<code>plot1[0] = Close[-1]*0.95;</code>
Nya kursnivåer		
Nytt 10 dagars högsta	Visa värdet 1 när ett nytt 10-dagars-högsta skett	<code>plot1[0] = Close[0]/2;</code>
Nytt All time high (ATH)	Visa värdet 1 när ett nytt All time high har skett	<code>plot1[0] = Close[0]/2;</code>
Medelvärden som indikatorer	I Hitta kursvinnare kan du använda förinställda indikatorer (MA1, MA2 osv). Medelvärde kan även räknas ut genom en matematisk funktion (se under kapitel matematiska funktioner)	
Visa MA (1-5)	Visa indikatorn MA1 = default 20 dagars medelvärde om inget annat angivits. (se även MA2, MA3, MA4, MA5)	<code>plot1[0] = MA1() [0]; // eller MA1(20)[0];</code>
Visa flera EMA	Visa indikatorn EMA2 , testa med flera olika parameterinställningar	<code>plot1[0] = EMA2(3) [0]; plot2[0] = EMA2(5) [0]; plot3[0] = EMA2(20)[0];</code> Lägg till plot 2-3 under inställningar -> lägg till utdata
MA Slope	Visar lutning på en MA-kurva, dvs skillnaden mellan intilliggande värden på MA. I detta fall används MA1.	<code>plot1[0] = MA1() [0]-MA1() [-1];</code>
Medelvärde som matematisk funktion		

Indikator	Beskrivning	Skript
sma (simple moving average)	Visa medelvärdet av stängningskurs under de senaste två dagarna.	<code>plot1[0] = Close.sma(2);</code>
Avstånd- MA200	Visar avståndet mellan aktuell kurs och dess MA200 i procent, med en linje som markerar 0-nivån. (t ex stort avstånd ökar sannolikhet för återgång mot MA).	<code>plot1[0] = ((Close[0] - Close.sma(200)) / Close[0]) * 100; plot2[0] = 0;</code> Lägg till plot 2 under inställningar -> lägg till utdata
Medelvärde som matematisk funktion på indikator	Visar medelvärdet av MACD de 200 senaste tidsenheterna.	<code>plot1[0] = MACD().sma(200);</code>
Användning av IF- satser		
Kurs ökande	Visa värdet 1 ifall dagens stängningskurs är större än gårdagens, annars 0.	<code>if (Close[0] > Close[-1]) plot1[0] = 1 else plot1[0] = 0;</code>
Stapels höjd	Visa värdet 100 ifall höjden på dagens kursstapel är större än 2%, annars -1.	<code>if (High[0] / Low[0] > 1.02) plot1[0] = 100 else plot1[0] = -1;</code>
Piercing Pattern	Skapa indikator för Piercing Pattern. Följande villkor skall vara sanna: 1. Gårdagens stängningskurs skall vara lägre än gårdagens öppning 2. dagens öppning är lägre än dagens stängning 3. dagens öppning är lägre än gårdagens lägsta 4. dagens stängning är lägre än gårdagens öppning	För att skapa indikator för Piercing pattern: <code>if ((Close[-1] < Open[-1]) && (Open[0] < Close[0]) && (Open[0] < Low[- 1]) && (Close[0] < Open[- 1]) && (Close[0] ></code>

Indikator	Beskrivning	Skript
	5. dagens stängning skall ligga i övre halvan (50%) av gårdagens kropp (svansar borträknade) 6. dagens kropp skall omfatta minst 60% av hela stapeln 7. kropparna skall båda dessa dagar vara STORA i relation till föregående dagar	<pre> (Close[-1] + 0.5*(Open[-1]- Close[-1])) && ((Close[0]- Open[0]) > 0.6*(High[0]- Low[0])) && (BarR().lowest(2) > BarR_MA()[0])) plot1[0] = 1 else plot1[0] = 0; </pre>
Volatilitet	Visa värdet 1 om stängningskurs ovanför Bollinger Band, visa värdet -1 om stängningskurs nedanför Bollinger Band, annars värdet 0.	<pre> if (Close[0] > BOL_Hi()[0]) plot1[0] = 1 else if (Close[0] < BOL_Lo()[0]) plot1[0] = -1 else plot1[0] = 0; </pre>
Rapportdatum	Visa värdet 1 om rapportdatum och senaste(stängning) kurs är högre jämfört gårdagens stängningskurs annars, värdet 0.	<pre> if (Fa.isReportDay() && Close.isUp()) plot1[0] = 1; else plot1[0] = 0; </pre>
Användning av variabler		
Öppning med Gap, med lokal variabel	Visar skillnad i procent(%) mellan stängningskurs och öppningskurs (period dag)	<pre> local Gap = ((Open[0]-Close[-1])/Close[-1]) *100.0; plot1[0] = Gap; </pre>
Öppning med Gap, med global variabel	Visar skillnad i procent(%) mellan stängningskurs och öppningskurs (period dag)	<pre> Gap[0] = ((Open[0]-Close[-1])/Close[-1]) *100.0; plot1[0] = Gap[0]; </pre> Lägg till Gap som global variabel, under inställningar -> lägg till global variabel. I

Indikator	Beskrivning	Skript
		rullistan välj att det är en variabel av typen "Data series". Ange namn på variabeln till "Gap".

18.9 Kodexempel: RSI m fl

Exempel på denna sida:

RSI (14) StoS (14, 3, 3) MultiMA
 BBW (20, 2.0) MACD (26, 12, 9)

RSI - Relative Strength Index (HkScript)

```
#global param("p1", 14)
#global param("p2", 30)
#global param("p3", 70)

#global nbrplots(3)

#global dataserie("upMove");
#global dataserie("downMove");
#global dataserie("avgUp");
#global dataserie("avgDown");

local n = p1;
local delta = 0.0;
local RsiCalc = 0.0;

// 1) Prisförändring (delta)
if (BarNumber == 0)
{
    delta = 0;
}
else
{
    delta = Close[0] - Close[-1];
}

// 2) Dela upp i upp/ned-rörelse
if (delta > 0)
{
    upMove[0] = delta;
    downMove[0] = 0;
}
else
{

```

```

    upMove[0] = 0;
    downMove[0] = fabs(delta); // fabs rekommenderas för
decimaltal
}

// 3) Wilder smoothing av avgUp/avgDown
// Init: första gången vi har minst n+1 bars (BarNumber
== n) sätter vi startvärde som SMA
if (BarNumber == n)
{
    avgUp[0] = upMove.sum(n) / n;
    avgDown[0] = downMove.sum(n) / n;
}
else if (BarNumber > n)
{
    avgUp[0] = avgUp[-1] + (1.0 / n) * (upMove[0] -
avgUp[-1]);
    avgDown[0] = avgDown[-1] + (1.0 / n) * (downMove[0] -
avgDown[-1]);
}
else
{
    // För tidiga bars: sätt 0 (eller lämna tomt beroende
på hur du vill att grafen ska se ut)
    avgUp[0] = 0;
    avgDown[0] = 0;
}

// 4) RSI = 100 - (100 / (1 + RS)), där RS =
avgUp/avgDown
if (BarNumber <= n)
{
    RsiCalc = 0; // ännu ej "riktig" RSI
}
else if (avgDown[0] == 0)
{
    RsiCalc = 100;
}
else
{
    local rs = avgUp[0] / avgDown[0];
    RsiCalc = 100 - (100 / (1 + rs));
}

// 5) Rita ut
plot1[0] = RsiCalc;
plot2[0] = p2; // undre nivå, t ex 30
plot3[0] = p3; // övre nivå, t ex 70

```

StoS - Stochastic Slow (HkScript)

```
#global param("p1", 14) // %K-linje, standard 14
#global param("p2", 3) // %D-linje, standard 3
#global param("p3", 3) // Slowfaktor, standard 3

#global nbrplots(2)

#global dataserie("fastK");
#global dataserie("slowK");
#global dataserie("slowD");

local kPeriod = p1;
local dPeriod = p2;
local slowFactor = p3;

local highestHigh = 0;
local lowestLow = 0;
local range = 0;

// Fast %K:
// 100 * (Close - lowest low) / (highest high - lowest low)
if (BarNumber >= kPeriod - 1)
{
    highestHigh = High.highest(kPeriod);
    lowestLow = Low.lowest(kPeriod);
    range = highestHigh - lowestLow;

    if (range != 0)
    {
        fastK[0] = 100 * (Close[0] - lowestLow) / range;
    }
    else
    {
        fastK[0] = 0;
    }
}
else
{
    fastK[0] = 0;
}

// Slow %K = utjämnad Fast %K.
// Detta motsvarar %D-linjen i Stochastic Fast.
if (BarNumber >= kPeriod + dPeriod - 2)
{
    slowK[0] = fastK.sum(dPeriod) / dPeriod;
}
else
{
    slowK[0] = 0;
}

// Slow %D = ytterligare utjämning av Slow %K med
```

```
slowfaktor.  
if (BarNumber >= kPeriod + dPeriod + slowFactor - 3)  
{  
    slowD[0] = slowK.sum(slowFactor) / slowFactor;  
}  
else  
{  
    slowD[0] = 0;  
}  
  
// Rita ut  
plot1[0] = slowK[0]; // StoS / %K  
plot2[0] = slowD[0]; // StoS_D / %D
```

MultiMA (HkScript)

```
// Indikatorn visas i huvudfönstret!  
#global setmainpanel(true);  
  
// 5 olika MA  
#global param("p1", 20) // ma1  
#global param("p2", 40) // ma3  
#global param("p3", 60) // ma3  
#global param("p4", 80) // ma4  
#global param("p5", 100) // ma5  
  
#global nbrplots(5)  
  
// Rita ut - sätt färger i inställningarna  
plot1[0] = MA1(p1)[0];  
plot2[0] = MA1(p2)[0];  
plot3[0] = MA1(p3)[0];  
plot4[0] = MA1(p4)[0];  
plot5[0] = MA1(p5)[0];
```

MACD (HkScript)

```
#global param("p1", 26) // Slow EMA  
#global param("p2", 12) // Fast EMA  
#global param("p3", 9) // Smooth / signallinje  
  
#global nbrplots(3)  
  
#global dataserie("fastEma");  
#global dataserie("slowEma");  
#global dataserie("avgLine");  
  
local slowPeriod = p1;  
local fastPeriod = p2;  
local smoothPeriod = p3;  
  
local fastAlpha = 2.0 / (1 + fastPeriod);
```

```

local slowAlpha = 2.0 / (1 + slowPeriod);
local smoothAlpha = 2.0 / (1 + smoothPeriod);
local macdLine = 0.0;
local diffLine = 0.0;

if (BarNumber == 0)
{
    fastEma[0] = Close[0];
    slowEma[0] = Close[0];
    avgLine[0] = 0;
}
else
{
    fastEma[0] = fastAlpha * Close[0] + (1 - fastAlpha) *
fastEma[-1];
    slowEma[0] = slowAlpha * Close[0] + (1 - slowAlpha) *
slowEma[-1];
    macdLine = fastEma[0] - slowEma[0];
    avgLine[0] = smoothAlpha * macdLine + (1 -
smoothAlpha) * avgLine[-1];
    diffLine = macdLine - avgLine[0];
}

// Rita ut - sätt plot3 visa staplar i inställningarna
plot1[0] = macdLine;    // MACD / Macd
plot2[0] = avgLine[0]; // Avg / signallinje
plot3[0] = diffLine;   // Diff / histogram

```

BBW - Bollinger Band Width (HkScript)

```

#global param("p1", 20) // Period
#global param("p2", 2.0) // Bredd på band
#global param("p3", 5) // MA-period

#global nbrplots(2)

#global dataserie("closeSq");
#global dataserie("bandWidth");

local period = p1;
local bandWidthFactor = p2;
local maPeriod = p3;

local mean = 0.0;
local variance = 0.0;
local middleBand = 0.0;
local stdDevLine = 0.0;
local upperBand = 0.0;
local lowerBand = 0.0;
local bandWidthMa = 0.0;

// Kvadrerad Close används för standardavvikelse

```

```
closeSq[0] = Close[0] * Close[0];

// Beräkna Bollinger-band och BBW
if (BarNumber >= period - 1)
{
    mean = Close.sum(period) / period;
    middleBand = mean;

    variance = (closeSq.sum(period) / period) - (mean *
mean);
    if (variance < 0)
    {
        variance = 0;
    }
    stdDevLine = sqrt(variance);

    upperBand = middleBand + bandWidthFactor * stdDevLine;
    lowerBand = middleBand - bandWidthFactor * stdDevLine;

    if (middleBand != 0)
    {
        bandWidth[0] = 100 * (upperBand - lowerBand) /
middleBand;
    }
    else
    {
        bandWidth[0] = 0;
    }

    if (BarNumber >= period + maPeriod - 2)
    {
        bandWidthMa = bandWidth.sum(maPeriod) / maPeriod;
    }
    else
    {
        bandWidthMa = 0;
    }
}
else
{
    bandWidth[0] = 0;
    bandWidthMa = 0;
}

// Rita ut
plot1[0] = bandWidth[0]; // BBW
plot2[0] = bandWidthMa; // BBW_MA
```

18.10 Om referenslistan

Se referensbibliotek för respektive kategori i vänstermarginalen: Språkbas, Deklarationer, TA, , FA, och Strategiscript.
Funktioner och defintioner anges i bokstavsordning under respektive kategori.

Kategorier

Referenslista - Språkbas

- Språk och Operatorer: t ex "&&" (och), "||" (eller)
- Generella funktioner: t ex min(), max(), avg(), RGB-färg, if-satser m fl

Referenslista - Deklarationer

- Lokala och globala variabler.
- Plot- och parameterinställningar.

Referenslista - TA

- Dataserie-funktioner (de allra flesta i listan). T ex funktionen "findGT" på RSI hittar hur många gånger RSI varit större än t ex värdet "70" de senaste 50 dagarna (perioder).
Formatet blir <Indikatornamn>(<inparam>).Funktion(), t ex RSI(14).findGT(70, 50)
- Tid. T ex "Time.month()" som anger aktuell månad.
- Kursdata-serier: t ex Close, High, Low, Open. Sista värdet av en kurs anges Close[0]
Indikatorer visas inte här. Se avsnittet Teknisk analys - Indikatorbibliotek för alla indikatorer. T ex RSI(14)[0] anger värdet för RSI14;

Referenslista - FA

- Fa-dataserier: t ex nyckeltal som PE, Sales, EBT m fl, och funktioner på dessa t ex ".growth()", "yearAvg()" m fl. Sista värdet av ett nyckeltal anges ".get(0)", t ex Fa.PE().get(0).
- Bolagshändelser: t ex Fa._isReportDay() m fl.

Referenslista - Strategi

- Signal och dess funktioner: t ex Signal.SetPositive, Signal.Changed() m fl.
- Strategi och dess funktioner: t ex Strategy.EnterLong, Strategy.ExitAll(), Strategy.SetPriceStop() m fl.

18.11 Referenslista Språkbas

18.11.1 == (lika med)

Namn	==
Parametrar	

Beskrivning	Lika med. Testar ifall ett värde är lika med sant.
Returnerar	SANT eller FALSKT
Exempel	För att testa ifall värdet är lika med 100: <pre>if ((Close[0] == 100)) plot1[0] = 1 else plot1[0] = 0;</pre>

18.11.2 != (skiljer)

Namn	!=
Parametrar	
Beskrivning	Skiljer. Testar ifall ett värde är lika med falskt.
Returnerar	SANT eller FALSKT
Exempel	För att testa ifall värdet inte är lika med 100: <pre>if ((Close[0] != 100)) plot1[0] = 1 else plot1[0] = 0;</pre>

18.11.3 && (och)

Namn	&&
Parametrar	
Beskrivning	Används för att binda ihop två villkor.
Returnerar	SANT eller FALSKT
Exempel	För att testa ifall värdet är över 100 OCH under 150: <pre>if ((Close[0] > 100) && (Close[0] < 150)) plot1[0] = 1 else plot1[0] = 0;</pre> För att skapa indikator för Piercing pattern: <pre>if (//1. Gårdagens stängningskurs skall vara lägre än gårdagens öppning (Close[-1] < Open[-1]) && //2. dagens öppning är lägre än</pre>

```

dagens stängning
(Open[0] < Close[0]) &&

//3. dagens öppning är lägre än
gårdagens lägsta
(Open[0] < Low[-1]) &&

//4. dagens stängning är lägre än
gårdagens öppning
(Close[0] < Open[-1]) &&

//5. dagens stängning skall ligga i
övre halvan (50%) av gårdagens kropp
(svansar borträknade)
(Close[0] > (Close[-1] + 0.5*(Open[-1]-Close[-1]))) &&

//6. dagens kropp skall omfatta minst
60% av hela stapeln
((Close[0]-Open[0]) > 0.6*(High[0]-Low[0])) &&

//7. kropparna skall båda dessa dagar
vara STORA i relation till föregående
dagar
(BarR().lowest(2) > BarR_MA()[0])
)
    plot1[0] = 1
else
    plot1[0] = 0;

```

18.11.4 || (eller)

Namn	
Parametrar	
Beskrivning	Eller. Används vid test av flera villkor.
Returnerar	SANT eller FALSKT
Exempel	För att testa ifall värdet är lika 100 eller 150 : <pre> if ((Close[0] == 100) (Close[0] == 150)) plot1[0] = 1 else plot1[0] = 0; </pre>

18.11.5 avg (math)

Namn	avg(a, b, ...)
Parametrar	2 eller flera värden, dvs valfritt antal.
Beskrivning	Räknar fram medelvärdet. Samma som sma.
Returnerar	Medelvärdet av två eller flera värden
Exempel	<pre>plot1[0] = avg(1, 5, 10, 15); // Exempel snittkurs: local avg_day = avg(Low[0], High[0]);</pre>

18.11.6 fabs, abs (math)

Namn	fabs(<decimaltal>) eller abs(<heltal>)
Parametrar	Ett tal som är positivt eller negativt
Beskrivning	Räknar fram absolutvärdet (positiva värdet). OBS. Använd som regel "fabs()" pga vanligast att decimaltal används.
Returnerar	Absolutvärdet
Exempel	<pre>// Alltid positivt tal, för avståndet Close och Open i relation till dagens Range: plot1[0] = fabs(Close[0]-Open[0]) / (High[0]-Low[0]); // Exempel heltalvärden: local val1 = -1; plot1[0] = abs(val1); // ger värdet 1 local val2 = -3.35; plot1[0] = fabs(val2); // ger värdet 3</pre>

18.11.7 for, while etc

HkScript stöder flera typer av loopar. Loopar används för att upprepa kod flera gånger, till exempel för att summera värden, söka bakåt i tiden eller gå igenom en lista med värden.

Viktigt om prestanda:

Ett HkScript körs normalt redan en gång för varje bar i dataserien. En loop inne i scriptet innebär därför ofta en extra upprepning för varje bar. Vid långa perioder eller många indikatorer kan detta göra scriptet långsammare.

Använd därför gärna inbyggda dataserie-funktioner som `sum`, `highest` och `lowest` när det passar, i stället för att alltid skriva egna loopar.

For-loop

En *for-loop* används när du vet ungefär hur många gånger koden ska upprepas.

```
local sum = 0;
local period = 10;
for (local i = 0; i < period; i++)
{
    sum = sum + Close[-i];
}
plot1[0] = sum / period;
```

Tips: *exemplet ovan kan normalt skrivas effektivare med* `Close.sum(period) / period`.

While-loop

En *while-loop* körs så länge villkoret är sant. Var noga med att ändra värden i loopen så att den inte riskerar att bli oändlig.

```
local i = 0;
local antal = 0;
while (i < 20)
{
    if (Close[-i] > Close[-i - 1])
    {
        antal++;
    }
    i++;
}
plot1[0] = antal;
```

Do/while

En *do/while-loop* liknar **while**, men kodblocket körs minst en gång eftersom villkoret kontrolleras efter blocket.

```
local i = 0;
do
{
    i++;
}
while (i < 5);
```

Foreach

En *foreach-loop* används för att gå igenom innehållet i en array eller tabell. Den är mindre vanlig i normal indikatorberäkning, men kan vara användbar vid listor eller hjälptabeller.

```
local values = [1, 2, 3, 4];
local sum = 0;
foreach (v in values)
{
    sum = sum + v;
}
plot1[0] = sum;
```

Break och continue

En *break* avbryter loopen direkt, och *continue* hoppar över resten av aktuell iteration och fortsätter med nästa varv i loopen.

```
local foundBarsBack = -1;
for (local i = 0; i < 50; i++)
{
    if (Close[-i] == 0)
    {
        continue;
    }
    if (Close[-i] > Close[0] * 1.10)
    {
        foundBarsBack = i;
        break;
    }
}
plot1[0] = foundBarsBack;
```

När bör loopar undvikas?

Loopar är användbara, men bör användas med eftertanke. Om du till exempel vill beräkna ett glidande medelvärde, högsta värdet, lägsta värdet eller en summa över en period är det ofta bättre att använda inbyggda dataseriefunktioner.

Exempel:

Skriv hellre `Close.sum(20)` än en egen *for-loop* som summerar 20 historiska värden.

Sammanfattning

- Använd **for** när antalet varv är känt.
- Använd **while** när loopen styrs av ett villkor.
- Använd **do/while** när koden ska köras minst en gång.
- Använd **foreach** för arrayer eller tabeller.
- Använd **break** för att avbryta loopen och **continue** för att hoppa till nästa varv.

18.11.8 If-satser

Namn	if
Parametrar	

Beskrivning	En if-sats testar ifall ett villkor uppfylls, och därefter följer rader/beräkning ifall det är SANT och ev. också rader/beräkning ifall det är FALSKT. Strukturen är som följer: <pre> if ([villkor du vill kontrollera]) { [beräkning om SANT] } else { [beräkning om FALSKT] } ("else" behöver inte anges) </pre>
Returnerar	Returnerar inget
Exempel	Visa värdet 1 ifall gårdagens stängningskurs är större än dagens, annars om dagens stängningskurs är mindre än MA1, visa värdet -1, annars visa värdet 0. <pre> if (Close[-1] > Close[0]) plot1[0] = 1 else if (Close[0] < MA1()[0]) plot1[0] = -1; else plot1[0] = 0; </pre>

18.11.9 iff (funktion)

Namn	iff(<villkor>, a, b)
Parametrar	
Beskrivning	En speciellt typ av if-sats som testar ifall ett villkor uppfylls, och sen returnerar ett angivet värde ifall det är SANT och ett annat angivet värde om det är FALSKT. Strukturen är som följer: <pre> iff ([villkor att kontrollera], [svarsvärde/beräkning om SANT], [svarsvärde/beräkning om FALSKT]) </pre>
Returnerar	Returnerar ett av angivna värden beroende på om villkoret är sant eller falskt
Exempel	Visa värdet 10 ifall gårdagens stängningskurs är större än dagens, annars 5.

	<pre>plot1[0] = iff(Close[-1] > Close[0], 10, 5);</pre> Observera att om resultat ska vara true/false eller 1/0 kan man skriva: <pre>plot1[0] = (Close[-1] > Close[0]);</pre> Exemplet ovan kan även skrivas med en vanlig "if-sats", dvs: <pre>if (Close[-1] > Close[0]) plot1[0] = 10; else plot1[0] = 5;</pre>

18.11.10 Info.country()

Namn	Info.country()																						
Beskrivning	Se vilket land aktuellt instrument/aktie tillhör.																						
Parametrar	Inga.																						
Returnerar	Returnerar landskod (ISO-3166) med två tecken, t ex "SE" för Sverige.																						
Exempel	<pre>if (Info.country() == "DK") plot1[0] = 0;</pre>																						
Landskoder	<table> <tr><td>Sverige</td><td>"SE"</td></tr> <tr><td>Danmark</td><td>"DK"</td></tr> <tr><td>Finland</td><td>"FI"</td></tr> <tr><td>Frankrike</td><td>"FR"</td></tr> <tr><td>Kanada</td><td>"CA"</td></tr> <tr><td>Norge</td><td>"NO"</td></tr> <tr><td>Schweiz</td><td>"CH"</td></tr> <tr><td>Spanien</td><td>"ES"</td></tr> <tr><td>Storbrit.</td><td>"GB"</td></tr> <tr><td>Tyskland</td><td>"DE"</td></tr> <tr><td>USA</td><td>"US"</td></tr> </table>	Sverige	"SE"	Danmark	"DK"	Finland	"FI"	Frankrike	"FR"	Kanada	"CA"	Norge	"NO"	Schweiz	"CH"	Spanien	"ES"	Storbrit.	"GB"	Tyskland	"DE"	USA	"US"
Sverige	"SE"																						
Danmark	"DK"																						
Finland	"FI"																						
Frankrike	"FR"																						
Kanada	"CA"																						
Norge	"NO"																						
Schweiz	"CH"																						
Spanien	"ES"																						
Storbrit.	"GB"																						
Tyskland	"DE"																						
USA	"US"																						

18.11.11 Info.currency()

Namn	Info.currency()
Beskrivning	Se i vilken valuta aktuellt instrument/aktie är noterad.
Parametrar	Inga.
Returnerar	Returnerar valutakod (ISO 4217) med tre tecken, t ex "SEK" för Svensk krona.
Exempel	<pre>if (Info.currency() == "USD") plot1[0] = 0;</pre>

Valutakoder , exempel	Sverige "SEK"
	Danmark "DKK"
	Finland "EUR"
	Frankrike "EUR"
	Kanada "CAD"
	Norge "NOK"
	Schweiz "CHF"
	Spanien "EUR"
	Storbrit. "GBP"
	Tyskland "EUR"
	USA "USD"

18.11.12 Info.isin()

Namn	Info.isin()
Beskrivning	Se vilken isin-identitet aktuellt instrument/aktie har.
Parametrar	Inga.
Returnerar	Returnerar isin, t ex "SE0000108656" för Ericsson B, "OMXS30" för OMX S30 Index.
Exempel	<pre>if (Info.isin() == "SE0000108656") plot1[0] += 10;</pre>

18.11.13 Info.listname()

Namn	Info.listname()
Beskrivning	Se vilken lista aktuellt instrument/aktie tillhör.
Parametrar	Inga.
Returnerar	Returnerar listnamn, t ex "Large Cap", "Mid Cap" osv i Sverige.
Exempel	<pre>if (Info.listname() == "Small Cap") plot1[0] = 0;</pre>
Listnamn , exempel	Large Cap Mid Cap Small Cap

18.11.14 max (math)

Namn	max(a, b)
Parametrar	Två tal som ska jämföras: a och b
Beskrivning	Hitta högsta värdet.
Returnerar	Returnerar det högsta värdet.
Exempel	<pre>plot1[0] = max(Close[0], Close[-1]);</pre>

18.11.15 min (math)

Namn	min(a, b)
Parametrar	Två tal som ska jämföras: a och b
Beskrivning	Hitta lägsta värdet.
Returnerar	Returnerar det lägsta värdet.
Exempel	<code>plot1[0] = min(Close[0], Close[-1]);</code>

18.11.16 mod (math)

Namn	mod(a, b)
Parametrar	Två tal: a och b
Beskrivning	Modulus (mod) returnerar "resten" efter en heltals-division, så att t ex 5 / 2 blir 2 + resten 1. Dvs talet 5 kan delas 2 gånger och sen blir 1 kvar, dvs mod(5, 2) blir 1, mod(4, 2) blir 0 osv. I språket kan mod(a, b) även skrivas som a % b.
Returnerar	Returnerar resten efter en division.
Exempel	<code>plot1[0] = mod(5, 2);</code> <code>plot1[0] = 5 % 2;</code>

18.11.17 RGB (funktion)

Namn	RGB(red, green, blue)
Parametrar	Mängden färg mellan 0-255 1. Red: 0-255 2. Green: 0-255 3. Blue: 0-255
Beskrivning	Standardfunktion för att ange färg - istället för hexadecimala tal (t ex 0xff0000 för rött) Där talet 0 anger ingen färg för resp. komponent och där 255 anger maximal färg. Ex RGB(0,0,0) är svart. RGB(255,255,255) är vitt, RGB(255,0,0) är rött.
Exempel	<code>plot1[0] = -1;</code> <code>Bar.SetColor (RGB (255, 0, 0)); // starkt röd färg</code>

18.11.18 sqr (math)

Namn	sqr(a)
Parametrar	Ett tal.

Beskrivning	Roten ur - $\sqrt{}$. Hitta det tal som, multiplicerat med sig självt, blir det angivna talet.
Returnerar	Returnerar roten ur värdet.
Exempel	<code>plot1[0] = sqr(2.0);</code>

18.12 Referenslista Deklarationer

18.12.1 local variabel

Lokala variabler i HkScript

I HkScript deklareras en lokal variabel med nyckelordet `local` följt av variabelns namn.

Datatypen bestäms automatiskt (*type inference*) av det första värdet som variabeln tilldelas.

Exempel på typbestämning

```
local x = 10;
```

→ x blir en heltalsvariabel (integer)

```
local x = "ABB";
```

→ x blir en sträng (text)

```
local x = (RSI(14)[0] > 80);
```

→ x blir en boolean (true eller false, 1 eller 0)

Viktigt om numeriska typer

Två numeriska typer i HkScript:

- Heltal (integer) – t ex: 10
- Decimaltal (decimal) – t ex: 10.0

Observera att tekniska indikatorer (TA/kursdata) normalt använder decimaler. För att deklarera ett decimaltal måste värdet innehålla decimalpunkt:

```
local x = 10.0;
```

→ x blir en decimaltals-variabel

```
local x = RSI(14)[0];
```

→ x blir en decimaltals-variabel

local variable

Namn	local
Beskrivning	Deklarerar lokal variabel.
Exempel	<pre>local is_above80 = RSI(14)[0] > 80; plot1[0] = 0; if (is_above80) plot1[0] = 1;</pre>

18.12.2 global variabel

Globala variabler i HkScript

Globala variabler deklareras normalt via Inställningarna i HkScript -> Lägg till globala Variabler

Men de kan även deklareras direkt i scriptet (längst upp i koden). Detta underlättar vid: kopiering mellan script, ger ökad tydlighet, och förenklar distribution av färdiga script.

Exempel på typbestämning

```
#global number("lastRSI", 80);
```

→ lastRSI blir en global heltalsvariabel (integer)

```
#global dataserie("myRSI");
```

→ myRSI blir en dataserie, för att

- lagra beräknade värden
- skapa en ny indikatorserie
- använda i vidare beräkningar

```
#global instrument("OMX", "OMX Stockholm 30 Index");
```

→ OMX blir en dataserie som innehåller data från indexet "OMX Stockholm 30 Index"

Observera:

- Stavning på namn måste exakt stämma med Hitta kursvinnarens databas.
- Isin måste vara korrekt om sådan används.

Viktigt om globala variabler

- Globala variabler är tillgängliga i hela scriptet.
- Globala variabler behåller sitt värde enligt sin definition (beroende på typ).
- Deklaration i script bör placeras högst upp.
- Namn får inte kollidera med lokala variabler, med inbyggda indikatorer, eller med egna script-indikatorer.

global variable

Namn	global
Beskrivning	Deklarerar global variabel.
Exempel	<pre>#global dataserie("myRSI") myRSI[0] = 10; myRSI[-1] = 5; plot1[0] = myRSI.ema(5);</pre>

18.12.3 dataserie

Namn	#global dataserie
Beskrivning	Deklarerar global variabel "dataserie".

Parametrar	Inparameter är seriens namn. Hela serien initieras med värdet 0.
Exempel	<code>#global dataserie ("myRSI")</code>

18.12.4 instrument

Namn	<code>#global dataserie</code>
Beskrivning	Deklarerar global variabel "instrument".
Parameter1	Seriens namn, t ex "E".
Parameter2	En referens till en aktie eller index, t ex "Ericsson B". Det måste vara exakt samma namn som Hitta kursvinnarens databas använder.
Exempel	<code>#global instrument("E", "Ericsson B") local val = E.RSI(14)[0];</code>

18.12.5 number

Namn	<code>#global number</code>
Beskrivning	Deklarerar global variabel "number".
Parametrar	Måste ha två inparametrar: namn och startvärde.
Exempel	<code>#global number("x", 10.0)</code>

18.12.6 nbrplots

Namn	<code>#global nbrplots</code>
Beskrivning	Sätter antalet "plots" detta script använder. (om det inte görs via Inställningsfliken) Namnen sätts automatiskt till plot1, plot2, plot3...osv Se även alternativ funktion "#global plot".
Parameter	Antal (heltal mellan 1 och 10)
Exempel	<code>#global nbrplots(2) plot1[0] = MA1(50)[0]; plot2[0] = MA2(100)[0];</code>

18.12.7 plot

Namn	<code>#global plot</code>
-------------	---------------------------

Beskrivning	Ett alternativ till "#global nbrplots" för att sätta detaljerad info per plot. Deklarerar en "plot", t ex "plot1", "plot2" (om det inte görs via Inställningsfliken) Inklusive färg och linjetjocklek. Tillåtna namn endast från "plot1" till "plot9".
Parametrar	1) Plotnamn, t ex "plot1" 2) Färg, t ex hex: "0x00ff00", eller rgb: RGB(0,255,0) 3) Linjetjocklek, ett tal mellan 1-5, t ex 1
Exempel	<pre>#global plot("plot1", RGB(0,200,0), 2); // grön #global plot("plot2", RGB(0,0,200), 2); // blå plot1[0] = MA1(5)[0]; plot2[0] = MA1(10)[0];</pre>

18.12.8 param

Namn	#global param
Beskrivning	Deklarerar en parameter, t ex "p1", "p2" (om det inte görs via Inställningsfliken) Tillåtna namn endast från "p1" till "p6".
Parametrar	1) Parameternamn, t ex "p1" 2) Startvärde, ett heltal eller ett decimaltal Obs! Parameterns datatyp sätts av startvärdet. T ex startvärdet "1" gör att parametern i scriptet alltid räknas med heltalsberäkning (t ex 1/10 = 0), medan "1.0" ger decimaltalsberäkning (t ex 1/10 = 0.1).
Exempel	<pre>#global param("p1", 2.0); // decimaltal x #global param("p2", 3.5); // decimaltal y plot1[0] = BOL(20, p1)[0]; plot2[0] = BOL(20, p2)[0];</pre>

18.12.9 setmainpanel

Namn	#global setmainpanel
Beskrivning	Anger om utdata för scriptet ska visas i huvudfönstret eller inte. (om inte så visas utdata i en egen indikatorpanel)
Parameter	true eller false (1 eller 0)
Exempel	<pre>#global setmainpanel(true);</pre>

```
plot1[0] = MA1(20)[0];
```

18.13 Referenslista TA

18.13.1 avg

Namn	avg(period)
Parametrar	period: Ett heltalsvärde som avser hur lång period beräkningen ska baseras på. Historiska sma: Spara beräkning i en global variabel av typ dataserie, eller använd plot1 om sådan beräknats enligt exemplet nedan. Välj valfri period bakåt, t ex -10.
Beskrivning	Räknar fram medelvärdet. Samma som sma
Returnerar	Medelvärdet av en dataserie för en period tillbaka
Exempel	<pre>plot1[0] = Close.avg(20);</pre> // Exempel gamla avg: <pre>local avg_10days_back = plot1[-10];</pre> Under inställningar, välj visa indikatorn i huvudfönstret.

18.13.2 ATH

Namn	ATH
Beskrivning	Ger nivån på All Time High. (Obs! ex Close resp High har olika ATH)
Returnerar	Returnerar värdet på All Time High (t ex för Close, High, RSI osv)
Exempel	<pre>// När kurs är mindre än 10% från All Time High if (Close[0] > 0.9*Close.ATH()) plot1[0] = 1; else plot1[0] = 0;</pre>

18.13.3 ATL

Namn	ATL
Beskrivning	Ger nivån på All Time Low. (Obs! ex Close resp Low har olika ATL)
Returnerar	Returnerar värdet på All Time Low (t ex för Close, Low, RSI osv)
Exempel	<pre>// När kurs är mindre än 10% från All Time Low</pre>

	<pre> if (Close[0] < 1.1*Close.ATL()) plot1[0] = 1; else plot1[0] = 0; </pre>
--	--

18.13.4 BarNumber

Namn	BarNumber eller "Info.BarNumber"
Parametrar	Inga
Beskrivning	HkScript körs en gång för varje tidpunkt i diagrammet (varje "Bar"). Värdet BarNumber anger vilket nummer i ordningen den aktuella iterationen har. Med start på 0 fram till diagrammets sista tidpunkt. T ex sista är 1000 om diagrammet har 1001 st tidpunkter (eller dagar) på x-axeln (LastBarNumber). Använd exempelvis ifall inget ska/kan ritas ut de 100 första perioderna. (se exempel nedan)
Returnerar	Returnerar det Bar-nummer som scriptet nu körs på.
Exempel	<pre> if (BarNumber < 100) plot1[0] = NoValue(); // skippa utritning första 100 bar else plot1[0] = MA2(100)[0] - MA1(50)[0]; </pre>

18.13.5 Bars.DayClose

Namn	Bars.DayClose(bars back)
Parametrar	Välj dag valfri period bakåt.
Beskrivning	Enbart för Intradaychart. Ger vald dags slutkurs, t ex föregående dags slutkurs.
Returnerar	Returnerar slutkurs.
Exempel	<pre> plot1[0] = Bar.DayClose(-1); // gårdagens slutkurs </pre>

18.13.6 Bars.DayHigh

Namn	Bars.DayHigh(bars back)
Parametrar	Välj dag valfri period bakåt.
Beskrivning	Enbart för Intradaychart. Ger vald dags högstakurs, t ex hittills högsta idag.
Returnerar	Returnerar högstakurs.

Exempel	<pre>plot1[0] = Bar.DayHigh(0); // dagens högsta just nu</pre>
----------------	--

18.13.7 Bars.DayLow

Namn	Bars.DayLow(bars back)
Parametrar	Välj dag valfri period bakåt.
Beskrivning	Enbart för Intradaychart. Ger vald dags lägstakurs, t ex hittills lägsta idag.
Returnerar	Returnerar lägstakurs.
Exempel	<pre>plot1[0] = Bar.DayLow(0); // dagens lägsta just nu</pre>

18.13.8 Bars.DayOpen

Namn	Bars.DayOpen(bars back)
Parametrar	Välj dag valfri period bakåt.
Beskrivning	Enbart för Intradaychart. Ger vald dags öppningskurs, t ex öppningskursen idag.
Returnerar	Returnerar öppningskurs.
Exempel	<pre>plot1[0] = Bar.DayOpen(0); // dagens öppningskurs</pre>

18.13.9 Bars.SetColor

Namn	Bars.SetColor(hexcolor)
Parametrar	Färg som hexadecimalt tal, t ex svart = 0x0, röd = 0xff0000, grön = 0x00ff00
Beskrivning	Sätt färg på aktuell stapel, så som för Candlestick, Barchart (jämför Signalfärg)
Returnerar	Inget.
Exempel	<pre>Bar.SetColor(0x00ff00); // grön färg</pre>

18.13.10 Close[]

Namn	Close
Beskrivning	Detta är en dataserie över stängningskurser. Close, liksom High, Low, Open ska skrivas med stor begynnelsebokstav.
Exempel	<pre>plot1[0] = Close[0];</pre>

Namn	Close
	Under inställningar, välj visa indikatorn i huvudfönstret.

18.13.11 countGT

Namn	countGT(val, period, startfrom)
Parametrar	Val: Värde Period: Ett heltalsvärde som avser hur lång period beräkningen ska baseras på Startfrom: Välj startpunkt valfri period bakåt.
Beskrivning	Står för Count Greater Than Räknar hur många gånger ett värde större än "val" förekommer i den senaste perioden i en dataserie
Returnerar	Hur många gånger ett större värde förekommer
Exempel	<pre>plot1[0] = Close.countGT(800, 100, 0);</pre>

18.13.12 countLT

Namn	countLT(val, period, startfrom)
Parametrar	Val: Värde Period: Ett heltalsvärde som avser hur lång period beräkningen ska baseras på Startfrom: Välj startpunkt valfri period bakåt.
Beskrivning	Står för Count Less Than Räknar hur många gånger ett värde mindre än "val" förekommer i den senaste perioden i en dataserie.
Returnerar	Hur många gånger ett mindre värde förekommer
Exempel	<pre>plot1[0] = Close.countLT(800, 200, 0);</pre>

18.13.13 crossAbove

Namn	crossAbove(series)
Beskrivning	Ser om dataserien korsar upp ovan annan dataserie
Parametrar	series: en dataserie att jämföra med
Returnerar	returnerar ett boolskt värde (true eller false) om serierna korsas som i beskrivningen
Exempel	<pre>if (MA1(3).crossAbove(MA1(5))) { plot1[0] = 10; } else</pre>

	<pre>{ plot1[0] = 0; }</pre>
--	----------------------------------

18.13.14 crossBelow

Namn	crossBelow(series)
Parametrar	series: en dataserie att jämföra med
Beskrivning	Ser om dataserien korsat ned under annan dataserie
Returnerar	Returnerar ett boolskt värde (true eller false) om serierna korsas som i beskrivningen
Exempel	<pre>if (MA1(3).crossBelow(MA1(5))) { plot1[0] = 10; } else { plot1[0] = 0; }</pre>

18.13.15 ema

Namn	ema(period)
Parametrar	period: Ett heltalsvärde som avser hur lång period beräkningen ska baseras på Historiska ema: Spara beräkning i en global variabel av typ dataserie, eller använd plot1 om sådan beräknats enligt exemplet nedan. Välj valfri period bakåt, t ex -10.
Beskrivning	Står för Exponential Moving Average Beräknar det exponentiella medelvärdet för en period
Returnerar	Det exponentiella medelvärdet för en period
Exempel	<pre>plot1[0] = Close.ema(20); // Exempel historiska ema: local ema_10days_back = plot1[-10];</pre> Under inställningar, välj visa indikatorn i huvudfönstret.

18.13.16 findGT

Namn	findGT(val, period)
Parametrar	Val: värdet att leta upp Period: perioden att leta i, defaultvärdet är 250
Beskrivning	Står för: Find Greater Than

	Ser hur många gånger ett värde större än "val" förekommer inom valfri "period" bakåt.
Returnerar	Returnerar ett numeriskt värde.
Exempel	<code>plot1[0] = Close.findGT(200, 20);</code>

18.13.17 findLT

Namn	findLT(val, period)
Parametrar	Val: värdet att leta upp Period: perioden att leta i, defaultvärdet är 250
Beskrivning	Står för: Find Less Than Ser hur många gånger ett värde mindre än "val" förekommer inom valfri "period" bakåt.
Returnerar	Returnerar ett numeriskt värde.
Exempel	<code>plot1[0] = Close.findLT(200, 20);</code>

18.13.18 High[]

Namn	High
Beskrivning	Detta är en dataserie över högstakurser
Exempel	<code>plot1[0] = High[0];</code> Under inställningar, välj visa indikatorn i huvudfönstret.

18.13.19 highest

Namn	highest(period)
Parametrar	period: ett heltal som anger hur lång period som avses.
Beskrivning	Anger det högsta värdet under perioden.
Returnerar	Returnerar det högsta värdet under perioden
Exempel	<code>plot1[0] = Close.highest(50);</code>

18.13.20 isDown

Namn	isDown(startfrom)
Parametrar	startfrom: Välj startpunkt valfri period bakåt.
Beskrivning	Anger om en dataserie är sjunkande
Returnerar	true eller false (1 eller 0) beroende på om data serien sjunker eller ej
Exempel	<code>plot1[0] = Close.isDown(10);</code>

18.13.21 isNewHigh

Namn	isNewHigh(start from bars back)
Parametrar	startfrom: Välj startpunkt valfri period bakåt.
Beskrivning	Anger om en dataserie anger senaste värdet som högsta
Returnerar	true eller false (1 eller 0) beroende på om senaste värdet är högsta
Exempel	<code>plot1[0] = Close.isNewHigh(10);</code>

18.13.22 isNewLow

Namn	isNewLow(start from bars back)
Parametrar	startfrom: Välj startpunkt valfri period bakåt.
Beskrivning	Anger om en dataserie anger senaste värdet som lägsta
Returnerar	true eller false (1 eller 0) beroende på om senaste värdet är lägsta
Exempel	<code>plot1[0] = Close.isNewLow(10);</code>

18.13.23 isUp

Namn	isUp(startfrom)
Parametrar	startfrom: Välj startpunkt valfri period bakåt.
Beskrivning	Anger om en dataserie är stigande
Returnerar	true eller false (1 eller 0) beroende på om data serien stiger eller ej
Exempel	<code>plot1[0] = Close.isUp(10);</code>

18.13.24 LastBarNumber

Namn	LastBarNumber eller "Info.LastBarNumber"
Parametrar	Inga
Beskrivning	HkScript körs en gång för varje tidpunkt i diagrammet (varje "Bar") där "BarNumber" räknas upp från start 0, till och med "LastBarNumber", som anger scriptets sista iteration. Dvs värdet LastBarNumber anger diagrammets sista nummer i ordningen. Värdet är t ex 1000 om diagrammet har 1001 st tidpunkter (eller dagar) på x-axeln. . Använd exempelvis ifall något ska göras på sista tidpunkt eller strax innan.
Returnerar	Returnerar diagrammets sista Bar-nummer.

Exempel	<pre>local final_value = 10; if (BarNumber == LastBarNumber) plot1[0] = plot1[-1] + final_value; // avslut</pre>
----------------	--

18.13.25 Low[]

Namn	Low
Beskrivning	Detta är en dataserie över lägstakurser
Exempel	<pre>plot1[0] = Low[0];</pre> Under inställningar, välj visa indikatorn i huvudfönstret.

18.13.26 lowest

Namn	lowest(period)
Parametrar	period: ett heltal som anger hur lång period som avses.
Beskrivning	Anger det lägsta värdet under perioden.
Returnerar	Returnerar det lägsta värdet under perioden
Exempel	<pre>plot1[0] = Close.lowest(50);</pre>

18.13.27 nbrOfBarsFromATH

Namn	nbrOfBarsFromATH
Beskrivning	Räknar antal perioder från All Time High
Returnerar	Returnerar ett numeriskt värde beroende på hur många perioder som passerats sedan högsta kurs.
Exempel	<pre>plot1[0] = Close.nbrOfBarsFromATH();</pre>

18.13.28 nbrOfBarsFromATL

Namn	nbrOfBarsFromATL
Beskrivning	Räknar antal perioder från All Time Low
Returnerar	Returnerar ett numeriskt värde beroende på hur många perioder som passerats sedan lägsta kurs.
Exempel	<pre>plot1[0] = Close.nbrOfBarsFromATL();</pre>

18.13.29 nbrOfUpBars

Namn	nbrOfUpBars()
Beskrivning	Räknar antal stigande perioder i rad

Returnerar	Returnerar ett numeriskt värde beroende på hur många perioder i raden som stängningskurs varit över öppningskurs
Exempel	<code>plot1[0] = Close.nbrOfUpBars();</code>

18.13.30 nbrOfDownBars

Namn	nbrOfDownBars()
Beskrivning	Räknar antal sjunkande perioder i rad
Returnerar	Returnerar ett numeriskt värde beroende på hur många perioder i raden som stängningskurs varit under öppningskurs
Exempel	<code>plot1[0] = Close.nbrOfDownBars();</code>

18.13.31 NoValue()

Namn	NoValue()
Parametrar	Inga
Beskrivning	Ett tomt värde. NoValue() kan användas för att skapa "gap" t ex i utritning av en indikator. T ex om inget ska ritas ut de 100 första perioderna. (se exempel nedan)
Returnerar	Returnerar ett tomt värde.
Exempel	<pre>if (BarNumber < 100) plot1[0] = NoValue(); // skippa utritning första 100 bar</pre>

18.13.32 Open[]

Namn	Open
Beskrivning	Detta är en dataserie över öppningskurser. Open, liksom Close, High och Low ska skrivas med stor begynnelsebokstav.
Exempel	<code>plot1[0] = Open[0];</code> Under inställningar, välj visa indikatorn i huvudfönstret.

18.13.33 sma

Namn	sma(period)
Parametrar	period: Ett heltalsvärde som avser hur lång period beräkningen ska baseras på.

	Historiska sma: Spara beräkning i en global variabel av typ dataserie, eller använd plot1 om sådan beräknats enligt exemplet nedan. Välj valfri period bakåt, t ex -10.
Beskrivning	Räknar fram medelvärde. Samma som avg
Returnerar	Medelvärde av en dataserie för en period tillbaka
Exempel	<pre>plot1[0] = Close.sma(20);</pre> // Exempel gamla sma: <pre>local sma_10days_back = plot1[-10];</pre> Under inställningar, välj visa indikatorn i huvudfönstret.

18.13.34 stdDev

Namn	stdDev(period, startfrom)
Parametrar	period: ett heltalsvärde som avser en period från dataseriens slut startfrom: Välj startpunkt valfri period bakåt.
Beskrivning	Står för: Standard Deviation Beräknar standardavvikelsen av datserien för en perioden "period"
Returnerar	Returnerar ett heltalsvärde som avser den just nu jällande standardavvikelsen för de senaste värdena i period "period"
Exempel	<pre>plot1[0] = Close.stdDev(10);</pre>

18.13.35 sum

Namn	sum(period, startfrom)
Parametrar	period: ett heltalsvärde för hur lång period från dataseriens slut som avses. startfrom: Välj startpunkt valfri period bakåt.
Beskrivning	Beräknar summan av de värden i dataserien som ingår i perioden.
Returnerar	Returnerar ett numeriskt värde som är summan av dataseriens värde inom perioden
Exempel	<pre>plot1[0] = Close.sum(10);</pre>

18.13.36 Time.date

Namn	date(barsBack=0)
Beskrivning	Aktuellt diagramdatum, med år(Y) *10 000, +månader(M) *100, +dagar(Day) - date är samma funktion som Time.YMD()

Parametrar	barsback = antal bar bakåt, t ex -1 för föregående bar
Returnerar	Returnerar ett numeriskt värde 0 - 20991231. T ex 1 nov 2025 blir värdet 20251101
Exempel	<code>plot1[0] = Time.date() == 20260102;</code>

18.13.37 Time.day

Namn	day()
Beskrivning	Dag
Returnerar	Returnerar ett numeriskt värde 1-31.
Exempel	<code>plot1[0] = Time.day();</code>

18.13.38 Time.dayOfWeek

Namn	dayOfWeek()
Beskrivning	Dag i veckan
Returnerar	Returnerar ett numeriskt värde 1-7.
Exempel	<code>plot1[0] = Time.dayOfWeek();</code>

18.13.39 Time.daysLeftInMonth

Namn	daysLeftInMonth()
Beskrivning	Antal dagar kvar i månaden.
Returnerar	Returnerar ett numeriskt värde 1-31.
Exempel	<code>plot1[0] = Time.daysLeftInMonth();</code>

18.13.40 Time.FrameIsDaily

Namn	FrameIsDaily()
Beskrivning	Används för att se diagrammets tidbas. (Dag)
Returnerar	Returnerar true eller false (1 eller 0) True för ett Dagschart.
Exempel	<code>if (Time.FrameIsDaily() && x == 2) Strategy.EntryLong();</code>

18.13.41 Time.FrameGetInterval

Namn	FrameGetInterval()
Beskrivning	Används för att se multipel(antal) för aktuell tidbas.

Returnerar	Returnerar ett tal som anger hur många av tidbasen som används i diagrammet. T ex för ett 5 min-diagram är det 5, för ett 60min-diagram är det 60.
Exempel	<pre> if (Time.FrameGetInterval() <=10 && Time.FrameIsMinute()) { // 10min-chart eller lägre if (x == 2) Strategy.EntryLong(); } </pre>

18.13.42 Time.FramelsMinute

Namn	FramelsMinute()
Beskrivning	Används för att se diagrammets tidbas. (Intradag / minuter)
Returnerar	Returnerar true eller false (1 eller 0) True för ett Intradaychart, t ex 5min, 60min osv.
Exempel	<pre> if (Time.FrameIsMinute() && Time.FrameGetInterval() <=10) { // 10min-chart eller lägre if (x == 2) Strategy.EntryLong(); } </pre>

18.13.43 Time.FramelsMonthly

Namn	FramelsMontly()
Beskrivning	Används för att se diagrammets tidbas. (Månad)
Returnerar	Returnerar true eller false (1 eller 0) True för ett Månadschart.
Exempel	<pre> if (Time.FrameIsMonthly() && x == 2) Strategy.EntryLong(); </pre>

18.13.44 Time.FramelsWeekly

Namn	FramelsWeekly()
Beskrivning	Används för att se diagrammets tidbas. (Vecka)
Returnerar	Returnerar true eller false (1 eller 0) True för ett Veckochart.
Exempel	<pre> if (Time.FrameIsWeekly() && x == 2) Strategy.EntryLong(); </pre>

18.13.45 Time.HM

Namn	HM(barsBack=0)
Beskrivning	Aktuell diagramtid, i timmar(Hour) *100, + minuter(Minute).
Parametrar	barsback = antal bar bakåt, t ex -1 för föregående bar
Returnerar	Returnerar ett numeriskt värde 0000 - 2359. T ex kl 9 blir värdet 900
Exempel	<code>plot1[0] = Time.HM() >= 1530;</code>

18.13.46 Time.hour

Namn	hour()
Beskrivning	Timme på dag
Returnerar	Returnerar ett numeriskt värde 1-24.
Exempel	<code>plot1[0] = Time.hour();</code>

18.13.47 Time.marketCloseHM

Namn	marketCloseHM()
Beskrivning	Stängningstid för marknaden där aktuella aktien handlas. Dvs diagramtiden i timmar(Hour) *100, + minuter(Minute).
Returnerar	Returnerar ett numeriskt värde 0000 - 2359. T ex kl 17:30 blir värdet 1730, kl 17 blir 1700
Exempel	<code>local do_not_buy = (Time.marketCloseHM() - Time.HM()) < 10;</code>

18.13.48 Time.marketOpenHM

Namn	marketOpenHM()
Beskrivning	Öppningstid för marknaden är aktuella aktien handlas. Dvs diagramtiden i timmar(Hour) *100, + minuter(Minute).
Returnerar	Returnerar ett numeriskt värde 0000 - 2359. T ex kl 9 blir värdet 900, kl 9:30 blir 930
Exempel	<code>local start_buy = (Time.HM() - Time.marketOpenHM()) > 30;</code>

18.13.49 Time.minute

Namn	minute()
Beskrivning	Minut i timme
Returnerar	Returnerar ett numeriskt värde 1-60.

Exempel	<code>plot1[0] = Time.minute();</code>
----------------	--

18.13.50 Time.month

Namn	month()
Beskrivning	Månad
Returnerar	Returnerar ett numeriskt värde 1-12.
Exempel	<code>plot1[0] = Time.month();</code>

18.13.51 Time.nbrBarsBeforeClose

Namn	nbrBarsBeforeClose()
Beskrivning	Antal bars innan stängningstid, för marknaden är aktuella aktien handlas. Dvs i ett 10min-diagram där börsen stänger kl 17:30, blir värdet = 3 vid kl 17. (3*10min)
Returnerar	Returnerar ett numeriskt värde 0 - ca 510 max. (1min-diagram).
Exempel	<code>plot1[0] = Time.nbrBarsBeforeClose() == 3;</code>

18.13.52 Time.nbrBarsSinceOpen

Namn	nbrBarsSinceOpen()
Beskrivning	Antal bars efter öppningstid, för marknaden är aktuella aktien handlas. Dvs i ett 10min-diagram där börsen öppnar kl 9, blir värdet = 3 vid kl 09:30. (3*10min)
Returnerar	Returnerar ett numeriskt värde 0 - ca 510 max. (1min-diagram).
Exempel	<code>plot1[0] = Time.nbrBarsSinceOpen() == 3;</code>

18.13.53 Time.passesDay

Namn	passesDay(n)
Beskrivning	Används för att markera särskild dag i månaden.
Returnerar	Returnerar true eller false (1 eller 0), endast sant en gång i månaden. I ett Dagschart så svarar funktionen True på angivna dagen eller första tillgängliga handelsdag efter datumet. I verklig handel skulle det inträffa på morgonen för aktuella dagen.
Exempel	<code>plot1[0] = Time.passesDay(25);</code>

18.13.54 Time.passesDayInYear

Namn	passesDayInYear(month, day)
Beskrivning	Används för att markera särskild dag på året. Exempelvis 25 juni anges inom parentes som Time.passesDayInYear(6, 25)
Returnerar	Returnerar true eller false (1 eller 0), endast sant en gång per år. I ett Dagschart så svarar funktionen True på angivna datumet eller första tillgängliga handelsdag efterdatumet. I verklig handel skulle det inträffa på morgonen för aktuella dagen.
Exempel	<pre>plot1[0] = Time.passesDayInYear(6, 25);</pre>

18.13.55 Time.passesHour

Namn	passesHour(hour)
Beskrivning	Används för att markera särskilt tidslag på dagen. Tidigast samma eller efter detta klockslag.
Returnerar	Returnerar true eller false (1 eller 0), endast sant en gång per dag. Obs. T ex i ett 60-minuterschart så svarar exempel-funktionen nedan True på historika stapeln som heter "kl 16", dvs den som visar tiden mellan kl 15:01 - 16:00. I verklig handel skulle det alltså inträffa kl 15:01.
Exempel	<pre>plot1[0] = Time.passesHour(15);</pre>

18.13.56 Time.passesHourMinute

Namn	passesHourMinute(hour, minute)
Beskrivning	Används för att markera särskilt klockslag på dagen. Tidigast samma eller efter detta klockslag. Exempelvis klockslaget 15:05 anges inom parentes som Time.passesHourMinute(15, 05)
Returnerar	Returnerar true eller false (1 eller 0), endast sant en gång per dag. Obs. T ex i ett 5-minuterschart så svarar exempel-funktionen nedan True på historika stapeln "15:10", dvs den som visar tiden mellan kl 15:05:01 - 15:10:00. I verklig handel skulle det alltså inträffa kl 15:05:01.
Exempel	<pre>plot1[0] = Time.passesHourMinute(15, 5);</pre>

18.13.57 Time.tradeDaysLeftInMonth

Namn	tradeDaysLeftInMonth()
Beskrivning	Antal dagar kvar i månaden, exklusive lördagar och söndagar. Tar ej hänsyn till övriga stängda handelsdagar.
Returnerar	Returnerar ett numeriskt värde 1-23.
Exempel	<pre>plot1[0] = Time.tradeDaysLeftInMonth()</pre>

18.13.58 Time.tradeDaysPassedInMonth

Namn	tradeDaysPassedInMonth()
Beskrivning	Antal passerade dagar i månaden, exklusive lördagar och söndagar. Tar ej hänsyn till övriga stängda handelsdagar.
Returnerar	Returnerar ett numeriskt värde 1-23.
Exempel	<pre>plot1[0] = Time.tradeDaysPassedInMonth()</pre>

18.13.59 Time.YMD

Namn	YMD(barsBack=0)
Beskrivning	Aktuellt diagramdatum, med år(Y) *10 000, +månader(M) *100, +dagar(Day) - YMD är samma funktion som Time.date()
Parametrar	barsback = antal bar bakåt, t ex -1 för föregående bar
Returnerar	Returnerar ett numeriskt värde 0 - 20991231. T ex 1 nov 2025 blir värdet 20251101
Exempel	<pre>plot1[0] = Time.YMD() == 20260102;</pre>

18.14 Referenslista FA

18.14.1 FA-Nyckeltal

Fundamental analys med HkScript i Hitta kursvinnare

I HkScript finns ca 40 st inbyggda fundamentala nyckeltal enligt listan nedan. Nyckeltal kan användas för jämförelser mot andra bolag eller för att beräkna nya värden: så som medelvärden, tillväxttal mm.
Se Hitta kursvinnares alla nyckeltal i listan Nyckeltal.

Utöver nyckeltal finns det generell info om resp bolags rapportering, t ex:

* **Fa._lastQuarter()** = senaste rapport-kvartal (tal mellan 1-4)

* **Fa._lastYear()** = senaste rapport-år (tal t ex 2025).

Se även avsnittet FA-analys i HkScript.

Fundamentala nyckeltal i HkScript	Beskrivning	Förklaring
Fa.BP	B/P (Book-to-price)	Eget kapital / Aktiekurs
Fa.Cash	Kassa per aktie	
Fa.CashRatio	Kassa jämfört tillgångar	Kassa /Totala Tillgångar (%)
Fa.DebtRatio	Skuldsättningsgrad	(Tot.Tillgångar - Eget kapital) / Eget kapital (%)
Fa.Dividend	Utdelning per aktie	
Fa.DividendPayoutRatio	Utdelningsandel	Andel av vinsten som delats ut (%)
Fa.DividendYears	Antal utdelningsår i rad	
Fa.DividendYield	Direktavkastning	Utdelning i % av aktiekurs
Fa.E	Vinst/aktie, efter skatt	
Fa.Ebit	Vinst/aktie, före räntor & skatt	Earnings Before Interests and Taxes
Fa.Ebt	Vinst/aktie före skatt	Earnings Before Taxes
Fa.EbtMargin	se "Fa.ProfitMargin"	Rörelseresultat(Ebt) / omsättning (%)
Fa.EbitMargin	Ebit-marginal	Rörelseresultat(Ebit) / omsättning (%)
Fa.EbitdaMargin	Ebitda-marginal	Rörelseresultat(Ebitda) / omsättning (%)
Fa.EpRatio	Omvända P/e-talet i %	Earnings per price (%)
Fa.Equity	Eget kapital per aktie	
Fa.EquityRatio	Soliditet	Eget kapital / Totala tillgångar (%)
Fa.EV	EnterpriseValue (Mkr)	Skuldfritt börsvärde
Fa.EvEbt	EV / Ebt	
Fa.EvEbit	EV / Ebit	
Fa.EvEbitda	EV / Ebitda	
Fa.EvOp	EV / Operativt kassaflöde	

Fundamentala nyckeltal i HkScript	Beskrivning	Förklaring
Fa.EvS	EV / Sales	
Fa.EY	Earnings Yield	EBIT/EV(%)
Fa.GrossMargin	Bruttoresultat (%)	Bruttovinst / Försäljning (%)
Fa.IntangAssetRatio	Intangible assets ratio (%)	Immater. tillg. / Tot. tillgångar (%)
Fa.LongLiabilities	Långfristiga skulder/aktie	
Fa.MarketValue	Börsvärde	Bolagets värde baserat på senaste kurs
Fa.NbrShares	Antal aktier	
Fa.NetDebt	Nettoskuld/aktie	
Fa.NetDebtRatio	Nettoskuld(%)	Nettoskuld / omsättning (%)
Fa.OpCash	Operativt kassaflöde (Tkr)	
Fa.OpCashMargin	Operativt kassaflöde- marginal	Op. kassaflöde / omsättning (%)
Fa.PB	P/B (Price-to-book)	Aktiekurs / Eget kapital
Fa.PBtang	P/B-tang	Aktiekurs / (Eget kapital - immateriella tillg.)
Fa.PE	P/e-talet	Aktiekurs / Vinst per aktie
Fa.ProfitMargin	Vinstmarginal (Ebt- margin)	Rörelseresultat(Ebt) / omsättning (%)
Fa.ProfitYears	Antal vinstår i rad (Ebt)	
Fa.PS	P/s-talet	Aktiekurs / Försäljning per aktie
Fa.RoaRatio	Return on assets Ratio	
Fa.RoagRatio	Return on tangible assets Ratio	
Fa.RoeRatio	Return on equity (Räntabilitet)	Räntabilitet (%)
Fa.Sales	Försäljning per aktie	
Fa.SalesToAssets	Omsättningshastigh et	Sales / Total assets
Fa.ShortLiabilities	Kortfristiga skulder/aktie	

Fundamentala nyckeltal i HkScript	Beskrivning	Förklaring
Fa.TotAssets	Totala tillgångar/aktie	
Fa.TotDebt	Totala skulder/aktie	

18.14.2 Fa-funktioner

Varje FA-nyckeltal i HkScript beskrivs av en dataserie med **samma längd som antalet rapporter tillgängliga**.

För ett diagram på 1 år är det normalt endast 4 st värden, dvs 4 rapporter - och **ett rapportvärde visar alltid 12 månader med rullande kvartal**.

Vid varje tillfälle kommer t ex `Fa.Sales().get(0)` visa senaste rapportens Omsättning per aktie (försäljning).

För att förenkla arbetet har HkScript ett antal inbyggda funktioner för den här typen av dataserier.

T ex kan "genomsnittliga års-försäljningstillväxten" under 5 år skrivas som:

`Fa.Sales().yearGrowthAvg(5)`

, eller antal år i rad som ett bolag haft utdelning, dvs "utdelning > 0" ("above 0"), skrivs som: `Fa.Dividend().nbrStreakAbove(0, "Y")`

För att ange ifall beräkning ska ske med perioden År eller Kvartal, så anges det "Y" resp "Q" - och om inget anges så sker beräkningen på År ("Y").

Lista funktioner på FA-dataserier

Namn	Beskrivning	Exempel
<code>.get(bars=0)</code>	Hämta rapport/nyckeltalets värde.	<code>Fa.Sales().get(0)</code>
<code>.growth(len=1, period="Y")</code>	Tillväxt i procent, ange längd och "Y" (Årsvis) eller "Q" (Kvartalsvis)	<code>Fa.Sales().growth(3)</code> = försäljningstillväxt 3 år.
<code>.growthPrev(len=1, period="Y")</code>	Föregående periods Tillväxt i procent. Dvs tillväxt föregående året, eller föreg. kvartalet.	<code>Fa.Sales().growth(1, "Q")</code> = försäljningstillväxt förra kvartalet.
<code>.nbrInRangeAbove(val = 0, len=10, period="Y")</code>	Antal värden som varit större än t ex 0, under en viss period. (t ex antal vinst>0 under 10 år)	<code>Fa.E().nbrInRangeAbove(0, 10)</code> = antal år med vinst under perioden 10 år.
<code>.nbrInRangeBelow(val = 0, len=10, period="Y")</code>	Antal värden som varit mindre än t ex 0, under en	<code>Fa.E().nbrInRangeBelow(0, 10)</code>

Namn	Beskrivning	Exempel
	viss period. (t ex antal vinst<0 under 10 år)	= antal år med förlust under perioden 10 år.
.nbrStreakAbove(val = 0, period= "Y")	Antal värden i en följd som varit större än t ex 0. (t ex 5 utdelningsår i rad)	<code>Fa.Dividend().nbrStreakAbove(0, 10)</code> = antal utdelningsår i en följd.
.nbrStreakBelow(val = 0, period= "Y")	Antal värden i en följd som varit mindre än t ex 0. (t ex antal förlustår i rad)	<code>Fa.E().nbrStreakBelow(0, 10)</code> = antal år med förlust i en följd.
.yearAvg(len=3)	Medelvärdet för angivet antal år.	<code>Fa.PE().yearAvg(5)</code> = genomsnittlig PE för 5 år.
.yearGrowthAvg(len=3)	Tillväxt-medelvärdet (%) för angivet antal år.	<code>Fa.E().yearGrowthAvg(5)</code> = genomsnittlig vinsttillväxt för 5 år.

18.14.3 Fa._isDividendXDay

Visar dag aktien handlas exklusive utdelning (X-dag).

Funktioner finns även för att visa hur många dagar (bars) det just nu är före resp efter en sådan utdelningsdag.

Utdelningsdag

Namn	Fa._isDividendXDay()
Beskrivning	Används för att hitta första dagen aktien handlas exklusive utdelning
Returnerar	Returnerar boolskt värde (true el. false, 1 el. 0)
Exempel	<pre>if (Fa._isDividendXDay()) plot1[0] = 1; else plot1[0] = 0;</pre>

Antalet bars före/efter en utdelning

Namn	Fa._nbrBarsBeforeDividend()
Beskrivning	Räkna hur långt är det kvar till Utdelningsdag.
Returnerar	Antalet bars.
Exempel	<pre>local bars_before = Fa._nbrBarsBeforeDividend();</pre>

Namn	Fa._nbrBarsAfterDividend()
------	----------------------------

Beskrivning	Räkna hur långt är det efter Utdelningsdag.
Returnerar	Antalet bars.
Exempel	<pre>local bars_after = Fa._nbrBarsAfterDividend();</pre>

18.14.4 Fa._isReportDay

Visar dag aktien har en bolagsrapport.
Funktioner finns även för att visa hur många dagar (bars) det just nu är före resp efter en sådan rapportdag.

Rapportdag

Namn	Fa._isReportDay()
Beskrivning	Används för att hitta rapportdatum.
Returnerar	Returnerar boolskt värde (true el. false, 1 el. 0), normalt endast sant en gång i kvartalet.
Exempel	<pre>if (Fa._isReportDay()) plot1[0] = 1; else plot1[0] = 0;</pre>

Antalet bars före/efter en rapport

Namn	Fa._nbrBarsBeforeReport()
Beskrivning	Räkna hur långt det är kvar till Rapportdag.
Returnerar	Antalet bars.
Exempel	<pre>local bars_before = Fa._nbrBarsBeforeReport();</pre>

Namn	Fa._nbrBarsAfterReport()
Beskrivning	Räkna hur långt det är efter Rapportdag.
Returnerar	Antalet bars.
Exempel	<pre>local bars_after = Fa._nbrBarsAfterReport();</pre>

18.14.5 Fa._isMeetingDay

Namn	Fa._isMeetingDay()
-------------	--------------------

Beskrivning	Används för att hitta stämmodatum.
Returnerar	Returnerar boolskt värde (true el. false, 1 el. 0), normalt endast sant en gång per år.
Exempel	<pre>if (Fa._isMeetingDay()[0]) plot1[0] = 1; else plot1[0] = 0;</pre>

18.14.6 Fa._lastQuarter

Namn	Fa._lastQuarter()
Beskrivning	Används för få fram senaste rapporterade kvartal. (ex Q1, Q2, Q3, Q4)
Returnerar	Numeriskt värde 1-4.
Exempel	<pre>if (Fa._lastQuarter() == 4) plot1[0] = 1;</pre>

18.14.7 Fa._lastYear

Namn	Fa._lastYear()
Beskrivning	Används för få fram senaste rapporterade år. (ex 2025)
Returnerar	Numeriskt värde, ett årtal.
Exempel	<pre>if (Fa._lastYear() == 2025) plot1[0] = 1;</pre>

18.14.8 Fa.Longname

Namn	Fa.Longname()
Beskrivning	Ger aktiens fulla namn
Returnerar	Returnerar namnet (t ex "Ericsson B")
Exempel	<pre>if (Fa.Longname() == "Ericsson B") plot1[0] = 1; else plot1[0] = 0;</pre>

18.14.9 Fa.Shortname

Namn	Fa.Shortname()
-------------	----------------

Beskrivning	Ger aktiens kortnamn
Returnerar	Returnerar korta namnet (t ex "AZN", inte AstraZeneca). Se Hitta kursvinnares FA-del.
Exempel	<pre>if (Fa.Shortname() == "AZN") plot1[0] = 1; else plot1[0] = 0;</pre>

18.15 Referenslista Strategi

18.15.1 BarsSinceEntry

Namn	Strategy.BarsSinceEntry()
Beskrivning	Kollar antalet bar (tidsenheter) sen Entry av aktuell position.
Returnerar	Antalet bars. (är 0 ifall Entry gjorts nu)
Exempel	<pre>if (Strategy.BarsSinceEntry() >= 5) Strategy.ExitAll();</pre>

18.15.2 ChangedSize

Namn	Strategy.ChangedSize()
Beskrivning	Kollar ifall innehavet (antal positioner) har förändrats. motsvarar: Strategy.PositionSize(0) != Strategy.PositionSize(-1)
Returnerar	true eller false. (1 eller 0)
Exempel	<pre>if (Strategy.ChangedSize() && Strategy.IsOpen()) Strategy.SetPriceStop(0.95 * Close[0]);</pre>

18.15.3 EnterLong

Namn	Strategy.EnterLong()
Beskrivning	Öppnar en Köp-position. (sätter antal positioner till 1) Sätter Strategy.PositionSize() till 1.
Returnerar	Inget.
Exempel	<pre>if (Signal.IsPositive()) Strategy.EnterLong();</pre>

18.15.4 EnterShort

Namn	Strategy.EnterShort()
-------------	-----------------------

Beskrivning	Öppnar en kort position, dvs blankning. (sätter antal positioner till -1) Sätter Strategy.PositionSize() till -1.
Returnerar	Inget.
Exempel	<pre>if (Signal.IsNegative()) Strategy.EnterShort();</pre>

18.15.5 ExitAll

Namn	Strategy.ExitAll()
Beskrivning	Avslutar innehavet (sätter antal positioner till 0) Sätter Strategy.PositionSize() till 0.
Returnerar	Inget.
Exempel	<pre>if (Signal.IsNegative()) Strategy.ExitAll();</pre>

18.15.6 GetPriceEntry

Namn	Strategy.GetPriceEntry()
Beskrivning	Hämtar kursnivå för senaste gjord Entry (via <i>EntryLong</i> eller <i>EntryShort</i>).
Returnerar	Entrykursen.
Exempel	<pre>local last_entry = Strategy.GetPriceEntry();</pre>

18.15.7 GetPriceStop

Namn	Strategy.GetPriceStop()
Beskrivning	Hämta aktuell stoploss-nivå (ifall sådan angetts). Se funktion <i>SetPriceStop()</i> .
Returnerar	Stoppkursen.
Exempel	<pre>local stop = Strategy.GetPriceStop();</pre>

18.15.8 GetPriceTarget

Namn	Strategy.GetPriceTarget()
Beskrivning	Hämta aktuell target-nivå (ifall sådan angetts). Se funktion <i>SetPriceTarget()</i> .
Returnerar	Målkursen.
Exempel	<pre>local target = Strategy.GetPriceTarget();</pre>

18.15.9 IsLong

Namn	Strategy.IsLong()
Beskrivning	Kollar ifall innehavet (antal positioner) är > 0. Dvs lång (Long). motsvarar: Strategy.PositionSize() > 0;
Returnerar	true eller false. (1 eller 0)
Exempel	<pre>if (Strategy.IsLong()) Strategy.ExitAll();</pre>

18.15.10 IsOpen

Namn	Strategy.IsOpen()
Beskrivning	Kollar ifall innehav finns (antal positioner är inte noll). motsvarar: Strategy.PositionSize() != 0;
Returnerar	true eller false. (1 eller 0)
Exempel	<pre>if (Strategy.IsOpen() && Signal.IsNegative()) Strategy.ExitAll();</pre>

18.15.11 IsShort

Namn	Strategy.IsShort()
Beskrivning	Kollar ifall innehavet (antal positioner) är < 0. Dvs kort (Short). motsvarar: Strategy.PositionSize() < 0;
Returnerar	true eller false. (1 eller 0)
Exempel	<pre>if (Strategy.IsShort()) Strategy.ExitAll();</pre>

18.15.12 IsZeroPos

Namn	Strategy.IsZeroPos()
Beskrivning	Kollar ifall innehav helt saknas (antal positioner är 0). Ibland kallad "IsFlat()". motsvarar: Strategy.PositionSize() == 0;
Returnerar	true eller false. (1 eller 0)
Exempel	<pre>if (Strategy.IsZeroPos() && Signal.IsPositive()) Strategy.EnterLong();</pre>

18.15.13 PositionSize

Namn	Strategy.PositionSize(bars back = 0)
Beskrivning	Aktuella innehavet (antalet positioner), eller historiskt innehav vid inparameter. Obs. Positivt antal vid lång (Long) position. Negativt antal vid kort (Short) position.
Parametrar	bars back: Välj valfri historisk period bakåt, där bakåt anges -1, -2 osv.
Returnerar	Antalet positioner. (är 0 ifall alla positioner är stängda)
Exempel	<pre>if (Strategy.PositionSize() > 0) Strategy.ExitAll();</pre>

18.15.14 RemovePriceStop

Namn	Strategy.RemovePriceStop()
Beskrivning	Tar bort eventuell inlagd stoploss-nivå.
Returnerar	Inget.
Exempel	<pre>Strategy.RemovePriceStop();</pre>

18.15.15 SetPriceStop

Namn	Strategy.SetPriceStop(price)
Beskrivning	Anger en stoploss-nivå för att lämna aktuell position (normalt för att begränsa förlust). Vid lång position då ett pris <u>under</u> Entry, vid kort position ett pris <u>ovan</u> Entry. När kursen (price) har uppnåtts stängs positionen, dvs ExitAll().
Parametrar	price: stoppkursen
Returnerar	Inget.
Exempel	<pre>if (Signal.IsPositive()) Strategy .SetPriceStop(Close[0]/1.10);</pre>

18.15.16 SetPriceTarget

Namn	Strategy.SetPriceTarget(price)
Beskrivning	Anger en målnivå för vinsttagning. Vid lång position ett pris <u>ovan</u> Entry, vid kort position ett pris <u>under</u> Entry. När kursen (price) har uppnåtts stängs positionen, dvs ExitAll().

Parametrar	price: målkursen
Returnerar	Inget.
Exempel	<pre>if (Signal.IsPositive()) Strategy.SetPriceTarget(Close[0] *1.10);</pre>

18.15.17 TightenPriceStop

Namn	Strategy.TightenPriceStop(price)
Beskrivning	Uppdaterar eller sätter en ny stoploss-nivå. Funktionen kontrollerar att ny nivå snävar in avståndet till aktuell kurs.(annars ändras ej) Se även Strategy.SetPriceStop(price).
Parametrar	price: stoppkursen
Returnerar	Inget.
Exempel	<pre>if (Close[0] > Close[-1]) { local stop = Strategy.GetPriceStop(); local add_stop = (Close[0] - Close[-1]) / 2.0; Strategy.TightenPriceStop(stop + add_stop); }</pre>

18.15.18 Signal.Changed

Namn	Signal.Changed()
Beskrivning	Kollar ifall en signal ändrats på något sätt, till/från "+", "-" eller neutral. En Signal kan vara "+", neutral eller "-", från +10 till -10. (positiv, neutral, negativ)
Returnerar	true eller false (1 eller 0)
Exempel	<code>plot1[0] = Signal.Changed();</code>

18.15.19 Signal.ChangedToNegative

Namn	Signal.ChangedToNegative()
Beskrivning	Kollar ifall en signal har ändrats till "-" (negativ) En Signal kan vara "+", neutral eller "-", från +10 till -10. (positiv, neutral, negativ)
Returnerar	true eller false (1 eller 0)

Exempel	<pre>plot1[0] = Signal.ChangedToNegative();</pre>
---------	---

18.15.20 Signal.ChangedToNeutral

Namn	Signal.ChangedToNeutral()
Beskrivning	Kollar ifall en signal har ändrats till "neutral" (värdet 0) En Signal kan vara "+", neutral eller "-", från +10 till -10. (positiv, neutral, negativ)
Returnerar	true eller false (1 eller 0)
Exempel	<pre>plot1[0] = Signal.ChangedToNeutral();</pre>

18.15.21 Signal.ChangedToPositive

Namn	Signal.ChangedToPositive()
Beskrivning	Kollar ifall en signal har ändrats till "+" (positiv) En Signal kan vara "+", neutral eller "-", från +10 till -10. (positiv, neutral, negativ)
Returnerar	true eller false (1 eller 0)
Exempel	<pre>plot1[0] = Signal.ChangedToPositive();</pre>

18.15.22 Signal.GetValue

Namn	Signal.GetValue(bars back = 0)
Beskrivning	Ger signalstyrkan, dvs ett värde från +10 till -10.
Parametrar	bars back: Välj valfri historisk period bakåt, där bakåt anges -1, -2 osv.
Returnerar	Ett värde från +10 till -10.
Exempel	<pre>// Köpa när signalstyrkan ökar if (Signal.GetValue() > Signal.GetValue(-1)) Strategy.EnterLong();</pre>

18.15.23 Signal.IsNegative

Namn	Signal.IsNegative()
Beskrivning	Kollar ifall en signal är "-" (negativ), dvs Signal.GetValue() är mindre än 0. En Signal kan vara "+", neutral eller "-". (positiv, neutral, negativ)
Returnerar	true eller false. (1 eller 0)

Exempel	<pre>if (Signal.IsNegative()) Strategy.ExitAll();</pre>
---------	---

18.15.24 Signal.IsNeutral

Namn	Signal.IsNeutral()
Beskrivning	Kollar ifall en signal är "neutral" (värdet 0), dvs Signal.GetValue() är lika med 0. En Signal kan vara "+", neutral eller "-". (positiv, neutral, negativ)
Returnerar	true eller false. (1 eller 0)
Exempel	<pre>if (Signal.IsNeutral()) Strategy.ExitAll();</pre>

18.15.25 Signal.IsPositive

Namn	Signal.IsPositive()
Beskrivning	Kollar ifall en signal är "+" (positiv), dvs Signal.GetValue() är större än 0. En Signal kan vara "+", neutral eller "-". (positiv, neutral, negativ)
Returnerar	true eller false. (1 eller 0)
Exempel	<pre>if (Signal.IsPositive()) Strategy.EnterLong();</pre>

18.15.26 Signal.SetNegative

Namn	Signal.SetNegative()
Beskrivning	* Använd när endast +1, 0 och -1 sätts. SetNegative() = Signal.SetValue(-1) . Dvs symbolen "-". En Signal kan vara positiv, neutral, negativ (+10 till -10). Den kan användas för att visuellt markera (grönt/rött) i diagram eller visa signal i en Urval/sök-lista, eller koppla larm till händelsen mm.
Returnerar	Inget.
Exempel	<pre>if (MA1(20)[0] < MA1(50)[0]) Signal.SetNegative();</pre>

18.15.27 Signal.SetNeutral

Namn	Signal.SetNeutral()
Beskrivning	* Använd när endast +1, 0 och -1 sätts. SetNeutral() = Signal.SetValue(0) . Dvs neutral. En Signal kan vara positiv, neutral, negativ (+10 till -10).

	Neutral kan användas för att visuellt markera (ofärgad/grå) i diagram, eller visa i en Urval/sök-lista, eller koppla larm till händelsen mm.
Returnerar	Inget.
Exempel	<pre>if (RSI(14)[0] > 50) Signal.SetNeutral();</pre>

18.15.28 Signal.SetPositive

Namn	Signal.SetPositive()
Beskrivning	* Använd när endast +1, 0 och -1 sätts. SetPositive() = Signal.SetValue(+1). Dvs symbolen "+". En Signal kan vara positiv, neutral, negativ (+10 till -10). Den kan användas för att visuellt markera (grönt/rött) i diagram eller visa signal i en Urval/sök-lista, eller koppla larm till händelsen mm.
Returnerar	Inget.
Exempel	<pre>if (MA1(20)[0] > MA1(50)[0]) Signal.SetPositive();</pre>

18.15.29 Signal.SetValue

Namn	Signal.SetValue(val)
Beskrivning	Sätter signalstyrkan till ett värde från +10 till -10 . Urval/sök-lista: Till skillnad från SetPositive("+") eller SetNegative("-") så visas en signalstyrka med värden på en skala från "+10" till "-10".
Returnerar	Inget.
Exempel	<pre>local val1 = iff(RSI(14)[0] < 70, 3, 0); // +1 local val2 = iff(Mom(20)[0] > 0, 1, 0); // +1 local val3 = iff(StoS(5).crossAbove(StoS_D(5)), 3, 0); // +3 // signalstyrka mellan 0-5: Signal.SetValue(val1 + val2 + val3);</pre>